Л. Л. Босова, А. Ю. Босова

ИНФОРМАТИКА

11 класс

Базовый уровень

Учебник

Рекомендовано
Министерством образования и науки Российской Федерации
к использованию при реализации имеющих государственную
аккредитацию образовательных программ начального общего,
основного общего, среднего общего образования

Москва
БИНОМ. Лаборатория знаний
2016

УДК 004.9
ББК 32.97
Б85

Босова Л. Л.
Б85 Информатика. 11 класс. Базовый уровень : учебник / Л. Л. Босова, А. Ю. Босова. — М. : БИНОМ. Лаборатория знаний, 2016. — 256 с. : ил.

ISBN 978-5-9963-3142-0

Учебник предназначен для изучения информатики на базовом уровне в 11 классе общеобразовательных организаций. Включает вопросы, касающиеся информационных технологий, алгоритмизации и программирования, информационного моделирования, компьютерных телекоммуникаций, социальной информатики, информационной безопасности.

Содержание учебника опирается на материал, изученный в 7–9 классах основной школы. Учебник позволяет каждому обучающемуся овладеть ключевыми понятиями и закономерностями, на которых строится современная информатика, научиться выполнять задания ЕГЭ базового уровня сложности. Мотивированный обучающийся научится выполнять ряд заданий ЕГЭ повышенного уровня сложности.

Соответствует федеральному государственному образовательному стандарту среднего общего образования и примерной основной образовательной программе среднего общего образования.

УДК 004.9
ББК 32.97

Учебное издание

Босова Людмила Леонидовна
Босова Анна Юрьевна

ИНФОРМАТИКА

11 класс
Базовый уровень
Учебник

Редактор *Е. В. Баклашова*
Художник *Н. А. Новак*
Технический редактор *Е. В. Денюкова*
Корректор *Е. Н. Клитина*
Компьютерная верстка: *Л. В. Катуркина*

Подписано в печать 22.07.16. Формат 70x100/16. Усл. печ. л. 20,8.
Тираж 15 экз. Заказ

ООО «БИНОМ. Лаборатория знаний»
127473, Москва, ул. Краснопролетарская, д. 16, стр. 1,
тел. (495)181-53-44, e-mail: binom@Lbz.ru
http://www.Lbz.ru, http://metodist.Lbz.ru

ISBN 978-5-9963-3142-0

Введение

Уважаемые старшеклассники!

Вы держите в руках учебник, продолжающий начатое в 10 классе изучение курса информатики на базовом уровне. Напомним, что задачи изучения любого школьного предмета на базовом уровне направлены на формирование у вас:

- понимания предмета, ключевых вопросов и основных составляющих элементов изучаемой предметной области;
- умения решать основные практические задачи, характерных для использования методов и инструментария данной предметной области;
- осознания рамок изучаемой предметной области, ограниченности методов и инструментов, типичных связей с некоторыми другими областями знания.

Работая с этим учебником, вы продолжите изучение вопросов, касающихся информационных технологий представления, хранения, поиска, обработки и анализа данных. Наличие представлений о соответствующих компьютерных инструментах и умение работать с ними — необходимое условие вашей дальнейшей учебной и профессиональной деятельности, вне зависимости от того, какие планы вы строите на будущее.

Важным качеством человека, живущего в современном высокотехнологичном обществе, является уровень развития его алгоритмического мышления. Продвигаясь в этом направлении, вы, прежде всего, вспомните и закрепите типовые приёмы написания программ для решения стандартных задач с использованием основных алгоритмических конструкций; расширите свои знания в области структур данных; познакомитесь с некоторыми методами исследования алгоритмов.

Значительное место в учебнике уделено информационному моделированию и формированию представлений о разнообразных компьютерных моделях.

Также мы считаем важным обсудить с вами разнообразные эффекты и проблемы, связанные с информатизацией и массовой коммуникацией современного общества.

В конце каждого параграфа размещены вопросы и задания. Отвечая на вопросы, вы сможете проконтролировать, хорошо ли

усвоены основные понятия и идеи информатики и информационных технологий. Выполняя задания и решая задачи, вы сможете проверить, насколько хорошо умеете применять полученные знания на практике, в том числе в новых условиях. Более сложные задания отмечены знаком *. Мы считаем, что их выполнение по силам каждому ученику, но настоятельно рекомендуем выполнять эти задания тем, кто готовится к сдаче единого государственного экзамена (ЕГЭ) по информатике.

