ИНФОРМАТИКА

Н.УГРИНОВИЧ

Информатика и информационные технологии

10-11 классы

Допущено
Министерством образования Российской Федерации
в качестве учебника по информатике
для учащихся 10-11 классов
естественно-математического профиля
общеобразовательных учреждений

Москва
БИНОМ. Лаборатория знаний
2003

УДК 004.9
ББК 32.97
У27

Угринович Н. Д.

У27 Информатика и информационные технологии. Учебник для 10–11 классов / Н. Д. Угринович. — М.: БИНОМ. Лаборатория знаний, 2003. — 512 с.: ил.

ISBN 5-94774-016-8

Учебник предназначен для изучения курса «Информатика и информационные технологии» (ИИТ) в общеобразовательных учреждениях. Учебно-методический комплект (учебник, практикум, методическое руководство, компьютерный практикум на CD-ROM), в который входит данный учебник, полностью соответствует разработанному Министерством образования новому общеобразовательному стандарту по ИИТ и обеспечивает возможность изучения углубленного курса ИИТ в 10–11 классах по естественно-математическому, информационно-технологическому и общеобразовательному профилям. Особое внимание уделено изучению объектно-ориентированного программирования на языке Visual Basic, основ логики, систем счисления и коммуникационных технологий. Содержание учебника соответствует программе вступительных экзаменов по информатике в вузы и может быть использовано для подготовки к экзаменам.

УДК 004.9
ББК 32.97

По вопросам приобретения обращаться:
в Москве
(095) 955-03-98, e-mail: lbz@aha.ru
в Санкт-Петербурге
(812) 247-93-01, e-mail: dialect@sndlct.ioffe.rssi.ru

ISBN 5-94774-016-8

Содержание

Рекомендации по использованию пособия

1. В состав программно-методического комплекса по информатике и информационным технологиям входят:
   - **Информатика и информационные технологии: Учебник для 10–11 классов (входит в Федеральный перечень учебников);**
   - Практикум по информатике и информационным технологиям: учебное пособие для общеобразовательных учреждений;
   - Практикум по информатике и информационным технологиям, программная и методическая поддержка курса: электронный учебник на CD-ROM;
   - Преподавание курса «Информатика и информационные технологии». Методическое пособие для учителей.
2. Комплекс представляет собой единую образовательную среду, связанную между собой гиперссылками вида:

   Установить систему программирования VB5.0 CCE **CD-ROM** 

3. Материал в учебнике связан между собой гиперссылками вида:

    *Глава 3. Основы логики и логические основы компьютера*

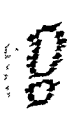
4. В тексте пособия приняты следующие обозначения и шрифтовые выделения:

   Шрифтом Arial выделены имена программ, файлов, папок, дисков и URL-адреса в Интернет.

   Курсивом выделены важные понятия и термины, а также названия диалоговых панелей, пунктов меню и управляющих элементов (текстовых полей, кнопок и так далее) графического интерфейса операционной системы Windows и ее приложений.

   Шрифтом `Courier` выделены тексты программ на языках программирования VBA, Visual Basic и представление Web-страниц на языке разметки гипертекста (HTML).

5. Важная информация (определения, формулы, синтаксис инструкций и методов в языке программирования) выделены в тексте восклицательным знаком:

   Важная информация

6. Начало выполнения каждого практического задания и разработки проекта обозначено значком .
7. Дополнительные материалы и тесты для проверки усвоения материала находятся в Интернете по адресу: http://iit.metodist.ru

Раздел I

ОСНОВЫ ИНФОРМАТИКИ

Введение в информатику

Глава 1
Компьютер
и программное обеспечение

Глава 2
Информация.
Двоичное кодирование информации

Глава 3
Основы логики
и логические основы компьютера

Глава 4
Основы алгоритмизации и объектно-ориентированного программирования

Глава 5
Моделирование и формализация

Глава 6
Информатизация общества

Введение в информатику

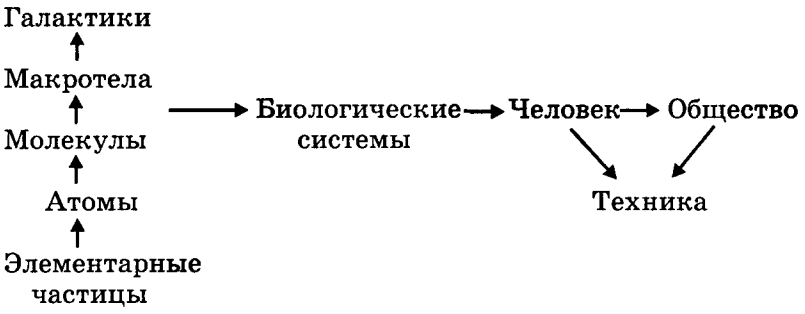
Вещественно-энергетическая картина мира. Мы живем в *макромире*, то есть в мире, который состоит из объектов, по своим размерам сравнимых с человеком. Обычно макрообъекты разделяют на неживые (здания, средства транспорта, мебель, одежда, станки и механизмы и так далее) и живые (растения, животные, сам человек).

Макрообъекты состоят из молекул и атомов, которые, в свою очередь, состоят из элементарных частиц, размеры которых чрезвычайно малы. Этот мир называется *микромиром*.

Мы живем на планете Земля, которая входит в Солнечную систему, Солнце вместе с миллионами других звезд образует нашу галактику Млечный путь, а миллионы галактик образуют Вселенную. Все эти объекты имеют громадные размеры и образуют *мегамир*. Все многообразие этих объектов состоит из *вещества*.

Согласно физической теории «Большого взрыва» наша Вселенная образовалась в результате взрыва сгустка «первоматерии» около 20 миллиардов лет назад. Тогда материя существовала фактически в форме энергии. Затем на протяжении долей секунды начало образовываться вещество в форме элементарных частиц. Постепенно структура вещества стала усложняться, из элементарных частиц стали образовываться атомы, а из атомов — молекулы. Из атомов и молекул за счет сил гравитационного притяжения образовались сложные структуры мегамира (звезды, планеты, галактики).

Окружающий мир можно представить в виде иерархического ряда объектов: элементарных частиц, атомов, молекул, макротел, звезд, галактик и так далее. Молекулы и макротела с течением времени образуют все более сложные биологические, социальные и технические системы.



Объекты окружающего мира

Поднятое над поверхностью земли тело обладает механической энергией, нагретый чайник — тепловой, заряженный проводник — электрической, ядра атомов — атомной.

Механическая энергия падающей воды вращает турбины гидроэлектростанций, тепловая энергия превращается в электрическую на тепловых электростанциях, атомная в электрическую — на атомных электростанциях. Электрическая энергия передается по проводам и с помощью электродвигателей превращается в механическую энергию (движение поездов, лифтов и так далее). Все материальные объекты взаимодействуют друг с другом и поэтому обладают *энергией*.

Вещественно-энергетическая картина мира начала складываться еще в античной философии, а с XVIII века формировалась в основном в рамках физической науки и химии. С середины XX века все большее внимание стало уделяться исследованию строения и функционирования сложных систем (биологических, социальных и технических) в рамках биологии и других наук. Однако не все особенности таких систем оказалось возможным объяснить в рамках традиционного вещественно-энергетического подхода.

Информационная картина мира. Строение и функционирование сложных систем различной природы (биологических, социальных, технических) оказалось невозможным объяснить, не рассматривая общих закономерностей информационных процессов. К концу XX века стала складываться, сначала в рамках кибернетики и биологии, а затем информатики, информационная картина мира. Информационная картина мира рассматривает окружающий мир под особым, информационным углом зрения, при этом она не противопоставлена вещественно-энергетической картине мира, но дополняет и развивает ее.

Информация в природе. Второе начало термодинамики, один из основных законов классической физики, утверждает, что если какую-либо систему «предоставить самой себе» и убрать все внешние воздействия (такую систему называют «закрытой»), то эта система будет стремиться к состоянию термодинамического равновесия. Составляющие ее элементы «перемешиваются», разрушается их структура и наступает полный беспорядок — хаос. Энтропия системы, которая является мерой беспорядка, возрастает, а информация (антиэнтропия), которая является мерой упорядоченности, уменьшается. В соответствии с такой точкой зрения нашу Вселенную ждет «тепловая смерть», то есть прекращение каких-либо изменений и развития.

Однако, по крайней мере, на нашей планете многое происходит наоборот: идет саморазвитие, эволюция живой природы, то есть повышение сложности и разнообразия живых систем. Жизнь является системой открытой, многообразными путями в нее поступают и вещество, и энергия, и информация. Потребляя энергию солнечного излучения в процессе фотосинтеза, растения строят сложные биологические молекулы из простых неорганических, далее животные, поедающие растения и друг друга, создают все более сложные живые структуры и так далее.

Таким образом, энтропия в живой природе уменьшается, а информация (антиэнтропия) — увеличивается.

Получение и преобразование информации является условием жизнедеятельности любого организма. Даже простейшие одноклеточные организмы постоянно воспринимают и используют информацию, например, о температуре и химическом составе среды для выбора наиболее благоприятных условий существования. Биологи образно говорят, что «живое питается информацией», создавая, накапливая и активно используя ее.

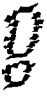
Генетическая информация. Любой живой организм, в том числе человек, является носителем генетической информации, которая передается по наследству. Генетическая информация хранится в каждой клетке организма в молекулах ДНК, которые состоят из отдельных участков (генов). Каждый ген «отвечает» за определенные особенности строения и функционирования организма и определяют как его возможности, так и предрасположенность к различным наследственным болезням.

Чем сложнее и высокоорганизованнее организм, тем большее количество генов содержится в молекуле ДНК. Работы

по расшифровке структуры генома человека, который содержит более 20 тысяч различных генов, проводились с использованием компьютерных технологий и были в основном закончены в 2000 году.

Человек и информация. Человек живет в мире *информации*. Человек воспринимает окружающий мир (*получает* информацию) с помощью органов чувств. Наибольшее количество информации (около 90%) человек получает с помощью зрения, около 9% — с помощью слуха и только 1% с помощью других органов чувств (обоняния, осязания и вкуса). Полученная человеком информация в форме зрительных, слуховых и других образов *хранится* в его памяти.

Человеческое мышление можно рассматривать как процессы *обработки информации* в мозгу человека. На основе информации, полученной с помощью органов чувств, и теоретических знаний, приобретенных в процессе обучения, человек создает информационные модели окружающего мира. Такие модели позволяют человеку ориентироваться в окружающем мире и принимать правильные решения для достижения поставленных целей.



Процессы, связанные с получением, хранением, обработкой и передачей информации, называются **информационными процессами**.

Информация и общество. В процессе общения с другими людьми человек *передает* и *получает* информацию. Обмен информацией между людьми может осуществляться в различных формах (письменной, устной или с помощью жестов). Для обмена информацией всегда используется определенный язык (русский, азбука Морзе и так далее). Для того чтобы информация была понятна, язык должен быть известен всем людям, участвующим в общении. Чем большее количество языков вы знаете, тем шире круг вашего общения.

История человеческого общества — это, в определенном смысле, история накопления и преобразования информации. Весь процесс познания является процессом получения, преобразования и накопления информации (знаний). Полученная информация хранится на носителях информации различных типов (книги, аудио- и видеокассеты и так далее), а в последнее время все больше на электронных носителях информации в цифровой форме (магнитные и лазерные диски и др.).

Объединение компьютеров в глобальную сеть Интернет позволило обеспечить для каждого человека потенциальную возможность быстрого доступа ко всему объему информации, накопленному человечеством за всю его историю.

Информационные процессы в технике. Информационные процессы характерны не только для природы, человека и общества, но и для техники. Нормальное функционирование технических устройств связано с процессами управления, которые включают в себя получение, хранение, преобразование и передачу информации. В некоторых случаях главную роль в процессе управления выполняет человек (например, вождение автомобиля), в других управление берет на себя само техническое устройство (например, кондиционер).

Аппаратные и программные средства информатизации. Человеком созданы специальные технические устройства, предназначенные для кодирования, обработки, хранения и передачи информации в цифровой форме (компьютер, принтер, сканер, модем и др.). Совокупность таких устройств принято называть *аппаратными средствами* информатизации.

Универсальным устройством, предназначенным для автоматической обработки информации, является компьютер. Управляют работой компьютера программы, которые имеют различные функции и назначение. Совокупность компьютерных программ называется программным обеспечением или *программными средствами информатизации.*

Информационные и коммуникационные технологии. Для создания на компьютере документа с использованием текстового редактора необходимо овладеть технологией обработки текстовой информации, для редактирования изображения с помощью графического редактора — технологией обработки графической информации, для проведения вычислений в электронных таблицах — технологией обработки числовой информации и так далее.

В процессе исследования информационных моделей приходится разрабатывать *алгоритмы* и затем кодировать их на языках программирования, то есть использовать технологию программирования.

Поиск и получение необходимой информации из глобальной компьютерной сети Интернет требует использования коммуникационных технологий.

В любом случае кроме использования определенных аппаратных и программных средств необходимо знать и уметь применять определенные информационные и коммуникационные технологии.

Информационные и коммуникационные технологии — это совокупность методов, устройств и производственных процессов, используемых обществом для сбора, хранения, обработки и распространения информации.

Информационное общество. В последние два десятилетия массовое производство персональных компьютеров и стремительный рост Интернета существенно ускорили становление информационного общества в развитых странах мира.

В информационном обществе главным ресурсом является информация, именно на основе владения информацией о самых различных процессах и явлениях можно эффективно и оптимально строить любую деятельность. Бóльшая часть населения в информационном обществе занята в сфере обработки информации или использует информационные и коммуникационные технологии в своей повседневной производственной деятельности.

Для жизни и деятельности в информационном обществе необходимо обладать информационной культурой, т. е. знаниями и умениями в области информационных технологий, а также быть знакомым с юридическими и этическими нормами в этой сфере.

Информатика. Информационный подход к исследованию мира реализуется в рамках информатики, комплексной науки об информации и информационных процессах, аппаратных и программных средствах информатизации, информационных и коммуникационных технологиях, а также социальных аспектах процесса информатизации.

Вопросы для размышления

1. Дайте краткую характеристику вещественно-энергетической картины мира.
2. Дайте краткую характеристику информационной картины мира.
3. Что изучает информатика?

Глава 1

Компьютер и программное обеспечение

1.1. Магистрально-модульный принцип построения компьютера

В основу архитектуры современных персональных компьютеров положен магистрально-модульный принцип. Модульный принцип позволяет потребителю самому комплектовать нужную ему конфигурацию компьютера и производить при необходимости ее модернизацию. Модульная организация компьютера опирается на магистральный (шинный) принцип обмена информацией между устройствами.

1.1.1. Магистраль

Магистраль (системная шина) включает в себя три многоразрядные шины: *шину данных*, *шину адреса* и *шину управления*, которые представляют собой многопроводные линии (рис. 1.1). К магистрали подключаются процессор и оперативная память, а также периферийные устройства ввода, вывода и хранения информации, которые обмениваются информацией на машинном языке (последовательностями нулей и единиц в форме электрических импульсов).

Шина данных. По этой шине данные передаются между различными устройствами. Например, считанные из оперативной памяти данные могут быть переданы процессору для обработки, а затем полученные данные могут быть отправлены обратно в оперативную память для хранения. Таким образом, данные по шине данных могут передаваться от устройства к устройству в любом направлении.

Разрядность шины данных определяется разрядностью процессора, то есть количеством двоичных разрядов, которые могут обрабатываться или передаваться процессором одновременно. Разрядность процессоров постоянно увеличивается по мере развития компьютерной техники.

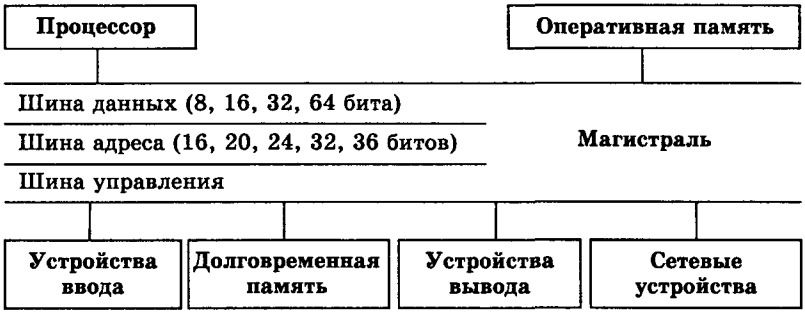


Рис. 1.1. Магистрально-модульное устройство компьютера

Шина адреса. Выбор устройства или ячейки памяти, куда пересылаются или откуда считываются данные по шине данных, производит процессор. Каждое устройство или ячейка оперативной памяти имеет свой адрес. Адрес передается по адресной шине, причем сигналы по ней передаются в одном направлении — от процессора к оперативной памяти и устройствам (однонаправленная шина).

Разрядность шины адреса определяет объем адресуемой памяти (адресное пространство), то есть количество однобайтовых ячеек оперативной памяти, которые могут иметь уникальные адреса. Количество адресуемых ячеек памяти можно рассчитать по формуле:

$$N = 2^I, \text{ где } I \text{ — разрядность шины адреса.}$$

Разрядность шины адреса постоянно увеличивалась и в современных персональных компьютерах составляет 36 бит. Таким образом, максимально возможное количество адресуемых ячеек памяти равно:

$$N = 2^{36} = 68\ 719\ 476\ 736.$$

Шина управления. По шине управления передаются сигналы, определяющие характер обмена информацией по магистрали. Сигналы управления показывают, какую операцию — считывание или запись информации из памяти — нужно производить, синхронизируют обмен информацией между устройствами и так далее.

1.1.2. Процессор и оперативная память

Процессор. Процессор аппаратно реализуется на большой интегральной схеме (БИС). Большая интегральная схема на самом деле не является «большой» по размеру и представляет собой, наоборот, маленькую плоскую полупроводниковую

пластину размером примерно 20×20 мм, заключенную в плоский корпус с рядами металлических штырьков (контактов). БИС является «большой» по количеству элементов.

Использование современных высоких технологий позволяет разместить на БИС процессора огромное количество (42 миллиона в процессоре Pentium 4 — рис. 1.2) функциональных элементов (переключателей), размеры которых составляют всего около 0,13 микрон (1 микрон = 10^{-6} метра).

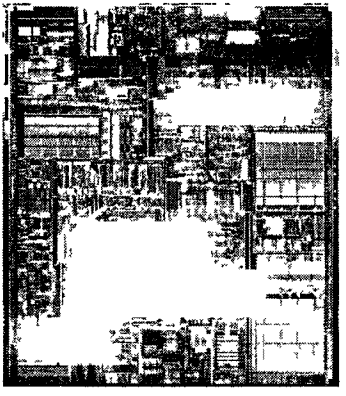


Рис. 1.2. Внутренняя структура процессора Intel Pentium 4

Важнейшей характеристикой, определяющей быстродействие процессора, является *тактовая частота*, то есть количество тактов в секунду. Такт — это промежуток времени между началами подачи двух последовательных импульсов специальной микросхемой — генератором тактовой частоты, синхронизирующим работу узлов компьютера. На выполнение процессором каждой базовой операции (например, сложения) отводится определенное количество тактов. Ясно, что чем больше тактовая частота, тем больше операций в секунду выполняет процессор. Тактовая частота измеряется в мегагерцах (МГц) и гигагерцах (ГГц). 1 МГц = миллион тактов в секунду. За 20 с небольшим лет тактовая частота процессора увеличилась почти в 500 раз, от 5 МГц (процессор 8086, 1978 год) до 2,4 ГГц (процессор Pentium 4, 2002 год) — табл. 1.1.

Таблица 1.1. Процессоры

Тип	Год выпуска	Частота (МГц)	Шина данных	Шина адреса	Адресуемая память
8086	1978	5–10	16	20	1 Мб
80286	1982	6–12,5	16	24	16 Мб
80386	1985	16–33	32	32	4 Гб
80486	1989	25–50	32	32	4 Гб
Pentium	1993	60–166	64	32	4 Гб
Pentium II	1997	200–300	64	36	64 Гб
Pentium III	1999	450–1000	64	36	64 Гб
Pentium 4	2000	1000–2400	64	36	64 Гб

Другой характеристикой процессора, влияющей на его производительность, является *разрядность* процессора. Разрядность процессора определяется количеством двоичных разрядов, которые могут передаваться или обрабатываться процессором одновременно. Часто уточняют разрядность процессора и пишут 64/36, что означает, что процессор имеет 64-разрядную шину данных и 36-разрядную шину адреса.

В первом отечественном школьном компьютере «Агат» (1985 год) был установлен процессор, имевший разрядность 8/16, соответственно одновременно он обрабатывал 8 битов, а его адресное пространство составляло 64 килобайта.

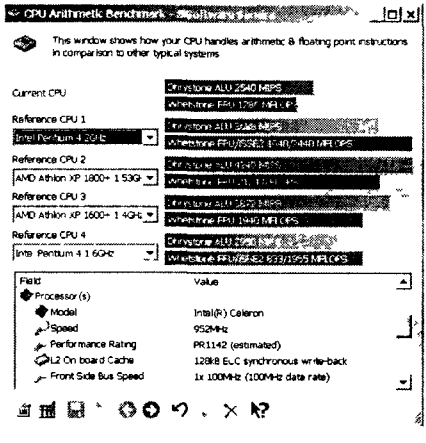
Современный процессор Pentium 4 имеет разрядность 64/36, то есть одновременно процессор обрабатывает 64 бита, а адресное пространство составляет 68 719 476 736 байтов — 64 гигабайта.

Производительность процессора является его интегральной характеристикой, которая зависит от частоты процессора, его разрядности, а также особенностей архитектуры (наличие кэш-памяти и др.). Производительность процессора нельзя вычислить, она определяется в процессе тестирования, по скорости выполнения процессором определенных операций в какой-либо программной среде.

Установить программу тестирования компьютера **SiSoftware Sandra** **CD-ROM**

Тестирование процессора

1. Запустить программу SiSoftware Sandra.
   В окне программы активизировать тестирующий модуль *CPU Arithmetic Benchmark*.
   Через несколько десятков секунд появится информационное окно, которое показывает производительность установленного процессора в сравнении с производительностью четырех типов процессоров.



Оперативная память. Оперативная память, предназначенная для хранения информации, изготавливается в виде модулей памяти. Модули памяти представляют собой пластины с рядами контактов, на которых размещаются БИС памяти. Модули памяти могут различаться между собой по размеру и количеству контактов (DIMM, RIMM, DDR — рис. 1.3), быстродействию, информационной емкости и так далее.

Важнейшей характеристикой модулей оперативной памяти является быстродействие, которое зависит от максимально возможной частоты операций записи или считывания информации из ячеек памяти. Современные модули памяти обеспечивают частоту до 800 МГц, а их информационная емкость может достигать 512 Мбайт.

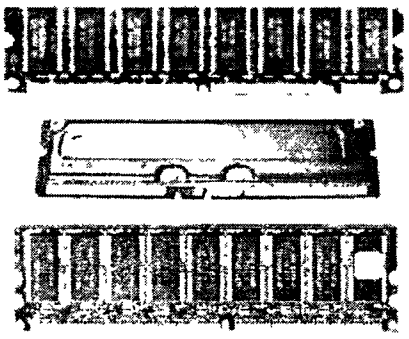


Рис. 1.3. Модули памяти DIMM, RIMM, DDR

В персональных компьютерах объем адресуемой памяти и величина фактически установленной оперативной памяти практически всегда различаются. Хотя объем адресуемой памяти может достигать 64 Гбайт, величина фактически установленной оперативной памяти может быть значительно меньше, например, «всего» 64 Мбайт.

Вопросы для размышления

1. Какие технические характеристики и как влияют на производительность компьютера?

Практические задания

1.1. Ознакомиться с историей создания, технологией изготовления и техническими характеристиками процессоров в виртуальном музее фирмы Intel, размещенном по адресу: http://www.intel.ru.

1.2. С помощью программы тестирования SiSoftware Sandra провести тестирование модулей оперативной памяти.

1.2. Аппаратная реализация компьютера

Современный персональный компьютер может быть реализован в настольном (desktop), портативном (notebook) или карманном (handheld) варианте.

1.2.1. Системный блок компьютера

Все основные компоненты настольного компьютера находятся внутри системного блока: системная плата с процессором и оперативной памятью, накопители на жестких и гибких дисках, CD-ROM и др. Кроме этого, в системном блоке находится блок питания.

Системная плата. Основным аппаратным компонентом компьютера является системная плата (рис. 1.4). На системной плате реализована магистраль обмена информацией, имеются разъемы для установки процессора и оперативной памяти, а также слоты для установки контроллеров внешних устройств.

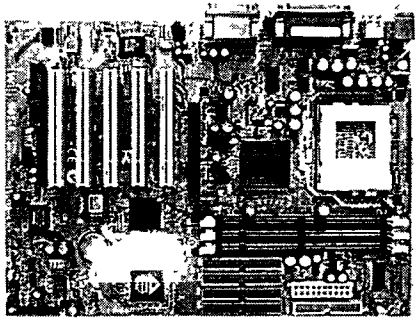


Рис. 1.4. Системная плата

Частота процессора, системной шины и шин периферийных устройств. Быстродействие различных компонентов компьютера (процессора, оперативной памяти и контроллеров периферийных устройств) может существенно различаться. Для согласования быстродействия на системной плате устанавливаются специальные микросхемы (чипсеты), включающие в себя контроллер оперативной памяти (так называемый северный мост) и контроллер периферийных устройств (южный мост) — рис. 1.5.

Северный мост обеспечивает обмен информацией между процессором и оперативной памятью по системной шине. В процессоре используется внутреннее умножение частоты, поэтому частота процессора в несколько раз больше, чем частота системной шины. В современных компьютерах частота процессора может превышать частоту системной шины в 10 раз (например, частота процессора 1 ГГц, а частота шины — 100 МГц).

Рис. 1.5. Логическая схема системной платы

К северному мосту подключается шина PCI (Peripherial Component Interconnect bus — шина взаимодействия периферийных устройств), которая обеспечивает обмен информацией с контроллерами периферийных устройств. Частота контроллеров меньше частоты системной шины, например, если частота системной шины составляет 100 МГц, то частота шины PCI обычно в три раза меньше — 33 МГц. Контроллеры периферийных устройств (звуковая плата, сетевая плата, SCSI-контроллер, внутренний модем) устанавливаются в слоты расширения системной платы.

По мере увеличения разрешающей способности монитора и глубины цвета требования к быстродействию шины, связывающей видеоплату с процессором и оперативной памятью, возрастают. В настоящее время для подключения видеоплаты обычно используется специальная шина AGP (Accelerated Graphic Port — ускоренный графический порт), соединенная с северным мостом и имеющая частоту, в несколько раз большую, чем шина PCI.

Южный мост обеспечивает обмен информацией между северным мостом и портами для подключения периферийного оборудования.

Устройства хранения информации (жесткие диски, CD-ROM, DVD-ROM) подключаются к южному мосту по шине UDMA (Ultra Direct Memory Access — прямое подключение к памяти).

Мышь и внешний модем подключаются к южному мосту с помощью *последовательных портов*, которые передают электрические импульсы, несущие информацию в машинном коде, последовательно один за другим. Обозначаются последовательные порты как COM1 и COM2, а аппаратно реализуются с помощью 25-контактного и 9-контактного разъемов, которые выведены на заднюю панель системного блока.

Принтер подключается к *параллельному порту*, который обеспечивает более высокую скорость передачи информации, чем последовательные порты, так как передает одновременно 8 электрических импульсов, несущих информацию в машинном коде. Обозначается параллельный порт как LPT, а аппаратно реализуется в виде 25-контактного разъема на задней панели системного блока.

Для подключения сканеров и цифровых камер обычно используется порт USB (Universal Serial Bus — универсальная последовательная шина), который обеспечивает высокоскоростное подключение к компьютеру сразу нескольких периферийных устройств.

Клавиатура подключается обычно с помощью порта PS/2.

Вопросы для размышления

1. Почему различаются частоты процессора, системной шины и шины периферийных устройств?
2. Почему мышь подключается к последовательному порту, а принтер к параллельному?

Практические задания

1.3. С помощью программы тестирования SiSoftware Sandra провести тестирование материнской платы и определить частоты процессора, системной шины, шины периферийных устройств и шины AGP.

1.2.2. Внешняя (долговременная) память

Основной функцией внешней памяти компьютера является способность долговременно хранить большой объем информации (программы, документы, аудио- и видеоклипы и пр.). Устройство, которое обеспечивает запись/считывание информации, называется *накопителем*, или *дисководом*, а хранится информация на *носителях* (например, дискетах).

Магнитный принцип записи и считывания информации. В накопителях на гибких магнитных дисках (НГМД) и накопителях на жестких магнитных дисках (НЖМД), или винчестерах, в основу записи информации положено намагничивание ферромагнетиков в магнитном поле, хранение информации основывается на сохранении намагниченности, а считывание информации базируется на явлении электромагнитной индукции.

Физика-10 ☞

В процессе записи информации на гибкие и жесткие магнитные диски головка дисковода с сердечником из магнитомягкого материала (малая остаточная намагниченность) перемещается вдоль магнитного слоя магнитожесткого носителя (большая остаточная намагниченность). На магнитную головку поступают последовательности электрических импульсов (последовательности логических единиц и нулей), которые создают в головке магнитное поле. В результате последовательно намагничиваются (логическая единица) или не намагничиваются (логический нуль) элементы поверхности носителя.

В отсутствие сильных магнитных полей и высоких температур элементы носителя могут сохранять свою намагниченность в течение долгого времени (лет и десятилетий).

При считывании информации при движении магнитной головки над поверхностью носителя намагниченные участки носителя вызывают в ней импульсы тока (явление электромагнитной индукции). Последовательности таких импульсов передаются по магистрали в оперативную память компьютера.

Гибкие магнитные диски. Гибкие магнитные диски помещаются в пластмассовый корпус. Такой носитель информации называется дискетой. В центре дискеты имеется приспособление для захвата и обеспечения вращения диска внутри пластмассового корпуса. Дискета вставляется в дисковод, который вращает диск с постоянной угловой скоростью.

При этом магнитная головка дисковода устанавливается на определенную концентрическую дорожку диска, на которую и производится запись или с которой производится считывание информации. Информационная емкость дискеты невелика и составляет всего 1,44 Мбайт. Скорость записи и считывания информации также мала (составляет всего около 50 Кбайт/с) из-за медленного вращения диска (360 об./мин).

В целях сохранения информации гибкие магнитные диски необходимо предохранять от воздействия сильных магнитных полей и нагревания, так как такие физические воздействия могут привести к размагничиванию носителя и потере информации.

Жесткие магнитные диски. Жесткий магнитный диск представляет собой несколько десятков дисков, размещенных на одной оси, заключенных в металлический корпус и вращающихся с большой угловой скоростью (рис. 1.6).

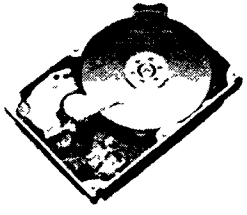


Рис. 1.6. Жесткий магнитный диск

За счет гораздо большего количества дорожек на каждой стороне дисков и большого количества дисков информационная емкость жесткого диска может в сотни тысяч раз превышать информационную емкость дискеты и достигать 150 Гбайт. Скорость записи и считывания информации с жестких дисков достаточно велика (может достигать 133 Мбайт/с) за счет быстрого вращения дисков (до 7200 об./мин).

В жестких дисках используются достаточно хрупкие и миниатюрные элементы (пластины носителей, магнитные головки и пр.), поэтому в целях сохранения информации и работоспособности жесткие диски необходимо оберегать от ударов и резких изменений пространственной ориентации в процессе работы.

Оптический принцип записи и считывания информации. В лазерных дисководах CD-ROM и DVD-ROM используется оптический принцип записи и считывания информации.

Физика-11

В процессе записи информации на лазерные диски для создания участков поверхности с различными коэффициентами отражения применяются различные технологии: от простой штамповки до изменения отражающей способности

участков поверхности диска с помощью мощного лазера. Информация на лазерном диске записывается на одну спиралевидную дорожку (как на грампластинке), содержащую чередующиеся участки с различной отражающей способностью.

При соблюдении правил хранения (в футлярах в вертикальном положении) и эксплуатации (без нанесения царапин и загрязнений) оптические носители могут сохранять информацию в течение десятков лет.

В процессе считывания информации с лазерных дисков луч лазера, установленного в дисководе, падает на поверхность вращающегося диска и отражается. Так как поверхность лазерного диска имеет участки с различными коэффициентами отражения, то отраженный луч также меняет свою интенсивность (логические 0 или 1). Затем отраженные световые импульсы преобразуются с помощью фотоэлементов в электрические импульсы и по магистрали передаются в оперативную память.

Лазерные дисководы и диски. Лазерные дисководы (CD-ROM и DVD-ROM — рис. 1.7) используют оптический принцип чтения информации.

На лазерных CD-ROM (CD — Compact Disk, компакт-диск) и DVD-ROM (DVD — Digital Video Disk, цифровой видеодиск) дисках хранится информация, которая была записана на них в процессе изготовления. Запись на них новой информации невозможна, что отражено во второй части их названий: ROM (Read Only Memory — только чтение). Производятся такие диски путем штамповки и имеют серебристый цвет.

Информационная емкость CD-ROM диска может достигать 650 Мбайт, а скорость считывания информации в CD-ROM-накопителе зависит от скорости вращения диска. Первые CD-ROM-накопители были односкоростными и обеспечивали скорость считывания информации 150 Кбайт/с. В настоящее время широкое распространение получили 52-скоростные CD-ROM-накопители, которые обеспечивают в 52 раза большую скорость считывания информации (до 7,8 Мбайт/с).

DVD-диски имеют гораздо большую информационную емкость (до 17 Гбайт) по сравнению CD-дисками. Во-первых, используются лазеры с меньшей длиной волны, что позволяет размещать оптические дорожки более плотно. Во-вторых, информация на DVD-дисках может быть записана на двух сторонах, причем в два слоя на одной стороне.

Рис. 1.7. CD-ROM и DVD-ROM

Первое поколение DVD-ROM-накопителей обеспечивало скорость считывания информации примерно 1,3 Мбайт/с. В настоящее время 16-скоростные DVD-ROM-дисководы достигают скорости считывания до 21 Мбайт/с.

Существуют CD-R и DVD-R-диски (R — recordable, записываемый), которые имеют золотистый цвет. Информация на такие диски может быть записана, но только один раз. На дисках CD-RW и DVD-RW (RW — ReWritable, перезаписываемый), которые имеют «платиновый» оттенок, информация может быть записана многократно.

Для записи и перезаписи на диски используются специальные CD-RW и DVD-RW-дисководы, которые обладают достаточно мощным лазером, позволяющим менять отражающую способность участков поверхности в процессе записи диска. Такие дисководы позволяют записывать и считывать информацию с дисков с различной скоростью. Например, маркировка CD-RW-дисковода «40×12×48» означает, что запись CD-R-дисков производится на 40-кратной скорости, запись CD-RW-дисков — на 12-кратной, а чтение — на 48-кратной скорости.

Flash-память. Flash-память — это энергонезависимый тип памяти, позволяющий записывать и хранить данные в микросхемах. Карты flash-памяти (рис. 1.8) не имеют в своем составе движущихся частей, что обеспечивает высокую сохранность данных при их использовании в мобильных устройствах (портативных компьютерах, цифровых камерах и др.).

Рис. 1.8. Карты flash-памяти

Flash-память представляет собой микросхему, помещенную в миниатюрный плоский корпус. Для считывания или записи информации карта памяти вставляется в специаль-

ные накопители, встроенные в мобильные устройства или подключаемые к компьютеру через USB-порт. Информационная емкость карт памяти может достигать 512 Мбайт.

К недостаткам flash-памяти следует отнести то, что не существует единого стандарта и различные производители изготавливают несовместимые друг с другом по размерам и электрическим параметрам карты памяти.

Вопросы для размышления

1. Каковы основные правила хранения и эксплуатации различных типов носителей информации?

Практические задания

1.4. Составить сравнительную таблицу основных параметров устройств хранения информации (емкость, скорость обмена, надежность хранения информации, цена хранения одного мегабайта).

1.2.3. Устройства ввода информации

Клавиатура. Универсальным устройством ввода информации является клавиатура (рис. 1.9). Клавиатура позволяет вводить числовую и текстовую информацию. Стандартная клавиатура имеет 104 клавиши и 3 информирующих о режимах работы световых индикатора в правом верхнем углу.

Координатные устройства ввода. Для ввода графической информации и для работы с графическим интерфейсом программ используются координатные устройства ввода информации: манипуляторы (*мышь*, *трекбол*), сенсорные панели *тачпад* и графические планшеты.

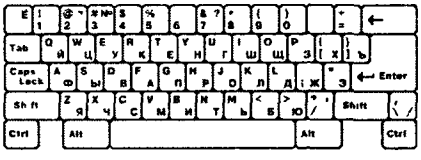


Рис. 1.9. Клавиатура

В оптико-механических манипуляторах *мышь* и *трекбол* основным рабочим органом является массивный шар (металлический, покрытый резиной). У мыши он вращается при перемещении ее корпуса по горизонтальной поверхности, а у трекбола вращается непосредственно рукой.

Вращение шара передается двум пластмассовым валам, положение которых с большой точностью считывается инфракрасными оптопарами (то есть парами «светоизлучатель–фотоприемник») и затем преобразуется в электрический сигнал, управляющий движением указателя мыши на экране монитора. Главным «врагом» мыши является загрязнение, а способом борьбы с ним — использование специального «мышиного» коврика.

В настоящее время широкое распространение получили оптические мыши, в которых нет механических частей. Источник света, размещенный внутри мыши, освещает поверхность, а отраженный свет фиксируется фотоприемником и преобразуется в перемещение курсора на экране.

Разрешающая способность мышей обычно составляет около 600 dpi (dot per inch — точек на дюйм). Это означает, что при перемещении мыши на 1 дюйм (1 дюйм = 2,54 см) указатель мыши на экране перемещается на 600 точек.

Манипуляторы имеют обычно две кнопки управления, которые используются при работе с графическим интерфейсом программ. В настоящее время появились мыши с дополнительным колесиком, которое располагается между кнопками. Оно предназначено для прокрутки вверх или вниз не умещающихся целиком на экране изображений, текстов или Web-страниц.

Современные модели мышей и трекболов часто являются беспроводными, то есть подключаются к компьютеру без помощи кабеля (рис. 1.10).

Рис. 1.10. Манипуляторы оптическая беспроводная мышь и трекбол

В портативных компьютерах вместо манипуляторов используется сенсорная панель *тачпад* (от английского слова TouchPad), которая представляет собой панель прямоугольной формы, чувствительную к перемещению пальца и нажатию пальцем. Перемещение пальца по поверхности сенсорной панели преобразуется в перемещение курсора на экране монитора. Нажатие на поверхность сенсорной панели эквивалентно нажатию на кнопку мыши.

Рис. 1.11. Графический планшет

Для рисования и ввода рукописного текста используются *графические планшеты* (рис. 1.11). С помощью

специальной ручки можно чертить, рисовать схемы, добавлять заметки и подписи к электронным документам. Качество графических планшетов характеризуется разрешающей способностью, которая измеряется в lpi (lines per inch — линиях на дюйм) и способностью реагировать на силу нажатия пера.

В хороших планшетах разрешающая способность достигает 2048 lpi (перемещение пера по поверхности планшета на 1 дюйм соответствует перемещению на 2048 точек на экране монитора), а количество воспринимаемых градаций нажатий на перо составляет 1024.

Сканер. Для оптического ввода в компьютер и преобразования в компьютерную форму изображений (фотографий, рисунков, слайдов), а также текстовых документов используется *сканер* (рис. 1.12).

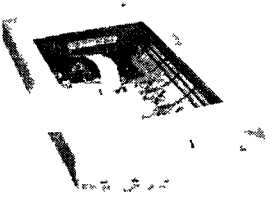


Рис. 1.12. Сканер

Сканируемое изображение освещается белым светом (черно-белые сканеры) или тремя цветами (красным, зеленым и синим). Отраженный свет проецируется на линейку фотоэлементов, которая движется, последовательно считывает изображение и преобразует его в компьютерный формат. В отсканированном изображении количество различаемых цветов может достигать десятков миллиардов.

Системы распознавания текстовой информации позволяют преобразовать отсканированный текст из графического формата в текстовый. Такие системы способны распознавать текстовые документы на различных языках, представленные в различных формах (например, таблицах) и с различным качеством печати (начиная от машинописных документов).

Разрешающая способность сканеров составляет 600 dpi и выше, то есть на полоске изображения длиной 1 дюйм сканер может распознать 600 и более точек.

Цифровые камеры и ТВ-тюнеры. Последние годы все большее распространение получают *цифровые камеры* (видеокамеры и фотоаппараты — рис. 1.13). Цифровые камеры позволяют получать видеоизображение и фотоснимки непосредственно в цифровом (компьютерном) формате.

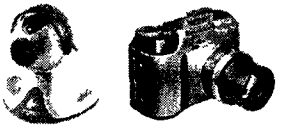


Рис. 1.13. Web-камера и цифровая фотокамера

Цифровые видеокамеры могут быть подключены к компьютеру, что позволяет сохранять видеозаписи в компьютерном формате.

Для передачи «живого» видео по компьютерным сетям используются недорогие Web-камеры, разрешающая способность которых обычно не превышает 640×480 точек.

Цифровые фотоаппараты позволяют получать высококачественные фотографии с разрешением до 2272×1704 точек (всего до 3,9 млн пикселей). Для хранения фотографий используются модули flash-памяти или жесткие диски очень маленького размера. Запись изображений на жесткий диск компьютера может осуществляться путем подключения камеры к компьютеру.

Если установить в компьютер специальную плату (ТВ-тюнер) и подключить к ее входу телевизионную антенну, то появляется возможность просматривать телевизионные передачи непосредственно на компьютере.

Звуковая карта. Звуковая карта производит преобразование звука из аналоговой формы в цифровую. Для ввода звуковой информации используется микрофон, который подключается к входу звуковой карты. Звуковая карта имеет также возможность синтезировать звук (в ее памяти хранятся звуки различных музыкальных инструментов, которые она может воспроизводить).

Многие звуковые платы имеют специальный игровой порт (GAME-порт), к которому подключаются игровые манипуляторы (джойстики), которые предназначены для более удобного управления ходом компьютерных игр.

Вопросы для размышления

1. Какие основные группы клавиш можно выделить на клавиатуре и каково их назначение?
2. Какие существуют типы координатных устройств ввода и каков их принцип действия?

Практические задания

1.5. Экспериментальным путем определить разрешение вашей мыши. Сравнить со значением, приведенным в техническом описании.

1.2.4. Устройства вывода информации

Монитор. *Монитор* является универсальным устройством вывода информации и подключается к видеокарте, установленной в компьютере.

Изображение в компьютерном формате (в виде последовательностей нулей и единиц) хранится в видеопамяти, размещенной на видеокарте. Изображение на экране монитора формируется путем считывания содержимого видеопамяти и отображения его на экран.

2.12. Двоичное кодирование графической информации

Частота считывания изображения влияет на стабильность изображения на экране. В современных мониторах обновление изображения происходит обычно с частотой 75 и более раз в секунду, что обеспечивает комфортность восприятия изображения пользователем компьютера (человек не замечает мерцание изображения). Для сравнения можно напомнить, что частота смены кадров в кино составляет 24 кадра в секунду.

В настольных компьютерах обычно используются мониторы на электронно-лучевой трубке (ЭЛТ) — рис. 1.14. Изображение на экране монитора создается пучком электронов, испускаемых электронной пушкой. Этот пучок электронов разгоняется высоким электрическим напряжением (десятки киловольт) и падает на внутреннюю поверхность экрана, покрытую люминофором (веществом, светящимся под воздействием пучка электронов).

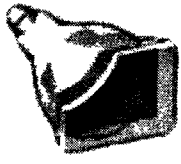


Рис. 1.14. ЭЛТ монитора

Система управления пучком заставляет пробегать его построчно весь экран (создает растр), а также регулирует его интенсивность (соответственно яркость свечения точки люминофора). Пользователь видит изображение на экране монитора, так как люминофор излучает световые лучи в видимой части спектра. Качество изображения тем выше, чем меньше размер точки изображения (точки люминофора), в высококачественных мониторах размер точки составляет 0,22 мм.

Однако монитор является также источником высокого статического электрического потенциала, электромагнитного и рентгеновского излучений, которые могут оказывать неблагоприятное воздействие на здоровье человека. Современные мониторы практически безопасны, так как соответ-

ствуют жестким санитарно-гигиеническим требованиям, зафиксированным в международном стандарте безопасности ТСО'99.

В портативных и карманных компьютерах применяют плоские мониторы на жидких кристаллах (ЖК). В последнее время такие мониторы стали использоваться и в настольных компьютерах.

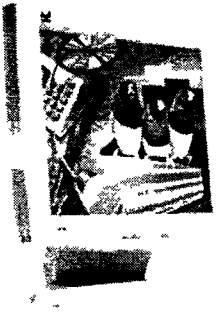


Рис. 1.15. Монитор на ЖК

LCD (Liquid Crystal Display, *жидкокристаллические мониторы* — рис. 1.15) сделаны из вещества, которое находится в жидком состоянии, но при этом обладает некоторыми свойствами, присущими кристаллическим телам. Фактически это жидкости, обладающие анизотропией свойств (в частности, оптических), связанных с упорядоченностью в ориентации молекул. Молекулы жидких кристаллов под воздействием электрического напряжения могут изменять свою ориентацию и вследствие этого изменять свойства светового луча, проходящего сквозь них.

Преимущество ЖК-мониторов перед мониторами на ЭЛТ состоит в отсутствии вредных для человека электромагнитных излучений и компактности.

Мониторы могут иметь различный размер экрана. Размер диагонали экрана измеряется в дюймах (1 дюйм = 2,54 см) и обычно составляет 15, 17 и более дюймов.

Принтеры. *Принтеры* предназначены для вывода на бумагу (создания «твердой копии») числовой, текстовой и графической информации. По своему принципу действия принтеры делятся на матричные, струйные и лазерные.

Матричные принтеры (рис. 1.16) — это принтеры ударного действия. Печатающая головка матричного принтера состоит из вертикального столбца маленьких стержней (обычно 9 или 24), которые под воздействием магнитного поля «выталкиваются» из головки и ударяют по бумаге (через красящую ленту). Перемещаясь, печатающая головка оставляет на бумаге строку символов.

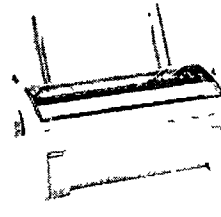


Рис. 1.16. Матричный принтер

Недостатки матричных принтеров состоят в том, что они печатают медленно, производят много шума и качество печати оставляет желать лучшего (соответствует примерно качеству пишущей машинки).

В последние годы широкое распространение получили черно-белые и цветные струйные принтеры (рис. 1.17). В них используется чернильная печатающая головка, которая под давлением выбрасывает чернила из ряда мельчайших отверстий на бумагу. Перемещаясь вдоль бумаги, печатающая головка оставляет строку символов или полоску изображения.

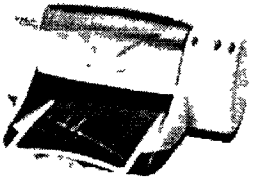


Рис. 1.17. Струйный принтер

Струйные принтеры могут печатать достаточно быстро (до нескольких страниц в минуту) и производят мало шума. Качество печати (в том числе и цветной) определяется разрешающей способностью струйных принтеров, которая может достигать фотографического качества 2400 dpi. Это означает, что полоска изображения по горизонтали длиной в 1 дюйм формируется из 2400 точек (чернильных капель).

Лазерные принтеры (рис. 1.18) обеспечивают практически бесшумную печать. Высокую скорость печати (до 30 страниц в минуту) лазерные принтеры достигают за счет постраничной печати, при которой страница печатается сразу целиком.

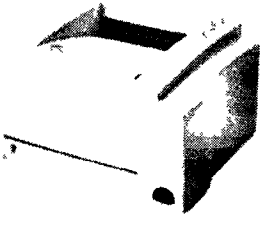


Рис. 1.18. Лазерный принтер

Высокое типографское качество печати лазерных принтеров обеспечивается за счет высокой разрешающей способности, которая может достигать 1200 dpi и более.

Плоттер. Для вывода сложных и широкоформатных графических объектов (плакатов, чертежей, электрических и электронных схем и пр.) используются специальные устройства вывода — *плоттеры* (рис. 1.19). Принцип действия плоттера такой же, как и струйного принтера.

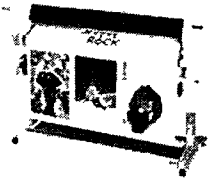


Рис. 1.19. Плоттер

Акустические колонки и наушники. Для прослушивания звука используются акустические колонки или наушники, которые подключаются к выходу звуковой платы.

Вопросы для размышления

1. Какие физические параметры влияют на качество изображения на экране монитора?

Практические задания

1.6. Ознакомиться с устройством компьютера и историей вычислительной техники, посетив виртуальные компьютерные музеи в Интернете.

1.3. Операционная система: назначение и состав

На IBM-совместимых персональных компьютерах используются операционные системы корпорации Microsoft Windows 9x/ME, а также свободно распространяемая операционная система Linux. На персональных компьютерах фирмы Apple используются различные версии операционной системы Mac OS. На рабочих станциях и серверах наибольшее распространение получили операционные системы Windows NT/2000/XP и UNIX.

Операционные системы разные, но их назначение и функции одинаковые. Операционная система является базовой и необходимой составляющей программного обеспечения компьютера, без нее компьютер не может работать в принципе.

Операционная система обеспечивает совместное функционирование всех устройств компьютера и предоставляет пользователю доступ к его ресурсам.

Современные операционные системы имеют сложную структуру, каждый элемент которой выполняет определенные функции по управлению компьютером.

Управление файловой системой. Процесс работы компьютера в определенном смысле сводится к обмену файлами между устройствами. В операционной системе имеются *программные модули, управляющие файловой системой.*

Командный процессор. В состав операционной системы входит специальная программа — *командный процессор*, — которая запрашивает у пользователя команды и выполняет их.

Пользователь может дать команду запуска программы, выполнения какой-либо операции над файлами (копирование, удаление, переименование), вывода документа на печать и так далее. Операционная система должна эту команду выполнить.

Драйверы устройств. К магистрали компьютера подключаются различные устройства (дисководы, монитор, клавиатура, мышь, принтер и др.). Каждое устройство выполняет определенную функцию (ввод информации, хранение информации, вывод информации), при этом техническая реализация устройств существенно различается.

В состав операционной системы входят *драйверы устройств*, специальные программы, которые обеспечивают управление работой устройств и согласование информационного обмена с другими устройствами, а также позволяют производить настройку некоторых параметров устройств. Каждому устройству соответствует свой драйвер.

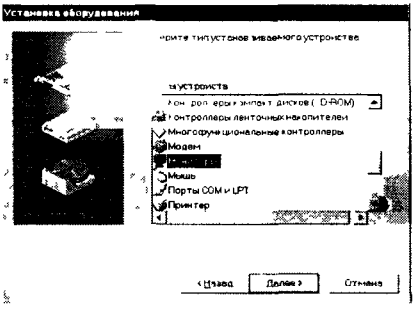
Технология «Plug and Play» (подключи и играй) позволяет автоматизировать подключение к компьютеру новых устройств и обеспечивает их конфигурирование. В процессе установки Windows определяет тип и конкретную модель установленного устройства и подключает необходимый для его функционирования драйвер. При включении компьютера производится загрузка драйверов в оперативную память.

Пользователь имеет возможность вручную установить или переустановить драйверы.

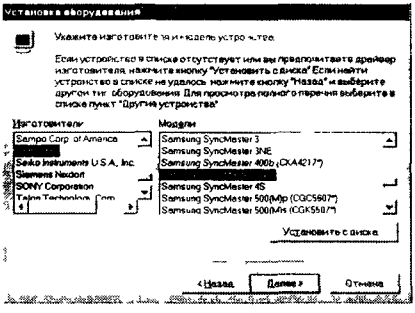
Установка драйвера монитора

1. Ввести команду [Настройка-Панель управления-Установка оборудования].

2. Запустится программа Мастер установки оборудования, которая предложит пользователю сначала выбрать тип устройства (в данном случае *Мониторы*).



3. На втором шаге необходимо выбрать фирму-производитель и марку монитора (например, *Samsung* и *Samsung SyncMaster 4NE*). В результате будет установлен драйвер данного монитора.



4. Если фирмы-производителя или марки устройства нет в списке, то необходимо щелкнуть левой кнопкой мыши на кнопке *Установить с диска*, чтобы установить драйвер с диска, который прилагается к устройству.

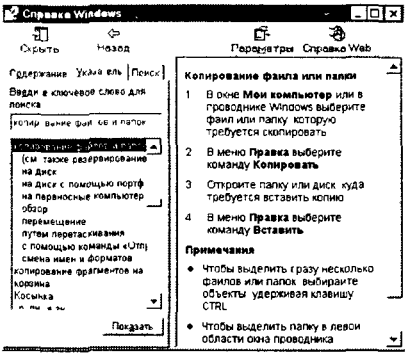
Графический интерфейс. Для упрощения работы пользователя в состав современных операционных систем, и в частности в состав Windows, входят *программные модули, создающие графический пользовательский интерфейс*. В операционных системах с графическим интерфейсом пользователь может вводить команды с помощью мыши, тогда как в режиме командной строки необходимо вводить команды с помощью клавиатуры.

Сервисные программы. В состав операционной системы входят также *сервисные программы*, или *утилиты*. Такие программы позволяют обслуживать диски (проверять, сжимать, дефрагментировать и так далее), выполнять операции с файлами (архивировать и так далее), работать в компьютерных сетях и так далее.

Справочная система. Для удобства пользователя в состав операционной системы обычно входит также *справочная система*. Справочная система позволяет оперативно получить необходимую информацию как о функционировании операционной системы в целом, так и о работе ее отдельных модулей.

Работа со справочной системой

1. Вызов справочной системы осуществляется из *Главного меню* командой [Справка].
2. На диалоговой панели *Справка Windows* пользователь может выбрать один из трех способов поиска необходимой ему справочной информации с помощью вкладок *Содержание, Указатель, Поиск.*



Воспользуемся вкладкой *Указатель* для поиска информации о способах копирования файлов и папок. Для этого:

- введем в поле ввода слово *копирование*, в левом окне появится список вариантов копирования различных объектов;
- найдем и активизируем нужный элемент списка: *копирование файлов и папок;*
- в правом окне появится пояснение по проведению операций копирования.

Вопросы для размышления

1. Для чего необходима операционная система?
2. Какие компоненты входят в состав операционной системы?

Практические задания

1.7. Проверить, какие марки монитора и видеоадаптера установлены в Windows и соответствуют ли они реально установленным в компьютере.

1.8. Найти в справочной системе операционной системы информацию об установке драйверов.

1.4. Загрузка операционной системы

Файлы операционной системы хранятся во внешней, долговременной памяти (на жестком, гибком или лазерном диске). Однако программы могут выполняться, только если они находятся в оперативной памяти, поэтому файлы операционной системы необходимо загрузить в оперативную память.

Диск (жесткий, гибкий или лазерный), на котором находятся файлы операционной системы и с которого производится ее загрузка, называется **системным**.

После включения компьютера производится загрузка операционной системы с системного диска в оперативную память. Загрузка должна выполняться в соответствии с программой загрузки. Однако для того чтобы компьютер выполнял какую-нибудь программу, эта программа должна уже находиться в оперативной памяти. Разрешение этого противоречия состоит в последовательной, поэтапной загрузке операционной системы.

Самотестирование компьютера. В состав компьютера входит энергонезависимое постоянное запоминающее устройство (ПЗУ), содержащее программы тестирования компьютера и первого этапа загрузки операционной системы — это BIOS (Basic Input/Output System — базовая система ввода/вывода).

После включения питания компьютера или нажатия кнопки *Reset* на системном блоке компьютера или одновременного нажатия комбинации клавиш {*Ctrl+Alt+Del*} на клавиатуре процессор начинает выполнение программы самотестирования компьютера POST (Power-ON Self Test). Производится тестирование работоспособности процессора, памяти и других аппаратных средств компьютера.

В процессе тестирования сначала могут выдаваться диагностические сообщения в виде различных последовательностей коротких и длинных звуковых сигналов (например, 1 длинный и 3 коротких — не подключен монитор, 5 коротких — ошибка процессора и так далее). После успешной инициализации видеокарты краткие диагностические сообщения выводятся на экран монитора.

Для установки правильной даты и времени, а также внесения изменений в конфигурацию аппаратных средств компьютера в процессе выполнения самотестирования необходимо нажать клавишу {*Del*}. Загрузится системная утилита BIOS Setup, имеющая интерфейс в виде системы иерархических меню. Пользователь может установить новые параметры конфигурации компьютера и запомнить их в специальной микросхеме памяти, которая при выключенном компьютере питается от батарейки, установленной на системной плате. В случае выхода из строя батарейки конфигурационные параметры теряются и компьютер перестает нормально загружаться.

Загрузка операционной системы. После проведения самотестирования специальная программа, содержащаяся в BIOS, начинает поиск загрузчика операционной системы. Происходит поочередное обращение к имеющимся в компьютере дискам (гибким, жестким, CD-ROM) и поиск на определенном месте (в первом, так называемом *загрузочном секторе* диска) наличия специальной программы Master Boot (программы-загрузчика операционной системы).

Если диск системный и программа-загрузчик оказывается на месте, то она загружается в оперативную память и ей передается управление работой компьютера. Программа ищет файлы операционной системы на системном диске и загружает их в оперативную память в качестве программных модулей (рис. 1.20).

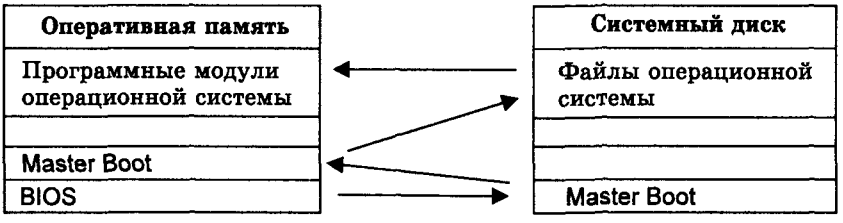


Рис. 1.20. Процесс загрузки операционной системы

Если системные диски в компьютере отсутствуют, на экране монитора появляется сообщение «*Non system disk*», и компьютер «зависает», то есть загрузка операционной системы прекращается и компьютер остается неработоспособным.

После окончания загрузки операционной системы управление передается командному процессору. В случае использования интерфейса командной строки на экране появляется приглашение системы к вводу команд. Приглашение

представляет собой последовательность символов, сообщающих о текущем диске и каталоге. Например, если загрузка операционной системы была произведена с диска C:, а операционная система была установлена в каталог WINDOWS, то появится приглашение:

C:\WINDOWS>

В случае загрузки графического интерфейса операционной системы команды могут вводиться с помощью мыши.

Вопросы для размышления

1. Каковы основные этапы самотестирования компьютера?
2. Что хранится в микросхеме конфигурационной памяти компьютера?
3. Каковы основные этапы загрузки операционной системы?

1.5. Графический интерфейс Windows

В настоящее время все операционные системы для персональных компьютеров обеспечивают взаимодействие с пользователем с помощью графического интерфейса.

Это позволяет даже начинающему пользователю компьютера уверенно работать в среде операционной системы (проводить операции с файлами, запускать программы и так далее).

Графический интерфейс позволяет осуществлять взаимодействие человека с компьютером в форме диалога с использованием окон, меню и элементов управления (диалоговых панелей, кнопок и так далее).

Работа с мышью. Для работы с графическим интерфейсом используется мышь или другое координатное устройство ввода, при этом пользователь должен уметь производить:

- *левый щелчок* — однократное нажатие и отпускание основной (обычно левой) кнопки мыши;

- *правый щелчок* — однократное нажатие и отпускание дополнительной (обычно правой) кнопки мыши;
- *двойной щелчок* — два нажатия основной кнопки мыши с минимальным интервалом времени между ними;
- *перетаскивание* — нажатие левой или правой кнопки мыши и перемещение объекта с нажатой кнопкой.

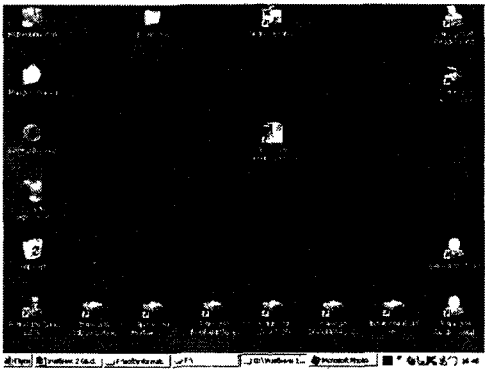
Рабочий стол. Основную часть экрана занимает *Рабочий стол*, на котором располагаются *значки* и *ярлыки* (значки с маленькими стрелочками в нижнем левом углу). Значки и ярлыки обеспечивают (с помощью двойного щелчка) быстрый доступ к дискам, папкам, документам, приложениям и устройствам.

Значки появляются на *Рабочем столе* после установки Windows. В левой части экрана обычно располагаются значки *Мой компьютер, Сетевое окружение, Корзина* и *Мои документы*.

Для быстрого доступа к дискам, принтеру, часто используемым документам целесообразно создать на рабочем столе ярлыки. Ярлык отличается от значка тем, что обозначает объект, фактически расположенный не на *Рабочем столе*, а в некоторой другой папке. Стрелочка означает, что мы имеем не сам объект, а ссылку на него. Ярлыки создаются перетаскиванием значков объектов на *Рабочий стол*.

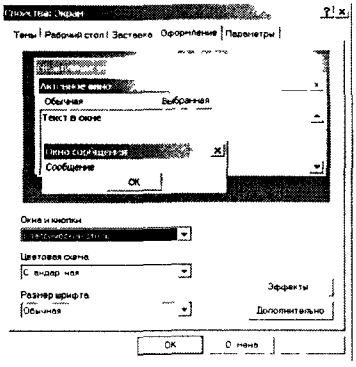
Знакомство с графическим интерфейсом Windows

1. Создать на *Рабочем столе* ярлыки всех дисков, принтера и сканера.
   Создать ярлыки часто используемых приложений и документов.



Внешний вид графического интерфейса можно настраивать.

2. Щелкнуть правой кнопкой мыши на *Рабочем столе*. В контекстном меню выбрать пункт *Свойства*.
На диалоговой панели *Свойства: Экран* на пяти вкладках установить стиль оформления, выбрать заставку и др.



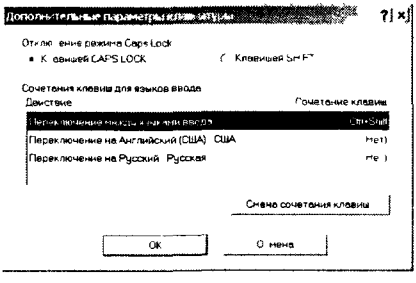
Панель задач. В нижней части экрана располагается *Панель задач*, на которой находятся кнопка *Пуск*, кнопки выполняемых задач и открытых папок, индикаторы и часы.

Кнопка *Пуск* позволяет вызывать *Главное меню*, которое обеспечивает доступ практически ко всем ресурсам системы и содержит команды запуска приложений, настройки системы, поиска файлов и документов, доступа к справочной системе и др.

Windows является *многозадачной* операционной системой, то есть параллельно могут выполняться несколько приложений. Каждое запущенное приложение обозначается кнопкой на *Панели задач*, при этом переход от работы в одном приложении к работе в другом может производиться с помощью щелчка по кнопке. Работающее (активное) приложение изображается на панели задач в виде нажатой кнопки.

В крайней правой части *Панели задач* находятся *Часы*. Левее часов располагаются индикаторы состояния системы. Например, индикатор *Ru* обозначает, что в текущий момент используется русская раскладка клавиатуры.

3. Левым щелчком мыши можно раскрыть индикатор и переключиться на английскую раскладку, а правым — открыть диалоговую панель *Свойства* и выбрать требуемое сочетание нажатия клавиш на клавиатуре для переключения раскладок.



Окна. Важнейшим элементом графического интерфейса Windows являются окна, действительно ведь «windows» в переводе означает «окна». Существуют два основных типа окон — *окна приложений* и *окна документов.*

Окна приложений. В окне приложения выполняется любое запущенное на выполнение приложение или отражается содержимое папки. Открыть или закрыть окно приложения — то же, что и запустить программу на выполнение или завершить ее. Окна приложений можно перемещать на любое место *Рабочего стола*, разворачивать на весь экран или сворачивать в кнопки на панели задач.

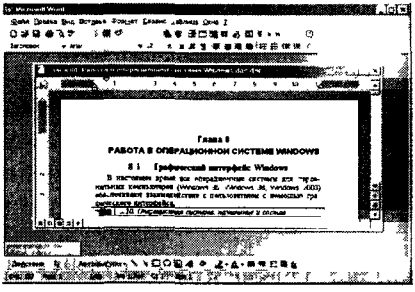
Основными элементами окна приложения являются:

- *рабочая область:* внутренняя часть окна, содержит вложенные папки или окна документов;
- *границы:* рамка, ограничивающая окно с четырех сторон. Размеры окна можно изменять, перемещая границу мышью;
- *заголовок:* строка непосредственно под верхней границей окна, содержащая название окна;
- *значок системного меню:* кнопка слева в строке заголовка открывает меню перемещения и изменения размеров окна;
- *строка горизонтального меню:* располагается непосредственно под заголовком, содержит пункты меню, обеспечивает доступ к командам;
- *панель инструментов:* располагается под строкой меню, представляет собой набор кнопок, обеспечивает быстрый доступ к некоторым командам;
- *кнопки Свернуть, Развернуть/Восстановить, Закрыть* расположены в верхней правой части окна.

Окна документов. Окна документов предназначены для работы с документами и «живут» внутри окон приложений. Можно раскрывать, сворачивать, перемещать или изменять размеры этих окон, однако они всегда остаются в пределах окна своего приложения. Окно документа имеет те же кнопки управления, что и окно приложения.

Окно документа всегда содержит зону заголовка (содержащую имя документа) и часто полосы прокрутки (появляющиеся, когда документ не помещается полностью в окне) и линейки. Открытое окно документа может находиться в *активном* либо в пассивном состоянии. Если окно находится в пассивном состоянии (зона заголовка не выделена цветом), то, щелкнув по любой его части мышью, можно перевести его в активное состояние.

4. После запуска приложения Word его окно появится на *Рабочем столе*. Если открыть в Word два документа, то в окне приложения появятся окна двух документов. Одно окно может быть развернуто и активно, другое — свернуто и пассивно.



Меню. Меню является одним из основных элементов графического интерфейса и представляет собой перечень команд (как правило, тематически сгруппированных), из которых необходимо сделать выбор (поместив на пункт меню указатель мыши и произведя щелчок). Выбор пункта меню приводит к выполнению определенной команды. Если за командой меню следует многоточие, то ее выбор приведет к появлению диалоговой панели, которая позволяет пользователю получить или ввести дополнительную информацию.

Диалоговые панели. Диалоговые панели могут включать в себя разнообразные элементы. Рассмотрим возможности диалоговых панелей на примере уточнения параметров поиска файлов.

Вкладки. Диалоговые панели могут включать в себя несколько «страниц», которые называются вкладками.

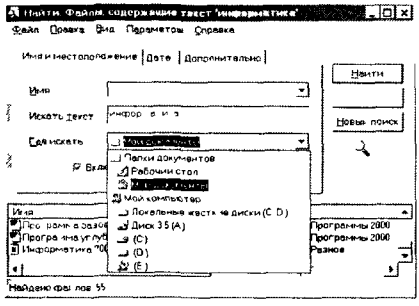
5. После ввода команды [Найти-Файлы и папки...] появится диалоговая панель *Найти: Все файлы*. Эта панель содержит три вкладки: *Имя и местоположение*, *Дата*, *Дополнительно*. Выбор вкладки осуществляется левым щелчком.

Командные кнопки. Нажатие на кнопку (щелчок) обеспечивает выполнение того или иного действия, а надпись на кнопке поясняет ее назначение. Так, щелчок по кнопке с надписью *Найти* позволяет начать процесс поиска.

Текстовые поля. Текстовое поле называется иногда *полем редактирования* и позволяет ввести какую-либо текстовую информацию.

6. Например, если пользователь хочет найти файлы, содержащие слово «информатика», то его необходимо ввести в текстовом поле *Искать текст:* вкладки *Имя и местоположение* диалоговой панели *Найти: Все файлы*.

Для этого следует осуществить левый щелчок в поле и ввести текст.

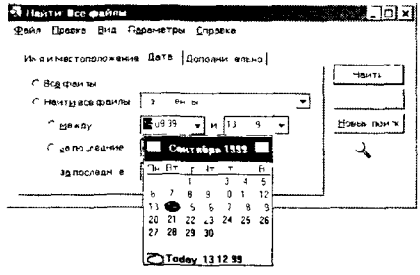


Списки. Список представляет собой набор предлагаемых на выбор значений. *Раскрывающийся список* выглядит как текстовое поле, снабженное кнопкой с направленной вниз стрелочкой. Раскрытие списка осуществляется с помощью левого щелчка по кнопке.

7. Раскрывающийся список *Где искать:* диалоговой панели *Найти: Все файлы* позволяет указать диск или папку (например, папку *Мои документы*), в которой будет осуществлен поиск.

Переключатели. Переключатели служат для выбора одного из взаимоисключающих вариантов, варианты выбора представлены в форме маленьких белых кружков. Выбранный вариант обозначается кружком с точкой внутри. Выбор варианта производится с помощью левого щелчка.

8. Так, на вкладке *Дата* диалоговой панели *Найти: Все файлы* имеются два переключателя: основной (на два варианта) и дополнительный (на три варианта). В процессе поиска файлов, установив основной переключатель в положение *Найти все файлы*, а дополнительный в положение *между*, можно ограничить область поиска периодом изменения файлов (в данном случае с 14.09.99 по 13.12.99).



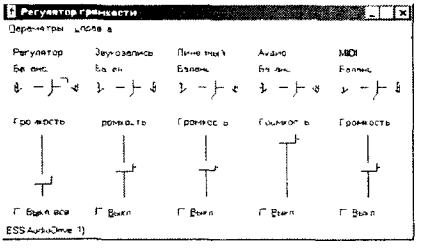
Флажки. Флажок обеспечивает присваивание какому-либо параметру определенного значения. Флажки могут располагаться как группами, так и поодиночке. Флажок имеет форму квадратика; когда флажок установлен, в нем присутствует «галочка». Установка флажков производится с помощью левого щелчка.

9. На вкладке *Имя и местоположение* диалоговой панели *Найти: Все файлы*, установив флажок *Включая вложенные папки*, можно обеспечить необходимую глубину поиска файлов.

Счетчики. Счетчик представляет собой пару стрелок, которые позволяют увеличивать или уменьшать значение в связанном с ними поле. Так, при поиске файла на вкладке *Дата* диалоговой панели *Найти: Все файлы* значения полей, задающих период изменения файла, можно менять с помощью счетчиков. Для увеличения соответствующего значения необходимо произвести щелчок по стрелке, направленной вправо, а для уменьшения — по стрелке, направленной влево.

Ползунки. Ползунок позволяет плавно изменять значение какого-либо параметра. Например, с помощью ползунков можно менять уровень громкости воспроизведения и записи звука, баланс левого и правого канала и т. п.

10. После двойного щелчка на индикаторе громкости, который находится на *Панели задач*, появится диалоговая панель *Регулятор громкости* с ползунками громкости и баланса каналов.



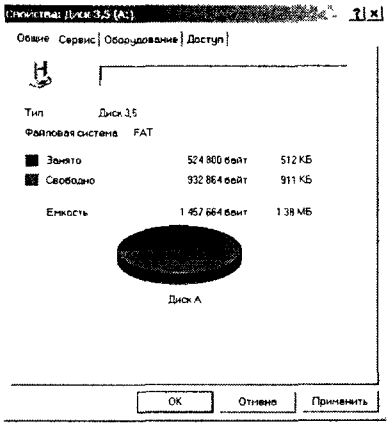
Контекстные меню. Объектно-ориентированный подход, используемый в операционной системе Windows, позволяет рассматривать диски, папки и файлы как объекты. Все эти объекты имеют определенные свойства, и над ними могут проводиться определенные операции.

Например, документы (документом называется любой файл, обрабатываемый с помощью приложений) имеют определенный объем и их можно копировать, перемещать и переименовывать; окна имеют размер, который можно изменять и так далее.

Хотя каждый из этих объектов имеет свои конкретные свойства и над ним возможны определенные операции, технология работы с объектами и интерфейс универсальны. Это позволяет пользователю достичь единообразия при работе с разными объектами.

Ознакомиться со свойствами объекта, а также выполнить над ним разрешенные операции можно с помощью *контекстного меню*. Для вызова контекстного меню необходимо осуществить правый щелчок на значке объекта.

11. Для того чтобы ознакомиться со свойствами диска, надо выбрать в контекстном меню пункт *Свойства* — появится диалоговая панель *Свойства: Диск 3,5 (A)*.
Панель содержит четыре вкладки: *Общие, Сервис, Оборудование, Доступ*.
На вкладке *Общие* содержится информация о типе файловой системы, общей, свободной и занятой информационной емкости диска и др.



Вопросы для размышления

1. Чем отличается окно документа от окна приложения?

2. Какие основные элементы могут содержать диалоговые панели?

Практические задания

1.9. Проверить правильность установки даты, времени и часового пояса на вашем компьютере.

1.10. Ознакомиться со свойствами папки и документа.

1.6. Программная обработка данных

Основной функцией компьютера является обработка информации. Выше была рассмотрена аппаратная реализация компьютера. Рассмотрим теперь, каким образом компьютер обрабатывает информацию.

В 50–60-е годы, когда компьютер еще назывался ЭВМ (электронно-вычислительная машина), он мог только вычислять. Процесс обработки информации состоял в операциях над числовыми данными.

В 70-е годы компьютер «научился» работать с текстом. Пользователь получил возможность редактировать и форматировать текстовые документы. В настоящее время бóльшая часть компьютеров и бóльшая часть времени используется для работы именно с текстовыми данными.

В 80-е годы появились первые компьютеры, способные работать с графической информацией. Сейчас компьютерная графика широко используется в деловой графике (построение диаграмм, графиков и так далее), в компьютерном моделировании, при подготовке презентаций, при создании Web-сайтов, в рекламе на телевидении, в анимационном кино и так далее. Применение компьютеров для обработки графических данных постоянно расширяется.

В 90-е годы компьютер получил возможность обрабатывать звуковую информацию. Любой пользователь современного персонального компьютера может воспользоваться стандартными приложениями Windows для прослушивания, записи и редактирования звуковых файлов. Работа со звуковыми данными является неотъемлемой частью мультимедиа технологии.

Для того чтобы числовая, текстовая, графическая и звуковая информация могли обрабатываться на компьютере, они должны быть представлены в форме данных. Данные хранятся и обрабатываются в компьютере на машинном языке, то есть в виде последовательностей нулей и единиц.

Информация, представленная в компьютерной форме (на машинном языке) и обрабатываемая на компьютере, называется **данными**.

Для того чтобы процессор компьютера «знал», что ему делать с данными, как их обрабатывать, он должен получить определенную команду (инструкцию). Такой командой может быть, например, «сложить два числа» или «заменить один символ на другой».

Обычно для решения какой-либо задачи процессору требуется не единичная команда, а их последовательность. Такая последовательность команд (инструкций) называется программой.

Последовательность команд, которую выполняет компьютер в процессе обработки данных, называется **программой**.

На заре компьютерной эры, в 40–50-е годы, программы разрабатывались непосредственно на машинном языке, то есть на том языке, который «понимает» процессор. Такие программы представляли собой очень длинные последовательности нулей и единиц, в которых человеку разобраться было очень трудно.

В 60-е годы началась разработка языков программирования высокого уровня (Алгол, Фортран, Basic, Pascal и др.), которые позволили существенно облегчить работу программистов. В настоящее время с появлением систем визуального программирования (Visual Basic, Delphi и др.) создание программ стало доступно даже для начинающих пользователей компьютера.

В течение нескольких десятилетий создавались программы, необходимые для обработки различных данных. Совокупность необходимых программ составляет *программное обеспечение компьютера*.

Таким образом, для обработки данных на компьютере необходимо иметь не только аппаратное обеспечение компьютера, так называемое hardware, но и программное обеспечение, так называемое software.

Программная обработка данных на компьютере реализуется следующим образом. После запуска на выполнение программы, хранящейся во внешней долговременной памяти, она загружается в оперативную память.

Процессор последовательно считывает команды программы и выполняет их. Необходимые для выполнения команды данные загружаются из внешней памяти в оперативную и над ними производятся необходимые операции. Данные, полученные в процессе выполнения команды, записываются процессором обратно в оперативную или внешнюю память.

В процессе выполнения программы процессор может запрашивать данные с устройств ввода информации и пересылать данные на устройства вывода информации.

4.8. Выполнение программ компьютером (интерпретаторы и компиляторы)

Вопросы для размышления

1. В чем состоит различие между данными и программами?
2. Где хранятся данные? Программы?

1.7. Файлы и файловая система

Все программы и данные хранятся в долговременной (внешней) памяти компьютера в виде файлов.

Файл – это определенное количество информации (программа или данные), имеющее имя и хранящееся в долговременной (внешней) памяти.

Имя файла. Имя файла состоит из двух частей, разделенных точкой: собственно имя файла и расширение, определяющее его тип (программа, данные и так далее). Собственно имя файлу дает пользователь, а тип файла обычно задается программой автоматически при его создании (табл. 1.2).

В различных операционных системах существуют различные форматы имен файлов. В операционной системе MS-DOS собственно имя файла должно содержать не более 8 букв латинского алфавита, цифр и некоторых специальных знаков, а расширение состоит из трех латинских букв, например: proba.txt

В операционной системе Windows имя файла может иметь длину до 255 символов, причем можно использовать русский алфавит, например: Единицы измерения информации.doc

Таблица 1.2. Типы файлов и расширений

Тип файла	Расширения
Программы	exe, com
Текстовые файлы	txt, doc
Графические файлы	bmp, gif, jpg и др
Звуковые файлы	wav, mid
Видеофайлы	avi
Программы на языках программирования	bas, pas и др

Файловая система. На каждом носителе информации (гибком, жестком или лазерном диске) может храниться большое количество файлов. Порядок хранения файлов на диске определяется используемой файловой системой.

Каждый диск разбивается на две области: область хранения файлов и каталог. Каталог содержит имя файла и указа-

ние на начало его размещения на диске. Если провести аналогию диска с книгой, то область хранения файлов соответствует ее содержанию, а каталог — оглавлению. Причем книга состоит из страниц, а диск — из секторов.

Для дисков с небольшим количеством файлов (до нескольких десятков) может использоваться *одноуровневая файловая система*, когда каталог (оглавление диска) представляет собой линейную последовательность имен файлов (табл. 1.3). Такой каталог можно сравнить с оглавлением детской книжки, которое содержит только названия отдельных рассказов.

Таблица 1.3. Одноуровневый каталог

Имя файла	Номер начального сектора
Файл_1	56
Файл_2	89
.	
Файл_112	1200

Если на диске хранятся сотни и тысячи файлов, то для удобства поиска используется *многоуровневая иерархическая файловая система*, которая имеет древовидную структуру. Такую иерархическую систему можно сравнить, например, с оглавлением данного учебника, которое представляет собой иерархическую систему разделов, глав, параграфов и пунктов.

Начальный, корневой каталог содержит вложенные каталоги 1-го уровня, в свою очередь, каждый из последних может содержать вложенные каталоги 2-го уровня и так далее. Необходимо отметить, что в каталогах всех уровней могут храниться и файлы.

Например, в корневом каталоге могут находиться два вложенных каталога 1-го уровня (Каталог_1, Каталог_2) и один файл (Файл_1). В свою очередь, в каталоге 1-го уровня (Каталог_1) находятся два вложенных каталога второго уровня (Каталог_1.1 и Каталог_1 2) и один файл (Файл_1.1) — рис. 1.21.

Файловая система — это система хранения файлов и организации каталогов.

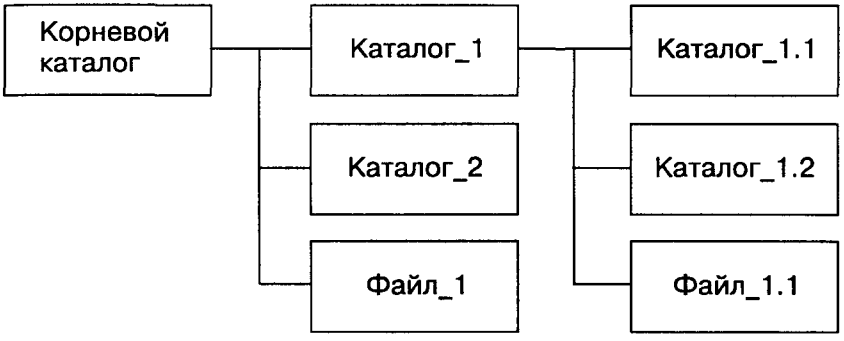


Рис. 1.21. Иерархическая файловая система

Рассмотрим иерархическую файловую систему на конкретном примере. Каждый диск имеет логическое имя (A:, B: — гибкие диски, C:, D:, E: и так далее — жесткие и лазерные диски).

Пусть в корневом каталоге диска C: имеются два каталога 1-го уровня (GAMES, TEXT), а в каталоге GAMES один каталог 2-го уровня (CHESS). При этом в каталоге TEXT имеется файл proba.txt, а в каталоге CHESS — файл chess.exe (рис. 1.22).

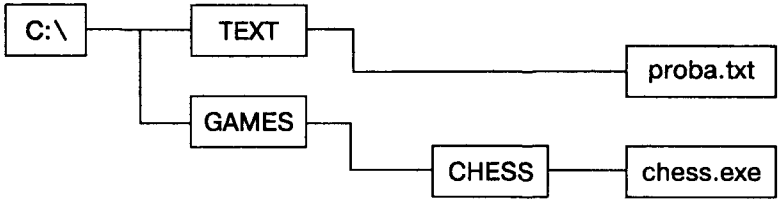


Рис. 1.22. Пример иерархической файловой системы

Путь к файлу. Как найти имеющиеся файлы (chess.exe, proba.txt) в данной иерархической файловой системе? Для этого необходимо указать путь к файлу. В путь к файлу входят записываемые через разделитель «\» логическое имя диска и последовательность имен вложенных друг в друга каталогов, в последнем из которых содержится нужный файл. Пути к вышеперечисленным файлам можно записать следующим образом:

C:\GAMES\CHESS\

C:\TEXT\

Путь к файлу вместе с именем файла называют иногда *полным именем файла*.

Пример полного имени файла:

C \GAMES\CHESS\chess.exe

Представление файловой системы с помощью графического интерфейса. Иерархическая файловая система MS-DOS, содержащая каталоги и файлы, представлена в операционной системе Windows с помощью графического интерфейса в форме иерархической системы папок и документов. Папка в Windows является аналогом каталога MS-DOS

Однако иерархическая структура этих систем несколько различается. В иерархической файловой системе MS-DOS вершиной иерархии объектов является корневой каталог диска, который можно сравнить со стволом дерева, на котором растут ветки (подкаталоги), а на ветках располагаются листья (файлы).

В Windows на вершине иерархии папок находится папка *Рабочий стол*. Следующий уровень представлен папками *Мой компьютер, Корзина* и *Сетевое окружение* (если компьютер подключен к локальной сети) — рис. 1.23.

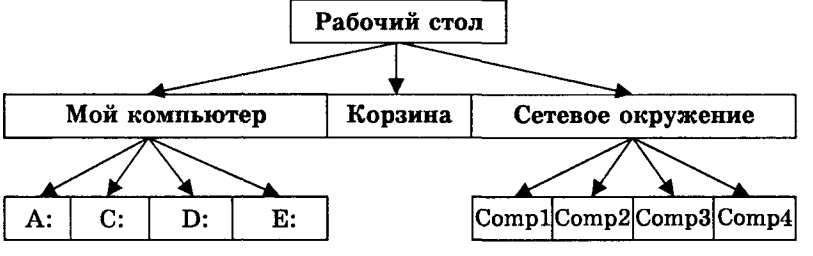
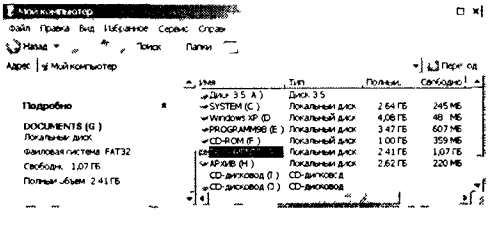


Рис. 1.23. Иерархическая структура папок

Если мы хотим ознакомиться с ресурсами компьютера, необходимо открыть папку *Мой компьютер*.

Иерархическая система папок Windows

1. В окне *Мой компьютер* находятся значки имеющихся в компьютере дисков. Активизация (щелчок) значка любого диска выводит в левой части окна информацию о его емкости, занятой и свободной частях.



2. Выбрав один из пунктов меню *Вид* (*Крупные значки, Мелкие значки, Список, Таблица*), можно настроить форму представления содержимого папки.

Папка *Сетевое окружение* содержит папки всех компьютеров, подключенных в данный момент к локальной сети.

Папка *Корзина* временно содержит все удаленные папки и файлы. При необходимости удаленные и хранящиеся в *Корзине* папки и документы можно восстановить.

3. Для окончательного удаления файлов необходимо ввести команду [Файл-Очистить корзину].

Операции над файлами. В процессе работы на компьютере наиболее часто над файлами производятся следующие операции:

- копирование (копия файла помещается в другой каталог);
- перемещение (сам файл перемещается в другой каталог);
- удаление (запись о файле удаляется из каталога);
- переименование (изменяется имя файла).

Графический интерфейс Windows позволяет проводить операции над файлами с помощью мыши с использованием метода Drag&Drop (перетащи и оставь). Существуют также специализированные приложения для работы с файлами, так называемые *файловые менеджеры*: Norton Commander, Windows Commander, Проводник и др.

Установить файловый менеджер Windows Commander CD-ROM

В некоторых случаях возникает необходимость работать с интерфейсом командной строки. В Windows предусмотрен режим работы с интерфейсом командной строки MS-DOS.

Интерфейс командной строки

1. Ввести команду [Программы-Сеанс MS-DOS]. Появится окно приложения *Сеанс MS-DOS*.

В ответ на приглашение системы можно вводить команды MS-DOS с клавиатуры, в том числе:

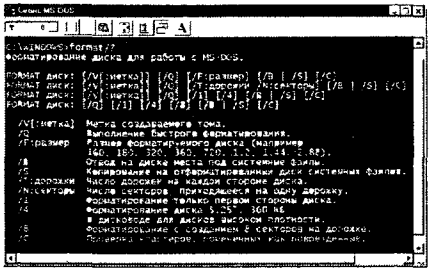
- команды работы с файлами (copy, del, rename и др.);
- команды работы с каталогами (dir, mkdir, chdir и др.);
- команды работы с дисками (format, defrag и др.).

2. Существуют десятки команд MS-DOS, при этом каждая команда имеет свой формат и параметры, запомнить которые достаточно трудно. Для того чтобы получить спра-

вочную информацию по команде, необходимо после имени команды ввести ключ /?.

Например, для получения справки по команде format в ответ на приглашение системы необходимо ввести:

C:\WINDOWS>format/?



Вопросы для размышления

1. Какой элемент является вершиной иерархии в файловой системе MS-DOS? В графическом интерфейсе Windows?

Практические задания

1.11. Осуществить копирование файлов с использованием интерфейса командной строки и файлового менеджера.

1.12. Ознакомиться с объемом дисков вашего компьютера, а также объемами занятого и свободного пространства.

1.13. Ознакомиться с форматом команды dir. Просмотреть корневой каталог диска C .

1.8. Логическая структура дисков

Форматирование дисков. Для того чтобы на диске можно было хранить информацию, диск должен быть отформатирован, то есть должна быть создана физическая и логическая структура диска.

Формирование физической структуры диска состоит в создании на диске концентрических дорожек, которые, в свою очередь, делятся на секторы. Для этого в процессе форматирования магнитная головка дисковода расставляет в определенных местах диска метки дорожек и секторов.

После форматирования гибкого диска 3,5" его параметры будут следующими (рис. 1.24):

- информационная емкость сектора — 512 байтов;
- количество секторов на дорожке — 18;
- дорожек на одной стороне — 80;
- сторон — 2.

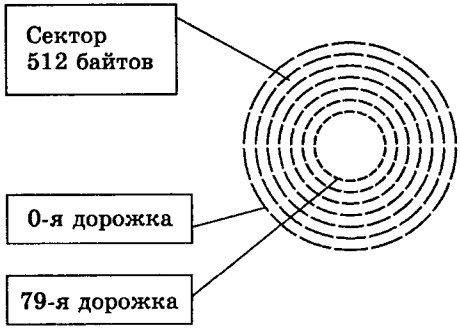


Рис. 1.24. Физическая структура дискеты

Логическая структура гибких дисков. Логическая структура магнитного диска представляет собой совокупность секторов (емкостью 512 байтов), каждый из которых имеет свой порядковый номер (например, 100). Сектора нумеруются в линейной последовательности от первого сектора нулевой дорожки до последнего сектора последней дорожки.

На гибком диске минимальным адресуемым элементом является **сектор**.

При записи файла на диск будет занято всегда целое количество секторов, соответственно минимальный размер файла — это размер одного сектора, а максимальный соответствует общему количеству секторов на диске.

Файл записывается в произвольные свободные сектора, которые могут находиться на различных дорожках. Например, Файл_1 объемом 2 Кбайта может занимать сектора 34, 35 и 47, 48, а Файл_2 объемом 1 Кбайт — сектора 36 и 49.

Таблица 1.4. Логическая структура гибкого диска формата 3,5" (2-я сторона)

№ дорожки	№ сектора																	
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
2	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54
								.....										
79																		2880

Для того чтобы можно было найти файл по его имени, на диске имеется каталог, представляющий собой *базу данных*.

11.1. Базы данных

Запись о файле содержит имя файла, адрес первого сектора, с которого начинается файл, объем файла, а также дату и время его создания (табл. 1.5).

Таблица 1.5. Структура записей в каталоге

Имя файла	Адрес первого сектора	Объем файла, Кбайт	Дата создания	Время создания
Файл_1	34	2	14 01 99	14 29
Файл_2	36	1	14 01 99	14 45

Полная информация о секторах, которые занимают файлы, содержится в таблице размещения файлов (FAT — File Allocation Table). Количество ячеек FAT соответствует количеству секторов на диске, а значениями ячеек являются цепочки размещения файлов, то есть последовательности адресов секторов, в которых хранятся файлы.

Например, для двух рассмотренных выше файлов таблица FAT с 1 по 54 сектор принимает вид, представленный в табл. 1.6.

Таблица 1.6. Фрагмент FAT

														35	47	49
									48	К	К					

Цепочка размещения для файла Файл_1 выглядит следующим образом: в начальном 34-м секторе хранится адрес 35, в 35-м секторе хранится адрес 47, в 47-м — 48, в 48-м — знак конца файла (К).

Для размещения каталога — базы данных и таблицы FAT на гибком диске отводятся секторы со 2 по 33. Первый сектор отводится для размещения загрузочной записи операционной системы. Сами файлы могут быть записаны, начиная с 34 сектора.

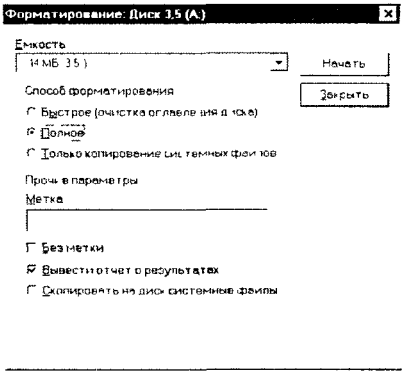
Виды форматирования. Существуют два различных вида форматирования дисков: полное и быстрое форматирование. Полное форматирование включает в себя как физическое форматирование (проверку качества магнитного покрытия

дискеты и ее разметку на дорожки и секторы), так и логическое форматирование (создание каталога и таблицы размещения файлов). После полного форматирования вся хранившаяся на диске информация будет уничтожена.

Быстрое форматирование производит лишь очистку корневого каталога и таблицы размещения файлов. Информация, то есть сами файлы, сохраняется и в принципе возможно восстановление файловой системы.

Стандартное форматирование гибкого диска

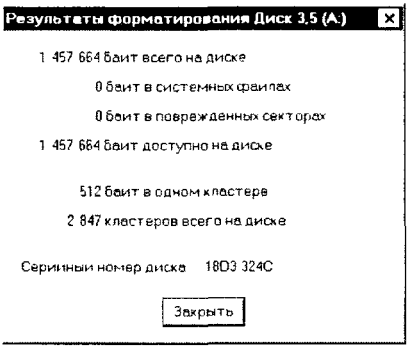
1. В контекстном меню выбрать пункт *Форматировать*. Откроется диалоговая панель *Форматирование*. С помощью переключателя *Способ форматирования* выбрать пункт *Полное*.

   В поле *Метка* можно ввести название диска. Для получения сведения о результатах форматирования установить флажок *Вывести отчет о результатах*.

   Щелкнуть по кнопке *Начать*.

2. После окончания форматирования диска появится информационная панель *Результаты форматирования*.

   Вы увидите, что доступный для размещения данных информационный объем диска оказался равен 1 459 664 байта (2047 секторов), а системные файлы и поврежденные сектора отсутствуют.

В целях защиты информации от несанкционированного копирования можно задавать нестандартные параметры форматирования диска (количество дорожек, количество секторов и др.). Такое форматирование возможно в режиме MS-DOS.

Нестандартное форматирование гибкого диска

1. Ввести команду [Программы-Сеанс MS-DOS]. Появится окно приложения *Сеанс MS-DOS*.
2. Ввести команду нестандартного форматирования гибкого диска A:, на котором будет 79 дорожек и 19 секторов на каждой дорожке:

   C:\WINDOWS>format A:/T:79/N:19

Информационная емкость гибких дисков. Рассмотрим различие между емкостью неформатированного гибкого магнитного диска, его информационной емкостью после форматирования и информационной емкостью, доступной для записи данных.

Заявленная емкость неформатированного гибкого магнитного диска формата 3,5" составляет 1,44 Мбайт.

Рассчитаем общую информационную емкость отформатированного гибкого диска:

Количество секторов: $N = 18 \times 80 \times 2 = 2880$.

Информационная емкость:

512 байт $\times$ N = 1 474 560 байт = 1 440 Кбайт = = 1,40625 Мбайт.

Однако для записи данных доступно только 2847 секторов, то есть информационная емкость, доступная для записи данных, составляет:

512 байт $\times$ 2847 = 1 457 664 байт = 1423,5 Кбайт $\approx$ $\approx$ 1,39 Мбайт.

Логическая структура жестких дисков. Логическая структура жестких дисков несколько отличается от логической структуры гибких дисков. Минимальным адресуемым элементом жесткого диска является *кластер*, который может включать в себя несколько секторов. Размер кластера зависит от типа используемой таблицы FAT и от емкости жесткого диска.

На жестком диске минимальным адресуемым элементом является **кластер**, который содержит несколько секторов.

Таблица FAT16 может адресовать 2^{16} = 65 536 кластеров. Для дисков большой емкости размер кластера оказывается слишком большим, таккак информационная емкость жестких дисков может достигать 150 Гбайт.

Например, для диска объемом 40 Гбайт размер кластера будет равен:

40 Гбайт/65536 = 655 360 байт = 640 Кбайт.

Файлу всегда выделяется целое число кластеров. Например, текстовый файл, содержащий слово «информатика», составляет всего 11 байтов, но на диске этот файл будет занимать целиком кластер, то есть 640 Кбайт дискового пространства для диска емкостью 150 Гбайт. При размещении на жестком диске большого количества небольших по размеру файлов они будут занимать кластеры лишь частично, что приведет к большим потерям свободного дискового пространства.

Эта проблема частично решается с помощью использования таблицы FAT32, в которой объем кластера принят равным 8 секторам или 4 килобайтам для диска любого объема.

В целях более надежного сохранения информации о размещении файлов на диске хранятся две идентичные копии таблицы FAT.

Преобразование FAT16 в FAT32 можно осуществить с помощью служебной программы Преобразование диска в FAT32, которая входит в состав Windows.

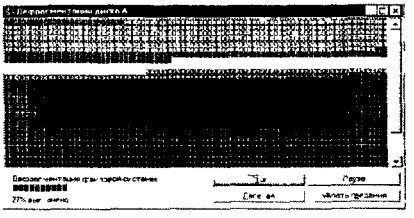
Дефрагментация дисков. Замедление скорости обмена данными может происходить в результате *фрагментации* файлов. Фрагментация файлов (фрагменты файлов хранятся в различных, удаленных друг от друга кластерах) возрастает с течением времени, в процессе удаления одних файлов и записи других.

Так как на диске могут храниться сотни и тысячи файлов в сотнях тысяч кластеров, то фрагментированность файлов будет существенно замедлять доступ к ним (магнитным головкам придется постоянно перемещаться с дорожки на дорожку) и в конечном итоге приводить к преждевременному износу жесткого диска. Рекомендуется периодически проводить дефрагментацию диска, в процессе которой файлы записываются в кластеры, последовательно идущие друг за другом.

Дефрагментация диска

1. Для запуска программы Дефрагментация диска, необходимо из *Главного меню* ввести команду [Стандартные-Служебные-Дефрагментация диска].
2. Диалоговая панель *Выбор диска* позволяет выбрать диск, нуждающийся в процедуре дефрагментации. После нажатия кнопки *OK* появится панель *Дефрагментация диска*.

3. Процесс дефрагментации диска можно визуально наблюдать, если щелкнуть по кнопке *Сведения*. Каждый квадратик соответствует одному кластеру, при этом *неоптимизированные*, уже *оптимизированные*, а также *считываемые* и *записываемые* в данный момент кластеры имеют различные цвета.



Вопросы для размышления

1. Какой минимальный объем занимает файл при его хранении:
   - на гибком магнитном диске;
   - на жестком магнитном диске.
2. Какова последовательность размещения файла Файл_2 из приведенного примера на секторах гибкого диска?
3. Почему различаются величины емкости отформатированного диска и информационной емкости, доступной для записи данных?
4. Чем различаются полное и быстрое форматирование диска?
5. Чем различаются таблицы размещения файлов FAT16 и FAT32?
6. С какой целью необходимо периодически проводить дефрагментацию жестких дисков?

Практические задания

1.14. Отформатировать гибкий диск с нестандартными параметрами.

1.15. Вычислить объем кластера вашего жесткого диска в системе FAT16.

1.16. С помощью служебной программы Сведения о системе определить тип FAT, используемый на ваших дисках.

1.17. С помощью служебной программы Проверка диска провести проверку целостности файловой системы.

1.18. С помощью служебной программы Дефрагментация диска провести дефрагментацию дисков вашего компьютера.

1.9. Прикладное программное обеспечение

Прикладное программное обеспечение можно разделить на две группы программ: *системы программирования* и *приложения.*

Системы программирования являются инструментами для программистов-профессионалов и позволяют разрабатывать программы на различных языках программирования (Basic, Pascal, C и др.). В настоящее время широкое распространение получили системы визуального программирования (Visual Basic, Borland Delphi и др.), которые позволяют даже начинающему пользователю компьютера создавать несложные программы.

Приложение функционирует под управлением определенной операционной системы. Так, текстовый редактор Word является приложением операционной системы Windows, а текстовый редактор Edit — приложением операционной системы MS-DOS. Приложения позволяют пользователю обрабатывать текстовую, графическую, числовую, аудио- и видеоинформацию, а также работать в компьютерных сетях, не владея программированием.

Практически каждый пользователь компьютера нуждается в *приложениях общего назначения*, к числу которых относятся: текстовые редакторы и графические редакторы, электронные таблицы, системы управления базами данных, а также приложения для создания мультимедиа-презентаций. Наиболее распространенными в настоящее время пакетами приложений общего назначения являются Microsoft Office и StarOffice.

В связи со стремительным развитием глобальных и локальных компьютерных сетей все большее значение приобретают различные *коммуникационные программы.* В последнее время разработчики операционных систем, и в частности разработчики Windows, включают коммуникационные программы непосредственно в состав операционной системы.

В отдельную группу в связи с широким распространением компьютерных вирусов можно отнести *антивирусные программы.*

Для профессионального использования в различных сферах деятельности квалифицированными пользователями компьютера используются *приложения специального назна-*

чения. К ним относятся системы компьютерной графики, системы автоматизированного проектирования (САПР), бухгалтерские программы, компьютерные словари, системы автоматического перевода и др.

Все большее число пользователей использует *обучающие программы* для самообразования или в учебном процессе. Прежде всего, это программы обучения иностранным языкам, программы-репетиторы, тесты по различным предметам и так далее.

Большую пользу приносят различные *мультимедиа-приложения* (энциклопедии, справочники и так далее) на лазерных дисках, которые содержат огромный объем информации и средства быстрого ее поиска.

Достаточно большое число пользователей начинают знакомство с компьютером с *компьютерных игр*, которые бывают самых различных типов: логические, стратегические, спортивные и так далее.

Вопросы для размышления

1. В чем состоит основное различие между операционной системой и прикладными программами?
2. Какие вы знаете виды приложений общего назначения?
3. Какие вы знаете виды приложений специального назначения?

1.10. Компьютерные вирусы и антивирусные программы

1.10.1. Типы компьютерных вирусов

Первая массовая эпидемия компьютерного вируса произошла в 1986 году, когда вирус Brain «заражал» дискеты для первых массовых персональных компьютеров. В настоящее время известно несколько десятков тысяч вирусов, заражающих компьютеры с различными операционными системами и распространяющихся по компьютерным сетям.

Обязательным свойством компьютерного вируса является способность к размножению (самокопированию) и незаметному для пользователя внедрению в файлы, загрузочные

секторы дисков и документы. Название «вирус» по отношению к компьютерным программам пришло из биологии именно по признаку способности к саморазмножению.

После заражения компьютера вирус может активизироваться и заставить компьютер выполнять какие-либо действия. Активизация вируса может быть связана с различными событиями (наступлением определенной даты или дня недели, запуском программы, открытием документа и так далее).

Компьютерные вирусы являются программами, которые могут «размножаться» и скрытно внедрять свои копии в файлы, загрузочные секторы дисков и документы. Активизация компьютерного вируса может вызывать уничтожение программ и данных.

Разнообразны последствия действия вирусов; по величине вредных воздействий вирусы можно разделить на:

- неопасные, влияние которых ограничивается уменьшением свободной памяти на диске, графическими, звуковыми и другими внешними эффектами;
- опасные, которые могут привести к сбоям и зависаниям при работе компьютера;
- очень опасные, активизация которых может привести к потере программ и данных (изменению или удалению файлов и каталогов), форматированию винчестера и так далее.

По «среде обитания» вирусы можно разделить на *файловые, загрузочные, макровирусы* и *сетевые*.

Файловые вирусы. Файловые вирусы различными способами внедряются в исполнимые файлы (программы) и обычно активизируются при их запуске. После запуска зараженной программы вирус находится в оперативной памяти компьютера и является активным (то есть может заражать другие файлы) вплоть до момента выключения компьютера или перезагрузки операционной системы.

При этом файловые вирусы не могут заразить файлы данных (например, файлы, содержащие изображение или звук).

Профилактическая защита от файловых вирусов состоит в том, что не рекомендуется запускать на выполнение файлы, полученные из сомнительного источника и предварительно не проверенные антивирусными программами.

Загрузочные вирусы. Загрузочные вирусы записывают себя в загрузочный сектор диска. При загрузке операционной системы с зараженного диска вирусы внедряются в оперативную память компьютера. В дальнейшем загрузочный вирус ведет себя так же, как файловый, то есть может заражать файлы при обращении к ним компьютера.

Профилактическая защита от таких вирусов состоит в отказе от загрузки операционной системы с гибких дисков и установке в BIOS вашего компьютера защиты загрузочного сектора от изменений.

Макровирусы. Макровирусы заражают файлы документов Word и электронных таблиц Excel. Макровирусы являются фактически макрокомандами (макросами), которые встраиваются в документ.

4.15. Язык объектно-ориентированного программирования Visual Basic for Applications

После загрузки зараженного документа в приложение макровирусы постоянно присутствуют в памяти компьютера и могут заражать другие документы. Угроза заражения прекращается только после закрытия приложения.

Профилактическая защита от макровирусов состоит в предотвращении запуска вируса. При открытии документа в приложениях Word и Excel сообщается о присутствии в них макросов (потенциальных вирусов) и предлагается запретить их загрузку. Выбор запрета на загрузку макросов надежно защитит ваш компьютер от заражения макровирусами, однако отключит и полезные макросы, содержащиеся в документе.

Сетевые вирусы. По компьютерной сети могут распространяться и заражать компьютеры любые обычные вирусы. Это может происходить, например, при получении зараженных файлов с серверов файловых архивов. Однако существуют и специфические сетевые вирусы, которые используют для своего распространения электронную почту и Всемирную паутину.

Интернет-черви (worm) — это вирусы, которые распространяются в компьютерной сети во вложенных в почтовое сообщение файлах. Автоматическая активизация червя и заражение компьютера могут произойти при обычном просмотре сообщения. Опасность таких вирусов состоит в том, что они по определенным датам активизируются и уничтожают файлы на дисках зараженного компьютера.

Кроме того, интернет-черви часто являются *троянами*, выполняя роль «троянского коня», внедренного в операционную систему. Такие вирусы «похищают» идентификатор и пароль пользователя для доступа в Интернет и передают их на определенный почтовый адрес. В результате злоумышленники получают возможность доступа в Интернет за деньги ничего не подозревающих пользователей.

Лавинообразная цепная реакция распространения вируса базируется на том, что вирус после заражения компьютера начинает рассылать себя по всем адресам электронной почты, которые имеются в адресной книге пользователя. Кроме того, может происходить заражение и по локальной сети, так как червь перебирает все локальные диски и сетевые диски с правом доступа и копируется туда под случайным именем.

Профилактическая защита от интернет-червей состоит в том, что не рекомендуется открывать вложенные в почтовые сообщения файлы, полученные из сомнительных источников.

Особой разновидностью вирусов являются активные элементы (программы) на языках JavaScript или VBScript, которые могут выполнять разрушительные действия, то есть являться вирусами (*скрипт-вирусами*). Такие программы передаются по Всемирной паутине в процессе загрузки Web-страниц с серверов Интернета в браузер локального компьютера.

Профилактическая защита от скрипт-вирусов состоит в том, что в браузере можно запретить получение активных элементов на локальный компьютер.

Вопросы для размышления

1. К каким последствиям может привести заражение компьютерными вирусами?
2. Какие типы компьютерных вирусов существуют, чем они отличаются друг от друга и какова должна быть профилактика заражения?
3. Почему даже чистая отформатированная дискета может стать источником заражения вирусом?

1.10.2. Антивирусные программы

Наиболее эффективны в борьбе с компьютерными вирусами антивирусные программы. Антивирусные программы могут использовать различные принципы для поиска и лечения зараженных файлов.

Установить полифаги
Kaspersky Anti-Virus, Dr.Web CD-ROM 

Полифаги. Самыми популярными и эффективными антивирусными программами являются антивирусные программы *полифаги* (например, Kaspersky Anti-Virus, Dr.Web). Принцип работы полифагов основан на проверке файлов, загрузочных секторов дисков и оперативной памяти и поиске в них известных и новых (неизвестных полифагу) вирусов.

Для поиска известных вирусов используются так называемые маски. Маской вируса является некоторая постоянная последовательность программного кода, специфичная для этого конкретного вируса. Если антивирусная программа обнаруживает такую последовательность в каком-либо файле, то файл считается зараженным вирусом и подлежит лечению.

Для поиска новых вирусов используются алгоритмы «эвристического сканирования», то есть анализ последовательности команд в проверяемом объекте. Если «подозрительная» последовательность команд обнаруживается, то полифаг выдает сообщение о возможном заражении объекта.

Полифаги могут обеспечивать проверку файлов в процессе их загрузки в оперативную память. Такие программы называются антивирусными *мониторами*.

К достоинствам полифагов относится их универсальность. К недостаткам можно отнести большие размеры используемых ими антивирусных баз данных, которые должны содержать информацию о максимально возможном количестве вирусов, что, в свою очередь, приводит к относительно небольшой скорости поиска вирусов.

Установить ревизор ADinf CD-ROM 

Ревизоры. Принцип работы ревизоров (например, ADinf) основан на подсчете контрольных сумм для присутствующих на диске файлов. Эти контрольные суммы затем сохраняются в базе данных антивируса, как и некоторая другая информация: длины файлов, даты их последней модификации и пр.

При последующем запуске ревизоры сверяют данные, содержащиеся в базе данных, с реально подсчитанными значениями. Если информация о файле, записанная в базе данных, не совпадает с реальными значениями, то ревизоры сигнализируют о том, что файл был изменен или заражен вирусом.

Недостаток ревизоров состоит в том, что они не могут обнаружить вирус в новых файлах (на дискетах, при распаковке файлов из архива, в электронной почте), поскольку в их базах данных отсутствует информация об этих файлах.

Блокировщики. Антивирусные *блокировщики* — это программы, перехватывающие «вирусоопасные» ситуации и сообщающие об этом пользователю. К таким ситуациям относится, например, запись в загрузочный сектор диска. Эта запись происходит при установке на компьютер новой операционной системы или при заражении загрузочным вирусом.

Наибольшее распространение получили антивирусные блокировщики в BIOS компьютера. С помощью программы BIOS Setup можно провести настройку BIOS таким образом, что будет запрещена (заблокирована) любая запись в загрузочный сектор диска и компьютер будет защищен от заражения загрузочными вирусами.

К достоинствам блокировщиков относится их способность обнаруживать и останавливать вирус на самой ранней стадии его размножения.

Практические задания

1.19. Проверить компьютер на заражение вирусами с помощью антивирусных программ-полифагов.

1.20. Проверить компьютер на заражение вирусами с помощью антивирусной программы-ревизора.

1.21. Проверить, установлена ли в BIOS Setup антивирусная защита.

Глава 2

Информация. Двоичное кодирование информации

2.1. Понятие «информация» и свойства информации

Понятие «информация». Слово «информация» происходит от латинского слова informatio, что в переводе означает сведение, разъяснение, ознакомление. Понятие «информация» является базовым в курсе информатики, невозможно дать его определение через другие, более «простые» понятия. В геометрии, например, невозможно выразить содержание базовых понятий «точка», «луч», «плоскость» через более простые понятия. Содержание основных, базовых понятий в любой науке должно быть пояснено на примерах или выявлено путем их сопоставления с содержанием других понятий.

В случае с понятием «информация» проблема его определения еще более сложная, так как оно является общенаучным понятием. Данное понятие используется в различных науках (информатике, кибернетике, биологии, физике и др.), при этом в каждой науке понятие «информация» связано с различными системами понятий.

Информация в физике. В физике мерой беспорядка, хаоса для термодинамической системы является энтропия системы, тогда как информация (антиэнтропия) является мерой упорядоченности и сложности системы. По мере увеличения сложности системы величина энтропии уменьшается, и величина информации увеличивается. Процесс увеличения информации характерен для открытых, обменивающихся веществом и энергией с окружающей средой, саморазвивающихся систем живой природы (белковых молекул, организмов, популяций животных и так далее).

Таким образом, в физике информация рассматривается как антиэнтропия или энтропия с обратным знаком.

Информация в биологии. В биологии, которая изучает живую природу, понятие «информация» связывается с целесообразным поведением живых организмов. Такое поведение строится на основе получения и использования организмом информации об окружающей среде.

Понятие «информация» в биологии используется также в связи с исследованиями механизмов наследственности. Генетическая информация передается по наследству и хранится во всех клетках живых организмов. Гены представляют собой сложные молекулярные структуры, содержащие информацию о строении живых организмов. Последнее обстоятельство позволило проводить научные эксперименты по клонированию, то еeть созданию точных копий организмов из одной клетки.

Информация в кибернетике. В кибернетике (науке об управлении) понятие «информация» связано с процессами управления в сложных системах (живых организмах или технических устройствах). Жизнедеятельность любого организма или нормальное функционирование технического устройства зависит от процессов управления, благодаря которым поддерживаются в необходимых пределах значения их параметров. Процессы управления включают в себя получение, хранение, преобразование и передачу информации.

Социально значимые свойства информации. Человек — существо социальное, для общения с другими людьми он должен обмениваться с ними информацией, причем обмен информацией всегда производится на определенном языке — русском, английском и так далее. Участники дискуссии должны владеть тем языком, на котором ведется общение, тогда информация будет *понятной* всем участникам обмена информацией.

Информация должна быть *полезной*, тогда дискуссия приобретает практическую ценность. Бесполезная информация создает информационный шум, который затрудняет восприятие полезной информации. Примерами передачи и получения бесполезной информации могут служить некоторые конференции и чаты в Интернете.

Широко известен термин «средства массовой информации» (газеты, радио, телевидение), которые доводят информацию до каждого члена общества. Такая информация должна быть *достоверной* и *актуальной*. Недостоверная информация вводит членов общества в заблуждение и может быть причиной возникновения социальных потрясе-

ний. Неактуальная информация бесполезна и поэтому никто, кроме историков, не читает прошлогодних газет.

Для того чтобы человек мог правильно ориентироваться в окружающем мире, информация должна быть *полной* и *точной*. Задача получения полной и точной информации стоит перед наукой. Овладение научными знаниями в процессе обучения позволяют человеку получить полную и точную информацию о природе, обществе и технике.

Вопросы для размышления

1. Почему невозможно дать определение понятию «информация», используя более «простые» понятия?
2. В каких науках используется понятие «информация» и какой смысл в каждой из них оно имеет?
3. Какие социально значимые свойства информации можно выделить?

2.2. Количество информации как мера уменьшения неопределенности знаний

Информация и знания. Человек получает информацию из окружающего мира с помощью органов чувств, анализирует ее и выявляет существенные закономерности с помощью мышления, хранит полученную информацию в памяти. Процесс систематического научного познания окружающего мира приводит к накоплению информации в форме знаний (фактов, научных теорий и так далее). Таким образом, с точки зрения процесса познания информация может рассматриваться как *знания*.

Процесс познания можно наглядно изобразить в виде расширяющегося круга знания (такой способ придумали еще древние греки). Вне этого круга лежит область незнания, а окружность является границей между знанием и незнанием. Парадокс состоит в том, что чем бóльшим объемом знаний обладает человек (чем шире круг знаний), тем больше он ощущает недостаток знаний (тем больше граница нашего незнания, мерой которого в этой модели является длина окружности) — рис. 2.1.

Рис. 2.1
Знание и незнание

Так, объем знаний выпускника школы гораздо больше, чем объем знаний первоклассника, однако и граница его незнания существенно больше. Действительно, первоклассник ничего не знает о законах физики и поэтому не осознает недостаточности своих знаний, тогда как выпускник школы при подготовке к экзаменам по физике может обнаружить, что существуют физические законы, которые он не знает или не понимает.

Информацию, которую получает человек, можно считать мерой уменьшения неопределенности знаний. Если некоторое сообщение приводит к уменьшению неопределенности наших знаний, то можно говорить, что такое сообщение содержит информацию.

Например, после сдачи экзамена по информатике вы мучаетесь неопределенностью, вы не знаете какую оценку получили. Наконец, экзаменационная комиссия объявляет результаты экзамена, и вы получаете сообщение, которое приносит полную определенность, теперь вы знаете свою оценку. Происходит переход от незнания к полному знанию, значит, сообщение экзаменационной комиссии содержит информацию.

Уменьшение неопределенности знаний. Подход к информации как мере уменьшения неопределенности знаний позволяет количественно измерять информацию, что чрезвычайно важно для информатики. Рассмотрим вопрос об определении количества информации более подробно на конкретных примерах.

Пусть у нас имеется монета, которую мы бросаем на ровную поверхность. С равной вероятностью произойдет одно из двух возможных событий — монета окажется в одном из двух положений: «орел» или «решка».

Можно говорить, что события равновероятны, если при возрастающем числе опытов количества выпадений «орла» и «решки» постепенно сближаются. Например, если мы бросим монету 10 раз, то «орел» может выпасть 7 раз, а решка — 3 раза, если бросим монету 100 раз, то «орел» может выпасть 60 раз, а «решка» — 40 раз, если бросим монету 1000 раз, то «орел» может выпасть 520 раз, а «решка» — 480 и так далее.

В итоге при очень большой серии опытов количества выпадений «орла» и «решки» практически сравняются.

Перед броском существует неопределенность наших знаний (возможны два события), и, как упадет монета, предсказать невозможно. После броска наступает полная определенность, так как мы видим (получаем зрительное сообщение), что монета в данный момент находится в определенном положении (например, «орел»). Это сообщение приводит к уменьшению неопределенности наших знаний в два раза, так как до броска мы имели два вероятных события, а после броска — только одно, то есть в два раза меньше (рис. 2.2).

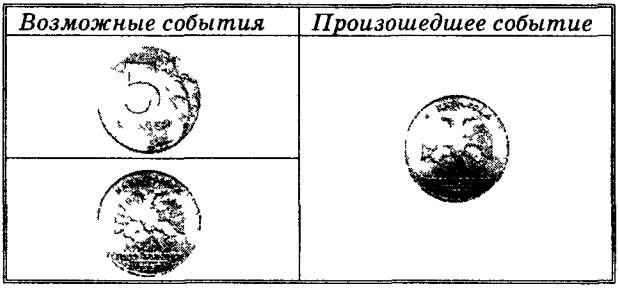


Рис. 2.2 Возможные и произошедшее события

В окружающей действительности достаточно часто встречаются ситуации, когда может произойти некоторое количество равновероятных событий. Так, при бросании равносторонней четырехгранной пирамиды существуют 4 равновероятных события, а при бросании шестигранного игрального кубика — 6 равновероятных событий.

Чем больше количество возможных событий, тем больше начальная неопределенность и соответственно тем большее количество информации будет содержать сообщение о результатах опыта.

Единицы измерения количества информации. Для количественного выражения любой величины необходимо определить единицу измерения. Так, для измерения длины в качестве единицы выбран метр, для измерения массы — килограмм и так далее. Аналогично, для определения количества информации необходимо ввести единицу измерения.

За **единицу количества информации** принимается такое количество информации, которое содержит сообщение, уменьшающее неопределенность в два раза. Такая единица названа «**бит**».

Если вернуться к опыту с бросанием монеты, то здесь неопределенность как раз уменьшается в два раза и, следовательно, полученное количество информации равно 1 биту.

Минимальной единицей измерения количества информации является бит, а следующей по величине единицей является байт, причем

$$1 \text{ байт} = 2^3 \text{ бит} = 8 \text{ бит}$$

В информатике система образования кратных единиц измерения количества информации несколько отличается от принятых в большинстве наук. Традиционные метрические системы единиц, например Международная система единиц СИ, в качестве множителей кратных единиц используют коэффициент 10^n, где $n = 3, 6, 9$ и так далее, что соответствует десятичным приставкам Кило (10^3), Мега (10^6), Гига (10^9) и так далее.

Компьютер оперирует числами не в десятичной, а в двоичной системе счисления, поэтому в кратных единицах измерения количества информации используется коэффициент 2^n.

Так, кратные байту единицы измерения количества информации вводятся следующим образом:

$$1 \text{ Кбайт} = 2^{10} \text{ байт} = 1024 \text{ байт};$$
$$1 \text{ Мбайт} = 2^{10} \text{ Кбайт} = 1024 \text{ Кбайт};$$
$$1 \text{ Гбайт} = 2^{10} \text{ Мбайт} = 1024 \text{ Мбайт}.$$

Количество возможных событий и количество информации. Существует формула, которая связывает между собой количество возможных событий N и количество информации I:

$$N = 2^I \qquad (2.1)$$

По этой формуле можно легко определить количество возможных событий, если известно количество информации. Например, если мы получили 4 бита информации, то количество возможных событий составляло:

$$N = 2^4 = 16.$$

Наоборот, для определения количества информации, если известно количество событий, необходимо решить показательное уравнение относительно I. Например, в игре «Крестики-нолики» на поле 8×8 перед первым ходом существует 64

возможных события (64 различных варианта расположения «крестика»), тогда уравнение принимает вид:

$$64 = 2^I.$$

Так как $64 = 2^6$, то получим:

$$2^6 = 2^I.$$

Таким образом, $I = 6$ битов, то есть количество информации, полученное вторым игроком после первого хода первого игрока, составляет 6 битов.

Вопросы для размышления

1. Приведите примеры уменьшения неопределенности знаний после получения информации о произошедшем событии.
2. В чем состоит неопределенность знаний в опыте по бросанию монеты?
3. Как зависит количество информации от количества возможных событий?

Задания

2.1. Какое количество информации получит второй игрок после первого хода первого игрока в игре в «Крестики-нолики» на поле размером 4×4?

2.2. Каково было количество возможных событий, если после реализации одного из них мы получили количество информации, равное 3 битам? 7 битам?

2.3. Алфавитный подход к определению количества информации

При определении количества информации на основе уменьшения неопределенности наших знаний мы рассматриваем информацию с точки зрения содержания, ее понятности и новизны для человека. С этой точки зрения в опыте по бросанию монеты одинаковое количество информации содержится и в зрительном образе упавшей монеты, и в коротком сообщении «Орел», и в длинной

фразе «Монета упала на поверхность земли той стороной вверх, на которой изображен орел».

Однако при хранении и передаче информации с помощью технических устройств целесообразно отвлечься от содержания информации и рассматривать ее как последовательность знаков (букв, цифр, кодов цветов точек изображения и так далее).

Набор символов знаковой системы (алфавит) можно рассматривать как различные возможные состояния (события). Тогда, если считать, что появление символов в сообщении равновероятно, по формуле (2.1) можно рассчитать, какое количество информации несет каждый символ.

Так, в русском алфавите, если не использовать букву ё, количество событий (букв) будет равно 32. Тогда:

$$32 = 2^I,$$

откуда $I = 5$ битов.

Каждый символ несет 5 битов информации (его информационная емкость равна 5 битов). Количество информации в сообщении можно подсчитать, умножив количество информации, которое несет один символ, на количество символов.

Количество информации, которое содержит сообщение, закодированное с помощью знаковой системы, равно количеству информации, которое несет один знак, умноженному на количество знаков.

Вопросы для размышления

1. Пусть две книги на русском и китайском языках содержат одинаковое количество знаков. В какой книге содержится большее количество информации с точки зрения алфавитного подхода?

2.4. Формула Шеннона

Существует множество ситуаций, когда возможные события имеют различные вероятности реализации. Например, если монета несимметрична (одна сторона тяжелее другой),

то при ее бросании вероятности выпадения «орла» и «решки» будут различаться.

Формулу для вычисления количества информации в случае различных вероятностей событий предложил К. Шеннон в 1948 году. В этом случае количество информации определяется по формуле:

$$I = -\sum_{i=1}^{N} p_i \log_2 p_i, \qquad (2.2)$$

где I — количество информации;

N — количество возможных событий;

p_i — вероятность i-го события.

Например, пусть при бросании несимметричной четырехгранной пирамидки вероятности отдельных событий будут равны:

$$p_1 = 1/2,\ p_2 = 1/4,\ p_3 = 1/8,\ \ p_4 = 1/8.$$

Тогда количество информации, которое мы получим после реализации одного из них, можно рассчитать по формуле (2.2):

$$I = -(1/2 \cdot \log_2 1/2 + 1/4 \cdot \log_2 1/4 + 1/8 \cdot \log_2 1/8 + 1/8 \cdot \log_2 1/8) =$$
$$= (1/2 + 2/4 + 3/8 + 3/8) \text{ битов} = 14/8 \text{ битов} = 1{,}75 \text{ бита}.$$

Этот подход к определению количества информации называется *вероятностным.*

Для частного, но широко распространенного и рассмотренного выше случая, когда события равновероятны ($p_i = 1/N$), величину количества информации I можно рассчитать по формуле:

$$I = -\sum_{i=1}^{N} \frac{1}{N} \log_2 \frac{1}{N} = \log_2 N. \qquad (2.3)$$

По формуле (2.3) можно определить, например, количество информации, которое мы получим при бросании симметричной и однородной четырехгранной пирамидки:

$$I = \log_2 4 = 2 \text{ бита}.$$

Таким образом, при бросании симметричной пирамидки, когда события равновероятны, мы получим большее количество информации (2 бита), чем при бросании несимметричной (1,75 бита), когда события неравновероятны.

Количество информации, которое мы получаем, достигает **максимального значения**, если события **равновероятны**.

Выбор оптимальной стратегии в игре «Угадай число». На получении максимального количества информации строится выбор оптимальной стратегии в игре «Угадай число», в которой первый участник загадывает целое число (например, 3) из заданного интервала (например, от 1 до 16), а второй — должен «угадать» задуманное число. Если рассмотреть эту игру с информационной точки зрения, то начальная неопределенность знаний для второго участника составляет 16 возможных событий (вариантов загаданных чисел).

При оптимальной стратегии интервал чисел всегда должен делиться пополам, тогда количество возможных событий (чисел) в каждом из полученных интервалов будет одинаково и отгадывание интервалов равновероятно. В этом случае на каждом шаге ответ первого игрока («Да» или «Нет») будет нести максимальное количество информации (1 бит).

Как видно из табл. 2.1, угадывание числа 3 произошло за четыре шага, на каждом из которых неопределенность знаний второго участника уменьшалась в два раза за счет получения сообщения от первого участника, содержащего 1 бит информации. Таким образом, количество информации, необходимое для отгадывания одного из 16 чисел, составило 4 бита.

Таблица 2.1. Информационная модель игры «Угадай число»

Вопрос второго участника	Ответ первого участника	Неопределенность знаний (количество возможных событий)	Полученное количество информации
		16	
Число больше 8?	Нет	8	1 бит
Число больше 4?	Нет	4	1 бит
Число больше 2?	Да	2	1 бит
Число 3?	Да	1	1 бит

Задания

2.3. Вычислить с помощью электронного калькулятора Wise Calculator количество информации, которое будет получено:
- при бросании симметричного шестигранного кубика;
- при игре в рулетку с 72 секторами;
- при игре в шахматы игроком за черных после первого хода белых, если считать все ходы равновероятными;
- при игре в шашки.

2.4. Вероятность первого события составляет 0,5, а второго и третьего — 0,25. Какое количество информации мы получим после реализации одного из них?

2.5. Какое количество информации получит второй игрок в игре «Угадай число» при оптимальной стратегии, если первый игрок загадал число: от 1 до 64? От 1 до 128?

2.5. Представление и кодирование информации

2.5.1. Язык как знаковая система

Для обмена информацией с другими людьми человек использует *естественные языки* (русский, английский, китайский и др.), то есть информация представляется с помощью естественных языков. В основе языка лежит алфавит, то есть набор символов (знаков), которые человек различает по их начертанию. В основе русского языка лежит кириллица, содержащая 33 знака, английский язык использует латиницу (26 знаков), китайский язык использует алфавит из десятков тысяч знаков (иероглифов).

Последовательности символов алфавита в соответствии с правилами *грамматики* образуют основные объекты языка — слова. Правила, согласно которым образуются предложения из слов данного языка, называются *синтаксисом*. Необходимо отметить, что в естественных языках грамматика и синтаксис языка формулируются с помощью большого количества правил, из которых существуют исключения, так как такие правила складывались исторически.

Наряду с естественными языками были разработаны *формальные* языки (системы счисления, язык алгебры, языки программирования и др.). Основное отличие формальных

языков от естественных состоит в наличии строгих правил грамматики и синтаксиса.

Например, системы счисления можно рассматривать как формальные языки, имеющие алфавит (цифры) и позволяющие не только именовать и записывать объекты (числа), но и выполнять над ними арифметические операции по строго определенным правилам.

Некоторые языки используют в качестве знаков не буквы и цифры, а другие символы, например химические формулы, ноты, изображения элементов электрических или логических схем, дорожные знаки, точки и тире (код азбуки Морзе) и др.

Представление информации может осуществляться с помощью **языков**, которые являются знаковыми системами. Каждая **знаковая система** строится на основе определенного **алфавита** и правил выполнения операций над знаками.

Знаки могут иметь различную физическую природу. Например, для представления информации с использованием языка в письменной форме используются знаки, которые являются изображениями на бумаге или других носителях, в устной речи в качестве знаков языка используются различные звуки (фонемы), а при обработке текста на компьютере знаки представляются в форме последовательностей электрических импульсов (компьютерных кодов).

Вопросы для размышления

1. Чем различаются естественные и формальные языки?
2. Какова может быть физическая природа знаков?

2.5.2. Представление информации в живых организмах

Общая биология 10-11 

Человек воспринимает информацию об окружающем мире с помощью органов чувств (зрения, слуха, обоняния, осязания и вкуса). Чувствительные нервные окончания органов

чувств (рецепторы) воспринимают воздействие (например, на глазном дне колбочки и палочки реагируют на воздействие световых лучей) и передают его нейронам (нервным клеткам), цепи которых составляют нервную систему.

Нейрон может находиться в одном из двух состояний: невозбужденном и возбужденном. Возбужденный нейрон генерирует электрический импульс, который передается по нервной системе.

Состояния нейрона (нет импульса, есть импульс) можно рассматривать как знаки некоторого алфавита нервной системы, с помощью которого происходит передача информации.

Генетическая информация во многом определяет строение и развитие живых организмов и передается по наследству.

Хранится генетическая информация в клетках организмов в структуре молекул ДНК (дезоксирибонуклеиновой кислоты) — рис. 2.3. Молекула ДНК состоит из двух скрученных друг с другом в спираль цепей, построенных из четырех нуклеотидов: A, G, T и C, которые образуют генетический алфавит.

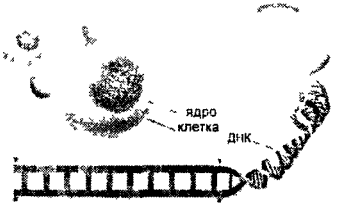


Рис. 2.3. Молекула ДНК

Молекула ДНК человека включает в себя около 3 миллиардов пар нуклеотидов и поэтому в ней закодирована вся информация об организме человека: его внешность, здоровье или предрасположенность к болезням, способности и пр.

В живых организмах информация **передается и хранится** с помощью **объектов различной физической природы** (состояния нейрона, нуклеотиды в молекуле ДНК), которые могут рассматриваться как **знаки биологических алфавитов.**

Вопросы для размышления

1. Какова физическая природа знака при представлении информации в нервной системе? В генетическом коде?

2.5.3. Кодирование информации

Представление информации происходит в различных формах в процессе восприятия окружающей среды живыми организмами и человеком, в процессах обмена информацией между человеком и человеком, человеком и компьютером, компьютером и компьютером и так далее. Преобразование информации из одной формы представления (знаковой системы) в другую называется *кодированием*.

Средством кодирования служит таблица соответствия знаковых систем, которая устанавливает взаимно однозначное соответствие между знаками или группами знаков двух различных знаковых систем. В пункте 2.10 приведена такая таблица, которая устанавливает соответствие между графическими изображениями знаков алфавита и их компьютерными кодами.

В процессе обмена информацией часто приходится производить операции *кодирования* и *декодирования* информации. При вводе знака алфавита в компьютер путем нажатия соответствующей клавиши на клавиатуре происходит кодирование знака, то есть преобразование его в компьютерный код. При выводе знака на экран монитора или принтер происходит обратный процесс — декодирование, когда из компьютерного кода знак преобразуется в его графическое изображение.

Кодирование — это операция преобразования знаков или групп знаков одной знаковой системы в знаки или группы знаков другой знаковой системы.

Рассмотрим в качестве примера кодирования соответствие цифрового и штрихового кодов товара. Такие коды имеются на каждом товаре и позволяют полностью идентифицировать товар (страну и фирму производителя, тип товара и др.).

Знакам цифрового кода (цифрам) соответствуют группы знаков штрихового кода (узкие и широкие штрихи, а также размеры промежутков между ними) — рис. 2.4. Для человека удобен цифровой код, а для автоматизированного учета — штриховой код, который считыва-

Рис. 2.4. Цифровой и штриховой коды товара

ется с помощью узкого светового луча и подвергается последующей обработке в компьютерных бухгалтерских системах учета.

Вопросы для размышления

1. Приведите примеры кодирования и декодирования информации.

2.5.4. Двоичное кодирование информации в компьютере

В компьютере для представления информации используется двоичное кодирование, так как удалось создать надежно работающие технические устройства, которые могут со стопроцентной надежностью сохранять и распознавать не более двух различных состояний (цифр):

- электромагнитные реле (замкнуто/разомкнуто), широко использовались в конструкциях первых ЭВМ;
- участок поверхности магнитного носителя информации (намагничен/размагничен);
- участок поверхности лазерного диска (отражает/не отражает);
- триггер (см. п. 3.7.3), может устойчиво находиться в одном из двух состояний, широко используется в оперативной памяти компьютера.

Все виды информации в компьютере кодируются на машинном языке, в виде логических последовательностей нулей и единиц — рис. 2.5.

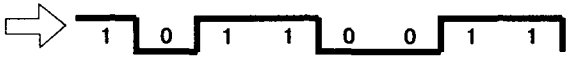
Вид информации	Двоичный код
Числовая	
Текстовая	
Графическая	
Звуковая	
Видео	

Рис. 2.5. Двоичное кодирование информации

Информация в компьютере представлена в **двоичном коде**, алфавит которого состоит из двух цифр (0 и 1).

Цифры двоичного кода можно рассматривать как два равновероятных состояния (события). При записи двоичной цифры реализуется выбор одного из двух возможных состояний (одной из двух цифр) и, следовательно, она несет количество информации, равное 1 биту.

Даже сама единица измерения количества информации бит (bit) получила свое название от английского словосочетания BInary digiT (двоичная цифра).

Важно, что каждая цифра машинного двоичного кода несет информацию в 1 бит. Таким образом, две цифры несут информацию в 2 бита, три цифры — в 3 бита и так далее. Количество информации в битах равно количеству цифр двоичного машинного кода.

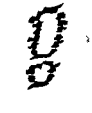
Каждая **цифра машинного двоичного кода** несет количество информации, равное **одному биту**.

Вопросы для размышления

1. Почему человек использует десятичную систему счисления, а компьютер — двоичную?

2.6. Представление числовой информации с помощью систем счисления

Для записи информации о количестве объектов используются числа. Числа записываются с использованием особых знаковых систем, которые называются системами счисления. Алфавит систем счисления состоит из символов, которые называются цифрами. Например, в десятичной системе счисления числа записываются с помощью десяти всем хорошо известных цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.



Система счисления — это знаковая система, в которой числа записываются по определенным правилам с помощью символов некоторого алфавита, называемых цифрами.

Все системы счисления делятся на две большие группы: *позиционные* и *непозиционные* системы счисления. В позиционных системах счисления значение цифры зависит от ее положения в числе, а в непозиционных — не зависит.

Римская непозиционная система счисления. Самой распространенной из непозиционных систем счисления является римская. В качестве цифр в ней используются: I (1), V (5), X (10), L (50), C (100), D (500), M (1000).

Значение цифры не зависит от ее положения в числе. Например, в числе XXX (30) цифра X встречается трижды и в каждом случае обозначает одну и ту же величину – число 10, три числа по 10 в сумме дают 30.

Величина числа в римской системе счисления определяется как сумма или разность цифр в числе. Если меньшая цифра стоит слева от большей, то она вычитается, если справа – прибавляется. Например, запись десятичного числа 1998 в римской системе счисления будет выглядеть следующим образом:

$$MCMXCVIII = 1000 + (1000 - 100) + (100 - 10) + 5 + 1 + 1 + 1.$$

Позиционные системы счисления. Первая позиционная система счисления была придумана еще в Древнем Вавилоне, причем вавилонская нумерация была шестидесятеричной, то есть в ней использовалось шестьдесят цифр! Интересно, что до сих пор при измерении времени мы используем основание, равное 60 (в 1 минуте содержится 60 секунд, а в 1 часе — 60 минут).

В XIX веке довольно широкое распространение получила двенадцатеричная система счисления. До сих пор мы часто употребляем дюжину (число 12): в сутках две дюжины часов, круг содержит тридцать дюжин градусов и так далее.

В **позиционных системах счисления** количественное значение цифры зависит от ее позиции в числе.

Наиболее распространенными в настоящее время позиционными системами счисления являются десятичная, двоичная, восьмеричная и шестнадцатеричная. Каждая позиционная система имеет определенный *алфавит цифр* и *основание*.

В *позиционных системах счисления* основание системы равно количеству цифр (знаков в ее алфавите) и определяет, во сколько раз различаются значения одинаковых цифр, стоящих в соседних позициях числа.

Десятичная система счисления имеет алфавит цифр, который состоит из десяти всем известных, так называемых арабских, цифр, и основание, равное 10, двоичная — две цифры и основание 2, восьмеричная — восемь цифр и основание 8, шестнадцатеричная — шестнадцать цифр (в качестве цифр используются и буквы латинского алфавита) и основание 16 (табл. 2.2).

Таблица 2.2. Позиционные системы счисления

Система счисления	Основание	Алфавит цифр
Десятичная	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Двоичная	2	0, 1
Восьмеричная	8	0, 1, 2, 3, 4, 5, 6, 7
Шестнадцатеричная	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A (10), B (11), C (12), D (13), E (14), F (15)

Десятичная система счисления. Рассмотрим в качестве примера десятичное число 555. Цифра 5 встречается трижды, причем самая правая цифра 5 обозначает пять единиц, вторая справа — пять десятков и, наконец, третья справа — пять сотен.

Позиция цифры в числе называется *разрядом*. Разряд числа возрастает справа налево, от младших разрядов к старшим. В десятичной системе цифра, находящаяся в крайней справа позиции (разряде), обозначает количество единиц, цифра, смещенная на одну позицию влево, — количество десятков, еще левее — сотен, затем тысяч и так далее. Соответственно имеем разряд единиц, разряд десятков и так далее.

Число 555 записано в привычной для нас *свернутой* форме. Мы настолько привыкли к такой форме записи, что уже не замечаем, как в уме умножаем цифры числа на различные степени числа 10.

В *развернутой* форме записи числа такое умножение записывается в явной форме. Так, в развернутой форме запись

числа 555 в десятичной системе будет выглядеть следующим образом:

$$555_{10} = 5 \cdot 10^2 + 5 \cdot 10^1 + 5 \cdot 10^0.$$

Как видно из примера, число в позиционной системе счисления записывается в виде суммы числового ряда степеней *основания* (в данном случае 10), в качестве коэффициентов которых выступают цифры данного числа.

Для записи десятичных дробей используются отрицательные значения степеней основания. Например, число 555,55 в развернутой форме записывается следующим образом:

$$555{,}55_{10} = 5 \cdot 10^2 + 5 \cdot 10^1 + 5 \cdot 10^0 + 5 \cdot 10^{-1} + 5 \cdot 10^{-2}.$$

В общем случае в десятичной системе счисления запись числа A_{10}, которое содержит n целых разрядов числа и m дробных разрядов числа, выглядит так:

$$A_{10} = a_{n-1} \cdot 10^{n-1} + \ldots + a_0 \cdot 10^0 + a_{-1} \cdot 10^{-1} + \ldots + a_{-m} \cdot 10^{-m}.$$

Коэффициенты a_i в этой записи являются цифрами десятичного числа, которое в свернутой форме записывается так:

$$A_{10} = a_{n-1}\, a_{n-2} \ldots a_0\,,\, a_{-1} \ldots a_{-m}\,.$$

Из вышеприведенных формул видно, что умножение или деление десятичного числа на 10 (величину основания) приводит к перемещению запятой, отделяющей целую часть от дробной, на один разряд соответственно вправо или влево. Например:

$$555{,}55_{10} \cdot 10 = 5555{,}5_{10};$$
$$555{,}55_{10} : 10 = 55{,}555_{10}.$$

Двоичная система счисления. В двоичной системе счисления основание равно 2, а алфавит состоит из двух цифр (0 и 1). Следовательно, числа в двоичной системе в развернутой форме записываются в виде суммы степеней основания 2 с коэффициентами, в качестве которых выступают цифры 0 или 1.

Например, развернутая запись двоичного числа может выглядеть так:

$$A_2 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2}.$$

Свернутая форма этого же числа:

$$A_2 = 101{,}01_2 .$$

В общем случае в двоичной системе запись числа A_2, которое содержит n целых разрядов числа и m дробных разрядов числа, выглядит так:

$$A_2 = a_{n-1} \cdot 2^{n-1} + a_{n-2} \cdot 2^{n-2} + \ldots a_0 \cdot 2^0 + a_{-1} \cdot 2^{-1} + \ldots + a_{-m} \cdot 2^{-m}.$$

Коэффициенты a_i в этой записи являются цифрами (0 или 1) двоичного числа, которое в свернутой форме записывается так:

$$A_2 = a_{n-1}\, a_{n-2} \ldots a_0 , a_{-1}\, a_{-2} \ldots a_{-m} .$$

Из вышеприведенных формул видно, что умножение или деление двоичного числа на 2 (величину основания) приводит к перемещению запятой, отделяющей целую часть от дробной на один разряд соответственно вправо или влево. Например:

$$101{,}01_2 \cdot 2 = 1010{,}1_2 ;$$
$$101{,}01_2 : 2 = 10{,}101_2 .$$

Позиционные системы счисления с произвольным основанием. Возможно использование множества позиционных систем счисления, основание которых равно или больше 2. В системах счисления с основанием q (q-ичная система счисления) числа в развернутой форме записываются в виде суммы степеней основания q с коэффициентами, в качестве которых выступают цифры 0, 1, $q-1$:

$$A_q = a_{n-1} \cdot q^{n-1} + a_{n-2} \cdot q^{n-2} + \ldots + a_0 \cdot q^0 + a_{-1} \cdot q^{-1} + \ldots + a_{-m} \cdot q^{-m}.$$

Коэффициенты a_i в этой записи являются цифрами числа, записанного в q-ичной системе счисления.

Так, в восьмеричной системе основание равно восьми ($q = 8$). Тогда записанное в свернутой форме восьмеричное

число $A_8 = 673{,}2_8$ в развернутой форме будет иметь вид:

$$A_8 = 6 \cdot 8^2 + 7 \cdot 8^1 + 3 \cdot 8^0 + 2 \cdot 8^{-1}.$$

В шестнадцатеричной системе основание равно шестнадцати ($q = 16$), тогда записанное в свернутой форме шестнадцатеричное число $A_{16} = 8\text{A},\text{F}_{16}$ в развернутой форме будет иметь вид:

$$A_{16} = 8 \cdot 16^1 + \text{A} \cdot 16^0 + \text{F} \cdot 16^{-1}.$$

Если выразить шестнадцатеричные цифры через их десятичные значения (A=10, F=15), то запись числа примет вид:

$$A_{16} = 8 \cdot 16^1 + 10 \cdot 16^0 + 15 \cdot 16^{-1}.$$

Вопросы для размышления

1. Чем отличаются позиционные системы счисления от непозиционных?
2. Может ли в качестве цифры использоваться символ буквы?
3. Какое количество цифр используется в q-ичной системе счисления?

Задания

2.6. Записать числа $19{,}99_{10}$, $10{,}10_2$, $64{,}5_8$, $39{,}\text{F}_{16}$ в развернутой форме.

2.7. Во сколько раз увеличатся числа $10{,}1_{10}$, $10{,}1_2$, $64{,}5_8$, $39{,}\text{F}_{16}$ при переносе запятой на один знак вправо?

2.8. При переносе запятой на два знака вправо число $11{,}11_x$ увеличилось в 4 раза. Чему равно x?

2.9. Какое минимальное основание может иметь система счисления, если в ней записаны числа 23 и 67?

2.10. Записать число 1999_{10} в римской системе счисления.

2.7. Перевод чисел в позиционных системах счисления

2.7.1. Перевод чисел в десятичную систему счисления

Преобразование чисел, представленных в двоичной, восьмеричной и шестнадцатеричной системах счисления, в десятичную выполнить довольно легко. Для этого необходимо записать число в развернутой форме и вычислить его значение.

Перевод числа из двоичной системы в десятичную. Возьмем любое двоичное число, например $10,11_2$. Запишем его в развернутой форме и произведем вычисления:

$$10,11_2 = 1\cdot2^1 + 0\cdot2^0 + 1\cdot2^{-1} + 1\cdot2^{-2} = \\ = 1\cdot2 + 0\cdot1 + 1\cdot1/2 + 1\cdot1/4 = 2,75_{10}.$$

Перевод чисел из восьмеричной системы в десятичную. Возьмем любое восьмеричное число, например $67,5_8$. Запишем его в развернутой форме и произведем вычисления:

$$67,5_8 = 6\cdot8^1 + 7\cdot8^0 + 5\cdot8^{-1} = 6\cdot8 + 7\cdot1 + 5\cdot1/8 = 55,625_{10}.$$

Перевод чисел из шестнадцатеричной системы в десятичную. Возьмем любое шестнадцатеричное число, например $19F_{16}$. Запишем его в развернутой форме (при этом необходимо помнить, что шестнадцатеричная цифра F соответствует десятичному числу 15) и произведем вычисления:

$$19F_{16} = 1\cdot16^2 + 9\cdot16^1 + F\cdot8^0 = 1\cdot256 + 9\cdot16 + 15\cdot1 = 415_{10}.$$

Задания

2.11. Перевести в десятичную систему следующие числа: 101_2, 110_2, 111_2, 7_8, 11_8, 22_8, $1A_{16}$, BF_{16}, $9C_{16}$.

2.12. Провести проверку выполнения задания 2.11 с помощью электронного калькулятора NumLock Calculator.

2.7.2. Перевод чисел из десятичной системы в двоичную, восьмеричную и шестнадцатеричную

Перевод чисел из десятичной системы в двоичную, восьмеричную и шестнадцатеричную более сложен и может осуществляться различными способами. Рассмотрим один из алгоритмов перевода на примере перевода чисел из десятичной

системы в двоичную. При этом необходимо учитывать, что алгоритмы перевода целых чисел и правильных дробей будут различаться.

Алгоритм перевода целых десятичных чисел в двоичную систему счисления. Пусть $A_{цд}$ — целое десятичное число. Запишем его в виде суммы степеней основания 2 с двоичными коэффициентами. В его записи в развернутой форме будут отсутствовать отрицательные степени основания (числа 2):

$$A_{цд} = a_{n-1}\cdot 2^{n-1} + a_{n-2}\cdot 2^{n-2} + \ldots + a_1\cdot 2^1 + a_0\cdot 2^0.$$

На первом шаге разделим число $A_{цд}$ на основание двоичной системы, то есть на 2. Частное от деления будет равно

$$a_{n-1}\cdot 2^{n-2} + a_{n-2}\cdot 2^{n-3} + \ldots + a_1,$$

а остаток — равен a_0.

На втором шаге целое частное опять разделим на 2, остаток от деления будет теперь равен a_1.

Если продолжать этот процесс деления, то после n-го шага получим последовательность остатков:

$$a_0,\ a_1,\ \ldots,\ a_{n-1}.$$

Легко заметить, что их последовательность совпадает с обратной последовательностью цифр целого двоичного числа, записанного в свернутой форме:

$$A_2 = a_{n-1}\ldots a_1 a_0\ .$$

Таким образом, достаточно записать остатки в обратной последовательности, чтобы получить искомое двоичное число.

Алгоритм перевода целого десятичного числа в двоичное будет следующим:

1. Последовательно выполнять деление исходного целого десятичного числа и получаемых целых частных на основание системы (на 2) до тех пор, пока не получится частное, меньшее делителя, то есть меньшее 2.
2. Записать полученные остатки в обратной последовательности.

В качестве примера рассмотрим перевод десятичного числа 19 в двоичную систему, записывая результаты в таблицу:

Десятичное число/ целое частное	Делитель (основание системы)	Остаток	Цифры двоичного числа
19	2	1	a_0
9	2	1	a_1
4	2	0	a_2
2	2	0	a_3
1	2	1	a_4

В результате получаем двоичное число:

$$A_2 = a_1a_3a_2a_1a_0 = 10011_2.$$

Алгоритм перевода правильных десятичных дробей в двоичную систему счисления. Пусть $A_{дд}$ — правильная десятичная дробь. В ее записи в развернутой форме будут отсутствать положительные степени основания (числа 2):

$$A_{дд} = a_{-1} \cdot 2^{-1} + a_{-2} \cdot 2^{-2} + \ldots$$

На первом шаге умножим число $A_{дд}$ на основание двоичной системы, то есть на 2. Произведение будет равно:

$$a_{-1} + a_{-2} \cdot 2^{1} + \ldots$$

Целая часть будет равна a_{-1}.

На втором шаге оставшуюся дробную часть опять умножим на 2, получим целую часть, равную a_{-2}.

Описанный процесс необходимо продолжать до тех пор, пока в результате умножения мы не получим нулевую дробную часть или не будет достигнута требуемая точность вычислений.

Легко заметить, что последовательность полученных чисел совпадает с последовательностью цифр дробного двоичного числа, записанного в свернутой форме:

$$A_2 = a_{-1}a_{-2} \ldots$$

Алгоритм перевода правильной десятичной дроби в двоичную будет следующим:

1. Последовательно выполнять умножение исходной десятичной дроби и получаемых дробных частей произведений на основание системы (на 2) до тех пор, пока не получится нулевая дробная часть или не будет достигнута требуемая точность вычислений.
2. Записать полученные целые части произведения в прямой последовательности.

В качестве примера рассмотрим перевод десятичной дроби 0,75 в двоичную систему, записывая результаты в таблицу:

Десятичная дробь/дробная часть произведения	Множитель (основание системы)	Целая часть произведения	Цифры двоичного числа
0,75	2	1	a_{-1} ↓
0,50	2	1	a_{-2} ↓
0,00	2		

В результате получаем двоичную дробь:

$$A_2 = 0,a_{-1}a_{-2} = 0,11_2.$$

Перевод чисел из системы с основанием p в систему с основанием q. Перевод чисел из позиционной системы с произвольным основанием p в систему с основанием q производится по алгоритмам, аналогичным рассмотренным выше.

Рассмотрим алгоритм перевода целых чисел на примере перевода целого десятичного числа $A_{10} = 424_{10}$ в шестнадцатеричную систему, то есть из системы счисления с основанием $p = 10$ в систему счисления с основанием $q = 16$.

В процессе выполнения алгоритма необходимо обратить внимание, что все действия необходимо осуществлять в исходной системе счисления (в данном случае десятичной), а полученные остатки записывать цифрами новой системы счисления (в данном случае шестнадцатеричной).

Десятичное число/ целое частное	Делитель (основание системы)	Остаток	Цифры двоичного числа
424	16	8	a_0 ↑
26	16	10 (A)	a_1 ↑
1	16	1	a_2

В результате получаем шестнадцатеричное число:

$$A_{16} = a_2a_1a_0 = 1A8_{16}.$$

Рассмотрим теперь алгоритм перевода дробных чисел на примере перевода десятичной дроби $A_{10} = 0{,}625$ в восьмеричную систему, то есть из системы счисления с основанием $p = 10$ в систему счисления с основанием $q = 8$.

В процессе выполнения алгоритма необходимо обратить внимание, что все действия необходимо осуществлять в исходной системе счисления (в данном случае десятичной), а полученные остатки записывать цифрами новой системы счисления (в данном случае восьмеричной).

Десятичная дробь/дробная часть произведения	Множитель (основание системы)	Целая часть произведения	Цифры двоичного числа
0,40625	8	3	a_{-1}
0,25	8	2	a_{-2} ↓
0,00	8		

В результате получаем восьмеричную дробь:

$$A_8 = a_{-1}a_{-2} = 0{,}32_8.$$

Перевод чисел, содержащих и целую и дробную части, производится в два этапа. Отдельно переводится по соответствующему алгоритму целая часть и отдельно — дробная. В итоговой записи полученного числа целая часть от дробной отделяется запятой.

Задания

2.13. Перевести целые десятичные числа 9_{10}, 17_{10} и 243_{10} в двоичную, восьмеричную и шестнадцатеричную системы счисления.

2.14. Перевести десятичные дроби $0{,}2_{10}$ и $0{,}35_{10}$ в двоичную, восьмеричную и шестнадцатеричную системы счисления с точностью до трех знаков после запятой.

2.15. Перевести десятичные числа $3{,}5_{10}$ и $47{,}85_{10}$ в двоичную, восьмеричную и шестнадцатеричную системы счисления с точностью до трех знаков после запятой.

2.7.3. Перевод чисел из двоичной системы счисления в восьмеричную и шестнадцатеричную и обратно

Перевод чисел между системами счисления, основания которых являются степенями числа 2 ($q = 2^n$), может производиться по более простым алгоритмам. Такие алгоритмы могут применяться для перевода чисел между двоичной ($q = 2^1$), восьмеричной ($q = 2^3$) и шестнадцатеричной ($q = 2^4$) системами счисления.

Перевод чисел из двоичной системы счисления в восьмеричную. Для записи двоичных чисел используются две цифры, то есть в каждом разряде числа возможны 2 варианта записи. Решаем показательное уравнение:

$2 = 2^I$. Так как $2 = 2^1$, то $I = 1$ бит.

Каждый разряд двоичного числа содержит 1 бит информации.

Для записи восьмеричных чисел используются восемь цифр, то есть в каждом разряде числа возможны 8 вариантов записи. Решаем показательное уравнение:

$8 = 2^I$. Так как $8 = 2^3$, то $I = 3$ бита.

Каждый разряд восьмеричного числа содержит 3 бита информации.

Таким образом, для перевода целого двоичного числа в восьмеричное его нужно разбить на группы по три цифры, справа налево, а затем преобразовать каждую группу в восьмеричную цифру. Если в последней, левой, группе окажется меньше трех цифр, то необходимо ее дополнить слева нулями.

Переведем таким способом двоичное число 101001_2 в восьмеричное:

$$101\ 001_2 \Rightarrow 1\cdot2^2+0\cdot2^1+1\cdot2^0 \qquad 0\cdot2^2+0\cdot2^1+1\cdot2^0 \Rightarrow 51_8.$$

Для упрощения перевода можно заранее подготовить таблицу преобразования двоичных триад (групп по 3 цифры) в восьмеричные цифры:

Двоичные триады	000	001	010	011	100	101	110	111
Восьмеричные цифры	0	1	2	3	4	5	6	7

Для перевода дробного двоичного числа (правильной дроби) в восьмеричное необходимо разбить его на триады слева направо и, если в последней, правой, группе окажется меньше трех цифр, дополнить ее справа нулями. Далее необходимо триады заменить на восьмеричные числа.

Например, преобразуем дробное двоичное число $A_2 = 0{,}110101_2$ в восьмеричную систему счисления:

Двоичные триады	110	101
Восьмеричные цифры	6	5

Получаем: $A_8 = 0{,}65_8$.

Перевод чисел из двоичной системы счисления в шестнадцатеричную. Для записи шестнадцатеричных чисел используются шестнадцать цифр, то есть в каждом разряде числа возможны 16 вариантов записи. Решаем показательное уравнение:

$16 = 2^I$. Так как $16 = 2^4$, то $I = 4$ бита.

Каждый разряд шестнадцатеричного числа содержит 4 бита информации.

Таким образом, для перевода целого двоичного числа в шестнадцатеричное его нужно разбить на группы по четыре цифры (тетрады), начиная справа, и, если в последней левой группе окажется меньше четырех цифр, дополнить ее слева нулями. Для перевода дробного двоичного числа (правильной дроби) в шестнадцатеричное необходимо разбить его на тетрады слева направо и, если в последней правой группе

окажется меньше четырех цифр, то необходимо дополнить ее справа нулями.

Затем надо преобразовать каждую группу в шестнадцатеричную цифру, воспользовавшись для этого предварительно составленной таблицей соответствия двоичных тетрад и шестнадцатеричных цифр.

Переведем целое двоичное число $A_2 = 101001_2$ в шестнадцатеричное:

Двоичные тетрады	0010	1001
Шестнадцатеричные цифры	2	9

В результате имеем: $A_{16} = 29_{16}$.

Переведем дробное двоичное число $A_2 = 0{,}110101_2$ в шестнадцатеричную систему счисления:

Двоичные тетрады	1101	0100
Шестнадцатеричные цифры	D	4

Получаем: $A_{16} = 0{,}D4_{16}$.

Для того чтобы преобразовать любое двоичное число в восьмеричную или шестнадцатеричную системы счисления, необходимо произвести преобразования по рассмотренным выше алгоритмам отдельно для его целой и дробной частей.

Перевод чисел из восьмеричной и шестнадцатеричной систем счисления в двоичную. Для перевода чисел из восьмеричной и шестнадцатеричной систем счисления в двоичную необходимо цифры числа преобразовать в группы двоичных цифр. Для перевода из восьмеричной системы в двоичную каждую цифру числа надо преобразовать в группу из трех двоичных цифр (триаду), а при преобразовании шестнадцатеричного числа — в группу из четырех цифр (тетраду).

Например, преобразуем дробное восьмеричное число $A_8 = 0{,}47_8$ в двоичную систему счисления:

Восьмеричные цифры	4	7
Двоичные триады	100	111

Получаем: $A_2 = 0{,}100111_2$.

Переведем целое шестнадцатеричное число $A_{16} = AB_{16}$ в двоичную систему счисления:

Шестнадцатеричные цифры	A	B
Двоичные тетрады	1010	1011

В результате имеем: $A_2 = 10101011_2$.

Задания

2.16. Составить таблицу соответствия двоичных тетрад и шестнадцатеричных цифр.

2.17. Перевести в восьмеричную и шестнадцатеричную системы счисления следующие целые числа: 1111_2, 1010101_2.

2.18. Перевести в восьмеричную и шестнадцатеричную системы счисления следующие дробные числа: $0{,}01111_2$, $0{,}10101011_2$.

2.19. Перевести в восьмеричную и шестнадцатеричную системы счисления следующие числа: $11{,}01_2$, $110{,}101_2$.

2.20. Перевести в двоичную систему счисления следующие числа: $46{,}27_8$, $EF{,}12_{16}$.

2.21. Сравнить числа, выраженные в различных системах счисления:
1101_2 и D_{16}; $0{,}11111_2$ и $0{,}22_8$; $35{,}63_8$ и $16{,}C_{16}$.

2.8. Арифметические операции в позиционных системах счисления

Арифметические операции во всех позиционных системах счисления выполняются по одним и тем же хорошо известным вам правилам.

Сложение. Рассмотрим сложение чисел в двоичной системе счисления. В его основе лежит таблица сложения одноразрядных двоичных чисел:

$$\begin{aligned} 0 + 0 &= 0 \\ 0 + 1 &= 1 \\ 1 + 0 &= 1 \\ 1 + 1 &= 10 \end{aligned}$$

Важно обратить внимание на то, что при сложении двух единиц происходит переполнение разряда и производится перенос в старший разряд. Переполнение разряда наступает тогда, когда величина числа в нем становится равной или большей основания.

Сложение многоразрядных двоичных чисел происходит в соответствии с вышеприведенной таблицей сложения с учетом возможных переносов из младших разрядов в старшие. В качестве примера сложим в столбик двоичные числа 110_2 и 11_2:

$$\begin{array}{r} 110_2 \\ + \quad 11_2 \\ \hline 1001_2 \end{array}$$

Проверим правильность вычислений сложением в десятичной системе счисления. Переведем двоичные числа в десятичную систему счисления и затем их сложим:

$$110_2 = 1\cdot2^2 + 1\cdot2^1 + 0\cdot2^0 = 6_{10};$$
$$11_2 = 1\cdot2^1 + 1\cdot2^0 = 3_{10};$$
$$6_{10} + 3_{10} = 9_{10}.$$

Теперь переведем результат двоичного сложения в десятичное число:

$$1001_2 = 1\cdot2^3 + 0\cdot2^2 + 0\cdot2^1 + 1\cdot2^0 = 9_{10}.$$

Сравним результаты — сложение выполнено правильно.

Вычитание. Рассмотрим вычитание двоичных чисел. В его основе лежит таблица вычитания одноразрядных двоичных чисел. При вычитании из меньшего числа (0) большего (1) производится заем из старшего разряда. В таблице заем обозначен 1 с чертой:

$$0 - 0 = \_0$$
$$0 - 1 = 11$$
$$1 - 0 = 1$$
$$1 - 1 = 0$$

Вычитание многоразрядных двоичных чисел происходит в соответствии с вышеприведенной таблицей вычитания с учетом возможных заемов из старших разрядов. В качестве примера произведем вычитание двоичных чисел 110_2 и 11_2:

$$\begin{array}{r} 110_2 \\ - \quad 11_2 \\ \hline 11_2 \end{array}$$

Умножение. В основе умножения лежит таблица умножения одноразрядных двоичных чисел:

$$0 \cdot 0 = 0$$
$$0 \cdot 1 = 0$$
$$1 \cdot 0 = 0$$
$$1 \cdot 1 = 1$$

Умножение многоразрядных двоичных чисел происходит в соответствии с вышеприведенной таблицей умножения по обычной схеме, применяемой в десятичной системе счисления с последовательным умножением множимого на цифры множителя. В качестве примера произведем умножение двоичных чисел 110_2 и 11_2:

$$\begin{array}{r} 110_2 \\ \times \quad 11_2 \\ \hline 110 \\ 110 \\ \hline 10010_2 \end{array}$$

Деление. Операция деления выполняется по алгоритму, подобному алгоритму выполнения операции деления в десятичной системе счисления. В качестве примера произведем деление двоичного числа 110_2 на 11_2:

$$\begin{array}{r|l} 110_2 & 11_2 \\ \hline 11 & 10_2 \\ \hline 0 & \end{array}$$

Арифметические операции в восьмеричной и шестнадцатеричной системах счисления. Аналогично можно выполнять арифметические действия в восьмеричной и шестнадцатеричной системах счисления. Необходимо только помнить, что величина переноса в следующий разряд при сложении и заем из старшего разряда при вычитании определяется величиной основания системы счисления:

$$\begin{array}{r} 37_8 \\ + \quad 25_8 \\ \hline 64_8 \end{array} \qquad \begin{array}{r} 9C_{16} \\ - \quad 78_{16} \\ \hline 24_{16} \end{array}$$

Для проведения арифметических операций над числами, выраженными в различных системах счисления, необходимо предварительно перевести их в одну и ту же систему.

Задания

2.22. Провести сложение, вычитание, умножение и деление двоичных чисел 1010_2 и 10_2 и проверить правильность выполнения арифметических действий с помощью электронного калькулятора Wise Calculator.

2.23. Сложить восьмеричные числа: 5_8 и 4_8, 17_8 и 41_8.

2.24. Провести вычитание шестнадцатеричных чисел: F_{16} и A_{16}, 41_{16} и 17_{16}.

2.25. Сложить числа: 17_8 и 17_{16}, 41_8 и 41_{16}.

2.9. Представление чисел в компьютере

Представление чисел в формате с фиксированной запятой. Целые числа в компьютере хранятся в памяти в формате *с фиксированной запятой*. В этом случае каждому разряду ячейки памяти соответствует всегда один и тот же разряд числа, а «запятая» «находится» справа после младшего разряда, то есть вне разрядной сетки.

Для хранения *целых неотрицательных чисел* отводится одна ячейка памяти (8 битов). Например, число $A_2 = 11110000_2$ будет храниться в ячейке памяти следующим образом:

1	1	1	1	0	0	0	0

Максимальное значение целого неотрицательного числа достигается в случае, когда во всех ячейках хранятся единицы. Для n-разрядного представления оно будет равно

$$2^n - 1.$$

Определим диапазон чисел, которые могут храниться в оперативной памяти в формате *целых неотрицательных чисел*. Минимальное число соответствует восьми нулям, хранящимся в восьми битах ячейки памяти, и равно нулю. Максимальное число соответствует восьми единицам и равно

$$A = 1\cdot2^7 + 1\cdot2^6 + 1\cdot2^5 + 1\cdot2^4 + 1\cdot2^3 + 1\cdot2^2 + 1\cdot2^1 + 1\cdot2^0 = 1\cdot2^8 - 1 = 255_{10}.$$

Диапазон изменения *целых неотрицательных чисел* чисел: от 0 до 255.

Для хранения *целых чисел со знаком* отводится две ячейки памяти (16 битов), причем старший (левый) разряд отводится под знак числа (если число положительное, то в знаковый разряд записывается 0, если число отрицательное — 1).

Представление в компьютере положительных чисел с использованием формата «знак–величина» называется *прямым кодом* числа. Например, число $2002_{10} = 11111010010_2$ будет представлено в 16-разрядном представлении следующим образом:

0	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0

Максимальное положительное число (с учетом выделения одного разряда на знак) для целых чисел со знаком в n-разрядном представлении равно:

$$A = 2^{n-1} - 1.$$

Для представления отрицательных чисел используется *дополнительный код*. Дополнительный код позволяет заменить арифметическую операцию вычитания операцией сложения, что существенно упрощает работу процессора и увеличивает его быстродействие.

Дополнительный код отрицательного числа A, хранящегося в n ячейках, равен $2^n - |A|$.

Дополнительный код представляет собой дополнение модуля отрицательного числа A до 0, так как в n-разрядной компьютерной арифметике:

$$2^n - |A| + |A| = 0,$$

поскольку в компьютерной n-разрядной арифметике $2^n \equiv 0$. Действительно, двоичная запись такого числа состоит из одной единицы и n нулей, а в n-разрядную ячейку может уместиться только n младших разрядов, то есть n нулей.

Для получения дополнительного кода отрицательного числа можно использовать довольно простой алгоритм:

1. Модуль числа записать в *прямом коде* в n двоичных разрядах.
2. Получить *обратный код* числа, для этого значения всех битов инвертировать (все единицы заменить на нули и все нули заменить на единицы).
3. К полученному обратному коду прибавить единицу.

Запишем дополнительный код отрицательного числа −2002 для 16-разрядного компьютерного представления:

Прямой код модуля	$\|-2002_{10}\|$	0000011111010010_2
Обратный код	Инвертирование	1111100000101101_2
	Прибавление единицы	$1111100000101101_2 + 0000000000000001_2$
Дополнительный код		1111100000101110_2

При n-разрядном представлении отрицательного числа A в дополнительным коде старший разряд выделяется для хранения знака числа (единицы). В остальных разрядах записывается положительное число

$$2^{n-1} - |A|.$$

Чтобы число было положительным, должно выполняться условие

$$|A| \leqslant 2^{n-1}.$$

Следовательно, максимальное значение модуля числа A в n-разрядном представлении равно:

$$|A| = 2^{n-1}.$$

Тогда минимальное отрицательное число равно:

$$A = -2^{n-1}.$$

Определим диапазон чисел, которые могут храниться в оперативной памяти в формате *длинных целых чисел со знаком* (для хранения таких чисел отводится четыре ячейки памяти — 32 бита).

Максимальное положительное целое число (с учетом выделения одного разряда на знак) равно:

$$A = 2^{31} - 1 = 2\ 147\ 483\ 647_{10}.$$

Минимальное отрицательное целое число равно:

$$A = -2^{31} = -2\ 147\ 483\ 648_{10}.$$

Достоинствами представления чисел в формате *с фиксированной запятой* являются простота и наглядность представления чисел, а также простота алгоритмов реализации арифметических операций.

Недостатком представления чисел в формате *с фиксированной запятой* является небольшой диапазон представления величин, недостаточный для решения математических, физических, экономических и других задач, в которых используются как очень малые, так и очень большие числа.

Представление чисел в формате с плавающей запятой. Вещественные числа хранятся и обрабатываются в компьютере в формате *с плавающей запятой*. В этом случае положение запятой в записи числа может изменяться.

Формат чисел *с плавающей запятой* базируется на экспоненциальной форме записи, в которой может быть представлено любое число. Так число A может быть представлено в виде:

$$A = m \cdot q^n, \qquad 2.3$$

где m — мантисса числа;
q — основание системы счисления;
n — порядок числа.

Для единообразия представления чисел *с плавающей запятой* используется нормализованная форма, при которой мантисса отвечает условию:

$$1/n \leqslant |m| < 1.$$

Это означает, что мантисса должна быть правильной дробью и иметь после запятой цифру, отличную от нуля.

Преобразуем десятичное число 555,55, записанное в естественной форме, в экспоненциальную форму с нормализованной мантиссой:

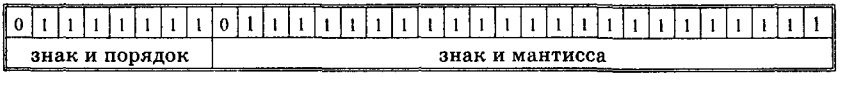
$$555{,}55 = 0{,}55555 \cdot 10^3.$$

Здесь нормализованная мантисса: $m = 0{,}55555$, порядок: $n = 3$.

Число в формате с плавающей запятой занимает в памяти компьютера 4 (*число обычной точности*) или 8 байтов (*число двойной точности*). При записи числа с плавающей запятой выделяются разряды для хранения знака мантиссы, знака порядка, порядка и мантиссы.

Диапазон изменения чисел определяется количеством разрядов, отведенных для хранения порядка числа, а точность (количество значащих цифр) определяется количеством разрядов, отведенных для хранения мантиссы.

Определим максимальное число и его точность для формата *чисел обычной точности*, если для хранения порядка и его знака отводится 8 разрядов, а для хранения мантиссы и ее знака — 24 разряда:



Максимальное значение порядка числа составит $1111111_2 = 127_{10}$, и, следовательно, максимальное значение числа составит:

$$2^{127} = 1{,}70141183460469231731687303715884 \cdot 10^{38}.$$

Максимальное значение положительной мантиссы равно:

$$2^{23} - 1 \approx 2^{23} = 2^{(10 \cdot 2{,}3)} \approx 1000^{2{,}3} = 10^{(3 \cdot 2{,}3)} \approx 10^7.$$

Таким образом максимальное значение *чисел обычной точности* с учетом возможной точности вычислений составит $1{,}701411 \cdot 10^{38}$ (количество значащих цифр десятичного числа в данном случае ограничено 7 разрядами).

Задания

2.26. Заполнить таблицу, записав отрицательные десятичные числа в прямом, обратном и дополнительном кодах в 16-разрядном представлении:

Десятичные числа	Прямой код	Обратный код	Дополнительный код
-50			
-500			

2.27. Определить диапазон представления *целых чисел со знаком* (отводится 2 байта памяти) в формате с фиксированной запятой.

2.28. Определить максимальное число и его точность для формата *чисел двойной точности*, если для хранения порядка и его знака отводится 11 разрядов, а для хранения мантиссы и ее знака — 53 разряда.

2.10. Двоичное кодирование текстовой информации

Начиная с конца 60-х годов, компьютеры все больше стали использоваться для обработки текстовой информации и в настоящее время большая часть персональных компьютеров в мире (и наибольшее время) занято обработкой именно текстовой информации.

Традиционно для кодирования одного символа используется количество информации, равное 1 байту, то есть $I =$ = 1 байт = 8 битов.

Для кодирования одного **символа** требуется **1 байт** информации.

Если рассматривать символы как возможные события, то по формуле (2.1) можно вычислить, какое количество различных символов можно закодировать:

$$N = 2^I = 2^8 = 256.$$

Такое количество символов вполне достаточно для представления текстовой информации, включая прописные и строчные буквы русского и латинского алфавита, цифры, знаки, графические символы и пр.

Кодирование заключается в том, что каждому символу ставится в соответствие уникальный десятичный код от 0 до 255 или соответствующий ему двоичный код от 00000000 до 11111111. Таким образом, человек различает символы по их начертаниям, а компьютер — по их кодам.

При вводе в компьютер текстовой информации происходит ее двоичное кодирование, изображение символа преобразуется в его двоичный код. Пользователь нажимает на клавиатуре клавишу с символом, и в компьютер поступает определенная последовательность из восьми электрических импульсов (двоичный код символа). Код символа хранится в оперативной памяти компьютера, где занимает один байт.

В процессе вывода символа на экран компьютера производится обратный процесс — декодирование, то есть преобразование кода символа в его изображение.

Важно, что присвоение символу конкретного кода — это вопрос соглашения, которое фиксируется в кодовой таблице. Первые 33 кода (с 0 по 32) соответствуют не символам, а операциям (перевод строки, ввод пробела и так далее).

Коды с 33 по 127 являются интернациональными и соответствуют символам латинского алфавита, цифрам, знакам арифметических операций и знакам препинания.

Коды с 128 по 255 являются национальными, то есть в национальных кодировках одному и тому же коду соответствуют различные символы. К сожалению, в настоящее время существуют пять различных кодовых таблиц для русских букв (КОИ8, CP1251, CP866, Mac, ISO — табл. 2.3), поэтому тексты, созданные в одной кодировке, не будут правильно отображаться в другой.

В настоящее время широкое распространение получил новый международный стандарт Unicode, который отводит

на каждый символ не один байт, а два, поэтому с его помощью можно закодировать не 256 символов, а $N = 2^{16} = 65536$ различных символов. Эту кодировку поддерживают последние версии платформы Microsoft Windows&Office (начиная с 1997 года).

Таблица 2.3. Кодировки символов

Двоичный код	Десятичный код	КОИ8	CP1251	CP866	Mac	ISO
00000000	0					
.....						
00001000	8	Удаление последнего символа (клавиша Backspace)				
.....						
00001101	13	Перевод строки (клавиша Enter)				
.....						
00100000	32	Пробел				
00100001	33	!				
.....						
01011010	90	Z				
.....						
01111111	127	▯				
10000000	128	-	Ъ	А	А	к
.....						
11000010	194	б	В	-	-	Т
.....						
11001100	204	л	М	|	|	Ь
.....						
11011101	221	щ	Э	_	Е	н
.....						
11111111	255	ь	я	Нераздел. пробел	Нераздел. пробел	п

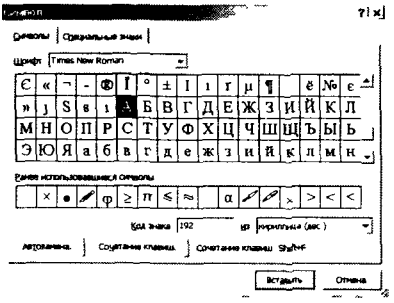
Каждая кодировка задается своей собственной кодовой таблицей. Как видно из табл. 2.3, одному и тому же двоичному коду в различных кодировках поставлены в соответствие различные символы.

Например, последовательность числовых кодов 221, 194, 204 в кодировке CP1251 образует слово «ЭВМ», тогда как в других кодировках это будет бессмысленный набор символов.

К счастью, в большинстве случаев пользователь не должен заботиться о перекодировках текстовых документов, так как это делают специальные программы-конверторы, встроенные в приложения.

Определение числового кода символа

1. Запустить текстовый редактор MS Word 2002. Ввести команду [Вставка-Символ...]. На экране появится диалоговая панель *Символ*. Центральную часть диалогового окна занимает таблица символов для определенного шрифта (например, Times New Roman).



Символы располагаются последовательно слева направо и построчно, начиная с символа *Пробел* в левом верхнем углу и кончая буквой «я» в правом нижнем углу таблицы.

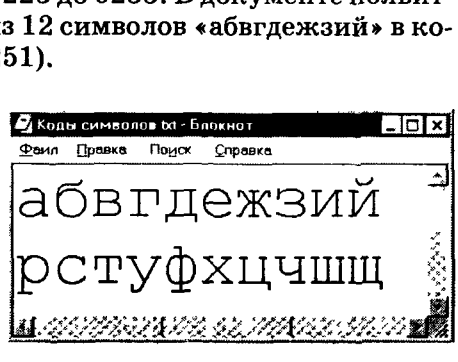
Выбрать символ и в раскрывающемся списке *из:* тип кодировки.

В текстовом поле *Код знака:* появится его числовой код.

Ввод символов по числовому коду

1. Запустить стандартную программу Блокнот. С помощью дополнительной цифровой клавиатуры при нажатой клавише {*Alt*} ввести число 0224, отпустить клавишу {*Alt*}. В документе появится символ «а». Повторить процедуру для числовых кодов от 0225 до 0233. В документе появится последовательность из 12 символов «абвгдежзий» в кодировке Windows (CP1251).

2. С помощью дополнительной цифровой клавиатуры при нажатой клавише {*Alt*} ввести число 224, в документе появится символ «р». Повторить процедуру для числовых кодов от 225 до 233, в документе появится последовательность из 12 символов «рстуфхцчшщ» в кодировке MS-DOS (CP866).



Практические задания

2.29. Используя таблицу символов (MS Word), записать последовательность десятичных числовых кодов в кодировке Windows (CP1251) для слова «компьютер».

2.30. Используя Блокнот, определить, какое слово в кодировке Windows (CP1251) задано последовательностью числовых кодов: 225, 224, 233, 242.

2.31. Какие последовательности букв будут в кодировках КОИ8 и ISO соответствовать слову «ЭВМ», записанному в кодировке CP1251?

2.11. Аналоговый и дискретный способы представления изображений и звука

Человек способен воспринимать и хранить информацию в форме образов (зрительных, звуковых, осязательных, вкусовых и обонятельных). Зрительные образы могут быть сохранены в виде изображений (рисунков, фотографий и так далее), а звуковые — зафиксированы на пластинках, магнитных лентах, лазерных дисках и так далее.

Информация, в том числе графическая и звуковая, может быть представлена в *аналоговой* или *дискретной* форме. При аналоговом представлении физическая величина принимает бесконечное множество значений, причем ее значения изменяются непрерывно. При дискретном представлении физическая величина принимает конечное множество значений, причем ее величина изменяется скачкообразно.

Приведем пример аналогового и дискретного представления информации. Положение тела на наклонной плоскости и на лестнице задается значениями координат X и Y. При движении тела по наклонной плоскости его координаты могут принимать бесконечное множество непрерывно изменяющихся значений из определенного диапазона, а при движении по лестнице — только определенный набор значений, причем меняющихся скачкообразно (рис. 2.6).

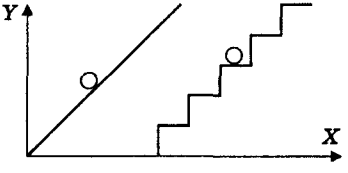


Рис. 2.6
Аналоговое и дискретное кодирование

Примером аналогового представления графической информации может служить, например, живописное полотно, цвет которого изменяется непрерывно, а дискретного — изображение, напечатанное с помощью струйного принтера и состоящее из отдельных точек разного цвета. Примером аналогового хранения звуковой информации является виниловая пластинка (звуковая дорожка изменяет свою форму непрерывно), а дискретного — аудиокомпакт-диск (звуковая дорожка которого содержит участки с различной отражающей способностью).

Преобразование графической и звуковой информации из аналоговой формы в дискретную производится путем *дискретизации*, то есть разбиения непрерывного графического изображения и непрерывного (аналогового) звукового сигнала на отдельные элементы. В процессе дискретизации производится кодирование, то есть присвоение каждому элементу конкретного значения в форме кода.

Дискретизация — это преобразование непрерывных изображений и звука в набор дискретных значений в форме кодов.

Вопросы для размышления

1. Приведите примеры аналогового и дискретного способов представления графической и звуковой информации.
2. В чем состоит суть процесса дискретизации?

2.12. Двоичное кодирование графической информации

Пространственная дискретизация. В процессе кодирования изображения производится его пространственная дискретизация. Пространственную дискретизацию изображения можно сравнить с построением изображения из мозаики (большого количества маленьких разноцветных стекол). Изображение разбивается на отдельные маленькие фрагменты (точки), причем каждому фрагменту присваивается значение его цвета, то есть код цвета (красный, зеленый, синий и так далее) — рис. 2.7.

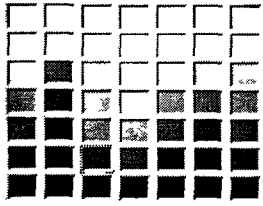


Рис. 2.7
Пространственная дискретизация изображения

Качество кодирования изображения зависит от двух параметров. Во-первых, качество кодирования изображения тем выше, чем меньше размер точки и соответственно большее количество точек составляет изображение.

Во-вторых, чем большее количество цветов, то есть большее количество возможных состояний точки изображения, используется, тем более качественно кодируется изображение (каждая точка несет большее количество информации). Совокупность используемых в наборе цветов образует *палитру цветов*.

Формирование растрового изображения. Графическая информация на экране монитора представляется в виде *растрового изображения*, которое формируется из определенного количества строк, которые в свою очередь содержат определенное количество точек (пикселей).

Качество изображения определяется разрешающей способностью монитора, т.е. количеством точек, из которых оно складывается. Чем больше разрешающая способность, то есть чем больше количество строк растра и точек в строке, тем выше качество изображения. В современных персональных компьютерах обычно используются три основные разрешающие способности экрана: 800 × 600, 1024 × 768 и 1280 × 1024 точки.

Рассмотрим формирование на экране монитора растрового изображения, состоящего из 600 строк по 800 точек в каждой строке (всего 480 000 точек). В простейшем случае (черно-белое изображение без градаций серого цвета) каждая точка экрана может иметь одно из двух состояний — «черная» или «белая», то есть для хранения ее состояния необходим 1 бит.

Цветные изображения формируются в соответствии с двоичным кодом цвета каждой точки, хранящимся в видеопамяти (рис. 2.8). Цветные изображения могут иметь различную *глубину цвета*, которая задается количеством битов, используемым для кодирования цвета точки. Наиболее распространенными значениями глубины цвета являются 8, 16, 24 или 32 бита.

Видеопамять	
№ точки	Двоичный код цвета точки
1	01010101
2	00100101
.	
800	11110000
.	
480000	11001100

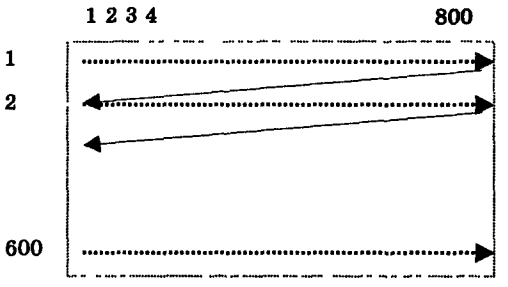
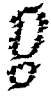


Рис. 2.8. Формирование растрового изображения



Качество двоичного кодирования изображения определяется **разрешающей способностью** экрана и **глубиной цвета**.

Каждый цвет можно рассматривать как возможное состояние точки, тогда количество цветов, отображаемых на экране монитора, может быть вычислено по формуле (2.1):

$$N = 2^I,$$

где I — глубина цвета (табл. 2.4).

Таблица 2.4. Глубина цвета и количество отображаемых цветов

Глубина цвета (I)	Количество отображаемых цветов (N)
8	$2^8 = 256$
16 (High Color)	$2^{16} = 65\,536$
24 (True Color)	$2^{24} = 16\,777\,216$
32 (True Color)	$2^{32} = 4\,294\,967\,296$

Цветное изображение на экране монитора формируется за счет смешивания трех базовых цветов: красного, зеленого и синего. Такая цветовая модель называется RGB-моделью по первым буквам английских названий цветов (Red, Green, Blue).

Для получения богатой палитры цветов базовым цветам могут быть заданы различные интенсивности. Например, при глубине цвета в 24 бита на каждый из цветов выделяется по 8 бит, то есть для каждого из цветов возможны $N = 2^8 = 256$ уровней интенсивности, заданные двоичными кодами (от минимальной — 00000000 до максимальной — 11111111) — табл. 2.5.

Таблица 2.5. Формирование цветов при глубине цвета 24 бита

Название цвета	Интенсивность		
	Красный	Зеленый	Синий
Черный	00000000	00000000	00000000
Красный	11111111	00000000	00000000
Зеленый	00000000	11111111	00000000
Синий	00000000	00000000	11111111
Голубой	00000000	11111111	11111111
Желтый	11111111	11111111	00000000
Белый	11111111	11111111	11111111

Графический режим. Графический режим вывода изображения на экран монитора определяется величиной разрешающей способности и глубиной цвета. Для того чтобы на экране монитора формировалось изображение, информация о каждой его точке (код цвета точки) должна храниться в видеопамяти компьютера. Рассчитаем необходимый объем видеопамяти для одного из графических режимов, например, с разрешением 800 × 600 точек и глубиной цвета 24 бита на точку.

Всего точек на экране: 800 · 600 = 480 000.

Необходимый объем видеопамяти:

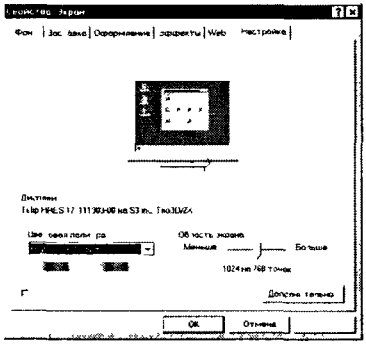
24 бит · 480 000 = 11 520 000 бит = 1 440 000 байт = = 1406,25 Кбайт = 1,37 Мбайт.

Аналогично рассчитывается необходимый объем видеопамяти для других графических режимов.

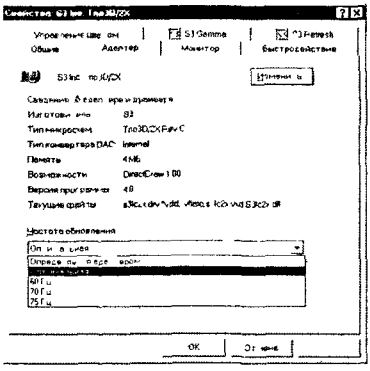
В Windows предусмотрена возможность выбора графического режима и настройки параметров видеосистемы компьютера, включающей монитор и видеоадаптер.

Установка графического режима

1. Щелкнуть по индикатору *Экран* на *Панели задач*, появится диалоговая панель *Свойства: Экран*.
   Выбрать вкладку *Настройка*, которая информирует нас о марке установленных монитора и видеоадаптера и предоставляет возможность установить графический режим экрана (глубину цвета и разрешающую способность).



2. Щелкнуть по кнопке *Дополнительно*, появится диалоговая панель, на которой выбрать вкладку *Адаптер*.
На вкладке имеется информация о фирме-производителе, марке видеоадаптера, объеме видеопамяти и др.
С помощью раскрывающегося списка можно выбрать оптимальную частоту обновления экрана.



Вопросы для размышления

1. В чем состоит суть метода пространственной дискретизации?
2. Объясните принцип формирования растрового изображения.
3. Какими параметрами задается графический режим, в котором изображения выводятся на экран монитора?

Задания

2.32. Используются графические режимы с глубинами цвета 8, 16, 24 и 32 бита. Вычислить объемы видеопамяти, необходимые для реализации данных глубин цвета при различных разрешающих способностях экрана.

2.13. Двоичное кодирование звуковой информации

Временна́я дискретизация звука. Звук представляет собой звуковую волну с непрерывно меняющейся амплитудой и частотой. Чем больше амплитуда сигнала, тем он громче для человека, чем больше частота сигнала, тем выше тон. Для того чтобы компьютер мог обрабатывать звук, непрерывный звуковой сигнал должен быть превращен в последовательность электрических импульсов (двоичных нулей и единиц).

В процессе кодирования непрерывного звукового сигнала производится его *временна́я дискретизация*. Непрерывная звуковая волна разбивается на отдельные маленькие временны́е участки, причем для каждого такого участка устанавливается определенная величина амплитуды.

Таким образом, непрерывная зависимость амплитуды сигнала от времени $A(t)$ заменяется на дискретную последовательность уровней громкости. На графике это выглядит как замена гладкой кривой на последовательность «ступенек» — рис. 2.9.

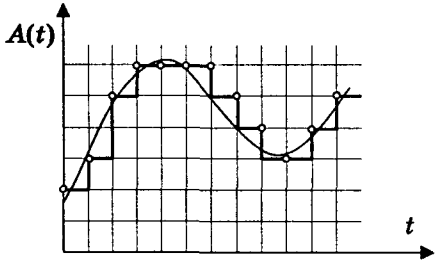


Рис. 2.9
Временна́я дискретизация звука

Каждой «ступеньке» присваивается значение уровня громкости звука, его код (1, 2, 3 и так далее). Уровни громкости звука можно рассматривать как набор возможных состояний, соответственно, чем большее количество уровней громкости будет выделено в процессе кодирования, тем большее количество информации будет нести значение каждого уровня и тем более качественным будет звучание.

Современные звуковые карты обеспечивают 16-битную глубину кодирования звука. Количество различных уровней сигнала (состояний при данном кодировании) можно рассчитать по формуле (2.1):

$$N = 2^I = 2^{16} = 65536,$$

где I — глубина звука.

Таким образом, современные звуковые карты могут обеспечить кодирование 65536 уровней сигнала. Каждому значению амплитуды звукового сигнала присваивается 16-битный код.

При двоичном кодировании непрерывного звукового сигнала он заменяется последовательностью дискретных уровней сигнала. Качество кодирования зависит от количества измерений уровня сигнала в единицу времени, то есть *частоты дискретизации*. Чем большее количество измерений производится за 1 секунду (чем больше частота дискретизации), тем точнее процедура двоичного кодирования.

Качество двоичного кодирования звука определяется *глубиной кодирования* и *частотой дискретизации*.

Количество измерений в секунду может лежать в диапазоне от 8000 до 48 000, то есть частота дискретизации аналогового звукового сигнала может принимать значения от 8 до 48 кГц. При частоте 8 кГц качество дискретизированного звукового сигнала соответствует качеству радиотрансляции, а при частоте 48 кГц — качеству звучания аудио-CD. Следует также учитывать, что возможны как моно-, так и стереорежимы.

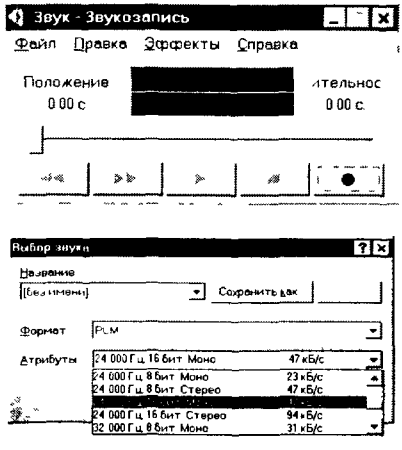
Можно оценить информационный объем стереоаудиофайла длительностью звучания 1 секунда при высоком качестве звука (16 битов, 48 кГц). Для этого количество битов, приходящихся на одну выборку, необходимо умножить на количество выборок в 1 секунду и умножить на 2 (стерео):

$$16 \text{ бит} \cdot 48\,000 \cdot 2 = 1\,536\,000 \text{ бит} = 192\,000 \text{ байт} = 187{,}5 \text{ Кбайт}.$$

Стандартное приложение Звукозапись играет роль цифрового магнитофона и позволяет записывать звук, то есть дискретизировать звуковые сигналы, и сохранять их в звуковых файлах в формате WAV. Эта программа позволяет редактировать звуковые файлы, микшировать их (накладывать друг на друга), а также воспроизводить.

Запись звукового файла

1. Запустить Звукозапись. Для установки параметров дискретизации звука ввести команду [Файл-Свойства]. На панели *Свойства объекта «Звук»* щелкнуть по кнопке *Преобразовать*.
2. На панели *Выбор звука* из раскрывающегося списка выбрать режим кодирования звука (глубина кодирования, частота дискретизации, моно/стерео).



Вопросы для размышления

1. В чем состоит принцип двоичного кодирования звука?
2. От каких параметров зависит качество двоичного кодирования звука?

Практические задания

2.33. С помощью программы Звукозапись записать при 16-битном кодировании и частоте дискретизации 44 кГц моноаудиофайл длительностью 10 секунд. Сравнить его реальный объем с вычисленным.

2.14. Хранение информации

Информация, закодированная с помощью естественных и формальных языков, а также информация в форме зрительных и звуковых образов хранится в памяти человека. Однако для долговременного хранения информации, ее накопления и передачи из поколения в поколение используются *носители информации.*

Материальная природа носителей информации может быть различной: молекулы ДНК, которые хранят генетическую информацию; бумага, на которой хранятся тексты и изображения; магнитная лента, на которой хранится звуковая информация; фото- и кинопленки, на которых хранится графическая информация; микросхемы памяти, магнитные и лазерные диски, на которых хранятся программы и данные в компьютере, и так далее.

По оценкам специалистов, объем информации, фиксируемой на различных носителях, превышает один эксабайт в год (10^{18} байт/год). Примерно 80% всей этой информации хранится в цифровой форме на магнитных и оптических носителях и только 20% — на аналоговых носителях (бумага, магнитные ленты, фото- и кинопленки). Если всю записанную в 2000 году информацию распределить на всех жителей планеты, то на каждого человека придется по 250 Мбайт, а для ее хранения потребуется 85 миллионов жестких магнитных дисков по 20 Гбайт.

Информационная емкость носителей информации. Носители информации характеризуются информационной емкостью, то есть количеством информации, которое они могут хранить. Наиболее информационно емкими являются молекулы ДНК, которые имеют очень малый размер и плотно упакованы. Это позволяет хранить огромное количество информации (до 10^{21} битов в 1 $см^3$), что дает возможность организму развиваться из одной-единственной клетки, содержащей всю необходимую генетическую информацию.

Современные микросхемы памяти позволяют хранить в 1 $см^3$ до 10^{10} битов информации, однако это в 100 миллиардов раз меньше, чем в ДНК. Можно сказать, что современные технологии пока существенно проигрывают биологической эволюции.

Однако если сравнивать информационную емкость традиционных носителей информации (книг) и современных компьютерных носителей, то прогресс очевиден. На каждом гибком магнитном диске может храниться книга объемом около 600 страниц, а на жестком магнитном диске или DVD — целая библиотека, включающая десятки тысяч книг.

Надежность и долговременность хранения информации. Большое значение имеет надежность и долговременность хранения информации. Большую устойчивость к возможным повреждениям имеют молекулы ДНК, так как существует механизм обнаружения повреждений их структуры (мутаций) и самовосстановления.

Надежность (устойчивость к повреждениям) достаточно высока у аналоговых носителей, повреждение которых приводит к потери информации только на поврежденном участке. Поврежденная часть фотографии не лишает возможности видеть оставшуюся часть, повреждение участка магнитной ленты приводит лишь к временному пропаданию звука и так далее.

Цифровые носители гораздо более чувствительны к повреждениям, даже утеря одного бита данных на магнитном или оптическом диске может привести к невозможности считать файл, то есть к потере большого объема данных. Именно поэтому необходимо соблюдать правила эксплуатации и хранения цифровых носителей информации.

Наиболее долговременным носителем информации является молекула ДНК, которая в течение десятков тысяч лет (человек) и миллионов лет (некоторые живые организмы), сохраняет генетическую информацию данного вида.

Аналоговые носители способны сохранять информацию в течение тысяч лет (египетские папирусы и шумерские глиняные таблички), сотен лет (бумага) и десятков лет (магнитные ленты, фото- и кинопленки).

Цифровые носители появились сравнительно недавно и поэтому об их долговременности можно судить только по оценкам специалистов. По экспертным оценкам, при правильном хранении оптические носители способны хранить информацию сотни лет, а магнитные — десятки лет.

Вопросы для размышления

1. Какие достоинства и недостатки имеют аналоговые и цифровые носители информации?

Задания

2.34. Составить таблицу сравнения различных типов носителей информации (аналоговых и цифровых) по их возможностям хранения информации.

Глава 3

Основы логики и логические основы компьютера

3.1. Формы мышления

Первые учения о формах и способах рассуждений возникли в странах Древнего Востока (Китай, Индия), но в основе современной логики лежат учения, созданные древнегреческими мыслителями. Основы формальной логики заложил Аристотель, который впервые отделил логические формы мышления (речи) от его содержания.

Логика — это наука о формах и способах мышления.

Законы логики отражают в сознании человека свойства, связи и отношения объектов окружающего мира. Логика позволяет строить формальные модели окружающего мира, отвлекаясь от содержательной стороны.

Мышление всегда осуществляется в каких-то формах. Основными формами мышления являются *понятие, высказывание* и *умозаключение.*

Понятие. Понятие выделяет существенные признаки объекта, которые отличают его от других объектов. Объекты, объединенные понятием, образуют некоторое множество. Например, понятие «компьютер» объединяет множество электронных устройств, которые предназначены для обработки информации и обладают монитором и клавиатурой. Даже по этому короткому описанию компьютер трудно спутать с другими объектами, например с механизмами, служащими для перемещения по дорогам и хранящимися в гаражах, которые объединяются понятием «автомобиль».

Понятие – это форма мышления, фиксирующая основные, существенные признаки объекта.

Понятие имеет две стороны: *содержание* и *объем*. Содержание понятия составляет совокупность существенных признаков объекта. Чтобы раскрыть содержание понятия, следует найти признаки, необходимые и достаточные для выделения данного объекта из множества других объектов.

Например, содержание понятия «персональный компьютер» можно раскрыть следующим образом: «Персональный компьютер — это универсальное электронное устройство для автоматической обработки информации, предназначенное для одного пользователя».

Объем понятия определяется совокупностью предметов, на которую оно распространяется. Объем понятия «персональный компьютер» выражает всю совокупность (сотни миллионов) существующих в настоящее время в мире персональных компьютеров.

Высказывание. Свое понимание окружающего мира человек формулирует в форме *высказываний* (суждений, утверждений). Высказывание строится на основе понятий и по форме является повествовательным предложением.

Высказывания могут быть выражены с помощью не только естественных языков, но и формальных. Например, высказывание на естественном языке имеет вид «Два умножить на два равно четырем», а на формальном, математическом языке оно записывается в виде: «$2 \cdot 2 = 4$».

Об объектах можно судить верно или неверно, то есть высказывание может быть *истинным* или *ложным*. Истинным будет высказывание, в котором связь понятий правильно отражает свойства и отношения реальных вещей. Примером истинного высказывания может служить следующее: «Процессор является устройством обработки информации».

Ложным высказывание будет в том случае, когда оно не соответствует реальной действительности, например: «Процессор является устройством печати».

Высказывание не может быть выражено повелительным или вопросительным предложением, так как оценка их истинности или ложности невозможна.

Конечно, иногда истинность того или иного высказывания является относительной. Истинность высказываний может зависеть от взглядов людей, от конкретных обстоятельств и так далее. Сегодня высказывание «На моем компьютере уста-

новлен самый современный процессор Pentium 4» истинно, но пройдет некоторое время, появится более мощный процессор, и данное высказывание станет ложным.

Высказывание – это форма мышления, в которой что-либо утверждается или отрицается о свойствах реальных предметов и отношениях между ними. Высказывание может быть либо **истинно**, либо **ложно**.

До сих пор мы рассматривали простые высказывания. На основании простых высказываний могут быть построены *составные высказывания*. Например, высказывание «Процессор является устройством обработки информации и принтер является устройством печати» является составным высказыванием, состоящим из двух простых, соединенных союзом «и».

Если истинность или ложность простых высказываний устанавливается в результате соглашения на основании здравого смысла, то истинность или ложность составных высказываний вычисляется с помощью использования *алгебры высказываний*.

Приведенное выше составное высказывание истинно, так как истинны входящие в него простые высказывания.

Умозаключение. Умозаключения позволяют на основе известных фактов, выраженных в форме суждений (высказываний), получать заключение, то есть новое знание. Примером умозаключений могут быть геометрические доказательства.

Например, если мы имеем суждение «Все углы треугольника равны», то мы можем путем умозаключения доказать, что в этом случае справедливо суждение «Этот треугольник равносторонний».

Умозаключение – это форма мышления, с помощью которой из одного или нескольких суждений (посылок) может быть получено новое суждение (заключение).

Посылками умозаключения по правилам формальной логики могут быть только истинные суждения. Тогда, если

умозаключение проводится в соответствии с правилами формальной логики, то оно будет истинным. В противном случае можно прийти к ложному умозаключению.

Вопросы для размышления

1. Какие существуют основные формы мышления?
2. В чем состоит разница между содержанием и объемом понятия?
3. Может ли быть высказывание выражено в форме вопросительного предложения?
4. Как определяется истинность или ложность простого высказывания? Составного высказывания?

3.2. Алгебра высказываний

Алгебра высказываний была разработана для того, чтобы можно было определять истинность или ложность составных высказываний, не вникая в их содержание.

В алгебре высказываний суждениям (простым высказываниям) ставятся в соответствие *логические переменные*, обозначаемые прописными буквами латинского алфавита. Рассмотрим два простых высказывания:

A = «Два умножить на два равно четырем».

B = «Два умножить на два равно пяти».

Высказывания, как уже говорилось ранее, могут быть истинными или ложными. Истинному высказыванию соответствует значение логической переменной 1, а ложному — значение 0. В нашем случае первое высказывание истинно ($A = 1$), а второе ложно ($B = 0$).

В алгебре высказываний **высказывания** обозначаются именами логических переменных, которые могут принимать лишь два значения: «истина» (1) и «ложь» (0).

В алгебре высказываний над высказываниями можно производить определенные логические операции, в результате которых получаются новые, составные высказывания.

Для образования новых высказываний наиболее часто используются базовые логические операции, выражаемые с помощью логических связок «и», «или», «не».

3.2.1. Логическое умножение (конъюнкция)

Объединение двух (или нескольких) высказываний в одно с помощью союза «и» называется *операцией логического умножения* или *конъюнкцией*.

Составное высказывание, образованное в результате **операции логического умножения (конъюнкции)**, истинно тогда и только тогда, когда истинны все входящие в него простые высказывания.

Так, из приведенных ниже четырех составных высказываний, образованных с помощью операции логического умножения, истинно только четвертое, так как в первых трех составных высказываниях хотя бы одно из простых высказываний ложно:

(1) «$2 \cdot 2 = 5$ и $3 \cdot 3 = 10$»,
(2) «$2 \cdot 2 = 5$ и $3 \cdot 3 = 9$»,
(3) «$2 \cdot 2 = 4$ и $3 \cdot 3 = 10$»,
(4) «$2 \cdot 2 = 4$ и $3 \cdot 3 = 9$».

Перейдем теперь от записи высказываний на естественном языке к их записи на формальном языке алгебры высказываний (алгебры логики). В ней операцию логического умножения (конъюнкцию) принято обозначать значком «&» либо «$\wedge$». Образуем составное высказывание F, которое получится в результате конъюнкции двух простых высказываний:

$$F = A \,\&\, B.$$

С точки зрения алгебры высказываний мы записали формулу функции логического умножения, аргументами которой являются логические переменные A и B, которые могут принимать значения «истина» (1) и «ложь» (0).

Сама функция логического умножения F также может принимать лишь два значения «истина» (1) и «ложь» (0). Значение логической функции можно определить с помо-

щью *таблицы истинности* данной функции, которая показывает, какие значения принимает логическая функция при всех возможных наборах ее аргументов (табл. 3.1).

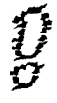
Таблица 3.1. Таблица истинности функции логического умножения

A	*B*	*F = A & B*
0	0	0
0	1	0
1	0	0
1	1	1

По таблице истинности легко определить истинность составного высказывания, образованного с помощью операции логического умножения. Рассмотрим, например, составное высказывание «2 · 2 = 4 и 3 · 3 = 10». Первое простое высказывание истинно ($A = 1$), а второе высказывание ложно ($B = 0$), по таблице определяем, что логическая функция принимает значение ложь ($F = 0$), то есть данное составное высказывание ложно.

3.2.2. Логическое сложение (дизъюнкция)

Объединение двух (или нескольких) высказываний с помощью союза «или» называется *операцией логического сложения* или *дизъюнкцией*.



Составное высказывание, образованное в результате **логического сложения (дизъюнкции)**, истинно тогда, когда истинно хотя бы одно из входящих в него простых высказываний.

Так, из приведенных ниже четырех составных высказываний, образованных с помощью операции логического сложения, ложно только первое, так как в последних трех составных высказываниях хотя бы одно из простых высказываний истинно:

(1) «2 · 2 = 5 или 3 · 3 = 10»,
(2) «2 · 2 = 5 или 3 · 3 = 9»,
(3) «2 · 2 = 4 или 3 · 3 = 10»,
(4) «2 · 2 = 4 или 3 · 3 = 9».

Запишем теперь операцию логического сложения на формальном языке алгебры логики. Операцию логического сложения (дизъюнкцию) принято обозначать либо значком «∨», либо знаком сложения «+». Образуем составное высказывание F, которое получится в результате дизъюнкции двух простых высказываний:

$$F = A \vee B.$$

С точки зрения алгебры высказываний мы записали формулу функции логического сложения, аргументами которой являются логические переменные A и B. Значение логической функции можно определить с помощью таблицы истинности данной функции, которая показывает, какие значения принимает логическая функция при всех возможных наборах ее аргументов (табл. 3.2).

Таблица 3.2. Таблица истинности функции логического сложения

A	B	$F = A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

По таблице истинности легко определить истинность составного высказывания, образованного с помощью операции логического сложения. Рассмотрим, например, составное высказывание «$2 \cdot 2 = 4$ или $3 \cdot 3 = 10$». Первое простое высказывание истинно ($A = 1$), а второе высказывание ложно ($B = 0$), по таблице определяем, что логическая функция принимает значение истина ($F = 1$), то есть данное составное высказывание истинно.

3.2.3. Логическое отрицание (инверсия)

Присоединение частицы «не» к высказыванию называется *операцией логического отрицания* или *инверсией*.

Логическое отрицание (инверсия) делает истинное высказывание ложным и, наоборот, ложное — истинным.

Пусть A = «Два умножить на два равно четырем» — истинное высказывание, тогда высказывание F = «Два умножить на два не равно четырем», образованное с помощью операции логического отрицания, — ложно.

Операцию логического отрицания (инверсию) над логическим высказыванием A **в алгебре логики** принято обозначать $\overline{A}$. Образуем высказывание F, являющееся логическим отрицанием A:

$$F = \overline{A}.$$

Истинность такого высказывания задается таблицей истинности функции логического отрицания (табл. 3.3).

Таблица 3.3. Таблица истинности функции логического отрицания

A	$F=\overline{A}$
0	1
1	0

Истинность высказывания, образованного с помощью операции логического отрицания, можно легко определить с помощью таблицы истинности. Например, высказывание «Два умножить на два не равно четырем» ложно (A = 0), а полученное из него в результате логического отрицания высказывание «Два умножить на два равно четырем» истинно (F = 1).

Задания

3.1. Составить составное высказывание, содержащее операции логического умножения, сложения и отрицания. Определить его истинность.

3.3. Логические выражения и таблицы истинности

Логические выражения. Каждое составное высказывание можно выразить в виде формулы (логического выражения), в которую входят *логические переменные*, обозначающие высказывания, и *знаки логических операций*, обозначающие логические функции.

Для записи составного высказывания в виде логического выражения на формальном языке (языке алгебры логики) в составном высказывании нужно выделить простые высказывания и логические связи между ними.

Запишем в форме логического выражения составное высказывание «(2 · 2 = 5 или 2 · 2 = 4) и (2 · 2 ≠ 5 или 2 · 2 ≠ 4)». Проанализируем составное высказывание. Оно содержит два простых высказывания:

A = «2 · 2 = 5» — ложно (0),

B = «2 · 2 = 4» — истинно (1).

Тогда составное высказывание можно записать в следующей форме:

«(A или B) и ($\overline{A}$ или $\overline{B}$)».

Теперь необходимо записать высказывание в форме логического выражения с учетом последовательности выполнения логических операций. При выполнении логических операций определен следующий порядок их выполнения: инверсия, конъюнкция, дизъюнкция. Для изменения указанного порядка могут использоваться скобки:

$$F = (A \vee B) \,\&\, (\overline{A} \vee \overline{B}).$$

Истинность или ложность составных высказываний можно определять чисто формально, руководствуясь законами алгебры высказываний, не обращаясь к смысловому содержанию высказываний.

Подставим в логическое выражение значения логических переменных и, используя таблицы истинности базовых логических операций, получим значение логической функции:

$$F = (A \vee B)\&(\overline{A} \vee \overline{B}) = (0 \vee 1)\&(1 \vee 0) = 1 \,\&\, 1 = 1.$$

Таблицы истинности. Для каждого составного высказывания (логического выражения) можно построить таблицу истинности, которая определяет его истинность или ложность при всех возможных комбинациях исходных значений простых высказываний (логических переменных).

При построении таблиц истинности целесообразно руководствоваться определенной последовательностью действий.

Во-первых, необходимо определить количество строк в таблице истинности. Оно равно количеству возможных комбинаций значений логических переменных, входящих в логическое выражение. Если количество логических переменных равно n, то:

$$\text{количество строк} = 2^n.$$

В нашем случае логическая функция $F = (A \vee B) \& (\overline{A} \vee \overline{B})$ имеет 2 переменные и, следовательно, количество строк в таблице истинности должно быть равно 4.

Во-вторых, необходимо определить количество столбцов в таблице истинности, которое равно количеству логических переменных плюс количество логических операций.

В нашем случае количество переменных равно двум, а количество логических операций — пяти, то есть количество столбцов таблицы истинности равно семи.

В-третьих, необходимо построить таблицу истинности с указанным количеством строк и столбцов, обозначить столбцы и внести в таблицу возможные наборы значений исходных логических переменных.

В-четвертых, необходимо заполнить таблицу истинности по столбцам, выполняя базовые логические операции в необходимой последовательности и в соответствии с их таблицами истинности (табл. 3.4). Теперь мы можем определить значение логической функции для любого набора значений логических переменных.

Таблица 3.4. Таблица истинности логической функции $F = (A \vee B) \& (\overline{A} \vee \overline{B})$

A	B	$A \vee B$	$\overline{A}$	$\overline{B}$	$\overline{A} \vee \overline{B}$	$(A \vee B) \& (\overline{A} \vee \overline{B})$
0	0	0	1	1	1	0
0	1	1	1	0	1	1
1	0	1	0	1	1	1
1	1	1	0	0	0	0

Равносильные логические выражения. Логические выражения, у которых последние столбцы таблиц истинности совпадают, называются *равносильными*. Для обозначения равносильных логических выражений используется знак «=».

Докажем, что логические выражения $\overline{A} \ \& \ \overline{B}$ и $\overline{A \vee B}$ равносильны. Построим сначала таблицу истинности логического выражения $\overline{A} \ \& \ \overline{B}$ (табл. 3.5).

Таблица 3.5. Таблица истинности логического выражения $\overline{A} \ \& \ \overline{B}$

A	B	$\overline{A}$	$\overline{B}$	$\overline{A} \& \overline{B}$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

Теперь построим таблицу истинности логического выражения $\overline{A \vee B}$ (табл. 3.6).

Таблица 3.6. Таблица истинности логического выражения $\overline{A \vee B}$

A	B	$A \vee B$	$\overline{A \vee B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Значения в последних столбцах таблиц истинности совпадают, следовательно, логические выражения равносильны:

$$\overline{A} \& \overline{B} = \overline{A \vee B}.$$

Вопросы для размышления

1. Что содержат таблицы истинности и каков порядок их построения?
2. Какие логические выражения называются равносильными?

Задания

3.2. Записать составное высказывание «(2 · 2 = 4 и 3 · 3 = 9) или (2 · 2 ≠ 4 и 3 · 3 ≠ 9)» в форме логического выражения. Построить таблицу истинности.

3.3. Доказать, используя таблицы истинности, что логические выражения $\overline{A} \vee \overline{B}$ и $A \& B$ равносильны.

3.4. Логические функции

Любое составное высказывание можно рассматривать как логическую функцию $F(X_1, X_2, ..., X_n)$, аргументами которой являются логические переменные $X_1, X_2, ..., X_n$ (простые высказывания). Сама функция и аргументы могут принимать только два различных значения: «истина» (1) и «ложь» (0).

Выше были рассмотрены функции двух аргументов: логическое умножение $F(A,B) = A\&B$, логическое сложение $F(A,B) = A \vee B$, а также логическое отрицание $F(A) = \overline{A}$, в котором значение второго аргумента можно считать равным нулю.

Каждая логическая функция двух аргументов имеет четыре возможных набора значений аргументов. По формуле (2.1) мы можем определить, какое количество различных логических функций двух аргументов может существовать:

$$N = 2^4 = 16.$$

Таким образом, существует 16 различных логических функций двух аргументов, каждая из которых задается своей таблицей истинности (табл. 3.7).

Таблица 3.7. Таблицы истинности логических функций двух аргументов

Аргументы		Логические функции															
A	B	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Легко заметить, что здесь логическая функция F_2 является функцией логического умножения, F_8 — функцией логического сложения, F_{13} — функцией логического отрицания для аргумента A и F_{11} — функцией логического отрицания для аргумента B.

В обыденной и научной речи кроме базовых логических связок «и», «или», «не» используются и некоторые другие: «если... то...», «... тогда и только тогда, когда...» и др. Некоторые из них имеют свое название и свой символ, и им соответствуют определенные логические функции.

Логическое следование (импликация). Логическое следование (импликация) образуется соединением двух высказываний в одно с помощью оборота речи «если..., то...».

Логическая операция импликации «если A, то B», обозначается $A \rightarrow B$ и выражается с помощью логической функции F_{14}, которая задается соответствующей таблицей истинности (табл. 3.8).

Таблица 3.8. Таблица истинности логической функции «импликация»

A	B	$F_{14} = A \to B$
0	0	1
0	1	1
1	0	0
1	1	1

Составное высказывание, образованное с помощью **операции логического следования (импликации)**, ложно тогда и только тогда, когда из истинной предпосылки (первого высказывания) следует ложный вывод (второе высказывание).

Например, высказывание «Если число делится на 10, то оно делится на 5» истинно, так как истинны и первое высказывание (предпосылка), и второе высказывание (вывод).

Высказывание «Если число делится на 10, то оно делится на 3» ложно, так как из истинной предпосылки делается ложный вывод.

Однако операция логического следования несколько отличается от обычного понимания слова «следует». Если первое высказывание (предпосылка) ложно, то вне зависимости от истинности или ложности второго высказывания (вывода) составное высказывание истинно. Это можно понимать таким образом, что из неверной предпосылки может следовать что угодно.

В алгебре высказываний все логические функции могут быть сведены путем логических преобразований к трем базовым: логическому умножению, логическому сложению и логическому отрицанию.

Докажем методом сравнения таблиц истинности (табл. 3.8 и 3.9), что операция импликации $A \to B$ равносильна логическому выражению $\overline{A} \vee B$.

Таблица 3.9. Таблица истинности логического выражения $\overline{A} \vee B$

A	B	$\overline{A}$	$\overline{A} \vee B$
0	0	1	1
0	1	1	1
1	0	0	0
1	1	0	1

Таблицы истинности совпадают, что и требовалось доказать.

Логическое равенство (эквивалентность). Логическое равенство (эквивалентность) образуется соединением двух высказываний в одно с помощью оборота речи «... тогда и только тогда, когда ...».

Логическая операция эквивалентности «*A* тогда и только тогда, когда *B*» обозначается *A*~*B* и выражается с помощью логической функции F_{10}, которая задается соответствующей таблицей истинности (табл. 3.10).

Таблица 3.10. Таблица истинности логической функции эквивалентности

A	*B*	F_{10}
0	0	1
0	1	0
1	0	0
1	1	1

Составное высказывание, образованное с помощью **логической операции эквивалентности** истинно тогда и только тогда, когда оба высказывания одновременно либо ложны, либо истинны.

Рассмотрим, например, два высказывания: *A* = «Компьютер может производить вычисления» и *B* = «Компьютер включен». Составное высказывание, полученное с помощью операции эквивалентности, истинно, когда оба высказывания либо истинны, либо ложны:

«Компьютер может производить вычисления тогда и только тогда, когда компьютер включен».

«Компьютер не может производить вычисления тогда и только тогда, когда компьютер не включен».

Составное высказывание, полученное с помощью операции эквивалентности, ложно, когда одно высказывание истинно, а другое — ложно:

«Компьютер может производить вычисления тогда и только тогда, когда компьютер не включен».

«Компьютер не может производить вычисления тогда и только тогда, когда компьютер включен».

Вопросы для размышления

1. Какое количество логических функций двух аргументов существует и почему?
2. Какие логические функции двух аргументов имеют свои названия?
3. Какое существует количество логических функций трех аргументов?

Задания

3.4. Доказать, используя таблицы истинности, что операция эквивалентности $A \sim B$ равносильна логическому выражению: $(A \vee \overline{B}) \& (\overline{A} \vee B)$.

3.5. Логические законы и правила преобразования логических выражений

Законы логики отражают наиболее важные закономерности логического мышления. В алгебре высказываний законы логики записываются в виде формул, которые позволяют проводить эквивалентные преобразования логических выражений.

Закон тождества. Всякое высказывание тождественно самому себе:

$$A = A$$

Закон непротиворечия. Высказывание не может быть одновременно истинным и ложным. Если высказывание A истинно, то его отрицание *не* A должно быть ложным. Следовательно, логическое произведение высказывания и его отрицания должно быть ложно:

$$A \& \overline{A} = 0$$

Закон исключенного третьего. Высказывание может быть либо истинным, либо ложным, третьего не дано. Это означает, что результат логического сложения высказывания и его отрицания всегда принимает значение «истина»:

$$A \vee \overline{A} = 1$$

Закон двойного отрицания. Если дважды отрицать некоторое высказывание, то в результате мы получим исходное высказывание:

$$\overline{\overline{A}} = A$$

Законы де Моргана.

$$\overline{A \vee B} = \overline{A} \,\&\, \overline{B}$$
$$\overline{A \,\&\, B} = \overline{A} \vee \overline{B}$$

Важное значение для выполнения преобразований логических выражений имеют законы алгебраических преобразований. Многие из них имеют аналоги в обычной алгебре.

Закон коммутативности. В обычной алгебре слагаемые и множители можно менять местами. В алгебре высказываний можно менять местами логические переменные при операциях логического умножения и логического сложения:

Логическое умножение	Логическое сложение
$A \,\&\, B = B \,\&\, A$	$A \vee B = A \vee B$

Закон ассоциативности. Если в логическом выражении используются только операция логического умножения или только операция логического сложения, то можно пренебрегать скобками или произвольно их расставлять:

Логическое умножение	Логическое сложение
$(A \,\&\, B) \,\&\, C = A \,\&\, (B \,\&\, C)$	$(A \vee B) \vee C = A \vee (B \vee C)$

Закон дистрибутивности. В отличие от обычной алгебры, где за скобки можно выносить только общие множители, в алгебре высказываний можно выносить за скобки как общие множители, так и общие слагаемые:

Дистрибутивность умножения относительно сложения	Дистрибутивность сложения относительно умножения
$ab + ac = a(b+c)$ — в алгебре $(A \,\&\, B) \vee (A \,\&\, C) = A \,\&\, (B \vee C)$	$(A \vee B) \,\&\, (A \vee C) = A \vee (B \,\&\, C)$

Рассмотрим в качестве примера применения законов логики преобразование логического выражения. Пусть нам необходимо упростить логическое выражение:

$$(A \,\&\, B) \vee (A \,\&\, \overline{B}).$$

Воспользуемся законом дистрибутивности и вынесем за скобки A:

$$(A \,\&\, B) \vee (A \,\&\, \overline{B}) = A \,\&\, (B \vee \overline{B}).$$

По закону исключенного третьего $B \vee \overline{B} = 1$, следовательно:

$$A \,\&\, (B \vee \overline{B}) = A \,\&\, 1 = A.$$

Задания

3.5. Доказать справедливость первого $\overline{A \vee B} = \overline{A} \,\&\, \overline{B}$ и второго $\overline{A \,\&\, B} = \overline{A} \vee \overline{B}$ законов де Моргана, используя таблицы истинности.

3.6. Упростить логические выражения:

а) $(A \vee \overline{A}) \,\&\, B$;

б) $A \,\&\, (A \vee B) \,\&\, (B \vee \overline{B})$.

3.6. Решение логических задач

Логические задачи обычно формулируются на естественном языке. В первую очередь их необходимо формализовать, то есть записать на языке алгебры высказываний. Полученные логические выражения необходимо упростить и проанализировать. Для этого иногда бывает необходимо построить таблицу истинности полученного логического выражения.

Условие задачи. В школе-новостройке в каждой из двух аудиторий может находиться либо кабинет информатики, либо кабинет физики. На дверях аудиторий повесили шутливые таблички. На первой повесили табличку «По крайней мере, в одной из этих аудиторий размещается кабинет информатики», а на второй аудитории — табличку с надписью «Кабинет физики находится в другой аудитории». Проверяющему, который пришел в школу, известно только, что надписи на табличках либо обе истинны, либо обе ложны. Помогите проверяющему найти кабинет информатики.

Решение задачи. Переведем условие задачи на язык логики высказываний. Так как в каждой из аудиторий может находиться кабинет информатики, то пусть:

A = «В первой аудитории находится кабинет информатики»;

B = «Во второй аудитории находится кабинет информатики».

Отрицания этих высказываний:

$\overline{A}$ = «В первой аудитории находится кабинет физики»;

$\overline{B}$ = «Во второй аудитории находится кабинет физики».