В работе с учебником вам помогут навигационные значки:

 — важное утверждение или определение;

 — интересная информация;

 — пример решения задачи;

 — ссылка на ресурс в Интернете;

 — дополнительный материал, содержащийся в авторской мастерской Л. Л. Босовой (**http://metodist.Lbz.ru**);

 — вопросы в тексте параграфа, вопросы и задания для самоконтроля;

 — задания для подготовки к итоговой аттестации;

 — домашний проект или исследование;

 — задания для выполнения на компьютере;

 — групповая работа;

 — межпредметные связи.

Желаем успехов в изучении информатики!

Глава 1
ОБРАБОТКА ИНФОРМАЦИИ В ЭЛЕКТРОННЫХ ТАБЛИЦАХ

Прикладные программы, предназначенные для работы с данными, представленными в таблицах, называются табличными процессорами (ТП) или просто электронными таблицами (ЭТ).

Первый табличный процессор был создан в 1979 году и предназначался для автоматизации рутинных вычислительных процедур. Современные электронные таблицы применяются не только для выполнения расчётов. Они позволяют проводить численные эксперименты с математическими моделями, их часто используют как простую базу данных, в них можно создавать разнообразные красиво оформленные документы с произвольной информацией — рекламные листовки с прайс-листами, каталоги, планы и графики работ, расписания и многое другое.

Вопросы, связанные с обработкой информации в электронных таблицах, занимают важное место в повседневной профессиональной деятельности многих специалистов, связанных с бухгалтерским и банковским учётом, планированием распределения ресурсов, проектно-сметными работами, инженерно-техническими расчётами, обработкой больших массивов информации, исследованием динамических процессов и др. Электронные таблицы полезны и в быту: для учёта семейных доходов и расходов, при расчётах взносов за коммунальные услуги и кредиты, при заполнении налоговой декларации и т. д. Вы можете использовать табличные процессоры в своей учебной деятельности: для решения систем уравнений и построения графиков функций по математике, для обработки результатов экспериментов по химии и физике, для визуализации экономических и статистических данных на занятиях по экономической географии.

§ 1
Табличный процессор. Основные сведения

1.1. Объекты табличного процессора и их свойства

Наиболее распространёнными табличными процессорами являются Microsoft Excel и OpenOffice Calc.

Весь материал, рассматриваемый в этой главе, носит универсальный характер, т. е. справедлив по отношению к любому табличному процессору. Мы будем иллюстрировать его примерами, выполненными в среде Microsoft Excel 2010.

Все примеры, имеющиеся в учебнике, желательно повторять в среде имеющегося в вашем распоряжении табличного процессора. Если вы располагаете версией табличного процессора Microsoft Excel, отличной от Microsoft Excel 2010, или же используете другое семейство табличных процессоров, ищите недостающую информацию в справочной системе самого табличного процессора или в сети Интернет.

После запуска программы Microsoft Excel на экране открываются два окна: окно табличного процессора и окно созданного в нём документа. Окно табличного процессора имеет типовую структуру (рис. 1.1).

Документ, создаваемый в табличном процессоре, называется **рабочей книгой** и по умолчанию получает имя **Книга1**. Вновь созданная в Microsoft Excel рабочая книга состоит из трёх **листов** с именами **Лист1**, **Лист2** и **Лист3**. Имена листов указываются на ярлычках. Пользователь может переименовать листы по своему усмотрению, добавить к книге новые листы или удалить ненужные. На листах могут быть размещены электронные таблицы (вычислительные таблицы, создаваемые с помощью ТП), диаграммы, графики, графические изображения и другие объекты. Перейти к просмотру любого листа книги можно выбором его ярлычка, а для просмотра содержимого той части листа, которая не отображается в окне, можно использовать полосы прокрутки.

Если в Microsoft Excel окно рабочей книги открыто в полноэкранном режиме, то имя книги отображается в строке заголовка окна табличного процессора, а кнопки управления окном рабочей книги — под кнопками управления окном табличного процессора.