Высказывание, содержащееся на табличке на двери первой аудитории, соответствует логическому выражению:

$$X = A \vee B.$$

Высказывание, содержащееся на табличке на двери второй аудитории, соответствует логическому выражению:

$$Y = \overline{A}.$$

Содержащееся в условии задачи утверждение о том, что надписи на табличках либо одновременно истинные, либо одновременно ложные в соответствии с законом исключенного третьего записывается следующим образом:

$$(X \,\&\, Y) \vee (\overline{X} \,\&\, \overline{Y}) = 1.$$

Подставим вместо X и Y соответствующие формулы:

$$(X \,\&\, Y) \vee (\overline{X} \,\&\, \overline{Y}) = ((A \vee B) \,\&\, \overline{A}) \vee (\overline{(A \vee B)} \,\&\, \overline{\overline{A}}).$$

Упростим сначала первое слагаемое. В соответствии с законом дистрибутивности умножения относительно сложения:

$$(A \vee B) \,\&\, \overline{A} = A \,\&\, \overline{A} \vee B \,\&\, \overline{A}.$$

В соответствии с законом непротиворечия:

$$A \,\&\, \overline{A} \vee B \,\&\, \overline{A} = 0 \vee B \,\&\, \overline{A}.$$

Упростим теперь второе слагаемое. В соответствии с первым законом де Моргана и законом двойного отрицания:

$$\overline{(A \vee B)} \,\&\, \overline{\overline{A}} = \overline{A} \,\&\, \overline{B} \,\&\, A = \overline{A} \,\&\, A \,\&\, \overline{B}.$$

В соответствии с законом непротиворечия:

$$\overline{A} \,\&\, A \,\&\, \overline{B} = 0 \,\&\, \overline{B} = 0.$$

В результате получаем:

$$(0 \vee B \,\&\, \overline{A}) \vee 0 = B \,\&\, \overline{A}.$$

Полученное логическое выражение оказалось простым и поэтому его можно проанализировать без построения таблицы истинности. Для того чтобы выполнялось равенство $B \,\&\, \overline{A} = 1$, B и $\overline{A}$ должны быть равны 1, то есть соответствующие им высказывания истинны.

Ответ: В первой аудитории находится кабинет физики, а во второй — кабинет информатики.

3.7. В процессе составления расписания уроков учителя высказали свои пожелания. Учитель математики высказал пожелание проводить первый или второй урок, учитель информатики — первый и третий, а учитель физики — второй или третий урок. Сколько существует возможных вариантов расписания и каковы они?

3.7. Логические основы устройства компьютера

3.7.1. Базовые логические элементы

Базовые логические элементы реализуют рассмотренные выше три основные логические операции:

- логический элемент «И» — логическое умножение;
- логический элемент «ИЛИ» — логическое сложение;
- логический элемент «НЕ» — инверсию.

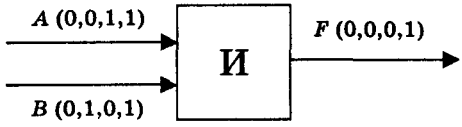
Поскольку любая логическая операция может быть представлена в виде комбинации трех основных, любые устройства компьютера, производящие обработку или хранение информации, могут быть собраны из базовых логических элементов, как из «кирпичиков».

Логические элементы компьютера оперируют с сигналами, представляющими собой электрические импульсы. Есть импульс — логический смысл сигнала — 1, нет импульса — 0. На входы логического элемента поступают сигналы-значения аргументов, на выходе появляется сигнал-значение функции.

Преобразование сигнала логическим элементом задается таблицей состояния, которая фактически является таблицей истинности, соответствующей логической функции.

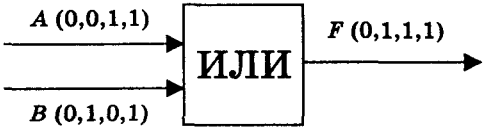
Логический элемент «И». На входы *A* и *B* логического элемента (рис. 3.1) подаются два сигнала (00, 01, 10 или 11). На выходе получается сигнал 0 или 1 в соответствии с таблицей истинности операции логического умножения.

Рис. 3.1
Логический элемент «И»



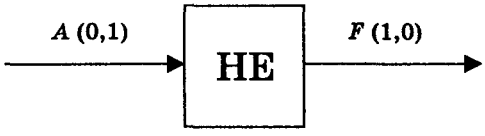
Логический элемент «ИЛИ». На входы *A* и *B* логического элемента (рис. 3.2) подаются два сигнала (00, 01, 10 или 11). На выходе получается сигнал 0 или 1 в соответствии с таблицей истинности операции логического сложения.

Рис. 3.2
Логический элемент «ИЛИ»



Логический элемент «НЕ». На вход *A* логического элемента (рис. 3.3) подается сигнал 0 или 1. На выходе получается сигнал 0 или 1 в соответствии с таблицей истинности инверсии.

Рис. 3.3
Логический элемент «НЕ»



3.7.2. Сумматор двоичных чисел

В целях максимального упрощения работы компьютера все многообразие математических операций в процессоре сводится к сложению двоичных чисел. Поэтому главной частью процессора являются сумматоры, которые как раз и обеспечивают такое сложение.

Полусумматор. Вспомним, что при сложении двоичных чисел в каждом разряде образуется сумма и при этом возможен перенос в старший разряд. Введем обозначения слагаемых (*A*, *B*), переноса (*P*) и суммы (*S*). Таблица сложения од-

норазрядных двоичных чисел с учетом переноса в старший разряд выглядит следующим образом:

Слагаемые		Перенос	Сумма
A	*B*	*P*	*S*
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Из этой таблицы сразу видно, что перенос можно реализовать с помощью операции логического умножения:

$$P = A \,\&\, B.$$

Получим теперь формулу для вычисления суммы. Значения суммы близки к результату операции логического сложения (кроме случая, когда на входы подаются две единицы, а на выходе должен получиться нуль).

Нужный результат достигается, если результат логического сложения умножить на инвертированный перенос. Таким образом, для определения суммы можно применить следующее логическое выражение:

$$S = (A \vee B) \,\&\, \overline{(A \,\&\, B)}.$$

Построим таблицу истинности для данного логического выражения и убедимся в правильности нашего предположения (табл. 3.11).

Таблица 3.11. Таблица истинности логической функции $F = (A \vee B) \& \overline{(A \& B)}$

A	*B*	$A \vee B$	$A \,\&\, B$	$\overline{A \,\&\, B}$	$(A \vee B) \& \overline{(A \,\&\, B)}$
0	0	0	0	1	0
0	1	1	0	1	1
1	0	1	0	1	1
1	1	1	1	0	0

Теперь на основе полученных логических выражений можно построить из базовых логических элементов схему сложения одноразрядных двоичных чисел.

По логической формуле переноса легко определить, что для получения переноса необходимо использовать логический элемент «И».

Анализ логической формулы для суммы показывает, что на выходе должен стоять элемент логического умножения

«И», который имеет два входа. На один из входов надо подать результат логического сложения исходных величин A и B, то есть на него должен подаваться сигнал с элемента логического сложения «ИЛИ».

На второй вход требуется подать результат инвертированного логического умножения исходных сигналов ($\overline{A \,\&\, B}$), то есть на второй вход должен подаваться сигнал с элемента «НЕ», на вход которого должен поступать сигнал с элемента логического умножения «И» (рис. 3.4).

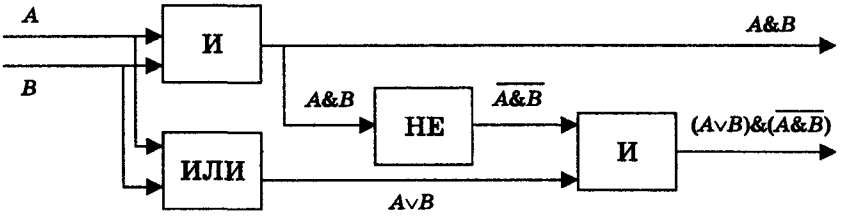


Рис. 3.4. Полусумматор двоичных чисел

Данная схема называется полусумматором, так как реализует суммирование одноразрядных двоичных чисел без учета переноса из младшего разряда.

Полный одноразрядный сумматор. Полный одноразрядный сумматор должен иметь три входа: A, B — слагаемые и P_0 – перенос из младшего разряда и два выхода: сумму S и перенос P. Таблица сложения в этом случае будет иметь следующий вид:

Слагаемые		Перенос из младшего разряда	Перенос	Сумма
A	B	P_0	P	S
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

Идея построения полного сумматора точно такая же, как и полусумматора. Из таблицы сложения видно, что перенос (логическая переменная P) принимает значение 1 тогда, когда хотя бы две входные логические переменные одновременно принимают значение 1. Таким образом, перенос реализуется путем логического сложения результатов попарного логического умножения входных переменных (A, B, P_0). Формула переноса получает следующий вид:

$$P = (A \,\&\, B) \vee (A \,\&\, P_0) \vee (B \,\&\, P_0).$$

Для получения значения суммы (логическая переменная S) необходимо результат логического сложения входных переменных (A, B, P_0) умножить на инвертированный перенос $\overline{P}$:

$$S = (A \vee B \vee P_0) \,\&\, \overline{P}.$$

Данное логическое выражение дает правильные значения суммы во всех случаях, кроме одного, когда на все входные логические переменные принимают значение 1. Действительно:

$$P = (1 \,\&\, 1) \vee (1 \,\&\, 1) \vee (1 \,\&\, 1) = 1;$$
$$S = (1 \vee 1 \vee 1) \,\&\, \overline{P} = 1 \,\&\, 0 = 0.$$

Для получения правильного значения суммы (для данного случая переменная S должна принимать значение 1) необходимо сложить полученное выше выражение для суммы с результатом логического умножения входных переменных (A, B, P_0). В результате логическое выражение для вычисления суммы в полном сумматоре принимает следующий вид:

$$S = (A \vee B \vee P_0) \,\&\, \overline{P}_0 \vee (A \,\&\, B \,\&\, P_0).$$

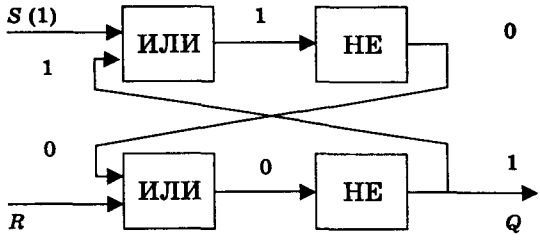
Многоразрядный сумматор. Многоразрядный сумматор процессора состоит из полных одноразрядных сумматоров. На каждый разряд ставится одноразрядный сумматор, причем выход (перенос) сумматора младшего разряда подключается ко входу сумматора старшего разряда.

3.7.3. Триггер

Важнейшей структурной единицей оперативной памяти компьютера, а также внутренних регистров процессора является триггер. Это устройство позволяет запоминать, хранить и считывать информацию (каждый триггер может хранить 1 бит информации).

Триггер можно построить из двух логических элементов «ИЛИ» и двух элементов «НЕ» (рис. 3.5).

Рис. 3.5
Триггер



В обычном состоянии на входы триггера подан сигнал 0, и триггер хранит 0. Для записи 1 на вход S (установочный) подается сигнал 1. Последовательно рассмотрев прохождение сигнала по схеме, видим, что триггер переходит в это состояние и будет устойчиво находиться в нем и после того, как сигнал на входе S исчезнет. Триггер запомнил 1, то есть с выхода триггера Q можно считать 1.

Для того чтобы сбросить информацию и подготовиться к приему новой, подается сигнал 1 на вход R (сброс), после чего триггер возвратится к исходному «нулевому» состоянию.

Задания

3.8. Построить таблицы истинности для логических формул, по которым определяются перенос и сумма полного одноразрядного сумматора.

3.9. Построить схему полного сумматора одноразрядных двоичных чисел с учетом переноса из младшего разряда.

3.10. Проследить по логической схеме триггера, что происходит после поступления сигнала 1 на вход R (сброс).

Глава 4

Основы алгоритмизации и объектно-ориентированного программирования

4.1. Алгоритм и его формальное исполнение

Алгоритм и его свойства. Алгоритмы могут описывать процессы преобразования самых разных объектов. Широкое распространение получили вычислительные алгоритмы, которые описывают преобразование числовых данных. Само слово «алгоритм» происходит от algorithmi — латинской формы написания имени выдающегося математика IX века аль-Хорезми, который сформулировал правила выполнения арифметических операций.

Алгоритм — это строго детерминированная последовательность действий, описывающая процесс преобразования объекта из начального состояния в конечное, записанная с помощью понятных исполнителю команд.

Выберем в качестве объекта текст и построим алгоритм, описывающий процесс его редактирования.

Для того чтобы изменить состояние объекта (значения его свойств), необходимо выполнить над ним определенную последовательность действий (операций). Выполняющий такие операции объект называется исполнителем. Исполнителем редактирования текста может быть человек, компьютер и др.

Алгоритмы состоят из отдельных команд, которые исполнитель выполняет одну за другой в определенной последовательности. Разделение информационного процесса в алго-

ритме на отдельные команды является важным свойством алгоритма и называется дискретностью.

Процесс преобразования текста необходимо разбить на отдельные операции, которые должны быть записаны в виде отдельных команд исполнителю.

Каждый исполнитель обладает определенным набором, системой команд, которые он может выполнить. Алгоритм должен быть понятен исполнителю, то есть должен содержать только те команды, которые входят в систему его команд.

В процессе редактирования текста возможны различные операции: удаление, копирование, перемещение или замена его фрагментов. Исполнитель редактирования текста должен быть в состоянии выполнить эти операции.

Запись алгоритма должна быть такова, чтобы, выполнив очередную команду, исполнитель точно знал, какую команду необходимо исполнять следующей. Это свойство алгоритма называется детерминированностью.

Должны быть определены начальное состояние объекта и его конечное состояние (цель преобразования). Алгоритм должен обеспечивать преобразование объекта из начального состояния в конечное за конечное число шагов. Такое свойство алгоритма называется результативностью.

Следовательно, для текста необходимо задать начальную последовательность символов и конечную последовательность, которая должна быть получена после редактирования.

Формальное выполнение алгоритма. Алгоритм позволяет *формализовать* выполнение информационного процесса. Если исполнителем является человек, то он может выполнять алгоритм формально, не вникая в содержание поставленной задачи, а только строго выполняя последовательность действий, предусмотренную алгоритмом.

Рассмотрим работу пользователя, например, в среде текстового редактора Word. Word предоставляет пользователю возможность работы в мире своих объектов, которыми являются документ, фрагмент документа, символ и так далее.

Предположим, что пользователю необходимо провести редактирование текста. Пусть у нас есть объект — фрагмент. Надо перевести его из исходного состояния (содержание фрагмента — текст «информационная модель», курсор находится перед первым символом) в конечное состояние (текст «модель информационная», курсор находится после последнего символа) — рис. 4.1.

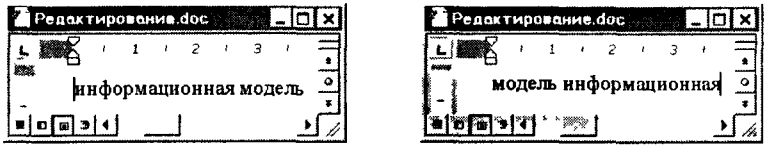


Рис. 4.1. Начальное и конечное состояния объекта

Необходимую для реализации такого преобразования последовательность действий, то есть алгоритм, запишем *на естественном языке*, который понятен пользователю компьютера:

1. Выделить слово *«информационная»* + *пробел.*
2. Вырезать этот фрагмент и поместить его в буфер обмена.
3. Установить курсор на позицию после слова *«модель»* + *пробел.*
4. Вставить вырезанный фрагмент текста.

Каждая команда алгоритма должна однозначно определять действие исполнителя, для этого необходимо формализовать запись алгоритма и заменить содержательную модель текста на его формальную модель. Формальная модель представляет текст делящимся на страницы, состоящие из определенного количество строк, которые, в свою очередь, включают определенное количество знакомест (символов).

Наш текст состоит из одной страницы, которая содержит одну строку. Команде *Выделить слово «информационная»* + *пробел* на формальном языке соответствует команда *Выделить символы с 1 по 15,* а команде *Установить курсор после слова «модель»* + *пробел* соответствует команда *Установить курсор после 7-го символа.*

1. Выделить символы с 1 по 15.
2. Вырезать этот фрагмент и поместить его в буфер обмена.
3. Установить курсор на позицию после 7-го символа.
4. Вставить вырезанный фрагмент текста.

Теперь этот алгоритм *«Редактирование»* пользователь может выполнять формально. В процессе выполнения алгоритма на компьютере пользователь будет выполнять команды алгоритма с помощью клавиатуры и мыши. Фактически же пользователь будет давать команды объектам программной среды Windows&Office, которые будут действительными исполнителями алгоритма.

Компьютер — автоматический исполнитель алгоритмов. Представление информационного процесса в форме алгоритма позволяет поручить его *автоматическое* исполнение различным техническим устройствам, среди которых особое место занимает компьютер. При этом говорят, что компьютер исполняет программу (последовательность команд), реализующую алгоритм.

Алгоритм, записанный на «понятном» компьютеру языке программирования, называется **программой**.

Развитие языков программирования. Информацию в компьютере обрабатывает процессор, следовательно, алгоритм должен быть записан на языке, «понятном» для процессора, то есть на машинном языке, представляющем собой логические последовательности нулей и единиц.

На заре компьютерной эры, в 50-е годы XX века, программы писались на машинном языке и представляли собой очень длинные последовательности нулей и единиц. Составление и отладка таких программ было чрезвычайно трудоемким делом.

В 60–70-е годы для облегчения труда программистов начали создаваться *языки программирования высокого уровня*, формальные языки, кодирующие алгоритмы в привычном для человека виде (в виде предложений). Такие языки программирования строились на основе использования определенного алфавита и строгих правил построения предложений (синтаксиса).

Наиболее широко распространенным типом языков программирования высокого уровня являются *процедурные языки*. В таких языках широко используются управляющие конструкции (операторы), которые позволяют закодировать различные алгоритмические структуры (линейную, ветвление, цикл).

Одним из первых процедурных языков программирования был известный всем Бейсик (Basic), созданный в 1964 году. В течение последующего времени Бейсик развивался, появлялись его различные версии (MSX-Basic, Бейсик-Агат, QBasic и др.). Другим широко распространенным языком программирования алгоритмического типа является Pascal.

В настоящее время наибольшей популярностью пользуются системы объектно-ориентированного визуального программирования Microsoft Visual Basic и Borland Delphi. Для создания приложений в среде Windows&Office используется язык программирования Visual Basic for Applications (VBA).

Вопросы для размышления

1. Какие из нижеперечисленных правил являются алгоритмами? Ответ обоснуйте:
   - орфографические правила;
   - правила выполнения арифметических операций;
   - правила техники безопасности;
   - правила перевода чисел из одной системы счисления в другую.
2. В чем состоит различие между естественными языками и языками программирования?

Задания

4.1. Составьте алгоритм преобразования слова «информатика» в слово «форма».

4.2. Основные типы алгоритмических структур

4.2.1. Линейный алгоритм

Существует большое количество алгоритмов (например, рассмотренный выше алгоритм «*Редактирование*»), в которых команды должны быть выполнены последовательно одна за другой. Такие последовательности команд будем называть *сериями*, а алгоритмы, состоящие из таких серий, *линейными*.

Алгоритм, в котором команды выполняются последовательно одна за другой, называется **линейным алгоритмом**.

Для того чтобы сделать алгоритм более наглядным, часто используют *блок-схемы*.

Различные элементы алгоритма изображаются с помощью различных геометрических фигур: для обозначения начала и конца алгоритма используются прямоугольники с закругленными углами, а для обозначения последовательности команд — прямоугольники (рис. 4.2).

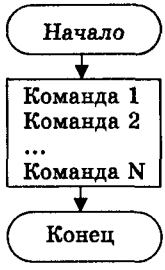


Рис. 4.2. Линейный алгоритм

На блок-схеме хорошо видна структура линейного алгоритма, по которой исполнителю (человеку) удобно отслеживать процесс его выполнения.

4.2.2. Алгоритмическая структура «ветвление»

В отличие от линейных алгоритмов, в которых команды выполняются последовательно одна за другой, в алгоритмическую структуру *«ветвление»* входит *условие*, в зависимости от выполнения или невыполнения которого реализуется та или иная последовательность команд (серия).

В алгоритмической структуре **«ветвление»** та или иная серия команд выполняется в зависимости от истинности **условия**.

Будем называть *условием* высказывание, которое может быть либо истинным, либо ложным. Условие, записанное на формальном языке, называется *условным* или *логическим выражением*.

Условные выражения могут быть *простыми* и *сложными*. Простое условие включает в себя два числа, две переменных или два *арифметических выражения*, которые сравниваются между собой с использованием операций сравнения (равно, больше, меньше и пр.). Например: `5>3`, `2*8=4*4` и т. д.

Сложное условие — это последовательность простых условий, объединенных между собой знаками логических операций. Например, `5>3` **And** `2*8=4*4`.

Алгоритмическая структура *«ветвление»* может быть зафиксирована различными способами:

- графически, с помощью блок-схемы;

- на языке программирования, например на языках Visual Basic и VBA с использованием специальной инструкции ветвления (оператора условного перехода).

После первого ключевого слова (**If**) должно быть размещено условие. После второго ключевого слова (**Then**) последовательность команд (серия 1), которая должна выполняться, если условие принимает значение «истина». После третьего ключевого слова (**Else**) размещается последовательность команд (серия 2), которая должна выполняться, если условие принимает значение «ложь» (рис. 4.3).

Блок-схема	Языки программирования Visual Basic и VBA

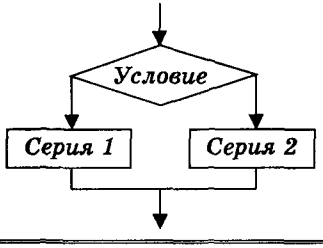
	**If** Условие **Then**
	Серия 1
	[**Else**
	Серия 2]
	End If
	If Условие _
	Then Серия 1 _
	[**Else** Серия 2]

Рис. 4.3. Алгоритмическая структура «ветвление»

Оператор условного перехода может быть записан в *многострочной форме* или в *однострочной форме*.

В многострочной форме он записывается с помощью инструкции **If ... Then ... Else ... End If** (Если ... То ... Иначе ... Конец Если). В этом случае ключевое слово **Then** размещается на той же строчке, что и условие, а последовательность команд (серия 1) — на следующей. Третье ключевое слово **Else** размещается на третьей строчке, а последовательность команд (серия 2) — на четвертой. Конец инструкции ветвления **End If** размещается на пятой строчке.

В однострочной форме он записывается с помощью инструкции **If ... Then ... Else ...** (Если ... То ... Иначе ...). Если инструкция не помещается на одной строке, она может быть разбита на несколько строк. Такое представление инструкций более наглядно для человека. Компьютер же должен знать, что разбитая на строки инструкция представляет единое целое. Это обеспечивает знак «переноса», который задается символом подчеркивания после пробела « _».

Третье ключевое слово **Else** в сокращенной форме инструкции может отсутствовать. (Необязательные части оператора записываются в квадратных скобках — см. табл. 4.3.) Тогда, в случае если условие ложно, выполнение оператора условного перехода заканчивается и выполняется следующая строка программы.

4.2.3. Алгоритмическая структура «выбор»

Алгоритмическая структура *«выбор»* применяется для реализации ветвления со многими вариантами серий команд. В структуру выбора входят несколько *условий*, проверка которых осуществляется в строгой последовательности их записи в команде выбора. При истинности одного из условий выполняется соответствующая последовательность команд.

В алгоритмической структуре «**выбор**» выполняется одна из нескольких последовательностей команд при истинности соответствующего **условия**.

На языках программирования Visual Basic и VBA инструкция выбора начинается с ключевых слов **Select Case**, после которых записывается выражение (переменная, арифметическое выражение и так далее). После ключевых слов **Case** заданное выражение сравнивается с определенными значениями — записываются условия, при истинности одного из которых начинает выполняться серия команд. Заканчивается инструкция ключевыми словами **End Select** (рис. 4.4).

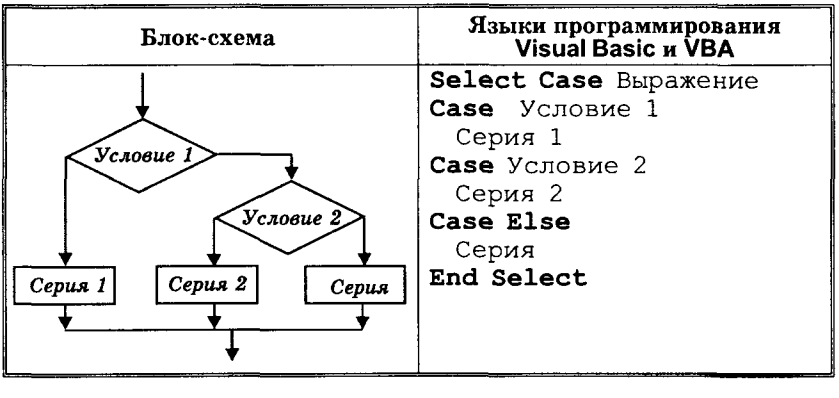


Рис. 4.4. Алгоритмическая структура «выбор»

4.2.4. Алгоритмическая структура «цикл»

В алгоритмическую структуру *«цикл»* входит серия команд, выполняемая *многократно*. Такая последовательность команд называется *телом цикла*.

Циклические алгоритмические структуры бывают двух типов:

- *циклы со счетчиком*, в которых тело цикла выполняется определенное количество раз;
- *циклы с условием*, в которых тело цикла выполняется, пока условие истинно.

В алгоритмической структуре **«цикл»** серия команд (тело цикла) выполняется многократно.

Алгоритмическая структура *«цикл»* может быть зафиксирована различными способами:

- графически — с помощью блок-схемы;
- на языке программирования, например на языках Visual Basic и VBA с использованием специальных инструкций, реализующих циклы различного типа.

Цикл со счетчиком. Когда заранее известно, какое число повторений тела цикла необходимо выполнить, можно воспользоваться циклической инструкцией (оператором цикла со счетчиком) **For ... Next** (рис. 4.5).

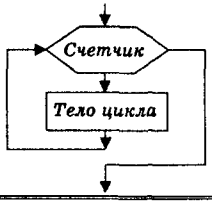
Блок-схема	Языки программирования Visual Basic и VBA
	**For** Счетчик=НачЗнач **To** КонЗнач [**Step** шаг] Тело цикла **Next** [Счетчик]

Рис. 4.5. Цикл со счетчиком

Синтаксис оператора **For ... Next** следующий: строка, начинающаяся с ключевого слова **For**, является заголовком цикла, а строка с ключевым словом **Next** — концом цикла, между ними располагаются операторы, являющиеся телом цикла.

В начале выполнения цикла значение переменной Счетчик устанавливается равным НачЗнач. При каждом проходе цикла переменная Счетчик увеличивается на величину шага. Если она достигает величины, большей КонЗнач, то цикл завершается и выполняются следующие за ним операторы.

Циклы с условием. Часто бывает так, что необходимо повторить тело цикла, но заранее неизвестно, какое количество раз это надо сделать. В таких случаях количество повторений зависит от некоторого условия. Такой цикл реализуется с помощью инструкции **Do ... Loop**.

Условие выхода из цикла можно поставить в начале, перед телом цикла. Такой цикл называется *циклом с предусловием* (рис. 4.6).

Проверка условия выхода из цикла проводится с помощью ключевых слов **While** или **Until**. Эти слова придают одному и тому же условию противоположный смысл. Ключевое слово **While** обеспечивает выполнение цикла, пока выполняется условие, то есть пока условие имеет значение «истина». Как только условие примет значение «ложь», выполнение цикла закончится. В этом случае условие является *условием продолжения цикла*.

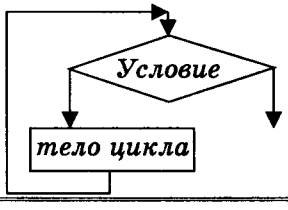
Блок-схема	Языки программирования Visual Basic и VBA
	**Do While** Условие Тело цикла **Loop** **Do Until** Условие Тело цикла **Loop**

Рис. 4.6. Цикл с предусловием

Ключевое слово **Until** обеспечивает выполнение цикла, пока не выполняется условие, то есть пока условие имеет значение «ложь». Как только условие примет значение «истина», выполнение цикла закончится. В этом случае условие является *условием завершения цикла*.

Условие выхода из цикла можно поставить в конце, после тела цикла. Такой цикл называется *циклом с постусловием*. Этот цикл реализуется также с помощью инструкции **Do ... Loop** (рис. 4.7).

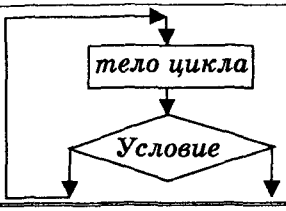
Блок-схема	Языки программирования Visual Basic и VBA
	**Do** Тело цикла **Loop While** Условие **Do** Тело цикла **Loop Until** Условие

Рис. 4.7. Цикл с постусловием

Проверка условия выхода из цикла проводится с помощью ключевых слов **While** или **Until**.

Цикл с постусловием, в отличие от цикла с предусловием, выполняется обязательно как минимум один раз, независимо от того, выполняется условие или нет.

Вопросы для размышления

1. Какой тип алгоритмической структуры необходимо применить, если:
   - последовательность команд должна быть выполнена определенное количество раз;
   - последовательность команд выполняется или не выполняется в зависимости от условия;
   - последовательность команд должна быть обязательно выполнена хотя бы один раз и должна повторяться до тех пор, пока условие справедливо?

Задания

4.2. Составьте и зафиксируйте в форме блок-схемы и на языке программирования алгоритм выбора большего из двух чисел.

4.3. Составьте и зафиксируйте в форме блок-схемы и на языке программирования алгоритм вычисления факториала числа.

4.3. Основы объектно-ориентированного визуального программирования

Приложения на языках объектно-ориентированного программирования Visual Basic и Visual Basic for Applications строятся из объектов, подобно тому, как из блоков и различных деталей строятся дома. Программные библиотеки готовых объектов входят в эти системы программирования, причем языки Visual Basic и VBA различаются между собой, главным образом, составом программных библиотек.

Системы объектно-ориентированного программирования дают возможность визуализировать процесс создания графического интерфейса разрабатываемого приложения, то есть позволяют создавать объекты и задавать значения их свойств с помощью диалоговых окон системы программирования.

Взаимодействие программных объектов между собой и их изменения описываются с помощью программного кода. Создание программного кода в объектно-ориентированном программировании базируется на использовании алгоритмических структур различных типов (линейной, ветвления, цикла), исполнителями которых выступают программные объекты.

4.3.1. Классы объектов, экземпляры класса и семейства объектов

Основной единицей в объектно-ориентированном программировании является программный объект, который объединяет в себе как описывающие его данные (свойства), так и средства обработки этих данных (методы). Если говорить образно, то объекты — это «существительные», свойства объекта — это «прилагательные», а методы объекта — это «глаголы».

Программные объекты обладают *свойствами*, могут использовать *методы* и реагируют на *события*.

Классы объектов. *Классы объектов* являются «шаблонами», определяющими наборы свойств, методов и событий. По этим шаблонам создаются объекты. В языке Visual Basic основными являются классы объектов, реализующие графический интерфейс приложения. В языке VBA еще используются более ста различных классов объектов, которые существуют в среде Windows&Office. В обоих языках существуют возможности подключения дополнительных библиотек про-

граммных объектов, а также создания новых классов объектов самим программистом.

Каждый из классов обладает специфическим набором свойств, методов и событий. Например, в приложении Word существует класс объектов *«документ»* (Document), который обладает определенными наборами:

- свойств: *имя* (Name), *полное имя* (FullName) и так далее;
- методов: *открыть документ* (Open), *напечатать документ* (PrintOut), *сохранить документ* (Save) и так далее;
- событий: *открытие документа* (Document_New()), *закрытие документа* (Document_Close()) и так далее.

Экземпляры класса. Объект, созданный по «шаблону» класса объектов, является *экземпляром класса* и *наследует* весь набор свойств, методов и событий данного класса. Каждый экземпляр класса имеет уникальное для данного класса имя, которое указывается в скобках после названия класса, например:

```
Document("Проба.doc")
```

Различные экземпляры класса обладают одинаковым набором свойств, однако значения свойств у них могут отличаться. Так, в приложении Word могут быть открыты несколько документов, экземпляров класса Document, которые имеют различные имена, хранятся в различных каталогах и так далее. В табл. 4.1 приведены значения некоторых свойств двух экземпляров класса Document: Document("Проба.doc"), который хранится в каталоге Документы на диске C:, и Document("Проба.txt"), который хранится в корневом каталоге этого диска.

Таблица 4.1. Некоторые свойства экземпляров класса Document

Имя объекта	Свойства объекта и их значения	
	FullName (полное имя)	Path (путь)
proba.doc	C:\Документы\proba.doc	C:\Документы\
proba.txt	C:\proba.txt	C:\

Семейства объектов. *Семейство* объектов представляет собой объект, содержащий несколько объектов, экземпляров одного класса. Например, все открытые в текущий момент в приложении Word документы образуют семейство, которое обозначается следующим образом:

```
Documents()
```

Обращение к объекту, входящему в семейство, производится по его имени или индексу. Например, обращение к документу производится по его имени:

```
Documents("Проба.doc")
```

Все символы, входящие в выделенный фрагмент документа (объект `Selection`) входят в семейство `Characters()`. Обращение к символу производится по его индексу, например:

```
Characters(7)
```

Вопросы для размышления

1. Чем различаются понятия «класс объектов», «экземпляр класса» и «семейство объектов»?
2. Как вы думаете, какие классы объектов существуют в приложении Word? В приложении Excel?

4.3.2. Объекты: свойства, методы, события

Свойства объектов (`Properties`). Каждый объект обладает определенным набором свойств, первоначальные значения которых можно установить с использованием диалогового окна системы программирования.

Значения свойств объектов можно изменять в программном коде. Для присвоения свойству объекта нового значения в левой части строки программного кода необходимо указать имя объекта и затем название свойства, которые в соответствии с правилами точечной нотации разделяются между собой точкой. В правой части строки (после знака равенства) необходимо записать конкретное значение свойства:

```
Объект.Свойство = ЗначениеСвойства
```

Например, установим в выделенном фрагменте текста (объект `Selection`) для первого символа (объект `Characters(1)`) начертание *полужирный* (свойство `Bold`).

Свойство `Bold` может быть установлено (значение **`True`** свойства) или не установлено (значение **`False`** свойства). Значения **`True`** и **`False`** являются ключевыми словами языка и поэтому выделяются полужирным начертанием.

Присвоим свойству Bold значение **True**:

```
Selection.Characters(1).Bold = True
```

Объект обычно имеет несколько свойств. С помощью инструкции **With ... End With** можно задать значения сразу нескольких свойств объекта. Синтаксис установки значения нескольких свойств объекта:

```
With Объект
   .Свойство1 = ЗначениеСвойства1
   .Свойство2 = ЗначениеСвойства2
   ...
   .СвойствоN = ЗначениеСвойстваN
End With
```

Например, для придания выделенному фрагменту текста, состоящему из 10 символов, начертания *«полужирный курсив»* можно использовать следующий программный код:

```
For I = 1 To 10
With Selection.Characters(I)
   .Bold = True
   .Italic = True
End With
Next I
```

Методы объектов (Methods). Для того чтобы объект выполнил какую-либо операцию, необходимо применить метод, которым он обладает. Многие методы имеют аргументы, которые позволяют задать параметры выполняемых действий. Для присваивания аргументам конкретных значений используется двоеточие и знак равенства, а друг от друга аргументы отделяются запятой.

Обратиться к методу объекта можно так же, как и к свойству объекта, с использованием *точечной нотации*. Чтобы определить, для какого объекта вызывается метод, перед именем метода указывается имя объекта, отделенное точкой:

```
Объект.Метод арг1:=значение, арг2:=значение
```

Так, сохранение на диске открытого в приложении Word документа реализуется методом Save, без аргументов:

```
Documents("Проба.doc").Save
```

Операция открытия в приложении Word документа Проба.doc должна содержать не только название метода Open, но

и указание пути к открываемому файлу (аргументу FileName метода Open необходимо присвоить конкретное значение):

```
Documents().Open  FileName:="C:\Документы\Проба.doc"
```

Печать трех первых страниц документа Проба.doc реализуется с помощью метода PrintOut с несколькими аргументами. В этом случае необходимо задать значения аргументов Range (*задает формат диапазона печати*), From и To (*задают номера начальной и конечной страниц печати*):

```
Documents("Проба.doc").PrintOut _
Range:=wdPrintFromTo,  From:="1",To:="3"
```

События (Events). Событие представляет собой действие, распознаваемое объектом. Событие может создаваться пользователем (например, щелчок мышью или нажатие клавиши) или быть результатом воздействия других программных объектов. В качестве реакции на события вызывается определенная процедура, которая может изменять значения свойств объекта, вызывать его методы и так далее.

Например, объект Document (*Документ*) *реагирует на события* Open (*Открытие*), New (*Создание*) *и* Close (*Закрытие*), *а объект* Selection (*Выделенный фрагмент документа*) реагирует на события Cut (*Вырезка*), Copy (*Копирование*), Paste (*Вставка*), Delete (*Удаление*) и так далее.

Вопросы для размышления

1. Можно ли в заданном программном объекте изменить: набор свойств? Набор методов? Набор событий? Значения свойств?
2. Каким образом можно изменить значения свойств программного объекта?

Задания

4.4. Составьте фрагмент программы, которая открывает документ, печатает его и сохраняет на диске.

4.5. Составьте фрагмент программы, которая в выделенном фрагменте текста (например, включающем 10 символов), буквам «а» задает начертание **полужирный**, а всем остальным символам – начертание *курсив*.

4.3.3. Графический интерфейс и событийные процедуры

Графический интерфейс. Графический интерфейс необходим для реализации интерактивного диалога пользователя с работающим приложением. Основой для создания графического интерфейса разрабатываемого приложения являются *форма* (в Visual Basic — класс объектов `Form`, в VBA — класс объектов `UserForm`), представляющая собой окно, в котором размещаются управляющие элементы. Необходимо отметить, что графический интерфейс проекта может включать в себя несколько форм.

Форма – это объект, представляющий собой окно на экране, в котором размещаются управляющие элементы.

Визуальное конструирование графического интерфейса приложения состоит в том, что на форму с помощью мыши помещаются и «рисуются» те или иные *управляющие элементы*.

Классы *управляющих элементов* (`Controls`) имеют различное назначение в графическом интерфейсе приложения. *Текстовые поля* (`TextBox`), *метки* (`Label`) и *списки* (`ListBox`) обычно используются для ввода и вывода данных, *графические окна* (`PictureBox`) — для вывода графики, *командные кнопки* (`CommandButton`), *переключатели* (`CheckBox`) и *флажки* (`OptionsButton`) — для организации диалога и так далее.

На форму может быть помещено несколько экземпляров одного класса управляющих элементов, например, несколько кнопок `Command1`, `Command2`, `Command3` и так далее, каждая из которых обладает индивидуальными значениями свойств (надпись, размеры и др.).

Управляющие элементы – это объекты, являющиеся элементами графического интерфейса приложения и реагирующие на события, производимые пользователем или программными объектами.

Форма и *управляющие элементы* обладают определенными наборами свойств, методов и событий (табл. 4.2).

Таблица 4.2. Некоторые классы объектов, их свойства, методы и события

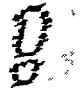
Класс объектов	Свойства	Методы	События
Form (*форма*) UserForm (*форма*)	Name (*Имя*) Caption (*Надпись*) Font (*Шрифт*) Height (*Высота*) Width (*Ширина*)	Show (*Показать*) Move (*Переместить*)	Load (*Загрузка*)
CommandButton (*командная кнопка*)	Name (*Имя*) Caption (*Надпись*) Font (*Шрифт*) Height (*Высота*) Width (*Ширина*)	Move (*Переместить*)	Click (*Щелчок*)
TextBox (*текстовое поле*)	Name (*Имя*) Text (*Текст*) Font (*Шрифт*) Height (*Высота*) Width (*Ширина*)	Move (*Переместить*)	DblClick (*Двойной щелчок*)

Соглашение об именах объектов. Целесообразно объектам проекта присваивать имена, которые дают возможность распознать их тип и назначение. Принято, что имя начинается с префикса, который определяет тип объекта. Для форм принят префикс `frm`, для командных кнопок — `cmd`, текстовых полей — `txt`, для надписей — `lbl` и так далее. После префикса идет информативная часть имени, которая пишется с прописной буквы (например, `frmFirst`, `lblText`, `cmdExit`) или содержит число (например, `txt1`, `txt2`, `txt3`).

Событийные процедуры. Для каждого события можно запрограммировать *отклик*, то есть реакцию объекта на произошедшее событие. Если пользователь производит какое-либо воздействие на элемент графического интерфейса (например, щелчок), в качестве отклика выполняется некоторая последовательность действий (событийная процедура).

Имя процедуры включает в себя имя объекта и имя события.

```
Объект_Событие()
```



Событийная процедура представляет собой подпрограмму, которая начинает выполняться после реализации определенного события.

В событийной процедуре может участвовать несколько объектов. Например, само событие происходит с первым объектом (Объект1), в результате второй (Объект2) изменяет значение своего свойства, а третий (Объект3) реализует какой-либо метод и так далее.

Каждая процедура представляет собой отдельный программный модуль, который реализует определенный алгоритм. В терминологии процедурного программирования такие процедуры соответствуют подпрограммам, поэтому каждая из событийных процедур начинается с ключевого слова **Sub** (subroutine — подпрограмма) и заканчивается ключевыми словами **End Sub**:

```
Sub Объект(1)_Событие()
Объект(2).Свойство = ЗначениеСвойства
Объект(3).Метод арг1:=знач, арг2:=знач
...
End Sub
```


Вопросы для размышления

1. Какие объекты целесообразно использовать при конструировании графического интерфейса проекта, если необходимо: вводить и выводить данные? Выводить надписи?
2. Могут ли в событийной процедуре объекта изменяться его свойства и применяться его методы?

4.4. Интегрированная среда разработки языка Visual Basic

Установить систему программирования Visual Basic 5.0 CCE CD-ROM 

Интегрированная среда разработки языка Visual Basic предоставляет пользователю удобный графический интерфейс в процессе разработки приложения. После запуска Visual Basic для начала работы над новым проектом необходимо ввести команду [File-New-Standart].

Появится окно интегрированной среды разработки языка Visual Basic (рис. 4.8). Интегрированная среда разработки включает в себя:

Строку заголовка, которая состоит из имени проекта Project1, после которого через тире указана программная среда Microsoft Visual Basic. Далее, слово [design] означает текущий режим работы — проектирование. В режиме выполнения проекта текст в квадратных скобках заменяется на [run]. Кнопки управления окном расположены в правом углу строки.

Под строкой заголовка расположена строка главного меню.

Под строкой главного меню находятся кнопки с пиктограммами наиболее часто используемых команд.

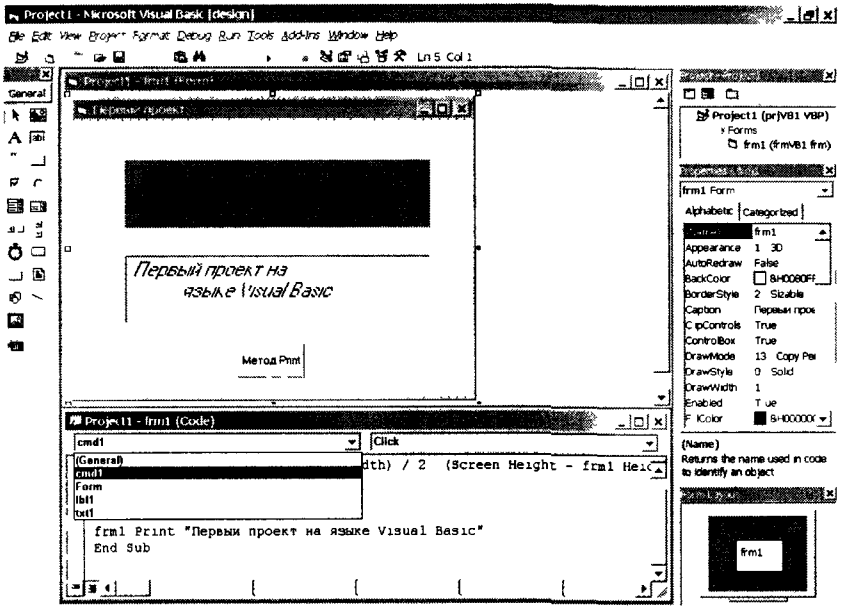


Рис. 4.8. Интегрированная среда разработки Visual Basic

Окно *Конструктор форм*. Окно *Конструктор форм* (рис. 4.9) является основным рабочим окном и расположено в центре окна интегрированной среды разработки языка Visual Basic. По умолчанию проекту присваивается имя *Project1*. Именно в этом окне происходит визуальное конструирование графического интерфейса разрабатываемого приложения.

В окне *Конструктор форм располагается сама форма* `frm1`, которая является также объектом и принадлежит классу объектов `Form`. Размеры формы можно менять, перетаскивая мышью правую или нижнюю границу формы.

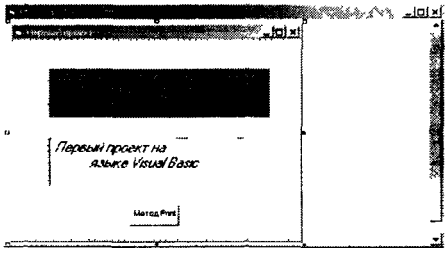


Рис. 4.9. Окно *Конструктор форм*

Первоначально форма пуста, в дальнейшем, в процессе создания графического интерфейса приложения, на ней размещаются элементы управления.

Окно *Программный код*. С формой связан программный модуль, содержащий программные коды процедур. Для ввода и редактирования текста программы служит окно *Программный код* (в данном случае *Project1-frm1(Code)* — рис. 4.10), которое вызывается командой [View-Code].

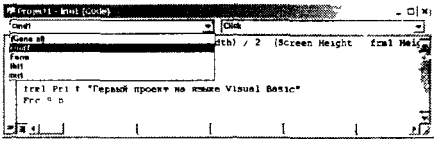


Рис. 4.10. Окно *Программный код*

Сразу под строкой заголовка окна *Программный код* размещаются два раскрывающихся списка. Левый список содержит перечень объектов проекта (объектов, размещенных на форме), а правый — перечень событий, доступных для выбранного объекта.

Панель инструментов. В левой части окна интегрированной среды разработки Visual Basic располагается *Панель инструментов* (*ToolBox*), содержащая пиктограммы *управляющих элементов* (рис. 4.11). Стандартный набор управляющих элементов включает в себя 21 класс объектов: `CommandButton` (*командная кнопка*), `TextBox` (*текстовое поле*), `Label` (*надпись*) и т. д. Существует возможность дополнить *панель инструментов* новыми классами управляющих элементов `RichTextBox` (*усовершенствованное текстовое поле*), `ImageList` (*список изображений*) и др.

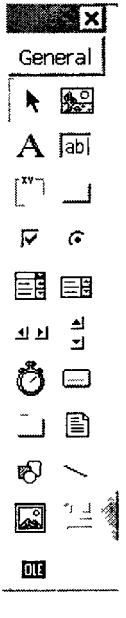


Рис. 4.11. Окно *Панель инструментов*

Выбрав щелчком мышью на *Панели инструментов* нужный элемент, мы можем поместить его на форму проектируемого приложения. Процесс размещения на форме управляющих элементов аналогичен рисованию графических примитивов с использованием графического редактора.

Фактически мы размещаем на форме экземпляры определенных классов объектов. Например, выбрав класс `CommandButton`, мы можем разместить на форме неограниченное количество экземпляров этого класса, то есть командных кнопок `Command1`, `Command2`, `Command3` и так далее.

Окно *Свойства объекта*. Справа располагается окно *Свойства объекта* (*Properties*) — рис. 4.12. Окно содержит список объектов и список свойств, относящихся к выбранному объекту (форме или управляющему элементу на форме). На рисунке выбран объект `frm1` класса `Form`.

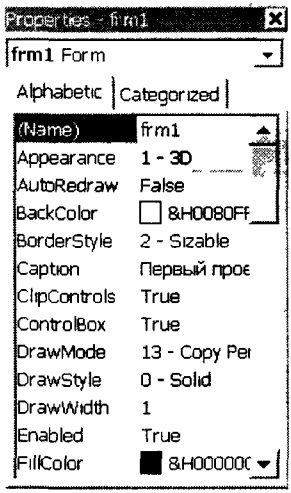


Рис. 4.12. Окно *Свойства объекта*

Список свойств разделен на две колонки. В левой находятся имена свойств, в правой — их значения. Установленные по умолчанию значения могут быть изменены. Свойством объекта является количественная или качественная характеристика этого объекта (размеры, цвет, шрифт и др.).

Для некоторых свойств предусмотрена возможность выбора значений из раскрывающегося списка, например, из списка можно выбрать значение цвета фона формы (свойства `BackColor`).

Окно *Просмотр объектов*. Еще одно важнейшее окно — окно *Просмотр объектов (Object Browser)* может быть вызвано командой [View-Object Browser] — рис. 4.13.

В левой колонке окна производится выбор объекта или класса объектов. В данном случае выбран класс объектов `Form`.

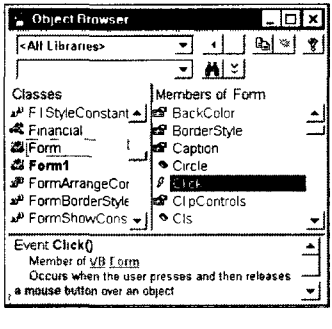


Рис. 4.13. Окно *Просмотр объектов*

В правой колонке появляется перечень свойств, методов и событий выбранного объекта или класса объектов. Во фрагмент списка, показанный в окне, входят, например, свойство `Caption`, метод `Circle` и событие `Click`.

Выбрав элемент списка (например, событие `Click`), можно получить о нем краткую информацию, которая появляется в нижней части окна.

Окно *Проводник проекта*. Окно *Проводник проекта (Project)* располагается в верхнем правом углу (рис. 4.14).

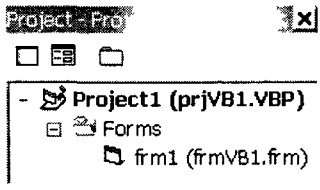


Рис. 4.14. Окно *Проводник проекта*

Оно отображает в виде иерархического каталога все составные части текущего проекта (в данном случае *Project1*) и позволяет переключаться между ними (по форме и по функциям оно аналогично окну *Проводник* Windows).

Проект хранится в файле с расширением vbp (в данном случае в файле prjVB1 vbp). Кроме того, входящие в проект формы хранятся в отдельных файлах с расширением frm (в данном случае форма, входящая в состав проекта, хранится в файле frmVB1.frm).

Окно *Расположение формы*. В нижнем правом углу находится окно *Расположение формы* (*Form Layout*) — рис. 4.15. Оно показывает, где будет располагаться окно формы на экране монитора в период выполнения программы. Положение формы можно изменять перетаскиванием мышью.

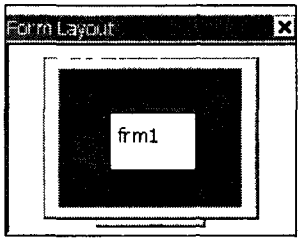


Рис. 4.15. Окно *Расположение формы*

Точное местоположение и размеры формы отображаются двумя парами чисел в правой части линейки инструментов окна приложения (рис. 4.16).

Рис. 4.16. Линейка инструментов

Первая пара чисел показывает расстояние от левого верхнего угла монитора до левого верхнего угла формы, а вторая пара — это размеры формы (ширина и высота). Размеры

отображаются в особых единицах — твипах (один твип равен примерно 0,018 мм).

Если необходимо задать точные значения местоположения и размеров формы, то это можно сделать, установив значения свойств формы `Left` (расстояние по горизонтали от левого верхнего угла монитора до верхнего левого угла формы), `Top` (расстояние по вертикали от левого верхнего угла монитора до верхнего левого угла формы), `Width` (ширина формы) и `Height` (высота формы).

Расположение вышеперечисленных окон на рабочем столе интегрированной среды разработки, а также их размеры можно изменять с помощью мыши или команд меню *View* (*Просмотр*).

Этапы разработки приложения. Создание приложения в среде Visual Basic можно условно разделить на несколько этапов:

1. Создание графического интерфейса будущего приложения. В окне *Конструктор форм* на форму помещаются управляющие элементы, которые должны обеспечить взаимодействие приложения с пользователем.
2. Задание значений свойств объектов графического интерфейса. С помощью окна *Свойства объекта* задаются значения свойств управляющих элементов, помещенных ранее на форму.
3. Создание программного кода. В окне *Редактор кода* производится ввод и редактирование программного кода процедур.
4. Сохранение проекта. Так как проекты включают в себя несколько файлов (в том числе несколько файлов форм), рекомендуется для каждого проекта создать отдельную папку на диске. Сохранение проекта производится с помощью меню *File*. Сначала необходимо сохранить форму и связанный с ней программный модуль с помощью пункта меню *Save FormVB1.frm As...* По умолчанию для файла формы предлагается имя, заданное в качестве значения свойства `Name` и расширение frm.

   Далее, необходимо сохранить файл проекта с помощью пункта меню *Save Project As...* В соответствие с соглашением об именах объектов целесообразно сохранить проект под именем с префиксом prj, например prjVB1.vbp.
5. Компиляция проекта в приложение. Сохраненный проект может выполняться только в самой системе програм-

мирования Visual Basic. Для того чтобы преобразовать проект в приложение, которое может выполняться непосредственно в среде операционной системы, необходимо сохранить проект в исполняемом файле (типа exe). Для компиляции проекта в исполняемый файл используется команда [File-Make ...] (в свободно распространяемой версии VB5.0 CCE такая возможность, к сожалению, отсутствует).

Вопросы для размышления

1. Перечислите основные окна интегрированной среды разработки Visual Basic и объясните их назначение.

Практические задания

4.6. С помощью окна *Просмотр объектов (Object Browser)* найти классы объектов, которые обладают свойствами `Name` (*Имя*), `Caption` (*Надпись*), `Font` (*Шрифт*), `Height` (*Высота*), `Width` (*Ширина*), могут использовать метод `Move` (*Перемещение*) и откликаются на событие `Click` (*Щелчок*).

4.5. Форма и размещение на ней управляющих элементов

Создадим приложение, в котором после запуска на форме печатается некоторый текст, например «Первый проект».

Вывод на форму текстовых сообщений можно производить различными способами:

- с помощью элемента управления `Label` *(Метка)*;
- с помощью элемента управления `TextBox` *(Текстовое поле)*;
- используя метод **`Print`**.

Проект «Создание графического интерфейса»

1. Запустить Visual Basic. Создать новый проект командой [File-New-Standart].

Работа над проектом начинается с создания графического интерфейса будущего приложения. В окне *Конструктор форм* на форму помещаются управляющие элементы, которые должны обеспечить взаимодействие приложения с пользователем.

2. С помощью *Панели инструментов* на форму (Form1) поместить метку (Label1), текстовое поле (Text1) и командную кнопку (Command1).

Далее необходимо задать новые значения свойств управляющих элементов.

3. С помощью окна *Свойства объекта* изменить значения свойств формы и управляющих элементов согласно таблице:

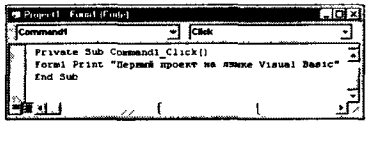
Класс объектов	Свойство	Значение по умолчанию	Новое значение
Form (*форма*)	Name	Form1	frm1
	Caption	Form1	Первый проект
Label (*надпись*)	Name	Label1	lbl1
	Caption	Label1	Первый проект
TextBox (*текстовое поле*)	Name	Text1	txt1
	Caption	Text1	Первый проект
CommandButton (*командная кнопка*)	Name	Command1	cmd1
	Caption	Command1	Метод Print

Следующим шагом является создание программного кода событийной процедуры.

Двойной щелчок мышью на объекте (форме или управляющем элементе) вызывает окно *Программный код* с пустой заготовкой событийной процедуры. Если осуществить двойной щелчок на кнопке cmd1, то появится заготовка событийной процедуры cmd1_Click:

```
Private Sub cmd1_Click()
End Sub
```

4. Двойным щелчком по кнопке cmd1 вызвать окно *Программный код* с пустой процедурой cmd1_Click(). Ввести в процедуру метод **Print**.

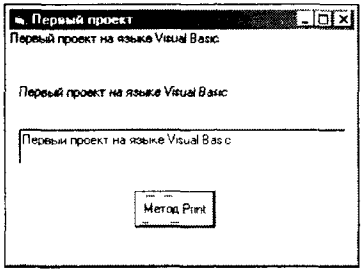


В теле процедуры должна быть записана последовательность инструкций (операторов), которые будут выполняться при наступлении события. Вывод текста на форму будет производиться с помощью метода **Print**:

```
frm1.Print "Первый проект на языке Visual Basic"
```

Первоначальный вариант проекта готов, и его можно запустить на выполнение.

5. Ввести команду [Run-Start]. Появится окно приложения *Первый проект*.
   Для выполнения событийной процедуры щелкнуть по кнопке *Метод Print*.



Установка цвета формы и параметров шрифта. Сделаем внешний вид проекта более привлекательным. Для этого изменим значения свойств объектов, определяющих внешний вид объектов (цвет фона формы, цвет, размер и способ выравнивания шрифта на метке, в текстовом поле и в методе **Print**).

6. Активизировать форму `frm1`. В окне *Свойства объекта* выбрать свойство `BackColor` (цвет фона) и двойным щелчком открыть диалоговое окно с цветовой палитрой. Выбрать цвет, например *желтый*.
7. Активизировать метку `lbl1`. В окне *Свойства объекта* установить значения свойств: `BackColor` — *зеленый*, `ForeColor` (цвет надписи) — *синий*, `Font` — *размер шрифта 18*, `Alignment` (выравнивание) — *center*.
8. Активизировать текстовое поле `txt1`. В окне *Свойства объекта* установить значения свойств: `ForeColor` (цвет надписи) — красный, `Font` — *размер шрифта 14 и начертание курсивное*, `Alignment` (выравнивание) — *center*.

Для установки параметров шрифта, которые используются в методе **Print**, необходимо для шрифта (объект `Font`) установить требуемые значения его свойств (размер шрифта, начертание и др.). Например, установим размер шрифта 12.

Для того чтобы новые установки свойств шрифта вступили в действие, они должны быть вставлены в событийную процедуру перед методом **Print**.

9. Двойным щелчком по кнопке `cmd1` вызвать окно *Программный код*. Ввести в процедуру `cmd1_Click()` строку `Font.Size = 12`.

10. После запуска проекта и щелчка по кнопке *Метод Print* получим новый вид графического интерфейса.

11. Сохранить файл формы командой [Save Form1.frm As ...] как frmVB1.frm и файл проекта командой [Save Project As ...] как prjVB1.vbp.

Проект хранится в каталоге \textbook\VB\prjVB1\ **CD-ROM**

Практические задания

4.7. Создать проект «Вывод сообщения», в котором на форму выводится текстовое сообщение «Первое задание выполнено!» с помощью метки и текстового поля, а выход из программы реализуется щелчком по кнопке *Exit*. Продумать графический интерфейс проекта.

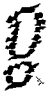
4.8. Создать проект «Вывод сообщений», в котором каждый из двух различных вариантов текста выводится в текстовое поле `TextBox` щелчком по одной из двух кнопок. Предусмотреть возможность выхода из программы щелчком по третьей кнопке.

4.9. Создать проект «Печать на форме», в котором с помощью метода **`Print`** на форме печатаются строки шрифтами со следующими параметрами:
Times New Roman, 18, курсив, красный;
Arial, 14, подчеркнутый, синий;
Courier New, 12, полужирный, зеленый.
Предусмотреть очистку формы от напечатанного текста с помощью метода `Cls`.

4.6. Тип, имя и значение переменной

В объектно-ориентированных языках программирования, и в частности в языке Visual Basic, *переменные* играют такую же важнейшую роль, как и в процедурных языках программирования. Переменные предназначены для хранения и обработки данных.

Переменные задаются *именами*, определяющими области памяти, в которых хранятся *значения* переменных. Значениями переменных могут быть *данные* различных типов (целые или вещественные числа, последовательности символов, логические значения и так далее).



Переменная в программе представлена *именем* и служит для обращения к данным определенного типа. Конкретное *значение* переменной хранится в ячейках оперативной памяти.

Тип переменной. Тип переменной определяется типом данных, которые могут быть значениями переменной. Значениями переменных числовых типов (**Byte**, **Integer**, **Long**, **Single**, **Double**) являются числа, логических (**Boolean**) — **True** или **False**, строковых (**String**) — последовательности символов и так далее. Обозначения типов переменных являются ключевыми словами языка и поэтому выделяются.

Над различными типами данных (различными типами переменных) допустимы различные операции. Над числовыми переменными возможны арифметические операции, над логическими переменными — логические операции, над строковыми — операции преобразования символьных строк и так далее.

Различные типы данных требуют для своего хранения в оперативной памяти компьютера различное количество ячеек (байтов). Для хранения целых чисел в интервале от 0 до 255 в переменных типа **Byte** достаточно одного байта; для хранения вещественного числа с двойной точностью в переменных типа **Double** требуется уже восемь байтов, а для хранения символьных строк в переменных типа **String** требуется один байт на каждый символ (табл. 4.3).

Таблица 4.3. Типы переменных

Тип переменной	Возможные значения	Объем занимаемой памяти	Приставка к имени
Byte	Целые числа от 0 до 255	1 байт	byt
Integer	Целые числа от –32768 до 32767	2 байта	int
Long	Целые числа двойной длины	4 байта	lng
Single	Десятичные числа одинарной точности от 1,401298E–35 до 3,4022823E38	4 байта	sng
Double	Десятичные числа двойной точности от 1,94065645841247E–324 до 1,79769313486232E+308	8 байтов	dbl
Boolean	Логическое значение **True** или **False**	2 байта	bln
String	Строка символов	1 байт на каждый символ	str
Currency	Число в денежном формате	8 байтов	cur
Date	Дата от 1 января 100 г. до 31 декабря 9999 г.	8 байтов	dtm
Object	Ссылки на любой объект	4 байта	obj
Variant	Любые значения	⩾ 16 байтов	vnt

Имя переменной. Имя каждой переменной (идентификатор) уникально и не может меняться в процессе выполнения программы. Имя переменной может состоять из различных символов (латинские и русские буквы, цифры и так далее), но должно обязательно начинаться с буквы и не должно включать знак «.» (точка). Количество символов в имени не может быть более 255.

Числовую переменную можно назвать, например, `A` или `Число`, а строковую — `A` или `Строка`. Однако разработчик языка Visual Basic — фирма Microsoft — рекомендует для большей понятности текстов программ для программиста в имена переменных включать особую *приставку*, которая обозначает тип переменных. Тогда имена целочисленных переменных целесообразно записывать как `intA` или `intЧисло`, а строковых — `strA` и `strСтрока`.

Объявление типа переменной. Важно, чтобы не только разработчик программы (программист) понимал, переменные какого типа используются в программе, но чтобы это мог учесть и исполнитель программы (компьютер). Второе даже еще более важно, так как, если компьютер не будет «знать», переменная какого типа используется в программе, он будет считать ее переменной универсального типа **Variant** и отведет для ее хранения в памяти 16 или более байтов. Это будет приводить к неэффективному использованию памяти и замедлению работы программы.

Для объявления типа переменной используется оператор определения переменной. Синтаксис этого оператора следующий:

```
Dim ИмяПеременной [As ТипПеременной]
```

С помощью одного оператора можно объявить сразу несколько переменных, например:

```
Dim intЧисло As Integer, strСтрока As String
```

Переменные, значения которых не меняются в процессе выполнения программы, называются *константами*. Синтаксис объявления констант следующий:

```
Const ИмяКонстанты [As Тип]= ЗначениеКонстанты
```

Вопросы для размышления

1. В чем разница между типом, именем и значением переменной?
2. Какие типы переменных используются в языке программирования Visual Basic и какую функцию выполняют приставки в именах переменных?
3. Почему рекомендуется объявлять переменные перед их использованием в программе?

Задания

4.10. Определить, какой диапазон чисел может храниться в переменной типа Long с учетом выделения одного бита для хранения знака числа.

4.11. Определить, какое количество ячеек памяти потребуется для хранения строк «ЭВМ» и «информатика».

4.7. Арифметические, строковые и логические выражения. Присваивание

Из переменных можно образовывать *арифметические, строковые* и *логические выражения.*

Арифметические выражения. В состав арифметических выражений могут входить кроме переменных числового типа также и числа, над переменными и числами могут производиться различные арифметические операции, а также математические операции, выраженные с помощью функций.

Порядок вычисления арифметических выражений соответствует общеизвестному порядку выполнения арифметических операций (возведение в степень, умножение или деление, сложение или вычитание), который может изменяться с помощью скобок.

Строковые выражения. В состав строковых выражений могут входить переменные строкового типа, *строки и строковые функции.*

Строками являются любые последовательности символов, заключенные в кавычки. Например:

```
"информатика", "2000", "2*2"
```

Над переменными и строками может производиться операция *конкатенации.* Операция конкатенации заключается в объединении строки или значения строковых переменных в единую строку. Операция конкатенации обозначается знаком «+», который не следует путать со знаком сложения чисел в арифметических выражениях.

Логические выражения. В состав логических выражений кроме логических переменных могут входить также числа, числовые или строковые переменные или выражения, которые сравниваются между собой с использованием операций сравнения (>, <, =, >=, <= и пр.).

Логическое выражение может принимать лишь два значения: «истина» или «ложь». Например:

5 > 3 — истинно;

2*2 = 5 — ложно.

Над элементами логических выражений могут производиться логические операции, которые на языке Visual Basic обозначаются следующим образом: логическое умножение — **And**, логическое сложение — **Or** и логическое отрицание **Not**. При записи сложных логических выражений часто используются скобки. Например:

(5 > 3) **And** (2*2 = 5) — ложно;

(5 > 3) **Or** (2*2 = 5) — истинно.

Присваивание переменным значений. Переменная может получить или изменить значение с помощью *оператора присваивания*. Синтаксис этого оператора следующий:

```
[Let] ИмяПеременной = Выражение
```

Ключевое слово **Let** в большинстве случаев не используется.

При выполнении оператора присваивания переменная, имя которой указано слева от знака равенства, получает значение, равное значению выражения (арифметического, строкового или логического), которое находится справа от знака равенства.

Создадим проект, который позволит продемонстрировать использование переменных различных типов, арифметических, строковых и логических выражений и операции присваивания.

Сначала произведем деление двух целых чисел, а для хранения результата будем использовать различные типы числовых переменных, то есть результаты будут вычисляться с различной точностью.

Проект «Переменные»

1. Создать новый проект. Для создания графического интерфейса разместить на *форме* (frm1) управляющий элемент *командная кнопка* (cmd1).

В качестве аргументов программы пусть выступают две целочисленные переменные intA и intB, а в качестве резуль-

татов неотрицательная целочисленная переменная bytC, вещественная переменная одинарной точности sngD и вещественная переменная двойной точности dblE.

2. Объявить переменные для их использования в программе:

```
Dim intA, intB As Integer, bytC As Byte,
sngD As Single, dblE As Double
```

Создадим заготовку событийной процедуры, в которой в качестве объекта будет использоваться *кнопка* cmd1, а в качестве события *щелчок* — Click().

3. Произвести двойной щелчок по *кнопке* cmd1, в окне *Программный код* появится заготовка событийной процедуры:

```
Sub cmd1_Click()

End Sub
```

Теперь необходимо разработать программный модуль, реализующий следующий алгоритм деления двух чисел:

1. Присвоить аргументам алгоритма, переменным intA и intB конкретные значения.
2. Присвоить каждой из переменных bytC, sngD и dblE результат деления аргументов.
3. Напечатать результаты (значения переменных) на форме.

Реализацию первых двух инструкций (команд) алгоритма произведем с помощью операторов присваивания. В качестве исходных чисел возьмем целые числа 2 и 3. Печать результатов алгоритма осуществим с помощью метода **Print**, которым обладает объект frm1.

Метод **Print** используется для печати на форме чисел и строк, а также значений числовых и строковых переменных или выражений, которые образуют *список печати*. В качестве разделителей списка печати используется либо запятая, либо точка с запятой. В первом случае элементы списка печатаются каждый в своей зоне (каждая зона имеет длину 14 символов), во втором случае элементы списка печатаются вплотную друг к другу. В случае отсутствия списка печати на форму выводится пустая строка.

Синтаксис метода **Print** следующий:

```
Объект.Print [СписокПечати]
```

4. В окне *Программный код* ввести первую событийную процедуру целиком:

```
Dim intA, intB As Integer, bytC As Byte, sngD
As Single, dblE As Double
Sub cmd1_Click()
intA = 2
intB = 3
bytC = intA / intB
sngD = intA / intB
dblE = intA / intB
frm1.Print bytC, sngD, dblE
End Sub
```

Теперь создадим событийную процедуру, реализующую операцию конкатенации строк и строковой переменной.

5. Разместить на *форме* (frm1) управляющий элемент *командная кнопка* (cmd2).

Создадим заготовку событийной процедуры, в которой в качестве объекта будет использоваться *кнопка* cmd2, а в качестве события — *щелчок* Click().

Объявим строковые переменные и произведем конкатенацию двух строковых выражений и строковой переменной.

6. В окне *Программный код* ввести вторую событийную процедуру целиком:

```
Dim strA, strB As String
Sub cmd2_Click()
strA = "форма"
strB = "ин" + strA + "тика"
frm1.Print strB
End Sub
```

Затем создадим событийную процедуру, реализующую логические операции с логическими переменными.

7. Разместить на *форме* (frm1) управляющий элемент *командная кнопка* (cmd3).

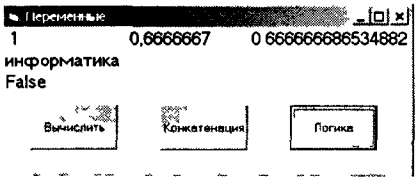
Создадим заготовку событийной процедуры, в которой в качестве объекта будет использоваться *кнопка* cmd3, а в качестве события *щелчок* Click().

Объявим логические переменные, присвоим им значения логических выражений, в которые входят операции сравнения, и произведем операцию логического умножения двух логических переменных.

8. В окне *Программный код* ввести третью событийную процедуру целиком:

```
Dim blnA, blnB, blnC As Boolean
Sub cmd3_Click()
blnA = 5 > 3
blnB = 2*2 = 5
blnC = blnA And blnB
frm1.Print blnC
End Sub
```

9. После запуска проекта на экране появится его графический интерфейс (форма с размещенными на ней командными кнопками). Последовательные щелчки по кнопкам вызовут выполнение событийных процедур и на форме будут напечатаны результаты выполнения проекта.



В первой событийной процедуре производится деление двух чисел с различной точностью, которая зависит от типа переменной, используемой для хранения результатов вычислений.

После выполнения второй событийной процедуры строковая переменная `strC` примет значение `"информатика"`.

После выполнения третьей событийной процедуры логическая переменная `blnC` примет значение **`False`** (*Ложь*).

Проект хранится в каталоге \textbook\VB\prjVB2\ **CD-ROM**

Вопросы для размышления

1. Могут ли в состав одного выражения входить переменные различных типов?
2. В чем разница между операцией арифметического сложения и операцией конкатенации?

Практические задания

4.12. Создать проект вычисления факториала числа. Предусмотреть визуализацию процесса вычисления факториала.

4.13. Создать проект, в котором сравниваются результаты сложения чисел и конкатенации строк, например 5 + 5 и "5" + "5".

4.14. Создать проект, в котором определяется истинность высказывания. Определить истинность составного высказывания «2 × 2 = 4 и 3 × 3 = 10 или 2 × 2 = 5 и 3 × 3 = 9».

4.8. Выполнение программ компьютером

Интерпретаторы и компиляторы. Для того чтобы процессор мог выполнить программу, эта программа и данные, с которыми она работает, должны быть загружены в оперативную память.

Итак, мы создали программу на языке программирования (некоторый текст) и загрузили ее в оперативную память. Теперь мы хотим, чтобы процессор ее выполнил, однако процессор «понимает» команды только на машинном языке, а наша программа написана на языке программирования. Как быть?

Необходимо, чтобы в оперативной памяти находилась программа-переводчик (*транслятор*), автоматически переводящая программу с языка программирования на машинный язык. Компьютер может выполнять программы, написанные только на том языке программирования, транслятор которого размещен в оперативной памяти компьютера.

Трансляторы языков программирования бывают двух типов: *интерпретаторы* и *компиляторы*. Интерпретатор — это программа, которая обеспечивает последовательный перевод инструкций программы на машинный язык и их выполнение. Поэтому при каждом запуске программы на выполнение эта процедура повторяется. Достоинством интерпретаторов является удобство отладки программы (поиска в ней ошибок), так как возможно пошаговое ее выполнение, а недостатком — сравнительно малая скорость выполнения.

Компилятор действует иначе, он переводит весь текст программы на машинный язык и сохраняет его в исполнимом файле (обычно с расширением exe). Затем этот уже го-

товый к выполнению файл, записанный на машинном языке, можно запускать на исполнение многократно. Достоинством компиляторов является большая скорость выполнения программы, а недостатком — трудоемкость отладки, так как невозможно пошаговое выполнение программы.

Современные системы программирования, и в том числе Visual Basic, позволяют работать в режиме как интерпретатора, так и компилятора. На этапе разработки и отладки программы используется режим интерпретатора, а для получения готовой исполняемой программы — режим компилятора.

Процесс выполнения программы. Рассмотрим процесс выполнения программы на примере рассмотренной выше программы (проект «Переменные»), написанной на языке программирования Visual Basic.

Ввод текста программы в оперативную память. Текст программы вводится в оперативную память с помощью клавиатуры или считывается из внешней памяти. Текст программы займет в памяти определенное количество ячеек (например, с ячейки номер I по ячейку $I+K$).

Перевод программы на машинный язык. Наша программа будет записана в памяти во внутреннем представлении языка программирования Visual Basic, который процессор «не понимает». Для перевода программы на машинный язык, понятный процессору, в памяти должна находиться программа-транслятор языка Visual Basic. Программа-транслятор после считывания в оперативную память из внешней памяти будет занимать в памяти определенное количество ячеек (например, с ячейки номер N по ячейку $N+M$).

Выполнение программы. После запуска программы на выполнение процессор последовательно будет считывать из памяти операторы и их выполнять.

В процессе выполнения оператора объявления переменных `Dim` в оперативной памяти для их хранения отводится необходимое количество ячееек: для целочисленных переменных `intA`, `intB` — по две ячейки, для неотрицательной целочисленной переменной `bytC` — одна ячейка, для переменной одинарной точности `sngD` — четыре ячейки, для переменной двойной точности `dblE` — восемь ячеек, для строковых переменных `strA` и `strB` количество ячеек, равное количеству символов, составляющих их значения, для логических переменных `blnA`, `blnB`, `blnC` — по две ячейки. Таким образом, в памяти для хранения данных (значений переменных) будет отведено определенное количество ячеек, например ячейки с 1-й по 39-ю (рис. 4.17).

Далее, в процессе выполнения операторов присваивания в отведенные переменным области оперативной памяти записываются их значения. Если в правой части оператора присваивания находятся арифметические выражения, то предварительно вычисляются их значения.

Затем с помощью метода **Print** производится вывод значений переменных на форму, реализующую графический интерфейс программы. В этом процессе значения переменных считываются из памяти и высвечиваются на экране монитора.

Имена переменных	Оперативная память	
	ячейки	значение
intA	1–2	2
intB	3–4	3
bytC	5	1
sngD	6–9	0,6666667
dblE	10–17	0,666666686534882
strA	18–22	форма
strB	23–33	информатика
blnA	34–35	True
blnB	36–37	False
blnC	38–39	False
	...	
	I	программный код
	I + *K*	
	...	
	N	транслятор языка программирования
	N + *M*	
	...	

Рис. 4.17. Программа и данные в оперативной памяти

Вопросы для размышления

1. Какую функцию выполняют трансляторы языков программирования?
2. В чем состоит различие между интерпретаторами и компиляторами?

4.15. Какое количество ячеек памяти было бы занято переменными в проекте «Переменные», если бы переменные не были объявлены?

4.9. Функции в языке Visual Basic

Понятие функции в языке программирования близко к понятию функции в математике. Функция может иметь один или более аргументов. При записи функции нескольких аргументов аргументы в списке отделяются друг от друга запятыми:

```
ИмяФункции(СписокАргументов)
```

Для каждого набора аргументов можно определить значение функции. В программировании говорят, что функция *возвращает* свое значение, если заданы значения ее аргументов. Функции обычно входят в состав выражений, значения которых присваиваются переменным.

Функции могут быть различных типов: *преобразования типов данных*, *математические*, *строковые*, *финансовые*, *даты* и др. Тип функции определяется возможными значениями аргументов и функции.

4.9.1. Функции преобразования типов данных

Функции преобразования реализуют преобразование данных из одного типа в другой.

Функция Val. Часто необходимо преобразовать строковое значение в числовое. Это можно сделать с помощью функции `Val`, аргументом которой является строка, а значением — число:

```
Val(Строка$)
```

Например, значением функции Val("2000") является число 2000. Эта функция часто применяется для преобразования строкового значения свойства Text текстовых полей в число, которое затем используется в арифметических выражениях.

Воспользуемся функцией Val для создания приложения «Обычный калькулятор», которое будет производить арифметические действия над целыми десятичными числами, которые будут вводиться и выводиться в текстовые поля на форме. Для создания графического интерфейса приложения разместим на форме три текстовых поля (два поля для ввода числовых данных и одно для вывода результата) и пять кнопок для реализации событийных процедур: сложения, вычитания, умножения, деления и завершения работы.

Проект «Обычный калькулятор»

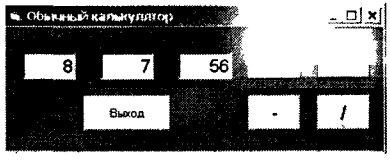
1. Создать новый проект. Разместить на форме три текстовых поля и пять кнопок. Присвоить им имена: txt1, txt2, txt3, cmdPlus, cmdMinus, cmdUmn, cmdDelen, cmdExit.

Событийная процедура сложения CmdPlus_Click должна изменять значения свойства Text текстового поля txt3 так, чтобы оно являлось суммой числовых значений свойства Text текстовых полей txt1 и txt2. Для преобразования строковых значений, вводимых в текстовые поля, в десятичные числа, воспользуемся функцией Val. Код событийной процедуры будет следующим:

```
Sub cmdPlus_Click()
txt3.Text = Val(txt1.Text) + Val(txt2.Text)
End Sub
```

Событийные процедуры вычитания, умножения и деления создаются аналогично.

2. Для каждой из кнопок ввести программные коды событийных процедур.
3. Установить для свойства Alignment текстовых полей значение Right Justufy.
4. Запустить проект на выполнение. Ввести числа в два левых текстовых поля и щелкнуть по кнопке арифметической операции. В правом поле будет выведен результат.

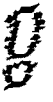


Проект хранится в каталоге \textbook\VB\prjVB3\ CD-ROM 

Строковое выражение, являющееся аргументом функции Val, может быть задано не только в десятичной, но также в восьмеричной (приставка "&O") и шестнадцатеричной (приставка "&H") системах счисления. Например, значением функций Val("&O3720") и Val("&H7D0") является десятичное число 2000.

Таким образом, появляется возможность перевода чисел, выраженных в строковой форме, из восьмеричной и шестнадцатеричной систем счисления в число десятичной системы счисления.

Функции Str, Hex, Oct. Функции Str, Oct и Hex позволяют производить преобразование десятичных чисел в десятичные, восьмеричные и шестнадцатеричные числа в строковой форме. Аргументом функции является десятичное число, а значением — строка:



```
Str(Число)
Oct(Число)
Hex(Число)
```

Например, значением функций Str(2000), Oct(2000), Hex(2000) является десятичное число 2000, восьмеричное число 3720 и шестнадцатеричное число H7D0 в строковой форме

Создадим проект, который позволит переводить целые числа из десятичной системы счисления в восьмеричную и шестнадцатеричную и обратно — из восьмеричной и шестнадцатеричной в десятичную.

Проект «Перевод чисел»

1. Создать новый проект. Разместить на форме три текстовых поля (txtDec, txtOct, txtHex) для ввода и вывода чисел, четыре кнопки (cmdDecOct, cmdDecHex, cmdOctDec, cmdHexDec) для создания событийных процедур, реализующих перевод чисел, и три метки (lblDec, lblOct, lblHex) для вывода поясняющих надписей над текстовыми полями.

К вводимым в текстовые поля txtOct и txtHex в строковой форме числам добавим восьмеричную и шестнадцате-

ричную приставки "&O" или "&H" с помощью операции конкатенации строк. Полученное восьмеричное или шестнадцатеричное число в строковой форме переведем в десятичную числовую форму с помощью функции Val.

2. Ввести событийную процедуру cmdOctDec_Click(), реализующую перевод чисел из восьмеричной системы в десятичную:

```
Sub cmdOctDec_Click()
txtDec.Text = Val("&O" + txtOct.Text)
End Sub
```

3. Создать событийную процедуру cmdHexDec_Click(), реализующую перевод чисел из шестнадцатеричной системы в десятичную.

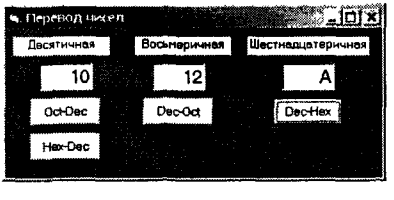
Введенные в текстовое поле txtDec числа будем переводить сначала из строковой формы в числовую с помощью функции Val, а затем из десятичной числовой в строковую восьмеричную или шестнадцатеричную с помощью функций Oct или Hex.

4. Создать событийную процедуру cmdDecHex_Click(), реализующую перевод чисел из десятичной системы в шестнадцатеричную:

```
Sub cmdDecHex_Click()
txtHex.Text = Hex(Val(txtDec.Text))
End Sub
```

5. Создать событийную процедуру cmdDecOct_Click(), реализующую перевод чисел из десятичной системы в восьмеричную.

6. Запустить проект. Для перевода десятичного числа в восьмеричную и шестнадцатеричную системы счисления ввести в левое текстовое поле десятичное число и последовательно щелкнуть по кнопкам *Dec-Oct* и *Dec-Hex*.



Проект хранится в каталоге \textbook\VB\prjVB4\ **CD-ROM**

Функция Asc. Функция Asc осуществляет преобразование строки в числовой код (в таблице кодировки) первого символа. Аргументом функции является строка, а значением — число:

```
Asc(Строка$)
```

Функция Chr. Функция `Chr` осуществляет преобразование числового кода в соответствующий ему символ. Аргументом функции является число, а значением — символ:

```
Chr(Число)
```

Создадим проект «Коды символов», который выводит числовой код введенного символа, а также распечатывает символы по числовым кодам (выводит кодировочную таблицу символов).

Проект «Коды символов»

1. Создать новый проект. Разместить на форме два текстовых поля `txtS`, `txtN` и две кнопки `cmdS` и `cmdT`.

Воспользуемся функцией `Asc()` для преобразования символа, вводимого в текстовое поле `txtS`, в его числовой код, выводимый в поле `txtN`.

2. Ввести событийную процедуру `cmdS_Click()`, реализующую преобразование символа в его числовой код:

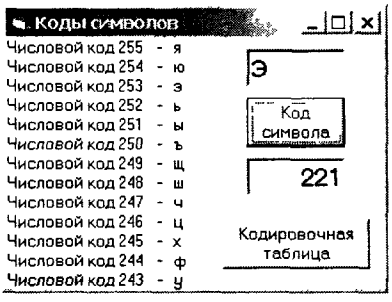
```
Sub cmdS_Click()
txtN.Text = Asc(txtS.Text)
End Sub
```

Воспользуемся циклом со счетчиком с шагом −1, для того чтобы печатать на форме символы, начиная с символа, имеющего наибольший числовой код (255). Такая последовательность вывода символов связана с тем, что первые 33 числовых кода (от 0 до 32) соответствуют не символам, а управляющим клавишам клавиатуры (*клавиши управления курсором, Пробел, Ввод и др.*).

3. Ввести код событийной процедуры `cmdT_Click()`, которая осуществляет вывод кодировочной таблицы символов на форму:

```
Dim strA As String, N As Integer
Sub cmdT_Click()
For N = 255 To 33 Step -1
strA = Chr(N)
Print "Числовой код"; N; "   -   "; strA
Next N
End Sub
```

4. Запустить проект. Для получения числового кода в нижнем текстовом поле ввести символ в верхнее поле и щелкнуть по кнопке *Код символа*.
Для получения таблицы кодировки символов щелкнуть по кнопке *Кодировочная таблица*.



Проект хранится в каталоге \textbook\VB\prjVB5\ **CD-ROM**

Практические задания

4.16. Разработать проект «Мультисистемный калькулятор», который позволяет производить арифметические операции над целыми числами в десятичной, восьмеричной и шестнадцатеричной системах счисления.

4.9.2. Математические функции

В математических функциях значениями как аргументов, так и функций являются числа. В языке Visual Basic имеется 12 математических функций: тригонометрические (`Sin()`, `Cos()`, `Tan()`, `Atn()`), квадратный корень `Sqr()`, логарифм `Log()`, показательная функция `Exp()`, получение случайного числа `Rnd()` и др.

Воспользуемся математическими функциями для расширения возможностей проекта «Обычный калькулятор» и превращения его в проект «Инженерный калькулятор».

Проект «Инженерный калькулятор»

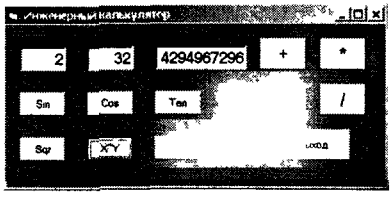
1. Открыть проект «Обычный калькулятор». Добавить на форму шесть кнопок `cmdSin`, `cmdCos`, `cmdTan`, `cmdSqr`, `cmdSt` и `cmdLog`.

Для каждой из этих кнопок создать событийные процедуры, реализующие вычисление соответствующих функций: синуса, косинуса, тангенса, квадратного корня, возведения в степень и натурального логарифма.

2. Например, для возведения в степень событийная процедура `cmdSt_Click()` примет вид:

```
Sub cmdSt_Click()
txt3Dec.Text=Val(txt1Dec.Text)^Val(txt2Dec.Text)
End Sub
```

3. Ввести самостоятельно программный код других событийных процедур с использованием встроенных функций языка Visual Basic: `Sin()`, `Cos()`, `Tan()`, `Sqr()` и `Log()`.
4. Запустить проект на выполнение.
   Произвести вычисление, например, 2^{32}: ввести числа 2 и 32 и щелкнуть по кнопке X^Y.



Проект хранится в каталоге \textbook\VB\prjVB6\ **CD-ROM**

Практические задания

4.17. Разработать проект, позволяющий вычислить гипотенузу и площадь прямоугольного треугольника, если известны его катеты.

4.9.3. Строковые функции

В строковых функциях строками являются либо аргументы, либо возвращаемые функциями значения.

Функция определения длины строки. В функции определения длины строки `Len(Строка$)` аргументом является строка `Строка$`, а возвращает функция числовое значение длины строки (количество символов в строке). Синтаксис функции:

```
Len(Строка$)
```

Пусть аргументом функции `Len` будет строка `"информатика"`, тогда значением целочисленной переменной `intДлинаСтроки = Len("информатика")` будет число 11.

Функции вырезания подстроки. В функциях вырезания подстроки (части строки) `Left(Строка$, Длина%)`, `Right(Строка$, Длина%)` и `Mid(Строка$, Позиция%, Длина%)` аргументами являются строка `Строка$` и числа или целочисленные переменные `Длина%` и `Позиция%`. Функции возвращают строковое значение, длина которого равна `Длина%`. Синтаксис функций:

```
Left(Строка$, Длина%)
Right(Строка$, Длина%)
Mid(Строка$, Позиция%, Длина%)
```

Значением функции `Left` является левая подстрока, которая начинается от крайнего левого символа строки и имеет количество символов, равное значению числового аргумента `Длина%`.

Пусть аргументом функции `Left` будет строка `"информатика"`, тогда значением строковой переменной `strЛевая Подстрока = Left("информатика",2)` будет строка `"ин"`.

Значением функции `Right` является правая подстрока, которая начинается от крайнего правого символа строки и имеет количество символов, равное значению числового аргумента `Длина%`.

Пусть аргументом функции `Right` будет строка `"информатика"`, тогда значением строковой переменной `strПравая Подстрока = Right("информатика",4)` будет строка `"тика"`.

Значением функции `Mid` является подстрока, которая начинается с символа, находящегося в позиции, заданной числовым аргументом `Позиция%`, и имеет количество символов, равное значению числового аргумента `Длина%`.

Пусть аргументом функции `Mid` будет строка `"информатика"`, тогда значением строковой переменной `strПодстрока = Mid("информатика",3,5)` будет строка `"форма"`.

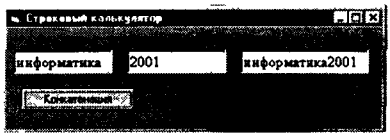
Создадим строковый калькулятор, который позволит производить различные преобразования строк.

Проект «Строковый калькулятор»

1. Разместить на форме три текстовых поля (`txt1`, `txt2` — для ввода строк, `txt3` — для вывода результата) и кнопку `cmdCon`.
2. Для кнопки ввести программный код событийной процедуры `cmdCon_Click()`, реализующий операцию конкатенации:

```
Sub cmdCon_Click()
txt3.Text = txt1.Text + txt2.Text
End Sub
```

3. Запустить проект, в два первых поля ввести строки и щелкнуть по кнопке *Конкатенация*. В третьем поле появится результат сложения двух строк.

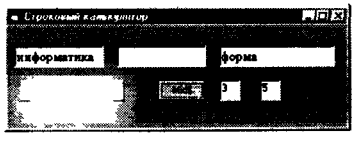


Воспользуемся теперь для преобразования строк строковой функцией Mid$(строка$,bytM,bytN). Функция вырезает из строки$ подстроку, начинающуюся с символа, позиция которого в строке задается целочисленной переменной bytM, и имеющую длину, заданную целочисленной переменной bytN.

4. Разместить на форме два текстовых поля txt1Mid и txt2Mid для ввода значений переменных bytM и bytN и кнопку cmdMid.
5. Для кнопки ввести программный код событийной процедуры cmdMid_Click(), реализующий операцию вырезания подстроки. Для преобразования строковых значений свойства Text текстовых полей в числа использовать функцию Val:

```
Sub cmdMid_Click()
txt3.Text = Mid$(txt1.Text, Val(txt1Mid.Text),
Val(txt2Mid.Text))
End Sub
```

6. Запустить проект, в первое поле ввести строку, в поля ввода аргументов функции вырезанной подстроки ввести числа и щелкнуть по кнопке *Mid$*. В третьем поле появится вырезанная подстрока.

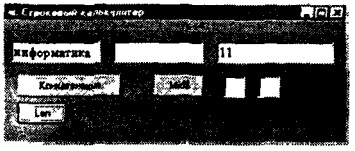


Для определения количества символов в строке используется функция определения длины строки Len(строка$), аргументом которой является строка, а возвращает функция число, равное количеству символов в строке.

7. Разместить на форме кнопку cmdLen и ввести программный код событийной процедуры cmdLen_Click(), реализующий операцию определения количества символов в строке:

```
Sub cmdLen_Click()
txt3.Text = Len(txt1.Text)
End Sub
```

8. Запустить проект, в первое поле ввести строку и щелкнуть по кнопке *Len*. В третьем поле появится число символов в строке.

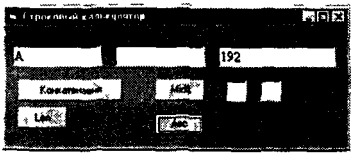


Для преобразования символов в соответствующий им числовой код используется функция `Asc(строка$)`, при этом необходимо иметь в виду, что функция возвращает числовой код первого символа строки.

9. Разместить на форме кнопку `cmdAsc` и ввести программный код событийной процедуры `cmdAsc_Click()`, реализующий операцию определения числового кода символа:

```
Sub cmdAsc_Click()
txt3.Text = Asc(txt1.Text)
End Sub
```

10. Запустить проект, в первое поле ввести символ и щелкнуть по кнопке *Asc*. В третьем поле появится числовой код символа.



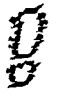
Проект хранится в каталоге \textbook\VB\prjVB7\ **CD-ROM**

Практические задания

4.18. Модернизировать проект «Строковый калькулятор». Добавить возможности вырезания из строки левой и правой подстрок, определение позиции начала подстроки в строке и преобразования числового кода символа в символ.

4.9.4. Функции ввода и вывода

Функция InputBox (*Окно Ввода*). Функция `InputBox` позволяет вводить данные с помощью диалоговой панели ввода. В качестве аргументов этой функции выступают три строки, значением функции является также строка. Синтаксис функции следующий:



```
InputBox(Приглашение$, Заголовок$,
[ПоУмолчанию$])
```

В процессе выполнения этой функции появляется диалоговая панель с текстовым полем. В строке заголовка панели будет печататься значение второго аргумента (Заголовок$), на самой панели печатается значение аргумента Приглашение$, в текстовом поле печатается значение аргумента ПоУмолчанию$ (если это значение отсутствует, содержимое текстового окна также отсутствует). Введенная пользователем в текстовом поле строка становится значением функции.

Функция MsgBox (*Панель Сообщений*). Функция MsgBox позволяет выводить сообщения не на форме, а на специальной панели сообщений. Кроме того, функция MsgBox возвращает определенное значение, которое может быть присвоено какой-либо переменной. Синтаксис функции следующий:

```
MsgBox(Сообщение$[, ЧисКод1+ЧисКод2][, Заголовок$])
```

Строка Сообщение$ выводится на панель сообщений, аргумент ЧисКод1+ЧисКод2 определяет внешний вид панели, а строка Заголовок$ печатается в строке заголовка панели. Последние два аргумента не являются обязательными.

Внешний вид панели сообщений можно менять, используя различные значения ЧисКод1 и ЧисКод2. Значение ЧисКод1 определяет вид пиктограммы, которая помещается на панель сообщений, а значение ЧисКод2 определяет набор кнопок, размещаемых на панели (табл. 4.4).

Таблица 4.4. Значения ЧисКод1 и ЧисКод2, определяющие вид панели сообщений

ЧисКод1	Пиктограмма
16	
32	
48	
64	

ЧисКод2	Набор кнопок
0	*ОК*
1	*ОК, Отмена*
2	*Стоп, Повтор, Пропустить*
3	*Да, Нет, Отмена*
4	*Да, Нет*
5	*Повтор, Отмена*

С помощью одного числа, являющегося суммой чисел ЧисКод1 и ЧисКод2, можно одновременно установить определенную пиктограмму и определенную комбинацию кнопок, размещенных на панели сообщений. Например, число 36 можно рассматривать как сумму чисел 32 (код пиктограммы «Вопрос») и 4 (код комбинации кнопок *Да, Нет*). В этом случае функция MsgBox выводит панель сообщений с текстом, пиктограммой, содержащей знак вопроса, и кнопками *Да, Нет*. Нажатие на кнопку приводит к вычислению значения функции, которое зависит от нажатой кнопки (табл. 4.5).

Таблица 4.5. Значения функции MsgBox

Нажатая кнопка	Значение функции
ОК	1
Отмена	2
Стоп	3
Повтор	4
Пропустить	5
Да	6
Нет	7

Разработаем проект, который позволит контролировать знания. Алгоритм контроля должен последовательно реализовывать следующие операции:

- задать (напечатать) вопрос;
- запросить ответ и запомнить введенное с клавиатуры значение;
- полученный ответ сравнить с правильным и, в зависимости от выполнения или невыполнения этого условия, реализовать различные действия.

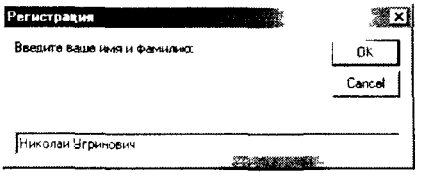
Сначала реализуем регистрацию проверяемого с использованием функций InputBox и MsgBox.

Проект «Проверка знаний»

1. Разместить на форме кнопку cmd1, задать значение *Начать проверку* свойства Caption. Создать событийную процедуру cmd1_Click().
2. С помощью функции InputBox запросить имя и фамилию и присвоить это значение строковой переменной strA, а с помощью функции MsgBox вывести результаты регистрации:

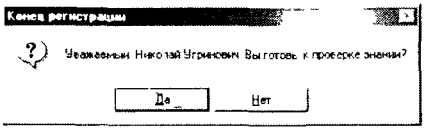
```
Dim strA As String, bytB As Byte
Sub cmd1_Click()
strA = InputBox("Введите ваше имя и фамилию:",
"Регистрация")
bytB = MsgBox("Уважаемый " + strA + ",
Вы готовы к проверке знаний?", 36,
"Конец регистрации")
End Sub
```

3. Запустить проект и щелкнуть по кнопке *Начать проверку*.
На появившейся диалоговой панели *Регистрация* ввести в текстовое поле имя и фамилию.



Вторым аргументом функции MsgBox является числовое значение, которое одновременно задает тип выводимого информационного окна и набор размещенных на нем кнопок.

4. Число 36 обеспечивает вывод информационного окна типа «Вопрос», которое имеет две кнопки *Да* и *Нет*.



Щелчок по одной из кнопок приводит к возвращению функцией определенного числового значения (*Да* — 6, *Нет* — 7), которое присваивается числовой переменной bytB.

5. С помощью условного оператора можно реализовать либо выход из программы (щелчок по кнопке *Нет*), либо продолжение работы и переход к проверке знаний (щелчок по кнопке *Да*):

```
If bytB = 7 Then End
```

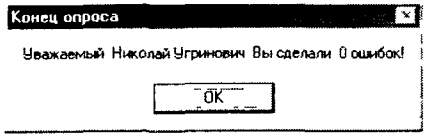
Вопрос задается с помощью функции InputBox, проверка правильности ответа производится с помощью оператора условного перехода **If...Then...Else**, а вывод информации о правильности или неправильности ответа — с помощью функции MsgBox в форме оператора (бесскобочная запись) и с числовым значением второго аргумента 0, что обеспечивает вывод информационного окна с одной кнопкой *OK*.

6. Ввести в событийную процедуру программный код, реализующий проверку знаний с помощью последовательно-

сти вопросов. В переменной bytN накапливать количество неправильных ответов:

```
strC = InputBox("Чему равен 1 байт?", "Первый
вопрос")
If strC = "8 битов" Then MsgBox "Правильно!",
0, "Первый вопрос" _
Else MsgBox "Неправильно!", 0, "Первый
вопрос": bytN = bytN + 1
strC = InputBox("Переведите десятичное число
5 в двоичную систему счисления:", "Второй
вопрос")
If strC = "101" Then MsgBox "Правильно!", 0,
"Второй вопрос" _
Else MsgBox "Неправильно!", 0, "Второй
вопрос": bytN = bytN + 1
MsgBox "Уважаемый " + strA + ", Вы сделали
" + Str(bytN) + " ошибок!", 0, "Конец опроса"
```

7. Запустить проект, пройти регистрацию и ответить на вопросы. Результат будет выведен с помощью панели сообщений функции MsgBox.



Проект хранится в каталоге \textbook\VB\prjVB8\ CD-ROM

Практические задания

4.19. Разработать проект, в котором проводится регистрация (запрашивается имя, отчество и фамилия) и существует возможность изменить регистрационные данные.

4.9.5. Функции даты и времени

Функция Date. Функция **Date** возвращает значение текущей даты, которое можно присвоить переменным типа **Date**. Значение даты представляется в виде тройки чисел #Месяц/Число/Год#, разделенных знаком «/». Разностью значений переменных типа **Date** является число дней между датами.

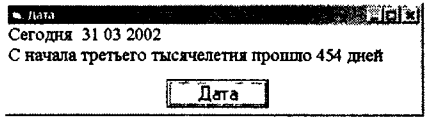
Создадим проект, позволяющий печатать текущую дату и количество дней, прошедших с начала третьего тысячелетия.

Проект «Дата»

1. Разместить на форме кнопку cmd1 и ввести событийную процедуру cmd1_Click():

```
Dim dtmA, dtmB As Date
Sub cmd1_Click()
dtmA = Date
dtmB = #1/1/2001#
Print "Сегодня   "; dtmA
Print " С начала третьего тысячелетия прошло
"; dtmA - dtmB; " дней"
End Sub
```

2. Запустить проект и щелкнуть по кнопке *Дата*. На форме будет напечатано текущее число и количество дней, прошедших с начала третьего тысячелетия.



Проект хранится в каталоге \textbook\VB\prjVB9\ CD-ROM

Функция Time$. Функция Time$ возвращает значение текущего времени, имеющее тип **String**, которое можно вывести в текстовое поле. Значение времени представляется в виде тройки чисел #Часы:Минуты:Секунды#, разделенных знаком «:».

Для периодического обновления значения времени используем объект Timer. Объект Timer не отображается на форме в процессе выполнения программы и реализует всего одну функцию — проверяет показания системных часов по событию Timer.

Периодичность события Timer может быть задана с помощью значения свойства Interval, задаваемого в миллисекундах (может изменяться от 0 до 65535). Для того чтобы событие Timer происходило каждую секунду, необходимо свойству Interval присвоить значение 1000.

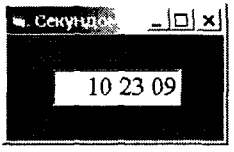
Создадим проект, выводящий текущее время в текстовое поле каждую секунду.

Проект «Секундомер»

1. Разместить на форме объект `Timer` и присвоить ему имя `tmr1`, а свойству `Interval` — значение 1000. Поместить на форму текстовое поле `txtTime`. Ввести событийную процедуру `tmr1_Timer()`:

```
Sub tmr1_Timer()
txtTime.Text = Time$
End Sub
```

2. Запустить проект. В текстовое окно с интервалом в одну секунду будет выводиться системное время компьютера.



Проект хранится в каталоге \textbook\VB\prj\VB10\ CD-ROM

Практические задания

4.20. Разработать проект, который позволяет определить количество дней, прошедших со дня вашего рождения.

4.21. Разработать проект, который выводит в текстовые поля текущие дату и время.

4.10. Графические возможности языка Visual Basic

На формах (`Form`) или в графических окнах (`PictureBox`) можно рисовать различные графические примитивы с использованием графических методов:

`Scale` — позволяет задать систему координат и масштаб для формы или графического окна:

```
object.Scale (X1,Y1) - (X2,Y2)
```

Аргументами метода являются `X1,Y1` — координаты левого верхнего угла объекта и `X2,Y2` — координаты правого нижнего угла объекта.

`Pset` — установка точки с заданными координатами и цветом:

```
object.Pset (X,Y) [,color]
```

Аргументами метода являются X,Y — координаты точки и color — цвет линии. Значение аргумента color можно задать различными способами:

- с помощью одной из восьми констант, определяющих цвет (vbBlack — черный, vbBlue — синий, vbGreen — зеленый, vbCyan — голубой, vbRed — красный, vbMagenta — сиреневый, vbYellow — желтый, vbWhite — белый);
- с помощью функции QBColor(number), аргументом которой являются числа от 0 до 15, а результат сотоветствует одному из основных 16 цветов;
- с помощью функции RGB(bytRed,bytGreen,bytBlue), аргументами которой являются три числа в диапазоне от 0 до 255 (интенсивности базовых цветов), а результатом — число типа **Long** в диапазоне от 0 до $256^3 - 1$ (16 777 215). Таким образом, определяется цветовая палитра с более чем 16 миллионами цветов, а каждый цвет задается числом, которое вычисляется по формуле bytRed + 256 · bytGreen + 256^2 · bytBlue.

В случае отсутствия аргумента color рисование будет производиться цветом, принятым по умолчанию (черным).

Line – рисование линии, прямоугольника или закрашенного прямоугольника заданного цвета:

```
object.Line (X1,Y1) - (X2,Y2) [,color][,B][F]
```

Аргументами метода являются X1,Y1 и X2,Y2 — координаты концов линии (левого верхнего и правого нижнего угла прямоугольника), color — цвет линии. Флажок **B** задает рисование прямоугольника, а флажок **F** — его закрашивание.

Circle – рисование окружности, овала или дуги с заданными координатами центра, радиусом, цветом, начальным и конечным углами дуги и коэффициентом сжатия:

```
object.Circle (X,Y),radius [,color, start, end,
aspect]
```

Аргументами метода являются X,Y — координаты центра окружности, radius — радиус окружности, color — цвет окружности, start и end — начальный и конечный угол дуги, aspect — коэффициент сжатия.

Если графический метод применяется к объекту «форма» (Form), то при его записи имя объекта object можно опускать.

Разработаем проект построения в графическом окне графика функции с использованием графических методов. В качестве примера рассмотрим построение графика функции $y = \sin x$.

Проект «Построение графика функции»

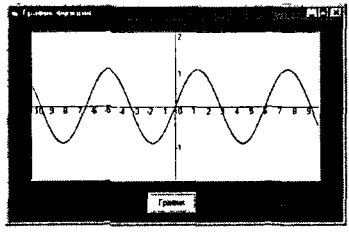
1. Разместить на форме графическое поле picGraph, в котором будет производиться построение графика.

Для большей понятности программного кода будем вводить в него комментарии, которые начинаются с символа апостроф «'».

2. Разместить на форме кнопку cmd1 и создать событийную процедуру построения графика, в которой устанавливается масштаб, в цикле осуществляется построение графика функции, рисуются оси координат и печатаются на них числовые шкалы:

```
Dim sngX As Single, intI As Integer
Sub cmd1_Click()
'Задание масштаба
picGraph.Scale (-10, 2)-(10, -2)
'Построение графика
For sngX = -10 To 10 Step 0.01
picGraph.PSet (sngX, Sin(sngX))
Next sngX
'Ось X
picGraph.Line (-10, 0)-(10, 0)
For intI = -10 To 10
picGraph.PSet (intI, 0)
picGraph.Print intI
Next intI
'Ось Y
picGraph.Line (0, 2)-(0, -2)
For intI = -2 To 2
picGraph.PSet (0, intI)
picGraph.Print intI
Next intI
End Sub
```

3. Запустить проект и щелкнуть по кнопке *График*.



Проект хранится в каталоге \textbook\VB\prjVB11\ CD-ROM 

Анимация. Для создания анимации (иллюзии движения на экране какого-либо объекта) применяется принцип смены кадров (изображений), как это делается в мультипликации. Программа, имитирующая движение, должна реализовывать следующие этапы:

- создание изображения на экране;
- реализация временно́й паузы для того, чтобы глаз зафиксировал изображение;
- проведение коррекции изображения.

Анимация часто используется для изображения движения объектов. Для регулирования скорости движения объекта используют пустой цикл: чем большее количество раз он будет выполняться, тем медленнее будет двигаться объект.

Проект «Движение круга»

1. Разместить на форме графическое поле `picAnim`, в котором будет производиться движение круга.

Движение по оси *OX* реализуем в цикле. Эффект анимации получим рисованием сначала круга синего цвета, а затем его стиранием кругом цвета фона (белым). Для получения закрашенного круга требуется установить значение свойства `FillColor` (при рисовании `vbBlue`, а при стирании `vbWhite`), которое определяет цвет заполнения рисуемой фигуры. Для свойства `FillStyle`, которое задает тип закраски (прозрачный — `Transparent` или непрозрачный — `Solid`) установить значение `Solid`.

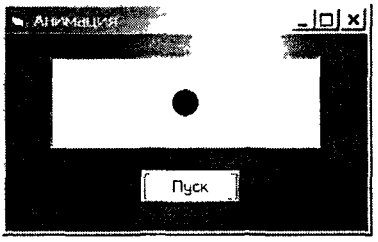
Скорость движения круга (скорость смены одного изображения другим) можно регулировать с помощью изменения количества повторений пустого цикла между рисованием и стиранием.

2. Поместить на форму кнопку `cmdStart` и создать событийную процедуру, реализующую анимацию:

```
Dim intX As Integer, lngI As Long
Private Sub cmdStart_Click()
'Масштаб
picAnim.Scale (-10, 10)-(10, -10)
'Анимация
For intX = -10 To 10
'Рисование
picAnim.FillColor = vbBlue
picAnim.Circle (intX, 0), 1, vbBlue
```

```
    'Задержка стирания
    For lngI = 1 To 10000000
    Next lngI
    'Стирание
    picAnim.FillColor = vbWhite
    picAnim.Circle (intX, 0), 1, vbWhite
    Next intX
    End Sub
```

3. Запустить проект и щелкнуть по кнопке *Старт*. В графическом поле начнется движение синего круга по оси *OX*.



Проект хранится в каталоге \textbook\VB\prjVB12\ CD-ROM

Практические задания

4.22. Разработать проект «Графический редактор», который позволяет нарисовать в графическом поле все графические примитивы (точку, линию, прямоугольник, закрашенный прямоугольник, окружность).

4.23. Создать проект, позволяющий задавать цвета различными способами (с помощью цветовых констант, функции `QBColor` и функции `RGB`) и демонстрирующий заданный цвет.

4.24. Изменить проект «Построение графика функции» так, чтобы дополнительно строились графики линейной, квадратичной и кубической функций.

4.11. Общие процедуры. Область видимости процедур

При разработке сложного алгоритма целесообразно стараться выделить в нем последовательности действий, которые выполняют решение каких-либо подзадач и могут многократно вызываться из основного алгоритма. Такие алгоритмы называются *вспомогательными* и в процедурных языках программирования реализуются в форме *подпрограмм*, которые вызываются из основной программы.

В объектно-ориентированных языках программирования вспомогательные алгоритмы реализуются с помощью *общих процедур*. Общие процедуры создаются в тех случаях, когда в программном модуле можно выделить многократно повторяющиеся последовательности действий (алгоритмы).

Определение общей процедуры. Каждой общей процедуре дается уникальное название – *имя процедуры* и устанавливается *список входных* и *выходных параметров* процедуры.

Список входных параметров представляет собой набор переменных, значение которых должно быть установлено до начала выполнения процедуры.

Список выходных параметров представляет собой набор переменных, значение которых должно быть установлено после окончания выполнения процедуры.

Синтаксис общей процедуры:

```
Sub ИмяПроцедуры(СписокПараметров)
программный код
End Sub
```

Вызов общей процедуры. Запуск общих процедур не связывается с какими-либо событиями, а реализуется путем вызова из других процедур.

Общая процедура представляет собой подпрограмму, которая начинает выполняться после ее вызова из другой процедуры.

Общая процедура вызывается на выполнение либо с помощью оператора **Call**, либо по имени. В случае вызова процедуры с использованием оператора **Call** список параметров заключается в скобки:

```
Call ИмяПроцедуры(СписокПараметров)
```

В случае вызова процедуры по имени список параметров приводится без скобок:

```
ИмяПроцедуры СписокПараметров
```

Размещение общей процедуры в проекте. Общая процедура может входить в состав программного модуля одной из форм проекта (в файл с расширением frm). Общая процедура может быть также размещена в стандартном программном модуле (файле с расширением bas).

Область видимости процедуры. Общие и событийные процедуры могут быть *локальными* и *глобальными*. Локальная процедура доступна только внутри данного программного модуля и не может быть вызвана из другого модуля. Например, локальная общая процедура, размещенная в программном модуле некоторой формы, не может быть вызвана из программного модуля другой формы. Локальная процедура задается с помощью ключевого слова **Private**:

```
Private Sub ИмяПроцедуры
программный код
End Sub
```

Глобальные процедуры доступны, то есть могут быть вызваны, из всех программных модулей проекта. Глобальная процедура задается с помощью ключевого слова **Public** (по умолчанию, если перед ключевым словом **Sub** отсутствуют ключевые слова, процедура является глобальной):

```
Public Sub ИмяПроцедуры
программный код
End Sub
```

Область видимости переменной. Переменные также могут быть *локальными* и *глобальными* (*открытыми*). Локальная переменная доступна только внутри процедуры или программного модуля и к ней невозможно обращение из другой процедуры или модуля. Локальная переменная определяется с помощью ключевого слова **Dim**.

Если переменная определена перед процедурой, то она может быть вызвана только в этой процедуре; если она определена перед программным модулем в области (*General*)(*Declaration*) программного кода, то она может быть вызвана только в этом модуле.

К глобальным переменным может быть произведено обращение из всех программных модулей проекта. Глобальная переменная определяется с помощью ключевого слова **Global** в области (*General*)(*Declaration*) программного кода.

Создадим проект, который наглядно продемонстрирует (с использованием графических методов) возможности использования общих процедур.

Пусть проект будет включать три формы. На первую форму должен выводиться рисунок простейшего домика, который будет состоять из стены (прямоугольник) и крыши (треугольник). На второй и третьей форме должны рисоваться несколько домиков различных размеров.

Программный модуль, реализующий рисование домика на первой форме, будет состоять из двух событийных процедур. Систему координат и масштаб зададим с помощью метода **Scale**, а для рисования стены и крыши будем использовать графический метод **Line**.

Проект «Рисование домика»

1. Разместить на форме frm1 две командные кнопки cmdСтена и cmdКрыша и создать для них событийные процедуры рисования стены и крыши:

```
Private Sub cmdСтена_Click()
Scale (0, 170)-(350, 0)
frm1.Line (20, 100)-(220, 20), , B
End Sub
Private Sub cmdКрыша_Click()
Scale (0, 170)-(350, 0)
frm1.Line (20, 100)-(220, 100)
frm1.Line (20, 100)-(120, 150)
frm1.Line (120, 150)-(220, 100)
End Sub
```

Теперь необходимо нарисовать несколько домиков различного размера на второй форме. Если использовать событийные процедуры, то для каждого домика нужно будет создавать свои процедуры, а это очень трудоемко.

Для рисования домика целесообразно создать общую процедуру Домик2(X1, X2, Y1, Y2 **As Single**), которая имеет *только список входных параметров (координаты углов* стены). Выходных параметров эта процедура не имеет.

Поместим код общей процедуры Домик2 в программный модуль первой формы и, для того чтобы ее вызов мог производиться из программного модуля второй формы, сделаем эту процедуру глобальной.

2. Ввести код глобальной общей процедуры Домик2 в программный код первой формы frm1:

```
Public Sub Домик2(X1, X2, Y1, Y2 As Single)
frm2.Line (X1, Y1)-(X2, Y2), , B
frm2.Line (X1, Y1)-(X2, Y1)
frm2.Line (X1, Y1)-((X1 + X2)/2, Y1 + Y1/2)
frm2.Line ((X1 + X2)/2, Y1 + Y1/2)-(X2, Y1)
End Sub
```

Создадим в программном модуле второй формы событийную процедуру cmdДомики2_Click(). Процедура будет обеспечивать рисование трех домиков путем вызова с помощью оператора **Call** из программного модуля первой формы общей процедуры Домик с различными значениями входных параметров.

3. Добавить в проект форму командой [Project-Add Form]. Поместить на форму frm2 кнопку cmdДомики2.
4. Ввести код событийной процедуры cmdДомики2_Click() в программный код второй формы frm2:

```
Private Sub cmdДомики2_Click()
frm2.Scale (0, 170)-(350, 0)
Call frm1.Домик2(10, 50, 50, 10)
Call frm1.Домик2(60, 150, 100, 40)
Call frm1.Домик2(160, 320, 110, 50)
End Sub
```

Для рисования трех домиков на третьей форме разместим общую процедуру Домик3 в отдельном стандартном программном модуле. В области (*General*)(*Declaration*) программного кода определим используемые в качестве координат на всех трех формах переменные X1, X2, Y1, Y2 как глобальные.

5. Добавить в проект стандартный программный модуль командой [Project-Add Module]. В окне *Программный код определить переменные и ввести процедуру:*

```
Global X1, X2, Y1, Y2 As Single
Public Sub Домик3(X1, X2, Y1, Y2 As Single)
frm3.Line (X1, Y1)-(X2, Y2), , B
frm3.Line (X1, Y1)-(X2, Y1)
frm3.Line (X1, Y1)-((X1 + X2)/2, Y1 + Y1/2)
frm3.Line ((X1 + X2)/2, Y1 + Y1/2)-(X2, Y1)
End Sub
```

6. Сохранить программный модуль в папке проекта командой [File-Save Module1.bas As ...].

Создадим в программном модуле третьей формы событийную процедуру cmdДомики3_Click(). Процедура будет обеспечивать рисование трех домиков путем вызова из стандартного программного модуля общей процедуры Домик с различными значениями входных параметров.

7. Добавить в проект форму командой [Project-Add Form]. Поместить на форму frm3 кнопку cmdДомики3.
8. Ввести код событийной процедуры cmdДомики3_Click() в программный код третьей формы frm3:

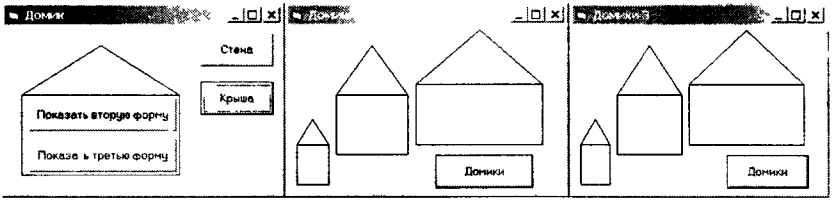
```
Private Sub cmdДомики3_Click()
frm3.Scale (0, 170)-(350, 0)
Module1.Домик 10, 50, 50, 10
Module1.Домик 60, 150, 100, 40
Module1.Домик 160, 320, 110, 50
End Sub
```

Для показа второй и третьей формы используем метод Show.

9. Поместить на форму frm1 кнопки cmd2 и cmd3. Создать для них событийные процедуры, реализующие показ второй и третьей форм:

```
Private Sub cmd2_Click()
frm2.Show
End Sub
Private Sub cmd3_Click()
frm3.Show
End Sub
```

10. Запустить проект. Последовательно щелкнуть по кнопкам событийных процедур. На первой форме будет нарисован один домик, а на второй и третьей формах по три домика.



Проект хранится в каталоге \textbook\VB\prjVB13\ CD-ROM

Вопросы для размышления

1. В чем состоит отличие общей процедуры от событийной процедуры?
2. Какой может быть общая процедура: локальной или глобальной?
3. Какие переменные — локальные или глобальные — могут использоваться в локальных процедурах? В глобальных процедурах?

Практические задания

4.25. Разработать проект, позволяющий рисовать на каждой из двух форм по пять треугольников.

4.12. Модульный принцип построения проекта и программного кода

В объектно-ориентированном программировании проект может включать несколько форм, причем каждой форме, с помощью которой реализуется графический интерфейс проекта, соответствует свой *программный модуль*. Кроме того, в состав проекта могут входить отдельные стандартные программные модули.

Проект включает в себя *программные модули форм и стандартные программные модули* в виде отдельных файлов. Проект может быть запущен на выполнение только из системы программирования Visual Basic.

Программный модуль формы может включать несколько *процедур*. В языке Visual Basic процедуры могут быть двух типов: *событийные* и *общие*. Событийные процедуры позволяют создавать интерактивные приложения, так как дают пользователю возможность выполнять тот или иной алгоритм с помощью определенного действия (например, щелчка по кнопке графического интерфейса). Общая процедура начинает выполняться после ее *вызова* из другой процедуры.

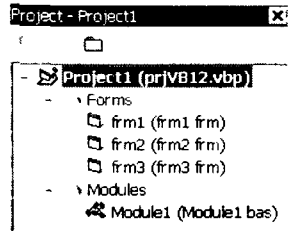
Ознакомимся с составом проекта на примере предыдущего проекта «Рисование домика». В этот проект входят три формы `frm1`, `frm2`, `frm3` и стандартный программный модуль `Module1`.

В программном модуле формы frm1.frm имеются одна глобальная общая процедура и четыре локальные событийные процедуры, в программных модулях frm2 frm и frm3.frm — по одной локальной событийной процедуре. В стандартном программном модуле Module1.bas находится глобальная общая процедура:

Проект prjVB12 vbp
Программный модуль формы frm1 frm Глобальная общая процедура Домик2(X1, X2, Y1, Y2 **As Single**) Локальная событийная процедура cmdСтена_Click Локальная событийная процедура cmdКрыша_Click Локальная событийная процедура cmd2_Click Локальная событийная процедура cmd3_Click
Программныи модуль формы frm2 frm Локальная событийная процедура cmdДомики2_Click
Программный модуль формы frm3 frm Локальная событийная процедура cmdДомики3_Click
Стандартный программный модуль Module1 bas Глобальная общая процедура Домик3(X1, X2, Y1, Y2 **As Single**)

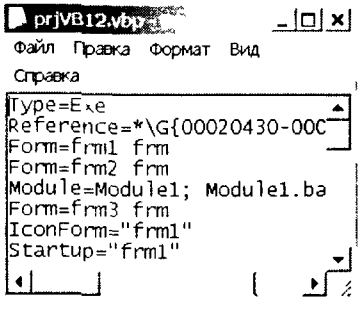
Проект и его компиляция в приложение

1. Запустить проект «Рисование домика». В окне *Проводник проекта* ознакомиться с файлами, которые входят в состав проекта: файл проекта prjVB12.vbp, программные модули форм frm1 frm, frm2.frm, frm3.frm и стандартный программный модуль Module1.bas.

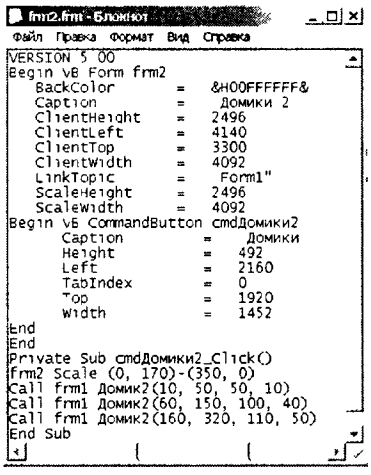


Файлы, входящие в проект, — текстовые, и поэтому проект может выполняться только в системе программирования Visual Basic. Для просмотра файлов проекта необходимо открыть их в текстовом редакторе, например в Блокноте.

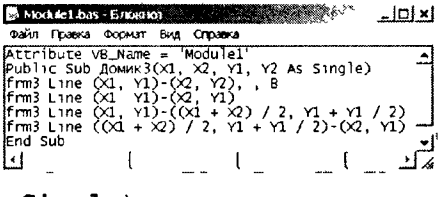
2. Ознакомиться с файлом проекта prjVB12.vbp, который содержит перечень входящих в него форм и стандартных программных модулей, определяет форму, с которой запускается проект, и др.



3. Ознакомиться с файлом формы, например формы frm2.frm. Файл формы содержит значения установленных свойств для всех объектов графического интерфейса (форма frm2 и кнопка cmdДомики2), а также программный код событийной процедуры cmdДомики2_Click().



4. Ознакомиться с файлом стандартного программного модуля Module1.bas. Файл содержит глобальную общую процедуру Домик3(X1,X2,Y1,Y2 **As Single**).



Проект, состоящий из текстовых файлов самого проекта, программных модулей форм и стандартных программных модулей, может выполняться только в самой системе программирования Visual Basic. Для того чтобы получить приложение, то есть программу, которая может выполняться непосредственно в среде операционной системы, необходимо осуществить компиляцию проекта в исполняемый файл (типа exe).

Приложение интегрирует программный код и графический интерфейс в одном исполняемом файле, который может запускаться вне системы программирования Visual Basic.

5. Сохранить проект в форме приложения. Для компиляции проекта в исполнимый файл ввести команду [File-Make ...].

Вопросы для размышления

1. Какие программные модули может включать в себя проект на языке Visual Basic? Каков может быть состав таких модулей?

Практические задания

4.26. Ознакомиться с составом проекта предыдущего задания. Просмотреть файлы проекта в текстовом виде.

4.13. Массивы

4.13.1. Типы и объявление массивов

Массив является набором переменных одного типа, объединенных одним именем. Массивы бывают *одномерные,* которые можно представить в форме одномерной таблицы, и *двумерные*, которые можно представить в форме двумерной таблицы.

Массив состоит из пронумерованной последовательности элементов. Номера в этой последовательности называются *индексами.* Каждый из этих элементов является переменной, то есть обладает именем и значением, и поэтому массив можно назвать переменной с индексом.

Обозначается массив следующим образом:

```
ИмяМассива(Индекс)
```

Массивы могут быть различных типов: *числовые, строковые* и так далее. Например, одномерный строковый массив `strA(I)`, содержащий буквы русского алфавита, можно представить себе в виде следующей таблицы:

I	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
A(I)	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я

Индексы являются целыми числами (в данном случае от 1 до 33). Обращение к элементу массива производится по имени элемента, состоящему из имени массива и значения индекса, например `strA(5)`.

Каждый элемент массива может принимать собственное значение. Так, значением элемента массива strA(5) является строка "д"

Объявление массива производится аналогично объявлению переменных, необходимо только дополнительно указать диапазон изменения индексов. После объявления массива для его хранения отводится определенное место в памяти.

Например, объявление одномерного строкового массива, содержащего 33 элемента, производится следующим образом:

```
Dim strA(1 To 33) As String
```

Вопросы для размышления

1. В чем состоит различие между переменной и массивом?

4.13.2. Заполнение массива

Для начала работы с массивом необходимо его предварительно заполнить, то есть присвоить элементам массива определенные значения. Заполнение массива можно производить различными способами.

Заполнение с клавиатуры. *Первый способ состоит в том,* что значения элементов массива вводятся пользователем с клавиатуры, например с помощью функции InputBox. Тогда для заполнения рассмотренного выше строкового массива strA(bytI) буквами русского алфавита можно использовать следующую событийную процедуру:

```
Dim strA(1 To 33) As String, bytI As Byte
' Заполнение массива с клавиатуры
Sub cmd1_Click()
For bytI = 1 To 33
strA(bytI) = InputBox("Введите букву", _
"Заполнение массива")
Next bytI
End Sub
```

Заполнение с помощью оператора присваивания. Второй способ заполнения массива состоит в использовании оператора присваивания. Заполним числовой массив bytA(bytI) целыми случайными числами в интервале от 1 до 100.

Для генерации последовательности случайных чисел используем функцию Rnd. При запуске программы функция Rnd дает равномерно распределенную псевдослучайную (то есть каждый раз повторяющуюся) последовательность чисел из интервала $0 \le X < 1$. Для того чтобы генерировались различающиеся между собой последовательности, можно использовать оператор Randomize.

Для получения последовательности случайных чисел в заданном интервале $A \le X < B$ необходимо использовать следующую формулу:

```
(B-A)* Rnd + A
```

Получение целочисленной последовательности случайных чисел из интервала $0 \le X < 100$ достигается использованием функции выделения целой части числа:

```
Int(Rnd * 100)
```

Создадим событийную процедуру для заполнения одномерного целочисленного массива случайными числами:

```
Dim bytA(1 To 100), bytI As Byte
' Заполнение массива присваиванием
Sub cmd1_Click()
For bytI = 1 To 100
bytA(bytI) = Int(Rnd * 100)
Next bytI
End Sub
```

Практические задания

4.27. Разработать проект, в котором массив заполняется значениями текущего времени.

4.13.3. Поиск в массивах

Поиск в строковых массивах. Поиск в строковых массивах обычно реализуется в форме поиска индекса элемента массива, значение которого совпадает с заданным.

Создадим проект, который в строковом массиве, содержащем русский алфавит, осуществляет поиск заданной буквы и определяет ее порядковый номер в алфавите.

Проект «Поиск в строковом массиве»

1. Поместить на форму frm1 кнопку cmd1 и создать для нее событийную процедуру cmd1_Click(), реализующую за-

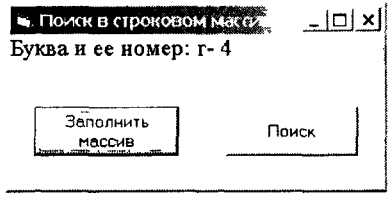
полнение массива буквами русского алфавита путем их последовательного ввода с клавиатуры.

Для реализации поиска создадим событийную процедуру, в которой запрашивается буква для поиска и в цикле производится ее сравнение со всеми элементами массива. В случае совпадения буквы и значения элемента массива переменной bytN присваивается значение индекса данного элемента. Результат поиска выведем на форму.

2. Поместить на форму кнопку cmd2 и создать для нее событийную процедуру cmd2_Click(), реализующую поиск:

```
Dim strA(1 To 33) As String, bytI, bytN As
Byte
'Поиск элемента
Sub cmd2_Click()
strB = InputBox("Введите искомую букву",
"Поиск")
For bytI = 1 To 33
If strB = strA(bytI) Then bytN = bytI
Next bytI
frm1.Print "Буква и ее номер: "; strB; "-";
bytN
End Sub
```

3. Запустить проект. Щелкнуть по кнопке *Заполнить массив* и последовательно ввести буквы русского алфавита. Щелкнуть по кнопке *Поиск* и ввести искомую букву, например «г».



Проект хранится в каталоге \textbook\VB\prjVB14\ CD-ROM

Поиск в числовых массивах. В числовых массивах обычно производится поиск наименьшего или наибольшего элемента.

Осуществим поиск наименьшего элемента (то есть элемента с наименьшим *значением*) в числовом массиве, состоящем из 100 элементов.

Сначала заполним массив целыми случайными числами в цикле с использованием генератора случайных чисел функции Rnd и функции выделения целой части числа Int.

Проект «Поиск в числовом массиве»

1. Поместить на форму frm1 кнопку cmd1 и создать для нее событийную процедуру cmd1_Click(), реализующую заполнение массива случайными числами.

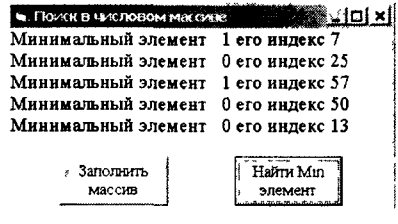
Значение минимального элемента будем хранить в переменной bytMin, а его индекс — в переменной bytN. Будем считать сначала, что минимальный элемент равен первому элементу массива bytA(1), поэтому присвоим переменной bytMin его значение.

Затем в цикле сравним последовательно элементы массива со значением переменной bytMin. Если какой-либо элемент окажется меньше, присвоим его значение переменной bytMin, а его индекс — переменной bytN — и так до самого последнего элемента. Результат поиска выведем на форму.

2. Поместить на форму кнопку cmd2 и создать для нее событийную процедуру cmd2_Click(), реализующую поиск:

```
Dim bytA (1 To 100), bytI, bytN As Byte
Sub cmd2_Click()
'Поиск минимального элемента
bytMin = bytA(1)
bytN = 1
For bytI = 2 To 100
If bytA(bytI) < bytMin Then
bytMin = bytA(bytI) : bytN = bytI
Next bytI
frm1.Print "Минимальный элемент  "; bytMin;
"его индекс"; bytN
End Sub
```

3. Запустить проект. Щелкнуть по кнопкам *Заполнить массив* и *Найти Min элемент* несколько раз. На форме будут напечатаны результаты поиска минимального элемента для различных вариантов заполнения массива.



Проект хранится в каталоге \textbook\VB\prjVB15\ CD-ROM

Практические задания

4.28. Усовершенствовать проект «Поиск в строковом массиве» так, чтобы заполнение массива и ввод искомого символа происходили с помощью текстовых полей.

4.29. Создать проект поиска максимального элемента числового массива.

4.13.4. Сортировка массива

Рассмотрим задачу упорядочения (сортировки) числового массива по возрастанию значений его элементов. Пусть у нас имеется целочисленный числовой массив `bytA(bytI)`, состоящий из 10 элементов и заполненный случайными числами. Представим массив в виде таблицы (табл. 5.9).

Таблица 5.9. Алгоритм сортировки массива по возрастанию

I	1	2	3	4	5	6	7	8	9	10	Значение счетчика цикла
bytA(bytI)	**70**	**53**	**57**	**28**	**30**	**77**	**1**	**76**	**81**	**70**	До сортировки
	70	←					→ 1				1
		53	←	→ 28							2
			57	←	→ 30						3
				53							4
					57						5
						77	←	→ 70			6
							77	←	→ 70		7
								76			8
									81	← → 77	9
bytA(bytI)	**1**	**28**	**30**	**53**	**57**	**70**	**70**	**76**	**77**	**81**	После сортировки

Идея алгоритма состоит в следующем. Проводим поиск минимального элемента в массиве среди элементов с 1-го по 10-й. Далее меняем найденный минимальный элемент местами с элементом с индексом 1 (выполняем перестановку).

Проводим поиск минимального элемента среди элементов со 2-го по 10-й и делаем перестановку.

Повторяем процедуру поиска минимального элемента среди оставшихся неупорядоченных элементов многократно. Повторение реализуем с помощью цикла со счетчиком, максимальное значение которого составляет $N-1$, где N — количество элементов массива (в рассматриваемом случае цикл повторяется 9 раз). В результате массив упорядочивается.

Поиск минимального элемента проводится многократно, поэтому реализуем его как общую процедуру МинЭлемент (bytI, bytN **As Byte**), где bytI является входным параметром, а bytN — выходным параметром.

Таким образом, программный модуль формы должен содержать: событийную процедуру заполнения массива случайными числами, событийную процедуру сортировки и общую процедуру поиска минимального элемента.

Проект «Сортировка числового массива»

1. Поместить на форму frm1 кнопку cmd1 и создать для нее событийную процедуру cmd1_Click(), реализующую заполнение массива случайными числами.
2. Определить переменные для всего программного модуля. Преобразовать событийную процедуру из проекта «Поиск в числовом массиве» в общую процедуру МинЭлемент(bytI, bytN **As Byte**):

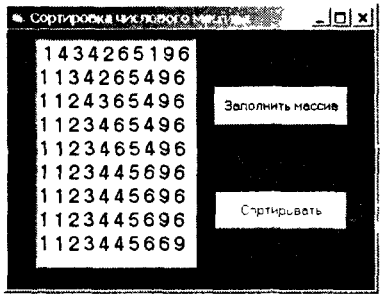
```
Dim bytA(1 To 10), bytMin, bytI, bytJ, bytK,
bytR, bytN As Byte
'Общая процедура поиска минимального элемента
Sub МинЭлемент(bytI, bytN As Byte)
bytMin = bytA(bytI)
bytN = bytI
For bytJ = bytI + 1 To 10
If bytA(bytJ) < bytMin Then
bytMin = bytA(bytJ): bytN = bytJ
Next bytJ
End Sub
```

Создать событийную процедуру сортировки. Для осуществления перестановки использовать промежуточную переменную bytR. Для визуализации процесса сортировки для каждого цикла перестановки элементов (цикл по переменной bytI) в цикле по переменной bytK выводить в текстовое поле txtSort значения элементов массива.

3.
```
'Событийная процедура сортировки
Private Sub cmd2_Click()
txtSort.Text = ""
For bytI = 1 To 9
'Вызов общей процедуры поиска минимального
элемента
Call МинЭлемент(bytI, bytN)
'Перестановка
bytR = bytA(bytI)
bytA(bytI) = bytA(bytN)
```

```
    bytA(bytN) = bytR
    'Печать массива для каждого цикла перестановки
    For bytK = 1 To 10
    txtSort.Text = txtSort.Text + Str(bytA(bytK))
    Next bytK
    Next bytI
    End Sub
```

4. Запустить проект. Щелкнуть по кнопкам *Заполнить массив* и *Сортировать*. В текстовом поле будет реализована визуализация процесса сортировки числового массива по шагам.



Проект хранится в каталоге \textbook\VB\prjVB16\ CD-ROM

Практические задания

4.30. Разработать проект, в котором реализуется сортировка числового массива по убыванию с использованием общей процедуры поиска максимального элемента.

4.13.5. Двумерные массивы и вложенные циклы

Двумерные массивы можно представить себе как таблицы, в ячейках которых хранятся значения элементов массива, а индексы элементов массива являются номерами строк и столбцов.

Объявляются двумерные массивы так же, как переменные и одномерные массивы. Например, целочисленный числовой массив, содержащий 9 строк и 9 столбцов, объявляется следующим образом:

```
Dim bytA(1 To 9, 1 To 9) As Byte
```

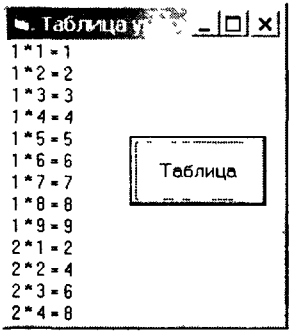
С помощью такого массива и двух вложенных циклов легко можно составить программу, реализующую таблицу умножения. Сомножителями будут значения индексов строк и столбцов, а их произведения будут значениями элементов массива.

Проект «Таблица умножения»

1. Поместить на форму frm1 кнопку cmd1 и создать для нее событийную процедуру cmd1_Click(), реализующую печать таблицы умножения на форме:

```
Dim bytA(1 To 9, 1 To 9), bytK, bytN As Byte
Sub Command1_Click()
For bytK = 1 To 9
For bytN = 1 To 9
bytA(bytK, bytN) = bytK * bytN
frm1.Print bytK; "*"; bytN; "="; bytA(bytK,
bytN)
Next bytN
Next bytK
End Sub
```

2. Запустить проект. Щелкнуть по кнопке *Таблица*. *Во внешнем цикле счетчик* K примет 9 различных значений, для каждого из которых счетчик N внутреннего цикла примет также 9 значений. Таким образом, тело вложенных циклов будет выполнено K*N раз, то есть 81 раз.



Проект хранится в каталоге \textbook\VB\prjVB17\ **CD-ROM**

Практические задания

4.31. Разработать проект, в котором моделируется движение электронного луча по экрану монитора.

4.14. Решение логических задач

Глава 3. Основы логики и логические основы компьютера

В языке Visual Basic основные логические операции могут быть реализованы с помощью логических операторов:

- **And** (логическое умножение);
- **Or** (логическое сложение);

- **Not** (логическое отрицание);
- **Xor** (исключающее «или», которое принимает логическое значение **True** тогда и только тогда, когда лишь один из аргументов имеет значение **True**);
- **Eqv** (операция эквивалентности, которая принимает логическое значение **True**, когда оба аргумента имеют одинаковые значения — **True** или **False**).

Логические операторы могут оперировать с логическими аргументами **True** (логическая единица) и **False** (логический нуль), а также с логическими переменными типа **Boolean**.

На языке Visual Basic можно составлять программы, которые формируют таблицы истинности логических выражений. Таблицы истинности содержат значения логических функций при всех возможных комбинациях значений аргументов.

Таким образом, задача построения таблицы истинности сводится к перебору всех возможных комбинаций значений аргументов и вычислению значений функции для каждой такой комбинации. Это можно реализовать с помощью вложенных циклов со счетчиком, в каждом из которых рассматриваются два значения аргументов: **True** и **False**.

Однако в цикле со счетчиком переменная `Счетчик` должна быть обязательно числового типа, логические значения она принимать не может. Поэтому необходимо использовать числовую форму представления логических значений: логическому значению **False** соответствует число 0, а логическому значению **True** соответствует -1.

Таким образом, для того чтобы таблица истинности выводилась программой в привычном виде, необходимо при выводе ее на печать перед аргументами и функцией ставить знак «–».

Составим, например, программу для получения таблицы истинности операции логического умножения.

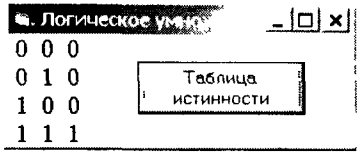
Проект «Таблица истинности операции логического умножения»

1. Поместить на форму `frm1` кнопку `cmd1` и создать для нее событийную процедуру `cmd1_Click()`, печатающую таблицу истинности на форме:

```
Dim intA, intB As Integer
Sub cmd1_Click()
For intA = 0 To -1 Step -1
```

```
For intB = 0 To -1 Step -1
frm1.Print -intA; -intB; -(intA And intB)
Next intB
Next intA
End Sub
```

2. Запустить проект и щелкнуть по кнопке *Таблица истинности*. На форме будет напечатана таблица истинности операции логического умножения.



Проект хранится в каталоге \textbook\VB\prjVB18\ CD-ROM

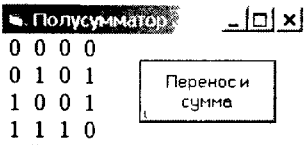
Аналогично можно получать таблицы истинности логических выражений. Разработаем проект, который позволяет получить таблицы истинности логических выражений, которые реализуют перенос и сумму в полусумматоре двоичных чисел.

Проект «Таблицы истинности переноса и суммы в полусумматоре»

1. Поместить на форму `frm1` кнопку `cmd1` и создать для нее событийную процедуру `cmd1_Click()`, печатающую таблицы истинности переноса и суммы на форме:

```
Dim intA, intB, intP, intS As Integer
Sub cmd1_Click()
For intA = 0 To -1 Step -1
For intB = 0 To -1 Step -1
intP = intA And intB
intS = (intA Or intB) And Not (intA And intB)
frm1.Print -intA; -intB; -intP; -intS
Next intB
Next intA
End Sub
```

2. Запустить проект и щелкнуть по кнопке *Перенос и сумма*. На форме будут напечатаны таблицы истинности переноса и суммы полусумматора двоичных чисел.



Проект хранится в каталоге \textbook\VB\prjVB19\ CD-ROM

В параграфе 3.6 была решена логическая задача, касающаяся размещения в двух аудиториях кабинетов информатики и физики. Было получено логическое выражение, которое должно принимать значение «Истина»:

$$((A \vee B) \,\&\, \overline{A}) \vee (\overline{(A \vee B)} \,\&\, \overline{\overline{A}}) = 1.$$

Задача была решена упрощением логического выражения путем его преобразования с использованием формул законов логики.

Теперь решим эту задачу с помощью программирования. Запишем это логическое выражение на языке Visual Basic:

```
((A Or B) And Not A) Or ((Not(A Or B)) And
(Not(Not A)))
```

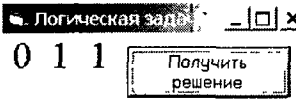
В качестве условия в операторе условного перехода запишем истинность этого логического выражения: `C = -1`. Распечатаем результат, то есть значения логических переменных, при которых это условие выполняется.

Проект «Решение логической задачи»

1. Поместить на форму `frm1` кнопку `cmd1` и создать для нее событийную процедуру `cmd1_Click()`, печатающую значения аргументов и логического выражения:

```
Dim intA, intB, intC As Integer
Sub cmd1_Click()
For intA = 0 To -1 Step -1
For intB = 0 To -1 Step -1
intC = ((intA Or intB) And Not intA) Or ((Not
(intA Or intB)) And (Not (Not intA)))
If intC = -1 Then frm1.Print -intA; -intB;
-intC
Next intB
Next intA
End Sub
```

2. Запустить проект и щелкнуть по кнопке *Получить решение*. На форме напечатан результат 0, 1, 1, что совпадает с результатом алгебраического решения $A = 0$ и $B = 1$, то есть в первой аудитории находится кабинет физики, а во второй – кабинет информатики.



Проект хранится в каталоге \textbook\VB\prjVB20\ CD-ROM

Практические задания

4.32. Разработать проект, в котором на форму выводятся таблицы истинности операций логического умножения, сложения, исключающего «или» и эквивалентности.

4.33. Разработать проект получения таблицы истинности логических выражений, реализующих перенос и сумму в полном одноразрядном сумматоре двоичных чисел.

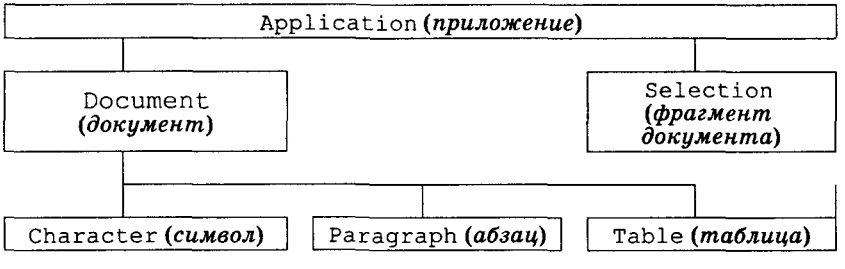
4.15. Язык объектно-ориентированного программирования Visual Basic for Applications

Язык объектно-ориентированного программирования Visual Basic for Applications предназначен для разработки приложений в среде Windows&Office.

4.15.1. Иерархия объектов в VBA

В VBA имеется более 100 различных классов объектов, которые образуют некоторую иерархию. На вершине иерархии находится объект `Application` (*приложение*), который включает все остальные объекты. Так, объект `Application`, вершина иерархии объектов приложения Word, включает в себя в том числе объекты `Document` (*документ*) и `Selection` (*выделенный фрагмент документа*), а объект `Document` включает объекты `Character` (*символ*), `Paragraph` (*абзац*), `Table` (*таблица*) и др. (табл. 4.6).

Таблица 4.6. Некоторые объекты приложения Word



На вершине иерархии объектов приложения Excel стоит объект Application (*приложение*), в котором может быть открыто несколько книг (семейство объектов Workbooks), каждая из которых содержит несколько листов (семейство Worksheets), на каждом из которых может быть выбран диапазон ячеек (семейство Range), которые включают в себя определенные ячейки (семейство Cells), и так далее.

Для обращения к объекту в объектно-ориентированном программировании используется ссылка на объект, которая состоит из ряда имен вложенных последовательно друг в друга объектов. В соответствии с принятой в объектно-ориентированном программировании *точечной нотацией* разделителями имен объектов в этом ряду являются точки, ряд начинается с объекта наиболее высокого уровня Application и заканчивается именем интересующего нас объекта.

Например, в приложении Excel ссылка на ячейку A1 будет выглядеть следующим образом:

```
Application.Workbooks("Проба.xls").
Worksheets("Лист1").Cells(1,1)
```

Однако делать каждый раз полную ссылку на объект необязательно. Если объект является активным, например, если в приложении Excel открыт документ Workbooks ("Проба.xls")и активным является лист Worksheets ("Лист1"), достаточно сделать относительную ссылку на саму ячейку:

```
Cells(1,1)
```

4.15.2. Интегрированная среда разработки языка VBA

4.4. Интегрированная среда разработки языка Visual Basic

Интерфейс интегрированной среды разработки VBA аналогичен интерфейсу интегрированной среды разработки Visual Basic. Запуск среды разработки VBA осуществляется из любого приложения, входящего в Microsoft Office, командой [Сервис-Макрос-Редактор Visual Basic].

Окно среды разработки VBA представляет собой стандартное окно приложения, в котором могут быть открыты с помощью пунктов меню *View* рабочие окна среды: *панель инструментов*, окна *Конструктор форм, Свойства объекта, Программный код, Просмотр объектов* и *Проводник проекта* (рис. 4.18).

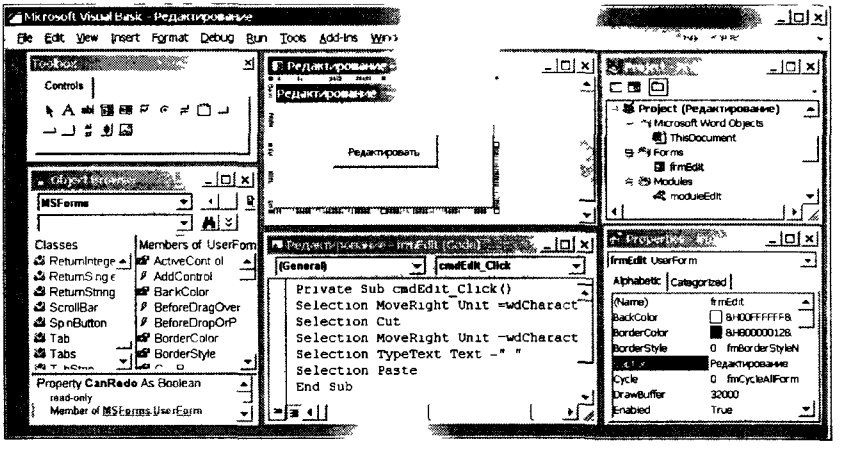


Рис. 4.18. Окно интегрированной среды разработки VBA

Панель инструментов в стандартном варианте включает 14 различных классов управляющих элементов: `CommandButton` (*командная кнопка*), `TextBox` (*текстовое поле*), `Label` (*надпись*) и др. Существует возможность дополнить *панель инструментов* новыми классами управляющих элементов с помощью команды [Tools-Additional Controls...].

В окне *Конструктор форм* можно конструировать графический интерфейс проектов, помещая на форму (объект `UserForm`) управляющие элементы.

С помощью окна *Свойства объекта* можно устанавливать требуемые значения свойств (`Name`, `Caption` и др.) объектов графического интерфейса (формы и управляющих элементов).

В окне *Программный код* можно вводить и редактировать код программных модулей. Редактор кода оказывает «интеллектуальную» помощь пользователю. Во-первых, он автоматически предлагает пользователю варианты завершения вводимой инструкции.

После ввода имени объекта (например, объекта Selection) и точки в окне появляется список его свойств, методов, а также входящих в него объектов более низкого уровня (рис. 4.19).

Двойной щелчок по выбранному объекту или нажатие клавиши {*Tab*} вставляет выбранное имя в код программы.

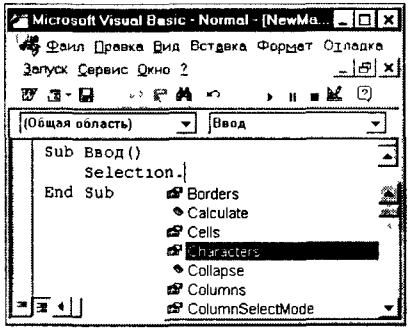


Рис. 4.19. Перечень свойств и методов выбранного объекта Selection

Во-вторых, в любой момент можно получить перечень всех классов объектов VBA, а для каждого класса — перечень свойств и методов. Для этого необходимо выделить в тексте программы имя объекта, осуществить правый щелчок и в появившемся контекстном меню выбрать пункт *Просмотр объектов* или ввести команду [Вид-Просмотр объектов].

В появившемся окне *Просмотр объектов* в его левой части можно выбрать класс объектов (например, Documents), а в правой части окна выбрать интересующее нас свойство или метод (например, метод Open). В нижней части окна будет выведена краткая информация о формате задания метода или свойства (рис. 4.20).

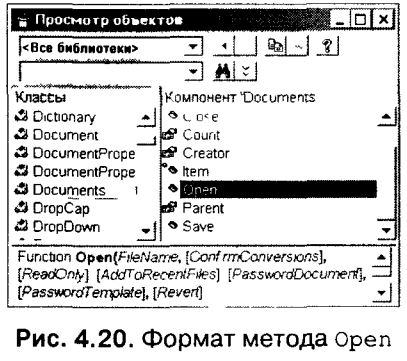


Рис. 4.20. Формат метода Open

В-третьих, редактор кода проводит автоматическую проверку синтаксиса набранной строки после нажатия клавиши {*Enter*}. Если в строке допущена ошибка, то она выделяется красным цветом. Для устранения ошибки в помощь пользователю на экране отображается диалоговая панель, на которой содержится информация о допущенной ошибке.

В-четвертых, если курсор расположить на ключевом слове языка VBA (имени объекта, метода, свойства или инструкции) и нажать клавишу {*F1*}, то появится окно с развернутой справочной информацией.

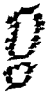
В окне *Проводник проекта* можно ознакомиться с иерархической структурой файлов программных модулей проек-

та. В проекте автоматически создаются программные модули для каждого документа, кроме того, можно создать программные модули форм и стандартные программные модули (макросы).

4.15.3. Кодирование алгоритмов в форме макросов

В программной среде Windows&Office существует достаточно простая возможность кодирования алгоритмов, связанных с преобразованием документов. Для этого пользователь должен выполнить алгоритм вручную, то есть ввести последовательность команд путем активизации пунктов меню приложения и нажатия клавиш клавиатуры. Последовательность действий пользователя будет записана в форме *макроса* (стандартного программного модуля на языке VBA).

Каждому макросу задается имя, а для быстрого запуска макроса можно создать кнопку макроса или присвоить ему «горячую» клавишу (клавишу, по нажатию на которую будет производиться запуск макроса). После запуска макрос будет автоматически выполнен тем приложением, в котором он запущен (Word, Excel и др.).



Макрос — это имеющая имя последовательность заданных пользователем команд, хранящаяся в форме стандартного программного модуля на языке VBA.

Макросы в приложении Word. В любом документе приложения Word можно создавать макросы, реализующие редактирование и форматированиие документа. Готовые макросы, являющиеся стандартными программными модулями на VBA, можно многократно запускать на выполнение. При необходимости программный код макроса можно редактировать.

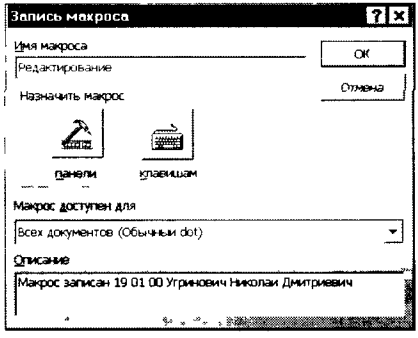
Создадим макрос, который будет содержать код на языке VBA рассмотренного ранее алгоритма *«Редактирование»*. Проанализируем полученный программный код стандартного макроса.

4.1. Алгоритм и его формальное исполнение

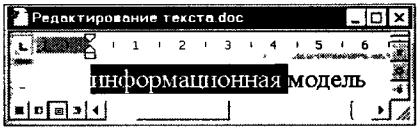
Сначала необходимо осуществить запись макроса.

Макрос *Редактирование*

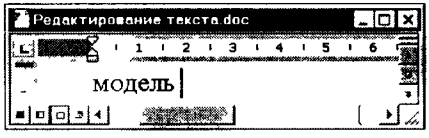
1. Открыть в приложении Word исходный документ Редактирование текста.doc, содержащий текст «информационная модель».
2. Ввести команду [Сервис-Макрос-Начать запись...].
   На появившейся диалоговой панели *Запись макроса* в поле *Имя макроса:* задать имя макроса — *Редактирование.*
   Для быстрого запуска макроса можно создать кнопку или назначить ему «горячую» клавишу.



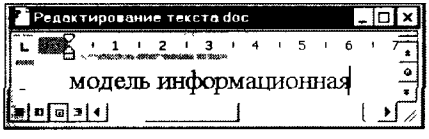
3. Выполнить алгоритм *«Редактирование»* (вручную выполнить последовательность команд):
   1) с помощью клавиатуры выделить символы с 1-го по 15-й;



   2) ввести команду [Правка-Вырезать];
   3) с помощью клавиатуры установить курсор после позиции 7-го символа;



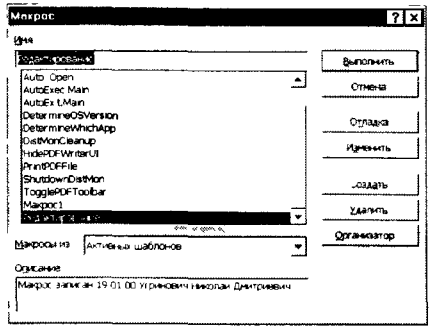
   4) ввести команду [Правка-Вставить] и получить результат выполнения алгоритма.



4. Ввести команду [Сервис-Макрос-Остановить запись].

Созданный макрос *Редактирование* является стандартным программным модулем на языке VBA, в котором закодирован алгоритм *«Редактирование»*. Теперь макрос может быть выполнен исполнителем (приложением Word) в автоматическом режиме.

5. Открыть в приложении Word исходный документ.
   Ввести команду [Сервис-Макрос-Макросы...]. На открывшейся диалоговой панели *Макросы* выбрать имя нужного макроса и щелкнуть по кнопке *Выполнить*.



Макрос хранится в файле Редактирование текста.doc ***в каталоге*** \textbook\VBA\ **CD-ROM**

Макрос является стандартным программным модулем на языке VBA, поэтому имеется возможность его доработки путем редактирования непосредственно программного кода.

6. Открыть в приложении Word исходный документ. Ввести команду [Сервис-Макрос-Макросы...]. На открывшейся диалоговой панели *Макросы* выбрать имя нужного макроса и щелкнуть по кнопке *Изменить*.
7. Появится окно *Программный код*, содержащее программный код стандартного программного модуля:

```
Sub Редактирование()
Selection.MoveRight Unit:=wdCharacter, _
Count:=15, Extend:=wdExtend
Selection.Cut
Selection.MoveRight Unit:=wdCharacter, Count:=7
Selection.Paste
End Sub
```

Проанализируем полученный программный код стандартного модуля. В алгоритме производятся операции над фрагментом текста (объект `Selection`), поэтому в программном модуле используются методы, которыми обладает этот объект.

В первой строке программы для выделения символов применяется метод `MoveRight` (переместить вправо) с тремя аргументами. Первый аргумент определяет единицу перемещения по документу и получает значение символа текста (`Unit:=wdCharacter`). Второй аргумент задает количество символов (`Count:=15`). Третий аргумент фиксирует, что все символы необходимо выделить (`Extend:=wdExtend`).

Во второй строке для вырезания символов и помещения их в буфер к выделенному фрагменту применяется метод `Cut` (*вырезать*).

В третьей строке для перемещения курсора и установки его в определенную позицию в тексте опять применяется метод `MoveRight`, но уже только с двумя аргументами: `Unit:=wdCharacter, Count:=7`. В этом случае курсор просто перемещается на заданное количество символов без их выделения.

В четвертой строке для вставки из буфера находящегося в нем вырезанного фрагмента текста применяется метод `Paste` (*вставить*).

Макросы в приложении Excel. Создадим макрос, который будет содержать код на языке VBA алгоритма «*Умножение*», и проанализируем программный код полученного стандартного программного модуля.

Запишем алгоритм «*Умножение*», который предусматривает ввод и умножение чисел:

1. Ввести число (например, 2) в первую ячейку (например, A1).
2. Ввести число (например, 3) во вторую ячейку (например, B1).
3. Ввести формулу (например, =A1*B1) в третью ячейку (например, C1).

Макрос *Умножение*

1. Запустить приложение Excel. Ввести команду [Сервис-Макрос-Начать запись...]. На появившейся диалоговой панели *Запись макроса в поле Имя макроса:* задать макросу имя *Умножение*.
2. Выполнить алгоритм «*Умножение*» вручную:
   1) выделить ячейку A1 и ввести в нее число 2;
   2) выделить ячейку B1 и ввести в нее число 3;
   3) выделить ячейку C1 и ввести в нее формулу =A1*B1.
3. Ввести команду [Сервис-Макрос-Остановить запись].

Макрос хранится в файле Макросы и проекты xls *в каталоге* \textbook\VBA\ CD-ROM

4. Ввести команду [Сервис-Макрос-Макросы...]. На открывшейся диалоговой панели *Макросы* выбрать макрос *Умножение* и щелкнуть по кнопке *Изменить*.

5. Появится окно, содержащее программный код стандартного модуля:

```
Sub Умножение()
ActiveCell.FormulaR1C1 = "2"
Range("B1").Select
ActiveCell.FormulaR1C1 = "3"
Range("C1").Select
ActiveCell.FormulaR1C1 = "=RC[-2]*RC[-1]"
End Sub
```

Проанализируем полученный программный код макроса *Умножение*. Первая строка содержит имя объекта `ActiveCell` (*активная ячейка*), свойству которого `FormulaR1C1` (*формула*) присваивается числовое значение 2.

Во второй строке выделяется ячейка B1 с помощью объекта `Range("B1")` и его метода `Select`.

Третья и четвертые строки аналогичны по своему синтаксису строкам, рассмотренным выше.

В пятой строке в активную ячейку вводится формула. Свойство `FormulaR1C1` требует ввода адресов ячеек в формате `R1C1`, в этом случае `Rows` (*строки*) и `Columns` (*столбцы*) отсчитываются от активной ячейки. После знака «–» указывается величина смещения от активной ячейки, которая в данном формате имеет адрес R1C3 или в обычном формате записи — C1. Следовательно, запись `"=RC[-2]*RC[-1]"` соответствует записи `"=A1*B1"`.

Практические задания

4.34. Записать макрос, который преобразует слово «информатика» в слово «форма». Проанализировать программный код полученного стандартного модуля.

4.15.4. Создание проектов

С помощью VBA можно создавать проекты с графическим интерфейсом.

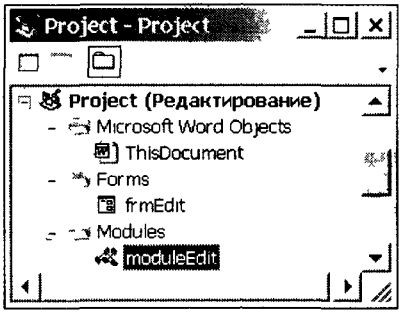
Проект в приложении Word. Преобразуем макрос (стандартный программный модуль) *Редактирование*, который осуществляет редактирование документа в приложении Word, в проект «Редактирование» с графическим интерфейсом в виде кнопки на форме.

Проект «Редактирование»

1. Открыть в приложении Word исходный документ Редактирование текста.doc. Запустить интегрированную среду разработки VBA командой [Сервис-Макрос-Редактор Visual Basic].
2. Добавить в проект форму `UserForm1` командой [Insert-UserForm].
3. С помощью *панели инструментов* поместить на форму кнопку `CommandButton1`.
4. С помощью окна *Свойства объекта* присвоить свойству `Name` формы и кнопки новые значения `frmEdit` и `cmdEdit`. Изменить также надписи, цвет и так далее.

Создадим событийную процедуру и скопируем в нее программный код макроса.

5. Осуществить двойной щелчок по кнопке `cmdEdit`. В окне *Программный код* появится заготовка событийной процедуры `cmdEdit_Click()`.
6. В окне *Проводник* проекта видно, что в состав проекта «Редактирование» входят программный модуль документа `ThisDocument`, программный модуль формы `frmEdit` и стандартный программный модуль `moduleEdit`.

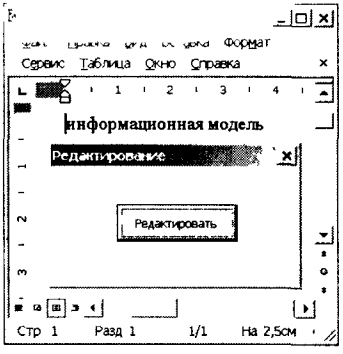


Осуществить двойной щелчок на имени стандартного программного модуля.

7. В открывшемся окне *Программный код* выделить код модуля и скопировать его в заготовку событийной процедуры. Событийная процедура примет вид:

```
Private Sub cmdEdit_Click()
Selection.MoveRight Unit:=wdCharacter, _
Count:=15, Extend:=wdExtend
Selection.Cut
Selection.MoveRight Unit:=wdCharacter, Count:=7
Selection.Paste
End Sub
```

8. В окне *Проводник* проекта осуществить двойной щелчок на имени программного модуля формы `frmEdit` и запустить проект на выполнение. Появится окно документа с формой внутри. Щелкнуть по кнопке *Редактировать*, будет выполнено редактирование исходного текста.



В состав проекта на VBA может входить несколько программных модулей: программный модуль документа, программные модули форм и стандартные программные модули (макросы). Однако все эти модули хранятся в одном файле – файле самого документа.

9. Сохранить проект командой [File-Save Редактирование текста.doc].

Проект хранится в файле Редактирование текста.doc ***в каталоге*** \textbook\VBA\ **CD-ROM**

Проект в приложении Excel. Разработаем в приложении Excel проект «Калькулятор», который будет выполнять четыре арифметических действия.

4.9.2. Математические функции

Интерактивный диалог пользователя будем осуществлять с помощью графического интерфейса, а вычисления — с использованием возможностей электронных таблиц.

Проект «Калькулятор»

1. Открыть в приложении Excel исходный документ Макросы и проекты.xls. Запустить интегрированную среду разработки VBA командой [Сервис-Макрос-Редактор Visual Basic].

Сначала сконструируем графический интерфейс проекта. Для ввода аргументов и для вывода результата арифметических действий используем текстовые поля, а для вызова событийных процедур, реализующих арифметические действия, — кнопки.

2. Добавить в проект форму `UserForm1` командой [Insert-UserForm]. Присвоить ей имя `frmCalc`.

3. С помощью *панели инструментов* поместить на форму три текстовых поля и пять кнопок. Присвоить полям имена `txt1`, `txt2`, `txt3`, а кнопкам — `cmdPlus`, `cmdMinus`, `cmdUmn`, `cmdDelen`, `cmdExit`.

Создадим теперь событийные процедуры сложения, вычитания, умножения, деления и выхода из программы. Сначала необходимо присвоить ячейкам электронной таблицы с помощью функции `Val` числовые значения, которые будут вводиться в текстовые поля аргументов. Затем в текстовое поле результата необходимо вывести значение одной из ячеек, содержащих формулы арифметических операций.

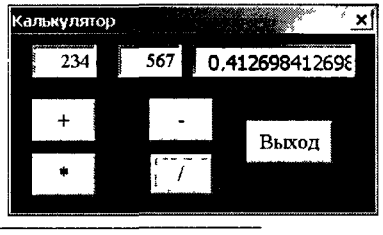
Аргументы в электронной таблице будут храниться в ячейках A1 и B1, а формулы сложения, вычитания, умножения и деления — соответственно в ячейках C1, C2, C3 и C4.

При адресации ячеек необходимо помнить, что в VBA ячейки являются объектами семейства `Cells (R,C)`, где `R` — номер строки, а `C` — номер столбца.

4. Для каждой из кнопок ввести программные коды событийных процедур. Программный код событийной процедуры сложения `cmdPlus_Click()` будет следующий:

```
Private Sub cmdPlus_Click()
Cells(1, 1) = Val(txt1.Text)
Cells(1, 2) = Val(txt2.Text)
txt3.Text = Cells(1, 3)
End Sub
```

5. Установить для свойства `Alignment` текстовых полей значение `Right Justufy` и подходящий размер шрифта с использованием свойства `Font`.
6. Запустить проект на выполнение. Ввести числа в два левых текстовых поля и щелкнуть по кнопке арифметической операции. В правом поле будет выведен результат.



Проект хранится в файле Макросы и проекты xls ***в каталоге*** \textbook\VBA\ **CD-ROM**

Практические задания

4.35. На базе проекта «Калькулятор» создать проект «Инженерный калькулятор».

Глава 5

Моделирование и формализация

5.1. Моделирование как метод познания

Моделирование. Человечество в своей деятельности (научной, образовательной, технологической, художественной) постоянно создает и использует модели окружающего мира. Строгие правила построения моделей сформулировать невозможно, однако человечество накопило богатый опыт моделирования различных объектов и процессов.

Модели позволяют представить *в наглядной форме* объекты и процессы, недоступные для непосредственного восприятия (очень большие или очень маленькие объекты, очень быстрые или очень медленные процессы и др.). Наглядные модели часто используются в процессе обучения. В курсе географии первые представления о нашей планете Земля мы получаем, изучая ее модель — глобус, в курсе физики изучаем работу двигателя внутреннего сгорания по его модели, в химии при изучении строения вещества используем модели молекул и кристаллических решеток, в биологии изучаем строение человека по анатомическим муляжам и др.

Модели играют чрезвычайно важную роль *в проектировании* и создании различных технических устройств, машин и механизмов, зданий, электрических цепей и т. д. Без предварительного создания чертежа (рис. 5.1) невозможно изготовить даже простую деталь, не говоря уже о сложном механизме.

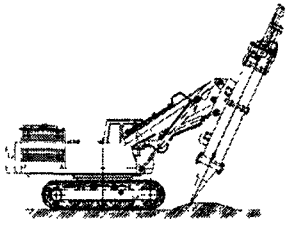


Рис. 5.1. Чертеж

В процессе проектирования зданий и сооружений кроме чертежей часто изготавливают макеты. В процессе разработки летательных аппаратов поведение их моделей в воздушных потоках исследуют в аэродинамической трубе.

Разработка электрической схемы обязательно предшествует созданию электрических цепей и так далее.

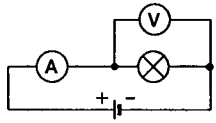


Рис. 5.2. Электрическая схема

Развитие науки невозможно без создания *теоретических моделей* (теорий, законов, гипотез и пр.), отражающих строение, свойства и поведение реальных объектов. Создание новых теоретических моделей иногда коренным образом меняет представление человечества об окружающем мире (гелиоцентрическая система мира Коперника, модель атома Резерфорда–Бора, модель расширяющейся Вселенной, модель генома человека и пр.). Адекватность теоретических моделей законам реального мира проверяется с помощью опытов и экспериментов.

Все *художественное творчество* фактически является процессом создания моделей. Например, такой литературный жанр, как басня, переносит реальные отношения между людьми на отношения между животными и фактически создает модели человеческих отношений. Более того, практически любое литературное произведение может рассматриваться как модель реальной человеческой жизни. Моделями, в художественной форме отражающими реальную действительность, являются также живописные полотна, скульптуры, тетральные постановки и пр.

Моделирование — это метод познания, состоящий в создании и исследовании моделей.

Модель. Каждый объект имеет большое количество различных свойств. В процессе построения модели выделяются главные, наиболее существенные для проводимого исследования свойства. В процессе исследования аэродинамических качеств модели самолета в аэродинамической трубе важно, чтобы модель имела геометрическое подобие оригинала, но не важен, например, ее цвет. При построении электрических схем — моделей электрических цепей — необходимо учитывать порядок подключения элементов цепи друг к другу, но не важно их геометрическое расположение друг относительно друга и так далее.

Разные науки исследуют объекты и процессы под разными углами зрения и строят различные типы моделей. В фи-

зике изучаются процессы взаимодействия и изменения объектов, в химии — их химический состав, в биологии — строение и поведение живых организмов и так далее.

Возьмем в качестве примера человека: в разных науках он исследуется в рамках различных моделей. В рамках механики его можно рассматривать как материальную точку, в химии — как объект, состоящий из различных химических веществ, в биологии — как систему, стремящуюся к самосохранению, и так далее.

Модель — это некий новый объект, который отражает существенные особенности изучаемого объекта, явления или процесса.

География, военное дело, судоходство и пр. невозможны без информационных моделей поверхности Земли в виде карт (рис. 5.3). Различные типы географических карт (политические, физические и пр.) представляют информационные модели, отражающие различные особенности земной поверхности, то есть один и тот же объект отражают несколько моделей.

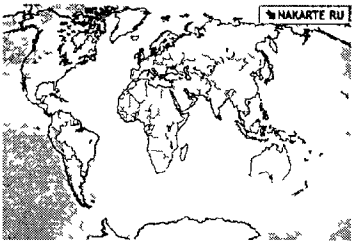


Рис. 5.3. Географическая карта

С другой стороны, разные объекты могут описываться одной моделью. Так, в механике различные материальные тела (от планеты до песчинки) могут рассматриваться как материальные точки.

Один и тот же объект может иметь **множество моделей**, а разные объекты могут описываться **одной моделью**.

Никакая модель не может заменить сам объект. Но при решении конкретной задачи, когда нас интересуют определенные свойства изучаемого объекта, модель оказывается полезным, а подчас и единственным инструментом исследования.

Вопросы для размышления

1. Может ли объект иметь несколько моделей? Приведите пример.
2. Могут ли разные объекты описываться одной и той же моделью? Если да, приведите пример.

5.2. Формы представления моделей. Формализация

Модели материальные и модели информационные. Все модели можно разбить на два больших класса: модели *предметные* (*материальные*) и модели *информационные*. Предметные модели воспроизводят геометрические, физические и другие свойства объектов в материальной форме (глобус, анатомические муляжи, модели кристаллических решеток, макеты зданий и сооружений и др.).

Информационные модели представляют объекты и процессы в *образной* или *знаковой форме*.

Образные модели (рисунки, фотографии и др.) представляют собой зрительные образы объектов, зафиксированные на каком-либо носителе информации (бумаге, фото- и кинопленке и др.). Широко используются образные информационные модели в образовании (вспомните учебные плакаты по различным предметам) и науках, где требуется классификация объектов по их внешним признакам (в ботанике, биологии, палеонтологии и др.).

Знаковые информационные модели строятся с использованием различных языков (знаковых систем). Знаковая информационная модель может быть представлена в форме текста (например, программы на языке программирования), формулы (например, второго закона Ньютона $F = m \cdot a$), таблицы (например, периодической таблицы элементов Д. И. Менделеева) и так далее.

Иногда при построении знаковых информационных моделей используются одновременно несколько различных языков. Примерами таких моделей могут служить географические карты, графики, диаграммы и пр. Во всех этих моделях используются одновременно как язык графических элементов, так и символьный язык.

На протяжении своей истории человечество использовало различные способы и инструменты для создания информационных моделей. Эти способы постоянно совершенствовались. Так, первые информационные модели создавались в форме наскальных рисунков, в настоящее же время информационные модели обычно строятся и исследуются с использованием современных компьютерных технологий.

Формализация. Естественные языки используются для создания *описательных информационных моделей*. В истории науки известны многочисленные описательные информационные модели; например, гелиоцентрическая модель мира, которую предложил Коперник, формулировалась следующим образом:

- Земля вращается вокруг своей оси и вокруг Солнца;
- орбиты всех планет проходят вокруг Солнца.

С помощью формальных языков строятся *формальные информационные модели* (математические, логические и др.). Одним из наиболее широко используемых формальных языков является математика. Модели, построенные с использованием математических понятий и формул, называются *математическими моделями*. Язык математики является совокупностью формальных языков. С некоторыми из них (алгебра, геометрия, тригонометрия) вы знакомитесь в школе, с другими (теория множеств, теория вероятностей и др.) сможете ознакомиться в процессе дальнейшего обучения.

Язык алгебры позволяет формализовать функциональные зависимости между величинами. Так, Ньютон формализовал гелиоцентрическую систему мира, открыв законы механики и закон всемирного тяготения и записав их в виде алгебраических функциональных зависимостей. В школьном курсе физики рассматривается много разнообразных функциональных зависимостей, выраженных на языке алгебры, которые представляют собой математические модели изучаемых явлений или процессов.

Язык алгебры логики (алгебры высказываний) позволяет строить *формальные логические модели*. С помощью алгебры высказываний можно формализовать (записать в виде логических выражений) простые и сложные высказывания, выраженные на естественном языке. Построение логических моделей позволяет решать логические задачи, строить логические модели устройств компьютера (сумматора, триггера) и так далее.

Процесс построения информационных моделей с помощью формальных языков называется **формализацией**.

В процессе познания окружающего мира человечество постоянно использует моделирование и формализацию. При изучении нового объекта сначала обычно строится его описательная информационная модель на естественном языке, затем она формализуется, то есть выражается с использованием формальных языков (математики, логики и др.).

Визуализация формальных моделей. В процессе исследования формальных моделей часто производится их визуализация. Для визуализации алгоритмов используются блок-схемы: пространственных соотношений между объектами — чертежи, моделей электрических цепей — электрические схемы, логических моделей устройств — логические схемы и так далее.

Так при визуализации формальных физических моделей с помощью анимации может отображаться динамика процесса, производиться построение графиков изменения физических величин и так далее. Визуальные модели обычно являются интерактивными, то есть исследователь может менять начальные условия и параметры протекания процессов и наблюдать изменения в поведении модели.

http //www college ru/physics/applets/11a htm **Интернет**

В качестве примера можно рассмотреть модель, которая демонстрирует свободные колебания математического маятника. С помощью анимации показываются движение тела и действующие силы, строятся графики зависимости от времени угловой координаты или скорости, диаграммы потенциальной и кинетической энергий (рис. 5.4). Исследователь может изменять длину нити l, угол начального отклонения маятника φ_0 и коэффициент вязкого трения b.

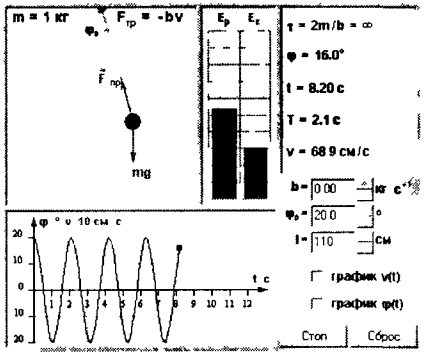


Рис. 5.4. Модель математического маятника

Вопросы для размышления

1. Какие бывают модели? Приведите примеры материальных и информационных моделей.
2. Что такое формализация? Приведите примеры формальных моделей.

Практические задания

5.1. Найти в Интернете и ознакомиться с визуализированными формальными моделями из различных предметных областей.

5.3. Системный подход в моделировании

Понятие о системе. Окружающий нас мир состоит из множества различных объектов, каждый из которых имеет разнообразные свойства, и при этом объекты взаимодействуют между собой. Например, такие объекты, как планеты нашей Солнечной системы, имеют различные свойства (массу, геометрические размеры и пр.) и по закону всемирного тяготения взаимодействуют с Солнцем и друг с другом.

Планеты входят в состав более крупного объекта — Солнечной системы, а Солнечная система — в состав нашей галактики «Млечный путь». С другой стороны, планеты состоят из атомов различных химических элементов, а атомы — из элементарных частиц. Можно сделать вывод, что практически каждый объект состоит из других объектов, то есть представляет собой *систему*.

Важным признаком системы является ее *целостное функционирование*. Система является не набором отдельных элементов, а совокупностью взаимосвязанных элементов. Например, компьютер является системой, состоящей из различных устройств, при этом устройства связаны между собой и аппаратно (физически подключены друг к другу) и функционально (между устройствами происходит обмен информацией).

Система является совокупностью взаимосвязанных объектов, которые называются элементами системы.

Состояние системы характеризуется ее *структурой*, то есть составом и свойствами элементов, их отношениями и связями между собой. Система сохраняет свою целостность под воздействием различных внешних воздействий и внутренних изменений до тех пор, пока она сохраняет неизменной свою структуру. Если структура системы меняется (например, удаляется один из элементов), то система может перестать функционировать как целое. Так, если удалить одно из устройств компьютера (например, процессор), компьютер выйдет из строя, то есть прекратит свое существование как система.

Статические информационные модели. Любая система существует в пространстве и во времени. В каждый момент времени система находится в определенном состоянии, которое характеризуется составом элементов, значениями их свойств, величиной и характером взаимодействия между элементами и так далее.

Так, состояние Солнечной системы в любой момент времени характеризуется составом входящих в нее объектов (Солнце, планеты и др.), их свойствами (размерами, положением в пространстве и др.), величиной и характером взаимодействия между собой (силами тяготения, с помощью электромагнитных волн и др.).

Модели, описывающие состояние системы в определенный момент времени, называются *статическими информационными моделями*.

В физике примером статических информационных моделей являются модели, описывающие простые механизмы, в биологии — модели строения растений и животных, в химии — модели строения молекул и кристаллических решеток и так далее.

Динамические информационные модели. Состояние систем изменяется во времени, то есть происходят процессы *изменения и развития систем*. Так, планеты движутся, изменяется их положение относительно Солнца и друг друга; Солнце, как и любая другая звезда, развивается, меняются ее химический состав, излучение и так далее.

Модели, описывающие процессы изменения и развития систем, называются *динамическими информационными моделями*.

В физике динамические информационные модели описывают движение тел, в биологии — развитие организмов или популяций животных, в химии — процессы прохождения химических реакций и так далее.

Вопросы для размышления

1. Образуют ли систему комплектующие компьютера: До сборки? После сборки? После включения компьютера?
2. В чем разница между статическими и динамическими информационными моделями? Приведите примеры статических и динамических информационных моделей.

5.4. Типы информационных моделей

Информационные модели отражают различные типы систем объектов, в которых реализуются различные структуры взаимодействия и взаимосвязи между элементами системы. Для отражения систем с различными структурами используются различные типы информационных моделей: табличные, иерархические и сетевые.

5.4.1. Табличные информационные модели

Одним из наиболее часто используемых типов информационных моделей является прямоугольная *таблица*, которая состоит из столбцов и строк. Такой тип моделей применяется для описания ряда объектов, обладающих одинаковыми наборами свойств. С помощью таблиц могут быть построены как статические, так и динамические информационные модели в различных предметных областях. Широко известно табличное представление математических функций, статистических данных, расписаний поездов и самолетов, уроков и так далее.

В *табличной информационной модели* обычно перечень объектов размещен в ячейках первого столбца таблицы, а значения их свойств — в других столбцах. Иногда используется другой вариант размещения данных в табличной модели, когда перечень объектов размещается в первой строке таблицы, а значения их свойств — в последующих строках. Подобным образом организованы таблицы истинности логи-

ческих функций, рассмотренные в главе 3. Перечень логических переменных и функций размещен в первой строке таблицы, а их значения — в последующих строках.

В **табличной информационной модели** перечень однотипных объектов или свойств размещен в первом столбце (или строке) таблицы, а значения их свойств размещаются в следующих столбцах (или строках) таблицы.

Построим табличную информационную модель «Цены устройств компьютера». В первом столбце таблицы будет содержаться перечень однотипных объектов (устройств, входящих в состав компьютера), а во втором — интересующее нас свойство (например, цена) — табл. 5.1. Построенная табличная модель позволяет оценить долю стоимости отдельных устройств в цене компьютера и приобрести за минимальную цену компьютер в наиболее производительной конфигурации.

Таблица 5.1. Цены устройств компьютера на конец 2001 г.

Наименование устройства	Цена (в у.е.)
Системная плата	80
Процессор Celeron (1 ГГц)	70
Память DIMM 128 Мб	15
Жесткий диск 40 Гб	130
Дисковод 3,5"	14
Видеоплата 16 Мб	30
Монитор 15"	180
Звуковая карта 16 битов	30
Дисковод CD-ROM x52	40
Корпус	25
Клавиатура	10
Мышь	5

Табличные информационные модели проще всего строить и исследовать на компьютере с помощью электронных таблиц и систем управления базами данных. Визуализируем полученную табличную модель путем построения диаграммы в электронных таблицах.

10.5. Построение диаграмм и графиков

Визуализация табличной модели

1. Ввести наименования устройств и их цены в столбцы электронной таблицы.
   Отсортировать данные по столбцу *Цена* в порядке убывания.
   Построить круговую диаграмму.

Анализ модели показывает, что увеличение расходов на приобретение более быстрого процессора и увеличение объема оперативной памяти не приведут к заметному увеличению цены компьютера, но позволят существенно повысить его производительность.

Модель хранится в файле Model xls ***в каталоге*** \textbook\Excel\ CD-ROM

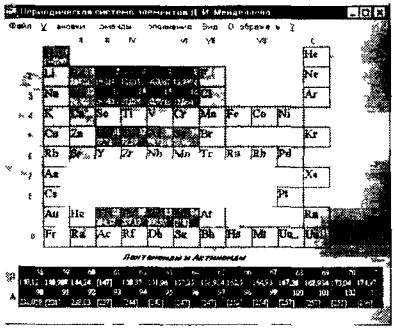
Представление объектов и их свойств в форме таблицы часто используется в научных исследованиях. Так, на развитие химии и физики решающее влияние оказало создание Д. И. Менделеевым в конце XIX века периодической системы элементов, которая представляет собой табличную информационную модель. В этой модели химические элементы располагаются в ячейках таблицы по возрастанию атомных весов, а в столбцах — по количеству валентных электронов, причем по положению в таблице можно определить некоторые физические и химические свойства элементов.

На уроках химии часто используется печатный вариант периодической системы элементов. Компьютерная модель системы более удобна, так как в интерактивном режиме позволяет знакомиться с различными физическими и химическими свойствами химических элементов (атомная масса, электропроводность, плотность и так далее), уравнивать химические реакции, решать стандартные химические задачи на нахождение массы веществ, участвующих в реакции, и др.

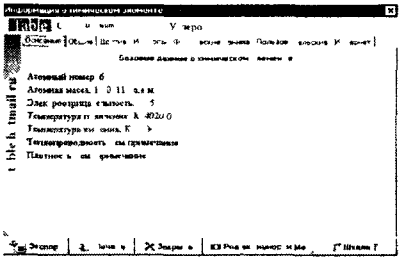
Установить приложение Table, ***которое хранится в каталоге*** \soft\model\chemistry\table\ CD-ROM

Периодическая таблица элементов Д. И. Менделеева

1. Запустить программу Table. С помощью меню *Отображать* выбрать физическое или химическое свойство элементов, значение которого будет выводиться в ячейки таблицы.



2. Для ознакомления с физическими и химическими свойствами элемента осуществить двойной щелчок на ячейке с названием элемента. На появившейся информационной панели элемента выбрать одну из вкладок со свойствами.



Вопросы для размышления

1. Какие системы объектов целесообразно и возможно представлять с помощью табличных моделей?

Практические задания

5.2. Построить и исследовать табличную модель, содержащую цены на компьютерные комплектующие на текущий момент.

5.3. Ознакомиться с физическими и химическими свойствами элементов с использованием компьютерной модели периодической таблицы элементов Д. И. Менделеева.

5.4.2. Иерархические информационные модели

Нас окружает множество различных объектов, каждый из которых обладает определенными свойствами. Однако некоторые группы объектов имеют одинаковые общие свойства, которые отличают их от объектов других групп.

Группа объектов, обладающих одинаковыми общими свойствами, называется *классом объектов*. Внутри класса объектов могут быть выделены подклассы, объекты которых обладают некоторыми особенными свойствами, в свою очередь подклассы могут делиться на еще более мелкие группы и так далее. Такой процесс систематизации объектов называется *процессом классификации*.

В процессе классификации объектов часто строятся информационные модели, которые имеют *иерархическую структуру*. В биологии весь животный мир рассматривается как иерархическая система (тип, класс, отряд, семейство, род, вид), в информатике используется иерархическая файловая система и так далее.

Статическая иерархическая модель. Рассмотрим процесс построения информационной модели, которая позволяет классифицировать современные компьютеры. Класс *Компьютеры* можно разделить на три подкласса: *Суперкомпьютеры*, *Серверы* и *Персональные компьютеры*.

Компьютеры, входящие в подкласс *Суперкомпьютеры*, отличаются сверхвысокой производительностью и надежностью и используются в крупных научно-технических центрах для управления процессами в реальном масштабе времени.

Компьютеры, входящие в подкласс *Серверы*, обладают высокой производительностью и надежностью и используются в качестве серверов в локальных и глобальных сетях.

Компьютеры, входящие в подкласс *Персональные компьютеры*, обладают средней производительностью и надежностью и используются в офисах и дома для работы с различными приложениями.

Подкласс *Персональные компьютеры* делится, в свою очередь, на *Настольные*, *Портативные* и *Карманные* компьютеры.

В иерархической структуре элементы распределяются по уровням, от первого (верхнего) уровня до нижнего (последнего) уровня. На первом уровне может располагаться только один элемент, который является «вершиной» иерархической структуры. Основное отношение между уровнями состоит в том, что элемент более высокого уровня может состоять из

нескольких элементов нижнего уровня, при этом каждый элемент нижнего уровня может входить в состав только одного элемента верхнего уровня.

В **иерархической информационной модели** объекты распределены по уровням. Каждый элемент более высокого уровня может состоять из элементов нижнего уровня, а элемент нижнего уровня может входить в состав только одного элемента более высокого уровня.

В рассмотренной иерархической модели, классифицирующей компьютеры, имеются три уровня. На первом, верхнем, уровне располагается элемент *Компьютеры*, в него входят три элемента второго уровня *Суперкомпьютеры*, *Серверы* и *Персональные компьютеры*. В состав последнего входят три элемента третьего, нижнего, уровня *Настольные*, *Портативные* и *Карманные* компьютеры.

Изображение информационной модели в форме графа. *Граф* является удобным способом наглядного представления структуры информационных моделей. *Вершины графа* (овалы) отображают элементы системы.

Элементы верхнего уровня находятся в отношении «состоять из» к элементам более низкого уровня. Такая связь между элементами отображается в форме *дуги графа* (направленной линии в форме стрелки). Графы, в которых связи между объектами несимметричны (как в данном случае), называются *ориентированными*.

Построим теперь компьютерную модель *Компьютеры* с использованием приложения Иерархика, которое позволяет создавать иерархические модели.

Изобразим иерархическую модель, классифицирующую компьютеры, в виде графа (рис. 5.5).

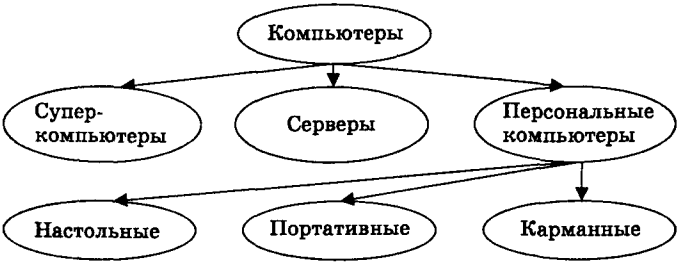


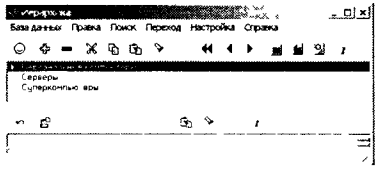
Рис. 5.5. Классификация компьютеров

Полученный граф напоминает дерево, которое растет сверху вниз, поэтому иерархические графы иногда называют *деревьями*.

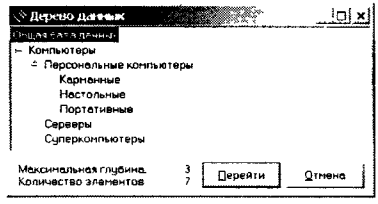
Установить приложение Иерархика, ***которое хранится в каталоге*** \soft\model\hierarchmod\hierarch\ **CD-ROM**

Иерархическая модель *Компьютеры*

1. Запустить программу Иерархика. С помощью *панели инструментов* вставить элементы трехуровневой иерархической модели.



2. Для просмотра модели ввести команду [База данных-Дерево данных...]. Появится окно *Дерево данных*, содержащее иерархическую модель.



Динамическая иерархическая модель. Для описания исторического процесса смены поколений семьи используются динамические информационные модели в форме генеалогического дерева. В качестве примера можно рассмотреть фрагмент (X–XI века) генеалогического дерева династии Рюриковичей (рис. 5.6).

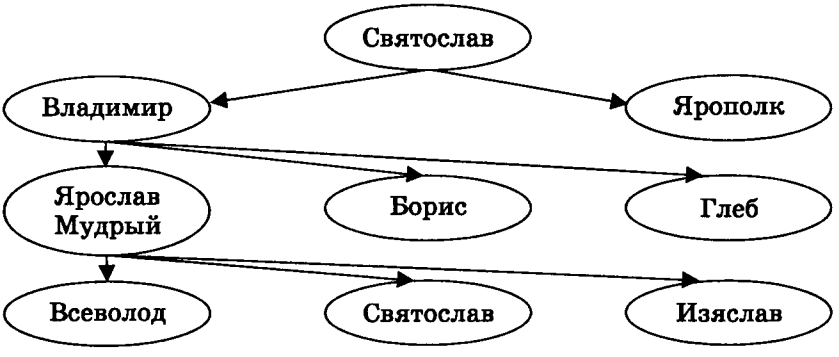


Рис. 5.6. Генеалогическое дерево Рюриковичей (X–XI века)

Вопросы для размышления

1. Какие системы объектов целесообразно и возможно представлять с помощью иерархических моделей?

Практические задания

5.4. Построить компьютерную модель фрагмента иерархической системы животного мира.

5.5. Построить компьютерную модель генеалогического дерева династии Романовых.

5.6. Построить компьютерную модель генеалогического дерева вашей семьи.

5.4.3. Сетевые информационные модели

Сетевые информационные модели применяются для отражения систем со сложной структурой, в которых связи между элементами имеют произвольный характер.

Например, различные региональные части глобальной компьютерной сети Интернет (американская, европейская, российская, австралийская и так далее) связаны между собой высокоскоростными линиями связи. При этом одни части (например, американская) имеют прямые связи со всеми региональными частями Интернета, а другие могут обмениваться информацией между собой только через американскую часть (например, российская и австралийская).

Построим граф, который отражает структуру глобальной сети Интернет (рис. 5.7). Вершинами графа являются региональные сети. Связи между вершинами носят двусторонний характер и поэтому изображаются ненаправленными линиями (*ребрами*), а сам граф поэтому называется *неориентированным*.

Представленная сетевая информационная модель является статической моделью. С помощью сетевой динамической модели можно, например, описать процесс передачи мяча между игроками в коллективной игре (футболе, баскетболе и так далее).

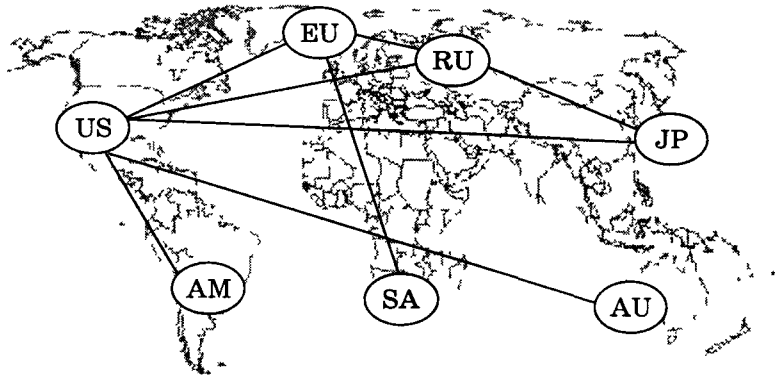


Рис. 5.7. Сетевая структура глобальной сети Интернет

Вопросы для размышления

1. Какие системы объектов целесообразно и возможно представлять с помощью сетевых моделей?

Задания

5.7. Построить информационную модель локальной сети школьного компьютерного класса.

5.5. Основные этапы разработки и исследования моделей на компьютере

Использование компьютера для исследования информационных моделей различных объектов и систем позволяет изучить их изменения в зависимости от значения тех или иных параметров. Процесс разработки моделей и их исследования на компьютере можно разделить на несколько основных этапов.

На первом этапе исследования объекта или процесса обычно строится *описательная информационная модель*. Такая модель выделяет существенные с точки зрения целей

проводимого исследования параметры объекта, а несущественными параметрами пренебрегает.

На втором этапе создается *формализованная модель*, то есть описательная информационная модель записывается с помощью какого-либо формального языка. В такой модели с помощью формул, уравнений, неравенств и пр. фиксируются формальные соотношения между начальными и конечными значениями свойств объектов, а также накладываются ограничения на допустимые значения этих свойств.

Однако далеко не всегда удается найти формулы, явно выражающие искомые величины через исходные данные. В таких случаях используются приближенные математические методы, позволяющие получать результаты с заданной точностью.

На третьем этапе необходимо формализованную информационную модель преобразовать в *компьютерную модель*, то есть выразить ее на понятном для компьютера языке. Существуют два принципиально различных пути построения компьютерной модели:

1) построение алгоритма решения задачи и его кодирование на одном из языков программирования;

2) построение компьютерной модели с использованием одного из приложений (электронных таблиц, СУБД и пр.).

В процессе создания компьютерной модели полезно разработать удобный графический интерфейс, который позволит визуализировать формальную модель, а также реализовать интерактивный диалог человека с компьютером на этапе исследования модели.

Четвертый этап исследования информационной модели состоит в проведении *компьютерного эксперимента*. Если компьютерная модель существует в виде программы на одном из языков программирования, ее нужно запустить на выполнение и получить результаты.

Если компьютерная модель исследуется в приложении, например в электронных таблицах, можно провести сортировку или поиск данных, построить диаграмму или график и так далее.

Пятый этап состоит в *анализе полученных результатов и корректировке исследуемой модели*. В случае различия результатов, полученных при исследовании информационной модели, с измеряемыми параметрами реальных объектов можно сделать вывод, что на предыдущих этапах построения модели были допущены ошибки или неточности. Например, при построении описательной качественной модели

могут быть неправильно отобраны существенные свойства объектов, в процессе формализации могут быть допущены ошибки в формулах и так далее. В этих случаях необходимо провести корректировку модели, причем уточнение модели может проводиться многократно, пока анализ результатов не покажет их соответствие изучаемому объекту.

Вопросы для размышления

1. В каких случаях могут быть опущены отдельные этапы построения и исследования модели? Приведите примеры создания моделей в процессе обучения.

5.6. Исследование физических моделей

Физика-9 

Рассмотрим процесс построения и исследования модели на конкретном примере движения тела, брошенного под углом к горизонту.

Содержательная постановка задачи. В процессе тренировок теннисистов используются автоматы по бросанию мячика в определенное место площадки. Необходимо задать автомату необходимую скорость и угол бросания мячика для попадания в мишень определенного размера, находящуюся на известном расстоянии.

Качественная описательная модель. Сначала построим качественную описательную модель процесса движения тела с использованием физических объектов, понятий и законов, то есть в данном случае идеализированную модель движения объекта. Из условия задачи можно сформулировать следующие основные предположения:

- мячик мал по сравнению с Землей, поэтому его можно считать материальной точкой;
- изменение высоты мячика мало, поэтому ускорение свободного падения можно считать постоянной величиной $g = 9{,}8$ м/с^2 и движение по оси OY можно считать равноускоренным;
- скорость бросания тела мала, поэтому сопротивлением воздуха можно пренебречь и движение по оси OX можно считать равномерным.

Формальная модель. Для формализации модели используем известные из курса физики формулы равномерного и равноускоренного движения. При заданных начальной скорости v_0 и угле бросания α значения координат дальности полета x и высоты y от времени можно описать следующими формулами:

$$x = v_0 \cdot \cos\alpha \cdot t; \quad (5.1)$$
$$y = v_0 \cdot \sin\alpha \cdot t - g \cdot t^2/2.$$

Пусть мишень высотой h будет размещаться на расстоянии s от автомата. Из первой формулы выражаем время, которое понадобится мячику, чтобы преодолеть расстояние s:

$$t = s/(v_0 \cdot \cos\alpha).$$

Подставляем это значение для t в формулу для y. Получаем l — высоту мячика над землей на расстоянии s:

$$l = s \cdot \mathrm{tg}\alpha - g \cdot s^2/(2 \cdot v_0^2 \cdot \cos^2\alpha).$$

Формализуем теперь условие попадания мячика в мишень. Попадание произойдет, если значение высоты l мячика будет удовлетворять условию в форме неравенства:

$$0 \leqslant l \leqslant h.$$

Если $l < 0$, то это означает «недолет», а если $l > h$, то это означает «перелет».

Компьютерная модель на языке Visual Basic. Преобразуем формальную модель в компьютерную с использованием системы программирования Visual Basic. Создадим сначала графический интерфейс проекта.

Проект «Движение тела, брошенного под углом к горизонту»

1. Разместить на форме четыре текстовых поля для ввода значений начальной скорости, угла бросания мячика, расстояния до мишения и ее высоты, а также два текстовых поля для вывода высоты мячика на заданном расстоянии и текстового сообщения о результатах броска.
2. Поместить на форму метки для обозначения полей и единиц измерения.
3. Поместить на форму кнопку и создать для нее событийную процедуру, которая обеспечивает присваивание переменным значений, введенных в текстовые поля, вычисление высоты мячика на заданном расстоянии и вывод результатов на форму с использованием конструкции выбора `Select Case`:

```
Const G As Single = 9.81
Const Pi As Single = 3.14
Dim V0, A, S, L As Double
Private Sub CmdCalc_Click()
'Ввод начальных значений
V0 = Val(txtV0.Text)
A = Val(txtA.Text)
S = Val(txtS.Text)
H = Val(txtH.Text)
'Попадание в мишень
L = S * Tan(A * Pi / 180) - (G * S ^ 2) /
(2 * V0 ^ 2 * Cos(A * Pi / 180) ^ 2)
txtL.Text = L
Select Case L
Case Is < 0
txtM.Text = "Недолет"
Case Is > H
txtM.Text = "Перелет"
Case Else
txtM.Text = "Попадание"
End Select
End Sub
```

Для визуализации формальной модели построим траекторию движениия тела (график зависимости высоты мячика над поверхностью земли от дальности полета). Снабдим график осями координат и выведем положение мишени.

4. Поместить на форму графическое поле, в котором будет осуществляться построение графика, и дополнить программный код событийной процедуры:

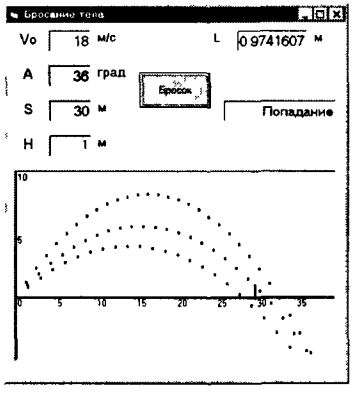
```
'Построение графика
For T = 0 To 10 Step 0.1
Y = V0 * Sin(A * Pi /180) * T - G * T *
T/2
X = V0 * Cos(A * Pi /180) * T
pic1.Scale (0, 15)-(S + 5, -5)
pic1.PSet (X, Y)
Next T
'Ось X
pic1.Line (0, 0)-(50, 0)
For I = 0 To 50 Step 5
pic1.PSet (I, 0)
pic1.Print I
Next I
'Ось Y
pic1.Line (0, -5)-(0, 15)
```

```
For I = -5 To 15 Step 5
pic1.PSet (0, I)
pic1.Print I
Next I
'Мишень
pic1.Line (S, 0)-(S, H)
```

Проект хранится в каталоге \textbook\VB\prjPhys1\ CD-ROM

Компьютерный эксперимент. Введем произвольные значения начальной скорости и угла бросания мячика: скорее всего, его попадания в мишень не будет. Затем, меняя один из параметров, например угол, произведем пристрелку. Для этого лучше всего использовать известный артиллерийский прием «взятие в вилку», который использует наиболее эффективный метод «деление пополам». Для этого находят угол, при котором мячик перелетит мишень, затем угол, при котором мячик не долетит до стены. Вычисляют среднее значение углов, составляющих «вилку», и смотрят, куда попадет мячик. Если он попадет в стену, то задача выполнена, если не попадет, то рассматривается новая «вилка», и так далее.

5. Запустить проект и ввести значения начальной скорости, угла, расстояния до мишени и ее высоты.
   Щелкнуть по кнопке *Бросок*. В текстовых полях будут выведены результаты, а в графическом поле появится траектория движения тела. Подобрать значения начальной скорости и угла бросания, обеспечивающие попадание в мишень.



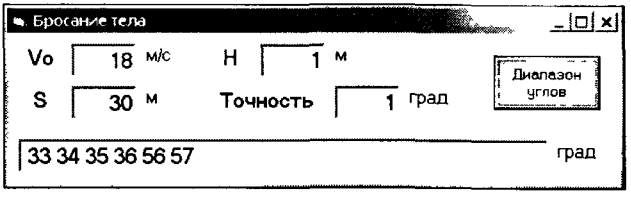
Анализ результатов и корректировка модели. Модернизируем проект так, чтобы можно было получить с заданной точностью для каждого значения скорости значение диапазона углов, обеспечивающее попадание мячика в мишень.

6. Удалить с формы текстовые поля для ввода значения угла, для вывода результатов и графическое поле и поместить на форму текстовые поля для ввода точности определения диапазона углов и для вывода значений из этого диапазона.

7. Внести изменения в программный код событийной процедуры:

```
Private Sub CmdCalc_Click()
'Ввод начальных значений
V0 = Val(txtV0.Text)
S = Val(txtS.Text)
H = Val(txtH.Text)
P = Val(txtP.Text)
txtA1.Text = ""
For A = 0 To 90 Step P
'Попадание в мишень
L = S * Tan(A * Pi / 180) - (G * S ^ 2) /
(2 * V0 ^ 2 * Cos(A * Pi / 180) ^ 2)
If 0 < L And L < H Then
txtA1.Text = txtA1.Text + Str(A)
End If
Next A
End Sub
```

8. Запустить проект и ввести скорость, расстояние до мишени и ее высоту, а также точность определения диапазона углов:



Анализ результатов показывает, что получен неочевидный результат: существуют два диапазона величин углов — от 33 до 36 и от 56 до 57 градусов, которые обеспечивают попадание мячика в мишень.

Проект хранится в каталоге \textbook\VB\prjPhys2\ **CD-ROM**

Компьютерная модель в электронных таблицах. Вернемся к третьему этапу создания и исследования модели движения тела, брошенного под углом к горизонту. Преобразуем теперь формальную модель в компьютерную с использованием электронных таблиц Excel.

Выделим в таблице определенные ячейки для ввода значений начальной скорости v_0 и угла α и вычислим по форму-

лам (5.1) значения координат тела x и y для определенных значений времени t с заданным интервалом.

Для преобразования значений углов из градусов в радианы используем функцию *РАДИАНЫ( )*.

Модель «Движение тела, брошенного под углом к горизонту» в электронных таблицах

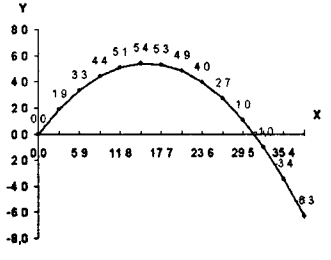
1. Для ввода начальной скорости будем использовать ячейку B1, а для ввода угла — ячейку B2.
2. Введем в ячейки A5:A18 значения времени с интервалом в 0,2 с.

	A	B	C
1	V_0 =	18,0 м/с	
2	α =	35 0 град	
3			
4	t	$x = v_0 \cos\alpha\ t$	$y = v_0 \sin\alpha\ t - g \cdot t^2/2$
5	0,0	0,0	0,0
6	0,2	2,9	1,9
7	0,4	5 9	3,3
8	0 6	8 8	4 4
9	0,8	11 8	5,1
10	1,0	14,7	5,4
11	1,2	17 7	5 3
12	1,4	20,6	4,8
13	1,6	23,6	4,0
14	1,8	26,5	2,7
15	2,0	29,5	1,0
16	2,2	32,4	1,0
17	2 4	35 4	3 5
18	2,6	38,3	-6,3

3. В ячейки B5 и C5 введем формулы:
   =$B$1*COS(РАДИАНЫ($B$2))*A5
   =$B$1*SIN(РАДИАНЫ($B$2))*A5–4,9*A5*A5
4. Скопируем формулы в ячейки B6:B18 и C6:C18 соответственно.

Визуализируем модель, построив график зависимости координаты y от координаты x (траекторию движения тела).

5. Построить диаграмму типа *График*, в которой используется в качестве категории диапазон ячеек B5:B18, а в качестве значений — диапазон ячеек C5:C18.



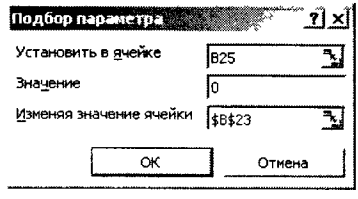
Исследование модели. Исследуем модель и определим с заданной точностью 0,1° диапазон изменений угла, который обеспечивает попадание в мишень, находящуюся на расстоянии 30 м и имеющую высоту 1 м, при заданной начальной скорости 18 м/с. Воспользуемся для этого методом *Подбор параметра*.

6. Установить для ячеек точность один знак после запятой.
7. Ввести в ячейки B21, B22 и B23 значения расстояния до мишени S = 30 м, начальной скорости v_0 = 18 м/с и угла α = 35°, а в ячейку B25 — формулу для вычисления высоты мячика над поверхностью для заданных начальных условий: =B21*TAN(РАДИАНЫ(B23))-(9,81*B21^2)/(2*B22^2*COS(РАДИАНЫ(B23))^2).

21	S =	30,0 м
22	V_0 =	18,0 м/с
23	α =	35,0 град
24		
25	L =	0,7 м

Для заданных начальных условий определим углы, которые обеспечивают попадание в мишень на высотах 0 и 1 м.

8. Выделить ячейку B25 и ввести команду [Сервис-Подбор параметра...].
На появившейся диалоговой панели ввести в поле *Значение:* наименьшую высоту попадания в мишень (то есть 0).
В поле *Изменяя значение ячейки:* ввести адрес ячейки, содержащей значение угла (в данном случае $B$23).



9. В ячейке B23 появится значение 32,6. Повторить процедуру подбора параметра для максимальной высоты попадания в мишень — в ячейке B23 получим значение 36,1.

Таким образом, исследование компьютерной модели в электронных таблицах показало, что существует диапазон значений угла бросания от 32,6 до 36,1°, который обеспечивает попадание в мишень высотой 1 м, находящуюся на расстоянии 30 м, мячиком, брошенным со скоростью 18 м/с.

Если повторить процедуру определения диапазона углов при начальном значении 55°, то получим значения предельных углов 55,8 и 57,4°. С учетом точности вычислений данные для обоих диапазонов углов подтверждают результаты, полученные при исследовании компьютерной модели на языке Visual Basic.

Модель хранится в файле Model.xls
в каталоге \textbook\Excel\ **CD-ROM**

Практические задания

5.8. Тело брошено вертикально вверх с некоторой высоты. Определить, через какое количество времени тело упадет на поверхность земли.

5.7. Исследование математических моделей

Алгебра-9, Геометрия-7

Исследование математических моделей начинается с записи формальной модели на языке определенной области математики: алгебры, геометрии и так далее.

5.7.1. Приближенное решение уравнений

На языке алгебры формальные модели записываются с помощью уравнений, точное решение которых основывается на поиске равносильных преобразований алгебраических выражений, позволяющих выразить переменную величину с помощью формулы. Точные решения существуют только для некоторых уравнений определенного вида (линейные, квадратные, тригонометрические и др.), поэтому для большинства уравнений приходится использовать методы приближенного решения с заданной точностью (графические, числовые и др.).

Графический метод. Построение графиков функций может использоваться для грубо приближенного решения уравнений. Для не имеющего точного алгебраического решения уравнения вида $f(x) = 0$, где $f(x)$ — некоторая непрерывная функция, корень (или корни) этого уравнения является точкой (или точками) пересечения графика функции с осью OX.

Задача. Найти графическим методом корень уравнения $x^3 - \cos x = 0$, которое не имеет точного алгебраического решения.

Формальная модель задана уравнением, для нахождения корня уравнения разработаем компьютерную модель на языке Visual Basic.

Проект «Приближенное решение уравнения»

1. В программный код проекта «Построение графика функции» в цикл построения графика ввести строку:

```
picGraph.PSet (sngX,
sngX ^ 3 - Cos(sngX))
```

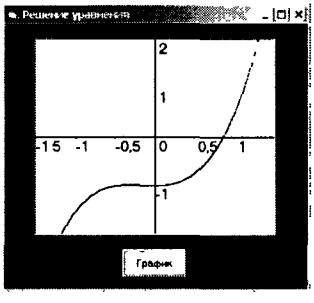


График функции пересекает ось OX один раз, и, следовательно, уравнение имеет один корень. По графику грубо приближенно можно определить, что $x \approx 0{,}8$.

Числовой метод половинного деления. Для решения уравнений с заданной точностью можно применить разработанные в вычислительной математике числовые итерационные методы решения уравнений. Если мы знаем отрезок, на котором существует корень, и функция на краях этого отрезка принимает значения разных знаков, то можно использовать метод половинного деления.

Идея метода состоит в выборе точности решения и сведении первоначального отрезка $[A;B]$, на котором существует корень уравнения, к отрезку заданной точности. Процесс сводится к последовательному делению отрезков пополам точкой $C = (A+B)/2$ и отбрасыванию той половины отрезка ($[A;C]$ или $[C;B]$), на котором корня нет.

Выбор нужной половины отрезка основывается на проверке знаков значений функции на его краях. Выбирается та половина, на которой произведение значений функции на краях отрицательно, то есть где функция пересекает ось абсцисс.

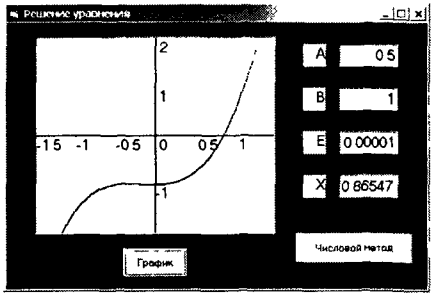
Процесс продолжается до тех пор, пока длина отрезка не станет меньше удвоенной точности. Деление этого отрезка пополам дает значение корня $x = (A+B)/2$ с заданной точностью.

2. Поместить на форму текстовые поля для ввода числовых значений концов отрезка A и B, поле для ввода точности вычислений и поле для вывода значений корня.
3. Поместить на форму кнопку и создать событийную процедуру, вычисляющую корень уравнения методом половинного деления с использованием цикла с постусловием:

```
Private Sub cmdNum_Click()
dblA = Val(txtA.Text)
dblB = Val(txtB.Text)
dblE = Val(txtE.Text)
Do
dblC = (dblA + dblB) / 2
If (dblA ^ 3 - Cos(dblA)) * (dblC ^ 3 - Cos(dblC))
< 0 Then
dblB = dblC
Else
dblA = dblC
End If
```

```
Loop While (dblB - dblA) / 2 > dblE
txtX.Text = (dblA + dblB) / 2
End Sub
```

4. Из графика функции видно, что корень находится на отрезке [0,5;1]. Введем в текстовые поля значения концов отрезка, а также точность вычислений (например, 0,00001).
В текстовое поле будет выведено значение корня с заданной точностью: $x = 0{,}86547$.



Проект хранится в каталоге \textbook\VB\prjMath1\ **CD-ROM**

Практические задания

5.9. Приближенно решить уравнение $x^3 - \cos x = 0$ с использованием компьютерной модели в электронных таблицах.

5.7.2. Вероятностные модели

Вероятностные модели базируются на использовании больших серий испытаний со случайными параметрами, причем точность полученных результатов зависит от количества проведенных опытов. Воспользуемся методом Монте-Карло для приближенного вычисления площадей геометрических фигур.

Качественная модель метода Монте-Карло. Сначала построим качественную вероятностную модель данного метода:

- поместим геометрическую фигуру полностью внутрь квадрата;
- будем случайным образом «бросать» точки в этот квадрат, то есть с помощью генератора случайных чисел задавать точкам координаты внутри квадрата;
- будем считать, что отношение числа точек, попавших внутрь фигуры, к общему числу точек в квадрате прибли-

зительно равно отношению площади фигуры к площади квадрата, причем это отношение тем точнее, чем больше количество точек.

Формальная модель. Построим формальную модель для вычисления площади круга радиуса R, центр которого совпадает с началом координат. Круг вписан в квадрат со стороной $2R$, площадь которого вычисляется как $4R^2$ (рис. 5.8).

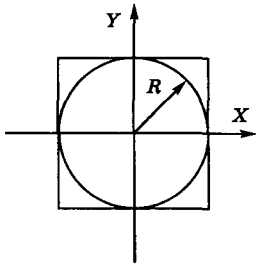


Рис. 5.8. Круг, вписанный в квадрат

Пусть N — количество точек, которые случайным образом генерируются внутри квадрата. Случайный выбор координат точек, которые попадают внутрь квадрата (N точек), должен производиться так, чтобы координаты точек x и y удовлетворяли условиям:

$$-R \leqslant x \leqslant R \quad \text{и} \quad -R \leqslant y \leqslant R.$$

Пусть M — количество точек, попавших внутрь круга, то есть их координаты удовлетворяют условию:

$$x^2 + y^2 \leqslant R^2.$$

Тогда площадь круга можно вычислить по формуле:

$$S = 4R^2 \cdot M/N.$$

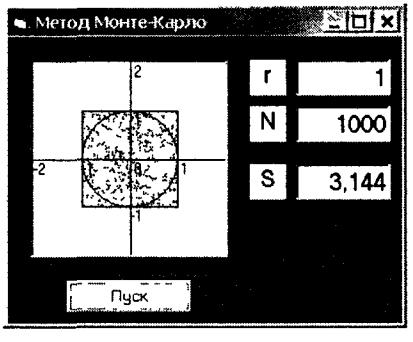
Компьютерная модель. Разработаем на языке Visual Basic компьютерную модель, позволяющую определять площадь круга методом Монте-Карло.

Проект «Метод Монте-Карло»

1. Поместить на форму графическое поле, в котором будет отображаться процесс случайной генерации точек, квадрат, круг и оси координат.
2. Поместить на форму два текстовых поля для ввода радиуса окружности и количества генерируемых точек и одно поле для вывода значения площади круга.
3. Поместить на форму кнопку и создать для нее событийную процедуру, которая обеспечивает ввод значений радиуса окружности в переменную `R`, ввод количества генерируемых точек в переменную `N`, генерацию случайных точек, подсчет в переменной `M` количества точек, попавших внутрь круга, вычисление и вывод значения площади круга в текстовое поле:

```
Dim dblX, dblY As Double, I, N, M, R As Long,
S As Double
Private Sub cmd1_Click()
M = 0
pic1.Cls
'Ввод значений
R = Val(txtR)
N = Val(txtN)
pic1.Scale (-(R + 1),R + 1)-(R + 1, -(R + 1))
pic1.Line (-R, R)-(R, -R), , B
pic1.Circle (0, 0), R
'Генерация точек
For I = 1 To N
dblX = 2 * R * Rnd - R
dblY = 2 * R * Rnd - R
pic1.PSet (dblX, dblY)
If dblX ^ 2 + dblY ^ 2 <= R Then M = M + 1
Next I
'Площадь
txtS.Text = 4 * R ^ 2 * (M / N)
'Ось X
pic1.Line (-(R + 1), 0)-(R + 1, 0)
For I = -(R + 1) To R + 1
pic1.PSet (I, 0)
pic1.Print I
Next I
'Ось Y
pic1.Line (0, -(R + 1))-(0, R + 1)
For I = -(R + 1) To R + 1
pic1.PSet (0, I)
pic1.Print I
Next I
End Sub
```

4. Ввести радиус окружности и количество генерируемых точек. После щелчка по кнопке *Пуск* в графическом поле будет отображен процесс генерации случайных точек, а в текстовое поле будет выведено значение площади круга.



Исследование модели. Существует геометрическая формула, позволяющая вычислить площадь круга: $S = \pi R^2$. Если в процессе исследования модели в качестве радиуса окружности выбрать 1, то числовое значение площади круга будет соответствовать числу π. Таким образом, с помощью метода Монте-Карло можно определить с необходимой точностью значение числа π (при увеличении количества генерируемых точек можно наблюдать все большее приближение значения площади к значению числа π).

Проект хранится в каталоге \textbook\VB\prjMath2\ CD-ROM

Практические задания

5.10. Определить методом Монте-Карло площадь треугольника, вершины которого имеют координаты (–1, 0); (0, 1) и (1, 0).

5.8. Биологические модели развития популяций

Общая биология 10–11

В биологии при исследовании развития биосистем строятся динамические модели изменения численности популяций различных живых существ (бактерий, рыб, животных и пр.) с учетом различных факторов. Взаимовлияние популяций рассматривается в моделях типа «хищник–жертва».

Формальная модель. Изучение динамики численности популяций естественно начать с простейшей модели *неограниченного роста*, в которой численность популяции ежегодно увеличивается на определенный процент. Математическую модель можно записать с помощью рекуррентной формулы, связывающей численность популяции следующего года с численностью популяции текущего года, с использованием коэффициента роста a:

$$x_{n+1} = a \cdot x_n.$$

Например, если ежегодный прирост численности популяции составляет 5%, то $a = 1{,}05$.

В модели *ограниченного роста* учитывается эффект перенаселенности, связанный с нехваткой питания, болезнями и так далее, который замедляет рост популяции с увеличени-

ем ее численности. Введем *коэффициент перенаселенности* b, значение которого обычно существенно меньше a ($b \ll a$). Тогда коэффициент ежегодного увеличения численности равен $(a - b \cdot x_n)$ и формула принимает вид:

$$x_{n+1} = (a - b \cdot x_n) \cdot x_n.$$

В модели *ограниченного роста с отловом* учитывается, что на численность популяций промысловых животных и рыб оказывает влияние величина ежегодного отлова. Если величина ежегодного отлова равна c, то формула принимает вид:

$$x_{n+1} = (a - b \cdot x_n) \cdot x_n - c.$$

Популяции обычно существуют не изолированно, а во взаимодействии с другими популяциями. Наиболее важным типом такого взаимодействия является взаимодействие между жертвами и хищниками (например, караси–щуки, зайцы–волки и так далее). В модели *«хищник–жертва»* количество жертв x_n и количество хищников y_n связаны между собой. Количество встреч жертв с хищниками можно считать пропорциональным произведению количеств жертв и хищников, а коэффициент f характеризует возможность гибели жертвы при встрече с хищниками. В этом случае численность популяции жертв ежегодно уменьшается на величину $f \cdot x_n \cdot y_n$ и формула для расчета численности жертв принимает вид:

$$x_{n+1} = (a - b \cdot x_n) \cdot x_n - c - f \cdot x_n \cdot y_n .$$

Численность популяции хищников в отсутствие жертв (в связи с отсутствием пищи) уменьшается, что можно описать рекуррентной формулой

$$y_{n+1} = d \cdot y_n,$$

где значение коэффициента $d < 1$ характеризует скорость уменьшения численности популяции хищников.

Увеличение популяции хищников можно считать пропорциональной произведению собственно количеств жертв и хищников, а коэффициент e характеризует величину роста численности хищников за счет жертв. Тогда для численности хищников можно использовать формулу:

$$y_{n+1} = d \cdot y_n + e \cdot x_n \cdot y_n.$$

Компьютерная модель. Построим в электронных таблицах компьютерную модель, позволяющую исследовать численность популяций с использованием различных моделей: неограниченного роста, ограниченного роста, ограниченного роста с отловом и «хищник–жертва».

Рост численности популяций

1. В ячейки B1 и B6 внести начальные значения численности популяций жертв и хищников.
   В ячейки B2:B5 внести значения коэффициентов *a*, *b*, *c* и *f*, влияющих на изменение численности жертв.
   В ячейки B7 и B8 внести значения коэффициентов *d* и *e*, влияющих на изменение численности хищников.

	A	B
1	X1=	1,50
2	a=	1,10
3	b=	0,03
4	c=	0,03
5	f=	0,04
6	Y1=	1,00
7	d=	0,90
8	e=	0,05

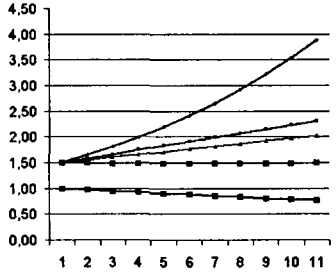
В столбце D будем вычислять численность популяции в соответствии с моделью неограниченного роста, в столбце E — ограниченного роста, в столбце F — ограниченного роста с отловом, в столбцах G и H —«хищник–жертва».

2. В ячейки D1, E1, F1 и G1 внести значения начальной численности популяций жертв, в ячейку H1 — хищников.
   В ячейку D2 внести рекуррентную формулу неограниченного роста =$B$2*D1.
   В ячейку E2 внести рекуррентную формулу ограниченного роста =($B$2–$B$3*E1)*E1.
   В ячейку F2 внести рекуррентную формулу ограниченного роста с отловом =($B$2–$B$3*F1)*F1–$B$4.
   В ячейку G2 внести рекуррентную формулу изменения количества жертв
   =($B$2–$B$3*G1)*G1–$B$4–$B$5*G1*H1.
   В ячейку H2 внести рекуррентную формулу изменения количества хищников =$B$7*H1+$B$8*G1*H1.
3. Скопировать внесенные формулы в ячейки столбцов командой [Правка-Заполнить-Вниз].
   В ячейках столбцов ознакомиться с динамикой изменения численности популяций.

D	E	F	G	H
1,50	1 50	1,50	1 50	1 00
1 65	1 58	1,55	1,49	0,98
1 82	1 67	1,61	1 49	0 95
2,00	1,75	1,66	1 48	0,93
2 20	1,83	1,71	1 48	0,90
2,42	1,91	1,77	1,48	0 88
2,66	2,00	1,82	1,48	0 86
2,92	2,08	1,87	1,48	0 83
3,22	2,15	1,92	1,48	0 81
3 54	2,23	1,97	1,49	0 79
3,89	2,30	2,02	1,49	0,77

Для визуализации компьютерной модели построим графики изменения популяций с течением времени.

4. Выделить столбцы данных и построить диаграмму типа *График*. Появятся графики изменения численности популяций в соответствии с моделями неограниченного роста, ограниченного роста, ограниченного роста с отловом, моделью хищник–жертва.



Исследование модели. Изменяя значения начальных численностей популяций, а также коэффициенты, можно получать различные варианты изменения численности популяций в зависимости от времени.

Модель хранится в файле Model.xls *в каталоге* \textbook\Excel\ CD-ROM

Практические задания

5.11. Исследовать модель развития популяций и определить, через сколько лет произойдет удвоение численности популяции в модели неограниченного роста.

5.12. В моделе ограниченного роста с отловом установить предельное значение величины отлова при заданных значениях коэффициентов a и b.

5.9. Геоинформационные модели

Геоинформационное моделирование базируется на создании многослойных элекронных карт, в которых опорный слой описывает географию определенной территории, а каждый из остальных — один из аспектов состояния этой территории. На географическую карту могут быть выведены различные слои объектов: города, дороги, аэропорты и др.

Широкое распространение получили интерактивные географические карты (мира, различных частей света, России, Москвы и других городов) в Интернете. Такие карты обычно реализуются с использованием векторной графики и поэтому дают возможность пользователю выбирать нужный ему масштаб. Карты связаны с базами данных, которые

хранят всю необходимую информацию об объектах, изображенных на картах.

http:// nakarte.ru Интернет

Например, на картографическом сервере можно сначала выбрать нужную карту из раскрывающегося списка, а затем выбрать вариант отображения карты (политико-административное деление, население, транспорт и так далее) — рис. 5.9. Если выбрать какой-либо регион на карте и щелкнуть на нем мышью, то появится окно с дополнительной информацией.

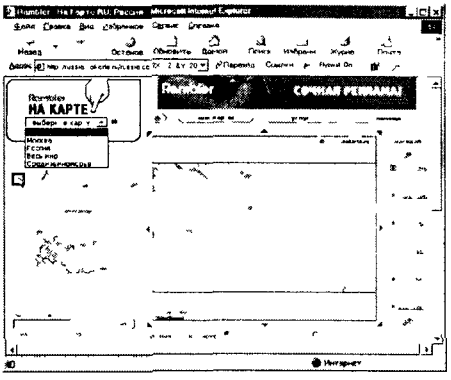


Рис. 5.9. Карта России (политико-административное деление)

http:// www moscowmap.ru Интернет

Пользователь может осуществлять поиск необходимого ему объекта на карте с помощью поисковой системы. Например, для того чтобы найти дом на интерактивной карте Москвы, требуется ввести название улицы и номер дома.

Так, после ввода в окно поиска адреса *Авиационный пер. д. 6* (рис. 5.10) на карте будет выделено более темным цветом здание Московского института открытого образования.

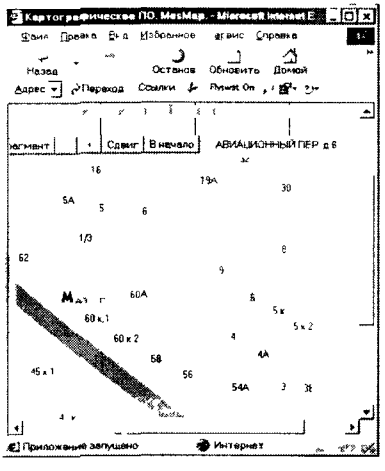


Рис. 5.10. Фрагмент карты Москвы

Геоинформационные модели позволяют с помощью географических карт представлять статистическую информацию о различных регионах. Хранящаяся в базах данных ин-

формация о количестве населения, развитии промышленности, загрязнении огружающей среды и др. может быть связана с географическими картами и отображена на них. Отображение информации может производиться различными способами: закрашиванием регионов различными цветами, построением диаграмм и так далее.

Построим в электронных таблицах Excel с использованием специальной надстройки Microsoft Data Map геоинформационную модель, отображающую информацию о количестве населения в различных странах мира.

Геоинформационная модель «Численность населения в странах мира»

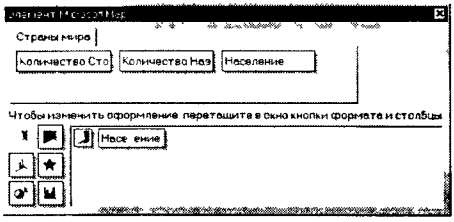
1. Запустить установку Microsoft Office и установить надстройку Microsoft Data Map.
   Открыть в электронных таблицах файл mapstats.xls, который записывается на диск в процессе установки Microsoft Office.
   На листах электронных таблиц содержатся базы данных по численности населения стран мира и Европы.

	A	B
1	[illegible]	Население
2	АВСТРАЛИЯ	17661468
3	АВСТРИЯ	7914127
4	АЗЕРБАЙДЖАН	7021178
5	АЗОРСКИЕ О-ВА (ПОРТ.)	236000
6	АЛБАНИЯ	1626315
7	АЛЖИР	22600957
8	АНГИЛЬЯ	9200
9	АНГОЛА	4830449
10	АНДОРРА	61599
11	АНТИГУА И БАРБУДА	64794
12	АРГЕНТИНА	32712930
13	АРМЕНИЯ	3611700
14	АРУБА (НИДЕР.)	66687
15	АФГАНИСТАН	15513267
16	БАГАМСКИЕ О-ВА	264175
17	БАНГЛАДЕШ	109291000
18	БАРБАДОС	255200
19	БАХРЕЙН	520653
20	БЕЛАРУСЬ	10222649
21	БЕЛИЗ	205000
22	БЕЛЬГИЯ	9967378

2. Для вызова карты щелкнуть по кнопке *Карта*. На появившейся диалоговой панели *Обнаружено несколько карт* выбрать требуемую карту, например *Страны мира*.
3. Для вывода на карту статистических данных ввести команду [Сервис-Данные]. Выделить на листе электронных таблиц столбцы, содержащие названия стран и численность населения. Созданная карта отображается на листе и открывается диалоговая панель *Оформление карты*.

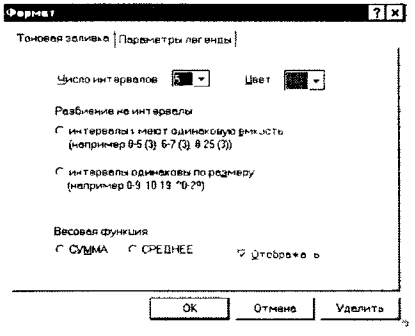
Статистические данные на карте могут отображаться различными способами. Форматы *Тоновая заливка*, *Цветовая заливка* и *Плотность точек* позволяют отобразить один ряд данных, а форматы *Круговая диаграмма* и *Гистограмма* могут отображать несколько рядов данных.

4. На диалоговой панели *Оформление карты* выбрать формат *Тоновая заливка*.



В выбранном формате *Тоновая заливка* значения численности населения разбиваются на интервалы и страны, входящие в каждый из интервалов, закрашиваются своим оттенком цвета. Количество интервалов и способ разбиения на интервалы можно изменить.

5. Щелкнуть по кнопке *Тоновая заливка*.
   На появившейся диалоговой панели *Формат* установить количество интервалов (например, 5), способы разбиения на интервалы и базовый цвет.



6. На карте мира каждая из стран будет закрашена одним из пяти оттенков выбранного цвета.
   Легенда карты содержит информацию о числовых значениях интервалов и количестве стран, попавших в каждый из интервалов.

Геоинформационная модель хранится в файле mapstats.xls ***в каталоге*** \textbook\Excel\ CD-ROM

Практические задания

5.13. Найти на интерактивной карте в Интернете свой регион и получить о нем дополнительную информацию.

5.14. Создать геоинформационную модель, отображающую статистические данные о численности населения стран Европы.

5.10. Оптимизационное моделирование в экономике

В сфере управления сложными системами (например, в экономике) применяется оптимизационное моделирование, в процессе которого осуществляется поиск наиболее оптимального пути развития системы.

Критерием оптимальности могут быть различные параметры; например, в экономике можно стремиться к максимальному количеству выпускаемой продукции, а можно к ее низкой себестоимости. Оптимальное развитие соответствует экстремальному (максимальному или минимальному) значению выбранного целевого параметра.

Развитие сложных систем зависит от множества факторов (параметров), следовательно, значение целевого параметра зависит от множества параметров. Выражением такой зависимости является целевая функция

$$K = F(X_1, X_2, \ldots, X_n),$$

где K — значение целевого параметра;

$X_1, X_2, \ldots, X_n$ — параметры, влияющие на развитие системы.

Цель исследования состоит в нахождении экстремума этой функции и определении значений параметров, при которых этот экстремум достигается. Если целевая функция нелинейна, то она имеет экстремумы, которые находятся определенными методами.

Однако часто целевая функция линейна и, соответственно, экстремумов не имеет. Задача поиска оптимального режима при линейной зависимости приобретает смысл только при наличии определенных ограничений на параметры. Если ограничения на параметры (система неравенств) также имеют линейный характер, то такие задачи являются задачами *линейного программирования*. (Термин «линейное про-

граммирование» в имитационном моделировании понимается как поиск экстремумов линейной функции, на которую наложены ограничения.)

Рассмотрим в качестве примера экономического моделирования поиск вариантов оптимального раскроя листов материала на заготовки определенного размера.

Содержательная постановка проблемы. В ходе производственного процесса из листов материала получают заготовки деталей двух типов *А* и *Б* тремя различными способами, при этом количество получаемых заготовок при каждом методе различается.

Таблица 5.2. Способы раскроя заготовок

Тип заготовки	Количество заготовок		
	Способ 1 раскроя	Способ 2 раскроя	Способ 3 раскроя
А	10	3	8
Б	3	6	4

Необходимо выбрать оптимальное сочетание способов раскроя, для того чтобы получить 500 заготовок первого типа и 300 заготовок второго типа при расходовании наименьшего количества листов материала.

Формальная модель. Параметрами, значения которых требуется определить, являются количества листов материала, которые будут раскроены различными способами:

X_1 — количество листов, раскроенное способом 1;

X_2 — количество листов, раскроенное способом 2;

X_3 — количество листов, раскроенное способом 3.

Тогда целевая функция, значением которой является количество листов материала, примет вид:

$$F = X_1 + X_2 + X_3.$$

Ограничения определяются значениями требуемых количеств заготовок типа *А* и *Б*, тогда с учетом количеств заготовок, получаемых различными способами, должны выполняться два равенства:

$$10X_1 + 3X_2 + 8X_3 = 500;$$
$$3X_1 + 6X_2 + 4X_3 = 300.$$

Кроме того, количества листов не могут быть отрицательными, поэтому должны выполняться неравенства:

$$X_1 \geq 0; \quad X_2 \geq 0; \quad X_3 \geq 0.$$

Таким образом, необходимо найти удовлетворяющие ограничениям значения параметров, при которых целевая функция принимает минимальное значение.

Компьютерная модель. Будем искать решение задачи путем создания и исследования компьютерной модели в электронных таблицах Excel.

Оптимизационное моделирование

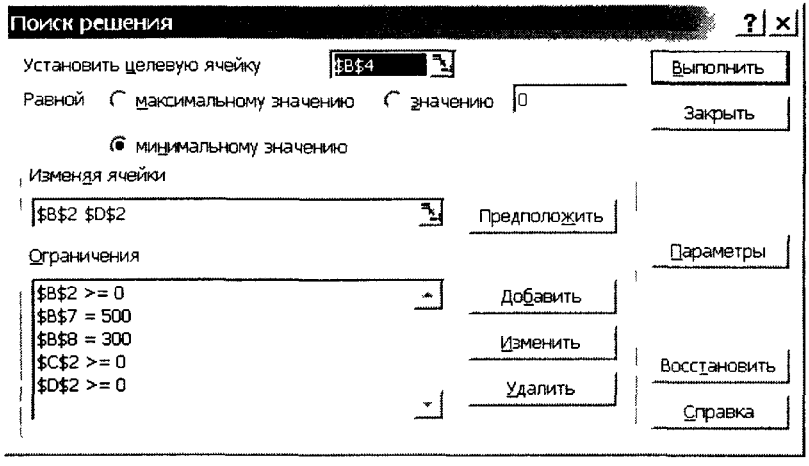
1. Ячейки B2, C2 и D2 выделить для хранения значений параметров X1, X2 и X3.
   В ячейку B4 ввести формулу вычисления целевой функции: =B2+C2+D2.
   В ячейку B7 ввести формулу вычисления количества заготовок типа А: =10*B2+ 3*C2 + 8*D2 .

	A	B	C	D
1		X1	X2	X3
2	Параметры:	0	0	0
3				
4	Целевая функция.	0		
5				
6	Ограничения			
7	Кол-во заготовок А:	0		
8	Кол-во заготовок Б:	0		

   В ячейку B8 ввести формулу вычисления количества заготовок типа Б: =3*B2+ 6*C2 + 4*D2 .

Исследование модели. Для поиска оптимального набора значений параметров, который соответствует минимальному значению целевой функции, воспользоваться надстройкой электронных таблиц *Поиск решения.*

2. Для активизации надстройки ввести команду [Сервис-Надстройки...]. На диалоговой панели установить флажок перед элементом списка *Поиск решения.*
3. Ввести команду [Сервис-Поиск решения...]. На появившейся диалоговой панели *Поиск решения* установить:
   - адрес целевой ячейки;
   - вариант оптимизации значения целевой ячейки (максимизация, минимизация или подбор значения);
   - адреса ячеек, значения которых изменяются в процессе поиска решения (в которых хранятся значения параметров);
   - ограничения (типа «=» для ячеек, хранящих количество деталей, и типа «≥» для параметров).



4. Щелкнуть по кнопке *Выполнить*. В ячейке целевой функции появится значение 70, а в ячейках параметров значения 20, 20, 30.

	A	B	C	D
1		X1	X2	X3
2	**Параметры:**	20	20	30
3				
4	**Целевая функция:**	70		
5				
6	**Ограничения**			
7	**Кол во заготовок А:**	500		
8	**Кол во заготовок Б:**	300		

Таким образом, для изготовления 500 деталей *А* и 300 деталей *Б* требуется 70 листов материала, при этом 20 листов необходимо раскроить по первому, 20 листов — по второму и 30 листов — по третьему варианту.

Модель хранится в файле model xls
в каталоге \textbook\Excel\ **CD-ROM**

Практические задания

5.15. Построить компьютерную модель оптимизации раскроя листов материала на языке Visual Basic.

5.11. Экспертные системы распознавания химических веществ

Химия-9 

Профессиональные экспертные системы — это интеллектуальные программы, способные делать логические выводы на основе знаний из конкретной предметной области, обеспечивающие решение диагностических задач и способные заменить специалиста (эксперта). Профессиональные экспертные системы имеют довольно сложную структуру и состоят обычно из нескольких компонентов (модулей):

- механизма представления знаний в конкретной предметной области (базы знаний);
- механизма, который на основании знаний, имеющихся в базе знаний, способен делать логические выводы (механизма логического вывода);
- пользовательского интерфейса ведения диалога «экспертная система — пользователь»;
- механизма получения знаний от эксперта; этот механизм позволяет дополнять и развивать базу знаний (модуль приобретения знаний);
- механизма, дающего комментарии и разъяснения найденного решения (модуля советов и объяснений).

Простейшие модели экспертных систем, учебные экспертные системы способны «решать» гораздо более простые задачи и, естественно, имеют намного более простую структуру. В школьном курсе встречается достаточно много учебных ситуаций, когда ученик выступает в роли эксперта и должен распознать (идентифицировать) тот или иной объект. Обычно такие задачи выполняются учеником методом проб и ошибок, без осознания и фиксации стратегии поиска.

Создание учебной экспертной системы как раз и является осознанием и фиксацией последовательности рассуждений (действий), которая приводит к распознаванию того или иного объекта среди некоторой совокупности.

В качестве примера можно рассмотреть лабораторную работу по химии «Распознавание химических удобрений». Учащемуся даются удобрения, химические реактивы и справочная таблица по взаимодействию шести различных удобрений с некоторыми реактивами (табл. 5.3). Учащемуся предлагается распознать каждое из удобрений.

Таблица 5.3. Свойства удобрений

№	Внешний вид	Взаимодействие раствора удобрения			Удобрение (результат распознавания)
		с H_2SO_4	с BaCl	с раствором щелочи	
1	Белая кристаллическая масса или гранулы	Выделяется бурый газ	—	Ощущается запах аммиака	Аммиачная селитра
2	Крупные бесцветные кристаллы	Выделяется бурый газ	Небольшое помутнение раствора	—	Натриевая селитра
3	Мелкие светло-серые кристаллы	—	Выпадает белый осадок	Ощущается запах аммиака	Сульфат аммония
4	Светло-серый порошок или гранулы	—	Выпадает белый осадок	—	Суперфосфат
5	Розовые кристаллы	—	—	—	Сильвинит
6	Бесцветные кристаллы	—	—	—	Калийная соль

Стратегия поиска может быть представлена в виде дерева поиска на основе структуры «если...то...иначе», причем может быть множество различных деревьев с различным количеством шагов. Выбор оптимальной стратегии распознавания (достижения цели за минимальное число шагов) и будет являться созданием учебной экспертной системы. Такая стратегия будет реализована, если каждый шаг будет максимально уменьшать неопределенность (нести максимальное количество информации).

Формальная модель. Целесообразно представить иерархическую модель экспертной системы в виде блок-схемы:

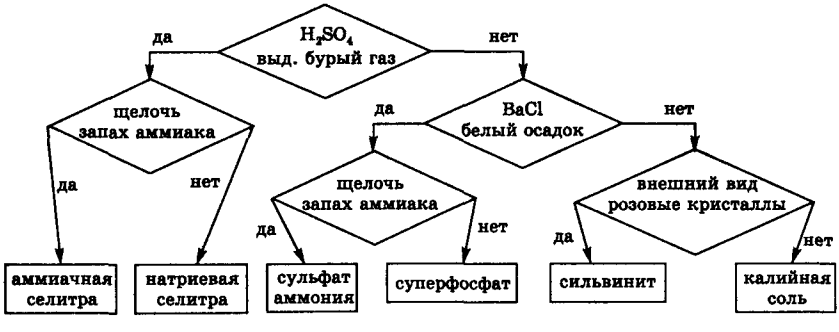


Рис. 5.11. Блок-схема экспертной системы «Распознавание удобрений»

Компьютерная модель. Реализуем экспертную систему распознавания удобрений с использованием языка Visual Basic. Функционирование такой учебной экспертной системы реализуем в диалоге «система–пользователь». Экспертная система задает пользователю серию вопросов, анализирует ответы и сравнивает с имеющимися в ней фактами. При этом производится логический вывод и формируется ответ на интересующий пользователя вопрос, то есть определяется название удобрения.

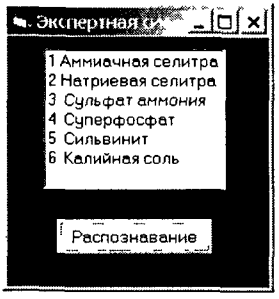
Экспертная система «Распознавание удобрений»

1. Поместить на форму кнопку и управляющий элемент ListBox (*список*). Создать событийную процедуру, реализующую диалог с пользователем путем вызова общих процедур и вывод названий удобрений в элементы списка:

```
Sub Щелочь()
bytA = MsgBox("При взаимодействии со щелочью
ощущается запах аммиака?", 36, "Второй вопрос")
If bytA = 6 Then lst1.AddItem
"1.Аммиачная селитра" Else lst1.AddItem
"2.Натриевая селитра"
End Sub
Sub Соль()
bytA = MsgBox("При взаимодействии с солью
выпадает белый осадок?", 36, "Второй вопрос")
If bytA = 6 Then Щелочь1 Else Внешний_вид
End Sub
Sub Щелочь1()
bytA = MsgBox("При взаимодействии со щелочью
ощущается запах аммиака?", 36, "Третий вопрос")
If bytA = 6 Then lst1.AddItem "3. Сульфат
аммония" Else lst1.AddItem "4. Суперфосфат"
End Sub
Sub Внешний_вид()
bytA = MsgBox("Розовые кристаллы?", 36,
"Третий вопрос")
If bytA = 6 Then lst1.AddItem "5. Сильвинит"
Else lst1.AddItem "6. Калийная соль"
End Sub
Private Sub cmd1_Click()
bytA = MsgBox("При взаимодействии с серной
кислотой выделяется бурый газ?", 36,
"Первый вопрос")
If bytA = 6 Then Щелочь Else Соль
End Sub
```

Компьютерный эксперимент. Работа с экспертной системой позволит более эффективно спланировать и провести распознавание удобрений в процессе выполнения лабораторной работы по химии.

2. Запустить экспертную систему и проводить химические опыты в соответствии с задаваемыми вопросами. Проделать процедуру распознавания для каждого вещества.



Проект хранится в каталоге \textbook\VB\prjHim\ CD-ROM

Практические задания

5.16. Построить экспертную систему для лабораторных работ «Распознавание волокон» и «Распознавание пластмасс».

5.12. Модели логических устройств

При изучении базовых логических устройств компьютера (сумматор, триггер) целесообразно использовать компьютерные модели. Такие модели позволяют визуализировать процесс преобразования логических значений входных сигналов в значения выходных сигналов.

3.7. Логические основы устройства компьютера

Ранее были построены формальные логические модели устройств компьютера. Так, двоичный одноразрядный полусумматор состоит из четырех базовых логических элементов (два конъюнктора, один дизъюнктор и один инвертор). На вход полусумматора подаются сигналы двух слагаемых A и B, а на выходе имеются сигнал суммы S и сигнал переноса в старший разряд P.

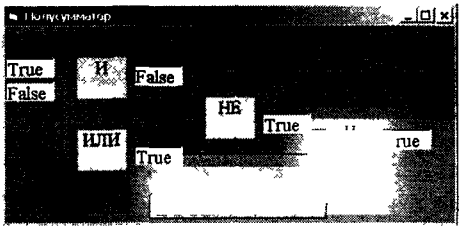
Построим компьютерную модель полусумматора с использованием языка программирования Visual Basic.

Модель полусумматора

1. Поместить на форму четыре метки для изображения базовых логических элементов и шесть текстовых полей для ввода и вывода логических значений.
2. Создать событийную процедуру, реализующую определение логических значений на выходе каждого базового логического элемента и их вывод в текстовые поля:

```
Dim blnA, blnB, blnP, blnS As Boolean
Sub cmd1_Click()
blnA = txtA.Text
blnB = txtB.Text
blnP = blnA And blnB
blnS = (blnA Or blnB) And Not (blnA And blnB)
txtP.Text = blnP
txtOtr.Text = Not blnP
txtOr.Text = blnA Or blnB
txtS.Text = blnS
End Sub
```

3. Запустить проект, ввести логические значения аргументов и щелкнуть по кнопке *Перенос и сумма*. В текстовые поля будут выведены логические значения на выходах логических элементов.



Проект хранится в каталоге \textbook\VB\prjLog\ **CD-ROM**

Практические задания

5.17. Создать компьютерную модель полусумматора с использованием электронных таблиц.

5.13. Информационные модели управления объектами

В процессе функционирования сложных систем (биологических, технических и пр.) входящие в них объекты постоянно обмениваются информацией. Для поддержания своей жизнедеятельности любой живой организм постоянно получает информацию из внешнего мира с помощью органов чувств, обрабатывает ее и управляет своим поведением (например, перемещаясь в пространстве, избегает опасности).

В процессе управления полетом самолета в режиме автопилота бортовой компьютер получает информацию от датчиков (скорости, высоты и пр.), обрабатывает ее и передает команды на исполнительные механизмы, изменяющие режим полета (закрылки, клапаны, регулирующие работу двигателей, и пр.).

В любом процессе управления всегда происходит взаимодействие двух объектов — *управляющего* и *управляемого*, которые соединены каналами *прямой* и *обратной связи*. По каналу прямой связи передаются управляющие сигналы, а по каналу обратной связи — информация о состоянии управляемого объекта.

Разомкнутые системы управления. Если в процессе управления не учитывается состояние управляемого объекта и обеспечивается управление только по прямому каналу (от управляющего объекта к управляемому), то такие системы управления называются *разомкнутыми*. Информационную модель разомкнутой системы управления можно наглядно представить с помощью схемы, представленной на рис. 5.12.

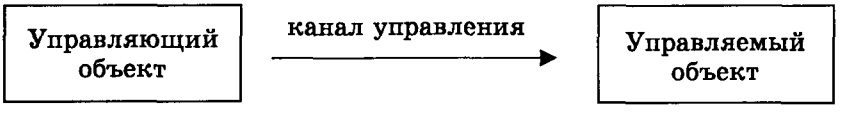


Рис. 5.12. Разомкнутая система управления

В качестве примера разомкнутой системы управления рассмотрим процесс записи информации на гибкий диск, в котором объект «*Дисковод*» (управляющий объект) изменяет состояние объекта «*Дискета*» (управляемый объект).

Для того чтобы информация могла быть записана, необходимо установить магнитную головку дисковода над определенной концентрической дорожкой диска. При записи информации на гибкие диски не требуется особой точности установки (имеется всего 80 дорожек) и можно не учитывать возможные (например, от нагревания) механические деформации носителя, поэтому управляющий объект (дисковод) просто перемещает магнитную головку на определенное расстояние вдоль радиуса управляемого объекта (дискеты).

Для демонстрации принципа работы разомкнутых систем управления разработаем компьютерную модель на языке программирования Visual Basic. Пусть управляемым объектом будет точка, которую управляющий объект (пользователь) должен переместить в центр мишени (круга). Прямое управление положением точки будем производить путем нажатия на кнопки, которые перемещают объект вверх, вниз, влево и вправо. Обратная связь будет отсутствовать.

Модель разомкнутой системы управления

1. Поместить на форму графическое поле, по которому будет перемещаться точка, кнопку для вывода первоначального положения точки, четыре кнопки для управления движением точки и кнопку для вывода мишени.

2. Событийная процедура первоначального вывода точки должна включать задание масштаба и случайную генерацию координат точки:

```
Dim bytX1, bytY1, bytX2, bytY2 As Byte
Private Sub cmdP_Click()
pic1.Scale (0, 20)-(20, 0)
bytX1 = Int(Rnd * 20)
bytY1 = Int(Rnd * 20)
pic1.PSet (bytX1, bytY1), vbRed
End Sub
```

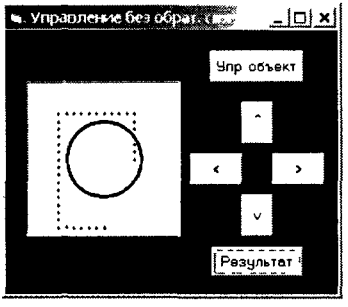
3. Четыре событийные процедуры перемещения точки должны обеспечивать изменение координат точки. Например, для перемещения влево служит событийная процедура:

```
Private Sub cmdL_Click()
pic1.Scale (0, 20)-(20, 0)
bytX1 = bytX1 - 1
pic1.PSet (bytX1, bytY1), vbRed
End Sub
```

4. Событийная процедура вывода мишени:

```
Private Sub cmd2_Click()
pic1.Scale (0, 20)-(20, 0)
pic1.Circle (10, 10), 5
pic1.PSet (bytX1, bytY1), vbBlack
End Sub
```

5. Щелкнуть по кнопке *Упр. объект* и перемещать его кнопками со стрелками. Щелкнуть по кнопке *Результат*. Отклонение точки от центра мишени будет велико.



Проект хранится в каталоге \textbook\VB\prjUpr\ CD-ROM

Замкнутые системы управления. В замкнутых системах управления управляющий объект по прямому каналу управления производит необходимые действия над объектом управления, а по каналу обратной связи получает информацию о его реальных параметрах. Это позволяет осуществлять управление с гораздо большей точностью.

Информационную модель замкнутой системы управления можно наглядно представить с помощью схемы, представленной на рис. 5.13.

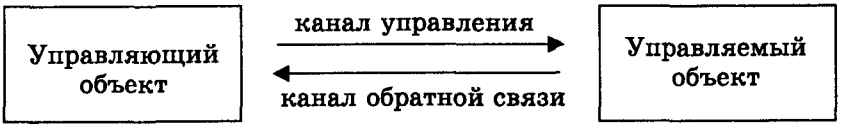


Рис. 5.13. Замкнутая система управления

Примером использования замкнутой системы управления является процесс записи на жесткие диски. При записи информации на жесткие диски требуется особая точность установки магнитных головок, так как на рабочей поверхности носителя имеются тысячи дорожек и необходимо учитывать механические деформации магнитного носителя (например, в результате изменения температуры). Система управления

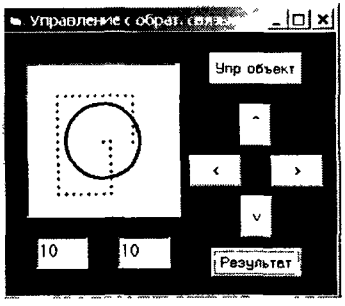
магнитными головками винчестера постоянно получает информацию о реальном положении магнитных головок по каналу обратной связи, а по прямому каналу выставляет головки над поверхностью носителя с большой точностью.

Для демонстрации принципа работы замкнутых систем управления усовершенствуем компьютерную модель перемещения точки в центр мишени. Для осуществления обратной связи будем выводить значения координат точки в текстовые поля.

Модель замкнутой системы управления

1. Усовершенствовать предыдущий проект и поместить на форму два текстовых поля.
   В коды процедур добавить строки:

```
txtX.Text = bytX1
txtY.Text = bytY1
```



Использование обратной связи обеспечивает гарантированное попадание точки в мишень.

Проект хранится в каталоге \textbook\VB\prjUpr1\ CD-ROM

Вопросы для размышления

1. В чем состоит различие разомкнутых и замкнутых систем управления? Приведите примеры.

Практические задания

5.18. Создать компьютерную модель замкнутой системы управления с автоматической обратной связью.

Глава 6

Информатизация общества

6.1. Информационное общество

Человеческое общество по мере своего развития прошло этапы овладения веществом, затем энергией и, наконец, информацией. В первобытно-общинном, рабовладельческом и феодальном обществах (в основе существования которых лежало ремесло) деятельность общества в целом и каждого человека в отдельности была направлена, в первую очередь, на овладение веществом.

На заре цивилизации (десятки тысяч лет до н. э.) люди научились изготавливать простые орудия труда и охоты (каменный топор, стрелы и так далее), в античности появились первые механизмы (рычаг и др.) и средства передвижения (колесницы, корабли), в средние века были изобретены первые сложные орудия труда и механизмы (ткацкий станок, часы).

Овладение энергией находилось в этот период на начальной ступени, в качестве источников энергии использовались Солнце, вода, огонь, ветер и мускульная сила человека.

С самого начала человеческой истории возникла потребность передачи и хранения информации. Для передачи информации сначала использовался язык жестов, а затем человеческая речь. Для хранения информации стали использоваться наскальные рисунки, а в IV тысячелетии до нашей эры появилась письменность и первые носители информации (шумерские глиняные таблички и египетские папирусы). История создания устройств для обработки числовой информации начинается также еще с древности — с абака (счетной доски, являющейся прообразом счетов).

Индустриальное общество. Начиная примерно с XVII века в процессе становления машинного производства на первый план выходит проблема овладения энергией (маши-

ны и станки необходимо было приводить в движение). Сначала совершенствовались способы овладения энергией ветра и воды (ветряные мельницы и водяные колеса), а затем человечество овладело тепловой энергией (в середине XVIII века была изобретена паровая машина, а в конце XIX века — двигатель внутреннего сгорания).

В конце XIX века началось овладение электрической энергией, были изобретены электрогенератор и электродвигатель. И наконец, в середине XX века человечество овладело атомной энергией, в 1954 году в СССР была пущена в эксплуатацию первая атомная электростанция.

Овладение энергией позволило перейти к массовому машинному производству потребительских товаров, было создано *индустриальное общество.* Основными показателями развитости индустриального общества являлись количественные показатели, то есть сколько было добыто угля и нефти, сколько произведено станков и так далее.

В этот период происходили также существенные изменения в способах хранения и передачи информации. В середине XV века было изобретено книгопечатание, что позволило сделать информацию доступной для гораздо большего количества людей. С конца XIX века для передачи информации на дальние расстояния по проводам стали широко использоваться телеграф и телефон, а в XX веке — электромагнитные волны (радио, телевидение).

Информационное общество. Первой попыткой автоматизированной обработки информации стало создание Чарльзом Бэббиджем в середине XIX века механической цифровой аналитической машины. Однако лишь с середины XX века, с момента появления электронных устройств обработки и хранения информации (ЭВМ, а затем персонального компьютера), начался постепенный переход от индустриального общества к информационному.

В *информационном обществе* главным ресурсом является информация, именно на основе владения информацией о самых различных процессах и явлениях можно эффективно и оптимально строить любую деятельность.

Важно не только произвести большое количество продукции, но произвести нужную продукцию в определенное время, с определенными затратами и так далее. Поэтому в информационном обществе повышается не только качество потребления, но и качество производства; человек, использующий информационные технологии, имеет лучшие условия труда, труд становится творческим, интеллектуальным и так далее.

В настоящее время развитые страны мира (США, Япония, страны Западной Европы) фактически уже вступили в информационное общество, другие же, в том числе и Россия, находятся на ближних подступах к нему.

В качестве критериев развитости информационного общества можно выбрать три: *наличие компьютеров, уровень развития компьютерных сетей* и *количество населения, занятого в информационной сфере*, а также использующего информационные и коммуникационные технологии в своей повседневной деятельности.

Производство компьютеров. Первые электронно-вычислительные машины (ЭВМ), которые могли автоматически по заданной программе обрабатывать большие объемы информации, были созданы в 1946 году в США (ЭНИАК) и в 1950 году в СССР (МЭСМ). В 40–60-х годах производство ЭВМ измерялась единицами, десятками и, в лучшем случае, сотнями штук. ЭВМ были очень дорогими и очень большими (занимали громадные залы) и поэтому оставались недоступными для массового потребителя.

Массовое производство сравнительно недорогих персональных компьютеров началось с середины 70-х годов XX века с компьютера Apple II (с этого компьютера начала свое существование фирма Apple). Количество производимых персональных компьютеров начало составлять десятки тысяч в год, что по тем временам было колоссальным достижением.

В начале 80-х годов приступила к массовому производству персональных компьютеров корпорация IBM (компьютеры так и назывались IBM Personal Computer — IBM PC). Достаточно скоро IBM-совместимые компьютеры стали выпускать многие фирмы, и их производство достигло сотен тысяч в год. Ежегодное производство персональных компьютеров постоянно росло и в 2000 году превысило 150 миллионов.

Персональный компьютер постоянно совершенствовался, его производительность возросла на три порядка, при этом, что очень важно, цена практически не изменилась. Персональный компьютер стал доступен массовому потребителю, и теперь в развитых странах мира компьютер имеется на большинстве рабочих мест и в большинстве семей.

Компьютерные сети. В настоящее время существенной тенденцией в информатизации общества является переход от использования компьютеров в автономном режиме к использованию их в информационных сетях.

Информационные сети создают реальную возможность быстрого и удобного доступа пользователя ко всей информации, накопленной человечеством за всю свою историю. Электронная почта и телеконференции, поиск информации во Всемирной паутине и в файловых архивах, интерактивное общение, прослушивание радиостанций и просмотр телевизионных программ, покупки в Интернет-магазинах стали повседневной практикой многих пользователей компьютеров в развитых странах.

Развитие глобальных компьютерных сетей началось в 80-е годы. В 1981 году в сети Интернет насчитывалось лишь 213 компьютеров, к концу 80-х годов количество подключенных к сети компьютеров возросло до 150 тысяч, однако наиболее быстрый экспоненциальный рост их количества происходил в 90-е годы.

Чтобы убедиться в этом, достаточно проследить рост количества серверов в глобальной компьютерной сети Интернет, которое к началу 2002 года достигло почти 150 миллионов (рис. 6.1). (Статистические данные о росте Интернета можно найти на сервере Internet Software Consortium по адресу http://www.isc.org).

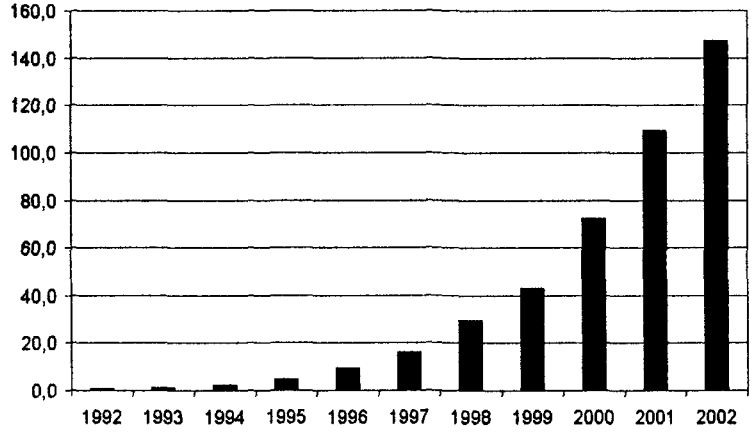


Рис. 6.1. Рост количества серверов Интернета

По количеству имеющихся серверов Интернета можно судить о степени информатизации отдельных стран. Наибольшее количество серверов зарегистрировано в доменах административного типа, которые находятся в основном в США (около 104 миллионов серверов), на втором месте, с большим

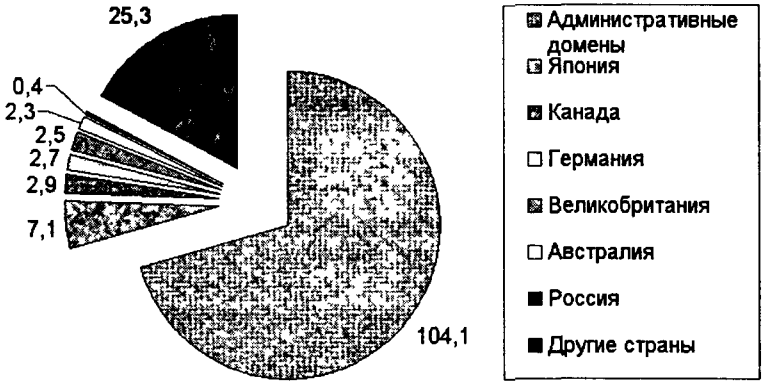


Рис. 6.2. Распределение серверов Интернета по странам мира

отставанием, Япония (7,1 миллионов серверов), Россия занимает в этом списке 24-е место (около 400 тысяч серверов).

Развитие глобальной компьютерной сети требует наличия каналов связи с высокой пропускной способностью. Основой глобальной компьютерной сети Интернет являются магистральные высокоскоростные линии связи, по которым передается информация между региональными сетями. В настоящее время наиболее мощные региональные сети функционируют в Северной Америке, Европе, Японии и Австралии. Они соединены между собой многочисленными оптоволоконными линиями связи с пропускной способностью до 20 Гбит/с и выше.

Внутри региональных сетей информация передается также преимущественно по оптоволоконным каналам с различной пропускной способностью (от 1 до 155 Мбит/с). В региональных сетях часто используются также выделенные линии (медные), а иногда (в пределах прямой видимости) и радиоканалы, пропускная способность которых может достигать 2 Мбит/с.

Для подключения отдаленных регионов наиболее экономически выгодным является подключение по спутниковым каналам, пропускная способность которых может достигать десятков мегабитов в секунду.

Однако для большинства индивидуальных пользователей (их сейчас в мире около 1 миллиарда) приемлемым по цене является доступ в Интернет только по коммутируемым телефонным каналам со скоростью до 56 Кбит/с. В России, по разным оценкам, таких пользователей от 3 до 5 миллионов.

Население, занятое в информационной сфере. По данным ООН, в 90-е годы количество работников, занятых в информационной сфере (для которых обработка информации является основной производственной функцией), возросло примерно на 25%, тогда как количество занятых в сельском хозяйстве и промышленности сократилось соответственно на 10 и 15%.

Компьютеры и информационные технологии интенсивно проникают и в сферу материального производства. Инженер, фермер, специалисты других традиционных профессий все чаще имеют на своем рабочем месте компьютер и используют информационные и коммуникационные технологии в своей профессиональной деятельности.

С развитием коммуникационных технологий и мобильной связи все большее количество людей осуществляют свою производственную деятельность дистанционно, то есть работая дома, а не в офисе (в США более 10 миллионов человек). Все большее распространение получает дистанционное образование и поиск работы через Интернет. В 2000 году оборот мирового рынка информационных и коммуникационных технологий составил около 1 триллиона долларов. При этом на закупку аппаратных средств было потрачено менее половины этой суммы, большая часть была вложена в разработку программного обеспечения, проектирование компьютерных сетей и так далее.

Информационное общество — это общество, в котором бóльшая часть населения занята получением, переработкой, передачей и хранением информации.

Курс информатики и информационных технологий играет особую роль в эпоху перехода от индустриального общества к информационному, так как готовит выпускников школы к жизни и деятельности в информационном обществе.

Вопросы для размышления

1. Какую роль играли вещество, энергия и информация на различных этапах развития общества?

2. По каким основным параметрам можно судить о степени развитости информационного общества и почему?
3. Как изменяется содержание жизни и деятельности людей в процессе перехода от индустриального к информационному обществу?

Практические задания

6.1. Ознакомиться в Интернете с его ростом по годам и распределением серверов по странам мира.

6.2. Информационная культура

Количество информации в современном обществе стремительно нарастает, человек оказывается погруженным в море информации. Для того чтобы в этом море «не утонуть», необходимо обладать информационной культурой, то есть знаниями и умениями в области информационных и коммуникационных технологий, а также быть знакомым с юридическими и этическими нормами в этой сфере.

Процесс информатизации общества меняет традиционные взгляды на перечень умений и навыков, необходимых для социальной адаптации. Возьмем традиционный навык письма. На заре цивилизации (Шумер, Египет), в античном мире (Эллада, Римская империя и др.) и в средние века (до изобретения книгопечатания) навык каллиграфического письма был залогом успешного продвижения по социальной лестнице. В индустриальном обществе (до изобретения персональных компьютеров) навыки письма ручкой также были необходимы для любого члена общества.

В настоящее время, на пороге информационного общества, социальная значимость навыка письма ручкой снижается и, наоборот, социальная значимость навыков ввода информации с помощью клавиатуры и работы с графическим интерфейсом приложений с помощью мыши возрастает.

Создание и редактирование документов с помощью компьютера, то есть *овладение офисными информационными технологиями*, становится в информационном обществе социально необходимым умением — достаточно просмотреть объявления о приеме на работу.

Современные информационные технологии позволяют включать в состав документа любые мультимедийные объекты (графику, звук, анимацию, видео). Дома вы можете привести в порядок фотоархив семьи, отсканировав старые фотографии и поместив их в упорядоченном виде в компьютерный фотоальбом; в процессе обучения вы можете подготовить реферат с иллюстрациями, в процессе профессиональной деятельности — создать компьютерную презентацию о деятельности вашей фирмы. *Умение работать с мультимедиа-документами, создавать компьютерные презентации* становится важным в информационном обществе.

В современном информационном обществе вряд ли необходимы навыки традиционного черчения на ватмане. Вместо этого полезно получить *первоначальное представление о назначении и возможностях компьютерных систем автоматизированного проектирования* (*САПР*). Такие системы позволят вам быстро рассмотреть различные варианты планировки интерьера дома или квартиры, создать чертеж или схему.

Использование электронных таблиц сделает более простыми и наглядными процессы исследования и построения графиков функций в процессе изучения математики, планирования и ведения домашнего бюджета, построения и исследования моделей различных объектов и процессов.

Необходимость упорядочить информацию, например, о людях, с которыми вы контактируете, требует использования записной книжки. Однако часто удобнее использовать для хранения такой информации компьютерную базу данных «Записная книжка». При поиске информации в современной библиотеке или в Интернете необходимо иметь навыки поиска информации в базах данных. В информационном обществе очень полезным является *умение создавать базы данных, а также вести в них поиск данных.*

Квалифицированный пользователь компьютера может на основе *использования средств визуального объектно-ориентированного программирования* создавать необходимые ему специализированные приложения. Например, можно создать приложение, которое автоматизирует заполнение многочисленных квитанций оплаты за квартиру, электроэнергию, газ и др.

Современному человеку необходимо овладеть *коммуникативной культурой*, то есть умениями создавать и посылать электронные письма, находить нужную информацию во Все-

мирной паутине или в файловых архивах, участвовать в чатах и так далее. Необходимым условием успешной профессиональной деятельности становится создание и публикация в Интернете Web-сайтов с информацией о деятельности организации или предприятия.

Информационная культура состоит не только в овладении определенным комплексом знаний и умений в области информационных и коммуникационных технологий, но предполагает *знание и соблюдение юридических и этических норм и правил*. Законы запрещают использование пиратского компьютерного обеспечения и пропаганду насилия, наркотиков и порнографии в Интернете. Общение с помощью электронной почты или в чатах, участие в телеконференциях предполагают соблюдение определенных правил: отвечать на письма и не рассылать знакомым и незнакомым людям многочисленные рекламные сообщения (спам), не отклоняться от темы обсуждения в телеконференциях и чатах и так далее.

Вопросы для размышления

1. Каковы основные компоненты информационной культуры, которые необходимы человеку для жизни в информационном обществе?

6.3. Правовая охрана программ и данных. Защита информации

6.3.1. Лицензионные, условно бесплатные и бесплатные программы

Программы по их юридическому статусу можно разделить на три большие группы: *лицензионные*, *условно бесплатные* (shareware) и *свободно распространяемые программы* (freeware).

Дистрибутивы лицензионных программ (дискеты или диски CD-ROM, с которых производится установка программ на компьютеры пользователей) распространяются разработчиками на основании договоров с пользователями на платной основе, проще говоря, лицензионные программы продаются. Довольно часто разработчики предоставляют существенные

скидки при покупке лицензий на использование программы на большом количестве компьютеров или на использование программы в учебных заведениях. В соответствии с лицензионным соглашением разработчики программы гарантируют ее нормальное функционирование в определенной операционной системе и несут за это ответственность.

Некоторые фирмы — разработчики программного обеспечения предлагают пользователям условно бесплатные программы в целях их рекламы и продвижения на рынок. Пользователю предоставляется версия программы с ограниченным сроком действия (после истечения указанного срока программа перестает работать, если за нее не произведена оплата) или версия программы с ограниченными функциональными возможностями (в случае оплаты пользователю сообщается код, включающий все функции).

Многие производители программного обеспечения и компьютерного оборудования заинтересованы в широком бесплатном распространении программного обеспечения. К таким программным средствам можно отнести следующие:

- новые недоработанные (бета) версии программных продуктов (это позволяет провести их широкое тестирование);
- программные продукты, являющиеся частью принципиально новых технологий (это позволяет завоевать рынок);
- дополнения к ранее выпущенным программам, исправляющие найденные ошибки или расширяющие возможности;
- устаревшие версии программ;
- драйверы к новым устройствам или улучшенные драйверы к уже существующим.

Вопросы для размышления

1. В чем состоит различие между лицензионными, условно бесплатными и бесплатными программами?

6.3.2. Правовая охрана информации

Правовая охрана программ и баз данных. Правовая охрана программ для ЭВМ и баз данных впервые в полном объеме введена в Российской Федерации Законом РФ «О право-

вой охране программ для электронных вычислительных машин и баз данных», который вступил в силу в 1992 году.

Предоставляемая настоящим законом правовая охрана распространяется на все виды программ для ЭВМ (в том числе на операционные системы и программные комплексы), которые могут быть выражены на любом языке и в любой форме, включая исходный текст на языке программирования и машинный код. Однако правовая охрана не распространяется на идеи и принципы, лежащие в основе программы для ЭВМ, в том числе на идеи и принципы организации интерфейса и алгоритма.

Для признания и осуществления авторского права на программы для ЭВМ не требуется ее регистрация в какой-либо организации. Авторское право на программы для ЭВМ возникает автоматически при их создании.

Для оповещения о своих правах разработчик программы может, начиная с первого выпуска в свет программы, использовать знак охраны авторского права, состоящий из трех элементов:

- буквы С в окружности или круглых скобках ©;
- наименования (имени) правообладателя;
- года первого выпуска программы в свет.

Например, знак охраны авторских прав на текстовый редактор Word выглядит следующим образом:

© Корпорация Microsoft, 1993–1997.

Автору программы принадлежит исключительное право осуществлять воспроизведение и распространение программы любыми способами, а также модификацию программы.

Организация или пользователь, правомерно владеющий экземпляром программы (купивший лицензию на ее использование), вправе без получения дополнительного разрешения разработчика осуществлять любые действия, связанные с функционированием программы, в том числе ее запись и хранение в памяти ЭВМ. Запись и хранение в памяти ЭВМ допускаются в отношении одной ЭВМ или одного пользователя в сети, если другое не предусмотрено договором с разработчиком.

Необходимо знать и выполнять существующие законы, запрещающие нелегальное копирование и использование лицензионного программного обеспечения. В отношении организаций или пользователей, которые нарушают авторские права, разработчик может потребовать возмещения причиненных убытков и выплаты нарушителем компенсации в определяемой по усмотрению суда сумме от 5000-кратного

до 50 000-кратного размера минимальной месячной оплаты труда.

Электронная подпись. В 2002 году был принят Закон РФ «Об электронно-цифровой подписи», который стал законодательной основой электронного документооборота в России. По этому закону электронная цифровая подпись в электронном документе признается юридически равнозначной подписи в документе на бумажном носителе.

При регистрации электронно-цифровой подписи в специализированных центрах корреспондент получает два ключа: секретный и открытый. Секретный ключ хранится на дискете или смарт-карте и должен быть известен только самому корреспонденту. Открытый ключ должен быть у всех потенциальных получателей документов и обычно рассылается по электронной почте.

Процесс электронного подписания документа состоит в обработке с помощью секретного ключа текста сообщения. Далее зашифрованное сообщение посылается по электронной почте абоненту. Для проверки подлинности сообщения и электронной подписи абонент использует открытый ключ.

Вопросы для размышления

1. Как можно зафиксировать свое авторское право на программный продукт?

Практические задания

6.2. Ознакомиться с Законами РФ «О правовой охране программ для электронных вычислительных машин и баз данных» и «Об электронно-цифровой подписи», которые находятся на CD-ROM в каталоге \textbook\.

6.3.3. Защита информации

Защита доступа к компьютеру. Для предотвращения несанкционированного доступа к данным, хранящимся на компьютере, используются пароли. Компьютер разрешает доступ к своим ресурсам только тем пользователям, которые зарегистрированы и ввели правильный пароль. Каждому конк-

ретному пользователю может быть разрешен доступ только к определенным информационным ресурсам. При этом может производиться регистрация всех попыток несанкционированного доступа.

Защита пользовательских настроек имеется в операционной системе Windows (при загрузке системы пользователь должен ввести свой пароль), однако такая защита легко преодолима, так как пользователь может отказаться от введения пароля. Вход по паролю может быть установлен в программе BIOS Setup, компьютер не начнет загрузку операционной системы, если не введен правильный пароль. Преодолеть такую защиту нелегко, более того, возникнут серьезные проблемы доступа к данным, если пользователь забудет этот пароль.

В настоящее время для защиты от несанкционированного доступа к информации все более часто используются биометрические системы авторизации и идентификации пользователей. Используемые в этих системах характеристики являются неотъемлемыми качествами личности человека и поэтому не могут быть утерянными и подделанными. К биометрическим системам защиты информации относятся системы распознавания речи, системы идентификации по отпечаткам пальцев, а также системы идентификации по радужной оболочке глаза.

Защита программ от нелегального копирования и использования. Компьютерные пираты, нелегально тиражируя программное обеспечение, обесценивают труд программистов, делают разработку программ экономически невыгодным бизнесом. Кроме того, компьютерные пираты нередко предлагают пользователям недоработанные программы, программы с ошибками или их демоверсии.

Для того чтобы программное обеспечение компьютера могло функционировать, оно должно быть установлено (инсталлировано). Программное обеспечение распространяется фирмами-производителями в форме *дистрибутивов* на CD-ROM. Каждый дистрибутив имеет свой серийный номер, что препятствует незаконному копированию и установке программ.

Для предотвращения нелегального копирования программ и данных, хранящихся на CD-ROM, может использоваться специальная защита. На CD-ROM может быть размещен закодированный программный ключ, который теряется при копировании и без которого программа не может быть установлена.

Защита от нелегального использования программ может быть реализована с помощью аппаратного ключа, который присоединяется обычно к параллельному порту компьютера. Защищаемая программа обращается к параллельному порту и запрашивает секретный код; если аппаратный ключ к компьютеру не присоединен, то защищаемая программа определяет ситуацию нарушения защиты и прекращает свое выполнение.

Защита данных на дисках. Каждый диск, папка и файл локального компьютера, а также компьютера, подключенного к локальной сети, может быть защищен от несанкционированного доступа. Для них могут быть установлены определенные права доступа (полный, только чтение, по паролю), причем права могут быть различными для различных пользователей.

Для обеспечения большей надежности хранения данных на жестких дисках используются RAID-массивы (Redantant Arrays of Independent Disks — избыточный массив независимых дисков). Несколько жестких дисков подключаются к специальному RAID-контроллеру, который рассматривает их как единый логический носитель информации. При записи информации она дублируется и сохраняется на нескольких дисках одновременно, поэтому при выходе из строя одного из дисков данные не теряются.

Защита информации в Интернете. Если компьютер подключен к Интернету, то в принципе любой пользователь, также подключенный к Интернету, может получить доступ к информационным ресурсам этого компьютера. Если сервер имеет соединение с Интернетом и одновременно служит сервером локальной сети (Интранет-сервером), то возможно несанкционированное проникновение из Интернета в локальную сеть.

Механизмы проникновения из Интернета на локальный компьютер и в локальную сеть могут быть разными:

- загружаемые в браузер Web-страницы могут содержать активные элементы ActiveX или Java-апплеты, способные выполнять деструктивные действия на локальном компьютере;
- некоторые Web-серверы размещают на локальном компьютере текстовые файлы cookie, используя которые можно получить конфиденциальную информацию о пользователе локального компьютера;
- с помощью специальных утилит можно получить доступ к дискам и файлам локального компьютера и др.

Для того чтобы этого не происходило, устанавливается программный или аппаратный барьер между Интернетом и Интранетом с помощью брандмауэра (firewall — межсетевой экран). Брандмауэр отслеживает передачу данных между сетями, осуществляет контроль текущих соединений, выявляет подозрительные действия и тем самым предотвращает несанкционированный доступ из Интернета в локальную сеть.

Вопросы для размышления

1. Какие используются способы идентификации личности при предоставлении доступа к информации?
2. Почему компьютерное пиратство наносит ущерб обществу?
3. Какие существуют программные и аппаратные способы защиты информации?
4. Чем отличается простое копирование файлов от инсталляции программ? Для чего каждый дистрибутив имеет серийный номер?

Раздел II

информационные и коммуникационные технологии

Глава 7

Технология обработки графической информации

7.1. Растровая и векторная графика

7.1.1. Растровые и векторные графические изображения

Все компьютерные изображения разделяют на два типа: растровые и векторные.

Растровая графика. Растровые графические изображения формируются в процессе преобразования графической информации из аналоговой формы в цифровую, например, в процессе сканирования существующих на бумаге или фотопленке рисунков и фотографий, при использовании цифровых фото- и видеокамер, при просмотре на компьютере телевизионных передач с использованием ТВ-тюнера и так далее.

Можно создать растровое графическое изображение и непосредственно на компьютере с использованием графического редактора, загрузить его с CD-ROM или DVD-ROM-дисков или «скачать» из Интернета.

Растровое изображение хранится с помощью точек различного цвета (пикселей), которые образуют строки и столбцы. Каждый пиксель имеет определенное положение и цвет. Хранение каждого пикселя требует определенного количества битов информации, которое зависит от количества цветов в изображении.

Пиксель – минимальный участок изображения, цвет которого можно задать независимым образом.

Качество растрового изображения зависит от размера изображения (количества пикселей по горизонтали и вертикали) и количества цветов, которые можно задать для каждого пикселя.

В качестве примера рассмотрим черно-белое (без градаций серого) изображение стрелки размером 8×7 (рис. 7.1). Легко подсчитать, какой информационный объем файла требуется для хранения этого изображения. Общее количество пикселей равно 56. Так как используется всего два цвета, то для хранения каждого пикселя необходим 1 бит. Таким образом, файл будет иметь объем 56 битов, или 7 байтов.

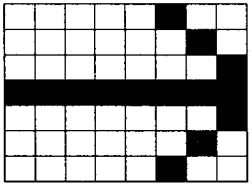


Рис. 7.1. Растровое изображение стрелки

Растровые графические изображения многоцветных фотографий и иллюстраций получают с помощью сканера. Такие изображения обычно имеют большой размер и большую глубину цвета (24 или 36 битов на точку). В результате файлы, хранящие растровые изображения, имеют большой информационный объем.

Растровые изображения очень чувствительны к масштабированию (увеличению или уменьшению). При уменьшении растрового изображения несколько соседних точек преобразуются в одну, поэтому теряется различимость мелких деталей изображения. При увеличении изображения увеличивается размер каждой точки и появляется ступенчатый эффект, который можно увидеть невооруженным глазом (рис. 7.2).

Рис. 7.2
Растровое изображение и его увеличенный фрагмент

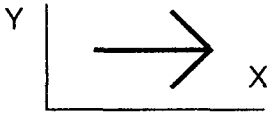
Векторная графика. Векторные графические изображения являются оптимальным средством хранения высокоточных графических объектов (чертежи, схемы и пр.), для которых имеет значение сохранение четких и ясных контуров. С векторной графикой вы сталкиваетесь, когда работаете с системами компьютерного черчения и автоматизированного проектирования (САПР), программами обработки трехмерной графики и др.

Векторные изображения формируются из объектов (точка, линия, окружность, прямоугольник и пр.), которые хранятся в памяти компьютера в виде графических примитивов и описывающих их математических формул.

Например, графический примитив *точка* задается своими координатами (X,Y), *линия* — координатами начала (X1,Y1) и конца (X2,Y2), *окружность* — координатами центра (X,Y) и радиусом (R), *прямоугольник* — координатами левого верхнего угла (X1,Y1) и правого нижнего угла (X2,Y2) и так далее. Для каждого примитива задается также цвет.

Рассмотренная выше стрелка в векторном формате будет задана с помощью трех линий:

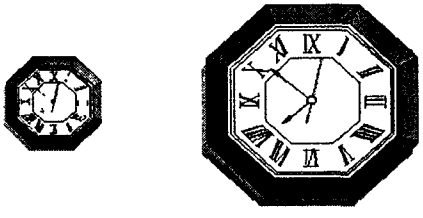
линия (1,4)–(8,4),
линия (6,7)–(8,4),
линия (6,1)–(8,4).



Достоинством векторной графики является то, что файлы, хранящие векторные графические изображения, имеют сравнительно небольшой объем.

Важно также, что векторные графические изображения могут быть увеличены или уменьшены без потери качества (рис. 7.3). Это возможно, так как масштабирование изображений производится с помощью простых математических операций (умножения параметров графических примитивов на коэффициент масштабирования).

Рис. 7.3
Векторное изображение и его увеличенная копия



Вопросы для размышления

1. В чем состоит различие растровых и векторных графических изображений?
2. Какой тип графического изображения (растровый или векторный) вы выберете для разработки символов нового шрифта, учитывая, что шрифт должен масштабироваться без потери качества изображения?

7.1.2. Форматы графических файлов

Форматы графических файлов определяют способ хранения информации в файле (растровый или векторный), а также форму хранения информации (используемый алгоритм сжатия).

Сжатие применяется для растровых графических файлов, так как они имеют обычно достаточно большой объем. Сжатие графических файлов отличается от их архивации с помощью программ-архиваторов (rar, zip, arj и пр.) тем, что алгоритм сжатия включается в формат графического файла.

Существуют различные алгоритмы сжатия, причем для различных типов изображения целесообразно применять подходящие типы алгоритмов сжатия.

Для сжатия рисунков типа аппликации, содержащих большие области однотонной закраски, наиболее эффективно применение алгоритма сжатия, который заменяет последовательность повторяющихся величин (пикселей одинакового цвета) на две величины (пиксель и количество его повторений). Такой алгоритм сжатия используется в графических файлах форматов BMP и PCX.

Для рисунков типа диаграммы целесообразно применение другого метода сжатия, который использует поиск повторяющихся в рисунке «узоров». Такой алгоритм используется в графических файлах форматов TIFF и GIF и позволяет сжать файл в несколько раз.

Для сжатия отсканированных фотографий и иллюстраций используется алгоритм сжатия JPEG. Этот алгоритм использует тот факт, что человеческий глаз очень чувствителен к изменению яркости отдельных точек изображения, но гораздо хуже замечает изменение цвета. Действительно, при глубине цвета 24 бита компьютер обеспечивает воспроизве-

дение более 16 млн различных цветов, тогда как человек вряд ли способен различить и тем более назвать более сотни цветов и оттенков.

Применение метода JPEG позволяет сжимать файлы в десятки раз, однако может приводить к необратимой потере информации (файлы не могут быть восстановлены в первоначальном виде).

Некоторые форматы графических файлов являются универсальными, так как могут быть обработаны большинством графических редакторов. Некоторые программы обработки изображений используют оригинальные форматы, которые распознаются только самой создающей программой. Преимущество оригинальных форматов файлов состоит в том, что они позволяют сохранять изображения при меньшем размере файла.

Рассмотрим некоторые форматы графических файлов более подробно.

Bit MaP image (BMP) — универсальный формат растровых графических файлов, используется в операционной системе Windows. Этот формат поддерживается многими графическими редакторами, в том числе редактором Paint. Рекомендуется для хранения и обмена данными с другими приложениями.

Tagged Image File Format (TIFF) — формат растровых графических файлов, поддерживается всеми основными графическими редакторами и компьютерными платформами. Включает в себя алгоритм сжатия без потерь информации. Используется для обмена документами между различными программами. Рекомендуется для использования при работе с издательскими системами.

Graphics Interchange Format (GIF) — формат растровых графических файлов, поддерживается приложениями для различных операционных систем. Включает алгоритм сжатия без потерь информации, позволяющий уменьшить объем файла в несколько раз. Рекомендуется для хранения изображений, создаваемых программным путем (диаграмм, графиков и так далее) и рисунков (типа аппликации) с ограниченным количеством цветов (до 256). Используется для размещения графических изображений на Web-страницах в Интернете.

Portable Network Graphic (PNG) — формат растровых графических файлов, аналогичный формату GIF. Рекомендуется для размещения графических изображений на Web-страницах в Интернете.

Joint Photographic Expert Group (JPEG) — формат растровых графических файлов, который реализует эффективный алгоритм сжатия (метод JPEG) для отсканированных фотографий и иллюстраций. Алгоритм сжатия позволяет уменьшить объем файла в десятки раз, однако приводит к необратимой потере части информации. Поддерживается приложениями для различных операционных систем. Используется для размещения графических изображений на Web-страницах в Интернете.

Windows MetaFile (WMF) — универсальный формат векторных графических файлов для Windows-приложений. Используется для хранения коллекции графических изображений Microsoft Clip Gallery.

Encapsulated PostScript (EPS) — формат векторных графических файлов, поддерживается программами для различных операционных систем. Рекомендуется для печати и создания иллюстраций в настольных издательских системах.

CorelDRaw files (CDR) — оригинальный формат векторных графических файлов, используемый в системе обработки векторной графики CorelDraw.

Если вы собираетесь работать с графическим файлом только в одном данном приложении, целесообразно выбрать оригинальный формат. Если же предстоит передавать данные в другое приложение, другую среду или иному пользователю, стоит использовать универсальный формат.

Вопросы для размышления

1. Перечислите свойства изображения, которое следует сохранять в формате GIF, и свойства изображения, которое лучше сохранять в формате JPEG.

Практические задания

7.1. Создайте копию экрана, сохраните ее как растровое изображение типа BMP в виде файла и определите его объем. Вычислите объем файла, зная разрешение экрана и глубину цвета, и сравните с объемом файла, полученным экспериментально.

Сохраните изображение в форматах GIF и JPEG, ответьте на вопросы:

- какой из форматов обеспечивает наилучшую степень сжатия?
- какой из форматов обеспечивает наименьшие потери качества изображения?

7.2. Графические редакторы

7.2.1. Растровые и векторные редакторы

Для обработки изображений на компьютере используются специальные программы — *графические редакторы*. Графические редакторы также можно разделить на две категории: растровые и векторные.

Растровые графические редакторы являются наилучшим средством обработки фотографий и рисунков, поскольку растровые изображения обеспечивают высокую точность передачи градаций цветов и полутонов.

Среди растровых графических редакторов есть простые, например стандартное приложение Paint, и мощные профессиональные графические системы, например Adobe Photoshop.

К векторным графическим редакторам относятся графический редактор, встроенный в текстовый редактор Word. Среди профессиональных векторных графических систем наиболее распространена CorelDRAW.

Графический редактор — это программа создания, редактирования и просмотра графических изображений.

Для создания рисунка традиционными методами необходимо выбрать инструмент рисования (это могут быть фломастеры, кисть с красками, карандаши, пастель и многое другое). Графические редакторы также предоставляют возможность выбора инструментов для создания и редактирования графических изображений, объединяя их в *панели инструментов*.

На рис. 7.4 представлены панели инструментов растрового графического редактора Paint и векторного графического редактора, входящего в состав Microsoft Word. Хорошо видно, что панели имеют много одинаковых инструментов.

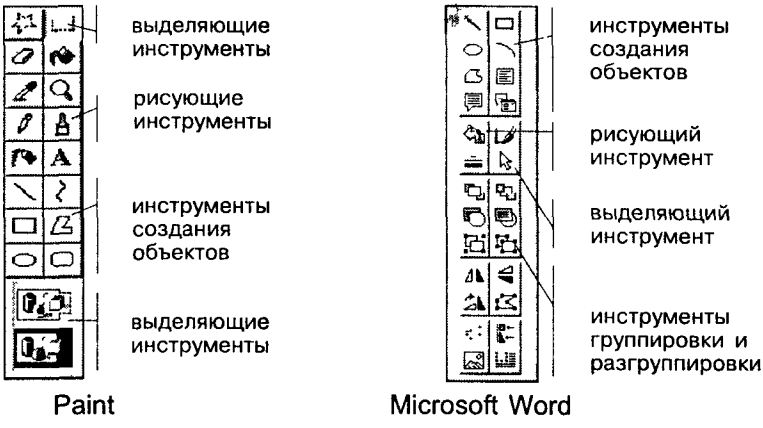


Рис. 7.4. Панели инструментов растрового и векторного графических редакторов

Инструменты рисования объектов. Графические редакторы имеют набор инструментов для рисования простейших графических объектов: прямой линии, кривой, прямоугольника, эллипса, многоугольника и так далее. После выбора объекта на панели инструментов его можно нарисовать в любом месте окна редактора.

Например, для рисования линии необходимо выбрать на панели инструментов инструмент *Линия*, переместить курсор на определенное место окна редактора и щелчком мыши зафиксировать точку, из которой должна начинаться линия. Затем следует перетащить линию в нужном направлении и, осуществив повторный щелчок, зафиксировать второй конец линии.

Такие инструменты имеются и в растровом, и в векторном графических редакторах, однако принципы работы с ними несколько различаются. В растровом графическом редакторе объект перестает существовать как самостоятельный элемент после окончания рисования и становится лишь группой пикселей на рисунке. В векторном редакторе нарисованный объект продолжает сохранять свою индивидуальность и его можно масштабировать, перемещать по рисунку и так далее.

В векторном редакторе существует группа инструментов *группировки и разгруппировки объектов*. Операция группировки объединяет несколько отдельных объектов в один, что позволяет производить в дальнейшем над ними общие операции (перемещать, удалять и так далее). Можно и, наоборот, разбивать объект, состоящий из нескольких объектов, на самостоятельные объекты (разгруппировывать).

Выделяющие инструменты. В графических редакторах над элементами изображения возможны различные операции: копирование, перемещение, удаление, поворот, изменение размеров и так далее. Для того чтобы выполнить какую-либо операцию над объектом, его сначала необходимо выделить.

Для выделения объектов в растровом графическом редакторе обычно имеются два инструмента: *выделение прямоугольной области* и *выделение произвольной области.* Процедура выделения производится аналогично процедуре рисования.

Выделение объектов в векторном редакторе осуществляется с помощью инструмента *выделение объекта* (на панели инструментов изображается стрелкой). Для выделения объекта достаточно выбрать инструмент выделения и щелкнуть по любому объекту на рисунке.

Инструменты редактирования рисунка. Инструменты редактирования позволяют вносить в рисунок изменения: стирать части рисунка, изменять цвета и так далее. Для стирания изображения в растровых графических редакторах используется инструмент *Ластик,* который стирает фрагменты изображения (пиксели), при этом размер *Ластика* можно менять.

В векторных редакторах редактирование изображения возможно только путем удаления объектов, входящих в изображение, целиком. Для этого сначала необходимо выделить объект, а затем выполнить операцию *Вырезать.*

Палитра цветов. Операцию изменения цвета можно осуществить с помощью меню *Палитра,* содержащего набор цветов, используемых при создании объектов (рис. 7.5). Различают *основной цвет,* которым рисуются контуры фигур, и цвет фона. В левой части палитры размещаются индикаторы основного цвета и цвета фона, которые отображают текущие установки (в данном случае установлен черный основной цвет и белый цвет фона). Для изменения основного цвета необходимо осуществить левый щелчок на выбранном цвете палитры, а для цвета фона — правый щелчок.

Рис. 7.5
Меню *Палитра*

Текстовые инструменты. Текстовые инструменты позволяют добавлять в рисунок текст и осуществлять его форматирование.

В растровых редакторах инструмент *Надпись* (буква **А** на панели инструментов) позволяет создавать текстовые области на рисунках. Установив курсор в любом месте текстовой области, можно произвести ввод текста.

Форматирование текста производится с помощью *панели атрибутов текста* (рис. 7.6). В редакторе Paint панель атрибутов текста добавляется (удаляется) при выбранном инструменте *Надпись* командой [Вид-Панель атрибутов текста].

Рис. 7.6. Панель атрибутов текста

В векторных редакторах тоже можно создавать текстовые области, в которые можно вводить и форматировать текст. Кроме того, для ввода надписей к рисункам можно использовать так называемые *выноски* различных форм (рис. 7.7). В векторном графическом редакторе, входящем в Word, выноска выбирается на панели *Рисование* командой [Автофигуры-Выноски].

Рис. 7.7. Выноски в векторном редакторе

Масштабирующие инструменты. В растровых графических редакторах масштабирующие инструменты позволяют увеличивать или уменьшать масштаб представления объекта на экране, но не влияют при этом на его реальные размеры. Обычно такой инструмент называется *Лупа*.

В векторных графических редакторах можно легко изменять реальные размеры объекта с помощью мыши.

Вопросы для размышления

1. С какими графическими редакторами вам приходилось работать? К какому типу (растровый или векторный) отнесится каждый из них?
2. Какой тип графического редактора (растровый или векторный) вы выберете для ретуширования отсканированной фотографии?

Практические задания

7.2. Если установить указатель мыши на инструмент растрового редактора Paint, то появится всплывающая подсказка с названием инструмента. Определить таким способом название каждого инструмента.

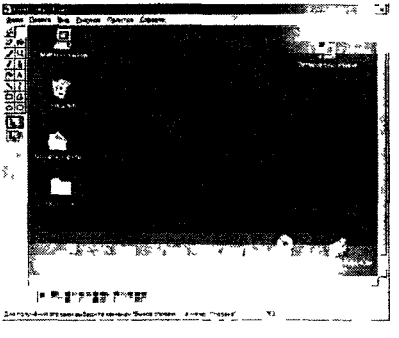
7.3. Запустить редактор Word. Добавить панель *Рисование*. Распределить имеющиеся инструменты по группам в зависимости от их назначения.

7.2.2. Редактирование изображений в растровом редакторе Paint

В качестве примера использования различных возможностей редактора Paint рассмотрим редактирование копии экрана рабочего стола Windows.

Редактирование растрового изображения

1. Поместить в буфер Windows копию экрана в тот момент, когда загружен *Рабочий стол*, для этого нажать клавишу {Print Screen}.
2. Запустить редактор Paint. Для загрузки в редактор Paint изображения из буфера ввести команду [Правка-Вставить].
   В окне редактора появится изображение *Рабочего стола*, содержащее значки и ярлыки.



3. Воспользоваться пунктом меню *Выделение* и выделить ярлык принтера в нижнем правом углу рисунка. Перетащить выделенный прямоугольный фрагмент в верхнюю часть изображения.
4. Закрасить оставшийся на месте перемещенного фрагмента белый прямоугольник цветом фона. Для этого выбрать инструмент *Выбор цветов* (*Пипетка*), установить его в любой точке фона и щелкнуть мышью. Цвет фона стал значением основного цвета. Далее выбрать инструмент *Заливка* и щелкнуть в поле белого прямоугольника.

5. Воспользоваться пунктом меню *Выделение произвольной области* для выделения значка сетевого окружения, находящегося в верхнем правом углу рисунка. Перетащить выделенный фрагмент в нижнюю часть изображения.
6. Закрасить оставшуюся на месте перемещенного фрагмента белую область с помощью инструмента *Распылитель*.
7. Создать прямоугольный контур красного цвета вокруг значка корзины. Воспользоваться для этого рисованием объекта *Прямоугольник незакрашенный*, в палитре выбрать красный основной цвет.
8. Заслонить значок *Мои документы* красным эллипсом с белым фоном. Воспользоваться для этого рисованием объекта *Эллипс закрашенный*, предварительно установив требуемые значения основного цвета и цвета фона.
9. Перечеркнуть значок *Downloads*. Установить белый цвет фона, выбрать инструмент *Ластик* и переместить его с нажатой левой клавишей мыши по значку. Аналогичный результат можно получить с использованием инструмента *Кисть* и установкой белого цвета в качестве основного.
10. Создать в центре рисунка надпись «Рабочий стол», выбрать инструмент *Надпись*, с помощью мыши создать область надписи и ввести текст. Форматирование текста можно провести, вызвав *Панель атрибутов текста* с помощью команды [Вид-Панель атрибутов текста] или контекстного меню.
11. Результатом редактирования будет являться данное изображение.

Практические задания

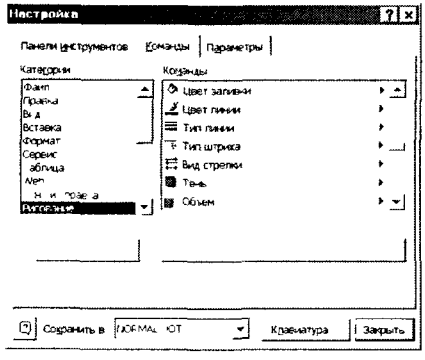
7.4. Нарисуйте шахматную доску и подпишите клетки с использованием растрового графического редактора StarOffice Image.

7.2.3. Создание изображений в векторном редакторе, входящем в состав текстового редактора Word

В качестве примера рассмотрим создание в векторном редакторе блок-схемы линейного алгоритма.

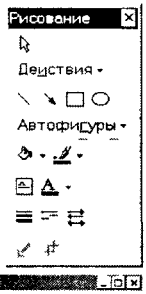
Создание векторного изображения

1. Запустить текстовый редактор Word. Командой [Вид-Панели инструментов-Рисование] вывести панель *Рисование* векторного редактора.
2. В контекстном меню панели *Рисование* выбрать пункт *Настройка*. Сформировать панель, выбрав перечень необходимых для работы команд и тематически сгруппировав их.

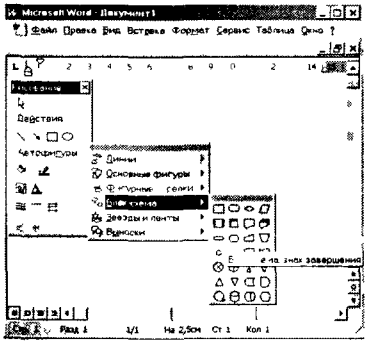


Создать, например, следующие группы на панели *Рисование*:

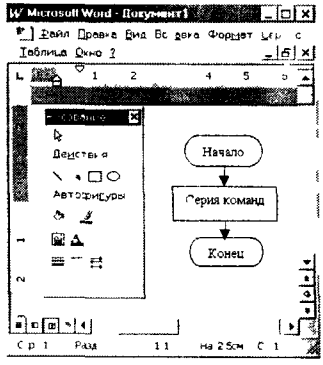
- выбор объекта и действия над объектом;
- графические примитивы и автофигуры;
- выбор цвета заливки и шрифта;
- работа с текстом;
- типы линий;
- работа с изображениями.



3. В группе *Автофигуры* выбрать пункт *Блок-схема*, содержащий различные элементы блок-схем.
Для построения блок-схемы линейного алгоритма сначала дважды выбрать *Блок-схема: знак завершения*, а потом *Блок-схема: процесс*.



4. Нарисовать элементы блок-схемы, расположить их в нужном порядке и соединить стрелочками.
5. В контекстном меню каждого из элементов блок-схемы выбрать пункт *Добавить текст* и ввести текст.
   При необходимости с помощью контекстного меню текста отформатировать текст.

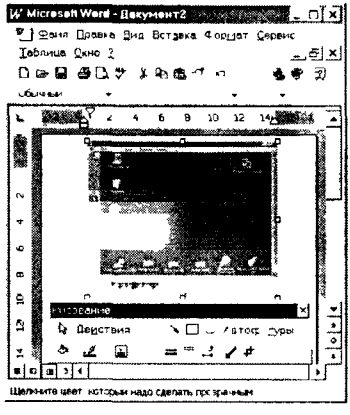


6. Сгруппировать все элементы блок-схемы в один объект, для этого нажать клавишу {*Shift*} и, не отпуская ее, последовательно активизировать все элементы мышью.
7. В результате получим единый графический объект, который можно с помощью пункта меню *Действия* изменять различными способами: изменять размер, поворачивать, сдвигать и так далее.

Графический редактор, входящий в Word, может производить некоторые преобразования и с растровыми изображениями (например, сделать прозрачным фон какой-то части рисунка или вырезать какую-либо его часть).

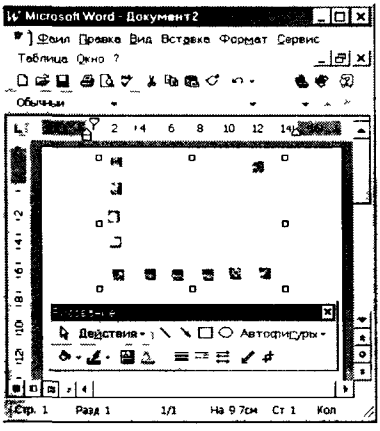
Преобразование растрового изображения

1. Запустить текстовый редактор Word.
2. Открыть новый документ и командой [Вставить-Рисунок-Из файла...] вставить в документ растровый рисунок.
   Рисунок оказывается вставленным в документ редактора Word.



3. На панели *Рисование* выбрать операцию *Установить прозрачный фон* (кнопка) и переместить указатель мыши, принявший форму, изображенную на кнопке, на любую точку фона и щелкнуть. Фон станет прозрачным.

4. На панели *Рисование* выбрать операцию *Обрезка* (кнопка). Подвести указатель мыши, принявший форму, изображенную на кнопке, по очереди к маркерам, отмечающим границы рисунка, и сместить их к центру. В результате «отрежутся» ненужные края изображения.
Обрезанное изображение с прозрачным фоном примет вид, показанный на рисунке.



Практические задания

7.5. Нарисовать блок-схемы алгоритмических конструкций «ветвление» и «цикл» с помощью векторного редактора StarOffice Draw.

7.6. Нарисовать генеалогическое дерево вашей семьи.

7.3. Система автоматизированного проектирования КОМПАС-3D

Системы автоматизированного проектирования (САПР) являются векторными графическими редакторами, предназначенными для создания чертежей.

При классическом черчении с помощью карандаша, линейки и циркуля производится построение элементов чертежа (отрезков, окружностей, прямоугольников и так далее) с точностью, которую предоставляют чертежные инструменты. Использование САПР позволяет создавать чертежи с абсолютной точностью и обеспечивает возможность реализации сквозной технологии проектирования и изготовления деталей. На основе компьютерных чертежей генерируются управляющие программы для станков с числовым программным управлением (ЧПУ), в результате по компьютерным чертежам изготавливаются высокоточные детали из металла, дерева и так далее.

7.3.1. Окно САПР КОМПАС-3D

В качестве примера системы автоматизированного проектирования рассмотрим систему КОМПАС-3D, которая позволяет создавать чертежи любого уровня сложности с полной поддержкой российских стандартов. Версия КОМПАС-3D LT специально предназначена для обучения компьютерному черчению в школах, техникумах и вузах, и право на ее использование в учебных целях предоставляется бесплатно российской компанией АСКОН.

В центре рабочего окна КОМПАС-3D размещается система координат. Положение курсора отсчитывается от начала системы координат, а текущие значения его координат *X* и *Y* отображаются в правой части *строки текущего состояния*, расположенной в нижней части окна приложения (рис. 7.8).

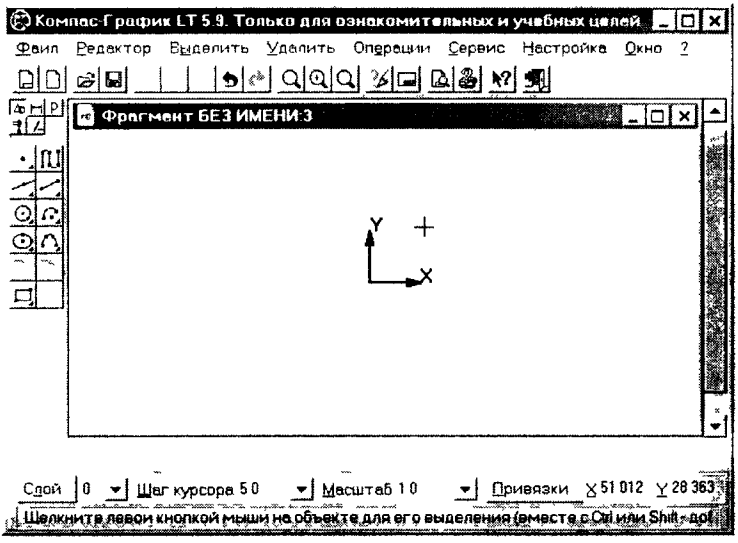


Рис. 7.8. Окно системы автоматизированного проектирования КОМПАС-3D

Создание и редактирование чертежа реализуется с помощью *инструментальной панели*, которая по умолчанию размещается в левой верхней части окна приложения. *Инструментальная панель* включает в себя пять различных рабочих панелей, каждая из которых содержит набор кнопок определенного функционального назначения и *панель переключения*, которая обеспечивает переход от одной рабочей панели к другой (рис. 7.9).

Рабочая панель *Геометрические построения* содержит кнопки, позволяющие рисовать на чертеже определенные объекты: точку, отрезок, окружность, прямоугольник и др.

Панель *Редактирование* содержит кнопки, которые позволяют вносить изменения в чертеж, производя над объектами различные операции: перемещение, копирование, масштабирование и пр.

Рис. 7.9. Инструментальная панель

Панель *Выделение* позволяет осуществлять различные варианты выделения объектов: выделение отдельных объектов, групп объектов и так далее.

Панель *Измерения* позволяет измерять расстояния (вычисляются и отображаются в миллиметрах), углы (в градусах), периметры и площади различных объектов.

Панель *Размеры и технологические обозначения* позволяет грамотно оформить чертеж: обозначить на чертеже размеры деталей, сделать надписи и так далее.

Практические задания

7.7. Запустить КОМПАС-3D и ознакомиться с основными элементами окна приложения.

7.3.2. Построение основных чертежных объектов

Выбор создаваемого чертежного объекта (точка, отрезок, окружность, прямоугольник и пр.) осуществляется с помощью панели *Геометрические построения.* После выбора объекта щелчком мыши по соответствующей кнопке появляется *строка параметров* объекта. Каждый объект обладает определенным набором параметров, которые характеризуют его размеры и положение на чертеже.

Например, после выбора на панели *Геометрические построения* кнопки *Ввод отрезка* появится строка с полями для задания значений параметров отрезка: координат его начальной (p1) и конечной (p2) точек, длины (ln), угла наклона (an) и стиля линии.

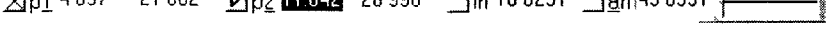


Рис. 7.10. Строка параметров отрезка

Строка параметров включает в себя *кнопки состояния полей* и сами *поля*. По внешнему виду кнопки можно судить о состоянии поля. Поле может находиться в одном из трех состояний: *фиксированном* (обозначается крестиком), *в режиме ожидания ввода* (обозначается галочкой) и просто *доступном для ввода*.

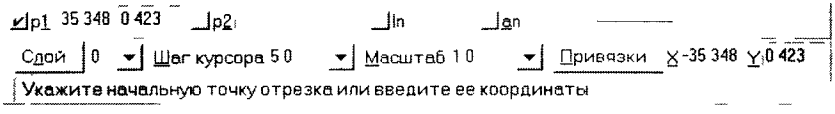
При создании и редактировании объектов работа со *строкой параметров* сводится к активизации нужных полей и вводу в них значений параметров. После ввода минимального набора значений параметров, достаточных для построения объекта (для отрезка — это координаты начальной и конечной точек), система автоматически создает объект.

Можно осуществлять *Автоматический ввод параметров*, *Ручной ввод параметров* и *Ввод параметров с использованием Геометрического калькулятора*.

Рассмотрим в качестве примера автоматического ввода параметров построение отрезка.

Построение отрезка в автоматическом режиме

1. На панели *Геометрические построения* щелкнуть по кнопке *Ввод отрезка*. Появится *строка параметров отрезка*, а в *строке сообщений* появится запрос «*Укажите начальную точку отрезка или введите ее координаты*»:



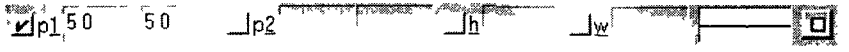
2. Установить курсор в поле чертежа на точку начала отрезка и произвести щелчок. При этом в поля координат точки p1 будут внесены значения координат указанной точки на чертеже, а в *строке параметров* символ «галочка» сменится на символ «крестик». Это означает, что введенные параметры зафиксированы.
3. Установить курсор в поле чертежа на точку p2 конца отрезка и произвести щелчок. Отрезок построен.

Рассмотрим в качестве примера ручного ввода параметров построение прямоугольника.

Построение прямоугольника в ручном режиме

1. На панели *Геометрические построения* щелкнуть по кнопке *Ввод прямоугольника*. Появится *строка параметров прямоугольника*, содержащая поле координат ле-

вой верхней (p1) и правой нижней (p2) вершин, высоты (h) и ширины (w) прямоугольника и стиль линии:

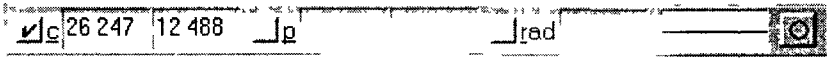


2. Активизировать поля координат точки p1 совместным нажатием на клавиатуре клавиш {*Alt*}+{*1*}. Ввести числовые значения координат, осуществляя переход между полями координат *X* и *Y* с помощью клавиши {*Tab*}.
3. Активизировать поля координат точки p2 совместным нажатием на клавиатуре клавиш {*Alt*}+{*2*}. Ввести числовые значения координат. Прямоугольник построен.

Рассмотрим теперь использование *Геометрического калькулятора*, который позволяет при рисовании объектов снимать значения для параметров с других объектов, размещенных на чертеже. Построим, например, окружность, радиус которой равен длине ранее начерченного отрезка.

Построение окружности с использованием *Геометрического калькулятора*

1. На панели *Геометрические построения* щелкнуть по кнопке *Ввод окружности*. Появится *строка параметров окружности*, содержащая поля координат центра окружности (c), точки на окружности (p), радиуса окружности (rad) и стиля линии:



2. Установить курсор в поле чертежа на точку центра окружности и произвести щелчок. В поля координат центра окружности будут внесены значения координат указанной на чертеже точки.
3. Щелкнуть правой клавишей мыши в поле *Радиус окружности* и в появившемся меню выбрать пункт *Длина кривой*. Курсор примет форму мишени.
4. Выбрать отрезок и щелкнуть левой клавишей мыши. Система автоматически измерит длину выбранного отрезка и построит окружность с таким радиусом.

Практические задания

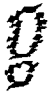
7.8. Построить окружность с использованием автоматического ввода параметров, отрезок с использованием ручного ввода параметров, прямоугольник с использованием *Геометрического калькулятора*.

Глава 8

Компьютерные презентации

8.1. Компьютерные презентации с использованием мультимедиа технологии

Мультимедиа технология. Термин «мультимедиа» — калька с английского слова multimedia, что можно перевести как «многие среды» (от multi — много и media — среда).



Мультимедиа технология позволяет одновременно использовать различные способы представления информации: числа, текст, графику, анимацию, видео и звук.

Важной особенностью мультимедиа технологии является ее *интерактивность*, то есть то, что в диалоге с компьютером пользователю отводится активная роль. Графический интерфейс мультимедийных проектов обычно содержит различные управляющие элементы (кнопки, текстовые окна и так далее).

В последнее время создано много мультимедийных программных продуктов. Это и энциклопедии из самых разных областей жизни (история, искусство, география, биология, музыка) и обучающие программы (по иностранным языкам, физике, химии) и так далее.

Компьютерные презентации. Компьютерные презентации являются одним из типов мультимедийных проектов. Компьютерные презентации часто применяются в рекламе, при выступлениях на конференциях и совещаниях, они могут также использоваться на уроках в процессе объяснения материала учителем или докладов учащихся.

В некоторых случаях презентацию запускают в автоматическом режиме, и она повествует о чем-то без участия человека. Автоматический режим презентации часто используют во время проведения различных выставок.

Что же представляет собой компьютерная презентация? Проведем аналогию с обычной книгой. Книга состоит из страниц с текстом, и презентация тоже состоит из страниц, но только электронных, которые кроме текста могут содержать также мультимедийные объекты. Электронные страницы презентации называются *слайдами*.

Книгу мы обычно читаем последовательно, просто перелистывая ее страницы. В процессе просмотра компьютерной презентации могут реализовываться различные последовательности представления слайдов. Для осуществления различных вариантов переходов между слайдами используются либо управляющие кнопки, либо гиперссылки.

Компьютерная презентация представляет собой последовательность слайдов, содержащих мультимедийные объекты. Переход между слайдами осуществляется с помощью управляющих объектов (кнопок) или гиперссылок.

Вопросы для размышления

1. В чем состоит разница между слайдами презентации и страницами книги?

8.2. Разработка презентации

Создание презентации целесообразно начинать с разработки проекта, в котором необходимо определить примерное количество слайдов в презентации и их содержание. Создадим, например, проект учебной презентации «Знакомимся с компьютером», которая будет посвящена рассмотрению устройства компьютера.

Последовательность слайдов этой презентации может быть, например, такой:

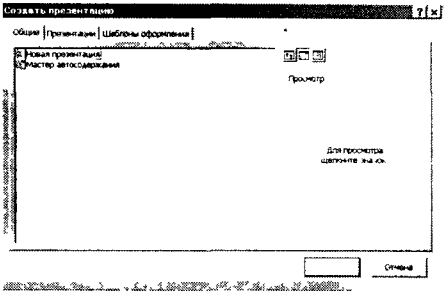
- слайд 1 «Знакомимся с компьютером»;
- слайд 2 «Структурная схема компьютера»;
- слайд 3 «Долговременная память»;
- слайд 4 «Устройства ввода».

8.2.1. Создание презентации с помощью PowerPoint

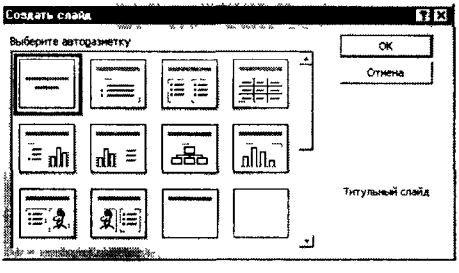
PowerPoint является офисным приложением, которое предназначено для создания презентаций. Приступим к практической реализации презентации «Знакомимся с компьютером».

Создание презентации «Знакомимся с компьютером»

1. В окне приложения PowerPoint ввести команду [Файл-Создать...]. Появится диалоговая панель *Создать презентацию*, содержащая три вкладки: *Общие, Презентации* и *Шаблоны оформления*.



2. Перейти на вкладку *Общие*, дважды щелкнуть по значку *Новая презентация*. Появится диалоговая панель *Создать слайд*.



Каждый раз при добавлении нового слайда необходимо выбрать тип *автомакета слайда*. Панель *Создать слайд* содержит 24 варианта разметки слайда. Текстовая информация на слайде может быть расположена либо в виде маркированного списка, либо в две колонки. Слайд целиком может занимать таблица или диаграмма, на слайде могут находиться текст и диаграмма, текст и графика и так далее. Большинство типов слайдов содержат также заголовки. Наконец, есть пустые заготовки слайдов с заголовком и без него.

Процедура заполнения слайда информацией одинакова для слайдов всех видов. Достаточно щелкнуть мышью в выбранной области и набрать свой текст или скопировать туда рисунок, диаграмму и пр.

Создадим титульный слайд презентации «Знакомимся с компьютером». Первый слайд презентации обычно содержит ее название и создается на основе *Титульного слайда*.

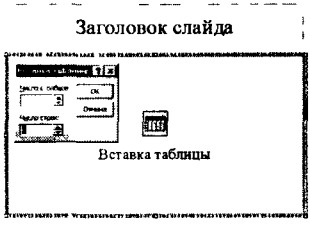
3. На диалоговой панели *Создать слайд* выбрать тип автомакета *Титульный слайд*, щелкнуть на поле заголовка и ввести текст «Знакомимся с компьютером».

Второй слайд называется «Структурная схема компьютера». Здесь мы в дальнейшем поместим рисунок структурной схемы компьютера.

4. Ввести команду [Вставка-Новый слайд...]. На диалоговой панели *Создать слайд* выбрать тип автомакета *Только заголовок*. Щелкнуть на поле заголовка и ввести текст «Структура компьютера».

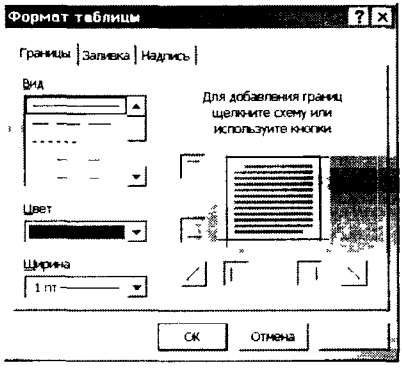
Третий слайд «Долговременная память» будет содержать таблицу из трех столбцов и четырех строк. В таблице будут содержаться названия устройств долговременной (внешней) памяти и их информационная емкость.

5. Ввести команду [Вставка-Новый слайд.....]. На диалоговой панели *Создать слайд* выбрать тип автомакета *Таблица*. Выбрать количество столбцов и строк таблицы. Ввести заголовок и заполнить таблицу.



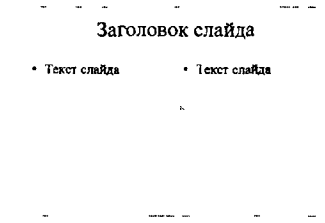
PowerPoint предоставляет возможность красиво оформить внешний вид таблицы.

6. Ввести команду [Формат-Таблица...]. На появившейся диалоговой панели *Формат таблицы* на вкладках *Границы*, *Заливка*, *Надпись* можно задать детали оформления таблицы.



Четвертый слайд «Устройства ввода» будет содержать названия устройств ввода и их изображения, которые будут размещены в две колонки.

7. Ввести команду [Вставка-Новый слайд...].
   На диалоговой панели *Создать слайд* выбрать тип автомакета *Текст в две колонки*.
   Ввести заголовок и текст.



8.2.2. Рисунки и графические примитивы на слайдах

Рисунки на слайдах. Рисунок для слайда можно создать с помощью графического редактора, а затем поместить на слайд командой [Вставка-Рисунок-Из файла...].

Однако проще воспользоваться коллекцией рисунков, которая имеется в Microsoft Office. Рисунки из коллекции добавляют с помощью команды [Вставка-Рисунок-Картинки...].

На появившейся диалоговой панели Microsoft Clip Gallery перед нами открывается коллекция рисунков Clip Art, в которой мы можем выбрать нужный рисунок для слайда (рис. 8.1).

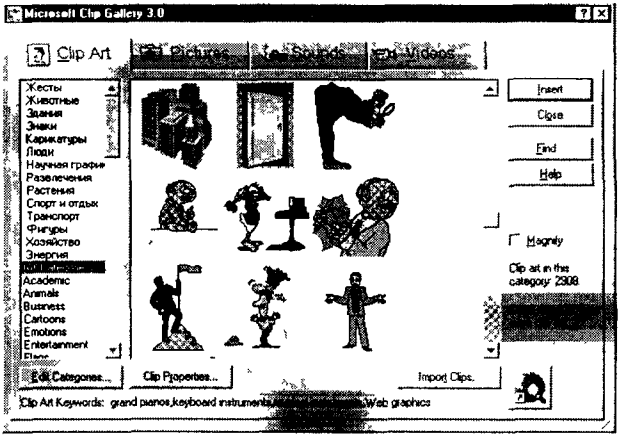
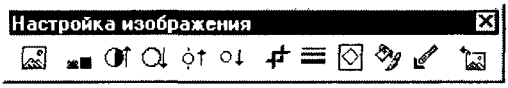


Рис. 8.1 Коллекция рисунков Clip Art

Картинки из коллекции Clip Art можно изменять. Для этого используется панель *Настройка изображения* (рис. 8.2). Эта панель появляется на экране после выделения какого-либо графического объекта или вызывается командой [Вид-Панели инструментов-Настройка изображения].

Рис. 8.2
Панель
Настройка изображения



Интересный зрительный образ можно создать из нескольких картинок Clip Art. Создадим для титульного слайда изображение, состоящее из двух картинок: человека и компьютера. Одну картинку (человека) увеличим, а другую (компьютер) уменьшим (рис. 8.3).

Рис. 8.3
Иллюстрация, составленная из двух картинок Clip Art

Графические примитивы на слайдах. На слайдах можно разместить различные графические примитивы (линии, прямоугольники, фигурные стрелки и пр.). Для этого необходимо воспользоваться панелью *Рисование*, которая находится обычно в нижней части окна приложения (рис. 8.4).

Рис. 8.4. Панель *Рисование*

Нужную фигуру необходимо выбрать, щелкнув по соответствующей кнопке на панели *Рисование*. Затем следует разместить фигуру в определенном месте слайда. Если создается несколько одинаковых фигур, целесообразно воспользоваться стандартными операциями: *Копирование — Вставка*.

Чтобы подписать фигуру, ее надо выделить (щелкнув по ней мышью) и набрать название на клавиатуре.

С помощью панели *Рисование* можно изменить цвет фигуры или надписи, толщину и цвет контурной линии, добавить к фигуре тень или объем. Для этого фигуру или надпись выделяют, а затем указывают выбранное оформление.

С помощью графических примитивов можно создавать различные схемы, например схему «Структурная схема компьютера» для слайда 2 (рис. 8.5).

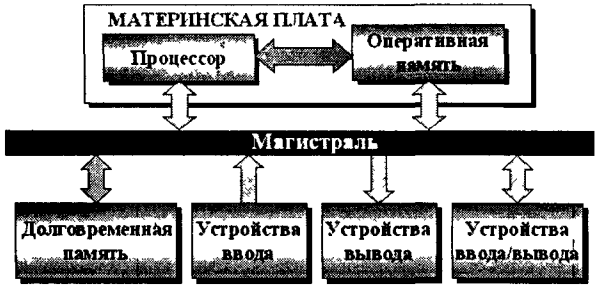


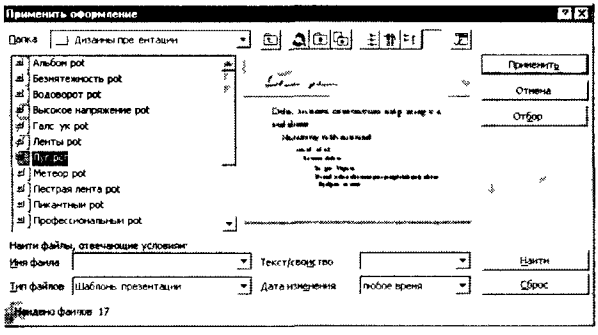
Рис. 8.5. Слайд 2 «Структурная схема компьютера»

8.2.3. Выбор дизайна презентации

Теперь можно выбрать дизайн презентации из коллекции, которая имеется в PowerPoint. Для этого необходимо ввести команду [Формат-Применить оформление...].

На появившейся диалоговой панели *Применить оформление* в раскрывающемся списке можно выбирать различные стили дизайна и просматривать их в окне просмотра (рис. 8.6). Выбрав подходящий, например *Луг*, надо нажать кнопку *Применить*. Все слайды разработанной презентации получат выбранный дизайн.

Рис. 8.6
Выбор дизайна презентации



8.2.4. Редактирование и сортировка слайдов

PowerPoint позволяет редактировать каждый слайд по отдельности в режиме *Слайды*, а также просматривать все слайды одновременно и сортировать их в режиме *Сортировщик слайдов*. Для переключения режимов просмотра можно использовать пункт *Вид* меню приложения или панель кнопок (рис. 8.7), которая располагается в нижнем левом углу экрана.

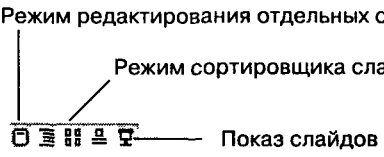


Рис. 8.7 Панель кнопок, переключающих режимы просмотра и сортировки

После выбора режима *Сортировщик слайдов* в окне приложения появятся все слайды созданной презентации (рис. 8.8). В этом режиме удобно редактировать последовательность слайдов презентации. Слайд можно выделить, скопировать в буфер, вырезать, вставить из буфера или удалить. Слайды также легко поменять местами, перетащив их мышью на нужное место.

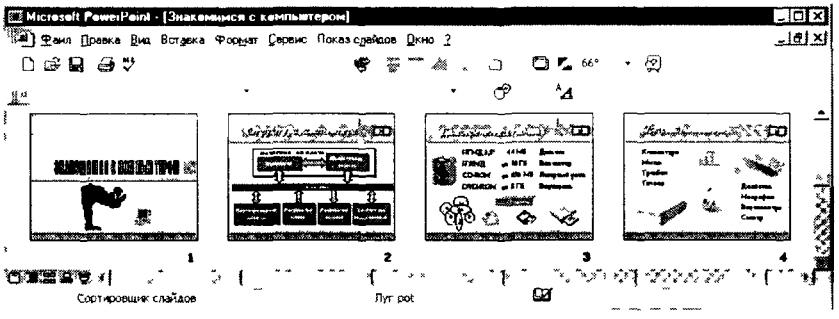


Рис. 8.8. Режим *Сортировщик слайдов*

Практические задания

8.1. Создать дополнительные слайды презентации «Знакомимся с компьютером»: слайд 5 «Устройства вывода», слайд 6 «Сетевые устройства», шутливый слайд 7 «Ну вот и познакомились...».

8.2. Найти в коллекции рисунков или в Интернете изображения устройств компьютера и вставить их в слайды презентации «Знакомимся с компьютером».

8.3. Разработать презентацию «Глобальная компьютерная сеть Интернет».

8.3. Использование анимации в презентации

Анимация в процессе смены слайдов. PowerPoint позволяет «оживить» демонстрацию презентации с помощью анимации. Можно создать эффекты анимации при смене одного слайда следующим.

Для настройки перехода от одного слайда к другому необходимо выделить слайд и ввести команду [Показ слайдов-Переход слайда...]. На появившейся диалоговой панели *Переход слайда* (рис. 8.9) с помощью раскрывающихся списков и установки флажков можно указать, какой анимационный эффект будет использоваться при смене слайдов, какими звуками это будет сопровождаться, что будет вызывать смену кадров — щелчок мыши или истекший интервал времени, и так далее.

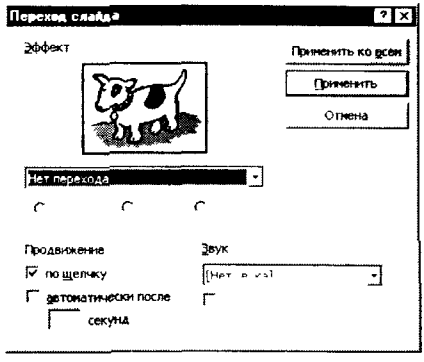


Рис. 8.9. Панель настройки перехода от одного слайда к другому

Например, в раскрывающемся списке *Эффект* можно выбрать один из типов анимационных эффектов, который будет реализовываться в процессе перехода от слайда к слайду (рис. 8.10).

Рис. 8.10. Примеры некоторых видов анимационных эффектов

В раскрывающемся списке *Звук* можно выбрать звук, которым будет сопровождаться переход: *Аплодисменты, Колокольчики, Пишущая машинка* и так далее. Можно установить любой другой звук, выбрав звуковой файл.

Выбранные настройки можно применить как к одному текущему слайду, так и сразу ко всем слайдам презентации.

Анимация объектов слайда. Любой объект, находящийся на слайде, можно заставить возникнуть на экране необычно: проявиться на экране, вылететь сбоку, развернуться до заданного размера, уменьшиться, вспыхнуть, вращаться и так далее. Текст может появляться целиком, по словам или даже по отдельным буквам.

Для установки значений параметров анимации объекта его необходимо выделить, а затем в контекстном меню выбрать пункт *Настройка анимации*. Появится диалоговая панель *Настройка анимации* (рис. 8.11).

На диалоговой панели в верхнем окне *Порядок анимации* перечислены объекты данного слайда. После выбора одного из них можно приступить к настройке анимационных эффектов.

Вкладка *Эффекты* позволяет с помощью двух раскрывающихся списков установить тип анимационного процесса при появлении объекта на слайде и звук, которым будет сопровождаться заданное действие, и так далее.

Если выделенным объектом является текст, то в раскрывающемся списке *Появление текста* задается способ появления текста: *Все вместе*, *По словам*, *По буквам*.

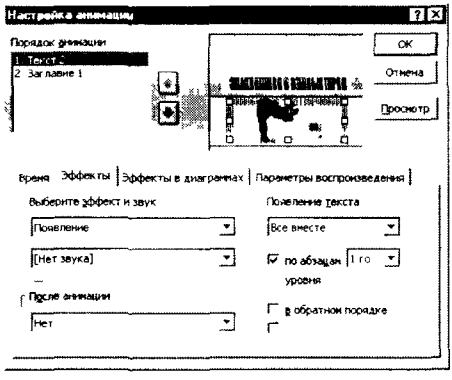


Рис. 8.11. Настройка анимации объектов слайда

Практические задания

8.4. Установить для каждого слайда презентации различные анимационные эффекты при переходе к другому слайду.

8.5. Настроить анимацию для слайда 2 «Структурная схема компьютера» таким образом, чтобы блоки схемы могли появляться последовательно по щелчку мыши, иллюстрируя доклад рассказчика о структуре компьютера.

8.4. Интерактивная презентация

8.4.1. Переходы между слайдами

Мультимедийная презентация создана, слайды содержат красиво оформленный текст, иллюстрации, звуковые эффекты и даже анимацию. Теперь необходимо сделать презентацию интерактивной. Для этого необходимо в процессе демонстрации презентации иметь возможность изменять последовательность предъявления слайдов.

Существуют два различных способа создания переходов. Первый способ состоит в создании *гиперссылок* на другие слайды или, в общем случае, на другие объекты (документы на локальном компьютере и Web-страницы в Интернет).

9.4. Гипертекст

Второй способ состоит в размещении на слайдах *управляющих элементов* (например, *Кнопок*). Если активизировать кнопку (щелкнуть мышью), то произойдет некоторое событие (в данном случае переход на другой слайд).

Рассмотрим в качестве примера создание прямых переходов между слайдами в презентации «Знакомимся с компьютером», которая после выполнения вами упражнений из предыдущих параграфов должна состоять из семи слайдов:

1. Знакомимся с компьютером.
2. Структурная схема компьютера.
3. Долговременная память.
4. Устройства ввода.
5. Устройства вывода.
6. Сетевые устройства.
7. Вот и познакомились...

Схема прямых переходов презентации «Знакомимся с компьютером» будет включать в себя (рис. 8.12):

- *гиперссылки*, реализующие прямые переходы со слайда 2 «Структурная схема компьютера», «центрального» слайда презентации, на слайды: 3 — «Долговременная память», 4 — «Устройства ввода», 5 — «Устройства вывода» и 6 — «Сетевые устройства»;
- *кнопки*, реализующие возврат из вышеперечисленных слайдов (3, 4, 5, 6, 7) на «центральный» слайд 2;
- *кнопку*, реализующую переход с «центрального» слайда 2 на конец презентации (слайд 7 «Вот и познакомились...»).

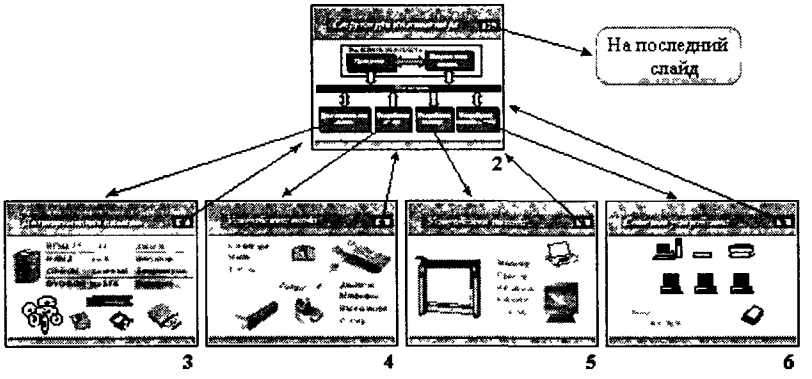
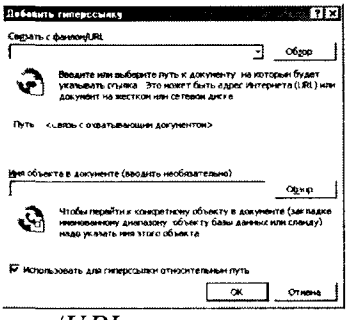


Рис. 8.12. Схема прямых переходов между слайдами в презентации «Знакомимся с компьютером»

Создание прямых переходов между слайдами в презентации «Знакомимся с компьютером»

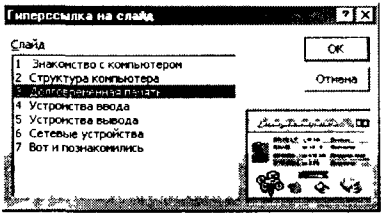
1. На слайде 2 «Структурная схема компьютера» щелчком мыши выделить блок «Долговременная память» и ввести команду [Вставка-Гиперссылка...].
   Появится диалоговая панель *Добавить гиперссылку.*



В текстовом поле *Связать с файлом/URL:* можно указать полный адрес какого-либо файла на локальном компьютере или адрес Web-страницы в Интернете. В этом случае в процессе демонстрации презентации при активизации данной ссылки в презентацию добавлялся бы внешний файл или документ.

Однако в нашем случае необходимо создать ссылку на слайд данной презентации.

2. Щелкнуть по кнопке *Обзор...* рядом с полем *Имя объекта в документе:* и в появившемся окне *Гиперссылка на слайд* выбрать слайд, на который будет осуществляться переход.



3. Повторить действия 1 и 2 для установки ссылок с блоков структурной схемы *Устройства ввода*, *Устройства вывода* и *Сетевые устройства* на соответствующие слайды презентации.

Теперь надо предусмотреть возврат со слайдов 3, 4, 5 и 6 на слайд 2 «Структурная схема компьютера». Реализуем это с помощью кнопок, которые должны быть размещены на соответствующих слайдах. Щелчок по кнопке будет приводить к переходу на слайд 2.

Сначала выберем тип кнопки (*Вперед, Назад, Возврат* и так далее).

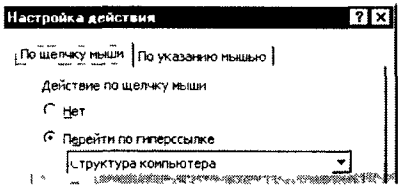
4. Ввести команду [Показ слайдов-Управляющие кнопки]. На панели *Кнопки действий* выбрать кнопку *Возврат*.

Далее необходимо выбрать для кнопки *Возврат* такое местоположение, размер и цвет, чтобы она хорошо смотрелась на слайде.

5. Изобразить с помощью мыши кнопку на слайде, подобрать цвет и размеры.

Теперь мы можем задать действия, которые будут производиться по нажатию на созданную кнопку *Возврат*.

6. В контекстном меню кнопки выбрать пункт *Настройка действия*. На появившейся одноименной панели из раскрывающегося списка выбрать нужный слайд.



Кнопки *Возврат* на всех четырех слайдах должны одинаково выглядеть и производить одинаковые действия (переход на слайд 2). Поэтому для размещения кнопки *Возврат* на оставшихся слайдах можно воспользоваться операцией *Копирование*.

Презентация хранится в каталоге \textbook\PP\ **CD-ROM**

Теперь полностью готовую мультимедийную интерактивную презентацию можно запустить на демонстрацию одним из описанных ниже способов.

8.4.2. Демонстрация презентации

Запуск демонстрации презентации может осуществляться либо командой [Вид-Показ слайдов], либо нажатием кнопки *Показ слайдов* на панели кнопок. Если делать это с помощью кнопки, то предварительно надо вызвать на экран первый слайд презентации, так как кнопка запускает демонстрацию, начиная с текущего слайда.

Для перехода от одного слайда к другому, следующему за ним, нажимают клавишу {*Enter*} или щелкают левой кнопкой мыши. Для перемещения по слайдам презентации вперед или назад можно пользоваться клавишей {*PageUp*} или {*PageDown*}.

В процессе показа слайдов указатель мыши не виден на экране, но он сразу появляется, стоит только начать перемещение мыши. Одновременно с курсором в нижнем левом углу экрана появляется изящная, почти сливающаяся с фоном кнопка . Нажатие на нее вызывает раскрывающееся меню, с помощью которого также можно управлять ходом демонстрации.

В процессе демонстрации презентации для перехода на нужный слайд можно также пользоваться управляющими кнопками и гиперссылками.

Вопросы для размышления

1. Какие существуют способы задания переходов между слайдами и чем они отличаются?

Практические задания

8.6. Создать на каждом слайде кнопки перехода на следующий слайд и возврата на предыдущий.

8.7. Создать на слайде 2 «Структурная схема компьютера» кнопку перехода на последний слайд презентации.

Глава 9

Технология обработки текстовой информации

9.1. Создание и редактирование документов

Для обработки текстовой информации на компьютере используются приложения общего назначения — текстовые редакторы. Текстовые редакторы позволяют создавать, редактировать, форматировать, сохранять и распечатывать документы.

Простые текстовые редакторы (например, стандартное приложение Windows Блокнот) позволяют редактировать текст, а также осуществлять простейшее форматирование шрифта.

Более совершенные текстовые редакторы (например, Microsoft Word и StarOffice Writer), которые называют иногда *текстовыми процессорами*, имеют широкий спектр возможностей по созданию документов (вставка списков и таблиц, средства проверки орфографии, сохранение исправлений и др.).

Для подготовки к изданию книг, журналов и газет в процессе макетирования издания используются мощные программы обработки текста — *настольные издательские системы* (например, Adobe PageMaker).

Для подготовки к публикации в Интернете Web-страниц и Web-сайтов используются специализированные приложения (например, Microsoft FrontPage).

Создание документа. Создание документа начинается с выбора *шаблона*, то есть готовой пустой заготовки документа определенного назначения (обычный документ, визитная карточка, резюме и др.). Шаблоны задают структуру документа, которую пользователь заполняет определенным содержанием.

Для создания документов со сложной структурой используются *Мастера*. Например, целесообразно использовать мастер при создании факсов, так как общепринятая форма факсов

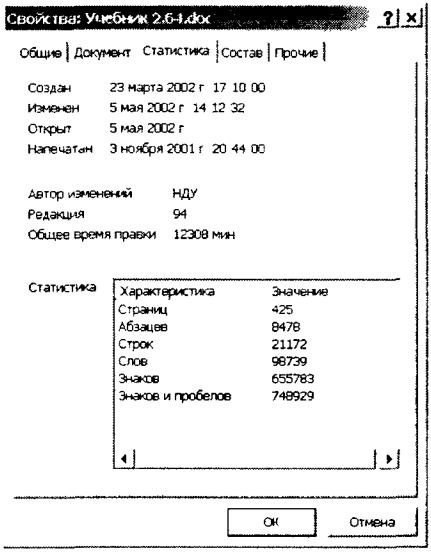
должна содержать обязательный набор правильно размещенных на странице полей (*Кому*, *От кого*, *Дата* и др.).

В процессе создания документа в текстовом редакторе пользователь вводит символы с клавиатуры.

Свойства документа. Ознакомимся со свойствами создаваемого документа.

Свойства документа

1. Запустить редактор Word. Открыть документ, например, содержащий данный учебник с помощью команды [Файл-Открыть...].
2. Ввести команду [Файл-Свойства]. Появится диалоговая панель для текущего документа *Свойства: Учебник 2.64. doc*.
   На вкладке *Статистика* ознакомиться с составом документа (количеством страниц, абзацев, строк, слов, символов и др.).

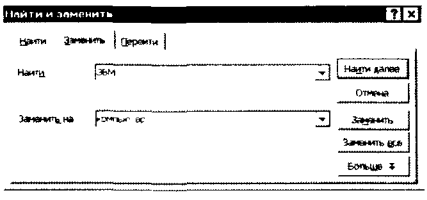


Редактирование документа. Редактирование документа производится путем копирования, перемещения или удаления выделенных символов или фрагментов текста. Копирование позволяет размножить выделенный фрагмент документа, то есть вставить его копии в указанные места документа. При перемещении выделенный фрагмент вырезается и вставляется в другое место документа, а при удалении он только вырезается.

В процессе работы над документом иногда бывает необходимо заменить одно многократно встречающееся слово на другое (например, слово «ЭВМ» на слово «компьютер»). В этом случае можно использовать функцию текстового редактора *Найти и заменить*. При вводе запросов на замену можно использовать звездочку (*), которая маскирует произвольное число символов или знак вопроса (?), который заменяет любой единичный символ.

Редактирование документа

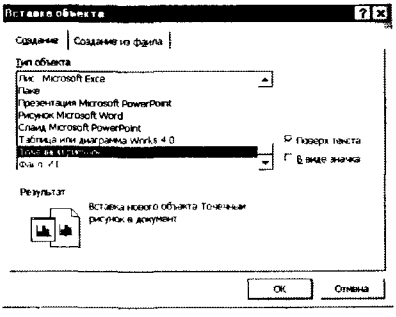
1. Ввести команду [Правка-Заменить...]. На появившейся диалоговой панели *Найти и заменить* на вкладке *Заменить* в поля *Найти:* и *Заменить на:* необходимо ввести соответствующие последовательности символов.

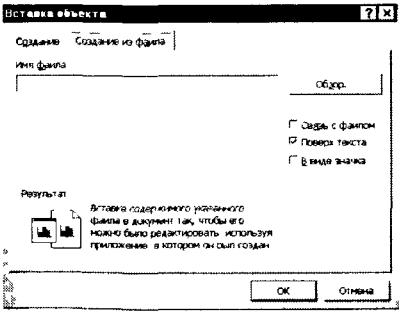


Вставка объектов в документ. Объектно-ориентированный подход позволяет реализовать *механизм встраивания и внедрения объектов* (OLE — Object Linking Embedding). Этот механизм позволяет копировать и вставлять объекты из одного приложения в другое. Например, работая с документом в текстовом редакторе Word, в него можно встроить изображения, анимацию, звук и даже видеофрагменты и таким образом из обычного текстового документа получить мультимедиа документ.

Вставка объектов в документ

1. Например, для вставки в документ рисунка необходимо ввести команду [Вставка-Объект...], появится диалоговая панель *Вставка объекта.*
   На вкладке *Создание в списке Тип объекта:* выбрать пункт *Точечный рисунок,* после чего загрузится графический редактор Paint, в котором можно создать требуемый рисунок.
2. Если мы хотим вставить в документ готовый рисунок, то на вкладке *Создание из файла* надо осуществить щелчок на кнопке *Обзор* и выбрать требуемый файл.





Проверка орфографии и синтаксиса. Для проверки орфографии (правильности написания слов) и синтаксиса (правильного построения предложений) используются специальные программные модули, которые обычно включаются в состав текстовых процессоров и редакционно-издательских систем. Такие системы содержат словари и грамматические правила для нескольких языков, что позволяет исправлять ошибки в многоязычных документах.

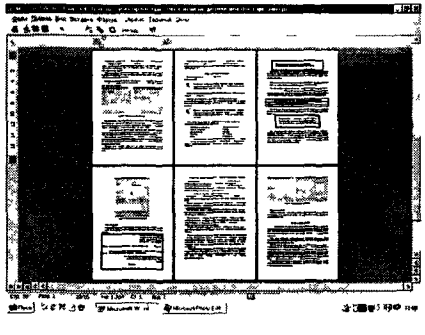
В процессе ввода текста иногда допускаются опечатки (например, вводятся ДВе прописные буквы в начале слова). В этом случае срабатывает функция *Автозамена*, которая автоматически исправляет наиболее часто встречающиеся опечатки.

Сохранение исправлений. В процессе работы над документом может участвовать несколько пользователей. Исправления, вносимые каждым из них, запоминаются и могут быть просмотрены и распечатаны. При работе над окончательной редакцией документа может быть проведено сравнение исправлений различных авторов и принят лучший вариант.

Печать документа. Перед выводом документа на печать полезно предварительно просмотреть, как будет выглядеть документ на бумаге, так как его вид может зависеть от используемого принтера.

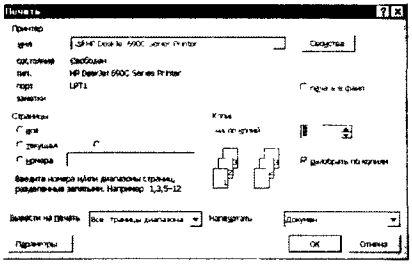
Печать документа

1. Для предварительного просмотра документа следует выбрать режим разметки страницы с помощью команды [Вид-Разметка страницы]. Вид документа в режиме *Разметка страницы* (в отличие от вида в режиме *Обычный*) позволяет создавать, форматировать и редактировать документ в том виде, в котором он будет напечатан. Масштаб просмотра документа можно изменять с помощью команды [Вид-Масштаб...].
2. Режим *Предварительного просмотра*, который задается командой [Файл-Предварительный просмотр], позволяет увидеть, как будут выглядеть в напечатанном виде сразу несколько страниц документа.



При подготовке документа к печати необходимо установить параметры печати, например номера страниц, выводимых на печать, количество копий и др.

3. Команда [Файл-Печать...] вызывает диалоговую панель *Печать*, которая позволяет выбрать принтер, число копий и номера страниц, выводимых на печать.



Вопросы для размышления

1. Влияет ли на вид напечатанного документа выбор принтера? Почему?

Практические задания

9.1. В текстовом документе произвести замену всех букв «а» на букву «о».

9.2. В текстовый документ сначала вставить готовый рисунок из файла, а затем рисунок, созданный в приложении Paint.

9.2. Различные форматы текстовых файлов (документов)

Формат файла определяет способ хранения текста в файле. Простейший формат текстового файла содержит только символы (числовые коды символов), другие же форматы содержат дополнительные управляющие числовые коды, которые обеспечивают форматирование текста.

Существуют универсальные форматы текстовых файлов, которые могут быть прочитаны большинством текстовых редакторов, и оригинальные форматы, которые используются отдельными текстовыми редакторами. Для преобразования

текстового файла из одного формата в другой используются специальные программы — программы-конверторы. В хороших текстовых редакторах такие конверторы входят в состав системы.

Рассмотрим некоторые наиболее распространенные форматы текстовых файлов.

Только текст (Text Only) (TXT). Наиболее универсальный формат. Сохраняет текст без форматирования, в текст вставляются только управляющие символы конца абзаца. Применяют этот формат для хранения документов, которые должны быть прочитаны в приложениях, работающих в различных операционных системах.

Текст в формате RTF (Rich Text Format) (RTF). Универсальный формат, который сохраняет все форматирование. Преобразует управляющие коды в команды, которые могут быть прочитаны и интерпретированы многими приложениями, в результате информационный объем файла существенно возрастает.

Документ Word (DOC). Оригинальный формат используемой в настоящее время версии Word. Полностью сохраняет форматирование. Использует 16-битную кодировку символов, что требует использования шрифтов Unicode.

Документ Word 2.0, Word 6.0/95 (DOC). Оригинальные форматы предыдущих версий редактора Word. При преобразовании из формата Word 97/2000 форматирование сохраняется не полностью.

Works 4.0 для Windows (WPS). Оригинальный формат интегрированной системы Works 4.0. При преобразовании из формата Word форматирование сохраняется не полностью.

HTML-документ (HTM, HTML). Формат хранения Web-страниц. Содержит управляющие коды (тэги) языка разметки гипертекста.

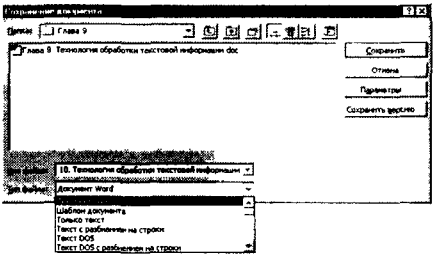
Формат Лексикон (LX). Оригинальный формат отечественного текстового редактора Лексикон.

Выбор требуемого формата текстового документа или его преобразование производится в процессе сохранения файла.

Сохранение и открытие документа в определенном формате

1. Ввести команду [Файл-Сохранить как...].
   С помощью диалоговой панели *Сохранение документа* выбрать папку для сохранения документа, в раскрывающемся списке *Имя файла:* присвоить документу имя и в

раскрывающемся списке *Тип файла:* выбрать требуемый формат.



Преобразование формата текстового документа можно также производить в процессе его открытия в редакторе.

2. Например, для того чтобы преобразовать документ, существующий в формате RTF, в формат документа Word, нужно ввести команду [Файл-Открыть...], на открывшейся диалоговой панели *Открытие документа* выбрать файл, а в окне раскрывающегося списка *Тип файла:* выбрать вариант *Текст в формате RTF*. В процессе загрузки файла будет произведено преобразование формата документа.

К сожалению, в состав текстовых редакторов не всегда входят необходимые конверторы, позволяющие импортировать и экспортировать документ из одного приложения в другое. Так, среди типов файлов, конвертируемых в Word, вы не найдете файлов Adobe PageMaker. В этих случаях необходимо либо найти конвертор для этих форматов (например, на файловых серверах Интернета), либо сохранить файл в универсальном формате, который читают оба приложения.

Вопросы для размышления

1. В каком формате нужно сохранить файл, чтобы он мог быть прочитан в других приложениях с сохранением форматирования? Без сохранения форматирования?

Практические задания

9.3. Создать текстовый документ в редакторе Word и сохранить его в таком формате, чтобы его можно было прочитать в стандартном приложении WordPad, в стандартном приложении Блокнот.

9.3. Форматирование документа

Основными объектами документа являются страница, абзац и символ. Для каждого из этих объектов необходимо задать значения параметров форматирования, которые определяют внешний вид документа.

9.3.1. Выбор параметров страницы

Любой документ состоит из страниц, поэтому в начале работы над документом необходимо задать значения параметров страницы: формат, ориентацию, размер полей и др.

При создании реферата или заявления целесообразно выбрать формат страницы А4 (21×29,7 см), который соответствует размеру стандартного листа бумаги для принтера. Для объявлений и плакатов подходит формат А3, размер которого в два раза больше стандартного листа. Наоборот, для писем можно выбрать формат А5, который в два раза меньше стандартного листа.

Существуют две возможные ориентации страницы — книжная и альбомная. Для обычных текстов чаще всего используется книжная ориентация, а для таблиц с большим количеством столбцов — альбомная (рис. 9.1).

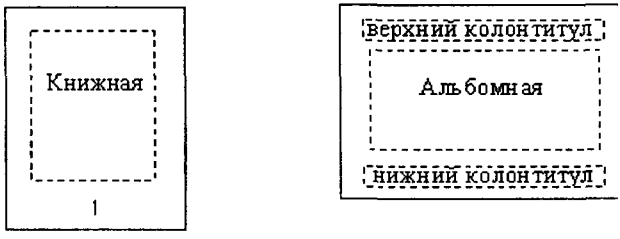
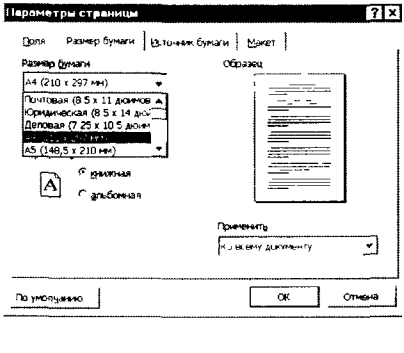


Рис. 9.1 Параметры страницы

Параметры страницы

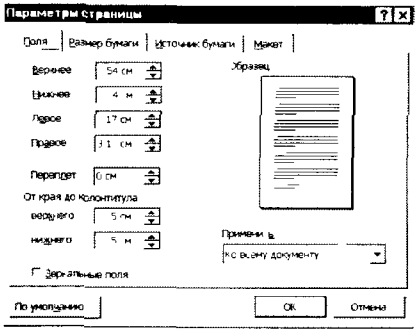
1. Ввести команду [Файл-Параметры страницы...]. Появится диалоговая панель *Параметры страницы* с четырьмя вкладками *Поля, Размер бумаги, Источник бумаги, Макет*. На вкладке *Размер бумаги* с помощью раскрывающегося списка *Размер*



бумаги необходимо выбрать используемый формат (рекомендуется формат стандартного листа А4), а также ориентацию страницы с помощью переключателя *Ориентация:* (*книжная* или *альбомная*).

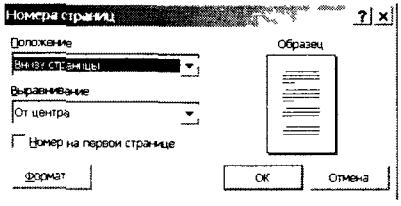
На странице можно установить требуемые размеры полей (верхнего, нижнего, правого и левого), которые определяют расстояние от краев страницы до границы текста. Для вывода на каждой странице документа одинакового текста (например, имени автора, названия документа и др.) удобно использовать верхний или нижний колонтитул. Расстояние от края страницы до колонтитула можно изменять.

2. На вкладке *Поля* задать размеры полей (расстояния от краев страницы до границы области редактирования) с помощью счетчиков *Верхнее, Нижнее, Левое* и *Правое*. Расстояния от краев страницы до колонтитулов задать с помощью счетчиков *верхнего:* и *нижнего:*.



Страницы документа требуется нумеровать, причем номера можно размещать по-разному (вверху или внизу страницы, по центру, справа или слева).

3. Ввести команду [Вставка-Номера страниц...]. На вкладке *Номера страниц* задать местоположение номера и его формат.



Практические задания

9.4. Создать текстовый документ, состоящий из страниц различного формата.

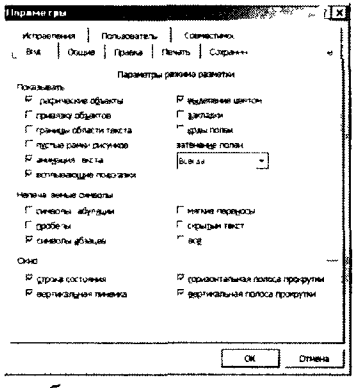
9.3.2. Форматирование абзацев

Абзац является одним из основных объектов текстового документа. Абзац с литературной точки зрения — это часть текста, представляющая собой законченный по смыслу фрагмент произведения, окончание которого служит естественной паузой для перехода к новой мысли.

В компьютерных документах абзацем считается любой текст, заканчивающийся управляющим символом (маркером) конца абзаца. Ввод конца абзаца обеспечивается нажатием клавиши {*Enter*} и отображается символом ¶, если включен режим отображения непечатаемых символов.

Форматирование абзацев

1. Для включения режима отображения непечатаемых символов абзаца ввести команду [Сервис-Параметры...] и на диалоговой панели *Параметры* на вкладке *Вид* установить флажок *символы абзацев*.



Абзац может состоять из любого набора символов, рисунков и объектов других приложений. Форматирование абзацев позволяет подготовить правильно и красиво оформленный документ.

Выравнивание абзацев. Выравнивание отражает расположение текста относительно границ полей страницы. Чаще всего используют четыре способа выравнивания абзацев:

По левому краю — левый край ровный, а правый рваный.

По центру — оба края имеют неровные очертания, однако каждая строка абзаца симметрична относительно середины.

По правому краю — правый край ровный, а левый рваный.

По ширине — оба края ровные, то есть располагаются точно по границам страницы. В этом случае последняя строка абзаца ведет себя как при левостороннем выравнивании.

Отступ первой строки (красная строка). Чаще всего абзац начинается отступом первой строки. Отступ может быть различных типов.

Положительный отступ (отступ), когда первая строка начинается правее всех остальных строк абзаца, применяется в обычном тексте.

Отрицательный отступ (выступ), когда первая строка выходит влево относительно остальных строк абзаца, применяется в словарях и определениях.

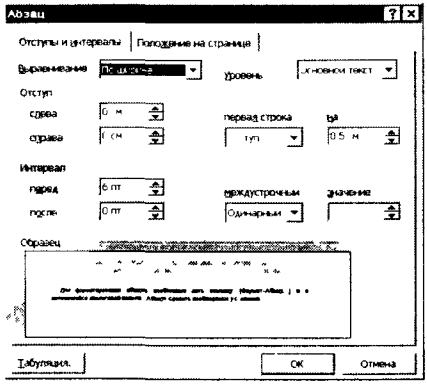
Нулевой отступ, применяется для абзацев, выровненных по центру и для обычного текста.

Отступы и интервалы. Весь абзац целиком может иметь отступы слева и справа, которые отмеряются от границ полей страницы. Так, эпиграф к художественному произведению или реквизиты адресата в заявлении имеют отступ слева, а при изготовлении углового штампа можно использовать отступ справа.

Отступ абзаца слева, все строки абзаца смещены на заданное расстояние вправо.

Отступ абзаца справа, все строки абзаца смещены на заданное расстояние влево.

2. Для выравнивания абзаца ввести команду [Формат-Абзац...] и на появившейся диалоговой панели *Абзац* сделать необходимые установки.
   На вкладке *Отступы и интервалы* для установки типа выравнивания выделенных абзацев выбрать соответствующий элемент раскрывающегося списка *Выравнивание:*.



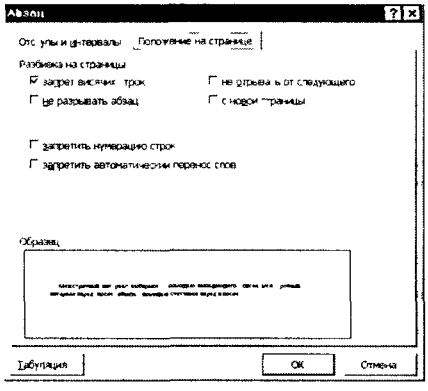
Для установки типа отступа первой строки абзаца выбрать необходимое значение в раскрывающемся списке *первая строка:* и установить конкретное числовое значение отступа с помощью счетчика *на:*.

Для задания отступа абзаца целиком от границ полей страницы выбрать нужное значение отступа с помощью счетчиков *Отступ слева:* и *справа:*.

Для того чтобы текст выходил на левое (правое) поле страницы, задать отрицательное значение отступа.

Расстояние между строками документа можно изменять, задавая различные значения междустрочных интервалов (одинарный, двойной и так далее). Для визуального отделения абзацев друг от друга можно устанавливать увеличенные интервалы между абзацами.

3. Межстрочный интервал выбирают с помощью выпадающего списка *междустрочный:*, а интервал перед (после) абзаца — с помощью счетчиков *Интервал перед:* и *после:*.
4. Вкладка *Положение на странице* позволяет установить требуемое распределение абзацев по страницам, то есть можно запретить разрывать абзац между страницами, оставлять на странице первую или последнюю (висячую) строку и так далее.
Часто бывает полезно запретить в абзаце автоматический перенос слов.



Практические задания

9.5. Набрать и отформатировать текст по образцу:

Абзац с выравниванием по ширине, отступ слева 6 см, шрифт Times New Roman, размер 12 пт, нормальный.

Абзац с выравниванием по центру, шрифт Arial, размер 14 пт, полужирный.

Абзац с выравниванием по левому краю, отступ первой строки, шрифт Courier, размер 10 пт, курсив.

9.3.3. Списки

Списки применяются для размещения в документе различных перечней. Существуют списки различных типов:

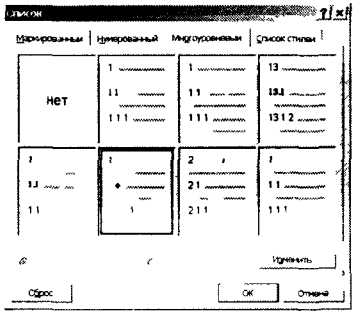
- нумерованные списки, когда элементы списка сопровождаются арабскими или римскими числами и буквами;
- маркированные списки, когда элементы списка отмечаются с помощью специальных символов-маркеров: •, ▪, ⇨ и др.

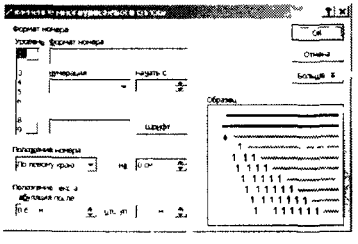
Возможно создание и вложенных списков, причем вкладываемый список может по своему типу отличаться от основного. Создадим нумерованный список из трех элементов, в первый элемент которого вложен маркированный список из двух элементов:

1. Первый нумерованный элемент списка.
   - ◆ Первый маркированный элемент списка.
   - ◆ Второй маркированный элемент списка.
2. Второй нумерованный элемент списка.
3. Третий нумерованный элемент списка.

Списки

1. Ввести команду [Формат-Список...] и на диалоговой панели *Списки* на вкладке *Многоуровневый* выбрать требуемый тип многоуровневого списка.
   Для детальной установки параметров списка щелкнуть по кнопке *Изменить*.
2. На панели *Изменение многоуровневого списка* уточнить порядок нумерации списка, отступы элементов списка, параметры шрифта, используемого для нумерации, и др.





Практические задания

9.6. Представить иерархическую модель «Компьютеры» с помощью многоуровневого списка.

9.3.4. Таблицы

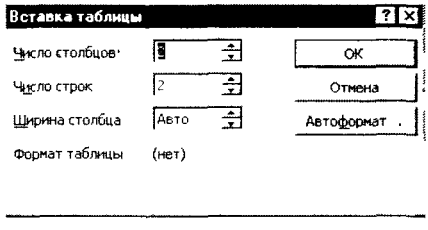
Таблица является объектом, состоящим из строк и столбцов, на пересечении которых образуются ячейки. В ячейках таблиц могут быть размещены различные данные (текст, числа и изображения). С помощью таблиц можно форматировать документы, например, расположить абзацы в несколько рядов, совместить рисунок с текстовой подписью и так далее.

При размещении в таблице чисел можно производить над ними вычисления по формулам: суммирование, умножение, поиск максимального и минимального чисел и др.

Преобразовать имеющийся текст в таблицу можно с помощью команды [Таблица-Преобразовать в таблицу...], однако часто бывает удобнее сначала создать таблицу и лишь затем заполнить ее данными. При вставке в документ таблицы можно задать необходимое количество столбцов и строк.

Таблицы

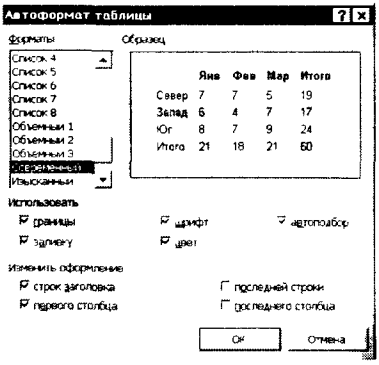
1. Вставить в документ таблицу можно при помощи команды [Таблица-Вставить таблицу...], указав на панели *Вставка таблицы* в соответствующих полях ввода число строк и столбцов создаваемой таблицы.



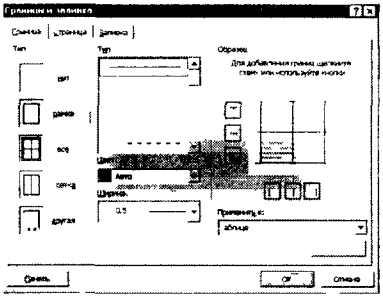
Можно подобрать подходящий дизайн таблицы, изменив тип, ширину и цвет границ ячеек, а также цвет фона ячеек. Изменение внешнего вида таблицы можно провести автоматически или вручную.

2. Автоматическое форматирование внешнего вида таблиц производится с помощью команды [Таблица-Автоформат...].

Диалоговая панель *Автоформат таблицы* предоставляет множество вариантов оформления таблицы. В списке *Форматы:* можно выбирать различные варианты и просматривать их в окне *Образец:*.



3. Форматирование таблицы вручную можно выполнить с помощью команды [Формат-Границы и заливка...]. Диалоговая панель *Границы и заливка* позволяет выбрать требуемые параметры.



На вкладке *Граница* можно задать тип границы (*Нет, Сетка, Рамка* и др.), тип и ширину линий границы. На вкладке *Заливка* можно задать цвет фона ячеек или выбрать узор.

Редактирование структуры таблиц. Изменение ширины столбцов или высоты строк реализуется с помощью мыши (перетаскиванием границ). Задать точную ширину столбца (высоту строки) можно с помощью команды [Таблица-Высота и ширина ячейки...].

Вставка или удаление строк и столбцов в имеющуюся таблицу производится с помощью команд *Вставить/удалить строку (столбец)* меню *Таблица*.

Практические задания

9.7. Создать документ, содержащий расписание уроков. Применить различные варианты форматирования таблиц (шрифт, выравнивание, границы и фон ячеек).

9.3.5. Форматирование символов

Символы являются теми основными объектами, из которых состоит документ. Символы — это буквы, цифры, пробелы, знаки пунктуации, специальные символы, такие как @, *, &. Символы можно форматировать (изменять их внешний вид).

Среди основных свойств символов можно выделить следующие: *шрифт, размер, начертание* и *цвет*.

Шрифт. Шрифт — это полный набор символов определенного начертания, включая прописные и строчные буквы, знаки препинания, специальные символы, цифры и знаки арифметических действий. Для каждого исторического периода и разных стран характерен шрифт определенного рисунка.

Каждый шрифт имеет свое название, например Times New Roman, Arial, Courier и др.

По способу представления в компьютере различаются шрифты *растровые* и *векторные*. Для представления растровых шрифтов используются методы растровой графики, и символы шрифта представляют собой группы пикселей. Растровые шрифты допускают масштабирование только с определенными коэффициентами.

В векторных шрифтах символы описываются математическими формулами и допускают произвольное масштабирование. Среди векторных шрифтов наибольшее распространение получили шрифты типа True Type.

Обычно различные символы шрифта имеют и различную ширину, например буква «Ш» шире, чем буква «А». Однако имеются и *моноширинные шрифты*, в которых ширина всех символов одинакова. Примером такого шрифта является шрифт Courier.

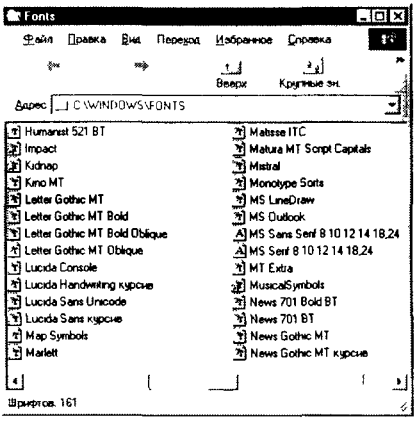
Шрифты также разделяют на две следующие группы: *шрифты с засечками* (например, Times New Roman) и *рубленые* (например, Arial). Считается, что шрифты с засечками легче воспринимаются глазом, видимо, поэтому в большинстве печатных текстов используются именно они. Рубленые шрифты используют обычно для заголовков, выделений в тексте и подписей к рисункам.

Шрифт

1. Полезно ознакомиться с набором шрифтов, установленных на компьютере. Для этого необходимо ввести команду [Настройка-Панель управления-Шрифты].

В появившемся окне *Fonts* содержится перечень установленных шрифтов, при этом шрифты True Type обозначаются значком

а растровые – значком



Размер шрифта. Единицей измерения размера шрифта является пункт (1 пт = 0,376 мм). Размеры шрифтов можно изменять в больших пределах (обычно от 1 до 1638 пунктов), причем в большинстве редакторов по умолчанию используется шрифт размером 10 пт. Ниже приведены примеры представления текста при различных размерах шрифта.

Шрифт размером 16 пт.
Шрифт размером 12 пт.
Шрифт размером 8 пт.

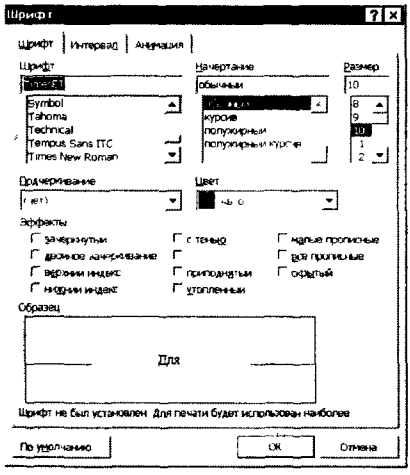
Начертание и вид символов. Кроме нормального (обычного) начертания символов обычно применяют **полужирное**, *курсивное*, ***полужирное курсивное***.

Можно установить дополнительные параметры форматирования символов: подчеркивание символов различными типами линий, изменение вида символов (верхний индекс, нижний индекс, ~~зачеркнутый~~), изменение расстояния между символами (разреженный, уплотненный) и др.

Цвет символов. Если планируется многоцветная печать документа, то для различных групп символов можно задать различные цвета, выбранные из предлагаемой текстовым редактором палитры.

2. Для форматирования шрифта необходимо ввести команду [Формат-Шрифт...], которая открывает диалоговую панель *Шрифт*.

На вкладке *Шрифт* можно с помощью раскрывающихся списков выбрать шрифт, размер, начертание, цвет символов, варианты подчеркивания. Кроме того, можно установить с помощью группы флажков *Эффекты:* дополнительные параметры форматирования символов: *верхний индекс, нижний индекс* и так далее.



Практические задания

9.8. Создать документ, содержащий десять строк, записанных различными шрифтами.

9.4. Гипертекст

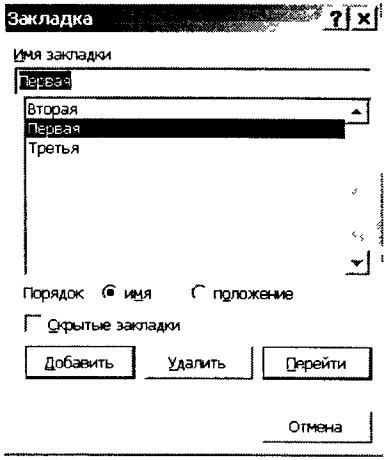
Для отображения в «плоском» тексте смысловых связей между основными разделами или понятиями можно использовать *гипертекст*. Гипертекст позволяет структурировать документ путем выделения в нем слов-ссылок (гиперссылок). При активизации гиперссылки (например, с помощью щелчка мышью) происходит переход на фрагмент текста, заданный в ссылке.

Гиперссылка состоит из двух частей: *указателя ссылки* и *адресной части ссылки*. Указатель ссылки — это объект (фрагмент текста или рисунок), который визуально выделяется в документе (обычно синим цветом и подчеркиванием). Адресная часть гиперссылки представляет собой название закладки в документе, на который указывает ссылка. Закладка — это элемент документа, которому присвоено уникальное имя.

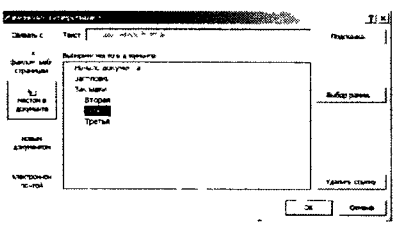
В качестве примера гипертекстового документа создадим текст, содержащий гиперссылки на три закладки, которые, в свою очередь, являются гиперссылками на начало текста.

Гипертекст

1. Создать документ, содержащий обычный текст.
   Для создания закладки выделить фрагмент текста, которому следует назначить закладку.
   Ввести команду [Вставка-Закладка...]. В поле *Имя закладки* ввести имя закладки, которое должно начинаться с буквы. Щелкнуть по кнопке *Добавить*.



2. Для создания гиперссылки выделить фрагмент текста, который будет указателем гиперссылки.
   Ввести команду [Вставка-Гиперссылка...]. На диалоговой панели *Вставка гиперссылки* в окне выбрать имя закладки. Щелкнуть по кнопке *ОК*.
3. Повторить процедуру для создания еще двух гиперссылок на закладки и трех гиперссылок с закладок на начало документа.
   Гипертекстовый документ создан.



Переход на первую закладку
Переход на вторую закладку
Переход на третью закладку

Первая закладка

Вторая закладка

Третья закладка

В качестве *указателей ссылок* и *закладок* могут использоваться не только фрагменты текста, но и графические изображения, поэтому такие структуры иногда называют *гипермедиа*. Кроме того, гипертекстовые структуры могут распространяться на документы различных типов. В Интернете они образуют Всемирную паутину, связывающую Web-страницы на сотнях миллионов серверов в единое целое.

12.9. Всемирная паутина

Таким образом, гипертекстовые структуры являются фактически структурами данных, так как в них могут использоваться объекты различных типов: фрагменты документов, управляющие элементы и др.

Вопросы для размышления

1. Что такое гипертекст?

Практические задания

9.9. Создать документ, содержащий гиперссылки на закладки и на другие документы.

9.5. Компьютерные словари и системы машинного перевода текстов

Установить компьютерный словарь EDictionary CD-ROM ☞

Компьютерные словари. Словари необходимы для перевода текстов с одного языка на другой. Первые словари были созданы около 5 тысяч лет назад в Шумере и представляли собой глиняные таблички, разделенные на две части. В одной части записывалось слово на шумерском языке, а в другой — аналогичное по значению слово на другом языке, иногда с краткими пояснениями.

Современные словари построены по такому же принципу. В настоящее время существуют тысячи словарей для перевода между сотнями языков (англо-русский, немецко-французский и так далее), причем каждый из них может содержать десятки тысяч слов. В бумажном варианте словарь представляет собой толстую книгу объемом в сотни страниц, где поиск нужного слова является достаточно трудоемким процессом.

Компьютерные словари могут содержать переводы на разные языки сотен тысяч слов и словочетаний, а также предоставляют пользователю дополнительные возможности.

Во-первых, компьютерные словари могут являться многоязычными, так как дают пользователю возможность выбрать языки и направление перевода (например, англо-русский, испано-русский и так далее).

Во-вторых, компьютерные словари могут кроме основного словаря общеупотребительных слов содержать десятки специализированных словарей по областям знаний (техника, медицина, информатика и др.).

В-третьих, компьютерные словари обеспечивают быстрый поиск словарных статей: «быстрый набор», когда в процессе набора слова возникает список похожих слов; доступ к часто используемым словам по закладкам; возможность ввода словосочетаний и др.

В-четвертых, компьютерные словари могут являться мультимедийными, то есть предоставлять пользователю возможность прослушивания слов в исполнении дикторов, носителей языка.

Среди российских словарей следует выделить словарь Lingvo, который содержит более 1,2 миллиона слов и словосочетаний, систему электронных словарей «Контекст» и словарь «Мультилекс», который базируется на использовании лучших академических словарей.

Установить on-line многоязычный переводчик Magic Translator CD-ROM

Системы машинного перевода. Происходящая в настоящее время глобализация нашего мира приводит к необходимости обмена документами между людьми и организациями, находящимися в разных странах мира и говорящими на различных языках.

В этих условиях использование традиционной технологии перевода «вручную» тормозит развитие межнациональных контактов. Перевод многостраничной документации вручную требует длительного времени и высокой оплаты труда переводчиков. Перевод полученного по электронной почте письма или просматриваемой в браузере Web-страницы необходимо осуществить немедленно, и нет возможности и времени пригласить переводчика.

Системы машинного перевода позволяют решить эти проблемы. Они, с одной стороны, способны переводить многостраничные документы с высокой скоростью (одна страница в секунду) и, с другой стороны, переводить Web-страницы «на лету», в режиме реального времени. Лучшими среди российских систем машинного перевода считаются PROMT и «Сократ».

Системы машинного перевода осуществляют перевод текстов, основываясь на формальном «знании» языка (синтаксиса языка — правил построения предложений, правил

словообразования) и использовании словарей. Программа-переводчик сначала анализирует текст на одном языке, а затем конструирует этот текст на другом языке.

Современные системы машинного перевода позволяют достаточно качественно переводить техническую документацию, деловую переписку и другие специализированные тексты. Однако они неприменимы для перевода художественных произведений, так как не способны адекватно переводить метафоры, аллегории и другие элементы художественного творчества человека.

Практические задания

9.10. С помощью компьютерного словаря перевести англоязычные термины, использующиеся в учебнике.

9.11. Соединиться с Интернетом и с помощью on-line многоязычного переводчика перевести текст.

9.6. Системы оптического распознавания документов

Установить систему оптического распознавания FineReader CD-ROM 

Системы оптического распознавания символов. При создании электронных библиотек и архивов путем перевода книг и документов в цифровой компьютерный формат, при переходе предприятий от бумажного к электронному документообороту, при необходимости отредактировать полученный по факсу документ используются *системы оптического распознавания символов*.

С помощью сканера достаточно просто получить изображение страницы текста в графическом файле. Однако для получения документа в формате текстового файла необходимо провести распознавание текста, то есть преобразовать элементы графического изображения в последовательности текстовых символов.

Сначала необходимо распознать структуру размещения текста на странице: выделить колонки, таблицы, изображения и так далее. Далее выделенные текстовые фрагменты графического изображения страницы необходимо преобразовать в текст.

Если исходный документ имеет типографское качество (достаточно крупный шрифт, отсутствие плохо напечатанных символов или исправлений), то задача распознавания решается методом сравнения с растровым шаблоном (рис. 9.2). Сначала растровое изображение страницы разделяется на изображения отдельных символов. Затем каждый из них последовательно накладывается на шаблоны символов, имеющихся в памяти системы, и выбирается шаблон с наименьшим количеством отличных от входного изображения точек.

Рис. 9.2. Растровые шаблоны символов

При распознавании документов с низким качеством печати (машинописный текст, факс и так далее) используется метод распознавания символов по наличию в них определенных структурных элементов (отрезков, колец, дуг и др.).

Любой символ можно описать через набор значений параметров, определяющих взаимное расположение его элементов. Например, буква «Н» и буква «И» состоят из трех отрезков, два из которых расположены параллельно друг другу, а третий соединяет эти отрезки. Различие между данными буквами — в величине углов, которые образует третий отрезок с двумя другими.

При распознавании структурным методом в искаженном символьном изображении выделяются характерные детали и сравниваются со структурными шаблонами символов. В результате выбирается тот символ, для которого совокупность всех структурных элементов и их расположение больше всего соответствует распознаваемому символу.

Наиболее распространенные системы оптического распознавания символов FineReader и CuneiForm используют как растровый, так и структурный методы распознавания. Кроме того, эти системы являются «самообучающимися» (для каждого конкретного документа они создают соответствующий набор шаблонов символов) и поэтому скорость и качество распознавания многостраничного документа постепенно возрастают.

Системы оптического распознавания форм. При заполнении налоговых деклараций, при проведении переписей населения и так далее используются различного вида бланки с полями. Рукопечатные тексты (данные вводятся в поля печатными буквами от руки) распознаются с помощью *систем оптического распознавания форм* и вносятся в компьютерные базы данных.

Сложность состоит в том, что необходимо распознавать написанные от руки символы, довольно сильно различающиеся у разных людей. Кроме того, система должна определить, к какому полю относится распознаваемый текст.

Системы распознавания рукописного текста. С появлением первого карманного компьютера Newton фирмы Apple в 1990 году начали создаваться *системы распознавания рукописного текста*. Такие системы преобразуют текст, написанный на экране карманного компьютера специальной ручкой, в текстовый компьютерный документ.

Практические задания

9.12. Отсканировать документ и перевести его в текстовый формат с помощью системы оптического распознавания.

Глава 10

Технология обработки числовых данных

10.1. Электронные калькуляторы

Установить электронный калькулятор NumLock Calculator — CD-ROM

Электронные калькуляторы являются специализированными программными приложениями, предназначенными для произведения вычислений. Электронные калькуляторы по своим функциональным возможностям соответствуют аппаратным микрокалькуляторам.

Аппаратные микрокалькуляторы могут существенно различаться по своим возможностям и областям применения. Простые микрокалькуляторы позволяют осуществлять только арифметические операции над числами и используются в быту. Инженерные микрокалькуляторы позволяют также вычислять значения различных функций (sin, cos и др.) и используются в процессе обучения и для инженерных расчетов; программистские микрокалькуляторы позволяют проводить вычисления в различных системах счисления и другие операции.

Электронные калькуляторы гораздо удобнее, так как могут обладать возможностями всех вышеперечисленных типов аппаратных микрокалькуляторов.

Электронный Калькулятор является стандартным приложением операционной системы Windows. Одним из удачных электронных калькуляторов является NumLock Calculator, в процессе работы с которым легко выбрать требуемый тип калькулятора (рис. 10.1).

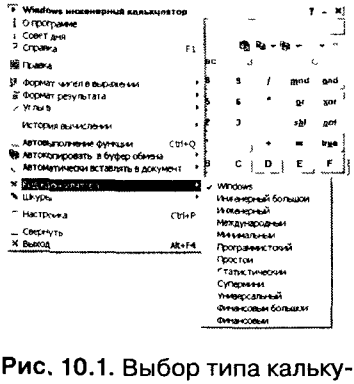


Рис. 10.1. Выбор типа калькулятора в NumLock Calculator

С помощью электронных калькуляторов можно:

- производить арифметические действия над целыми и дробными числами;
- переводить числа из одной системы счисления в другую (например, из десятичной в двоичную);
- вычислять значения математических функций (например, x^2, $1/x$);
- вычислять значения статистических функций (например, среднее арифметическое заданных чисел);
- вычислять значения финансовых функций (например, вычислять сумму банковского вклада с заданным процентом) и др.

Электронные калькуляторы позволяют проводить сложные многоступенчатые вычисления с записью промежуточных результатов в ячейки памяти калькулятора. По мере необходимости такие результаты можно извлекать из памяти и использовать в дальнейших вычислениях.

Электронные калькуляторы позволяют обмениваться числовыми данными с другими приложениями с использованием *буфера обмена* операционной системы.

Практические задания

10.1. С помощью электронного калькулятора проверить правильность выполнения заданий к главе 2 по переводу чисел из одной системы счисления в другую и выполнению арифметических действий в различных системах счисления.

10.2. Электронные таблицы

Установить офисный пакет StarOffice, ***в состав которого входят электронные таблицы*** StarCalc **CD-ROM** 

Электронные таблицы позволяют обрабатывать большие массивы числовых данных, например результаты экспериментов, статистические данные и так далее. Наибольшее распространение получили электронные таблицы Microsoft Excel и StarCalc.

Электронная таблица — это работающее в диалоговом режиме приложение, хранящее и обрабатывающее данные в прямоугольных таблицах.

Электронная таблица состоит из *столбцов* и *строк*. Заголовки столбцов обозначаются буквами или сочетаниями букв (A, C, AB и т. п.), заголовки строк — числами (1, 2, 3 и далее). *Ячейка* — место пересечения столбца и строки.

Каждая ячейка таблицы имеет свой собственный адрес. *Адрес ячейки электронной таблицы* составляется из заголовка столбца и заголовка строки, например A1, B5, E3. Ячейка, с которой производятся какие-то действия, выделяется рамкой и называется *активной*. На рис. 10.2 активной ячейкой является ячейка C3.

	A	B	C	D	E
1					
2					
3					
4					
5					

Рис. 10.2. Структура электронных таблиц

Электронные таблицы, с которыми работает пользователь в приложении, называются *рабочими листами*. Можно вводить и изменять данные одновременно на нескольких рабочих листах, а также выполнять вычисления на основе данных из нескольких листов. Документы электронных таблиц могут включать несколько рабочих листов и называются *рабочими книгами*.

Основные типы и форматы данных. В работе с электронными таблицами можно выделить три основных типа данных: *число, текст* и *формула*. В зависимости от решаемой задачи возникает необходимость применять различные форматы представления данных. В каждом конкретном случае важно выбрать наиболее подходящий формат.

Для представления чисел по умолчанию электронные таблицы используют *числовой* формат, который отображает два десятичных знака после запятой (например, 195,20).

Экспоненциальный формат применяется, если число, содержащее большое количество разрядов, не умещается в ячейке (например, число 2 000 000 000 в экспоненциальном формате будет записано в следующем виде: 2,00E+09).

По умолчанию числа выравниваются в ячейке по правому краю. Это объясняется тем, что при размещении чисел друг под другом (в столбце таблицы) удобно иметь выравнивание по разрядам (единицы под единицами, десятки под десятками и так далее).

Текстом в электронных таблицах является последовательность символов, состоящая из букв, цифр и пробелов, например, запись «32 Мбайт» является текстовой. По умолчанию текст выравнивается в ячейке по левому краю. Это объясняется традиционным способом письма (слева направо).

Формула должна начинаться со знака равенства и может включать в себя числа, имена ячеек, функции и знаки математических операций. В формулу не может входить текст.

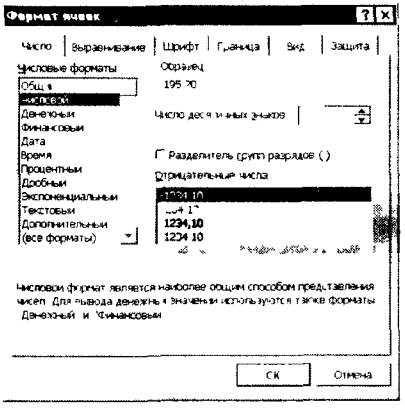
Например, формула =A1+B2 обеспечивает сложение чисел, хранящихся в ячейках A1 и B2, а формула =A1*5 — умножение числа, хранящегося в ячейке A1, на 5.

При вводе формулы в ячейке отображается не сама формула, а результат вычислений по этой формуле. При изменении исходных значений, входящих в формулу, результат пересчитывается немедленно.

Для представления данных можно использовать также специализированные форматы: *денежный* формат (12000,00р.) удобен для бухгалтерских расчетов, форматы *дата* и *время* позволяют хранить значения временны́х данных (15.01.2002 17:45:10).

Выбор формата данных

1. Запустить приложение Excel.
2. Выделить ячейку и ввести команду [Формат-Ячейки...].
3. На диалоговой панели *Формат ячеек* выбрать вкладку *Число*.
4. В списке *Числовые форматы:* выбрать наиболее подходящий формат.
   С помощью счетчика *Число десятичных знаков:* установить необходимое количество знаков после запятой.
   В поле *Отрицательные числа:* выбрать форму представления отрицательных чисел.



Относительные и абсолютные ссылки. В формулах используются ссылки на адреса ячеек. Существуют два основных типа ссылок: *относительные* и *абсолютные*. Различия между

относительными и абсолютными ссылками проявляются при копировании формулы из активной ячейки в другую ячейку.

Относительные ссылки в формулах используются для указания адреса ячейки, вычисляемого относительно ячейки, в которой находится формула.

При перемещении или копировании формулы из активной ячейки относительные ссылки автоматически обновляются в зависимости от нового положения формулы. Относительные ссылки имеют следующий вид: A1, B3.

При копировании формулы, содержащей только относительные ссылки, из ячейки C1 в ячейку D2 обозначения столбцов и строк в формуле изменятся на один шаг вправо и вниз (рис. 10.3).

	A	B	C	D	E
1			=A1*B1		
2				=B2*C2	
3			=$A$1*$B$1		
4				=$A$1*$B$1	
5					

Рис. 10.3. Относительные и абсолютные ссылки

Абсолютные ссылки в формулах используются для указания фиксированного адреса ячейки. При перемещении или копировании формулы абсолютные ссылки не изменяются. В абсолютных ссылках перед неизменяемыми значениями адреса ячейки ставится знак доллара (например, $A$1).

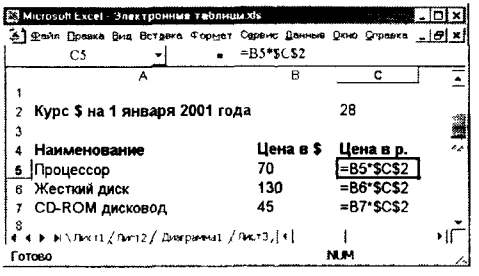
При копировании формулы, содержащей только абсолютные ссылки, из ячейки C3 в ячейку D4 обозначения столбцов и строк в формуле не изменятся (см. рис. 10.3).

Рассмотрим действие относительных и абсолютных ссылок на примере. Пусть нам необходимо вычислить стоимость комплектующих для компьютера в рублях, если известны их цены в долларах и курс доллара. Для вычисления цены в рублях необходимо умножить цену в долларах на величину курса доллара.

Копирование формул, содержащих относительные и абсолютные ссылки

1. Ввести в ячейки A5, A6 и A7 названия устройств, а в ячейки B5, B6, B7 — их цены в долларах.

2. Ввести в ячейку C2 курс доллара.
3. Ввести в ячейку C5 формулу =B5*$C$2, где B5 – относительная ссылка, а $C$2 — абсолютная.
4. Скопировать формулы в ячейки C6 и C7, абсолютная ссылка на ячейку $C$2 останется неизменной, а относительная B5 изменится на величину смещения от активной ячейки.



Если символ доллара стоит перед буквой (например $A1), то координата столбца абсолютная, а строки — относительная. Если символ доллара стоит перед числом (например, A$1), то, наоборот, координата столбца относительная, а строки — абсолютная. Такие ссылки называются *смешанными*.

Практические задания

10.2. Ознакомиться с окнами электронных таблиц Excel и StarCalc.

10.3. В ячейку B2 ввести формулу =A1/$B$1 и скопировать ее в ячейки C2, C3 и B3. Какие формулы будут содержаться в этих ячейках? Проверить на практике.

10.4. В ячейку C4 ввести формулу =$A1/B$1 и скопировать ее в ячейки D4, D5 и C4. Какие формулы будут содержаться в этих ячейках? Проверить на практике.

10.5. Создать в электронных таблицах таблицу умножения.

10.3. Встроенные функции

Формулы могут состоять не только из арифметических операторов и адресов ячеек. Часто в вычислениях приходится использовать формулы, содержащие функции. Электронные таблицы имеют несколько сотен встроенных функций, которые подразделяются на категории: *Математические*, *Статистические*, *Финансовые*, *Дата и время* и так далее.

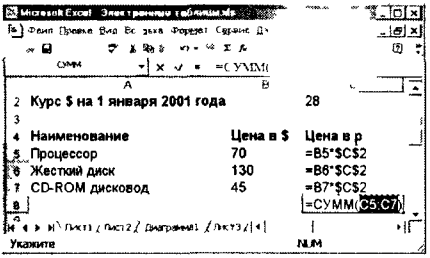
10.3.1. Математические функции

Одной из наиболее часто используемых операций является суммирование значений диапазона ячеек для расчета итоговых результатов. На панели инструментов *Стандартная* расположена кнопка [Σ], *Автосуммирование*, которая используется для автоматического суммирования чисел с помощью функции СУММ.

Воспользуемся рассмотренной выше таблицей, содержащей цены на комплектующие компьютера, и вычислим суммарную стоимость комплектующих.

Суммирование значений диапазона ячеек

1. Выделить ячейку С8, в которую следует поместить сумму.
2. Щелкнуть по кнопке [Σ], после чего будет выделен диапазон ячеек для суммирования СУММ (С5:С7).

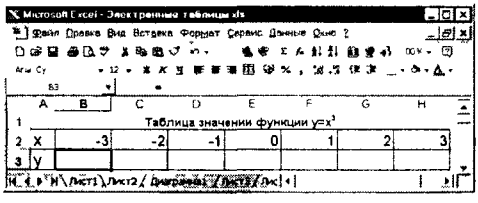


3. Если предложенный диапазон не подходит, следует протащить указатель мыши по ячейкам, которые нужно просуммировать. Нажать клавишу {*Enter*}.

При вводе в формулу функций удобно использовать *Мастер функций*. Например, пусть нам необходимо составить таблицу значений функции $y = x^3$ на отрезке [−3; 3] с шагом 1.

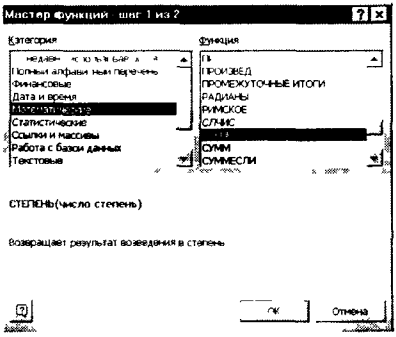
Составление таблицы значений функции с использованием *Мастера функций*

1. Подготовить таблицу, ввести значения аргумента.
   Выделить ячейку, в которую нужно вставить первое значение функции.

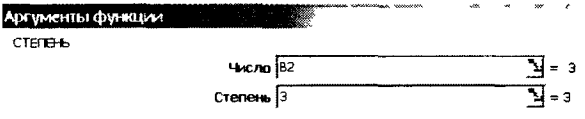


2. Ввести команду [Вставка-Функция...].

3. На диалоговой панели *Мастер функций: шаг 1* в списке *Категория:* выбрать вариант *Математические*, а в списке *Функция:* выбрать вариант *Степень*. Щелкнуть по кнопке *ОК*.



4. На появившейся панели *Аргументы функции* вести значения числа и показателя степени.
Для ввода имени ячейки, где хранится число, щелкнуть по кнопке со стрелочкой в поле *Число* и в электронных таблицах выделить ячейку B2.
Ввести в поле *Степень* число 3.



5. Выделить ячейку B3, в которой теперь хранится формула =Степень(B2;3) и заполнить ряд значений функций с помощью команды [Правка-Заполнить-Вправо].

Практические задания

10.6. С использованием *Мастера функций* получить таблицу значений функции $y=(x-5)^2$ на отрезке $[-5; 5]$ с шагом 1.

10.3.2. Логические функции

3.2. Алгебра высказываний

Ранее были рассмотрены базовые логические операции (умножения, сложения, отрицания) и их таблицы истинности. В электронных таблицах имеются соответствующие логические функции, с помощью которых достаточно просто построить таблицы истинности логических операций.

Аргументами логических функций являются логические значения ИСТИНА и ЛОЖЬ. Логические значения, в свою очередь, могут быть получены как результат определения

значений логических выражений. Например, для логического выражения 10>5 результатом будет логическое значение ИСТИНА, а для логического выражения A1<A2 (где в ячейке A1 хранится число 10, а в ячейке A2 — число 5) — значение ЛОЖЬ.

Логическая функция «И» имеет в качестве аргументов логические значения, которые могут быть истинными или ложными, и задается формулой =И(*лог_знач1*;*лог_знач2*;...). Принимает значение ИСТИНА тогда и только тогда, когда все аргументы имеют значение ИСТИНА.

Например, значение функции =И(10>5;10<5) — ЛОЖЬ.

Логическая функция «ИЛИ» имеет в качестве аргументов логические значения и задается формулой =ИЛИ(*лог_знач1*; *лог_знач2*; ...). Принимает значение ИСТИНА, если хотя бы один из аргументов имеет значение ИСТИНА.

Например, значение функции =ИЛИ(10>5;10<5) — ИСТИНА.

Логическая функция «НЕ» имеет один аргумент и задается формулой =НЕ(*лог_знач*). Принимает значение ИСТИНА, если аргумент имеет значение ЛОЖЬ, и наоборот.

Например, значение функции =НЕ(10>5) — ЛОЖЬ.

Построим с помощью электронных таблиц таблицу истинности операции логического умножения, используя логическую функцию «И».

Построение таблицы истинности операции логического умножения

1. В пары ячеек (A1,B1), (A2,B2), (A3,B3), (A4,B4) ввести пары значений аргументов логической операции (ЛОЖЬ,ЛОЖЬ), (ИСТИНА,ЛОЖЬ), (ЛОЖЬ,ИСТИНА) и (ИСТИНА,ИСТИНА).
2. В ячейку C1 ввести формулу логической функции «И» (=И(A1;B1).
3. Скопировать формулу в ячейки C2, C3 и C4.
4. Значением этой функции в трех случаях является ЛОЖЬ и только в последнем — ИСТИНА. Мы получили таблицу истинности операции логического умножения.

	A	B	C
1	ЛОЖЬ	ЛОЖЬ	ЛОЖЬ
2	ИСТИНА	ЛОЖЬ	ЛОЖЬ
3	ЛОЖЬ	ИСТИНА	ЛОЖЬ
4	ИСТИНА	ИСТИНА	ИСТИНА

Практические задания

10.7. В электронных таблицах получить таблицы истинности операций логического сложения и логического отрицания.

10.4. Сортировка и поиск данных

10.4.1. Сортировка данных

Электронные таблицы позволяют осуществлять сортировку данных, то есть производить их упорядочение. Данные (числа, текст, даты) в электронных таблицах можно сортировать по возрастанию или убыванию. При сортировке по возрастанию данные выстраиваются в следующем порядке:

- числа сортируются от наименьшего отрицательного до наибольшего положительного числа;
- текст сортируется в следующем порядке: числа, знаки, латинский алфавит, русский алфавит;
- пустые ячейки всегда помещаются в конец списка.

Для сортировки строк таблицы необходимо выбрать столбец, данные которого будут упорядочиваться. После сортировки изменяется порядок следования строк, но сохраняется их целостность.

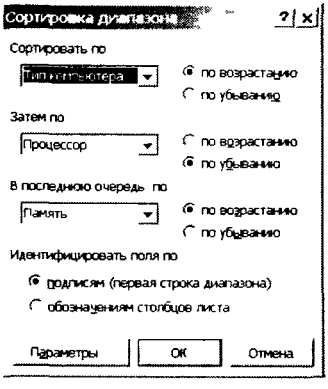
Можно проводить вложенные сортировки, то есть сортировать данные последовательно по нескольким столбцам. При вложенной сортировке строки, имеющие одинаковые значения в ячейках первого столбца, будут упорядочены по значениям в ячейках второго столбца, а строки, имеющие одинаковые значения во втором столбце, будут упорядочены по значениям третьего столбца. Так, результат вложенной сортировки таблицы (исключая первую строку), содержащей данные о компьютерах, при сортировке столбца A по возрастанию, столбца B по убыванию и столбца C по возрастанию будет таким, как показано на рис. 10.4.

	A	B	C
1	**Тип компьютера**	**Процессор**	**Память**
2	Настольный	Pentium III	64
3	Настольный	Pentium 4	64
4	Настольный	Pentium 4	128
5	Портативный	Pentium III	64
6	Портативный	Pentium III	128
7	Портативный	Pentium 4	64

Рис. 10.4. Вложенная сортировка

Сортировка данных

1. Выделить одну из ячеек с данными и ввести команду [Данные-Сортировка...]
2. На диалоговой панели *Сортировка диапазона* в списке *Сортировать по* выбрать столбец *Тип компьютера* и установить переключатель в положение *по возрастанию.*
   В списке *Затем по* выбрать столбец *Процессор* и установить переключатель в положение *по убыванию.*
   В списке *В последнюю очередь по* выбрать столбец *Память* и установить переключатель в положение *по возрастанию.*



3. После щелчка по кнопке *ОК* строки таблицы будут отсортированы.

Практические задания

10.8. В файле mapstats.xls (хранится в каталоге \textbook\Excel\) находятся статистические данные о населении стран мира. Отсортировать страны по численности населения.

10.4.2. Поиск данных

В электронных таблицах можно осуществлять поиск данных (строк) в соответствии с заданными условиями. Такие условия называются *фильтром.* В результате поиска будут найдены строки, удовлетворяющие заданному фильтру.

Условия задаются с помощью операций сравнения. Для числовых данных это операции *равно* (знак =), *меньше* (знак <), *больше* (знак >), *меньше или равно* (знак <=) и *больше или равно* (знак >=). Для задания условия необходимо выбрать операцию сравнения и задать число.

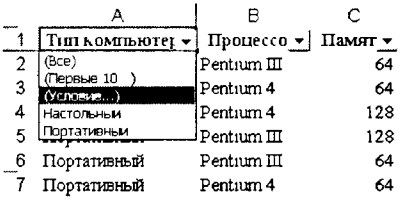
Для текстовых данных возможны операции сравнения *равно, начинается с* (сравниваются первые символы), *заканчивается на* (сравниваются последние символы), *содержит* (сравниваются символы в любой части текста). Для задания условия необходимо выбрать операцию сравнения и задать последовательность символов.

Можно осуществлять поиск данных, вводя условия поиска для нескольких столбцов. В этом случае фильтр будет содержать несколько условий, которые должны выполняться одновременно. Например, если мы хотим в таблице, изображенной на рис. 10.4, найти данные о настольном компьютере с процессором Pentium 4 и памятью 128 Мб, то необходимо задать фильтр, состоящий из трех условий:

- для столбца «Компьютеры» *равно Настольный*;
- для столбца «Процессор» *равно Pentium 4*;
- для столбца «Память» *больше 64*.

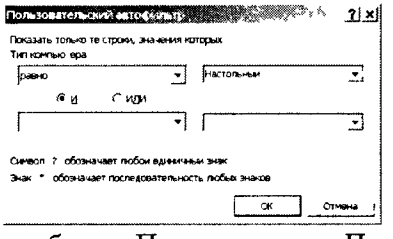
Поиск данных

1. Ввести команду [Данные-Фильтр-Автофильтр].
   В названиях столбцов таблицы появятся раскрывающиеся списки, содержащие стандартные условия поиска.



   Развернуть список в столбце «Тип компьютера» и выбрать пункт (*Условие...*) для ввода пользовательских условий.

2. На диалоговой панели *Пользовательский автофильтр* в полях ввести оператор условия поиска *равно* и значение *Настольный*.



3. Ввести условия поиска для столбцов «Процессор» и «Память». В результате будет найдена одна строка (4), удовлетворяющая заданному фильтру.

Практические задания

10.9. В файле mapstats xls (хранится в каталоге \textbook\Excel\) находятся статистические данные о населении стран мира. Найти страны с численностью населения более 100 миллионов человек.

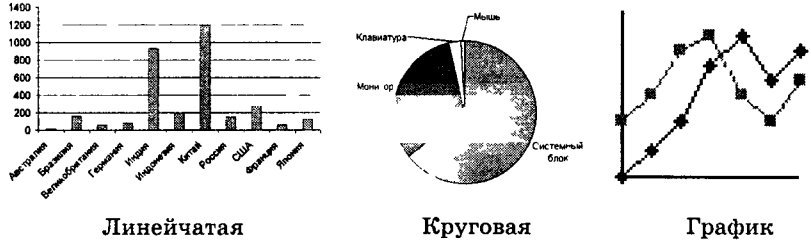
10.5. Построение диаграмм и графиков

Типы диаграмм и графиков. Электронные таблицы позволяют визуализировать данные, размещенные на рабочем листе, в виде диаграммы или графика. Диаграммы и графики наглядно отображают зависимости между данными, что облегчает восприятие и помогает при анализе и сравнении данных.

Диаграммы могут быть различных типов и соответственно представлять данные в различной форме. Для каждого набора данных важно правильно подобрать тип создаваемой диаграммы. Для наглядного сравнения различных величин используются линейчатые диаграммы. Например, с помощью линейчатой диаграммы можно наглядно представить данные о численности населения различных стран.

Для отображения величин частей от целого применяется круговая диаграмма. Круговая диаграмма позволяет, например, наглядно показать доли стоимости отдельных устройств компьютера в его цене.

Для отображения изменения величин в зависимости от времени и построения графиков функций используются диаграммы типа «график» (рис. 10.5).



Линейчатая Круговая График

Рис. 10.5. Типы диаграмм

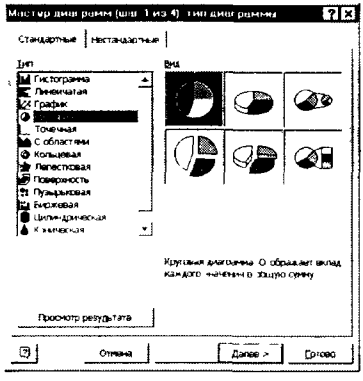
Диаграммы могут располагаться как на листе с данными (внедренные диаграммы), так и на отдельных листах. Диаграммы связаны с исходными данными на рабочем листе и обновляются при обновлении данных на рабочем листе.

Для создания диаграмм используется *Мастер диаграмм*. *Мастер диаграмм* позволяет создавать диаграмму по шагам с помощью серии диалоговых панелей. В качестве примера рассмотрим таблицу «Цены устройств компьютера» и представим в наглядной форме долю цены каждого устройства в цене компьютера.

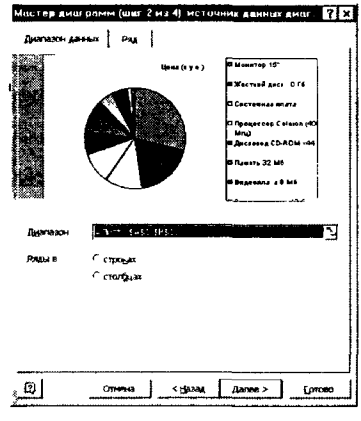
5.4.1. Табличные информационные модели

Создание диаграммы

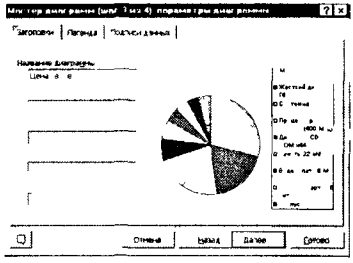
1. Выделить диапазон ячеек, содержащих данные. Запустить *Мастер диаграмм* с помощью команды [Вставка-Диаграмма...] или кнопки .
2. На первом шаге необходимо выбрать тип диаграммы.
   Для наглядного отображения части и целого наиболее подходит круговая диаграмма.
   В списке *Тип:* выбираем пункт *Круговая*.
   Круговые диаграммы могут быть различных видов (плоские, объемные и так далее).
   В окне *Вид:* выбираем плоскую диаграмму.



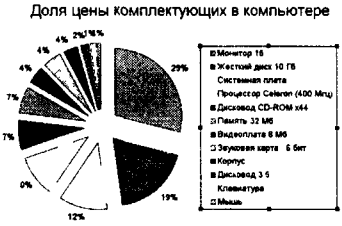
3. На втором шаге мы увидим, как будет выглядеть наша диаграмма.
   Справа от диаграммы появляется *Легенда*, которая содержит необходимые пояснения к диаграмме.
   Окно *Диапазон:* содержит диапазон адресов ячеек, содержащих данные для диаграммы.
   Этот диапазон можно изменить.



4. На третьем шаге мы можем уточнить детали отображения диаграммы, изменить формат диаграммы и легенды (размеры, шрифт, цвета, подписи и так далее).



5. На четвертом шаге необходимо определить, где разместить диаграмму: на отдельном листе или на листе вместе с данными.
Наконец, в результате мы получим готовую диаграмму.



Построение графиков. Построение графиков является частным случаем построения диаграмм. Графики выбирают в тех случаях, когда хотят отобразить изменение данных с течением времени. Графики позволяют анализировать закономерности изменения величин.

Представим с помощью графика динамику изменения цен по годам на различные модели компьютеров (на базе процессоров Pentium II и Pentium III) — рис. 10.6.

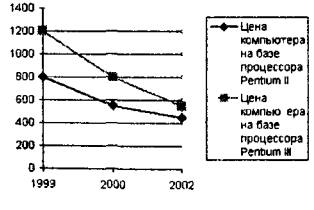
	1999	2000	2002
Цена компьютера на базе процессора Pentium II	800	550	450
Цена компьютера на базе процессора Pentium III	1200	800	550

Рис. 10.6. Изменение цены компьютеров

Процесс построения графика с помощью *Мастера диаграмм* аналогичен рассмотренному выше.

Построение графика

1. Запустить *Мастер диаграмм*. Выбрать тип диаграммы *График*. Подобрать параметры удобного представления графика.



В результате мы получаем два графика, которые позволяют сделать интересные выводы. Оказывается, цены различных моделей компьютеров стремятся достичь некоторого минимального значения (около 500$) в течение 2–3 лет с начала производства. Попробуйте объяснить эту закономерность.

Практические задания

10.10. Построить гистограмму, которая показывает сравнительное количество серверов Интернета (статистика хранится в каталоге \soft\internet\Internet-statistic\) в разных странах.

10.11. Построить график, который показывает рост количества серверов Интернета по годам.

10.12. Пользуясь данными, приведенными в таблице, построить диаграмму, характеризующую соотношение между неметрическими единицами длины. Какой тип диаграммы целесообразно выбрать?

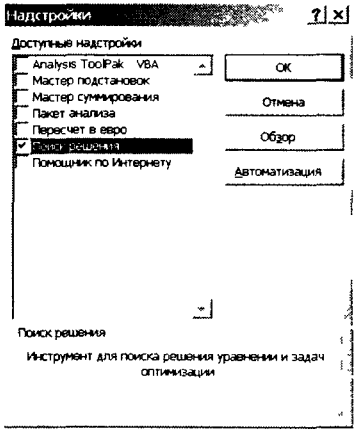
Единицы	Значение в мм
Сотка	21,336
Аршин	713,20
Четверть	177,80
Вершок	44,45
Фут	304,80
Дюйм	25,40
Линия	2,54

10.6. Надстройки в электронных таблицах

Возможности электронных таблиц не ограничиваются вычислениями по формулам и построением диаграмм и графиков. С помощью надстроек электронных таблиц можно строить геоинформационные модели (выводить на географические карты данные), приближенно с заданной точностью решать уравнения методом подбора параметра, решать задачи оптимизационного моделирования методом поиска решений и так далее. Некоторые из надстроек не устанавливаются по умолчанию и требуют установки.

Установка надстроек

1. Ввести команду [Сервис-Надстройки].
   На панели *Надстройки* в списке *Доступные надстройки* выбрать нужные путем установки флажков.
   Щелкнуть по кнопке ОК.



Поиск решения. *Поиск решения* является надстройкой, которая позволяет решать задачи оптимизационного моделирования. Процедура поиска решения позволяет найти оптимальное значение формулы, содержащейся в ячейке, которая называется целевой. Эта процедура работает с группой ячеек, прямо или косвенно связанных с формулой в целевой ячейке. Чтобы получить по формуле, содержащейся в целевой ячейке, искомый результат, процедура изменяет значения во влияющих ячейках. Чтобы сузить множество значений, используемых в модели, применяются ограничения. Эти ограничения могут содержать ссылки на другие влияющие ячейки.

5.10. Оптимизационное моделирование в экономике

Подбор параметра. *Подбор параметра* является одним из инструментов анализа «что, если». Этот метод используется при поиске значения аргумента функции, который обеспечивает требуемое значение функции. При подборе параметра изменяется значение в ячейке аргумента функции до тех пор, пока значение в ячейке самой функции не будет возвращать нужный результат.

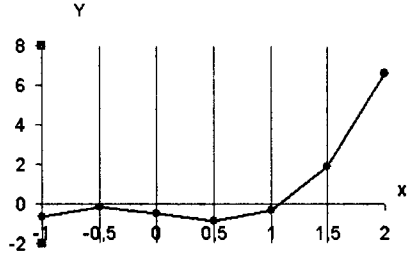
В качестве примера рассмотрим поиск корня уравнения $x^3 - \sin x - 0{,}5 = 0$. Представим функцию в табличной форме, построим ее график, который позволит определить корень уравнения грубо приближенно. Для поиска решения с заданной точностью используем метод *Подбор параметра*. Точность подбора зависит от заданной точности представления чисел в ячейках таблицы.

Подбор параметра

1. Представить функцию в табличной форме.

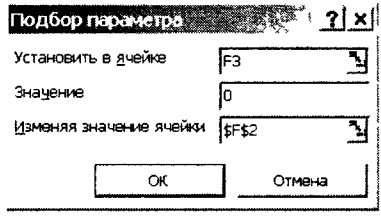
	A	B	C	D	E	F	G	H
1	Таблица значений функции $y = x^2 - \sin x - 0{,}5$							
2	x	-1	-0,5	0	0,5	1	1,5	2
3	y	-0,8585	-0,1456	-0,5000	-0,8544	-0,3415	1,8775	6,5907

2. Построить график функции. По графику грубо приближенно можно определить, что уравнение имеет корень $x = 1$.

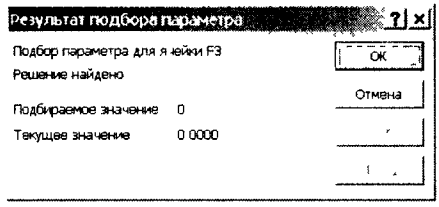


Методом подбора параметра необходимо определить значение аргумента x (ячейка F2), при котором значение функции y (ячейка F3) равно нулю.

3. Ввести команду [Сервис-Подбор параметра...].
4. На панели *Подбор параметра* в поле *Значение* ввести требуемое значение функции (в данном случае 0). В поле *Изменяя значение ячейки* ввести адрес ячейки $F$2, в которой будет производиться подбор значения аргумента.



5. На панели *Результат подбора параметра* будет выведена информация о величине подбираемого и подобранного значений.



6. В ячейке аргумента F2 появится подобранное значение 1,1185.

Таким образом, корень уравнения $x = 1{,}1185$ найден с заданной точностью.

Практические задания

10.13. Методом подбора параметра решить уравнение $x^2 - \sin x + 0{,}1 = 0$ с точностью четырех знаков после запятой.

Глава 11

Технология хранения, поиска и сортировки информации

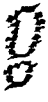
11.1. Базы данных

Любой из нас, начиная с раннего детства, многократно сталкивался с «базами данных». Это — всевозможные справочники (например, телефонный), энциклопедии и т. п. Записная книжка — это тоже «база данных», которая есть у каждого из нас.

Базы данных представляют собой информационные модели, содержащие данные об объектах и их свойствах. Базы данных хранят информацию о группах объектов с одинаковым *набором свойств*.

Например, база данных «Записная книжка» хранит информацию о людях, каждый из которых имеет фамилию, имя, телефон и так далее. Библиотечный каталог хранит информацию о книгах, каждая из которых имеет название, автора, год издания и так далее.

Информация в базах данных хранится в упорядоченном виде. Так, в записной книжке все записи упорядочены по алфавиту, а в библиотечном каталоге — либо по алфавиту (алфавитный каталог), либо по области знания (предметный каталог).



База данных (БД) – это информационная модель, позволяющая в упорядоченном виде хранить данные о группе объектов, обладающих одинаковым набором свойств.

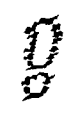
Существует несколько различных структур информационных моделей и соответственно различных типов баз данных: *табличные, иерархические и сетевые.*

5.4. Типы информационных моделей

11.1.1. Табличные базы данных

Табличная база данных содержит перечень объектов одного типа, то есть объектов, имеющих одинаковый набор свойств. Такую базу данных удобно представлять в виде двумерной таблицы: в каждой ее строке последовательно размещаются значения свойств одного из объектов; каждое значение свойства — в своем столбце, озаглавленном именем свойства.

Столбцы такой таблицы называют *полями*; каждое поле характеризуется своим именем (именем соответствующего свойства) и типом данных, представляющих значения данного свойства.

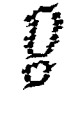


Поле базы данных – это столбец таблицы, содержащий значения определенного свойства.

Строки таблицы являются *записями* об объекте; эти записи разбиты на поля столбцами таблицы, поэтому каждая запись представляет собой набор значений, содержащихся в полях.

Запись базы данных – это строка таблицы, содержащая набор значений свойств, размещенный в полях базы данных.

Каждая таблица должна содержать, по крайней мере, одно *ключевое поле*, содержимое которого уникально для каждой записи в этой таблице. Ключевое поле позволяет однозначно идентифицировать каждую запись в таблице.



Ключевое поле – это поле, значение которого однозначно определяет запись в таблице.

В качестве ключевого поля чаще всего используют поле, содержащее тип данных *счетчик*. Однако иногда удобнее в качестве ключевого поля таблицы использовать другие поля: код товара, инвентарный номер и т. п.

Тип поля определяется типом данных, которые оно содержит. Поля могут содержать данные следующих основных типов:

- *счетчик* — целые числа, которые задаются автоматически при вводе записей. Эти числа не могут быть изменены пользователем;
- *текстовый* — тексты, содержащие до 255 символов;
- *числовой* — числа;
- *дата/время* — дата или время;
- *денежный* — числа в денежном формате;
- *логический* — значения *Истина* (Да) или *Ложь* (Нет);
- *гиперссылка* — ссылки на информационный ресурс в Интернете (например, Web-сайт).

Поле каждого типа имеет свой набор свойств. Наиболее важными свойствами полей являются:

- *размер поля* — определяет максимальную длину текстового или числового поля;
- *формат поля* — устанавливает формат данных;
- *обязательное поле* — указывает на то, что данное поле обязательно надо заполнить.

Рассмотрим, например, базу данных «Компьютер», которая содержит перечень объектов (компьютеров), каждый из которых имеет имя (название). В качестве характеристик (свойств) можно рассмотреть тип установленного процессора и объем оперативной памяти. Поля *Название* и *Тип процессора* являются текстовыми, *Оперативная память* — числовым, а поле *№ п/п* — счетчиком (табл. 11.1).

При этом каждое поле обладает определенным набором свойств. Например, для поля *Оперативная память* задан формат данных *целое число*.

Таблица 11.1. Табличная база данных

№ п/п	Название	Тип процессора	Оперативная память (Мбайт)
1	Compaq	Celeron	64
2	Dell	Pentium III	128
3	IBM	Pentium 4	256

Вопросы для размышления

1. В чем заключается разница между записью и полем в табличной базе данных?
2. Поля каких типов полей могут присутствовать в базе данных?
3. Чем отличается ключевое поле от остальных полей?

11.1.2. Иерархические и сетевые базы данных

Иерархические базы данных. Иерархические базы данных графически могут быть представлены как перевернутое дерево, состоящее из объектов различных уровней. Верхний уровень (корень дерева) занимает один объект, второй — объекты второго уровня и так далее.

Между объектами существуют связи, каждый объект может включать в себя несколько объектов более низкого уровня. Такие объекты находятся в отношении *предка* (объект, более близкий к корню) к *потомку* (объект более низкого уровня), при этом объект-предок может не иметь потомков или иметь их несколько, тогда как объект-потомок обязательно имеет только одного предка. Объекты, имеющие общего предка, называются *близнецами*.

Иерархической базой данных является *Каталог папок Windows*, с которым можно работать, запустив Проводник. Верхний уровень занимает папка *Рабочий стол*. На втором уровне находятся папки *Мой компьютер*, *Мои документы*, *Сетевое окружение* и *Корзина*, которые являются потомками папки *Рабочий стол*, а между собой является близнецами. В свою очередь, папка *Мой компьютер* является предком по отношению к папкам третьего уровня —папкам дисков (*Диск 3,5(A:)*, (*C:*), (*D:*), (*E:*), (*F:*)) и системным папкам (*Принтеры*, *Панель управления* и др.) — рис. 11.1.

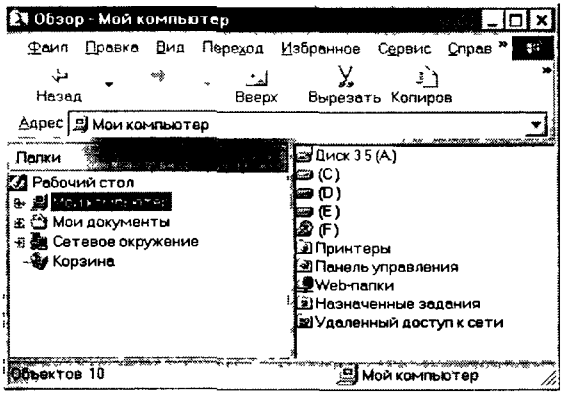


Рис. 11.1 Иерархическая база данных *Каталог папок Windows*

Иерархической базой данных является *Реестр Windows*, в котором хранится вся информация, необходимая для нормального функционирования компьютерной системы (данные о конфигурации компьютера и установленных драйверах, сведения об установленных программах, настройки графического интерфейса и др.).

Содержание реестра автоматически обновляется при установке нового оборудования, инсталляции программ и т. п. Для просмотра и редактирования реестра Windows в ручном режиме можно использовать специальную программу regedit.exe, которая хранится в папке Windows. Однако редактирование реестра можно проводить только в случае крайней необходимости и при условии понимания выполняемых действий. Неквалифицированное редактирование реестра может привести компьютер в неработоспособное состояние.

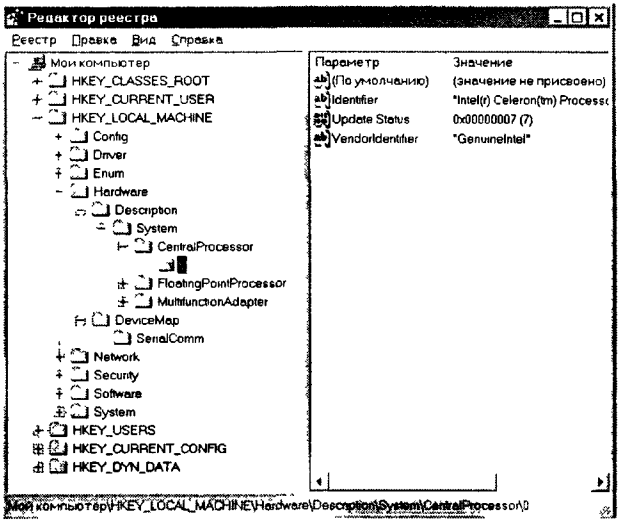
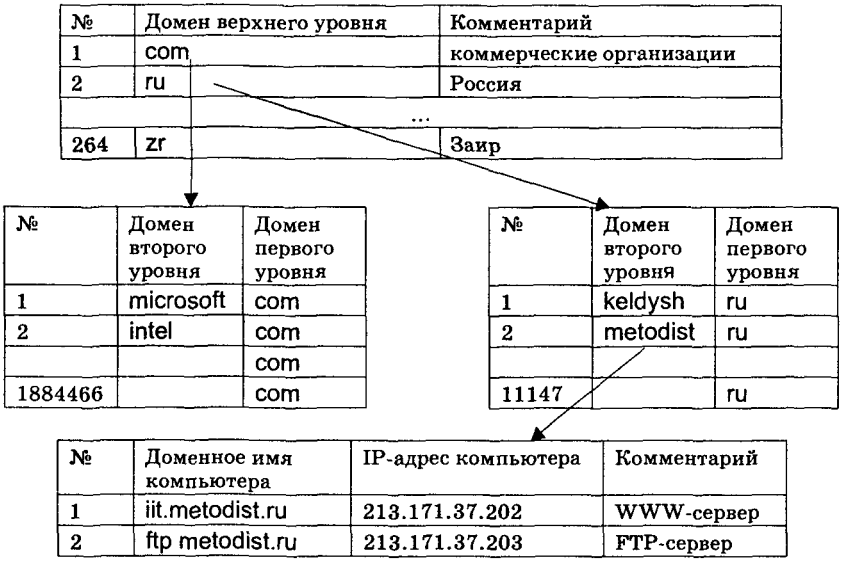


Рис. 11.2 Иерархическая база данных *Реестр Windows*

Еще одним примером иерархической базы данных является база данных *Доменная система имен* подключенных к Интернету компьютеров. На верхнем уровне находится табличная база данных, содержащая перечень доменов верхнего уровня (всего 264 домена), из которых 7 — административные, а остальные 257 — географические. Наиболее крупным доменом (данные на январь 2002 года) является домен net (около 48 миллионов серверов), а в некоторых доменах (например, в домене zr) до сих пор не зарегистрировано ни одного сервера.

На втором уровне находятся табличные базы данных, содержащие перечень доменов второго уровня для каждого домена первого уровня.

На третьем уровне могут находиться табличные базы данных, содержащие перечень доменов третьего уровня для каждого домена второго уровня, и таблицы, содержащие IP-адреса компьютеров, находящихся в домене второго уровня (рис. 11.3).



№	Домен верхнего уровня	Комментарий
1	com	коммерческие организации
2	ru	Россия
	...	
264	zr	Заир

№	Домен второго уровня	Домен первого уровня
1	microsoft	com
2	intel	com
		com
1884466		com

№	Домен второго уровня	Домен первого уровня
1	keldysh	ru
2	metodist	ru
11147		ru

№	Доменное имя компьютера	IP-адрес компьютера	Комментарий
1	iit.metodist.ru	213.171.37.202	WWW-сервер
2	ftp metodist.ru	213.171.37.203	FTP-сервер

Рис. 11.3. Иерархическая база данных *Доменная система имен*

База данных *Доменная система имен* должна содержать записи обо всех компьютерах, подключенных к Интернету, то есть более 150 миллионов записей. Размещение такой огромной базы данных на одном компьютере сделало бы поиск информации очень медленным и неэффективным. Решение этой проблемы было найдено путем размещения отдельных составных частей базы данных на различных DNS-серверах. Таким образом, иерархическая база данных *Доменная система имен* является *распределенной базой данных*.

Поиск информации в такой иерархической распределенной базе данных ведется следующим образом. Например, мы хотим ознакомиться с содержанием WWW-сервера фирмы Microsoft.

Сначала наш запрос, содержащий доменное имя сервера www.microsoft.com, будет оправлен на DNS-сервер нашего провайдера, который переадресует его на DNS-сервер самого верхнего уровня базы данных. В таблице первого уровня будет найден интересующий нас домен com и запрос будет адресован на DNS-сервер второго уровня, который содержит перечень доменов второго уровня, зарегистрированных в домене com.

В таблице второго уровня будет найден домен microsoft и запрос будет переадресован на DNS-сервер третьего уровня. В таблице третьего уровня будет найдена запись, соответствующая доменному имени, содержавшемуся в запросе. Поиск информации в базе данных *Доменная система имен* будет завершен и начнется поиск компьютера в сети по его IP-адресу.

Сетевые базы данных. Сетевая база данных является обобщением иерархической за счет допущения объектов, имеющих более одного предка. Вообще, на связи между объектами в сетевых моделях не накладывается никаких ограничений.

Сетевой базой данных фактически является *Всемирная паутина* глобальной компьютерной сети Интернет. Гиперссылки связывают между собой сотни миллионов документов в единую распределенную сетевую базу данных.

12.9. Всемирная паутина

Вопросы для размышления

1. Чем различаются между собой табличные, иерархические и сетевые базы данных? Приведите примеры.
2. Чем различаются между собой сетевые и распределенные базы данных?

11.2. Система управления базами данных Access

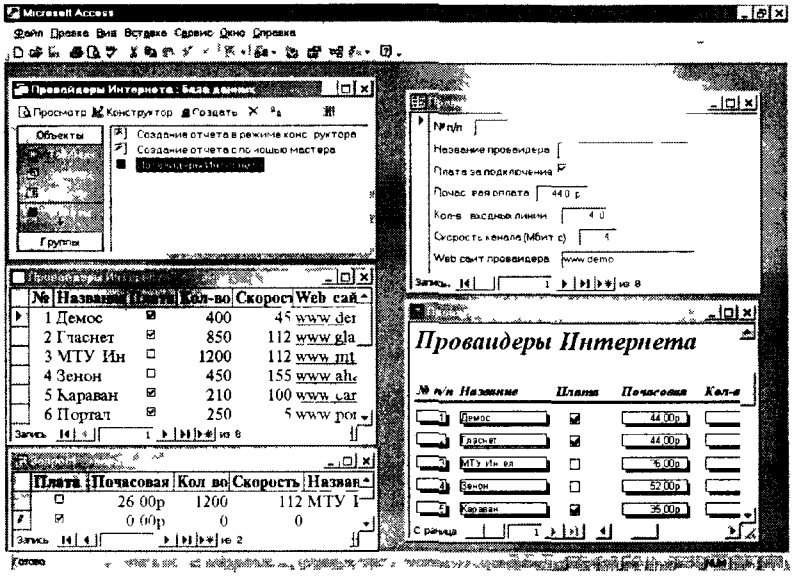
Системы управления базами данных (СУБД). Развитие информационных технологий привело к созданию компьютерных баз данных. Создание баз данных, а также операции поиска и сортировки данных выполняются специальными программами — *системами управления базами данных* (*СУБД*). Таким образом, необходимо различать собственно базы данных (БД), которые являются упорядоченными наборами данных, и системы управления базами данных (СУБД) — программы, управляющие хранением и обработкой данных.

Система управления базами данных (СУБД) – это программа, позволяющая создавать базы данных, а также обеспечивающая обработку (сортировку) и поиск данных

Системой управления базами данных является приложение Access, входящее в Microsoft Office.

Знакомство с СУБД Access

1. Запустить Access командой [Пуск-Программы-Microsoft Access].
2. С помощью команды [Файл-Открыть базу данных...] открыть существующую базу данных, например, базу данных «Провайдеры Интернета», хранящуюся на CD-ROM в каталоге \textbook\Access\.



В Access используется стандартный для среды Windows&Office многооконный интерфейс, но в отличие от других приложений, не многодокументный. Единовременно может быть открыта только одна база данных, содержащая обязательное *окно базы данных* и *окна для работы с объектами базы данных*. В каждый момент времени одно из окон является активным и в нем курсором отмечается активный объект.

Окно базы данных — один из главных элементов интерфейса Access. Здесь систематизированы все объекты БД: *таблицы, запросы, формы, отчеты, макросы и модули.* В данном случае открыто и активно окно *Провайдеры Интернета: база данных.*

Таблица. В базах данных вся информация хранится в двумерных таблицах. Это *базовый* объект БД, все остальные объекты создаются на основе существующих таблиц (*производные* объекты). Каждая строка в таблице — *запись* БД, а столбец — *поле.* Запись содержит набор данных об одном объекте, а поле — однородные данные обо всех объектах.

В данном случае открыто окно *Провайдеры Интернета: таблица.*

Запросы. В СУБД запросы являются важнейшим инструментом. Главное предназначение запросов — отбор данных на основании заданных условий. С помощью запроса из базы данных можно выбрать информацию, удовлетворяющую определенным условиям.

В данном случае открыто окно *Запрос1: запрос на выборку.*

Формы. Формы позволяют отображать данные, содержащиеся в таблицах или запросах, в более удобном для восприятия виде. При помощи форм можно добавлять в таблицы новые данные, а также редактировать или удалять существующие. Форма может содержать рисунки, графики и другие внедренные объекты.

В данном случае открыто окно формы *Провайдеры Интернета.*

Отчеты. Отчеты предназначены для печати данных, содержащихся в таблицах и запросах, в красиво оформленном виде.

В данном случае открыто окно отчета *Провайдеры Интернета.*

Макросы. Макросы служат для автоматизации повторяющихся операций. Запись макроса производится так же, как в других приложениях, например как в приложении Word.

Модули. Модули также служат для автоматизации работы с БД. Модули еще называют *процедурами обработки событий* и пишутся на *языке VBA.*

Практические задания

11.1. Открыть в СУБД Access готовую базу данных и ознакомиться с окном базы данных и окнами объектов базы данных.

11.3. Создание базы данных

Пусть нам необходимо разработать базу данных «Провайдеры Интернета», которая содержит информацию, необходимую для обоснованного выбора провайдера. Целесообразно в качестве основных критериев выбора взять стоимость подключения, тариф почасовой оплаты, количество входных телефонных линий и пропускную способность канала связи, который соединяет провайдера с Интернетом.

11.3.1. Создание структуры базы данных

Прежде всего необходимо определить структуру базы данных, то есть количество полей, их названия и тип данных, в них хранящихся. База данных «Провайдеры Интернета» будет содержать следующие поля:

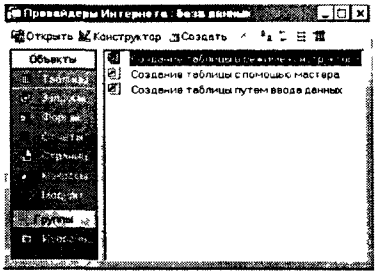
- «*№ п/п*» (*счетчик*) — ключевое поле, однозначно идентифицирующее запись;
- «*Название провайдера*» (*текстовый*) — содержит название фирмы;
- «*Плата за подключение*» (*логический*) — принимает значения *Да* (плата взимается) и *Нет* (плата не берется);
- «*Почасовая оплата*» (*денежный*) — содержит величину оплаты в рублях за 1 час подключения;
- «*Кол-во входных линий*» (*числовой*) — содержит число входных телефонных линий;
- «*Скорость канала (Мбит/с)*» (*числовой*) — содержит значение суммарной пропускной способности каналов связи в Мбит/с, которые соединяют провайдера с Интернетом;
- «*Web-сайт провайдера*» (*гиперссылка*) — содержит ссылку на сайт провайдера в Интернете.

Приступим к практическому созданию базы данных «Провайдеры Интернета».

Создание базы данных «Провайдеры Интернета»

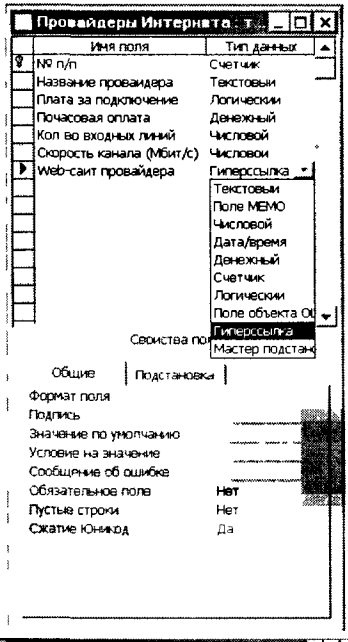
1. Создать в приложении Access новую базу данных с помощью команды [Файл-Создать базу данных...] и присвоить ей имя «Провайдеры Интернета».

2. В окне *Провайдеры Интернета: база данных* выбрать группу объектов *Таблицы*, затем пункт *Создание таблицы в режиме конструктора*.

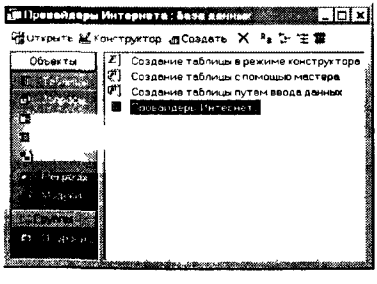


Режим *Конструктор* позволяет создавать и изменять структуру таблицы. Таблица «Провайдеры Интернета» должна содержать шесть полей, для каждого из которых нужно задать имя, тип данных и определить его свойства. Кроме того, необходимо задать ключевое поле, которым в данном случае является поле «*№ п/п*».

3. В появившемся окне *Провайдеры Интернета: таблица* в столбцах *Имя поля* и *Тип данных* ввести названия полей и требуемые типы данных.
   В нижней части окна задать свойства полей.
   При задании типов данных и свойств полей воспользоваться раскрывающимися списками.
   В качестве ключевого поля задать поле «*№ п/п*».
   Для сохранения таблицы ввести команду [Файл-Сохранить как...].
   Присвоить таблице имя «Провайдеры Интернета».

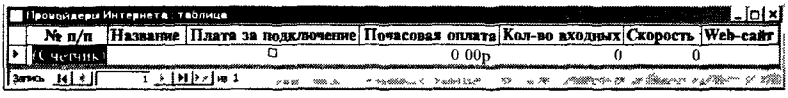


4. После создания таблицы ее имя добавляется в окно базы данных и ее можно легко открыть либо в режиме *Конструктор* (кнопка *Конструктор*), либо в режиме *Таблица* (кнопка *Открыть*).



Режим *Таблица* позволяет просматривать и изменять структуру таблицы, а также вводить и редактировать данные.

5. Для просмотра структуры таблицы в окне БД на вкладке *Таблицы* дважды щелкнуть на значке таблицы *Провайдеры Интернета*. Появится окно таблицы.



Практические задания

11.2. Создать базы данных «Записная книжка» и «Библиотечный каталог».

11.3.2. Ввод и редактирование данных

Ввод данных в таблицу базы данных и их редактирование мало чем отличается от аналогичных действий в других офисных приложениях.

При вводе данных в режиме *Таблица* в поле маркера записи, которое расположено слева от полей таблицы, может отображаться один из следующих символов:

* (звездочка) — обозначает пустую запись в конце таблицы;

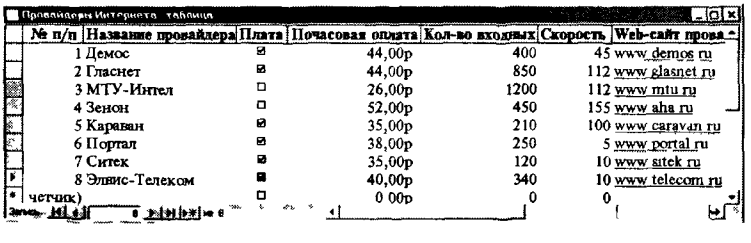
▸ (стрелка) — обозначает выделенную (активную) запись;

✎ (карандаш) — обозначает, что в записи были сделаны изменения.

Введем в таблицу данные:

Ввод данных в БД «Провайдеры Интернета»

1. Открыть таблицу двойным щелчком на ее значке в окне *Провайдеры Интернета: таблица*.
2. Заполнить БД, последовательно вводя записи о провайдерах:



Перемещение между записями можно осуществлять с помощью мыши, клавиш управления курсором или полосы прокрутки. Для быстрого перемещения между записями в базе данных можно использовать кнопки перемещения на панели *Запись*, которая находится в нижней части окна таблицы.

Практические задания

11.3. Ввести в базы данных «Записная книжка» и «Библиотечный каталог» по 5–10 записей.

11.3.3. Использование формы для просмотра и редактирования записей

Записи БД можно просматривать и редактировать в виде таблицы или в виде формы. Выше мы работали с базами данных, представленными в виде таблицы, когда запись образует строку в этой таблице. Такое представление БД позволяет наблюдать несколько записей одновременно, и в этом состоит достоинство табличного представления.

Однако часто вид *Таблица* не позволяет видеть полностью всю информацию на экране. Если БД содержит достаточно много полей, а значения полей содержат много символов, то все поля таблицы могут не умещаться на экране, а значения полей могут быть видны не полностью.

Форма отображает одну запись в удобном для пользователя виде. В процессе создания формы можно указать, какие поля БД включить в форму, как расположить поля в окне формы, а также как можно сделать форму визуально привлекательной.

Фактически с помощью формы создается графический интерфейс доступа к БД, который может содержать различные *управляющие элементы* (текстовые поля, кнопки, переключатели и так далее), а также *надписи*. Обычно на форме размещаются *надписи*, являющиеся именами полей БД, и *текстовые поля*, содержащие данные из БД.

Пользователь может изменять *дизайн* формы (размер, цвет и так далее) управляющих элементов и надписей.

Примерами форм могут являться «Визитка» в БД «Записная книжка» или «Карточка» в БД «Библиотечный каталог», которые содержат лишь одну запись БД, зато представленную в удобном для пользователя виде.

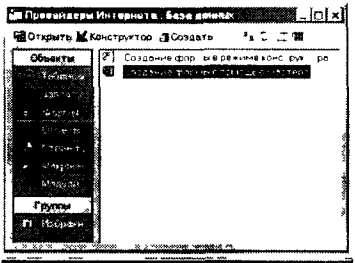
Создание формы можно проводить различными способами с использованием:

- Конструктора (сложный путь), который позволяет начать создание формы с нуля;
- Мастера форм (более простой путь), который с помощью серии диалоговых панелей помогает пользователю в создании формы.

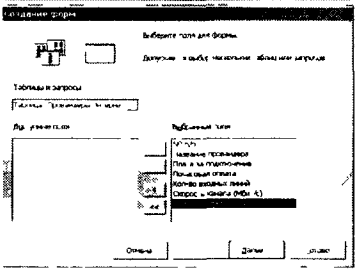
Создадим теперь форму для БД «Провайдеры Интернета» с помощью Мастера форм.

Создание формы для БД «Провайдеры Интернета»

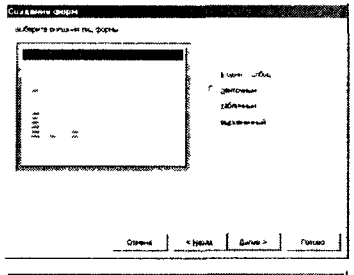
1. В окне *Провайдеры Интернета: база данных* выбрать группу объектов *Формы*. Выбрать пункт *Создание формы с помощью мастера*.



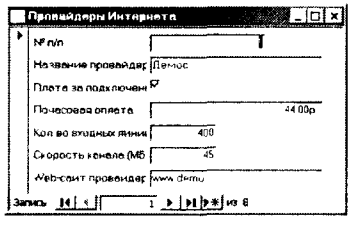
2. На появившейся панели *Создание форм* выбрать в окне *Таблицы и запросы* исходную таблицу, а в окне *Доступные поля:* — поля для *Формы*. Щелкнуть по кнопке *Далее*.



3. На появившейся следующей панели с помощью переключателей выбрать способ размещения полей на *Форме* (например, *в один столбец*). Щелкнуть по кнопке *Далее*.

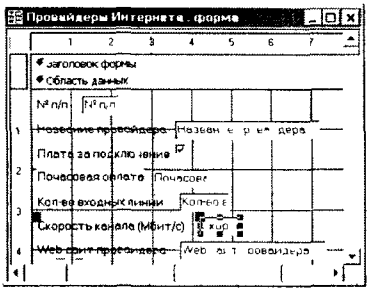


4. В результате появится окно формы «Провайдеры Интернета», которое содержит *надписи* (названия полей БД) и *текстовые поля* для ввода значений полей БД, расположенные в столбик.



Вид формы можно изменять в режиме *Конструктор*. В созданной нами форме «Провайдеры Интернета» не все надписи видны полностью, поэтому отводимое для них место на форме нужно увеличить. С другой стороны, поля вывода числовых значений можно уменьшить, чтобы приблизить значения к надписям.

5. В окне *Провайдеры Интернета: база данных* выделить форму «Провайдеры Интернета» и щелкнуть по кнопке *Конструктор*.
В появившемся окне с помощью мыши изменить местоположение, размеры надписей и текстовых полей.



Теперь форма для БД «Провайдеры Интернета» готова. С ее помощью можно просматривать записи, редактировать их или вводить новые.

После открытия форма содержит запись № 1. При работе с формой для перехода от одной записи к другой необходимо воспользоваться панелью *Запись*, которая находится в нижней части окна формы. Панель *Запись* содержит кнопки со стрелками, щелчки по которым позволяют перемещаться по записям, а также поле номера записи, позволяющее ввести номер искомой записи.

6. В окне *Провайдеры Интернета: база данных* выделить форму «Провайдеры Интернета» и щелкнуть по кнопке *Открыть*. Появится форма с измененными местоположением, размерами надписей и текстовых полей.

Практические задания

11.4. Создать формы для баз данных «Записная книжка» и «Библиотечный каталог».

11.4. Обработка данных в БД

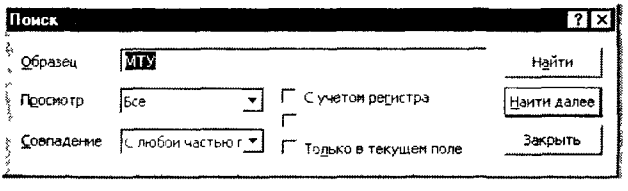
11.4.1. Быстрый поиск данных

Access позволяет производить поиск записей, в которых значения определенного поля полностью или частично совпадают с некоторой величиной.

Например, в БД «Провайдеры Интернета» мы хотим найти запись, содержащую сведения о провайдере МТУ, но мы не помним его полное название. Можно ввести лишь часть названия и осуществить поиск записи.

Быстрый поиск данных в БД «Провайдеры Интернета»

1. Открыть таблицу БД «Провайдеры Интернета», дважды щелкнув по соответствующему значку в окне БД.
2. Ввести команду [Правка-Найти...]. Появится диалоговая панель *Поиск*. В поле *Образец:* необходимо ввести искомый текст, а в поле *Совпадение:* выбрать пункт *С любой частью поля*.



3. В результате будет найдена и отмечена как активная запись № 3.

11.4.2. Поиск данных с помощью фильтров

Гораздо больше возможностей для поиска данных в БД предоставляют *фильтры*. Фильтры позволяют отбирать записи, которые удовлетворяют заданным условиям. Условия отбора записей создаются с использованием операторов сравнения (=, >, < и так далее).

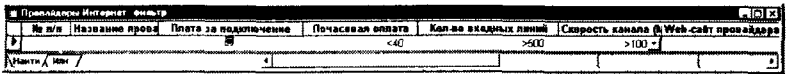
Простые фильтры содержат условие отбора записей только для одного поля. *Сложные фильтры* содержат несколько условий для различных полей. В результате применения сложного фильтра будут отобраны только те записи, которые удовлетворяют всем условиям одновременно. Можно сказать, что условия в сложных фильтрах связаны между собой операцией логического умножения.

Пусть, например, мы будем искать оптимального провайдера, то есть провайдера, который не берет плату за подключение, почасовая оплата достаточно низка (<40 рублей в час), до него легко дозвониться (количество входных линий >500), и он обладает высокоскоростным доступом в Интернет (скорость канала >100 Мбит/с).

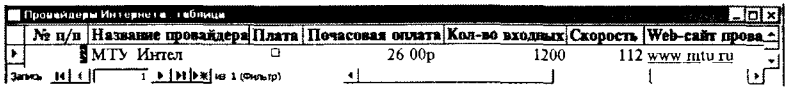
Создадим сложный фильтр для базы данных «Провайдеры Интернета».

Поиск данных с помощью фильтра

1. Открыть таблицу БД «Провайдеры Интернета», дважды щелкнув по соответствующему значку в окне БД.
2. Ввести команду [Записи-Фильтр-Изменить фильтр]. В появившемся окне таблицы ввести условия поиска в соответствующих полях. Фильтр создан.



3. Ввести команду [Записи-Применить фильтр]. В появившемся окне таблицы будут выведены записи, удовлетворяющие условиям поиска. В данном случае найден лишь один такой провайдер — МТУ-Интел.



11.4.3. Поиск данных с помощью запросов

Запросы осуществляют поиск данных в БД так же, как и фильтры. Различие между ними состоит в том, что запросы являются самостоятельными объектами БД, а фильтры привязаны к конкретной таблице.

Запрос является производным объектом от таблицы. Однако результатом выполнения запроса является также таблица, то есть запросы могут использоваться вместо таблиц. Например, форма может быть создана как для таблицы, так и для запроса.

Запросы позволяют отобрать те записи, которые удовлетворяют заданным условиям. Запросы, как и фильтры, бывают простые и сложные. Простой запрос содержит одно условие, а сложный запрос содержит несколько условий для различных полей.

В процессе создания запроса можно отбирать не только записи, но и поля, которые будут присутствовать в запросе.

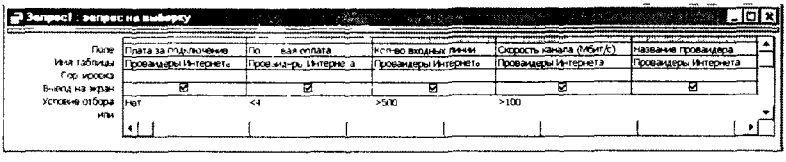
Создадим сложный запрос по выявлению оптимального провайдера в БД «Провайдеры Интернета».

Поиск данных с помощью запроса

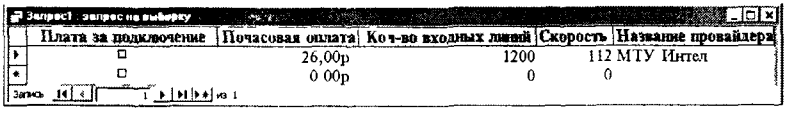
1. В окне *Провайдеры Интернета: база данных* выделить группу объектов *Запросы* и выбрать пункт *Создание запроса с помощью конструктора.*
2. На диалоговой панели *Добавление таблицы* выбрать таблицу «Провайдеры Интернета», для которой создается запрос. Щелкнуть по кнопке *Добавить.*
3. В окне запроса в строке *Поле:* из раскрывающегося списка выбрать имена полей, для которых будут заданы условия.

   В строке *Условие отбора:* ввести условия для выбранных полей.

   В строке *Вывод на экран:* задать поля, которые будут представлены в запросе.



4. Сохранить запрос под именем *Запрос1* с помощью команды [Файл-Сохранить как...].
5. В окне *Провайдеры Интернета: база данных* выделить *Запрос1* и щелкнуть по кнопке *Открыть.* В появившемся окне запроса будут выведены записи, удовлетворяющие условиям поиска. В данном случае найден лишь один такой провайдер — МТУ-Интел.



Практические задания

11.5. Осуществить в базах данных «Записная книжка» и «Библиотечный каталог» различные виды поиска: быстрый, с помощью фильтра и с помощью запроса.

11.6. В базе данных «Провайдеры Интернета» осуществить поиск провайдеров, которые не берут плату за подключение и взимают самую низкую почасовую оплату.

11.4.4. Сортировка данных

Базы данных могут содержать сотни и тысячи записей. Часто бывает необходимо упорядочить записи, то есть расположить в определенной последовательности. Упорядочение записей называется сортировкой.

Сортировка записей производится по какому-либо полю. Значения, содержащиеся в этом поле, располагаются в определенном порядке, который определяется типом поля:

- по алфавиту, если поле текстовое;
- по величине числа, если поле числовое;
- по дате, если тип поля — *Дата/Время* и так далее.

Сортировка записей может производиться либо по возрастанию, либо по убыванию значений поля. В процессе сортировки целостность записей сохраняется, то есть они переносятся из одного места таблицы в другое целиком.

Сортировка записей базы данных — это их упорядочение по значениям одного из полей.

Произведем сортировку в БД «Провайдеры Интернета», например, по полю «Скорость канала (Мбит/с)».

Быстрая сортировка данных

1. В окне *Провайдеры Интернета: база данных* в группе объектов *Таблицы* выделить таблицу «Провайдеры Интернета» и щелкнуть по кнопке *Открыть*.
2. Выделить поле *Скорость канала* и ввести команду [Записи-Сортировка-Сортировка по возрастанию]. Записи в БД будут отсортированы по возрастанию скорости канала.

Провайдеры Интернета : таблица

№ п/п	Название прова	Плата	Почасовая	Кол-во	Скорость канала (Мбит/с)	Web-сайт провайдер
6	Портал	☑	38 00р	250	5	www portal ru
8	Элвис Телеком	☑	40,00р	340	10	www telecom ru
7	Ситек	☑	35,00р	120	10	www sitek ru
1	Демос	☑	44,00р	400	45	www demos ru
5	Караван	☑	35,00р	210	100	www caravan ru
3	МТУ Интел	☐	26,00р	1200	112	www mtu ru
2	Гласнет	☑	44,00р	850	112	www glasnet ru
4	Зенон	☐	52,00р	450	155	www aha ru
(четчик)		☐	0,00р	0	0	

Запись 5

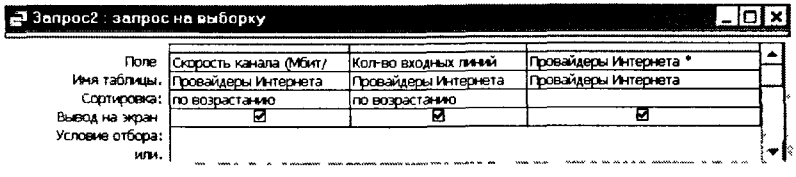
Могут реализовываться *вложенные сортировки*, то есть сортировки, которые последовательно производятся по нескольким полям. После сортировки по первому столбцу производится сортировка по второму столбцу и так далее.

В нашем случае в поле *Скорость канала*, по которому была произведена сортировка, две записи (8 и 7) имеют одинаковое значение 10 и две записи (3 и 2) — одинаковое значение 112. Чтобы упорядочить эти записи, произведем вложенную сортировку, сначала по полю «Скорость канала», а затем по полю «Кол-во входных линий».

Access позволяет выполнять вложенные сортировки с помощью запросов.

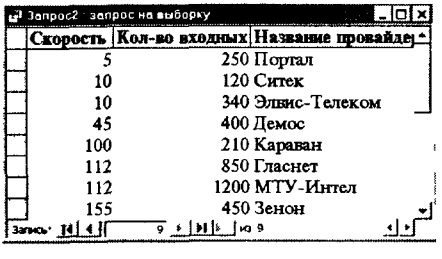
Вложенная сортировка данных с помощью запроса

1. В окне *Провайдеры Интернета: база данных* выделить группу объектов *Запросы* и выбрать пункт *Создание запроса с помощью конструктора*.
2. На диалоговой панели *Добавление таблицы* выбрать таблицу «Провайдеры Интернета», для которой создается запрос. Щелкнуть по кнопке *Добавить*.
3. В окне запроса в строке *Сортировка:* из раскрывающегося списка выбрать имена полей, в которых задать тип сортировки.



В строке *Вывод на экран:* задать поля, которые будут представлены в запросе.

4. Сохранить запрос под именем *Запрос2* с помощью команды [Файл-Сохранить как...].
5. На вкладке *Запросы* выделить *Запрос2* и щелкнуть по кнопке *Открыть*. В появившемся окне запроса будет выведена отсортированная таблица.



Практические задания

11.7. Осуществить в базе данных «Провайдеры Интернета» вложенную сортировку по полям «Почасовая оплата» и «Название провайдера».

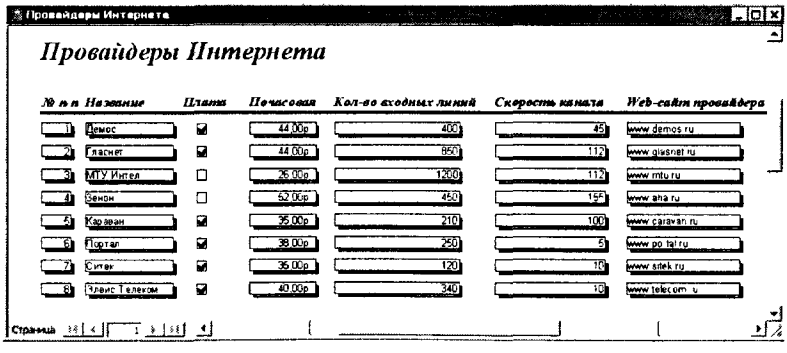
11.4.5. Печать данных с помощью отчетов

Можно осуществлять печать непосредственно таблиц, форм и запросов с помощью команды [Файл-Печать]. Однако для красивой печати документов целесообразно использовать *отчеты*. Отчеты являются производными объектами БД и создаются на основе таблиц, форм и запросов.

Создадим отчет, который будет красиво распечатывать БД «Провайдеры Интернета». Воспользуемся для этого *Мастером отчетов*.

Вывод БД на печать с помощью отчета

1. В окне *Провайдеры Интернета: база данных* выделить группу объектов *Отчеты* и выбрать пункт *Создание отчета с помощью мастера*.
2. С помощью серии диалоговых панелей задать параметры внешнего вида отчета.
3. В окне *Провайдеры Интернета: база данных* щелкнуть по кнопке *Просмотр*. Появится документ в том виде, в котором он может быть распечатан.



4. Если внешний вид документа вас удовлетворяет, распечатать его с помощью команды [Файл-Печать].

Практические задания

11.8. Создать отчет «Визитка» для базы данных «Записная книжка» и отчет «Библиотечная карточка» для базы данных «Библиотечный каталог».

11.5. Реляционные базы данных

11.5.1. Однотабличные и многотабличные базы данных

Достаточно часто встречается ситуация, когда хранить всю базу данных в одной таблице неудобно и нерационально. Таблица может содержать слишком большое количество полей, что неудобно пользователю. Различные записи при этом во многих полях дублируют друг друга, что увеличивает информационный объем базы данных и замедляет процедуры ее обработки.

Поясним это на примере. Пусть табличная база данных «Комплектующие компьютера и поставщики» содержит информацию о различных комплектующих и имеет поля: «Счетчик», «Наименование», «Описание», «Название фирмы», «Адрес», «Цена» (в рублях) — табл. 11.2.

Таблица 11.2. Комплектующие компьютера и поставщики

Счетчик	Наименование	Описание	Название фирмы	Адрес	Цена
1	Системный блок	Pentium	Фирма1	Адрес1	10000
2	Системный блок	Pentium	Фирма2	Адрес2	9000
3	Монитор	15"	Фирма1	Адрес1	5000
4	Монитор	15"	Фирма2	Адрес2	6000
5	Клавиатура	104 кл.	Фирма1	Адрес1	250
6	Клавиатура	104 кл.	Фирма2	Адрес2	300
7	Мышь	3 кн	Фирма1	Адрес1	100
8	Мышь	3 кн	Фирма2	Адрес2	150

Мы видим, что почти половину объема таблицы составляет избыточная, дублированная информация.

Проанализируем причину дублирования. Комплектующие компьютера имеют два неотъемлемых свойства: «Наименование» и «Описание». «Название фирмы», «Адрес» и «Цена» не являются свойствами комплектующих компьютера, они являются свойствами поставщика.

Естественно разделить исходную таблицу на две: «Комплектующие» (табл. 11.3) и «Поставщики» (табл. 11.4).

Каждая таблица должна содержать, по крайней мере, одно *ключевое поле*, содержимое которого уникально для каждой записи в этой таблице. В таблицу «Комплектующие» введем поле «Код комплектующих». Именно это поле будет ключевым в данной таблице.

Таблица 11.3. Комплектующие

Код комплектующих	Наименование	Описание
К1	Системный блок	Pentium
К2	Монитор	15"
К3	Клавиатура	104 кл.
К4	Мышь	3 кн.

В таблицу «Поставщики» введем дополнительное поле «Код поставщика». Именно это поле будет ключевым в данной таблице.

Таблица 11.4. Поставщики

Код поставщика	Название фирмы	Адрес
П1	Фирма1	Адрес1
П2	Фирма2	Адрес2

11.5.2. Связывание таблиц

После создания различных таблиц, содержащих данные, относящиеся к различным аспектам базы данных, необходимо обеспечить целостность базы данных. Для этого надо *связать* таблицы между собой.

При связи «один-ко-многим» каждой записи в одной (главной) таблице могут соответствовать несколько записей в другой (подчиненной) таблице, а запись в подчиненной таблице не может иметь более одной соответствующей ей записи в главной таблице.

Если одной записи в первой таблице могут соответствовать несколько записей во второй таблице и, наоборот, одной записи во второй таблице — несколько записей в первой таблице, то реализуется связь «многие-ко-многим».

В нашем случае реализуется именно такая связь. Одной записи в таблице «Комплектующие» соответствуют две записи в таблице «Поставщики», так как устройства одного типа продаются двумя фирмами. Одной же записи таблицы «Поставщики» соответствуют четыре записи таблицы «Комплектующие», так как одна фирма продает устройства четырех типов.

Две таблицы, находящиеся в отношении «многие-ко-многим», могут быть связаны только с помощью третьей (связующей) таблицы. Таблицы «Комплектующие» и «Поставщики» можно связать в отношении «многие-ко-многим» путем создания двух связей «один-ко-многим» по отношению к таблице «Цена».

Таблицы «Комплектующие» и «Поставщики» будут являться главными по отношению к таблице «Цена».

Связь между таблицами устанавливает отношения между совпадающими значениями в полях с одинаковыми именами. С ключевым полем главной таблицы (*первичный ключ*) связывается одноименное поле подчиненной таблицы (*внешний ключ*).

В главной таблице «Комплектующие» поле «Код комплектующих» является первичным ключом, соответственно в подчиненной таблице «Цена» должно существовать одноименное поле, которое является внешним ключом.

Таблица «Поставщики» также является главной по отношению к таблице «Цена». Ее поле «Код поставщика» является первичным ключом, соответственно в подчиненной таблице «Цена» должно существовать одноименное поле, которое является внешним ключом.

Таким образом, таблица «Цена» должна содержать следующие поля (табл. 11.5):

- «Счетчик» (ключевое поле);
- «Код комплектующих» (поле внешнего ключа для таблицы «Комплектующие»);
- «Код поставщика» (поле внешнего ключа для таблицы «Поставщики»);
- «Цена» (числовое поле).

Таблица 11.5. Цена

Счетчик	Код комплектующих	Код поставщика	Цена
1	К1	П1	9000
2	К1	П2	10000
3	К2	П1	5000
4	К2	П2	6000
5	К3	П1	250
6	К3	П2	300
7	К4	П1	100
8	К4	П2	150

Межтабличная связь *обеспечивает целостность данных.* Связанные таблицы представляют собой единую базу данных, в которой можно создавать новые таблицы, а также запросы и отчеты, содержащие данные из связанных таблиц.

Базы данных, состоящие из связанных двумерных таблиц, принято называть **реляционными.**

Прежде чем приступить к созданию реляционной базы данных, необходимо продумать ее *проект*. Проект представляет собой модель будущей БД, состоящей из объектов и их связей, необходимых для выполнения поставленных задач.

Процесс проектирования включает, прежде всего, определение перечня необходимых таблиц и задание их структуры, а также установление типа связей между этими таблицами.

Вопросы для размышления

1. Почему в некоторых случаях целесообразно использовать многотабличные, а не однотабличные базы данных?
2. Какие типы связей между таблицами возможны в реляционных базах данных?

Практические задания

11.9. Разработать проект реляционной базы данных «Коллекция аудиозаписей», которая бы содержала главную таблицу «Список аудио-CD» и подчиненную таблицу «Содержание аудио-CD».

11.6. Создание реляционной базы данных

Система управления реляционными базами данных Microsoft Access позволяет создавать реляционные базы данных, а также обеспечивать их обработку с помощью запросов, форм и отчетов.

Создадим реляционную базу данных «Компьютер», в качестве основных объектов которой будут использованы три таблицы: «Комплектующие», «Поставщики» и «Цена». Таблицы «Комплектующие» и «Поставщики» должны быть связаны отношением «многие-ко-многим» с помощью таблицы «Цена».

Итак, прежде всего необходимо создать три таблицы: «Комплектующие» «Поставщики» и «Цена».

Создание реляционной базы данных «Компьютеры»

1. Создать в приложении Access новую базу данных с помощью команды [Файл-Создать базу данных...] и присвоить ей имя «Компьютеры».
2. В окне *Компьютеры: база данных* выбрать группу объектов *Таблицы* и пункт *Создание таблицы в режиме конструктора*.

Таблица «Комплектующие» должна содержать три текстовых поля: «Код комплектующих», «Наименование» и «Описание». Ключевым полем является поле «Код комплектующих».

3. В окне *Комплектующие: таблица* ввести имена полей, тип данных и параметры полей.
   В качестве ключевого поля задать поле «Код комплектующих».
   Для сохранения таблицы ввести команду [Файл-Сохранить].
   Присвоить таблице имя «Комплектующие».
4. Для ввода данных в таблицу щелкнуть на значке *Комплектующие*. Ввести данные в таблицу.

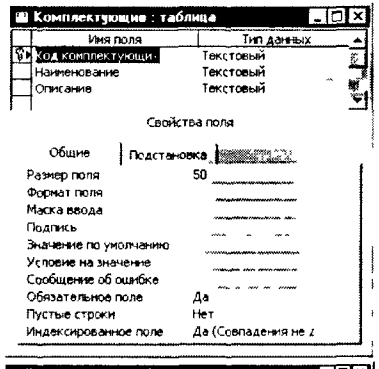


Таблица «Поставщики» должна содержать три текстовых поля: «Код поставщика», «Название фирмы» и «Адрес». Ключевым полем является поле «Код поставщика».

5. Создать таблицу «Поставщики», выполнив рассмотренную выше последовательность действий. Ввести данные.

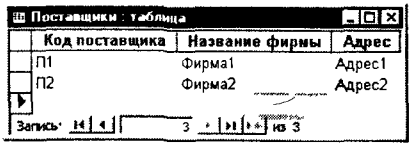
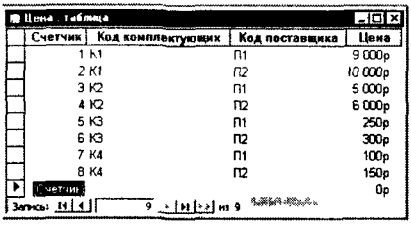


Таблица «Цена» должна содержать поля *Счетчик*, *Код комплектующих*, *Код поставщика*, а также поле *Цена*. В качестве ключа этой таблицы будет использоваться поле *Счетчик*.

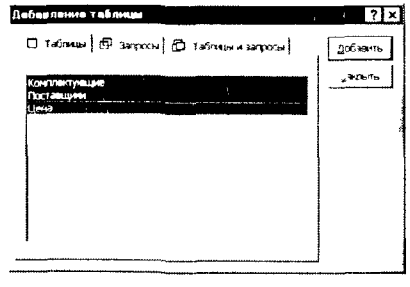
6. С помощью аналогичных действий создать таблицу «Цена» и ввести данные. В полях внешних ключей не могут содержаться значения, отсутствующие в соответствующих ключевых полях главных таблиц.



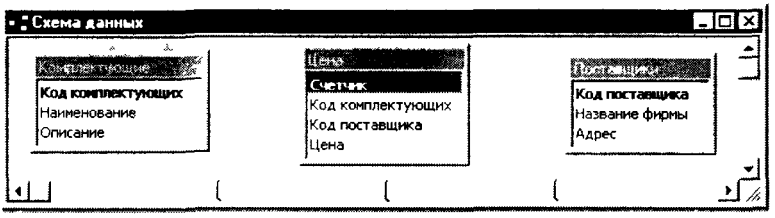
Таблицы «Комплектующие» и «Поставщики» должны быть связаны отношением «один-ко-многим» с таблицей «Цена». Таблица «Цена» содержит однотипные с полями первых двух таблиц поля «Код комплектующих» и «Код поставщика», являющиеся внешними ключами исходных таблиц.

Установим связи между таблицами с помощью окна *Схема данных*.

7. Ввести команду [Сервис-Схема данных]. Появится диалоговая панель *Добавление таблицы*. Выделить в этом окне нужные таблицы и щелкнуть по кнопке *Добавить*.



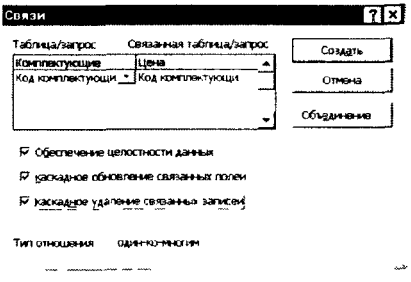
8. Выделенные таблицы будут добавлены в специальное окно — *Схема данных*.



Для установки между таблицами «Комплектующие» и «Поставщики» связи в отношении «многие-ко-многим» необходимо связать их с таблицей «Цена» в отношении «один-ко-многим».

9. Перетащить мышью из таблицы «Комплектующие» ключевое поле — «Код комплектующих» (оно выделено жирным шрифтом) к одноименному полю внешнего ключа таблицы «Цена».

10. На появившейся диалоговой панели *Связи* установить опцию *Обеспечение целостности данных*, а затем опции *каскадное обновление связанных полей* и *каскадное удаление связанных записей*. В завершение щелкнуть по кнопке *Создать*.



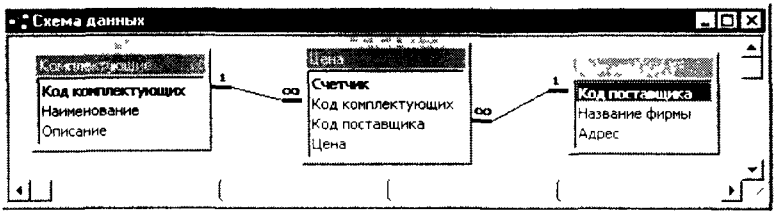
Теперь установим связь в отношении «один-ко-многим» между таблицами «Поставщики» и «Цена».

11. Перетащить мышью из таблицы «Поставщики» ключевое поле — «Код поставщика» (оно выделено жирным шрифтом) к одноименному полю — внешнему ключу таблицы «Цена».

12. На появившейся диалоговой панели *Связи* установить опцию *Обеспечение целостности данных*, а затем опции *каскадное обновление связанных полей* и *каскадное удаление связанных записей*. В завершение щелкнуть на кнопке *Создать*.

Теперь связь в отношении «многие-ко-многим» между таблицами «Комплектующие» и «Поставщики» через таблицу «Цена» установлена.

13. Это наглядно представлено в окне *Схема данных*.

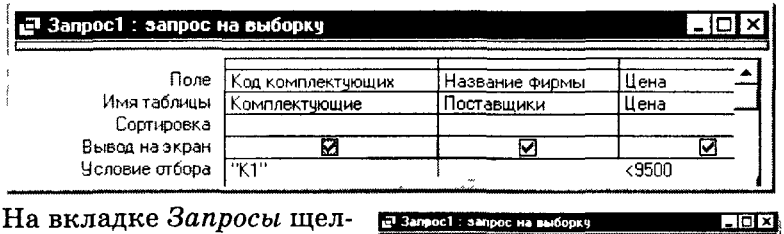


Созданная реляционная база данных «Компьютеры» состоит из трех связанных таблиц и поэтому обладает целостностью данных. Это значит, что можно создавать запросы, формы и отчеты, которые используют данные из разных таблиц.

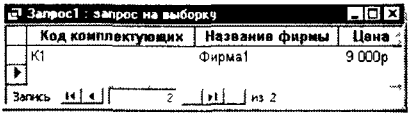
Создадим, например, запрос, который осуществляет выбор информации, необходимой для закупки дешевого системного блока.

Создание запроса в реляционной базе данных «Компьютеры»

1. На диалоговой панели *Новый запрос* выбрать опцию *Конструктор* и щелкнуть по кнопке *ОК*.
2. В таблице «Комплектующие» для поля «Код комплектующих» ввести условие «К1», в таблице «Поставщики» для поля «Название фирмы» установить вывод на экран, в таблице «Цена» для поля «Цена» ввести условие <9500.



3. На вкладке *Запросы* щелкнуть по кнопке *Открыть*. Появится результат выполнения запроса.



Практические задания

11.10. В реляционной базе данных «Провайдеры Интернета» создать запрос на поиск наиболее дешевого монитора.

11.11. В соответствии с разработанным ранее проектом создать реляционную базу данных «Коллекция аудиозаписей».

Глава 12

Коммуникационные технологии

12.1. Передача информации

Обмен информацией производится по каналам передачи информации. Каналы передачи информации могут использовать различные физические принципы. Так, при непосредственном общении людей информация передается с помощью звуковых волн, а при разговоре по телефону — с помощью электрических сигналов, которые распространяются по линиям связи. Компьютеры могут обмениваться информацией с использованием каналов связи различной физической природы: кабельных, оптоволоконных, радиоканалов и др.

Общая схема передачи информации включает в себя отправителя информации, канал передачи информации и получателя информации (рис. 12.1). Если производится двусторонний обмен информацией, то отправитель и получатель информации могут меняться ролями.

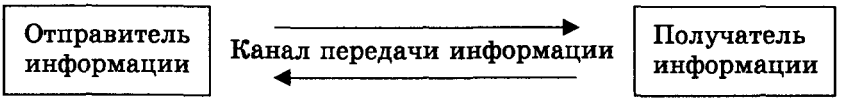


Рис. 12.1. Канал обмена информацией

Основной характеристикой каналов передачи информации является их пропускная способность (скорость передачи информации). Пропускная способность канала равна количеству информации, которое может передаваться по нему в единицу времени.

Обычно пропускная способность измеряется в битах в секунду (бит/с) и кратных единицах Кбит/с и Мбит/с. Однако иногда в качестве единицы измерения используется байт в секунду (байт/с) и кратные ему единицы Кбайт/с и Мбайт/с.

Соотношения между единицами пропускной способности канала передачи информации такие же, как между единицами измерения количества информации:

1 байт/с = 2^3 бит/с = 8 бит/с;
1 Кбит/с = 2^{10} бит/с = 1024 бит/с;
1 Мбит/с = 2^{10} Кбит/с = 1024 Кбит/с;
1 Гбит/с = 2^{10} Мбит/с = 1024 Мбит/с.

Задания

12.1. Какое количество байтов будет передаваться за одну секунду по каналу с пропускной способностью 100 Мбит/с?

12.2. Локальные компьютерные сети

При работе на персональном компьютере в автономном режиме пользователи могут обмениваться информацией (программами, документами и так далее), лишь копируя ее на дискеты. Однако перемещение дискеты между компьютерами не всегда возможно и может занимать достаточно продолжительное время.

Создание компьютерных сетей вызвано практической потребностью совместного использования информации пользователями, работающими на удаленных друг от друга компьютерах. Сети предоставляют пользователям возможность не только быстрого обмена информацией, но и совместного использования принтеров и других периферийных устройств и даже одновременной работы с документами.

Локальная сеть объединяет компьютеры, установленные в одном помещении (например, школьный компьютерный класс, состоящий из 8–12 компьютеров) или в одном здании (например, в здании школы могут быть объединены в локальную сеть несколько десятков компьютеров, установленных в различных предметных кабинетах).

Локальная сеть объединяет несколько компьютеров и дает возможность пользователям совместно использовать ресурсы компьютеров, а также подключенных к сети периферийных устройств (принтеров, плоттеров, дисков, модемов и др.).

В небольших локальных сетях все компьютеры обычно равноправны, то есть пользователи самостоятельно решают, какие ресурсы своего компьютера (диски, каталоги, файлы) сделать общедоступными по сети. Такие сети называются *одноранговыми*.

Если к локальной сети подключено более 10 компьютеров, одноранговая сеть может оказаться недостаточно производительной. Для увеличения производительности, а также в целях обеспечения большей надежности при хранении информации в сети некоторые компьютеры специально выделяются для хранения файлов и программных приложений. Такие компьютеры называются *серверами*, а локальная сеть — *сетью на основе сервера*.

Аппаратное обеспечение сети. Каждый компьютер, подключенный к локальной сети, должен иметь специальную плату (сетевой адаптер — рис.12.2).

Основной функцией сетевого адаптера является передача и прием информации из сети. В настоящее время наиболее часто используются сетевые адаптеры типа EtherNet, которые могут объединять в сеть компьютеры различных аппаратных и программных платформ (IBM-совместимые, Macintosh, Unix-компьютеры).

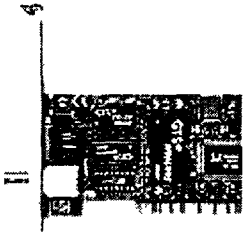


Рис. 12.2. Сетевой адаптер

Соединение компьютеров (сетевых адаптеров) между собой производится с помощью кабелей различных типов (*коаксиального, витой пары, оптоволоконного*). Для подключения к локальной сети портативных компьютеров часто используется беспроводное подключение, при котором передача данных осуществляется с помощью электромагнитных волн.

Важнейшей характеристикой локальных сетей, которая определяется типом используемых сетевых адаптеров и кабелей, является скорость передачи информации по сети. Скорость передачи информации по локальной сети обычно находится в диапазоне от 10 до 100 Мбит/с.

Топология сети. Общая схема соединения компьютеров в локальной сети называется *топологией сети*. Топологии сети могут быть различными.

Вариант соединения компьютеров между собой, когда кабель проходит от одного компьютера к другому, последовательно соединяя компьютеры и периферийные устройства между собой, называется *линейной шиной* (рис. 12.3).

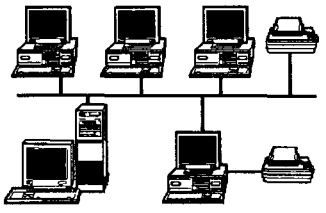


Рис. 12.3. Локальная сеть типа «линейная шина»

Если к каждому компьютеру подходит отдельный кабель из одного центрального узла, то реализуется локальная сеть типа *«звезда»*.

Обычно при такой схеме соединения центральным узлом является более мощный компьютер.

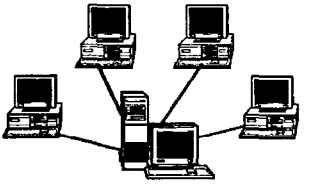


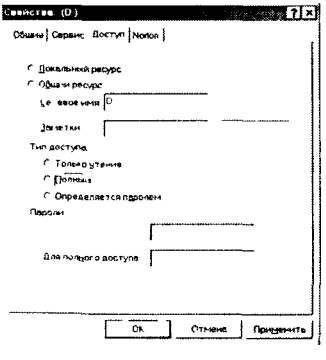
Рис. 12.4. Локальная сеть типа «звезда»

Преимущество локальной сети типа «звезда» перед локальной сетью типа «линейная шина» состоит в том, что при выходе из строя сетевого кабеля у одного компьютера локальная сеть в целом продолжает нормально функционировать.

Предоставление доступа к ресурсам компьютера. В операционной системе Windows пользователь любого компьютера, подключенного к сети, может предоставить доступ к своим дискам, папкам или файлам. Пользователи, работающие за другими компьютерами, могут после этого пользоваться предоставленными ресурсами.

Предоставление доступа к ресурсам компьютера

1. В контекстном меню объекта (диск, файл, папка) необходимо выбрать команду *Доступ*.
2. На появившейся диалоговой панели *Свойства:* выбрать вкладку *Доступ*.
   С помощью переключателей установить *Общий ресурс*, а также выбрать тип доступа (*Только чтение, Полный, Определяется паролем*).



В текстовом окне *Для полного доступа:* можно ввести пароль, необходимый для доступа к данному ресурсу.

Вопросы для размышления

1. Какую топологию целесообразно использовать в локальной сети компьютерного класса? Локальной сети учебного заведения?

Практические задания

12.2. Предоставить полный доступ к каталогу Мои документы на сервере локальной сети.

12.3. Глобальная компьютерная сеть Интернет

Локальные сети обычно объединяют несколько десятков компьютеров, размещенных в одном здании, однако они не позволяют обеспечить совместный доступ к информации пользователям, находящимся, например, в различных частях города. На помощь приходят *региональные сети*, объединяющие компьютеры в пределах одного региона (города, страны, континента).

Многие организации, заинтересованные в защите информации от несанкционированного доступа (например, военные, банковские и пр.), создают собственные, так называемые *корпоративные сети*. Корпоративная сеть может объединять тысячи и десятки тысяч компьютеров, размещенных в различных странах и городах (в качестве примера можно привести сеть корпорации Microsoft — MicroSoft Network (MSN)).

Потребности формирования единого мирового информационного пространства привели к созданию глобальной компьютерной сети Интернет. В настоящее время на более чем 150 миллионах компьютеров, подключенных к Интернету, хранится громадный объем информации (сотни миллионов файлов, документов и так далее). Глобальная сеть Интернет привлекает пользователей своими информационными ресурсами и сервисами (услугами), которыми пользуется около миллиарда человек во всех странах мира.

Интернет — это глобальная компьютерная сеть, объединяющая многие *локальные, региональные* и *корпоративные сети* и включающая сотни миллионов компьютеров.

В каждой такой локальной или корпоративной сети обычно имеется, по крайней мере, один компьютер, который имеет постоянное подключение к Интернету с помощью линии связи с высокой пропускной способностью (сервер Интернета). В качестве таких «магистральных» линий связи обычно используются оптоволоконные линии с пропускной способностью до 20 Гбит/с и более.

Надежность функционирования глобальной сети обеспечивает большое количество линий связи между региональными сегментами сети. Например, российский региональный сегмент Интернета имеет несколько магистральных линий связи, соединяющих его с североамериканским, европейским и японским сегментами.

Основу, «каркас» Интернета составляют более 150 миллионов серверов, постоянно подключенных к сети, из которых в России насчитывается около 400 тысяч (на начало 2002 г.).

К серверам Интернета могут подключаться с помощью локальных сетей или коммутируемых телефонных линий сотни миллионов пользователей Интернета (рис. 12.5).

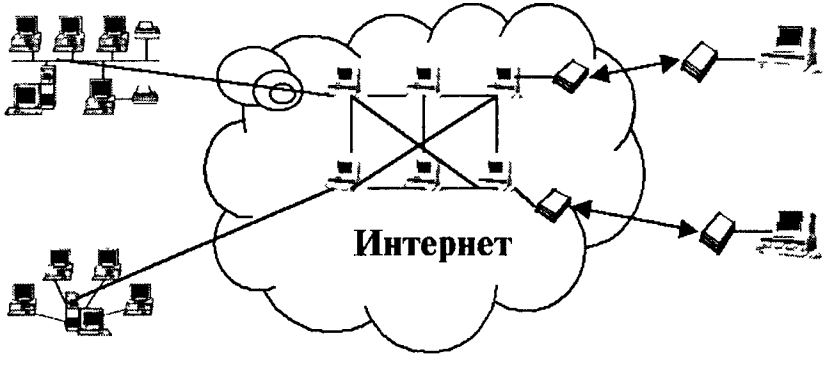


Рис. 12.5. Структура глобальной сети Интернет

Вопросы для размышления

1. Почему глобальная компьютерная сеть Интернет продолжает нормально функционировать даже после выхода из строя отдельных серверов и линий связи?

12.4. Адресация в Интернете

IP-адрес. Для того чтобы в процессе обмена информацией компьютеры могли найти друг друга, в Интернете существует единая система адресации, основанная на использовании IP-адреса.

Каждый компьютер, подключенный к Интернету, имеет свой уникальный 32-битный (в двоичной системе) **IP-адрес**.

По формуле (2.1) легко подсчитать, что общее количество различных IP-адресов составляет более 4 миллиардов:

$N = 2^{32} = 4\ 294\ 967\ 296.$

Система IP-адресации учитывает структуру Интернета, то есть то, что Интернет является сетью сетей, а не объединением отдельных компьютеров. IP-адрес содержит адрес сети и адрес компьютера в данной сети.

Для обеспечения максимальной гибкости в процессе распределения IP-адресов, в зависимости от количества компьютеров в сети, адреса разделяются на три класса A, B, C. Первые биты адреса отводятся для идентификации класса, а остальные разделяются на адрес сети и адрес компьютера (табл. 12.1).

Таблица 12.1. IP-адресация в сетях различных классов

Класс A	0	Адрес сети (7 битов)	Адрес компьютера (24 бита)		
Класс B	1	0	Адрес сети (14 битов)	Адрес компьютера (16 битов)	
Класс C	1	1	0	Адрес сети (21 бит)	Адрес компьютера (8 битов)

Например, адрес сети класса А имеет только 7 битов для адреса сети и 24 бита для адреса компьютера, то есть может существовать лишь $2^7 = 128$ сетей этого класса, зато в каждой сети может содержаться $2^{24} = 16\ 777\ 216$ компьютеров.

В десятичной записи IP-адрес состоит из 4 чисел, разделенных точками, каждое из которых лежит в диапазоне от 0 до 255. Например, IP-адрес сервера компании МТУ-Интел записывается как 195.34.32.11.

Достаточно просто определить по первому числу IP-адреса компьютера его принадлежность к сети того или иного класса:

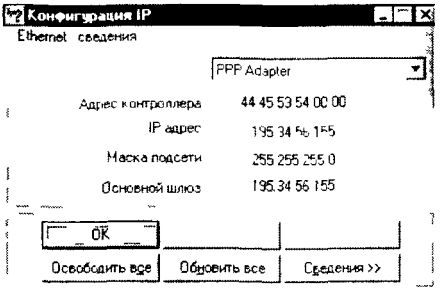
- адреса класса А — число от 0 до 127;
- адреса класса В — число от 128 до 191;
- адреса класса С — число от 192 до 223.

Так, сервер компании МТУ-Интел относится к сети класса С, адрес которой 195, а адрес компьютера в сети 34.32.11.

Провайдеры часто предоставляют пользователям доступ в Интернет не с постоянным, а с динамическим IP-адресом, который может меняться при каждом подключении к сети. В процессе сеанса работы в Интернете можно определить свой текущий IP-адрес.

Определение IP-адреса компьютера

1. Соединиться с Интернетом, ввести команду [Программы-Сеанс MS-DOS].
2. В окне *Сеанс MS-DOS* в ответ на приглашение системы ввести команду winipcfg.
   Появится диалоговая панель *Конфигурация IP*, на которой имеется полная информация о параметрах текущего подключения к Интернету, в том числе и IP-адрес вашего компьютера.



Доменная система имен. Компьютеры легко могут найти друг друга по числовому IP-адресу, однако человеку запомнить числовой адрес нелегко, и для удобства была введена *Доменная Система Имен* (DNS — Domain Name System).

Доменная система имен ставит в соответствие числовому IP-адресу компьютера уникальное доменное имя.

Доменные имена и IP-адреса распределяются международным координационным центром доменных имен и IP-адресов (ICANN), в который входят по 5 представителей от каждого континента (адрес в Интернете www.icann.org).

Доменная система имен имеет иерархическую структуру: домены верхнего уровня — домены второго уровня и так далее. Домены верхнего уровня бывают двух типов: географические (двухбуквенные — каждой стране соответствует двухбуквенный код) и административные (трехбуквенные) (табл. 12.2).

России принадлежит географический домен ru. Интересно, что давно существующие серверы могут относиться к домену su (СССР). Обозначение административного домена позволяет определить профиль организации, владельца домена.

Таблица 12.2. Некоторые имена доменов верхнего уровня

Административные	Тип организации	Географические	Страна
com	Коммерческая	ca	Канада
edu	Образовательная	de	Германия
gov	Правительственная США	jp	Япония
int	Международная	ru	Россия
mil	Военная США	su	бывший СССР
net	Компьютерная сеть	uk	Англия /Ирландия
org	Некоммерческая	us	США

Так, компания Microsoft зарегистрировала домен второго уровня microsoft в административном домене верхнего уровня com, а Московский институт открытого образования (МИОО) — домен второго уровня metodist в географическом домене верхнего уровня ru.

Имена компьютеров, которые являются серверами Интернета, включают в себя полное доменное имя и собственно имя компьютера. Так, основной сервер компании Microsoft имеет имя www.microsoft.com, а сервер компании МИОО — iit.metodist.ru (рис.12.6).

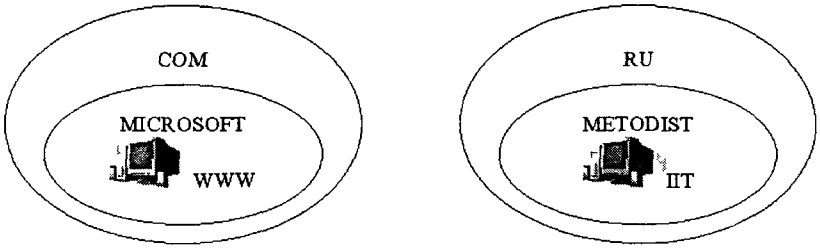


Рис. 12.6. Доменная система имен

Вопросы для размышления

1. Имеет ли каждый компьютер, подключенный к Интернету IP-адрес? Доменное имя?

Практические задания

12.3. Подсчитайте максимальное количество сетей класса B и максимальное количество адресов компьютеров в сети класса C.

12.4. Определите, к какому классу адресов относится IP-адрес вашего компьютера после подключения к Интернету.

12.5. Протокол передачи данных TCP/IP

Сеть Интернет, являющаяся сетью сетей и объединяющая громадное количество различных локальных, региональных и корпоративных сетей, функционирует и развивается благодаря использованию единого протокола передачи данных TCP/IP. Термин TCP/IP включает название двух протоколов:

- Transmission Control Protocol (TCP) — транспортный протокол;
- Internet Protocol (IP) — протокол маршрутизации.

Протокол маршрутизации. Протокол IP обеспечивает передачу информации между компьютерами сети. Рассмотрим

работу данного протокола по аналогии с передачей информации с помощью обычной почты. Для того чтобы письмо дошло по назначению, на конверте указывается адрес получателя (кому письмо) и адрес отправителя (от кого письмо).

Аналогично передаваемая по сети информация «упаковывается в конверт», на котором «пишутся» IP-адреса компьютеров получателя и отправителя, например «Кому: 198.78.213.185», «От кого: 193.124.5.33». Содержимое конверта на компьютерном языке называется *IP-пакетом* и представляет собой набор байтов.

В процессе пересылки обыкновенных писем они сначала доставляются на ближайшее к отправителю почтовое отделение, а затем передаются по цепочке почтовых отделений на ближайшее к получателю почтовое отделение. На промежуточных почтовых отделениях письма сортируются, то есть определяется, на какое следующее почтовое отделение необходимо отправить то или иное письмо.

IP-пакеты на пути к компьютеру-получателю также проходят через многочисленные промежуточные серверы Интернета, на которых производится операция *маршрутизации*. В результате маршрутизации IP-пакеты направляются от одного сервера Интернета к другому, постепенно приближаясь к компьютеру-получателю.

Internet Protocol (IP) обеспечивает маршрутизацию IP-пакетов, то есть доставку информации от компьютера-отправителя к компьютеру-получателю.

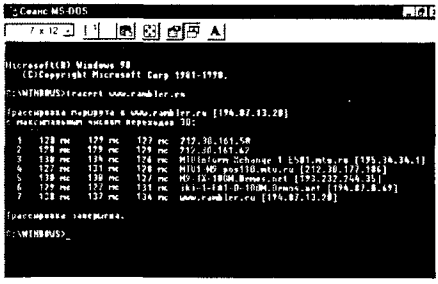
Определение маршрута прохождения информации. «География» Интернета существенно отличается от привычной нам географии. Скорость получения информации зависит не от удаленности Web-сервера, а от количества промежуточных серверов и качества линий связи (их пропускной способности), по которым передается информация от узла к узлу.

С маршрутом прохождения информации в Интернете можно познакомиться достаточно просто. Специальная программа tracert.exe, которая входит в состав Windows, позволяет проследить, через какие серверы и с какой задержкой передается информация с выбранного сервера Интернет на ваш компьютер.

Проследим, как реализуется доступ к информации в «московской» части Интернета к одному из наиболее популярных поисковых серверов российского Интернета www.rambler.ru.

Определение маршрута прохождения информации

1. Соединиться с Интернетом, ввести команду [Программы-Сеанс MS-DOS].
2. В окне *Сеанс MS-DOS* в ответ на приглашение системы ввести команду [tracert www.rambler.ru].
3. Через некоторое время появится трассировка передачи информации, то есть список узлов, через которые передается информация на ваш компьютер, и время передачи между узлами.



Трассировка маршрута передачи информации показывает, что сервер www.rambler.ru находится от нас на «расстоянии» 7 переходов, т. е. информация передается через шесть промежуточных серверов Интернета (через серверы московских провайдеров МТУ-Информ и Демос). Скорость передачи информации между узлами достаточно высока, на один «переход» тратится от 126 до 138 мс.

Транспортный протокол. Теперь представим себе, что нам необходимо переслать по почте многостраничную рукопись, а почта бандероли и посылки не принимает. Идея проста: если рукопись не помещается в обычный почтовый конверт, ее надо разобрать на листы и переслать их в нескольких конвертах. При этом листы рукописи необходимо обязательно пронумеровать, чтобы получатель знал, в какой последовательности потом эти листы соединить.

В Интернете часто случается аналогичная ситуация, когда компьютеры обмениваются большими по объему файлами. Если послать такой файл целиком, то он может надолго «закупорить» канал связи, сделать его недоступным для пересылки других сообщений.

Для того чтобы этого не происходило, на компьютере-отправителе необходимо разбить большой файл на мелкие части, пронумеровать их и транспортировать в отдельных

IP-пакетах до компьютера-получателя. На компьютере-получателе необходимо собрать исходный файл из отдельных частей в правильной последовательности.

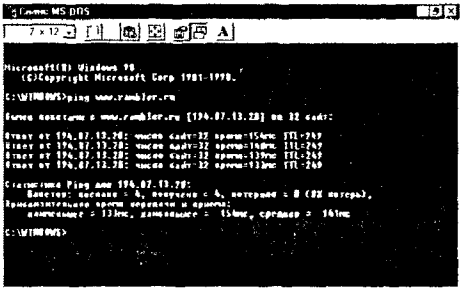
Transmission Control Protocol (TCP), то есть транспортный протокол, обеспечивает разбиение файлов на IP-пакеты в процессе передачи и сборку файлов в процессе получения.

Интересно, что для IP-протокола, ответственного за маршрутизацию, эти пакеты совершенно никак не связаны между собой. Поэтому последний IP-пакет вполне может по пути обогнать первый IP-пакет. Может сложиться так, что даже маршруты доставки этих пакетов окажутся совершенно разными. Однако протокол TCP дождется первого IP-пакета и соберет исходный файл в правильной последовательности.

Определение времени обмена IP-пакетами. Время обмена IP-пакетами между локальным компьютером и сервером Интернета можно определить с помощью утилиты ping, которая входит в состав операционной системы Windows. Утилита посылает четыре IP-пакета по указанному адресу и показывает суммарное время передачи и приема для каждого пакета.

Определение времени обмена IP-пакетами

1. Соединиться с Интернетом, ввести команду [Программы-Сеанс MS-DOS].
2. В окне *Сеанс MS-DOS* в ответ на приглашение системы ввести команду [ping www.rambler.ru].
3. В окне *Сеанс MS-DOS* высветится результат пробного прохождения сигнала в четырех попытках. Время отклика характеризует скоростные параметры всей цепочки линий связи от сервера до локального компьютера.



Вопросы для размышления

1. Что обеспечивает целостное функционирование глобальной компьютерной сети Интернет?

Практические задания

12.5. Проследить маршрут прохождения информации от одного из наиболее популярных поисковых серверов Интернета www.yahoo.com, расположенного в «американском» сегменте Интернета.

12.6. Определить время обмена IP-пакетами с сервером www.yahoo.com.

12.6. Подключение к Интернету по коммутируемым телефонным каналам

Существует несколько различных способов подключения к Интернету, которые различаются предоставляемыми пользователю возможностями и стоимостью подключения. Наилучшие возможности обеспечиваются при непосредственном подключении к Интернету с помощью высокоскоростного (оптоволоконного или спутникового) канала связи. Однако такое подключение достаточно дорого и обычно используется большими организациями для подключения локальных сетей.

Провайдеры услуг Интернета имеют высокоскоростное соединение своих серверов с Интернетом, что позволяет им предоставлять пользователям доступ к Интернету на коммерческой основе по коммутируемым телефонным каналам.

12.6.1. Модем

Модуляция и демодуляция. Осуществлять передачу информации по коммутируемым телефонным линиям компьютеры не могут, так как обмениваются данными с помощью цифровых электронных импульсов, а по телефонной линии можно передавать только аналоговые (непрерывные) сигналы.

Для подключения компьютера к телефонной линии используется *модем* (рис. 12.7). На передающей стороне реализуется *модуляция* аналогового электрического сигнала определенной частоты (несущей) последовательностями электрических импульсов. Компьютер посылает модему последовательности электрических импульсов, а модем преобразует цифровые сигналы компьютера в модулированный аналоговый сигнал (рис. 12.8).

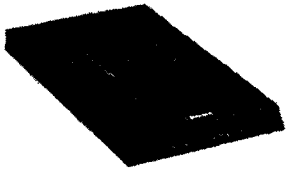


Рис. 12.7. Модем

Простейшим случаем модуляции, известным из курса физики, является амплитудная модуляция, в этом случае несущий аналоговый сигнал с постоянной амплитудой в процессе модуляции преобразуется в аналоговый сигнал с переменной амплитудой.

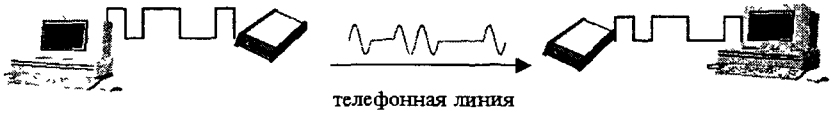
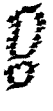


Рис. 12.8. Подключение по телефонной линии с помощью модема

Модулированный аналоговый сигнал передается по телефонной линии. На принимающей стороне модем производит обратное преобразование — *демодуляцию*, то есть преобразует входящий аналоговый сигнал в последовательность цифровых импульсов.



Модем обеспечивает **модуляцию** и **демодуляцию** сигнала при его передаче по телефонным линиям.

Модемы различаются по конструктивному исполнению на внутренние и внешние. Внутренние модемы устанавливаются в один из слотов системной платы, а внешние подключаются к последовательному порту компьютера.

Основной характеристикой качества модема является скорость передачи информации, которую он может обеспечить в линии. В настоящее время наибольшее распространение имеют модемы, обеспечивающие скорость передачи информации 33,6 Кбит/с и 56 Кбит/с.

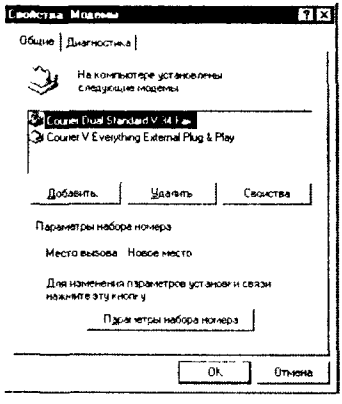
Скорость передачи информации определяется используемым протоколом модемной связи (в данном случае протоколами V.34+ и V.90), которые разрабатываются и утверждаются Международным телекоммуникационным союзом. Правда, до настоящего времени используются и модемы, поддерживающие скорость передачи 56 Кбит/с с нестандартными и несовместимыми друг с другом протоколами x2 и K56flex.

Установка и тестирование модема. Прежде всего, необходимо убедиться в правильности подключения модема. Если вы используете внутренний модем, то он должен быть правильно установлен в слот расширения системной платы, если внешний, то он должен быть подключен к блоку питания и к последовательному порту компьютера. В обоих случаях модем должен быть подключен к телефонной линии.

После этого необходимо запустить программу Установка оборудования, которая определит марку модема и установит соответствующий драйвер. Для проверки правильности проведенной установки модема необходимо его протестировать.

Тестирование модема

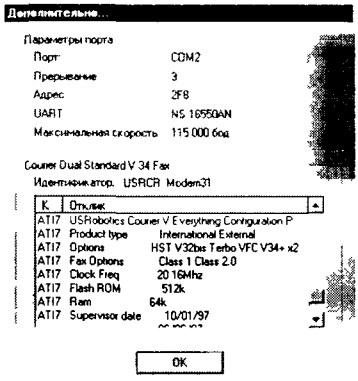
1. Открыть папку *Панель управления* и активизировать значок *Модемы*. Появится диалоговая панель *Свойства: Модемы*.
   Выбрать из списка подключенный модем, активизировать вкладку *Диагностика* и щелкнуть по кнопке *Дополнительно*.



Начнется тестирование модема, которое может занять несколько десятков секунд. В течение этого времени должны мигать лампочки на панели управления внешнего модема.

В процессе тестирования будут определены различные технические параметры модема, и в частности протоколы передачи данных, которые поддерживает данный модем.

2. На открывшейся информационной панели *Дополнительно* в списке найти пункт *Options* и определить, какие протоколы поддерживает установленный модем.
   Для данного модема наиболее высокоскоростным протоколом является протокол x2, который обеспечивает скорость передачи данных 56 Кбит/с.

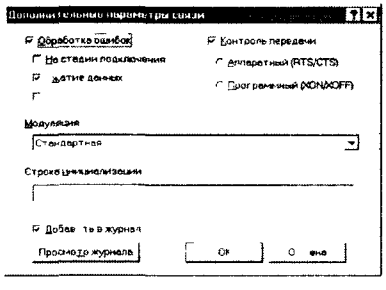


Сжатие данных и коррекция ошибок. В целях сокращения времени передачи данных по линиям связи необходимо осуществлять *сжатие данных*. На передающей стороне получаемые из последовательного порта компьютера данные модем по определенным алгоритмам сжимает, а на принимающей восстанавливает в исходном виде.

В процессе передачи данных модем реализует также *коррекцию ошибок*. Это очень важно, так как при передаче программного файла ошибка только в одном бите может привести к неработоспособности программы. Если в процессе передачи блока данных произошла ошибка, модем на принимающей стороне отправляет запрос на повторную передачу этого блока.

Установка режимов сжатия данных и коррекции ошибок

1. Ввести команду [Настройка-Панель управления-Модемы].
2. На появившейся диалоговой панели *Свойства: Модемы* щелкнуть по кнопке *Свойства*.
3. На появившейся диалоговой панели *Свойства: Courier...* активизировать вкладку *Подключение* и щелкнуть по кнопке *Дополнительно*.
4. На появившейся диалоговой панели *Дополнительные параметры связи* установить флажки *Обработка ошибок* и *Сжатие данных*.



Вопросы для размышления

1. В чем состоят процессы модуляции и демодуляции сигнала? Какие типы модуляции вам известны из курса физики?
2. Зачем в процессе передачи данных осуществляются их сжатие и коррекция ошибок?

Практические задания

12.7. Определите максимальную скорость передачи данных, которую может обеспечить ваш модем.

12.8. Проверьте правильность установки режимов сжатия данных и коррекции ошибок.

12.6.2. Управление модемом с использованием AT-команд

Модем может находиться в одном из двух режимов работы: передачи данных или AT-команд. AT-команды используются для настройки и управления работой модема. AT-команды представляют собой последовательности символов, начинающиеся с латинских букв AT. Перечень AT-команд у различных модемов практически одинаков, а с их назначением и синтаксисом можно ознакомиться в руководстве пользователя модема.

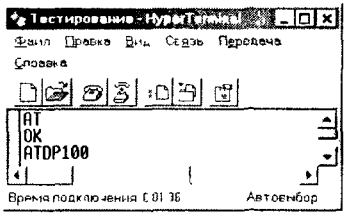
Для управления модемом с помощью AT-команд используются терминальные программы (в Windows — программа Hyper Terminal). Команды, введенные с клавиатуры, и отклики на них модема отображаются в окне терминальной программы. Если будет введена «пустая» команда AT и нажата клавиша {*Enter*}, правильно подключенный модем должен дать отклик Ok.

С помощью терминальной программы можно заставить модем «снять трубку» и позвонить по заданному телефонному номеру (для этого используется команда ATD). После буквы «D» в команде должен стоять символ, определяющий метод набора номера: «T» — тоновый набор или «P» — импульсный (в российских телефонных сетях используется импульсный набор). Например, звонок в Москве для получе-

нии информации о точном времени может быть выполнен с помощью команды ATDP100.

Телефонный звонок с помощью AT-команд

1. Ввести команду [Стандартные-Связь-Hyper Terminal].
2. В открывшемся окне папки Hyper Terminal запустить на выполнение программу Hyper Terminal.
3. В открывшемся окне набрать команду AT; если модем подключен и исправен, то на экране высветится OK.
4. Если вы находитесь в Москве, то можно набрать команду ATDP100 (*звонок по номеру* **100**).
   Модем наберет указанный номер, и вы услышите автоответчик службы «Точное время».



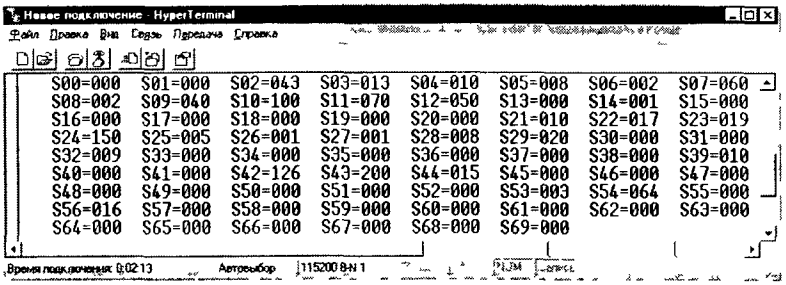
С помощью AT-команд можно менять состояние регистров энергонезависимой памяти модема (S00, S01, ..., S69), которые определяют режимы его работы. Текущее состояние регистров можно узнать с помощью команды ATI4.

При необходимости можно изменить числовое значение, хранящееся в регистре. Например, если периодически происходят разрывы связи по причине временного пропадания сигнала в линии, рекомендуется увеличить значения регистров S09 и S10. Этими регистрами задаются временны́е параметры захвата модемом несущей в линии. В каждом из этих регистров может быть записано число от 0 до 255, которое показывает время, измеряемое в десятых долях секунды. Если несущая на линии пропадает на время, большее, чем значение, хранящееся в регистре S10 (например, из-за помех или плохого контакта), то связь разрывается. Если несущая появилась на линии, то прежде, чем модем ее захватит, должно пройти время, определяемое регистром S09. По умолчанию значения этих регистров составляют: S09=6, S10=7. Рекомендуется увеличить значения этих регистров, например, до S09=40, S10=100.

Установка значений регистров модема

1. Ввести AT-команду для установки новых значений: ATS9=40S10=100.

2. Ввести АТ-команду для просмотра содержимого регистров: `ATI4`. В окне терминала появятся значения всех регистров.



Практические задания

12.9. Позвоните с помощью модема по заданному телефонному номеру.

12.10. Ознакомьтесь с текущим состоянием регистров вашего модема.

12.11. В целях уменьшения количества разрывов связи измените текущее состояние регистров `S09` и `S10` вашего модема.

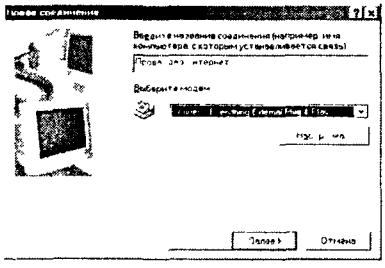
12.7. Настройка соединения и подключение к Интернету

После решения аппаратных (компьютер подключен с помощью модема к телефонной сети) и организационно-финансовых проблем (провайдер предоставил возможность доступа к Интернету, то есть сообщил вам идентификатор, пароль и другую необходимую информацию) необходимо произвести настройку программного обеспечения.

Пусть доступ к Интернету предоставляет некоторая фирма «Провайдер Интернет». Рассмотрим процесс настройки соединения с Интернетом с использованием специальной утилиты Удаленный доступ к сети, которая входит в состав операционной системы Windows.

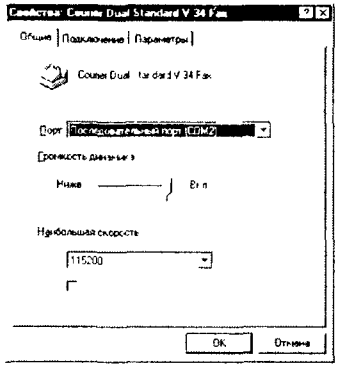
Создание соединения с Интернетом

1. Ввести команду [Стандартные-Связь-Удаленный доступ к сети-Новое соединение].
   На появившейся диалоговой панели *Новое соединение* необходимо ввести название соединения, например *Провайдер Интернет*, и выбрать из списка подключенный модем.



Далее необходимо настроить параметры подключения и режимы работы модема, для этого необходимо щелкнуть по кнопке *Настройка*.

2. Появится диалоговая панель *Свойства:* с тремя вкладками: *Общие, Подключение, Параметры.*
   На вкладке *Общие* необходимо в списке *Порт* выбрать порт, к которому подключен модем (в данном случае *Параллельный порт COM2*), и в списке *Наибольшая скорость* выбрать скорость передачи информации от порта компьютера к модему.

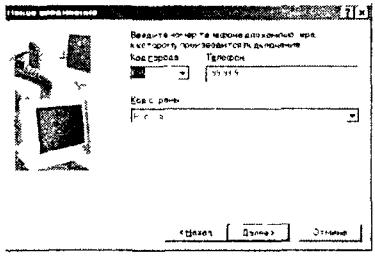


При использовании модемов стандарта V.34 и выше рекомендуется выбрать максимальную скорость передачи 115 200 бит/с.

На вкладках *Подключение* и *Параметры* рекомендуется оставить те значения, которые предлагаются по умолчанию.

После настройки параметров подключения и работы модема необходимо возвратиться к диалоговой панели *Новое соединение* и нажать кнопку *Далее*.

3. На появившейся следующей диалоговой панели необходимо ввести в поле *Телефон:* номер телефона провайдера, а также выбрать нужные пункты списков *Код города:* и *Код страны:*
   Щелкнуть по кнопке *Далее*.

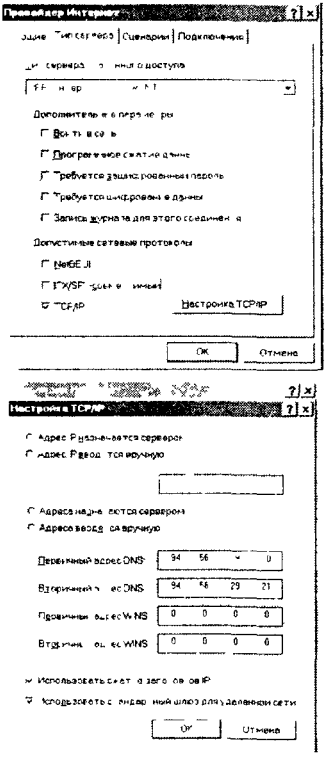


Появится следующая диалоговая панель, на которой для подтверждения создания соединения *Провайдер Интернет* необходимо нажать кнопку *Готово*.

Для дальнейшей настройки соединения *Провайдер Интернет* необходимо вызвать контекстное меню этого соединения и ввести команду [Свойства]. Появится диалоговая панель *Провайдер Интернет* с четырьмя вкладками: *Общие, Тип сервера, Сценарии* и *Подключения*.

Вкладка *Общие* содержит установки, которые были введены выше при создании соединения *Провайдер Интернет*, поэтому в них ничего изменять не надо. Содержимое вкладок *Сценарии* и *Подключения* также можно не изменять.

4. На вкладке *Тип сервера* необходимо из списка *Тип сервера удаленного доступа* выбрать *PPP: Интернет, Windows NT Server, Windows 98*.
   В группе флажков *Допустимые сетевые протоколы* установить флажок *TCP/IP*, а в группе флажков *Дополнительные параметры* флажки можно не устанавливать.
5. Следующим шагом является настройка протокола TCP/IP, для чего необходимо щелкнуть по кнопке *Настройка TCP/IP*, появится диалоговая панель *Настройка TCP/IP*.
   Обычно провайдер присваивает вашему компьютеру переменный IP-адрес, который может меняться от подключения к подключению, поэтому верхний переключатель необходимо установить в положение *Адрес IP назначается сервером*.



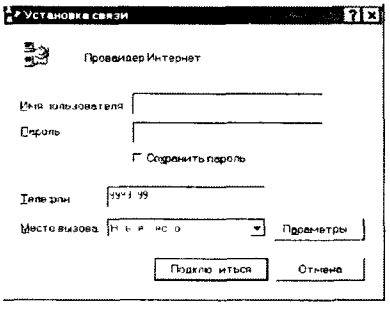
Функционирование Доменной системы имен в Интернете обеспечивается специальными серверами *DNS*, которые в целях надежности дублируются. Поэтому первичный и вторичный адреса *DNS* сообщаются пользователю провайдером и вводятся вручную (например, 194.56.29.20 и 194.56.29.21).

Теперь удаленное соединение *Провайдер Интернет* для подключения к Интернету полностью настроено.

Подключение к Интернету. Для подключения к Интернету необходимо запустить настроенное соединение.

Подключение к Интернету

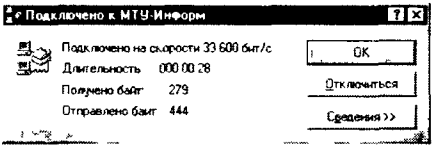
1. Ввести команду [Стандартные-Связь-Удаленный доступ к сети-Провайдер Интернет].
   Появится диалоговая панель *Установка связи*. В текстовое поле *Имя пользователя:* необходимо ввести идентификатор, а в поле *Пароль:* — пароль. Затем щелкнуть по кнопке *Подключиться*.



Модем произведет набор указанного телефонного номера, и если номер провайдера свободен, его модем «снимет трубку». В целях определения максимально высокоскоростного соединения модемы в течение нескольких секунд будут обмениваться сигналами (будет слышен звуковой шум различных частот).

Если модемам удастся установить связь и идентификатор и пароль введены правильно, то через несколько секунд произойдет подключение к Интернету.

2. Появится информационное окно, которое содержит информацию о подключении (скорость, длительность и др.).



Практические задания

12.12. Создать и настроить соединение для подключения к вашему провайдеру Интернета с помощью программы Удаленный доступ к сети.

12.13. Произвести подключение к Интернету, используя созданное соединение удаленного доступа.

12.8. Электронная почта и телеконференции

12.8.1. Электронная почта

Электронная почта (e-mail) — наиболее распространенный сервис Интернета, так как она является исторически первой информационной услугой компьютерных сетей и не требует обязательного наличия высокоскоростных и качественных линий связи.

Широкую популярность электронная почта завоевала потому, что имеет несколько серьезных преимуществ перед обычной почтой. Наиболее важное из них — это скорость пересылки сообщений. Если письмо по обычной почте может идти до адресата дни и недели, то письмо, посланное по электронной почте, сокращает время передачи до нескольких десятков секунд или, в худшем случае, до нескольких часов.

Другое преимущество состоит в том, что электронное письмо может содержать не только текстовое сообщение, но и вложенные файлы (программы, графику, звук и пр.). Однако не рекомендуется пересылать по почте слишком большие файлы, так как это замедляет работу сети. Для того чтобы этого не происходило, на некоторых почтовых серверах вводятся ограничения на размер пересылаемых сообщений (обычно почтовый сервер не пропускает сообщения более 2 000 000 байтов).

Кроме того, электронная почта позволяет:

- посылать сообщение сразу нескольким абонентам;
- пересылать письма на другие адреса;
- включить автоответчик, на все приходящие письма будет автоматически отсылаться ответ;
- создать правила для выполнения определенных действий с однотипными сообщениями (например, удалять рекламные сообщения, приходящие от определенных адресов) и так далее.

Адрес электронной почты. Для того чтобы электронное письмо дошло до адресата, оно, кроме самого *сообщения*, обязательно должно содержать *адрес электронной почты* получателя письма.

Первая часть почтового адреса (user_name — имя пользователя) имеет произвольный характер и задается самим пользователем при регистрации почтового ящика. Вторая часть (server_name — имя сервера) является доменным именем почтового сервера, на котором пользователь зарегистрировал свой почтовый ящик.

Адрес электронной почты записывается по определенной форме и состоит из двух частей, разделенных символом @:
user_name@server_name

Адрес электронной почты записывается только латинскими буквами и не должен содержать пробелов. Например, почтовый сервер компании МТУ-Интел имеет имя mtu-net.ru. Соответственно имена почтовых ящиков пользователей будут иметь вид:

user_name@mtu-net.ru

Функционирование электронной почты. Любой пользователь Интернета может зарегистрировать почтовый ящик на одном из серверов Интернета (обычно на почтовом сервере провайдера), в котором будут накапливаться передаваемые и получаемые электронные письма. В настоящее время достаточно большое количество серверов Интернета предоставляют возможность бесплатно зарегистрировать почтовый ящик.

Для работы с электронной почтой необходимы специальные почтовые программы, причем для любой компьютерной платформы существует большое количество почтовых программ. Почтовые программы входят в состав широко распространенных коммуникационных пакетов: Outlook Express входит в Microsoft Internet Explorer, Netscape Messenger — в Netscape Communicator.

С помощью почтовой программы создается почтовое сообщение на локальном компьютере. На этом этапе кроме написания текста сообщения необходимо указать адрес получателя сообщения, тему сообщения и вложить в сообщение при необходимости файлы.

Процесс передачи сообщения начинается с подключения к Интернету и доставки сообщения в свой почтовый ящик на удаленном почтовом сервере. Почтовый сервер сразу же отправит это сообщение через систему почтовых серверов Интернета на почтовый сервер получателя в его почтовый ящик.

Адресат для получения письма должен соединиться с Интернетом и доставить почту из своего почтового ящика на удаленном почтовом сервере на свой локальный компьютер (рис. 12.9).

Почтовые программы обычно предоставляют пользователю также многочисленные дополнительные сервисы по работе с почтой (выбор адресов из адресной книги, автоматическую рассылку сообщений по указанным адресам и др.).

Рис. 12.9. Функционирование электронной почты

Почтовая программа Outlook Express. После запуска программы Outlook Express появится окно программы, которое состоит из четырех частей (рис. 12.10). В левой верхней части окна находится перечень папок, в которых хранится корреспонденция:

- *Входящие* — содержит получаемые адресатом письма;
- *Исходящие* — содержит отправляемые адресатом письма с момента их создания и до момента их доставки с локального компьютера пользователя на почтовый сервер провайдера;
- *Отправленные* — содержит все письма, доставленные на почтовый сервер;
- *Удаленные* — содержит удаленные письма;
- *Черновики* — содержит заготовки писем.

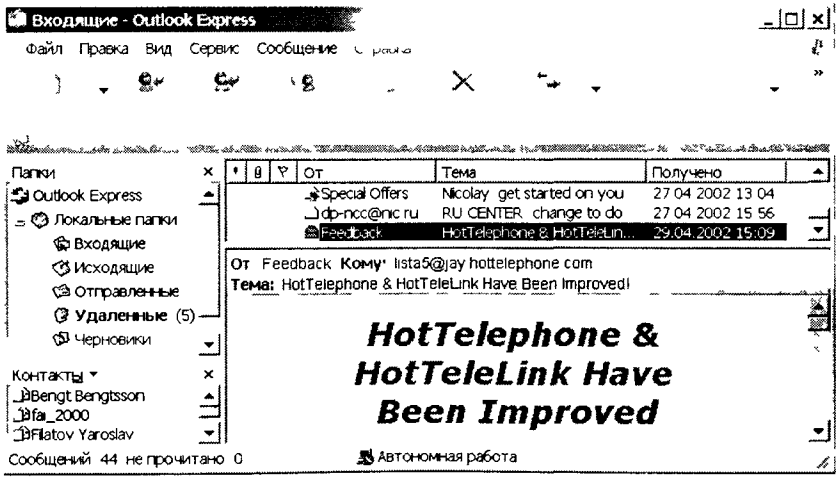


Рис. 12.10. Окно почтовой программы Outlook Express

Пользователь может создавать собственные папки для хранения тематически сгруппированных сообщений. В папках могут храниться не только сообщения, но и файлы, созданные с помощью других приложений.

В нижней левой части окна размещается *список контактов*, который предоставляет доступ к информации, хранящейся в *Адресной книге* (адреса электронной почты, телефоны и так далее).

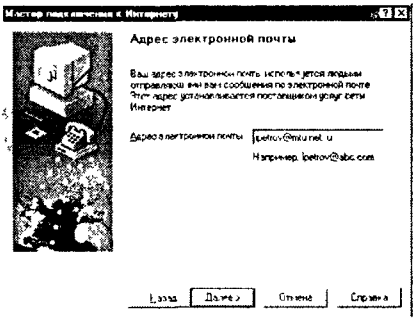
Правое окно разделено на две части. В верхней части высвечивается список сообщений, хранящихся в выделенной папке.

В нижней части правого окна отображается содержание выделенного сообщения.

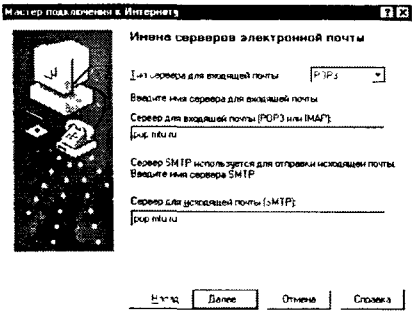
В первую очередь необходимо в соответствии с полученными в процессе регистрации почтового ящика данными (имя почтового ящика, пароль и др.) настроить почтовую программу. Создадим в почтовой программе Outlook Express учетную запись «Почта Интернета», при помощи которой можно будет отправлять и принимать электронную почту с конкретного почтового ящика.

Создание учетной записи

1. В окне программы Outlook Express ввести команду [Сервис-Учетные записи]. Откроется диалоговая панель *Учетные записи Интернета*.
   Выбрать вкладку *Почта*. Щелкнуть по кнопке *Добавить* и выбрать пункт *Почта...*
2. Откроется диалоговая панель *Мастер подключения к Интернету*. В поле *Ваше имя:* указать имя, которое будет видеть человек, получивший от вас письмо. Щелкнуть по кнопке *Далее*.
3. В появившемся окне в поле *Адрес электронной почты*: указать тот адрес, который вы задали при регистрации подключения.
   Адрес следует указать целиком и именно в том виде, в котором вы его создали.
   Щелкнуть по кнопке *Далее*.



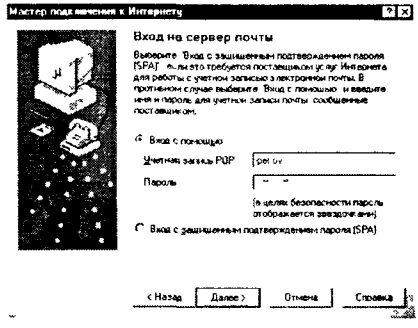
4. На появившейся диалоговой панели в поле *Тип сервера для входящей почты:* выберите POP3. Этот протокол наиболее часто используется для электронной почты.



В полях *Сервер для входящей почты (POP3 или IMAP):* и *Сервер для исходящей почты (SMTP):* необходимо указать имена серверов входящей и исходящей почты, которые сообщает провайдер при регистрации подключения.
Щелкнуть по кнопке *Далее*.

Теперь необходимо указать имя почтового ящика и пароль для входа на почтовый сервер.

5. В поле *Учетная запись POP:* ввести имя, которое вы указали при создании своего почтового адреса перед значком @. В поле *Пароль:* необходимо указать тот пароль, который был получен при регистрации подключения у провайдера.



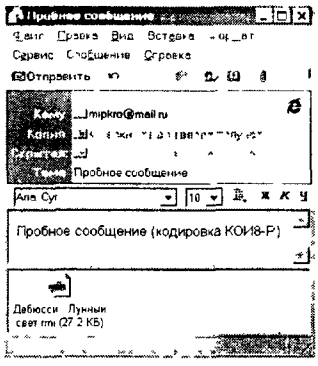
6. Заданные выше параметры электронной почты объединяются вместе под одним именем — именем учетной записи.
В поле *Имя учетной записи почты сети Интернет:* необходимо ввести имя для созданной учетной записи, например «Почта Интернета».

7. На следующих диалоговых панелях необходимо указать способ соединения с Интернетом, выбрать тип модема и используемое соединение с Интернетом.

Создание, отправка и получение сообщений. Создадим пробное сообщение в определенной кодировке с вложенным файлом.

Создание, отправка и получение сообщения

1. Ввести команду [Сообщение-Создать].
   В окне *Создать сообщение* в поле *Кому:* необходимо указать электронный адрес адресата, например: mipkro@mail.ru
   В поле *Копии:* можно указать адреса получателей копии сообщения.
   В поле *Тема:* указывается тема сообщения, например «Пробное сообщение».



2. В области, отведенной для сообщения, вводится текст сообщения, например «Пробное сообщение (кодировка КОИ8-Р)».

Достаточно важен выбор правильной кодировки русских букв сообщения. При пользовании электронной почтой чаще всего используются кодировки Windows и КОИ8-Р.

3. Выбор кодировки осуществить с помощью команды [Формат-Вид кодировки-Кириллица (КОИ8-Р)].

В сообщение можно вставлять файлы (текстовые, графические, звуковые и так далее).

4. Для вставки файла в сообщение необходимо ввести команду [Вставка-Вложение файла...]. В появившемся окне *Вставка* необходимо выбрать требуемый файл, и он будет вложен в сообщение.
5. Вставим, например, в сообщение звуковой файл Дебюсси-Лунный свет.rmi из папки Media, которая находится в папке Windows. Название вложенного файла появится в нижней части окна сообщения.

Если создание сообщения производилось в автономном режиме без подключения к Интернету, сообщение необходимо сохранить в папке *Исходящие*.

6. После завершения работы над сообщением щелкнуть по кнопке *Отправить*, сообщение будет помещено в папку *Исходящие*.

Для того чтобы отправить сообщение адресату, необходимо подключиться к Интернету.

7. Щелкнуть по кнопке *Доставить почту*. Произойдет соединение с почтовым сервером, и все сообщения, находящиеся в папке *Исходящие* на локальном компьютере, бу-

дут доставлены на почтовый сервер. Одновременно отправленные сообщения будут перемещены на локальном компьютере в папку *Отправленные*.

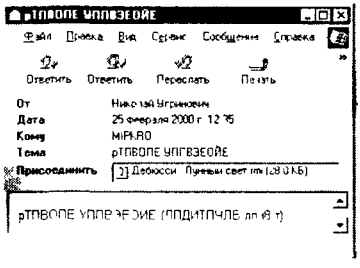
Почтовый сервер провайдера передаст сообщения в Интернет и через некоторое время они будут доставлены на почтовые сервера получателей. В данном случае пробное сообщение попадет в почтовый ящик mipkro@mail.ru

Для получения сообщения абонент должен соединиться с Интернетом и произвести операцию доставки почты с почтового сервера провайдера на свой локальный компьютер.

8. Щелкнуть по кнопке *Доставить почту*. В процессе доставки почты сообщения, хранящиеся в почтовом ящике на почтовом сервере, будут переданы на локальный компьютер получателя и размещены в папке *Входящие*.

В случае установки кодировки, отличной от использованной при создании сообщения, сообщение будет представлять собой полную абракадабру.

9. В этом случае необходимо подобрать кодировку с помощью команды [Формат-Вид кодировки...].



Вопросы для размышления

1. Могут ли почтовые ящики, размещенные на разных почтовых серверах, иметь одинаковые идентификаторы?
2. В чем состоит отличие между операциями отправки и доставки почтового сообщения?

Практические задания

12.14. Настроить почтовую программу на работу с имеющимся у вас ящиком электронной почты.

12.15. Создать и отправить сообщение по определенному почтовому адресу.

12.8.2. Электронная почта с Web-интерфейсом

В последнее время для работы с электронной почтой стала использоваться Web-технология. Появились Web-сайты, которые предлагают всем желающим зарегистрировать бесплатный почтовый ящик (например, по адресу: http://mail.ru).

Преимуществом такой почты является то, что для работы с ней не требуются специальные почтовые программы. Работа с почтой может производиться с помощью любого браузера после загрузки соответствующей Web-страницы.

На рис. 12.11 представлен интерфейс начальной страницы Web-почты. Зарегистрированные пользователи должны ввести свой идентификатор (*логин*) и пароль, после чего они могут войти в почтовую систему.

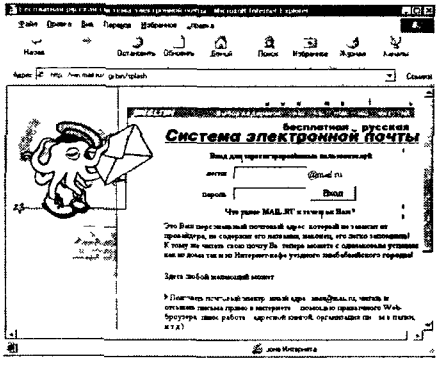


Рис. 12.11. Вход в Web-почту

Для новых пользователей предлагается процедура регистрации.

Почтовая система с Web-интерфейсом по своим возможностям аналогична традиционной электронной почте.

Сообщения группируются по папкам, можно отправлять сообщения с вложенными файлами, одновременно нескольким абонентам и так далее.

Существенной особенностью Web-почты является то, что все сообщения постоянно хранятся на удаленном сервере, а не на локальном компьютере пользователя.

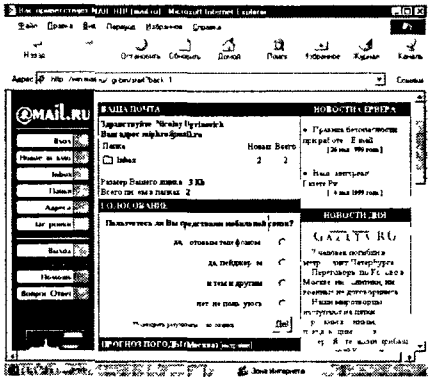


Рис. 12.12. Интерфейс Web-почты

Практические задания

12.16. Зарегистрируйте почтовый ящик на одном из серверов бесплатной почты: mail.ru, www.yandex.ru, www.rambler.ru. Создайте и отправьте письмо.

12.8.3. Телеконференции

В Интернете существуют десятки тысяч конференций или групп новостей (news), каждая из которых посвящена обсуждению какой-либо проблемы. Каждой конференции выделяется свой почтовый ящик на серверах Интернета, которые поддерживают работу этой телеконференции.

Пользователи могут посылать свои сообщения на любой из этих серверов. Сервера периодически синхронизируются, то есть обмениваются содержимым почтовых ящиков телеконференций, поэтому материалы конференций в полном объеме доступны пользователю на любом таком сервере.

Принцип работы в телеконференциях мало чем отличается от принципа работы с электронной почтой. Пользователь может посылать свои сообщения в любую телеконференцию и читать сообщения, посланные другими участниками.

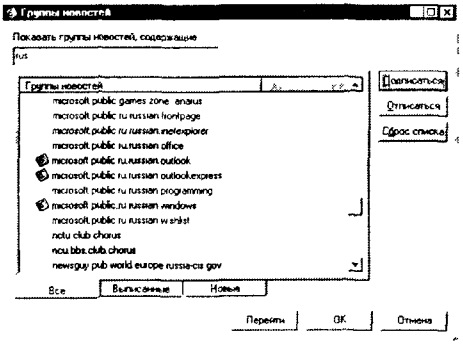
Для работы в телеконференциях используют обычно те же самые почтовые программы, что и при работе с электронной почтой, например Outlook Express. Настройка Outlook Express для работы с телеконференциями происходит аналогично настройке для работы с электронной почтой, то есть создается учетная запись для работы с новостями, например «Конференции». Outlook Express создает одноименную папку *Конференции*, которая первоначально пуста.

Для того чтобы иметь доступ к почтовому ящику какой-либо конференции, на нее необходимо «подписаться».

Работа с телеконференциями

1. В окне почтовой программы Outlook Express выделить папку новостей (в данном случае *Конференции* и щелкнуть по кнопке *Группы новостей*.

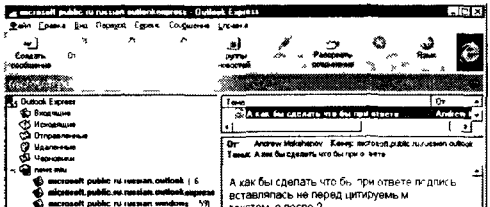
2. Появится диалоговая панель *Группы новостей*, в которой высветятся названия десятков тысяч конференций.
   Для поиска интересующей тематики можно в поле *Показать группы новостей, содержащие:* задать символьный шаблон.



Пусть, например, вас интересуют русскоязычные конференции, с большой долей вероятности их можно найти, если задать шаблон «rus». В окне откроется перечень конференций, в названии которых содержится этот шаблон. Следует выбрать некоторые из них (отметить с помощью мыши) и щелкнуть по кнопке *Подписаться*. Произойдет загрузка содержимого этих конференций (писем участников) на локальный компьютер пользователя.

Вы выбрали три интересующие вас конференции, которые теперь существуют в форме папок, вложенных в папку конференций *Конференции*. Выделите любую из них, например microsoft.public.ru.russian.outlookexpress (конференцию, посвященную проблемам работы в Outlook Express).

3. В правом верхнем окне вы увидите ее содержание, то есть перечень присланных в нее сообщений.
   Выделить интересующее вас сообщение, и в правом нижнем окне вы увидите содержание этого сообщения.



Ознакомившись с содержанием телеконференции, при желании можно послать в нее собственное сообщение, используя для создания бланка сообщения команду [Сообщение-Ответить в группу новостей].

Практические задания

12.17. Найти телеконференцию, посвященную работе с почтовой программой Outlook Express, и ознакомиться с ее содержанием.

12.9. Всемирная паутина

12.9.1. Технология World Wide Web

Всемирная паутина — это вольный перевод английского словосочетания *World Wide Web*, которое часто обозначается как WWW или Web. Бурное развитие сети Интернет, которое происходило на протяжении 90-х годов, в первую очередь обусловлено появлением новой технологии WWW.

9.4. Гипертекст

Технология WWW. Технология WWW позволяет создавать ссылки (их также называют гиперссылками), которые реализуют переходы не только внутри исходного документа, но и на любой другой документ, находящийся на данном компьютере и, что самое главное, на любой документ любого компьютера, подключенного в данный момент к Интернету (рис. 12.13).

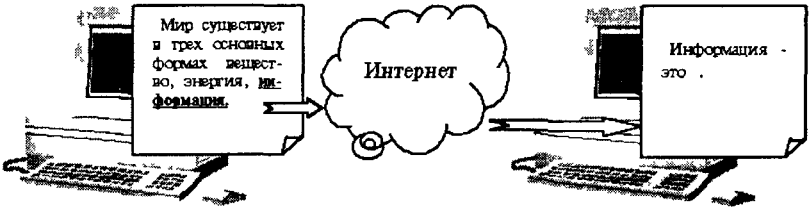


Рис. 12.13. Технология WWW

В качестве *указателей ссылок*, то есть объектов, активизация которых вызывает переход на другой документ, могут использоваться не только фрагменты текста, но и графические изображения.

Серверы Интернета, реализующие WWW-технологию, называются Web-серверами, а документы, реализованные по технологии WWW, называются Web-страницами.

Всемирная паутина — это десятки миллионов Web-серверов Интернета, содержащих Web-страницы, в которых используется технология гипертекста.

Создание Web-страниц осуществляется с помощью языка разметки гипертекста (Hyper Text Markup Language — HTML). Основа используемой в HTML технологии состоит в том, что в обычный текстовый документ вставляются управляющие символы (тэги), и в результате мы получаем текстовый документ, который при просмотре в браузере мы видим в форме Web-страницы. С помощью тэгов можно изменять размер, начертание и цвет символов, фон, определять положение текста на странице, вставлять гиперссылки и так далее.

Web-страница может быть мультимедийной, то есть может содержать ссылки на различные мультимедийные объекты: графические изображения, анимацию, звук и видео.

Интерактивные Web-страницы содержат формы, которые может заполнять посетитель. Динамический HTML использует объектную модель документа, то есть рассматривает документ как совокупность объектов, свойства которых можно изменять. Это позволяет создавать динамические Web-страницы, то есть страницы, которые могут меняться уже после загрузки в браузер. Например, текст может менять цвет, когда к нему подводится курсор, заголовок — перемещаться и так далее. Кроме того, пользователь может активизировать ссылки на выполняемые сценарии на языках JavaScript и VBScript, а также элементы управления ActiveX.

Тематически связанные Web-страницы обычно бывают представлены в форме Web-сайта, то есть целостной системы документов, связанных между собой в единое целое с помощью гиперссылок.

Универсальный указатель ресурсов. Найти Web-страницу или файл в Интернете можно с помощью *универсального указателя ресурсов* (адреса Web-страницы).

Универсальный указатель ресурсов (URL — Universal Resource Locator) включает в себя протокол доступа к документу, доменное имя или IP-адрес сервера, на котором находится документ, а также путь к файлу и собственно имя файла:
protocol://domain_name/path/file_name

Протокол доступа к документу определяет способ передачи информации. Для доступа к Web-страницам используется протокол передачи гипертекста HTTP (Hyper Text Transfer Protocol). При записи протокола после его имени следует двоеточие и два прямых слэша: http:// .

Запишем URL-адрес титульной страницы Web-сайта «Информатика и информационные технологии». Страница расположена на сервере schools.keldysh.ru, в каталоге info2000 в файле index.htm. Следовательно, универсальный указатель ресурсов принимает вид:

http://schools.keldysh.ru/info2000/index.htm .

Он состоит из трех частей:

http:// — протокол доступа;

schools.keldysh.ru — доменное имя сервера;

/info2000/index.htm — путь к файлу и имя файла Web-страницы.

Вопросы для размышления

1. В чем состоит отличие технологии WWW от технологии гипертекста?

Задания

12.18. Записать URL начальной страницы сайта «Информатика и информационные технологии» с использованием IP-адреса сервера 194.226.57.46. Начальная страница default.asp находится в каталоге iit.

12.9.2. Браузеры — средство доступа к информационным ресурсам Всемирной паутины

Просмотр Web-страниц осуществляется с помощью специальных программ просмотра — браузеров. В настоящее время наиболее распространенными браузерами являются Internet Explorer (его русскоязычная версия часто называется Обозреватель) и Netscape Communicator (Коммуникатор).

Если компьютер подключен к Интернету, то можно запустить один из браузеров и отправиться в виртуальное путешествие по Всемирной паутине.

Настройка браузера. После запуска откроется окно Обозревателя, в которое будет загружена начальная Web-страница.

По умолчанию в русскоязычной версии Internet Explorer такой страницей является страница Web-сервера фирмы Microsoft.

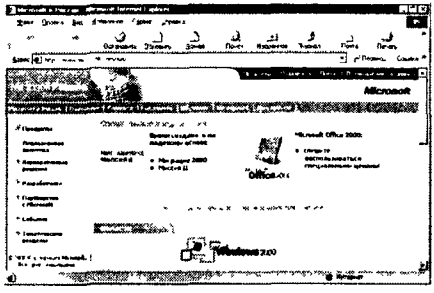


Рис. 12.14. Окно браузера Internet Explorer

Окно браузера содержит меню, панель инструментов, окно *Адрес:*, а также рабочую область, в которой и просматриваются Web-страницы.

Окно *Адрес* позволяет ввести с клавиатуры или выбрать из списка URL нужной Web-страницы.

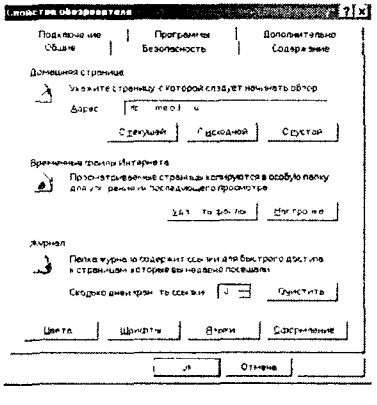
Панель инструментов позволяет переходить с одной Web-страницы на другую (кнопки *Вперед, Назад, Домой*), управлять процессом загрузки (кнопки *Остановить, Обновить*) и др.

Внешний вид и местоположение панелей инструментов можно изменять, перетаскивая компоненты панели инструментов с помощью мыши или используя меню *Вид*.

Параметры просмотра Web-страниц в браузере можно изменять с помощью многочисленных настроек. Так, можно изменить адрес начальной страницы, загружаемой в Обозреватель.

Настройка браузера

1. В меню *Сервис* выбрать пункт *Свойства обозревателя*. Откроется диалоговая панель *Свойства обозревателя*.
   На вкладке *Общие* в группе *Домашняя страница* в поле *Адрес:* ввести URL нужной страницы, например начальной страницы Web-сайта «Информатика и информационные технологии»:
   http://iit.metodist.ru .



2. Чтобы убедиться, что введен правильный адрес, щелкнуть по кнопке *Домой*. Обозреватель должен загрузить начальную страницу Web-сайта «Информатика и информационные технологии».

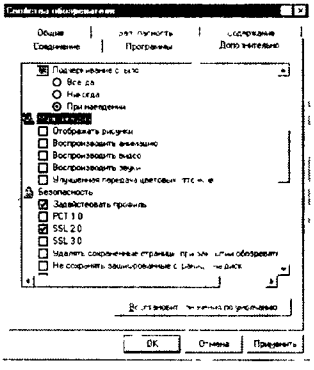
Большое значение имеет настройка Обозревателя на просмотр Web-страницы в правильной кодировке, то есть той кодировке, в которой Web-страница была создана. В большинстве случаев Обозреватель автоматически определяет кодировку и, соответственно, правильно отображает Web-страницу.

Однако в некоторых случаях пользователю необходимо настроить Обозреватель на требуемую кодировку вручную.

3. Например, для просмотра Web-страницы в кодировке *Windows* необходимо ввести команду [Вид-Вид кодировки-Кириллица (Windows)].

Можно ускорить процесс загрузки Web-страниц в случае соединения с Интернетом на низкой скорости передачи информации (16,4 Кбит/с и менее) или в случае перегруженности Web-страниц мультимедийными объектами, имеющими большой информационный объем. Для этого необходимо отключить загрузку мультимедиа объектов (рисунки, анимация, звук, видео).

4. Ввести команду [Сервис-Свойства обозревателя...], на появившейся диалоговой панели *Свойства обозревателя* выбрать вкладку *Дополнительно*.
С помощью прокрутки найти в окне раздел *Мультимедиа* и снять все флажки этого раздела.



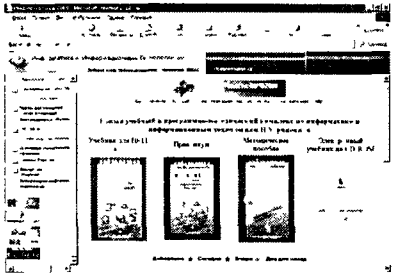
Путешествия по Всемирной паутине. Для того чтобы начать путешествие по Всемирной паутине, необходимо подключиться к Интернету и запустить какой-нибудь браузер, например Internet Explorer. После загрузки начальной (домашней) страницы можно поступать различными способами:

- воспользоваться ссылками загруженной Web-страницы браузера;
- в строку *Адрес* ввести адрес (URL) интересующей Web-страницы;
- воспользоваться «закладками» Web-страниц.

Использование ссылок. Путешествие по Всемирной паутине можно начать с путешествия по страницам сайта. Web-сайты обычно содержат достаточно большое количество страниц, поэтому для переходов с одной страницы на другую внутри сайта используется *панель навигации*. Панель навигации содержит названия основных разделов сайта.

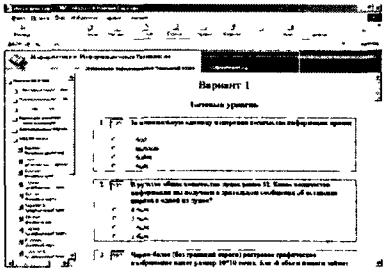
Путешествие по Всемирной паутине

1. Загрузить начальную страницу Web-сайта «Информатика и информационные технологии».
   Загрузится страница, содержащая *панель навигации*, которая расположена в левой части страницы.



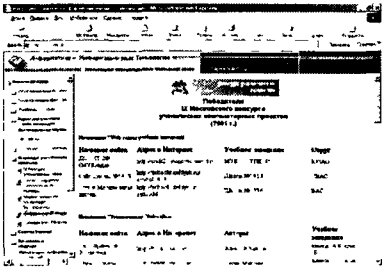
Теперь, активизируя те или иные пункты панели навигации, можно переходить на соответствующие страницы сайта.

2. На панели навигации активизировать, например, пункт *ON-LINE тесты-Вариант 1 (Базовый уровень)*. Загрузится страница, содержащая тесты по базовому курсу информатики.
   Здесь каждый учащийся может анонимно проверить свои знания.



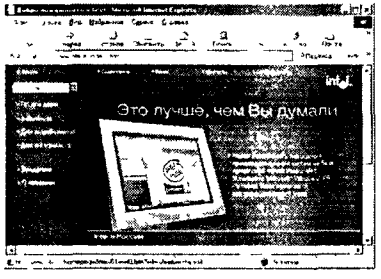
Для перехода на другие Web-сайты можно воспользоваться гиперссылками. Так, на одной из страниц Web-сайта «Информатика и информационные технологии» имеются ссылки на Web-сайты — победители конкурса ученических компьютерных проектов.

3. Для просмотра Web-сайтов, представленных на конкурс, последовательно активизируйте ссылки на работы учащихся (ссылки расположены во втором столбце таблицы).



Ввод URL в окне *Адрес*. Пусть нас интересует информация о процессоре Pentium 4 фирмы Intel. Достаточно легко «вычислить» имя русскоязычного Web-сервера фирмы Intel, на котором можно найти такую информацию. Компания обладает представительствами во многих странах мира, следовательно, должен быть Web-сервер, который относится к домену верхнего уровня ru. Фирма зарегистрировала имя домена второго уровня, скорее всего, совпадающее с ее названием, получаем intel.ru. Наконец, большинство Web-серверов имеют имя www, получаем полное имя сервера www.intel.ru.

4. Набрать в поле *Адрес* URL данного сервера:
   http://www.intel.ru .
   Произойдет загрузка интересующей нас Web-страницы.



Создание и использование «закладок». В процессе чтения книги (учебника, справочника, энциклопедии) достаточно часто требуется вернуться к прочитанному материалу. Для более быстрого поиска нужной страницы часто в книгу вставляют закладку. В процессе путешествий по Всемирной паутине также целесообразно оставлять «закладки» на интересных Web-страницах, так как найти нужную страницу среди десятков миллионов Web-страниц гораздо труднее, чем найти страницу в книге.

5. Для создания закладки ввести команду [Избранное-Добавить в избранное...], на появившейся диалоговой панели *Добавление в избранное* выбрать название закладки (по умолчанию оно совпадает с названием Web-страницы) и щелкнуть по кнопке *ОК*. Теперь для загрузки данной страницы достаточно в пункте меню *Избранное* выбрать название закладки.

Путешествия по виртуальной реальности. С помощью специального языка моделирования виртуальной реальности (Virtual Reality Modeling Language — *VRML*) можно создавать виртуальные трехмерные миры, в которых можно затем перемещаться в различных направлениях и рассматривать предметы с различных сторон. В Интернете существует достаточно много серверов, содержащих виртуальные

миры, и в частности виртуальные города мира, по которым можно совершать виртуальные экскурсии (например, сервер www.intoronto.com — рис. 12.15).

Для посещения виртуальных миров необходим специальный программный модуль, который подключается к браузеру и позволяет просматривать анимированную графику сцен виртуальной реальности. Наиболее распространенным является CosmoPlayer.

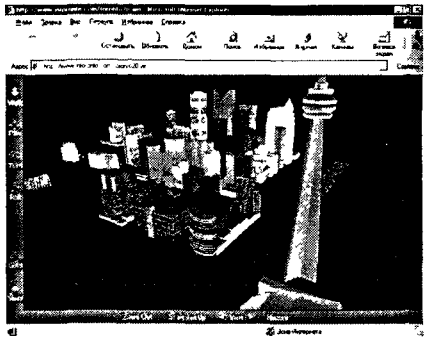


Рис. 12.15. Виртуальный Торонто

Практические задания

12.19. Исследуйте влияние установки различных уровней безопасности в настройках браузера на просмотр Web-страниц.

12.10. Файловые архивы

Серверы файловых архивов. Десятки тысяч серверов Интернета являются *серверами файловых архивов*, и на них хранятся сотни миллионов файлов различных типов (программы, драйверы устройств, графические и звуковые файлы и так далее). Наличие таких серверов файловых архивов очень удобно для пользователей, так как многие необходимые файлы можно «скачать» непосредственно из Интернета.

Файловые серверы поддерживают многие компании — разработчики программного обеспечения и производители аппаратных компонентов компьютера и периферийных устройств. Размещаемое на таких серверах программное обеспечение является свободно распространяемым (freeware) или условно бесплатным (shareware) и поэтому, «скачивая» тот или иной файл, пользователь не нарушает закон об авторских правах на программное обеспечение.

Для удобства пользователей многие серверы файловых архивов (freeware.ru, www.freesoft.ru, www.download.ru) имеют Web-интерфейс, что позволяет работать с ними с использованием браузеров.

Протокол передачи файлов (FTP). Доступ к файлам на серверах файловых архивов возможен как по протоколу HTTP, так и по специальному протоколу передачи файлов FTP (File Transfer Protocol). Протокол FTP позволяет не только загружать файлы (Download) с удаленных серверов файловых архивов на локальный компьютер, но и, наоборот, производить передачу файлов (Upload) с локального компьютера на удаленный Web-сервер, например, в процессе публикации Web-сайта.

Например, для загрузки с сервера файлового архива ftp.cuteftp.com компании GlobalScape файла cute4232.exe необходимо указать URL-адрес этого файла. При указании URL-адреса файла протокол FTP записывается следующим образом: ftp:// .

В результате универсальный указатель ресурсов принимает вид:

ftp://ftp.cuteftp.com/pub/cuteftp/cute4232.exe

и состоит из трех частей:

ftp:// — протокол доступа,

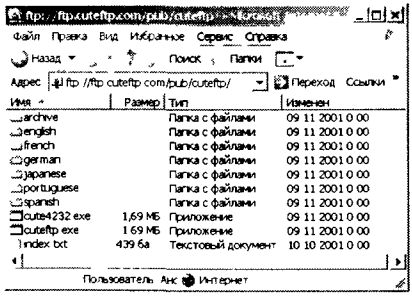
ftp.cuteftp.com — доменное имя сервера файлового архива,

/pub/cuteftp/cute4232.exe — путь к файлу и имя файла.

Загрузка файлов с помощью браузера. Браузеры являются интегрированными системами для работы с различными информационными ресурсами Интернета и поэтому включают в себя *менеджеры загрузки файлов* (Download Manager).

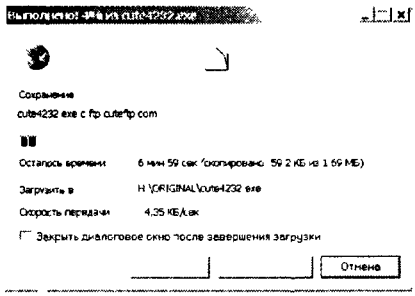
Загрузка файла с помощью браузера

1. Запустить браузер. В поле *Адрес* ввести URL, например: ftp://ftp.cuteftp.com/pub/cuteftp/ . Произойдет соединение с сервером и в окне браузера высветится содержимое указанного каталога.



После активизации ссылки на файл в открывшемся окне требуется указать папку на локальном компьютере, в которой файл должен быть сохранен.

2. Начнется загрузка файла, процесс которой отображается на информационной панели (скорость передачи, объем загруженной части файла и так далее).



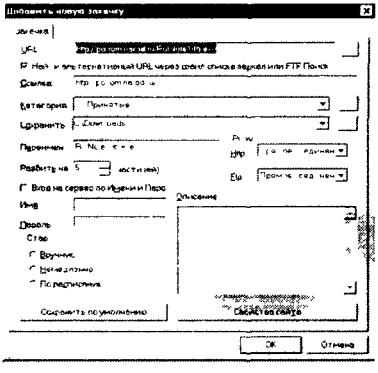
Загрузка файлов с помощью специализированных менеджеров загрузки. Однако удобнее для работы с файловыми архивами использовать специализированные менеджеры загрузки файлов (например, FlashGet, Go!Zilla, ReGet и др.). Такие менеджеры позволяют увеличить скорость загрузки за счет разбиения файлов на части и одновременной загрузки всех частей. Кроме того, они позволяют продолжить загрузку файла после разрыва соединения с сервером, содержат средство поиска файла на других серверах файловых архивов, позволяют архивировать файлы в процессе их загрузки и так далее.

Пользователю предоставляется в числовом и графическом виде подробная информация о процессе загрузки файла (текущая и средняя скорость загрузки, процент выполнения загрузки, ориентировочное время загрузки и др.).

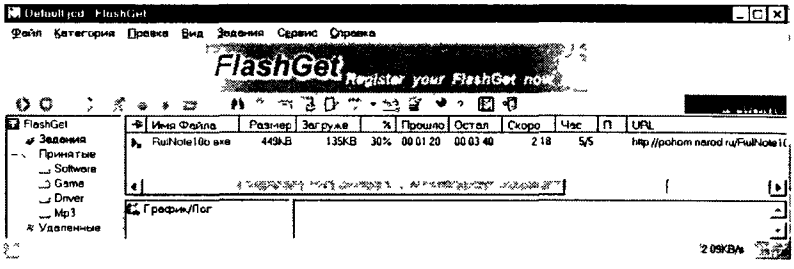
Менежеры загрузки файлов интегрируются в браузеры и при активизации ссылки на файл в окне браузера начинают процесс его загрузки.

Загрузка файла с помощью менеджера загрузки файлов FlashGet

1. Запустить браузер и активизировать ссылку на файл. FlashGet начнет процесс загрузки.
   В поле *Добавить новую закачку* можно ознакомиться с параметрами загрузки файла и изменить URL, папку на локальном компьютере, количество разбиений файла и др.
   Щелкнуть по кнопке *ОК*.



2. Начнется загрузка файла, процесс которой отображается в окне Менеджера загрузки.



FTP-клиенты. Обмен файлами (загрузка и передача) с серверами файловых архивов и Web-серверами производится с помощью специализированных программ — FTP-клиентов (AceFTP, CuteFTP и др.).

FTP-клиенты включают в себя *Менеджер сайтов*, позволяющий провести идентификацию пользователя (ввод имени пользователя и пароля). Это позволяет обеспечить доступ к Web-серверу с целью передачи файлов в процессе публикации Web-сайта. Кроме того, FTP-клиенты создают список серверов, с которыми планируется работа, представляют в удобном для пользователя виде каталоги локального и удаленного компьютера, поддерживают технологию Drag&Drop, обеспечивают продолжение загрузки файла после обрыва соединения и др.

6.4.3. Обмен файлами с FTP-серверами Практикум

Вопросы для размышления

1. В чем состоит преимущество протокола FTP перед протоколом HTTP при загрузке файлов?

Практические задания

12.20. Загрузить файл с одного из серверов файловых архивов сначала с помощью браузера, а затем — менеджера загрузки файлов.

12.11. Поиск информации в Интернете

Сеть Интернет растет очень быстрыми темпами, и найти нужную информацию среди миллиардов Web-страниц и файлов становится все сложнее. Для поиска информации используются специальные поисковые серверы, которые содержат более или менее полную и постоянно обновляемую информацию о Web-страницах, файлах и других документах, хранящихся на десятках миллионов серверов Интернета.

Различные поисковые сервера могут использовать различные механизмы поиска, хранения и предоставления пользователю информации. Поисковые серверы Интернета можно разделить на две группы:

- поисковые системы общего назначения;
- специализированные поисковые системы.

Современные поисковые системы часто являются информационными *порталами*, которые предоставляют пользователям не только возможности поиска документов в Интернете, но и доступ к другим информационным ресурсам (новостям, информации о погоде, о валютном курсе, интерактивным географическим картам и так далее).

12.11.1. Поисковые системы общего назначения

Поисковые системы общего назначения являются базами данных, содержащими тематически сгруппированную информацию об информационных ресурсах Всемирной паутины. Такие поисковые системы позволяют находить Web-сайты или Web-страницы *по ключевым словам в базе данных* или путем поиска *в иерархической системе каталогов*.

Интерфейс таких поисковых систем общего назначения содержит *список разделов каталога* и *поле поиска*. В поле поиска пользователь может ввести ключевые слова для поиска документа, а в каталоге выбрать определенный раздел, что сужает поле поиска и таким образом ускоряет его.

Заполнение баз данных осуществляется с помощью специальных программ-роботов, которые периодически «обходят» Web-серверы Интернета. Программы-роботы читают все встречающиеся документы, выделяют в них ключевые слова и заносят в базу данных, содержащую URL-адреса документов.

Так как информация в Интернете постоянно меняется (создаются новые Web-сайты и страницы, удаляются старые, меняются их URL-адреса и так далее), поисковые робо-

ты не всегда успевают отследить все эти изменения. Информация, хранящаяся в базе данных поисковой системы, может отличаться от реального состояния Интернета, и тогда пользователь в результате поиска может получить адрес уже не существующего или перемещенного документа.

В целях обеспечения большего соответствия между содержанием базы данных поисковой системы и реальным состоянием Интернета большинство поисковых систем разрешают автору нового или перемещенного Web-сайта самому внести информацию в базу данных, заполнив регистрационную анкету. В процессе заполнения анкеты разработчик сайта вносит URL-адрес сайта, его название, краткое описание содержания сайта, а также ключевые слова, по которым легче всего будет найти сайт.

Сайты в базе данных ранжируются по количеству их посещений в день, неделю или месяц. Посещаемость сайтов определяется с помощью специальных счетчиков, которые могут быть установлены на сайте. Счетчики фиксирует каждое посещение сайта и передают информацию о количестве посещений на сервер поисковой системы.

Поиск по ключевым словам. Поиск документа в базе данных поисковой системы осуществляется с помощью введения запросов в поле поиска. Простой запрос содержит одно или несколько ключевых слов, которые, по вашему мнению, являются главными для этого документа. Можно также использовать сложные запросы, использующие логические операции, шаблоны и так далее.

Через некоторое время после отправки запроса поисковая система вернет аннотированный список URL-адресов документов, в которых были найдены указанные вами ключевые слова. Для просмотра этого документа в браузере достаточно активизировать указывающую на документ ссылку.

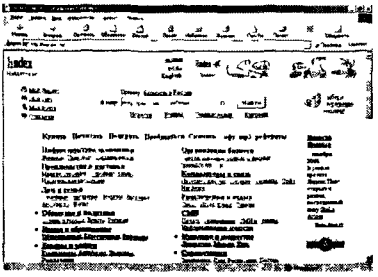
Если ключевые слова были выбраны неудачно, то список URL-адресов документов может быть слишком большим (может содержать десятки и даже сотни тысяч ссылок). Для того чтобы уменьшить список, можно в поле поиска ввести дополнительные ключевые слова или воспользоваться каталогом поисковой системы.

Наиболее мощными поисковыми системами общего назначения в русскоязычной части Интернета являются серверы Rambler (http://www.rambler.ru), Апорт (http://www.aport.ru), и Яндекс (http://www.yandex.ru), а по всему Интернету — сервер Yahoo (адрес http://www.yahoo.com).

Попробуем с помощью российского поискового сервера Яндекс найти сайт «Информатика и информационные технологии».

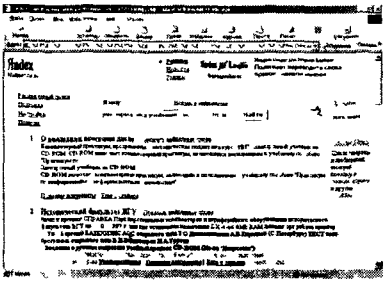
Поиск сайта по ключевым словам

1. В браузере открыть начальную страницу поискового сервера Яндекс.
   В поле поиска ввести ключевые слова, например «информатика учебники тесты CD-ROM».



Пробелы между словами соответствуют логической операции AND, то есть результатом поиска будет список сайтов, на которых присутствуют все вышеперечисленные ключевые слова.

2. В результате проведенного 3 ноября 2001 года поиска было найдено 118 Web-сайтов, содержащих все перечисленные выше ключевые слова.
   Для каждого документа кроме ссылки приводится еще адрес сайта (URL) и его краткая аннотация.



Искомый сайт «Информатика и информационные технологии» занимает первое место в этом списке, так как в наибольшей степени соответствует запросу. Щелчок по ссылке приведет к загрузке титульной страницы сайта.

В статистике поиска можно ознакомиться с количеством сайтов, содержащих каждое из ключевых слов: *информатика* — 553896, *учебники* — 1274027, *тесты* — 2485000, *CD* — 7024321, *ROM* — 2128526.

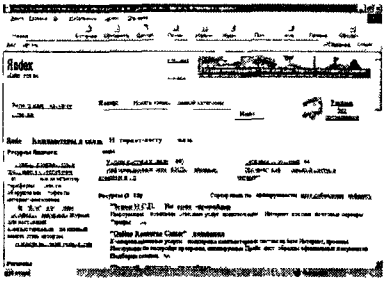
Поиск в иерархической системе каталогов. Web-сайты в базе данных поисковой системы группируются в тематические каталоги — аналоги тематического указателя в библиотеке. Тематические разделы верхнего уровня, например «Интернет», «Компьютеры», «Культура и искусство» и др., содержат вложенные каталоги. Например, каталог «Интернет» может содержать подкаталоги «Провайдеры», «Поиск», «Общение» и др.

Поиск информации в каталоге сводится к выбору определенного каталога, после чего пользователю будет представлен список ссылок на URL-адреса наиболее посещаемых и важных Web-сайтов и Web-страниц. Каждая ссылка обычно аннотирована, то есть содержит короткий комментарий к содержанию документа.

Воспользуемся *иерархической системой каталогов* для поиска информации о провайдерах Интернета.

Поиск в иерархической системе каталогов

1. Выбрать из списка каталогов раздел «Компьютеры и связь — Интернет-доступ». Вам будет представлен перечень ссылок на 1113 наиболее посещаемых сайтов по этой проблематике.



Вопросы для размышления

1. В каких случаях активизация найденной с помощью поисковой системы ссылки на документ может выдавать сообщение об ошибке?

Практические задания

12.21. Осуществить поиск сайта «Информатика и информационные технологии» с помощью различных поисковых систем. Сравнить результаты поиска.

12.11.2. Специализированные поисковые системы

Специализированные поисковые системы позволяют искать информацию в других информационных «слоях» Интернета: серверах файловых архивов, почтовых серверах и др.

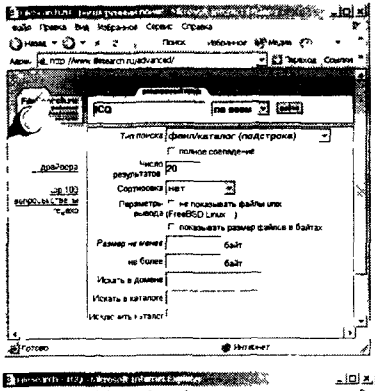
Поиск файлов. Для поиска файлов на серверах файловых архивов существуют специализированные поисковые системы двух типов: поисковые системы на основе использования баз данных и каталоги файлов. Для поиска файла в системе с использованием базы данных достаточно ввести имя файла в поле поиска и поисковая система выдаст URL-адреса мест хранения данного файла.

В базе данных российской файловой поисковой системы (http://www.filesearch.ru) содержатся сведения о 6 миллионах файлов, размещенных на двух тысячах серверов файловых архивов российской части Интернета.

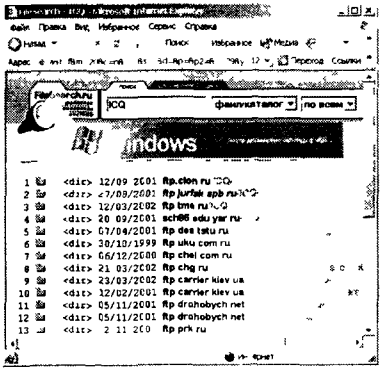
Осуществим сначала поиск файла программы интерактивного общения ICQ в базе данных российской файловой поисковой системы.

Поиск файлов

1. Открыть в браузере сервер www.filesearch.ru.
   В поле поиска ввести имя файла, например ICQ.
   Другие поля формы поиска позволяют уточнить условия поиска, но их заполнение необязательно.



2. Через некоторое время в окне браузера появятся результаты поиска, то есть перечень ссылок на серверы файловых архивов, на которых хранится этот файл.
   Активизировав одну из ссылок, вы соединитесь с файловым сервером и можете приступить к загрузке найденного файла на свой компьютер.



Если имя файла неизвестно, но зато известно его назначение (например, драйвер), то можно воспользоваться тематическим каталогом драйверов.

3. Для поиска, например, драйвера принтера на начальной странице поисковой системы щелкнуть по ссылке *Драйверы*.
   Затем выбрать в иерархическом дереве каталогов нужный тип принтера и активизировать ссылку на него.
   Будет выведен перечень URL-адресов серверов файловых архивов, откуда можно загрузить требуемый драйвер.

Поиск адресов электронной почты. Специализированные поисковые системы позволяют искать адрес электронной почты по имени человека или, наоборот, имя человека, хозяина определенного адреса электронной почты. Примером такой системы может служить поисковая система WhoWhere? (КтоГде?), расположенная по адресу: http://www.whowhere.com.

Практические задания

12.22. Осуществить поиск последних версий драйверов для периферийных устройств вашего компьютера.

12.12. Интерактивное общение в Интернете

В последнее время все более широко распространяется интерактивное общение в Интернете в реальном режиме времени. Увеличившаяся скорость передачи данных и возросшая производительность компьютеров дают пользователям возможность не только обмениваться текстовыми сообщениями в реальном времени, но и осуществлять аудио- и видеосвязь.

Серверы интерактивного общения. В Интернете существуют тысячи серверов Internet Relay Chat (IRC), на которых реализуется интерактивное общение. Любой пользователь может подключиться к такому серверу и начать общение с одним из посетителей этого сервера или участвовать в коллективной встрече.

Простейший способ общения *разговор* (*chat*) — это обмен сообщениями, набираемыми с клавиатуры. Вы вводите сообщение с клавиатуры, и оно высвечивается в окне, которое одновременно видят все участники встречи.

Если ваш компьютер, а также компьютеры собеседников оборудованы звуковой картой, микрофоном и наушниками или акустическими колонками, то вы можете обмениваться звуковыми сообщениями. Однако «живой» разговор одновременно возможен только между двумя собеседниками.

Для того чтобы вы могли видеть друг друга, то есть обмениваться видеоизображениями, к компьютерам должны быть подключены видеокамеры. Обычные аналоговые видеокамеры подключаются к специальным видеоплатам, а цифровые камеры — к параллельному порту компьютера.

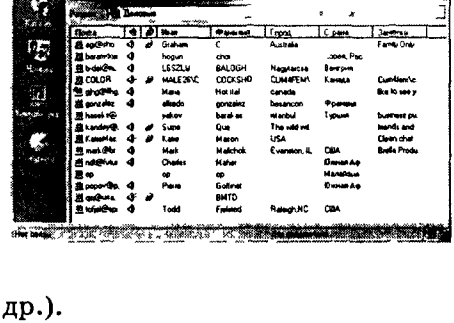
Конечно, качество звука и изображения в большой мере зависит от скорости модема и пропускной способности канала связи, которые должны быть не менее 28,8 Кбит/с.

Интерактивное общение на серверах Интернета может быть реализовано в форме *обмена текстовыми сообщениями, аудио- или видеоконференций.*

Для организации интерактивного общения необходимо специальное программное обеспечение (например, программа NetMeeting, которая входит в состав Internet Explorer).

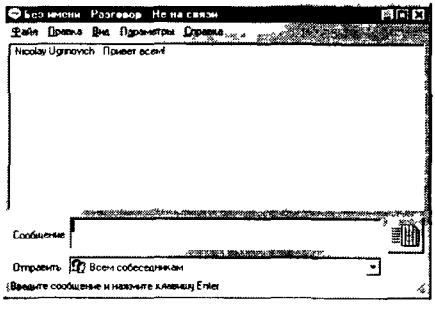
Интерактивное общение с использованием программы NetMeeting

1. После загрузки NetMeeting в поле *Сервер* из списка необходимо выбрать один из серверов, на котором происходит общение (например, ils.microsoft.com).
   В главном окне появится список участников общения, содержащий некоторые сведения о каждом из них (имя, фамилия, адрес электронной почты, страна и др.).



В строках некоторых участников имеются значки микрофона и видеокамеры, это обозначает потенциальную возможность организации с ними обмена аудио- или видеосообщениями.

2. Для присоединения к беседе необходимо ввести команду [Сервис-Разговор]. В появившемся окне *Разговор* в поле *Сообщение* можно вводить с клавиатуры текст сообщений. В поле *Отправить* можно выбрать из списка получателя или получателей сообщения. В основном окне программы появляются сообщения всех участников разговора.



Интерактивное общение с помощью ICQ. В последние годы большую популярность приобрело интерактивное общение через серверы ICQ (эта трехбуквенная аббревиатура образована из созвучия слов «I seek you» — «Я ищу тебя»).

В настоящее время в системе ICQ зарегистрировано более 150 миллионов пользователей, причем каждый пользователь имеет уникальный идентификационный номер. После подключения к Интернету пользователь может начинать общение с любым зарегистрированным в системе ICQ и подключенным в данный момент к Интернету пользователем. Программа уведомляет о присутствии в данное время в Интернете абонентов из предварительно составленного списка (My Contact List) и дает возможность инициализировать контакт с ними.

Для того чтобы стать абонентом системы ICQ, достаточно скачать программу ICQ-клиент (рис. 12.16) с файлового сервера (например, www freeware ru) и в процессе ее установки на компьютер зарегистрироваться.

Система интерактивного общения ICQ интегрирует различные формы общения: электронную почту, обмен текстовыми сообщениями (chat), интернет-телефонию, передачу файлов, поиск в сети людей и так далее.

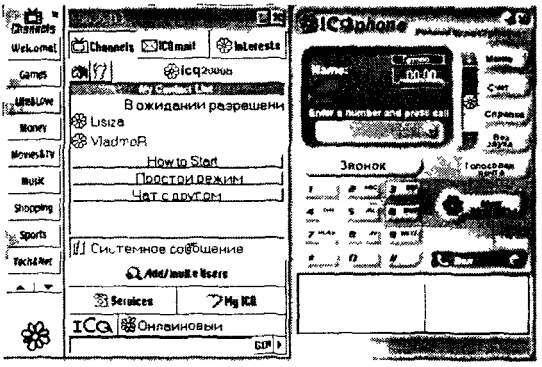


Рис. 12.16 Программа интерактивного общения ICQ

Интернет-телефония. Интернет-телефония дает возможность пользователю Интернета использовать телефонную связь компьютер–телефон, компьютер–компьютер и телефон–компьютер. Провайдеры Интернет-телефонии обеспечивают такую связь с помощью специальных телефонных серверов Интернета, которые подключены и к Интернету, и к телефонной сети.

Мобильный Интернет. С мобильного телефона на компьютер, подключенный к Интернету, и с компьютера на мобильный телефон можно отправлять SMS (Short Messsage Service — короткие текстовые сообщения).

Для беспроводного доступа с мобильных телефонов к информационным и сервисным ресурсам Интернета используется протокол WAP (Wireless Application Protocol). Для работы в сети Интернет по этому протоколу не нужны дополнительные устройства (компьютер и модем), достаточно одного мобильного телефона с поддержкой WAP.

WAP-сайты располагаются на Web-серверах и представлены в специальном формате WML (Wireless Markup Language). Этот язык разметки специально адаптирован под возможности мобильного телефона — двухцветную графику, маленький экран и небольшую память.

WAP-сайты содержат разнообразные политические, экономические и спортивные новости, прогноз погоды, курс валют и так далее. Можно также отправить e-mail и принять участие в WAP-чате.

Полноценный высокоскоростной доступ в Интернет с мобильного телефона можно осуществить по технологии GPRS (General Packet Radio Service). В этом случае можно работать с WAP-сайтами непосредственно с мобильного телефона, а на подключенном к нему компьютере можно просмат-

ривать HTML-страницы, перекачивать файлы, работать с электронной почтой и любыми другими ресурсами Интернета.

В технологии GPRS максимально возможная скорость передачи данных составляет 171,2 Кбит/с — это приблизительно в 3 раза больше скорости доступа по коммутируемым телефонным линиям, и почти в 12 раз больше скорости передачи данных в мобильных телефонных сетях стандарта GSM (9,6 Кбит/с).

Вопросы для размышления

1. В чем состоит различие между интернет-телефонией и мобильным Интернетом?

Практические задания

12.23. Установить программу интерактивного общения ICQ и зарегистрироваться в системе.

12.13. Мультимедиа технологии в Интернете

В Интернете существует достаточно большое количество серверов, на которых хранятся мультимедиа (звуковые, графические и видео-) файлы. Мультимедиа файлы имеют большой информационный объем: объем высококачественного звукового файла в цифровом формате, содержащего звучание длительностью в 1 секунду, составляет 187,5 Кбайт (см. параграф 2.13), высококачественного графического файла — 1,37 Мбайт (см. параграф 2.12), а одна секунда видео (из расчета 25 кадров в секунду) — 34,25 Мбайт.

Таким образом, для передачи по компьютерным сетям мультимедиа файлов в стандартных цифровых форматах требуются линии связи с высокой пропускной способностью, а воспроизведение файлов возможно только после их предварительной загрузки на локальный компьютер.

Для прослушивания и просмотра мультимедиа файлов непосредственно в процессе их получения из сети в режиме реального времени были разработаны специальные методы, реализующие технологию потокового сжатия, передачи и воспроизведения звуковых и видеоданных.

Принцип сжатия основан на удалении психофизиологической избыточности передаваемой звуковой или видеоинформации, то есть на удалении некоторых избыточных для человека частот в звуковом или видеосигнале, которые он все равно не воспринимает. Например, если воспроизводится громкий звук на частоте 1000 Гц, то более слабый звук на частоте 1100 Гц уже не будет слышен человеку; если в изображении имеются очень яркие точки, то соседние точки человек «не видит».

Технология сжатия MP3. Для сжатия звуковых файлов был разработан и в настоящее время широко используется стандарт MPEG layer3 (сокращенно MP3). Название свое этот стандарт получил по имени организации-разработчика (Moving Picture Experts Group — экспертная группа кинематографии). MP3 при сжатии разбивает весь частотный спектр звукового сигнала на части, а затем удаляет звуки, не воспринимаемые человеком.

Для передачи и прослушивания звука в формате MP3 по сети в режиме реального времени достаточно пропускной способности линий связи 128 Кбит/с. Это привело к высокой популярности этого формата и накоплению на серверах Интернета огромного количества музыкальных и других записей в формате MP3. Для прослушивания таких записей используются программные проигрыватели (WinAmp, MusicMath Jukebox и др.). Однако прослушивание по сети звуковых файлов в формате MP3 в режиме реального времени с использованием модемного подключения (максимальная скорость 56 Кбит/с) невозможно из-за недостаточной скорости передачи информации.

Технологии потокового воспроизведения. Широкое распространение в Интернете получили технологии передачи потокового звука и видео, которые можно использовать и при модемном подключении. Компанией Progressive Networks были разработаны технологии RealAudio и RealVideo (в настоящее время Real8), а компанией Microsoft — Windows Media Technology 7 (WMT7).

Эти технологии передают звуковые и видеофайлы по частям в буфер локального компьютера, что обеспечивает возможность их потокового воспроизведения даже при исполь-

зовании модемного подключения. Снижение скорости передачи по каналу может приводить к временным пропаданиям звука или пропускам видеокадров. Для прослушивания и просмотра таких файлов необходимы специальные проигрыватели (RealPlayer или Windows Media Player).

В настоящее время потоковые аудио- и видеотехнологии получили в Интернете широкое распространение. Существует достаточно много радио- и телевизионных станций, которые осуществляют вещание через Интернет. Широкой популярностью пользуются LiveCam, видеокамеры, установленные в самых разных местах (на улицах городов, в музеях, в заповедниках и так далее) по всему миру и непрерывно передающие изображение.

На рис. 12.17 представлен прием телеканала ТВЦ с помощью RealPlayer с узла www.tvc.ru и радиостанции Радио-101 с помощью Windows Media Player с узла www.101.ru. В процессе приема можно работать в других приложениях и слушать или смотреть новости в фоновом режиме.

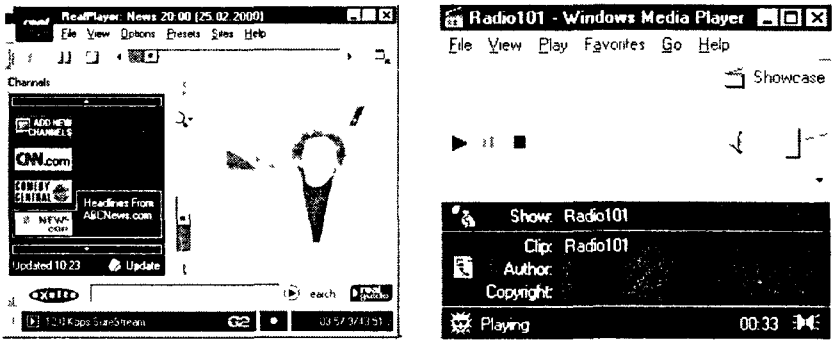


Рис. 12.17. Прием теле- и радиопрограмм через Интернет

Задания

12.24. Во сколько раз необходимо сжимать звуковой файл для потоковой передачи с помощью модемного соединения со скоростью 28,8 Кбит/с, если реализуется двоичное 16-битное кодирование с частотой дискретизации 22 кГц?

12.25. Какую пропускную способность должна иметь линия связи для потоковой передачи без сжатия видеоизображения размером 640×480 точек с палитрой из 65 536 цветов и с частотой 10 кадров в секунду?

12.14. Электронная коммерция в Интернете

Электронная коммерция в Интернете (e-commerce) — это коммерческая деятельность в сфере рекламы и распространения товаров и услуг посредством использования сети Интернет. В настоящее время электронная коммерция быстро развивается и, по статистике, уже более 100 миллионов человек во всем мире совершили хотя бы одну покупку в Интернет-магазинах, а годовой оборот электронной коммерции в 2000 году превысил 100 миллиардов долларов.

Одной из самых быстроразвивающихся областей электронной коммерции является хостинг (от английского слова host — сервер), то есть услуги по размещению информации во Всемирной паутине. Хостинг включает в себя предоставление дискового пространства для размещения Web-сайтов на Web-сервере, предоставления доступа к ним по каналу связи с определенной пропускной способностью, а также прав администрирования сайта.

Важной составляющей электронной коммерции является информационно-рекламная деятельность. Многие фирмы размещают на своих Web-сайтах в Интернете важную для потребителя информацию (описание товаров и услуг, их стоимость, адрес фирмы, телефон и e-mail, по которым можно сделать заказ, и др.). Существуют специализированные серверы, предоставляющие потребителю систематизированную (по видам товара, производителям, ценам и др.) информацию об определенной группе товаров. Например, на сервере www.newman.ru содержится информация о ценах на все виды компьютерного оборудования, которые предлагают различные фирмы в Москве.

Реклама в Интернете реализуется с помощью баннеров (от английского слова «*banner*» — «рекламный заголовок»). В Интернете баннер представляет собой небольшую прямоугольную картинку, на которой размещается реклама Web-сайта или Web-страницы.

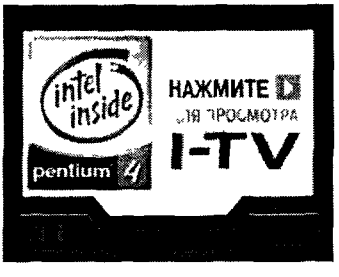


Рис. 12.18. Баннер

Баннеры могут быть как статическими (показывается одна и та же картинка), так и динамическими (картинки постоянно меняют-

ся). Щелчок по баннеру мышью приводит к переходу на соответствующий сайт или страницу, где можно более подробно узнать о том, что рекламирует баннер.

Баннеры размещаются на сайтах либо на платной основе, либо путем обмена. Использование системы обмена баннерами, которая связывает между собой множество сайтов и позволяет им рекламировать друг друга, повышает посещаемость каждого из них.

Широкое распространение в Интернете получила *электронная торговля*. Простейшим ее вариантом является виртуальная «*барахолка*» (доска объявлений), где продавцы и покупатели просто обмениваются информацией о предлагаемом товаре (аналог газеты «Из рук в руки») — рис. 12.19.

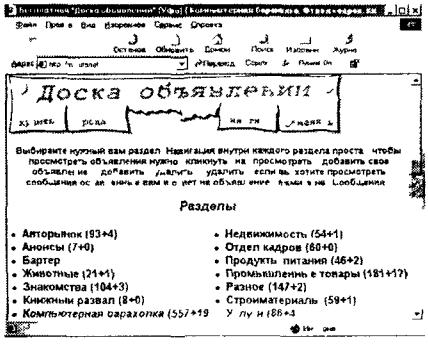


Рис. 12.19. Доска объявлений

Интересной формой электронной торговли являются *Интернет-аукционы*. На такие аукционы выставляются самые разные товары: произведения искусства, компьютерная техника, автомобили и так далее. Например, на сервере www greatdomains ru на аукцион выставляются доменные имена Интернета (рис. 12.20).

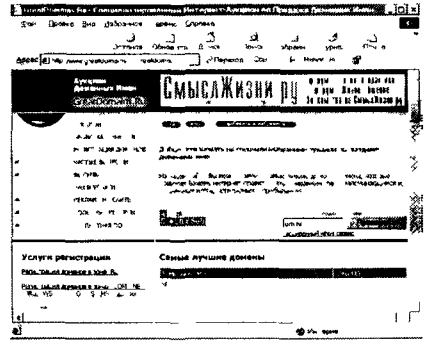


Рис. 12.20. Интернет-аукцион доменных имен

Самой удобной для покупателя формой электронной торговли являются *интернет магазины*. В российском Интернете существуют уже сотни магазинов, в которых можно купить все: компьютеры и программы, книги и CD, продукты питания и др.

Например, в виртуальном магазине «Библио-Глобус» (shop biblio-globus ru) можно найти более 90 000 библиографических описаний книг и более 25 000 наименований находится ежедневно в продаже (рис. 12.21).

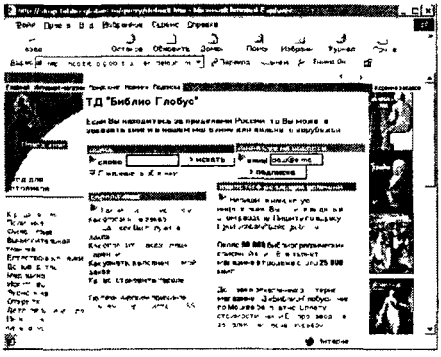


Рис. 12.21
Интернет-магазин
«Библио-Глобус»

Покупатель в Интернет-магазине имеет возможность ознакомиться с товаром (техническими характеристиками, внешним видом товара и так далее), а также его ценой. Выбрав товар, потребитель может сделать непосредственно из Интернета заказ на его покупку, в котором указывается форма оплаты, время и место доставки и так далее. Оплата производится либо наличными деньгами после доставки товара, либо по кредитным карточкам.

В последнее время для расчетов через Интернет стали использоваться *цифровые деньги*. Покупатель перечисляет опреленную сумму обычных денег в банк, а взамен получает определенную сумму цифровых денег, которые существуют только в электронном виде и хранятся в «кошельке» (с использованием специальной программы) на компьютере покупателя. При расчетах через Интернет цифровые деньги поступают к продавцу, который переводит их в банк, а взамен получает обычные деньги.

Практические задания

12.26. Найти с использованием поисковых систем адреса Интернет-магазинов и ознакомиться с правилами электронной торговли.

Глава 13

Основы языка гипертекстовой разметки документов

13.1. Web-сайты и Web-страницы

Публикации во Всемирной паутине реализуются в форме Web-сайтов. Web-сайт по своей структуре напоминает журнал, который содержит информацию, посвященную какой-либо теме или проблеме. Как журнал состоит из печатных страниц, так и Web-сайт состоит из компьютерных Web-страниц.

Сайт является *интерактивным* средством представления информации. Интерактивность сайта обеспечивают различные формы, с помощью которых посетитель сайта может зарегистрироваться на сайте, заполнить анкету и так далее.

Обычно сайт имеет титульную страницу (страницу с оглавлением), на которой имеются гиперссылки на его основные разделы (Web-страницы). Гиперссылки также имеются на других Web-страницах сайта, что обеспечивает возможность пользователю свободно перемещаться по сайту.

Web-сайты обычно являются *мультимедийными*, так как кроме текста могут содержать иллюстрации, анимацию, звуковую- и видеоинформацию.

Web-страницы сайта могут содержать *динамические объекты* (исполнимые модули), созданные с использованием сценариев на языках JavaScript и VBScript или элементов управления ActiveX. Расположенные на сайте управляющие элементы (например, кнопки) позволяют пользователю запускать те или иные динамические объекты.

Web-сайт состоит из Web-страниц, объединенных гиперссылками. Web-страницы могут быть интерактивными и могут содержать мультимедийные и динамические объекты.

Создание Web-сайтов реализуется с использованием языка разметки гипертекстовых документов HTML. Технология HTML состоит в том, что в обычный текстовый документ вставляются управляющие символы (тэги) и в результате мы получаем Web-страницу. Браузер при загрузке Web-страницы представляет ее на экране в том виде, который задается тэгами.

Основными достоинствами HTML-документов являются:

- малый информационный объем;
- возможность просмотра на персональных компьютерах, оснащенных различными операционными системами.

Для создания Web-страниц используются простейшие текстовые редакторы, которые не включают в создаваемый документ управляющие символы форматирования текста. В качестве такого редактора в Windows можно использовать стандартное приложение Блокнот.

Рассмотрим, как создаются Web-сайты, на примере разработки тематического сайта «Компьютер». Сначала необходимо разработать проект сайта, то есть определить, сколько Web-страниц будет входить в сайт, какова будет их тематика и как они будут связаны между собой.

Пусть наш сайт кроме титульной страницы «Компьютер» будет содержать:

- страницу «Программы», содержащую классификацию программного обеспечения;
- страницу «Словарь», содержащую словарь компьютерных терминов;
- страницу «Комплектующие» с ценами на устройства компьютера;
- страницу «Анкета», содержащую анкету для посетителей сайта.

Практические задания

13.1. На сайте по адресу iit.metodist.ru ознакомиться с Web-сайтами, участниками московских конкурсов ученических компьютерных проектов.

13.2. Форматирование текста и размещение графики

Создание структуры Web-страницы. Приступим к созданию титульной Web-страницы сайта. Для этого используем привычный и достаточно удобный для этих целей текстовый редактор Блокнот.

Создание Web-сайта «Компьютер»

1. Открыть окно текстового редактора Блокнот.

Вид Web-страницы задается тэгами, которые заключаются в угловые скобки. Тэги могут быть одиночными или парными, для которых обязательно наличие открывающего и закрывающего тегов (такая пара тэгов называется *контейнером*). Закрывающий тэг содержит прямой слэш (/) перед обозначением. Тэги могут записываться как прописными, так и строчными буквами.

HTML-код страницы помещается внутрь контейнера <HTML></HTML>. Без этих тэгов браузер не в состоянии определить формат документа и правильно его интерпретировать. Web-страница разделяется на две логические части: заголовок и содержание.

Заголовок Web-страницы заключается в контейнер <HEAD></HEAD> и содержит название документа и справочную информацию о странице (например, тип кодировки), которая используется браузером для ее правильного отображения.

Название Web-страницы содержится в контейнере <TITLE></TITLE> и отображается в строке заголовка браузера при просмотре страницы. Назовем нашу Web-страницу «Компьютер»:

```
<HEAD>
<TITLE>Компьютер</TITLE>
</HEAD>
```

Основное содержание страницы помещается в контейнер <BODY></BODY> и может включать текст, таблицы, бегущие строки, ссылки на графические изображения и звуковые файлы и так далее. Поместим для начала на страницу текст «Все о компьютере»:

```
<BODY>
Все о компьютере
</BODY>
```

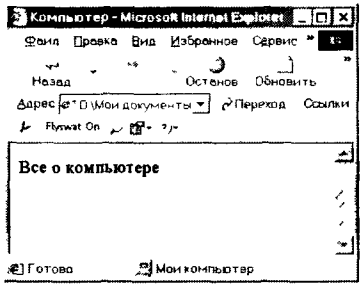
Созданную Web-страницу необходимо сохранить в виде файла. Принято сохранять титульный файл сайта, то есть тот, который первый загружается в браузер, под именем index.htm. В качестве расширения файла Web-страницы можно также использовать html.

Рекомендуется создать для размещения сайта специальную папку и сохранять все файлы разрабатываемого сайта в этой папке.

Необходимо различать имя файла index.htm, то есть имя, под которым Web-страница хранится в файловой системе, и собственно имя Web-страницы «Компьютер», которая высвечивается в верхней строке окна браузера и в первую очередь анализируется поисковыми системами. Имя Web-страницы должно в максимальной степени соответствовать ее содержанию.

2. В окне приложения Блокнот ввести HTML-код Web-страницы. Сохранить файл под именем index.htm в папке сайта. Загрузить этот файл в окно браузера для просмотра.

```
<HTML>
<HEAD>
<TITLE>Компьютер</TITLE>
</HEAD>
<BODY>
Все о компьютере
</BODY>
</HTML>
```



Форматирование текста. Пока страница выглядит не слишком привлекательно. Мелкий шрифт и черный текст на белом фоне почти не обращают на себя внимания. С помощью HTML-тэгов можно задать различные параметры форматирования текста.

Размер шрифта для имеющихся в тексте заголовков задается тэгами от <H1> (самый крупный) до <H6> (самый мелкий). Заголовок страницы целесообразно выделить самым крупным шрифтом:

```
<H1>Все о компьютере</H1>
```

Некоторые тэги имеют атрибуты, которые являются именами свойств и могут принимать определенные значения. Так, заголовок по умолчанию выровнен по левому краю страницы, однако принято заголовок размещать по центру. Задать тип выравнивания заголовка для тэга заголовка по-

зволяет атрибут ALIGN, которому требуется присвоить определенное значение. Выравнивание по правой границе окна реализуется с помощью записи ALIGN="right", а по центру — ALIGN="center":

```
<H1 ALIGN="center">Все о компьютере</H1>
```

С помощью тэга FONT и его атрибутов можно задать параметры форматирования шрифта любого фрагмента текста. Атрибут FACE позволяет задать гарнитуру шрифта (например, FACE="Arial"), атрибут SIZE — размер шрифта (например, SIZE=4), атрибут COLOR — цвет шрифта (например, COLOR="blue").

Значение атрибута COLOR можно задать либо названием цвета (например, "red", "green", "blue" и так далее), либо его шестнадцатеричным значением. Шестнадцатеричное представление цвета использует RGB-формат "#RRGGBB", где две первые шестнадцатеричные цифры задают интенсивность красного (red), две следующие — интенсивность зеленого (green) и две последние — интенсивность синего (blue) цветов. Минимальная интенсивность цвета задается шестнадцатеричным числом 00, а максимальная — FF. Легко догадаться, что синему цвету будет соответствовать значение "#0000FF".

Таким образом, задать синий цвет заголовка можно с помощью тэга FONT с атрибутом COLOR:

```
<FONT COLOR="blue">
<H1 ALIGN="center">Все о компьютере</H1>
</FONT>
```

Заголовок целесообразно отделить от остального содержания страницы горизонтальной линией с помощью одиночного тэга <HR>.

Разделение текста на абзацы производится с помощью контейнера <P></P>. При просмотре в браузере абзацы отделяются друг от друга интервалами. Для каждого абзаца можно задать определенный тип выравнивания.

На титульной странице обычно размещается текст, кратко описывающий содержание сайта. Поместим на титульную страницу текст, разбитый на абзацы с различным выравниванием:

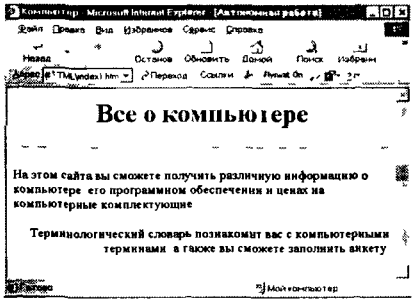
```
<P ALIGN="left">На этом сайте вы сможете получить различную информацию о компьютере, его программном обеспечении и ценах на компьютерные комплектующие.</P>
```

```
<P ALIGN="right">Терминологический словарь по-
знакомит вас с компьютерными терминами, а также
вы сможете заполнить анкету.</P>
```

Таким образом, если вставить в текст определенную последовательность тэгов, то мы получим Web-страницу, содержащую отцентрированный заголовок синего цвета, выводимый крупным шрифтом и отделенный горизонтальной линией от остального текста.

3. В окне приложения Блокнот в контейнер <BODY> вставить последовательность тэгов и просмотреть результат в браузере:

```
<FONT COLOR="blue">
<H1 ALIGN="center">
Все о компьютере
</H1>
</FONT>
<HR>
<P ALIGN="left">На
этом сайта...</P>
<P ALIGN="right">
Терминологический
словарь ...</P>
```



Вставка изображений. На Web-страницы обычно помещают изображения, чтобы сделать их визуально более привлекательными. На Web-страницах могут размещаться графические файлы трех форматов — GIF, JPG и PNG. Если рисунок сохранен в другом формате, то его необходимо предварительно преобразовать в один из вышеуказанных форматов с помощью графического редактора. Для этих целей можно использовать редактор Photo Editor, который входит в пакет Microsoft Office.

На титульной странице создаваемого сайта уместно разместить изображение того объекта, которому посвящен сайт (в нашем случае — компьютера). Многочисленные фотографии компьютеров можно найти, например, на сайтах производителей и продавцов компьютерной техники.

4. «Скачать» изображение компьютера из Интернета и сохранить его в файле с именем computer.gif в каталоге сайта.

Для вставки изображения используется тэг <IMG> с атрибутом SRC, который указывает на место хранения файла на локальном компьютере или в Интернете. Если графический

файл находится на локальном компьютере в том же каталоге, что и файл Web-страницы, то в качестве значения атрибута SRC достаточно указать только имя файла:

```
<IMG SRC="computer.gif">
```

Если файл находится в другом каталоге на данном локальном компьютере, то значением атрибута должно быть полное имя файла. Например:

```
<IMG SRC="C:\computer\computer.gif">
```

Если файл находится на удаленном сервере в Интернете, то должен быть указан URL-адрес этого файла. Например:

```
<IMG SRC="http://www.server.ru/computer.gif">
```

Иллюстрации на Web-страницах стали неотъемлемым элементом дизайна. Однако пользователи иногда в целях экономии времени отключают в браузере загрузку графических изображений и читают только тексты. Поэтому, чтобы *не терялся смысл и функциональность страницы, вместо рисунка* должен выводиться поясняющий текст.

Поясняющий текст выводится с помощью атрибута ALT, значением которого является текст, поясняющий, что должен был бы увидеть пользователь на рисунке:

```
<IMG SRC="computer.gif" ALT="Компьютер">
```

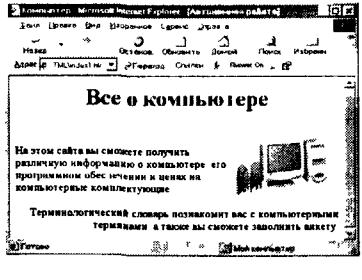
Расположить рисунок относительно текста различными способами позволяет атрибут ALIGN тэга <IMG>, который может принимать пять различных значений: TOP (верх), MIDDLE (середина), BOTTOM (низ), LEFT (слева) и RIGHT (справа).

Для того чтобы рисунок располагался по правому краю текста, тэг вставки изображения должен принять следующий вид:

```
<IMG SRC="computer.gif" ALT="Компьютер"
ALIGN="right">
```

5. В окне приложения Блокнот в контейнер <BODY> вставить перед абзацами текста тэг вставки изображения, просмотреть результат в браузере.

```
<IMG SRC="computer.gif"
ALT="Компьютер"
ALIGN="right">
```



Вопросы для размышления

1. Какие тэги (контейнеры) должны присутствовать в HTML-документе обязательно? Какова логическая структура Web-страницы?

Практические задания

13.2. Создать титульную страницу вашего сайта. Поэкспериментируйте с форматированием текста и размещением графики. Задайте для страницы фон.

13.3. Гиперссылки на Web-страницах

Первая титульная страница должна предоставлять посетителю Web-сайта возможность начать путешествие по сайту. Для этого на титульную страницу должны быть помещены гиперссылки на другие страницы сайта.

Для создания гиперсвязей между титульной страницей и другими страницами сайта необходимо, прежде всего, создать заготовки Web-страниц. Такие «пустые» страницы должны иметь заголовок, но могут пока не иметь содержания. Все создаваемые страницы необходимо сохранить в виде файлов в папке сайта.

Каждая страница будет содержать следующий HTML-код:

```
<HTML>
<HEAD>
<TITLE>Заголовок страницы</TITLE>
</HEAD>
<BODY>

</BODY>
</HTML>
```

6. Создать пустые страницы «Программы», «Словарь», «Комплектующие» и «Анкета» и сохранить их в файлах с именами software.htm, glossary.htm, hardware.htm и anketa.htm в каталоге сайта.

Гиперссылка состоит из двух частей: *указателя ссылки* и *адресной части ссылки*. Указатель ссылки — это то, что мы видим на Web-странице (текст или рисунок), обычно выделенный синим цветом и подчеркиванием. Активизация гиперссылки вызывает переход на другую страницу.

Адресная часть гиперссылки представляет собой URL-адрес документа, на который указывает ссылка. URL-адрес может быть абсолютным и относительным. Абсолютный URL-адрес документа полностью определяет компьютер, каталог и файл, в котором располагается документ.

Абсолютный адрес документа, находящегося на локальном компьютере, будет включать в себя путь к файлу и имя файла, например:

```
C:/Web-сайт/filename.htm
```

Абсолютный адрес документа, находящегося на удаленном компьютере в Интернете, будет включать имя сервера Интернета, путь к файлу и имя файла, например:

```
http://www.host.ru/Web-сайт/filename.htm
```

Относительный URL-адрес указывает на местоположение документа относительно того, в котором находится указатель ссылки. При разработке сайта рекомендуется входящие в него Web-страницы связывать относительными ссылками. Это позволит не изменять адресную часть ссылок при перемещении Web-сайта в другой каталог локального компьютера или при его публикации в Интернете.

Гиперссылка на Web-странице существует в форме указателя ссылки, щелчок по которому обеспечивает переход на Web-страницу, указанную в адресной части ссылки.

Прежде всего необходимо разместить на титульной странице тексты гиперссылок на каждую страницу сайта. Для представления гиперссылок удобнее всего выбрать названия страниц, на которые осуществляется переход.

Принято размещать гиперссылки в нижней части страницы, поэтому разместим их под введенным текстом в новом абзаце в одну строку разделенными несколькими пробелами. Такое размещение гиперссылок часто называют *панелью навигации*.

Панель навигации будет представлять собой абзац, выровненный по центру, в котором указатели гиперссылок разделены пробелами ( ):

```
<P ALIGH= "center">
[Программы]   [Словарь]  
[Комплектующие]   [Анкета]
</P>
```

Теперь для каждой гиперссылки определим адрес перехода. Для этого используется контейнер гиперссылки <A></A> с атрибутом HREF, значением которого является URL-адрес документа на локальном компьютере или в Интернете. Контейнер должен содержать указатель гиперссылки:

```
<A HREF="URL">Указатель гиперссылки</A>
```

7. Вставить в титульную страницу код, создающий панель навигации:

```
<P ALIGN="center">
[<A HREF="software.htm">Программы</A>]  
[<A HREF="glossary.htm">Словарь</A>]  
[<A HREF="hardware.htm">Комплектующие</A>] 
[<A HREF="anketa.htm">Анкета</A>]
</P>
```

Панель навигации на титульной странице создана, теперь активизация указателей гиперссылок будет приводить к переходу на другие страницы сайта.

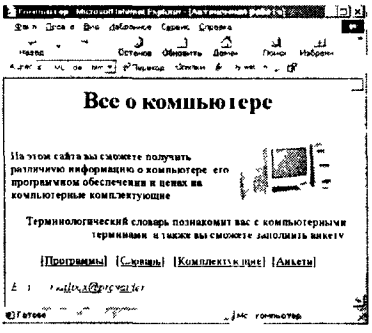
Полезно на титульной странице сайта создать ссылку на адрес электронной почты, по которому посетители могут связаться с администрацией сайта. Для этого необходимо атрибуту ссылки HREF присвоить URL-адрес электронной почты и вставить ее в контейнер <ADDRESS></ADDRESS>, который задает стиль абзаца, принятый для указания адреса.

8. Вставить в титульную страницу код, создающий ссылку на адрес электронной почты:

```
<ADDRESS>
<A HREF="mailto:mailbox@provaider.ru">E-mail:
mailbox@provaider.ru</A>
</ADDRESS>
```

По щелчку мыши по ссылке на адрес электронной почты будет открываться почтовая программа Outlook Express (или другая используемая по умолчанию почтовая программа), где в строке «Кому» будет указан заданный в ссылке адрес.

9. Созданная титульная страница Web-сайта «Компьютер» содержит заголовок, изображение компьютера, текст, панель навигации и ссылку на адрес электронной почты.



Вопросы для размышления

1. В каких случаях удобнее использовать в гиперссылках абсолютные, а в каких — относительные адреса?

Практические задания

13.3. Разместить панель навигации и электронный почтовый адрес на титульной странице разрабатываемого сайта.

13.4. Списки на Web-страницах

Довольно часто при размещении текста на Web-страницах удобно использовать списки в различных вариантах:

- нумерованные списки, когда элементы списка идентифицируются с помощью чисел;
- маркированные списки (в HTML их принято называть ненумерованными), когда элементы списка идентифицируются с помощью специальных символов (маркеров);
- списки определений позволяют составлять перечни определений в так называемой словарной форме.

Возможно создание и вложенных списков, причем вкладываемый список может по своему типу отличаться от основного.

На странице «Программы» разместим информацию об основных типах программного обеспечения компьютера в форме вложенного списка.

Сначала создадим главный нумерованный список основных категорий программного обеспечения. Список располагается внутри контейнера <OL></OL>, а каждый элемент списка определяется тэгом <LI>. С помощью атрибута TYPE тэга <OL> можно задать тип нумерации: арабские цифры (по умолчанию), "I" (римские цифры), "a" (строчные буквы) и др.

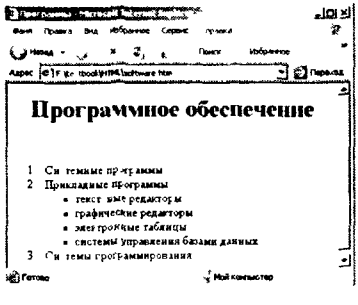
9. Открыть в Блокноте файл software.htm, ввести заголовок «Программное обеспечение» и добавить следующий HTML-код, задающий список:

```
<OL>
<LI>Системные программы
<LI>Прикладные программы
<LI>Системы программирования
</OL>
```

Создадим теперь вложенный ненумерованный список для одного из элементов основного списка. Список располагается внутри контейнера <UL></UL>, а каждый элемент списка определяется также тэгом <LI>. С помощью атрибута TYPE тэга <UL> можно задать вид маркера списка: "disc" (диск), "square" (квадрат) или "circle" (окружность).

10. Добавить HTML-код, задающий вложенный список для элемента <LI> Прикладные программы:

```
<UL>
<LI TYPE="square">
текстовые редакторы;
<LI> графические
редакторы;
<LI> электронные таблицы;
<LI> системы управления
базами данных.
</UL>
```

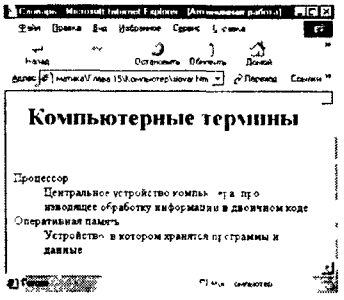


Страницу «Словарь» мы представим в виде словаря компьютерных терминов.

Для этого используем контейнер списка определений </DL>. Внутри него текст оформляется в виде *термина*, который выделяется непарным тэгом <DT>, и *определения*, которое следует за тэгом <DD>.

11. Открыть в Блокноте файл glossary.htm, ввести заголовок «Компьютерные термины» и добавить следующий HTML-код, задающий список определений:

```
<DL>
<DT>Процессор
<DD>Центральное устройство
компьютера, производящее
обработку информации в
двоичном коде.
<DT>Оперативная память
<DD>Устройство, в котором
хранятся программы и
данные.
</DL>
```



Практические задания

13.4. Разместить на Web-странице вложенный список, содержащий разные типы списков.

13.5. Формы на Web-страницах

Для того чтобы посетители сайта могли не только просматривать информацию, но и отправлять сведения его администраторам сайта, на его страницах размещают формы. Формы включают в себя управляющие элементы различных типов: текстовые поля, раскрывающиеся списки, флажки, переключатели и так далее.

Разместим на странице «Анкета» анкету для посетителей, чтобы выяснить, кто из наших посетителей, с какими целями и с помощью каких программ получает и использует информацию из сети Интернет, а также выясним, какую информацию они хотели бы видеть на нашем сайте.

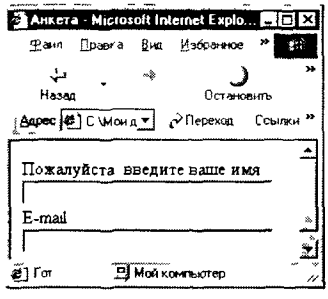
Вся форма заключается в контейнер `<FORM></FORM>`. В первую очередь выясним имя посетителя нашего сайта и его электронный адрес, чтобы иметь возможность ответить ему на замечания и поблагодарить за посещение сайта.

Текстовые поля. Для получения этих данных разместим на форме два однострочных текстовых поля для ввода информации. Текстовые поля создаются с помощью тэга `<INPUT>` со значением атрибута `TYPE="text"`. Атрибут `NAME` является обязательным и служит для идентификации полученной информации. Значением атрибута `SIZE` является число, задающее длину поля ввода в символах.

Для того чтобы анкета «читалась», необходимо разделить строки с помощью тэга перевода строки `<BR>`.

12. Открыть в Блокноте файл anketa.htm и добавить HTML-код, создающий текстовые поля для ввода данных. Просмотреть страницу в браузере:

```
<FORM>
Пожалуйста, введите ваше
имя: <BR>
<INPUT TYPE="text"
NAME="name" SIZE=30> <BR>
E-mail: <BR>
<INPUT TYPE="text"
NAME="e-mail" SIZE=30>
<BR>
</FORM>
```



Переключатели. Далее, мы хотим выяснить, к какой группе пользователей относит себя посетитель. Предложим выбрать ему один из нескольких вариантов: учащийся, студент, учитель.

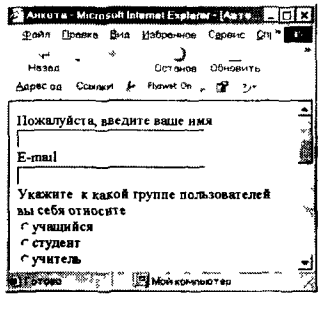
Для этого необходимо создать группу переключателей («радиокнопок»). Создается такая группа с помощью тэга <INPUT> со значением атрибута TYPE="radio". Все элементы в группе должны иметь одинаковые значения атрибута NAME. Например, NAME="group".

Еще одним обязательным атрибутом является VALUE, которому присвоим значения "schoolboy", "student" и "teacher". Значение атрибута VALUE должно быть уникальным для каждой «радиокнопки», так как при ее выборе именно они передаются серверу.

Атрибут CHECKED, который задает выбор кнопки по умолчанию, в данном случае не используется.

13. Добавить HTML-код, создающий группу переключателей для выбора одного варианта. Просмотреть страницу в браузере:

```
Укажите, к какой группе
пользователей вы себя от-
носите: <BR>
<INPUT TYPE="radio"
NAME="group" VALUE=
"schoolboy">учащийся<BR>
<INPUT TYPE="radio"
NAME="group" VALUE=
"student">студент<BR>
<INPUT TYPE="radio"
NAME="group" VALUE=
"teacher">учитель<BR>
```



Флажки. Далее, мы хотим узнать, какими сервисами Интернета наш посетитель пользуется наиболее часто. Здесь из предложенного перечня он может выбрать одновременно несколько вариантов, пометив их флажками. Флажки создаются в тэге <INPUT> со значением атрибута TYPE="checkbox".

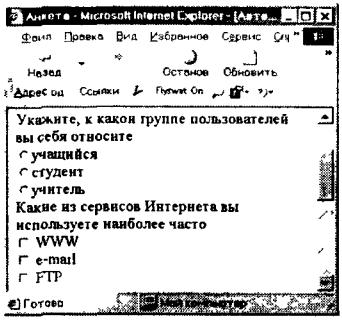
Флажки, объединенные в группу, могут иметь одинаковые значения атрибута NAME. Например, NAME="group".

Еще одним обязательным атрибутом является VALUE, которому присвоим значения "www", "e-mail" и "ftp". Значение атрибута VALUE должно быть уникальным для каждого флажка, так как при его выборе именно они передаются серверу.

Атрибут CHECKED, который задает выбор кнопки по умолчанию, в данном случае не используется.

14. Добавить HTML-код, создающий флажки для выбора нескольких вариантов. Просмотреть страницу в браузере:

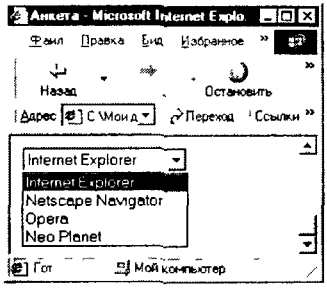
```
Какие из сервисов Интернета вы используете наиболее часто:
<BR>
<INPUT TYPE="checkbox" NAME="group" VALUE="www">
WWW<BR>
<INPUT TYPE="checkbox" NAME="group" VALUE=
"e-mail"> e-mail<BR>
<INPUT TYPE="checkbox" NAME="group" VALUE="ftp">
FTP<BR>
```



Поля списков. Теперь выясним, какой из браузеров предпочитает посетитель сайта. Перечень браузеров представим в виде раскрывающегося списка, из которого можно выбрать только один вариант. Для реализации такого списка используется контейнер <SELECT></SELECT>, в котором каждый элемент списка определяется тэгом <OPTION>. Выбираемый по умолчанию элемент задается с помощью атрибута SELECTED.

15. Добавить HTML-код, создающий раскрывающийся список для выбора одного варианта. Просмотреть страницу в браузере:

```
<SELECT NAME="browsers">
<OPTION SELECTED> Internet
Explorer
Internet Explorer
<OPTION>
Netscape Navigator
<OPTION> Opera
<OPTION> Neo Planet
</SELECT>
```

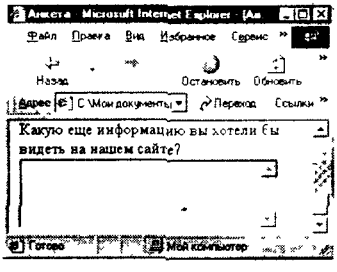


Текстовая область. В заключение поинтересуемся, что хотел бы видеть посетитель на наших страницах, какую информацию следовало бы в них добавить. Так как мы не можем знать заранее, насколько обширным будет ответ читателя, отведем для него текстовую область с линейкой прокрутки. В такое поле можно ввести достаточно подробный текст.

Создается такая область с помощью тэга <TEXTAREA> с обязательными атрибутами: NAME, задающим имя области, ROWS, определяющим число строк, и COLS — число столбцов области.

16. Добавить HTML-код, создающий текстовую область для ввода комментариев, просмотреть страницу в браузере:

```
Какую еще информацию вы
хотели бы видеть на нашем
сайте?
<BR>
<TEXTAREA NAME="resume"
ROWS=4 COLS=30>
</TEXTAREA
<BR>
```

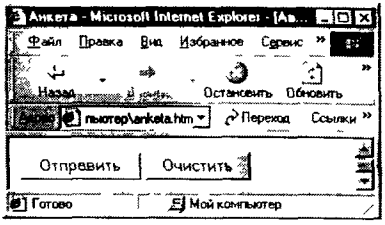


Отправка данных из формы. Отправка введенной в форму информации или очистка полей от уже введенной информации осуществляется с помощью кнопок. Кнопки создаются с помощью тэга <INPUT>. Для создания кнопки, которая отправляет информацию, атрибуту TYPE необходимо присвоить значение "submit", а атрибуту VALUE, который задает надпись на кнопке, — значение "Отправить".

Для создания кнопки, которая производит очистку формы, атрибуту TYPE необходимо присвоить значение "reset", а атрибуту VALUE — значение "Очистить".

17. Добавить HTML-код, создающий кнопки, просмотреть страницу в браузере:

```
<INPUT TYPE="submit"
VALUE="Отправить">
<INPUT TYPE="reset"
VALUE="Очистить">
```



Заполненная форма отправляется на сервер, где обрабатывается специальной программой — CGI-скриптом, или по электронной почте автору сайта, где он уже самостоятельно обрабатывает полученные данные.

Для того чтобы при щелчке по кнопке «Отправить» данные из формы передавались на сервер и там обрабатывались, необходимо указать, куда их отправить и какая программа будет их обрабатывать. Эти параметры задаются с помощью атрибута ACTION контейнера <FORM>.

```
<FORM ACTION="URL"></FORM>
```

Атрибут ACTION определяет URL-адрес программы, расположенной на Web-сервере, которая обрабатывает полученные данные из формы. Пусть программой, которая заносит данные из формы в базу данных, будет программа bd.exe. Обычно такие программы хранятся в каталоге cgi-bin. Тогда атрибут ACTION будет выглядеть следующим образом:

```
ACTION="http:/www.mycompany.ru/cgi-bin/bd.exe"
```

Практические задания

13.5. Разместите на своем сайте анкету для посетителей, использующую все известные вам элементы форм.

13.6. Инструментальные средства создания Web-страниц

Создание Web-страниц непосредственно на HTML требует больших усилий, времени и досконального знания синтаксиса языка. Применение специальных инструментальных программных средств (HTML-редакторов) делает работу по созданию Web-сайтов простой и эффективной.

Существуют мощные инструментальные системы разработки сайтов, например Microsoft FrontPage или Macromedia Dreamweawer. Для разработки отдельных Web-страниц можно использовать свободно распространяемые редакторы FrontPage Express (входит в состав Microsoft Internet Explorer) и Netscape Composer (входит в состав Netscape Communicator).

Рассмотрим в качестве примера использование Netscape Composer (рис. 13.1) для создания и редактирования Web-страниц без программирования на HTML. Composer позволяет форматировать текст, создавать различного типа списки, помещать на страницу графические изображения и таблицы, вставлять гиперссылки и так далее.

Процесс создания и редактирования страницы очень нагляден, так как производится в режиме WYSIWYG («What You See Is What You Get» — «Что видишь, то и получишь»).

Запустим Netscape Composer командой [Программы-Netscape Communicator-Netscape Composer]. Окно приложения представляет собой стандартное окно, в верхней части которого расположены строка меню и горизонтальные панели инструментов.

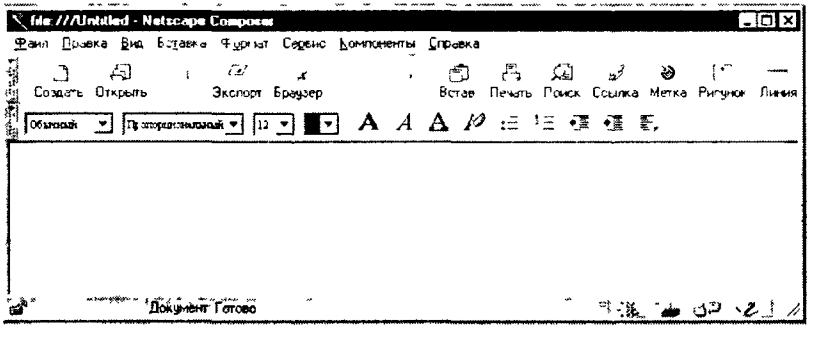


Рис. 13.1. Окно редактора Netscape Composer

Создадим с помощью этого редактора Web-страницу «Комплектующие», содержащую таблицу с наименованиями комплектующих и их ценами.

18. Ввести команду [Файл-Создать-Пустую Страницу].

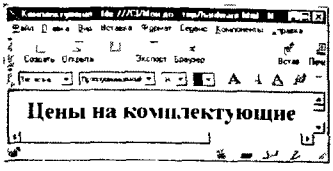
Сначала дадим странице название и сохраним ее в виде файла в папке сайта.

19. Ввести команду [Формат-Цвета и Свойства Страницы]. На появившейся диалоговой панели *Параметры страницы* в окне *Заголовок:* ввести текст «Комплектующие».

20. Ввести команду [Файл-Сохранить как...]. На диалоговой панели *Сохранить как...* ввести имя файла hardware.htm и выбрать папку для сохранения.

Введем заголовок страницы «Цены на комплектующие», установим требуемые размер и цвет шрифта, а также тип выравнивания.

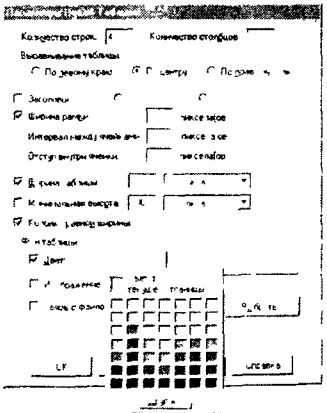
21. В окне приложения ввести текст. Задать команды:
[Формат-Заголовок-1]
[Формат-Выравнивание-По Центру]
[Формат-Цвет...]



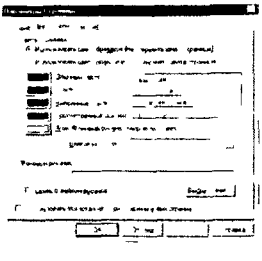
Заголовки различных уровней, различного типа списки и абзацы предоставляют большие возможности по форматированию текста на Web-странице. Однако, если необходимо соблюдать точное размещение текста и графики или распределять текстовую и числовую информацию по строкам и столбцам, целесообразно использовать таблицы.

Разместим информацию о ценах на комплектующие компьютера в таблице, которая будет содержать три столбца («№», «Наименование» и «Цена») и четыре строки.

22. Ввести команду [Вставка-Таблица-Таблица].
На появившейся диалоговой панели *Параметры Новой Таблицы* в текстовых окнах с помощью переключателей, флажков и раскрывающихся списков установить необходимые параметры таблицы, в том числе цвет фона таблицы.

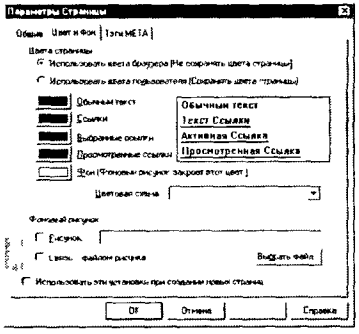


23. Ввести и отформатировать данные в таблице с помощью пункта меню *Формат*.
Ввести команду [Формат-Свойства Таблицы].
На диалоговой панели *Свойства Таблицы* установить необходимую ширину столбцов и другие параметры.



Для того чтобы Web-страницы выглядели привлекательнее, мы можем подбирать различные сочетания цветов для текста, непросмотренных, просмотренных и активных гиперссылок. В качестве фона страницы можно использовать цвет или какое-то графическое изображение. Как правило, это изображение небольшого размера, которое, повторясь много раз, заполняет все окно браузера, в котором отображается документ. Получаются своеобразные «обои» на странице.

24. Ввести команду [Формат-Цвета и Свойства Страницы]. На диалоговой панели *Параметры Страницы* на вкладке *Цвет и фон* установить оптимальную цветовую схему для отображения текста, гиперссылок и фона.



Готовый сайт хранится в каталоге \textbook\HTML\ CD-ROM

Практические задания

13.6. Используя инструментальное средство разработки Web-сайта, попробуйте улучшить дизайн вашего сайта.

13.7. Тестирование и публикация Web-сайта

Прежде чем разместить свой Web-сайт на сервере в Интернете, его необходимо тщательно протестировать, так как потенциальными посетителями вашего сайта являются десятки миллионов пользователей Интернета.

Необходимо просмотреть, как выглядят ваши страницы в наиболее распространенных браузерах Internet Explorer и Netscape Navigator (а они могут выглядеть по-разному). Необходимо проверить:

- нормально ли читается текст на выбранном фоне;

- правильно ли расположены рисунки;
- правильно ли работают гиперссылки.

Для публикации Web-сайта необходимо найти подходящее место на одном из серверов Интернета. Многие провайдеры предоставляют своим клиентам возможность бесплатного размещения Web-сайтов на своих серверах (бесплатный хостинг).

Для публикации Web-сайта необходимо получить от провайдера необходимые сведения:

- URL-адрес сайта;
- секретные имя пользователя и пароль, которые необходимы администратору сайта для его редактирования.

Технология публикации Web-сайтов может иметь несколько вариантов:

- для публикации можно воспользоваться инструментальным средством, которое использовалось для создания Web-сайта (например, Netscape Composer);
- организация, предоставившая место на своем сервере, может предложить публиковать сайты через Web-интерфейс с помощью браузера или с помощью оригинального Диспетчера файлов;
- наибольшие возможности при публикации предоставляют FTP-клиенты.

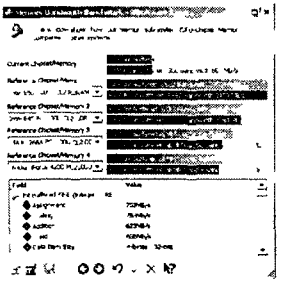
Практические задания

13.7. Найдите в Интернете серверы, которые предлагают услуги по размещению сайтов, и опубликуйте разработанный вами сайт.

Ответы и указания к решению

Ответы к главе 1. Компьютер и программное обеспечение

1.2. В окне программы активизировать тестирующий модуль *Memory Bandwidth Benchmark*.
Через несколько десятков секунд появится информационное окно, которое показывает быстродействие модулей памяти в сравнении с быстродействием четырех типов памяти.



1.3. В окне программы активизировать информационные модули *Mainboard Information* и *CPU&BIOS Information*.

1.6. http://schools.techno.ru/sch444, http://www.computer-museum.ru

Ответы к главе 2. Информация. Двоичное кодирование информации

2.1. 4 бита.

2.2. 8, 128.

2.3. Приближенно 2,58 бита, 6,17 бита, 4,32 бита, 2,80 бита.

2.4. 1,5 бита.

2.5. 6 битов, 7 битов.

2.7. В 10, в 2, в 8, в 16.

2.8. $x = 2$.

2.9. 8.

2.10. MCMXCIX.

2.11. 5, 6, 7, 7, 9, 18, 26, 191, 156.

2.13. $9_{10} = 1001_2 = 11_8 = 9_{16}$, $17_{10} = 10001_2 = 21_8 = 11_{16}$,
$243_{10} = 11110011_2 = 363_8 = F3_{16}$.

2.14. $0{,}2_{10} = 0{,}001_2 = 0{,}146_8 = 0{,}333_{16}$; $0{,}35_{10} = 0{,}010_2 = 0{,}263_8 = 0{,}599_{16}$.

2.15. $3{,}5_{10} = 11{,}1_2 = 3{,}4_8 = 3{,}8_{16}$;
$47{,}85_{10} = 101111{,}110_2 = 57{,}663_8 = 2F{,}D99_{16}$.

2.16.

0000	0	0100	4	1000	8	1100	C
0001	1	0101	5	1001	9	1101	D
0010	2	0110	6	1010	A	1110	E
0011	3	0111	7	1011	B	1111	F

2.17. $1111_2 = 17_8 = F_{16}$; $1010101_2 = 125_8 = 55_{16}$.

2.18. $0{,}01111_2 = 0{,}36_8 = 0{,}78_{16}$; $0{,}10101011_2 = 0{,}526_8 = 0{,}AB_{16}$.

2.19. $11{,}01_2 = 3{,}2_8 = 3{,}4_{16}$; $110{,}101_2 = 6{,}5_8 = 6{,}A_{16}$.

2.20. $46{,}27_8 = 100110{,}010111_2$; $EF{,}12_{16} = 11101111{,}0001001_2$.

2.21. $1101_2 = D_{16}$; $0{,}11111_2 > 0{,}22_8$; $35{,}63_8 > 16{,}C_{16}$.

2.22. $1010_2 + 10_2 = 1100_2$; $1010_2 - 10_2 = 1000_2$; $1010_2 \cdot 10_2 = 10100_2$; $1010_2 : 10_2 = 101_2$.

2.23. $5_8 + 4_8 = 11_8$; $17_8 + 41_8 = 60_8$.

2.24. $F_{16} - A_{16} = 5_{16}$; $41_{16} - 17_{16} = 2A_{16}$.

2.25. $17_8 + 17_{16} = 26_{16}$; $41_8 + 41_{16} = 142_8$.

2.26.

Десятичные числа	Прямой код	Обратный код	Дополнительный код
−50	0000000000110010	1111111111001101	1111111111001110
−500	0000000111110100	1111111000001011	1111111000001100

2.27. От − 32 768 до 32 767.

2.28. Максимальное значение чисел двойной точности с учетом возможной точности вычислений составит $8{,}98846567431157 \cdot 10^{307}$ (количество значащих цифр десятичного числа в данном случае ограничено 15–16 разрядами).

2.29. 234, 238, 236, 239, 252, 254, 242, 229, 240.

2.30. байт.

2.31. В кодировке КОИ8 — щбл, в кодировке ISO — нЕЬ.

2.32.

Разрешающая способность экрана	Глубина цвета (бит на точку)			
	8	16	24	32
800×600	469 Кбайт	938 Кбайт	1,4 Мбайт	1,8 Мбайт
1024×768	768 Кбайт	1,5 Мбайт	2,25 Мбайт	3 Мбайт
1280×1024	1,25 Мбайт	2,5 Мбайт	3,75 Мбайт	5 Мбайт

2.33. 16 бит $\cdot$ 44 000 $с^{-1} \cdot$ 10 с = 859, 375 Кбайт.

Ответы к главе 3. Основы логики и логические основы компьютера

3.2. $A = \{2 \times 2 = 4\}$; $B = \{3 \times 3 = 9\}$; $(A\&B)\vee(\overline{A}\&\overline{B})$

A	B	$A\&B$	$\overline{A}$	$\overline{B}$	$\overline{A}\&\overline{B}$	$(A\&B)\vee(\overline{A}\&\overline{B})$
0	0	0	1	1	1	1
0	1	0	1	0	0	0
1	0	0	0	1	0	0
1	1	1	0	0	0	1

3.6. а) $(A\vee\overline{A})\&B = 1\&\underline{B} = B$;
б) $A\&(A\vee B)\&(B\&\overline{B}) = A\&(A\vee B)\&0 = 0$.

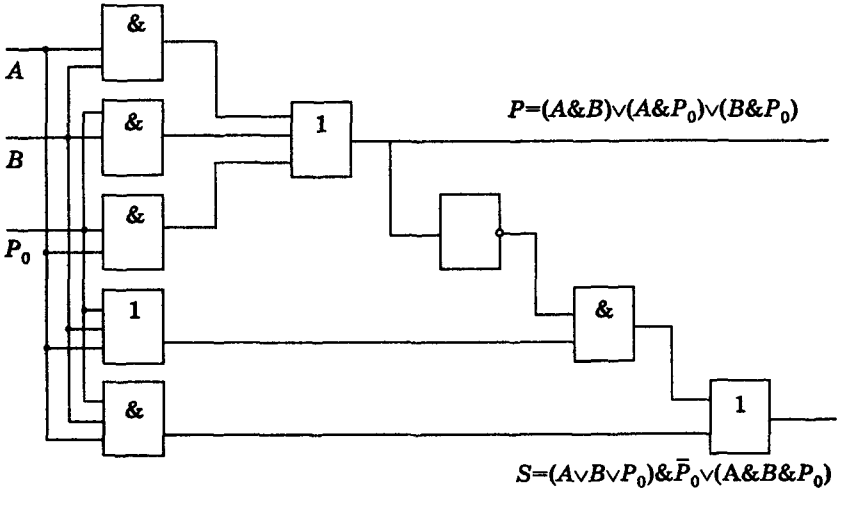
3.7. Возможны два варианта расписания:

1. Информатика	1. Математика
2. Математика	2. Физика
3. Физика	3. Информатика

3.8.

A	B	P_0	$P = (A\&B)\vee(A\&P_0)\vee(B\&P_0)$	$S = (A\vee B\vee P_0)\&\overline{P}\vee(A\&B\&P_0)$
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	0	1	1	0
1	1	1	1	1

3.9.



Ответы к главе 4. Основы алгоритмизации и объектно-ориентированного программирования

Готовые проекты хранятся в каталоге \textbook\VB\ в отдельных папках CD-ROM

4.4. Программный код:

```
'Открытие документа
Documents().Open FileName:=
"C:\Документы\Проба.doc"
'Печать документа
Documents("Проба.doc").PrintOut
Range:=wdPrintFromTo,   From:="1",To:="3"
'Сохранение документа
Documents("Проба.doc").Save
```

4.5. Программный код:

```
For i = 1 To 10
If Selection.Characters(i) = "a"
Then Selection.Characters(i).Bold = True
Else Selection.Characters(i).Italic = True
Next i
```

4.6. Form (Форма), CommandButton (Командная кнопка), TextBox (Текстовое поле), Label (Метка) и др.

4.7. Проект «Вывод сообщения» хранится в папке prjZ4-7.

4.8. Проект «Вывод сообщений» хранится в папке prjZ4-8.

4.9. Проект «Печать на форме» хранится в папке prjZ4-9.

1. Поместить на форму две кнопки. Для первой кнопки создать событийную процедуру, выводящую текст на форму с помощью метода **Print**.

   Для каждой строки в событийной процедуре необходимо задать параметры шрифта и цвет шрифта. Перед печатью каждой строки текста можно присвоить значения сразу нескольким свойствам шрифта с помощью инструкции **With...End With**.

   Цвет текста зададим, присвоив свойству ForeColor (цвет надписи) значение (цвет) с помощью функции QBColor(). Аргументом функции являются числа, каждому числу соответствует свой цвет (например, 2 — зеленый, 9 — синий, 12 — красный и так далее).

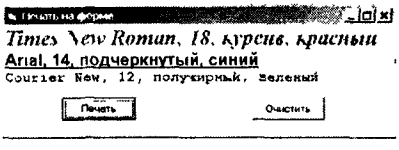
   Программный код для первой строки текста:

```
With Font
.Name = "Times New Roman"
.Size = 18
.Italic = True
```

```
End With
frm1.ForeColor = QBColor(12)
frm1.Print "Times New Roman, 18, курсив, красный"
```

Программный код для печати других строк создается аналогично.

2. Для очистки формы создадим событийную процедуру для второй кнопки с использованием метода `Cls`: `frm1.Cls`



4.10. От –2 147 483 648 до 2 147 483 647.

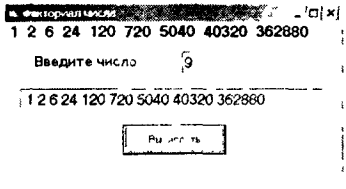
4.11. 3 и 11 ячеек.

4.12. Проект «Факториал числа» хранится в папке prjZ4-12.

1. Поместить на форму два текстовых окна `txtNum` (для ввода числа) и `txtF` (для визуализации процесса вычисления факториала), метку `lblNum` и кнопку `cmdStart`.
2. Объявить переменные и создать событийную процедуру вычисления факториала `cmdStart_Click()`. Для визуализации процесса вычисления факториала использовать метод **Print** и текстовое поле `txtF`:

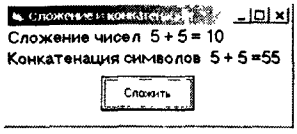
```
Dim bytI As Byte, intN As Integer, lngF As Long
Private Sub cmdStart_Click()
intN = txtNum.Text
lngF = 1
For bytI = 1 To intN
lngF = lngF * bytI
Print lngF;
txtF.Text = txtF.Text + Str(lngF)
Next bytI
End Sub
```

3. Запустить проект и щелкнуть по кнопке *Вычислить*. На форме и в текстовом поле будут напечатаны результаты процесса вычисления факториала.



4.13. Проект «Сложение и конкатенация» хранится в папке prjZ4-13.

Программный код:




```
Private Sub cmd1_Click()
bytA = 5 + 5
strB = "5" + "5"
Print "Сложение чисел: 5 +
5 ="; bytA
Print "Конкатенация символов: 5 + 5 ="; strB
End Sub
```

4.14. Проект «Истинность высказывания» хранится в папке prjZ4-14. Программный код:

```
Dim blnA As Boolean
Private Sub cmd1_Click()
blnA = ((2 * 2 = 4) And (3 * 3 = 10)) Or
((2 * 2 = 5) And (3 * 3 = 9))
Print "Высказывание 2*2=4 и 3*3=10 или 2*2=5
и 3*3=9   "; blnA
End Sub
```

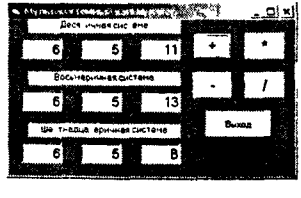
4.15. 16 байтов · 11 = 176 байтов, 176 ячеек.

4.16. Проект «Мультисистемный калькулятор» хранится в папке prjZ4-16.

1. Разместить на форме девять текстовых полей (txt1Dec, txt2Dec, txt3Dec, txt1Oct, txt2Oct, txt3Oct, txt1Hex, txt2Hex, txt3Hex) для ввода и вывода чисел, четыре кнопки (cmdPlus, cmdMinus, cmdUmn, cmdDelen) для создания событийных процедур, реализующих арифметические операции, и три метки (lblDec, lblOct, lblHex) для вывода поясняющих надписей над текстовыми полями.
2. Создать событийную процедуру сложения чисел:

```
Sub cmdPlus_Click()
txt3Dec.Text = Val(txt1Dec.Text) +
Val(txt2Dec.Text)
txt3Oct.Text = Oct(Val(txt1Oct.Text) +
Val(txt2Oct.Text))
txt3Hex.Text = Hex(Val(txt1Hex.Text) +
Val(txt2Hex.Text))
End Sub
```

3. Аналогично создать событийные процедуры вычитания, деления и умножения чисел.
4. Запустить проект. Ввести числа в поля аргументов арифметических операций и щелкнуть по кнопке арифметической операции.

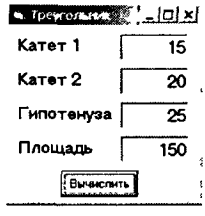


4.17. Проект «Треугольник» хранится в папке prjZ4-17.

1. Разместить на форме четыре текстовых поля (txtK1 и txtK2 — для ввода значений катетов, txtH и txtS — для вывода вычисленных значений гипотенузы и площади), кнопку cmd1 для создания событийной процедуры, реализующей вычисления, и четыре метки (lblK1, lblK2, lblH, lblS) для вывода поясняющих надписей рядом с текстовыми полями.
2. Создать событийную процедуру вычисления гипотенузы и площади:

```
Sub cmd1_Click()
txtH = Sqr(Val(txtK1.Text)^2+Val(txtK2.Text)^2)
txtS = (Val(txtK1.Text) * Val(txtK2.Text))/2
End Sub
```

3. Запустить проект на выполнение, ввести значения катетов и щелкнуть по кнопке *Вычислить*.
   В текстовых полях появятся результаты процесса вычисления гипотенузы и площади треугольника.



4.18. Проект «Усовершенствованный строковый калькулятор» хранится в папке prjZ4-18.

1. Открыть файл проекта prjVB7.vbp. Добавить кнопку cmdLeft и текстовое поле txtLeft. Создать событийную процедуру вырезания левой подстроки из строки с использованием функции Left$(строка$,bytN):

```
Sub cmdLeft_Click()
txt3.Text = Left$(txt1.Text, Val(txtLeft.Text))
End Sub
```

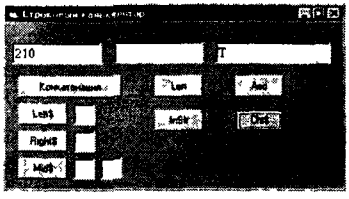
2. Добавить кнопку cmdRight и текстовое поле txtRight. Создать событийную процедуру вырезки правой подстроки из строки с использованием функции Right$(строка$,bytN) самостоятельно.
3. Добавить кнопку cmdInStr и текстовое поле txtInStr. Создать событийную процедуру определения позиции первого символа подстроки в строке с использованием функции InStr(строка$,подстрока$):

```
Sub cmdInStr_Click()
txt3.Text = InStr(txt1.Text, txt2.Text)
End Sub
```

4. Добавить кнопку cmdChr и текстовое поле txtChr. Создать событийную процедуру преобразования числового кода в символ с использованием функции Chr$(bytN):

```
Sub cmdChr_Click()
txt3.Text = Chr$(Val(txt1.Text))
End Sub
```

5. Запустить проект, ввести, например, в первое поле число и щелкнуть по кнопке *Chr$*. В третьем поле появится соответствующий данному числовому коду символ.



4.19. Проект «Регистрация» хранится в папке prjZ4-19. Программный код:

```
Dim bytB As Byte, strA, strB, strC As String
Sub cmd1_Click()
Repeat:
strA = InputBox("Введите ваше имя:",
"Регистрация")
strB = InputBox("Введите ваше отчество:",
"Регистрация")
strC = InputBox("Введите вашу фамилию:",
"Регистрация")
bytB = MsgBox("Уважаемый " + strA + " " +
strB + " " + strC + "! Изменить регистра-
ционные данные?", 36, "Конец регистрации")
If bytB = 6 Then
GoTo Repeat :
Else : End
End If
End Sub
```

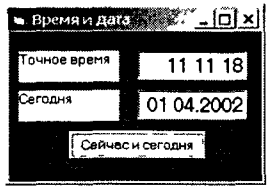
4.20. В проекте «Дата» присвоить переменной dtmB дату рождения.

4.21. Проект «Время и дата» хранится в папке prjZ4-21

1. Поместить на форму два текстовых окна txtTime (для вывода текущего времени) и txtDate (для вывода текущей даты), две метки lblTime и lblDate и кнопку cmdTD.
2. Создать событийную процедуру вывода в текстовые поля текущих времени и даты:

```
Private Sub cmdTD_Click()
txtTime.Text = Time$
txtData.Text = Date
End Sub
```

3. Запустить проект и щелкнуть по кнопке *Сейчас и сегодня*.
   В текстовых окнах появятся значения времени и даты, которые установлены на вашем компьютере.



4.22. Проект «Графический редактор» хранится в папке prjZ4-22.

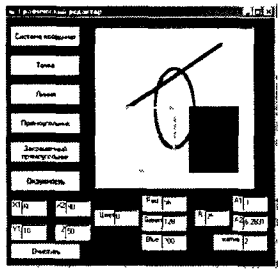
1. Поместить на форму четыре текстовых поля для ввода координат: `txtX1`, `txtY1`, `txtX2` и `txtY2`. Присвоить свойству `Text` значение 0.
2. Поместить на форму четыре текстовых поля для ввода числовых кодов цвета: `txtC`, `txtR`, `txtG` и `txtB`. Присвоить свойству `Text` значение 0.
3. Поместить на форму четыре текстовых поля для ввода параметров рисования окружности: `txtRad` (радиус), `txtAng1` и `txtAng2` (начальный и конечный углы дуги) и `txtAsp` (степень сжатия). Присвоить свойству `Text` поля `txtAng2` значение 6,28, а поля `txtAsp` — значение 1.
4. Поместить на форму 12 меток для обозначения текстовых полей.
5. Создать событийные процедуры рисования графических примитивов так, чтобы параметры считывались из текстовых окон. Предусмотреть с помощью оператора условного перехода возможность задания цвета с использованием функции `QBColor` или функции `RGB`.
   Например, событийная процедура рисования закрашенного прямоугольника запишется следующим образом:

```
Private Sub cmdLineBF_Click()
bytX1 = Val(txtX1)
bytY1 = Val(txtY1)
bytX2 = Val(txtX2)
bytY2 = Val(txtY2)
If Val(txtC.Text) > 0
Then lngC = QBColor(Val(txtC))
Else bytR = Val(txtR): bytG = Val(txtG):
bytB = Val(txtB): lngC = RGB(bytR, bytG,
bytB)
picPaint.Line (bytX1, bytY1)-(bytX2, bytY2),
lngC, BF
End Sub
```

6. Поместить на форму кнопку `txtCls` и создать событийную процедуру очистки графического поля:

```
Private Sub txtCls_Click()
picPaint.Cls
End Sub
```

7. Запустить проект. Ввести в поля числовые значения (значения углов измеряются в радианах и должны находиться в интервале от -2π до 2π).
   Щелкнуть по кнопкам, в графическом поле будут нарисованы графические примитивы.



4.23. Проект «Выбор цвета» хранится в папке prjZ4-23.

1. Поместить на форму графическое окно `picColor` для просмотра выбранного цвета и текстовое поле `txtColor` для вывода его названия.
2. Для вывода списка цветов удобно использовать управляющий элемент *Окно списка* (`ListBox`), который представляет собой текстовое поле, в котором отображается упорядоченный список значений. Поместить на форму список `lst1` и внести в него перечень цветовых констант с использованием свойства `List`.
3. Создать событийную процедуру (событие — двойной щелчок), реализующую задание цвета с помощью цветовых констант путем выбора цвета из списка. Его отображение в графическом и текстовом полях осуществить с помощью оператора **Select Case**:

```
Private Sub lst1_DblClick()
Select Case lst1.Text
Case Is = "vbBlack"
picColor.BackColor = vbBlack
txtColor = "черный"
Case Is = "vbBlue"
picColor.BackColor = vbBlue
txtColor = "синий"
Case Is = "vbGreen"
picColor.BackColor = vbGreen
txtColor = "зеленый"
Case Is = "vbCyan"
picColor.BackColor = vbCyan
txtColor = "голубой"
Case Is = "vbRed"
picColor.BackColor = vbRed
txtColor = "красный"
```

```
Case Is = "vbMagenta"
picColor.BackColor = vbMagenta
txtColor = "пурпурный"
Case Is = "vbYellow"
picColor.BackColor = vbYellow
txtColor = "желтый"
Case Is = "vbWhite"
picColor.BackColor = vbWhite
txtColor = "белый"
End Select
End Sub
```

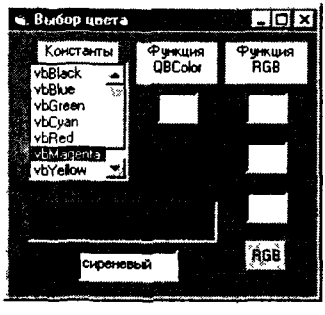
4. Поместить на форму текстовое поле txtQBC для ввода числового кода цвета. Создать событийную процедуру (событие — изменение содержания поля, например ввод нового числового кода), реализующую выбор цвета с использованием функции QBColor:

```
Private Sub txtQBC_Change()
picColor.BackColor = QBColor(Val(txtQBC.Text))
txtColor = ""
End Sub
```

5. Поместить на форму три текстовых поля txtRed, txtGreen и txtBlue для ввода числовых кодов базовых цветов и кнопку cmdRGB. Создать событийную процедуру, реализующую выбор цвета с использованием функции RGB:

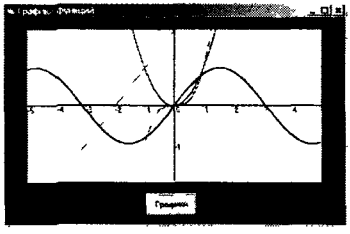
```
Private Sub cmdRGB_Click()
picColor.BackColor = RGB(Val(txtRed.Text),
Val(txtGreen.Text), Val(txtBlue.Text))
txtColor = ""
End Sub
```

6. Запустить проект.
Выбрать цвет из списка «Константы» двойным щелчком.
Выбрать цвет путем ввода числового кода в поле «Функция QBColor».
Выбрать цвет путем ввода числовых кодов в поле «Функция RGB».



4.24. Проект «Графики функций» хранится в папке prjZ4-24.

1. В проекте «Построение графика функции» в цикл построения графика ввести код:



```
picGraph.PSet (sngX,
sngX + 2), vbGreen
picGraph.PSet (sngX,
sngX ^ 2), vbRed
picGraph.PSet (sngX,
sngX ^ 3), vbMagenta
```

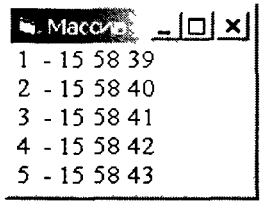
4.25. Проект «Треугольники» хранится в папке prjZ4-25.

1. В проекте «Рисование домика» заменить общую процедуру Домик на общую процедуру Треугольник:

```
Public Sub Треугольник(X1, X2, X3, Y1, Y2,
Y3 As Single)
frm1.Line (X1, Y1)-(X2, Y2)
frm1.Line (X1, Y1)-(X3, Y3)
frm1.Line (X2, Y2)-(X3, Y3)
End Sub
```

4.27. Проект «Массив» хранится в папке prjZ4-27.

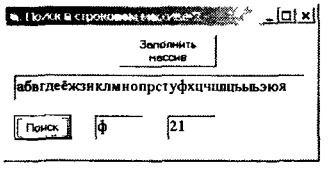
1. В проекте «Секундомер» в событийную процедуру ввести запись текущего времени в строковый массив и его распечатку на форме:




```
Dim strTime(1 To 100) As
String, bytI As Byte
Sub tmr1_Timer()
bytI = bytI + 1
strTime(bytI) = Time$
Print bytI; " - "; strTime(bytI)
End Sub
```

4.28. Проект «Поиск в строковом массиве-2» хранится в папке prjZ4-28.

1. В проекте «Поиск в строковом массиве» в событийную процедуру ввести:

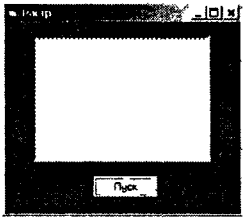


- считывание алфавита из текстового поля txt1:
  `strA(bytI) = Mid$(txt1.Text, bytI, 1);`
- ввод искомого символа в поле txt2: `strB = txt2.Text;`
- вывод номера символа в поле txt3: `txt3.Text = bytN.`

4.29. Проект «Поиск максимального элемента» хранится в папке prjZ4-29.

4.30. Проект «Сортировка по убыванию» хранится в папке prjZ4-30.

4.31. Проект «Растр» хранится в папке prjZ4-31.
Программный код:



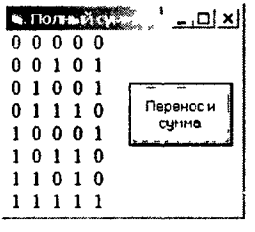

```
Dim intX As Integer, lngI As
Long
Private Sub cmdStart_Click()
'Масштаб
picAnim.Scale (0,600)-(800,0)
For intY = 600 To 0 Step -20
For intX = 0 To 800 Step 20
'Рисование растра
picAnim.PSet (intX, intY), vbBlue
'Задержка
For lngI = 1 To 50000
Next lngI
Next intX
Next intY
End Sub
```

4.32. Проект «Логические операции» хранится в папке prjZ4-32
В проекте «Таблица истинности операции логического умножения» ввести в код событийной процедуры строку:


```
frm1.Print -intA; -intB;
-(intA And intB); -(intA Or
intB); -(intA Xor intB);
-(intA Eqv intB)
```

4.33. Проект «Логические операции» хранится в папке prjZ4-33.

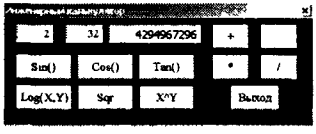
1.
```
Dim intA, intB, intP0, intP, intS As Integer
Sub cmd1_Click()
For intA = 0 To -1 Step -1
For intB = 0 To -1 Step -1
For intP0 = 0 To -1 Step -1
intP = (intA And intB) Or
(intA And intP0) Or (intB
And intP0)
intS = (intA Or intB Or
intP0) And Not intP Or (intA
And intB And intP0)
frm1.Print -intA; -intB; -intP0; -intP; -intS
Next intP0
Next intB
Next intA
End Sub
```



4.34. Макрос «Форма» хранится в файле Редактирование doc в каталоге \textbook\VBA\ .

4.35. Проект «Инженерный калькулятор» хранится в файле Макросы и проекты xls в каталоге \textbook\VBA\ .

1. Усовершенствовать проект «Калькулятор»: поместить на форму дополнительно шесть кнопок для создания событийных процедур вычисления тригонометрических функций sin(x), cos(x), tan(x), логарифма log(x,y), квадратного корня sqr(x) и возведения числа в степень x^y.
2. Ввести в ячейки электронной таблицы D1, D2, D3, D4, D5 и D6 формулы для вычисления функций.
3. Создать событийные процедуры. Событийная процедура возведения числа в степень:




```
Private Sub cmdSt_Click()
Cells(1, 1) =
Val(txt1.Text)
Cells(1, 2) = Val(txt2.Text)
txt3.Text = Cells(6, 4)
End Sub
```

Ответы к главе 5. Моделирование и формализация

5.1. http //www college ru.

5.5. Сайт о династии Романовых хранится в каталоге \textbook\dinasty\.

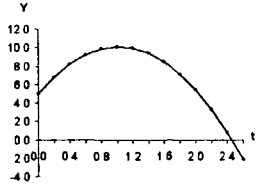
5.8. Модель в электронных таблицах хранится в файле Model xls в каталоге \textbook\Excel\, модель на языке Visual Basic хранится в каталоге \textbook\VB\prjZ5-8\.

1. *Качественная модель.* Если начальная скорость бросания тела v_0 существенно меньше первой космической скорости и высота бросания h существенно меньше радиуса Земли, можно использовать модель, рассмотренную ранее.
2. *Формализованная модель.* Движение по вертикали — равноускоренное, поэтому изменение координаты y в зависимости от времени описывается с помощью формулы:
$$y = h_0 + v_0 \cdot t - g\,t^2/2.$$
3. *Компьютерная модель в электронных таблицах*
Ввести начальные значения высоты (h_0) и скорости (v_0) в ячейки B1 и B2 соответственно.

Создать таблицу значений зависимости координаты от времени. В ячейки A5:A18 ввести значения моментов времени t (в секундах) от 0 до 2,6 с шагом 0,2 с. В ячейку B5 ввести формулу уравнения движения =$B$2+$B$3*A5–4,9*A5^2, в которой использованы абсолютные ссылки на ячейки, содержащие начальные условия.
Скопировать формулу в ячейки B6:B18.

	A	B
1	h_0=	5
2	v_0=	10
3		
4	t	$y=h_0+v_0*t-4,9*t^2$
5	0,0	5,0
6	0,2	6,8
7	0,4	8,2
8	0,6	9 2
9	0,8	9,9
10	1,0	10,1
11	1,2	9,9
12	1,4	9,4
13	1,6	8,5
14	1,8	7,1
15	2,0	5,4
16	2,2	3,3
17	2,4	0 8
18	2,6	-2,1

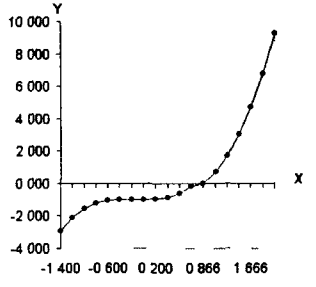
4. *Исследование модели.*
Построить диаграмму типа *График*.
Точка пересечения графика с осью t соответствует времени падения тела (примерно 2,4 с).



Методом «Подбор параметра» найти более точное значение времени падения с использованием ячейки B17. В ячейке A17 появится значение времени — 2,46 с.

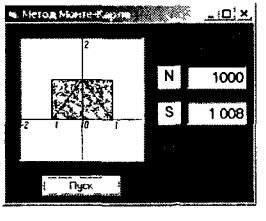
5.9. Модель хранится в файле Model.xls в каталоге \textbook\Excel\.

1. Представить заданное уравнение в табличной форме.
Для грубо приближенного определения корня построить диаграмму типа *График*.
Для определения корня с заданной точностью использовать метод «Подбор параметра».
При представлении чисел в ячейках таблицы с точностью три знака после запятой `x = 0,866`.



5.10. Проект хранится в каталоге \textbook\VB\prjZ5-10\.

1. Внести изменения в код событийной процедуры проекта «Метод Монте-Карло».
Условие попадания точек внутрь треугольника:
`dblY >= 0 And Abs(dblX) + Abs(dblY) <= 1`



5.14. Геоинформационная модель хранится в файле mapstats.xls в каталоге \textbook\Excel\.

5.15. Проект хранится в каталоге \textbook\VB\prjZ5-15\.

1. Наборы параметров, при которых выполняются условия ограничений, можно найти перебором вариантов с помощью вложенных циклов:

```
Private Sub cmd1_Click()
For X1 = 0 To 100
For X2 = 0 To 100
For X3 = 0 To 100
If 10 * X1 + 3 * X2 + 8 * X3 = 500 And 3
* X1 + 6 * X2 + 4 * X3 = 300 Then _
Print "Параметры: "; X1; X2; X3: Print
"Целевая функция:   "; X1 + X2 + X3
Next X3
Next X2
Next X1
End Sub
```

2. Если таких наборов много, то поиск минимального или максимального значения целевой функции можно осуществить стандартным методом поиска элемента в массиве.

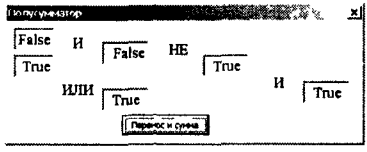
5.17. Проект хранится в файле Model.xls в каталоге \textbook\Excel\.

1. На листе «Полусумматор» предусмотреть ввод логических значений аргументов в ячейки B1 и B2.
   В ячейку B3 ввести формулу =И(B1;B2).
   В ячейку B4 ввести формулу =НЕ(B3).
   В ячейку B5 ввести формулу =ИЛИ(B1;B2).
   В ячейку B6 ввести формулу =И(B5;B4).

2. Добавить форму и поместить на нее четыре метки и шесть текстовых полей. Создать событийную процедуру:

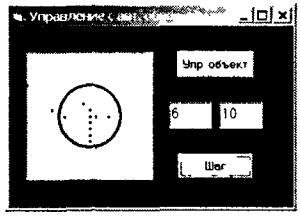
```
Private Sub cmd1_Click()
Cells(1, 2) = txtA.Text
Cells(2, 2) = txtB.Text
txtP.Text = Cells(3, 2)
txtNot.Text = Cells(4, 2)
txtOr.Text = Cells(5, 2)
txtS.Text = Cells(6, 2)
End Sub
```

3. Запустить проект, ввести значения аргументов и щелкнуть по кнопке *Перенос и сумма*.



5.18. Проект хранится в каталоге \textbook\VB\prjZ5-18\ .

1. Модернизировать модель замкнутой системы управления: в событийную процедуру добавить код автоматической корректировки координат точки:




```
Select Case bytX2 - bytX1
Case Is > 0
bytX1 = bytX1 + 1
Case Is < 0
bytX1 = bytX1 - 1
Case Is = 0
bytX1 = bytX1
End Select
```

Ответы к главе 10. Технология обработки числовых данных

10.3, 10.4, 10.5, 10.6, 10.7, 10.10, 10.11, 10.12, 10.13 Готовые задания хранятся в файле calc.xls в каталоге \textbook\Excel\.

Приложения

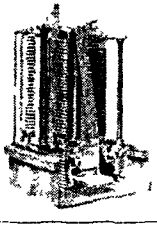
Словарь компьютерных терминов

ActiveX	Программные компоненты, загружаемые с удаленного сервера и выполняющиеся на локальном компьютере. С их помощью можно демонстрировать различную информацию, включающую видео и звук, без запуска дополнительных программ
CGI-скрипт	Программы, написанные на любом языке программирования, обрабатывающие данные, полученные из форм на Web-страницах с использованием Common Gateway Interface (CGI)
CP1251	CP1251 означает Code Page1251 — «кодовая страница 1251». Это наиболее распространенная кодировка, используемая в операционной системе Microsoft Windows и ее приложениях
CP866	CP866 означает Code Page866 — «кодовая страница 866», «альтернативная» кодировка, используемая в операционной системе MS-DOS и ее приложениях
DIMM	Dual In-line Memory Module, модули памяти, которые имеют 168 контактов, а их информационная емкость составляет 16, 32, 64, 128 и 256 Мбайт
DDR	Double Data Rate, современные модули памяти с частотой до 200 МГц, имеющие 184 контакта и информационную емкость 64, 128, 256 и 512 Мбайт
dpi	Dots per Inch, количество точек на дюйм – мера разрешающей способности мониторов, принтеров и сканеров
ISO	Кодировка ISO 8859-5 была разработана Международной организацией по стандартизации (International Standards Organization), однако сегодня практически не используется
Java-апплет	Программные модули на языке Java, которые могут запускаться под управлением любой операционной системы после загрузки с Web-сервера Интернета на локальный компьютер
Login	Идентификатор пользователя для входа в систему или подключения к Интернету
Mac	Кодировка для операционной системы Mac OS, которую разработала фирма Apple для компьютеров Macintosh
Password	Пароль — уникальная последовательность символов, известная только конкретному пользователю, которую необходимо ввести для доступа к системе или подключения к Интернету

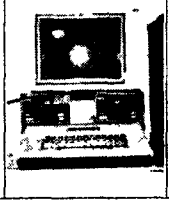
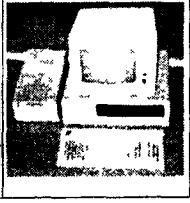
POP3	Post Office Protocol 3 – протокол получения сообщений с почтового сервера провайдера
PPP	Point to Point Protocol (Протокол точка–точка), который используется для удаленного доступа в Интернет по коммутируемым телефонным линиям
RIMM	Rambus In-line Memory Module, современные модули памяти с частотой до 800 МГц, имеющие 184 контакта и информационную емкость 64, 128, 256 и 512 Мбайт
SCSI	Small Computer System Interface (интерфейс малых вычислительных систем) используется для подключения к компьютеру дополнительных устройств (винчестеров, сканеров, CD-ROM-дисководов и др.)
SIMM	Single In-line Memory Module, устаревшие модули памяти, могут иметь различное количество контактов (30 или 72), а их информационная емкость составляет 1, 4, 8, 16 или 32 Мбайт
SMTP	Simple Mail Transfer Protocol – протокол доставки сообщений на почтовый сервер провайдера
Архиватор	Программа, выполняющая сжатие (архивирование) файлов для более компактного хранения во внешней памяти и восстановление (разархивирование) сжатых файлов в первоначальном состоянии
Атрибут	Признак или свойство, характеризующее объект
БИС	Большая интегральная схема – электронная схема, изготовленная по миллимикронной технологии внутри полупроводникового кристалла и содержащая десятки миллионов электронных элементов, способных выполнять логические операции или хранить информацию
Витая пара	Содержит две или более пары проводов, скрученных один с другим по всей длине кабеля. Скручивание позволяет повысить помехоустойчивость кабеля. Применяется в локальных сетях
Гипертекст	В широком смысле – это компьютерное представление данных, в котором могут быть заданы любые связи между объектами различных типов (фрагментами документов, управляющими элементами и др.)
Граф	Позволяет визуализировать связи между объектами в сложноорганизованных системах, объекты представляются с помощью множества вершин (точек), а связи с помощью множества ребер (дуг), соединяющих некоторые пары вершин
Графический интерфейс	Система окон, меню, диалоговых панелей и элементов управления, которая обеспечивает интерактивный диалог пользователя с операционной системой, системой программирования или приложением
Дерево	Иерархический граф, который используется для визуализации иерархических информационных моделей
Дистрибутив	Лицензионный программный продукт, который может быть установлен на компьютер пользователя

Доменная система имен	Иерархическая система уникальных имен компьютеров, подключенных к Интернету
Коаксиальный кабель	Состоит из центрального проводника (сплошного или многожильного), покрытого слоем полимерного изолятора, поверх которого расположен другой проводник (экран). Применяется в локальных сетях
КОИ8	Код Обмена Информацией 8-битный. Он хронологически был одним из первых стандартов кодирования русских букв. Эта кодировка применяется до сих пор, в основном на компьютерах с операционной системой UNIX
Кэш-память	Быстродействующая память, предназначенная для временного хранения данных. Аппаратно кэш-память реализуется в процессорах и жестких дисках, а программно в операционной системе, браузерах и других приложениях
Оптоволоконный кабель	Состоит из тонкого стеклянного цилиндра, покрытого оболочкой с другим коэффициентом преломления. Применяется в глобальных сетях
Пиксель	Минимальный элемент изображения (точка), которому можно задать цвет и яркость. Пиксель является элементом растра
Портал	Стартовый сайт, предлагающий пользователю доступ к тематически подобранным информационным ресурсам в форме каталогов, новостей и обзоров, а также информационные сервисы: почту, чаты, форумы и поисковые системы
Равновероятные события	События называются равновероятными, если при возрастающем количестве испытаний (10, 100, 1000 и так далее) количество реализаций событий будут все более сближаться
Разрешающая способность	Характеристика качества изображения. Разрешающая способность экрана монитора определяется количеством точек по горизонтали и вертикали
Растр	Двумерный массив точек, упорядоченных в строки и столбцы, который используется для создания изображения на экране монитора
Синтаксис	В естественных языках — совокупность правил построения предложений. В языках программирования — правила записи операторов, методов и так далее
Слот	Разъем на материнской плате компьютера, в который устанавливаются платы контроллеров устройств (например, видеоадаптер) и дополнительные устройства (например, модем)

История развития вычислительной техники

	Первое автоматическое вычислительное устройство	Первая ЭВМ	Первая отечественная ЭВМ
Год выпуска, производитель	1832, Чарльз Бэббидж, Великобритания	1946, США	1950, СССР
Название устройства	Аналитическая машина	ENIAC (Electronic Numeral Integrator and Computer)	МЭСМ (Малая Электронная Счетная Машина)
Элементная база	Механические устройства	Электронные лампы (18900 шт.)	Электронные лампы (6000 шт.)
Быстродействие (кол-во операций в секунду)	Мельница (арифметическое устройство)	5 тысяч операций сложения в секунду	5 тысяч операций сложения в секунду
Разрядность		30 бит	16 бит
Оперативная память	Склад (устройство для хранения чисел)	600 бит	1800 бит
Долговременная память	Перфокарты	4 100 магнитных элементов памяти	Магнитный барабан на 5 000 чисел
Фото			

История развития персональных компьютеров

	Первый персональный компьютер	Персональный компьютер IBM	Первый отечественный персональный компьютер	Современный персональный компьютер
Год выпуска, производитель	1976, фирма Apple	1983, корпорация IBM	1985, СССР	2002
Тип компьютера	Apple II	IBM PC/XT	Агат	Платформа Windows
Процессор, частота	Motorola 6502, 1 МГц	Intel 8086 10 МГц	1 МГц	Intel Pentium 4, 2 ГГц
Разрядность процессора	8 бит	16 бит	8 бит	64 бита
Оперативная память	48 Кбайт	640 Кбайт	48 Кбайт	128 Мбайт
Долговременная память	НГМД, 1400 Кбайт	НЖМД, 10 Мбайт, НГМД, 360 Кбайт	НГМД, 840 Кбайт	НЖМД, 50 Гбайт, DVD-ROM
Фото				

Основные тэги HTML

Назначение	Формат	Значения аргументов
Структура Web-страницы		
Начало и конец страницы	<HTML></HTML>	
Описание страницы, в том числе ее имя	<HEAD></HEAD>	
Имя страницы	<TITLE></TITLE>	
Содержание страницы	<BODY></BODY>	
Форматирование текста		
Заголовок (уровни от 1 до 6)	<H?></H?>	
Заголовок с выравниванием	<H? ALIGN="*"> </H?>	left center right
Абзац	<P></P>	
Абзац с выравниванием	<P ALIGN="*"></P>	left center right
Перевод строки	 	
Горизонтальный разделитель	<HR>	
Выравнивание по центру	<CENTER></CENTER>	
Адрес автора	<ADDRESS></ADDRESS>	
Форматирование шрифта		
Жирный	<B></B>	
Курсив	<I></I>	
Верхний индекс		
Нижний индекс		
Размер шрифта (от 1 до 7)	<FONT SIZE=?></FONT>	
Цвет шрифта (задается названием цвета или его 16-ричным кодом)	<FONT COLOR="#RRGGBB"> </FONT>	red blue #FFFFFF и др.
Гарнитура шрифта	<FONT FACE="*"> </FONT>	Arial TimesET и др.
Вставка изображений		
Вставка изображения	<IMG SRC="URL">	
Выравнивание текста около изображения	<IMG SRC="URL" ALIGN="*">	top bottom middle left right
Вывод текста вместо изображения	<IMG SRC="URL" ALT="*">	текст
Цвет фона, текста и ссылок		
Фоновое изображение	<BODY BACKGROUND="URL">	
Цвет фона	<BODY BGCOLOR="#RRGGBB">	red blue #FFFFFF и др.

Назначение	Формат	Значения аргументов
Цвет текста	<BODY TEXT="#RRGGBB">	red blue #FFFFFF и др.
Цвет ссылки	<BODY LINK="#RRGGBB">	
Цвет пройденной ссылки	<BODY VLINK="#RRGGBB">	
Цвет активной ссылки	<BODY ALINK="#RRGGBB">	
Вставка гиперссылок		
Ссылка на другую страницу	<A HREF="URL">указатель ссылки</A>	
Ссылка на закладку в другом документе	<A HREF="URL#*">указатель ссылки</A>	
Ссылка на закладку в том же документе	<A HREF="#*">указатель ссылки</A>	
Определение закладки	<A NAME="*"></A>	
Списки		
Ненумерованный	<UL><LI></UL>	
Тип метки	<UL TYPE="*">	disk circle square
Нумерованный	<OL><LI></OL>	
Тип нумерации	<OL TYPE="*">	A, a, I, i, 1
Первый номер списка	<OL START=?>	1,2,...
Список определений <DT>термин <DD>определение	<DL> <DT> <DD> </DL>	
Меню	<MENU><LI></MENU>	
Каталог	<DIR><LI></DIR>	
Формы		
Форма	<FORM> </FORM>	
Текстовое поле NAME="name"	<INPUT TYPE="text" NAME="name" SIZE=?>	1,2,3 ...
Группа переключателей NAME="group"	<INPUT TYPE="radio" NAME="group" VALUE= "*">	rad1 rad2 rad3
Группа флажков NAME="group"	<INPUT TYPE="checkbox" NAME="group" VALUE="*">	ch1 ch2 ch3
Раскрывающийся список NAME="list"	<SELECT NAME="list"> <OPTION>Первый <OPTION>Второй </SELECT>	
Текстовая область NAME="resume"	<TEXTAREA> NAME="resume" ROWS=? COLS=?> </TEXTAREA>	1,2,3 ...
Кнопка *Отправить*	<INPUT TYPE="submit" VALUE="Отправить">	
Кнопка *Очистить*	<INPUT TYPE="reset" VALUE="Очистить">	

Учебное издание

Угринович Николай Дмитриевич
Информатика и информационные технологии

Художник *Н. Лозинская*
Компьютерная верстка *В. Носенко*

Подписано в печать 03.07.03. Формат 60x90 $^1/_{16}$.
Бумага офсетная. Гарнитура Школьная. Печать офсетная.
Усл. печ. л. 32,0. Тираж 50000 экз. Заказ 2645

Издательство «БИНОМ. Лаборатория знаний»
Адрес для переписки: 119071, Москва, а/я 32
Телефон: (095)955-0398. E-mail: lbz@aha.ru

Лицензия на издательскую деятельность № 06331
от 26 ноября 2001 г.

Отпечатано с готовых диапозитивов
в полиграфической фирме «Полиграфист».
160001, г. Вологда, ул. Челюскинцев, 3