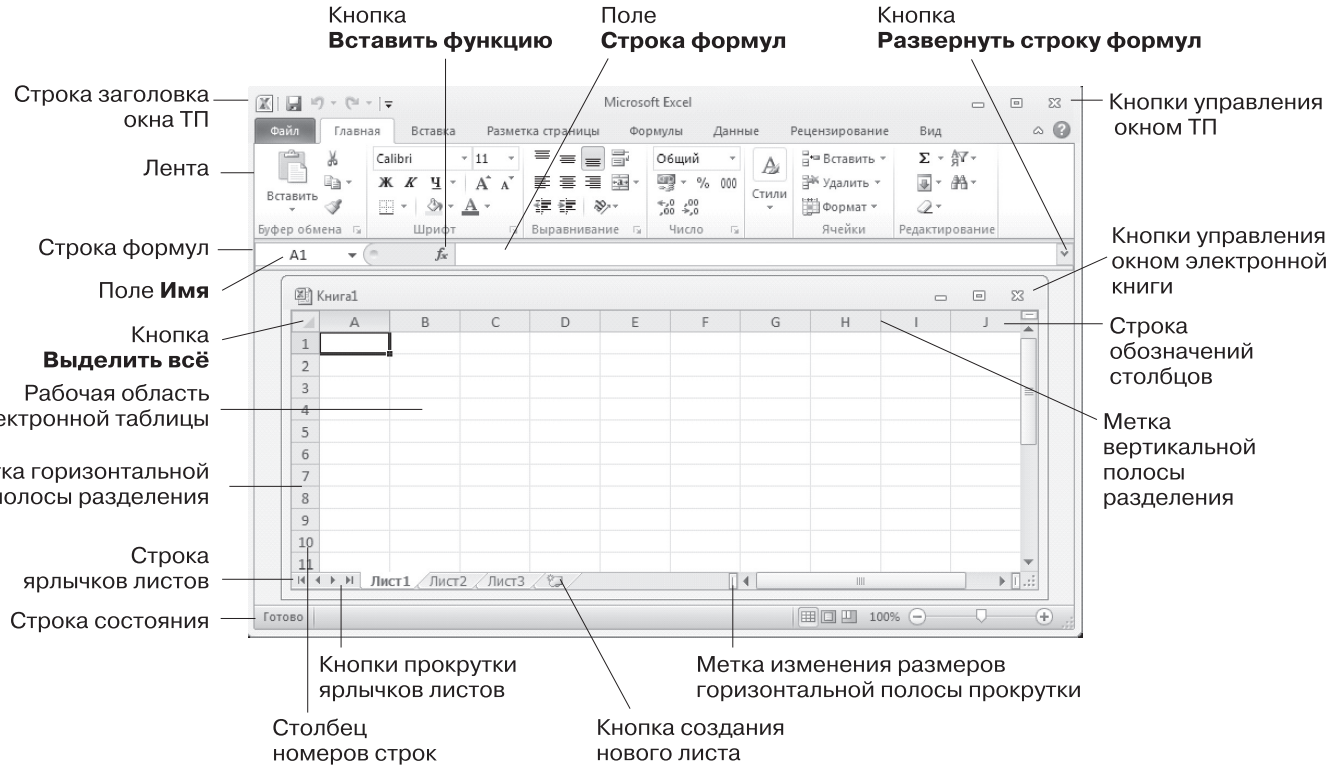


Рис. 1.1. Основные элементы интерфейса Microsoft Excel

В окне рабочей книги отображается содержимое текущего листа. Рабочая область листа с электронной таблицей **столбцами** и **строками** разбита на **ячейки**. Столбцы обозначены буквами латинского алфавита, строки пронумерованы. Адрес ячейки образуется из имени столбца и номера строки, на пересечении которых она находится (например: A1, AB12).

Ячейка — это наименьшая структурная единица электронной таблицы, которая образуется на пересечении столбца и строки.

Иногда для обозначения ячеек используется система адресов RC, название которой произошло от английских слов «Row» (строка) и «Column» (столбец). В этой системе ячейка C3 будет иметь адрес R3C3.

Выясните, как осуществляется переход к системе адресов RC в табличном процессоре, имеющемся в вашем распоряжении. Посмотрите, каким в этой системе будет адрес ячейки AB12.

Ячейка может иметь имя, представляющее собой последовательность символов, отличную от адресов ячеек, не содержащую пробелов и не начинающуюся с цифры. Чтобы присвоить ячейке имя, достаточно её выделить, ввести желаемую последовательность символов в поле **Имя** и нажать Enter.

Две и более ячейки листа электронной таблицы образуют диапазон ячеек. В диапазон ячеек могут входить как смежные, так и несмежные ячейки. Прямоугольный диапазон из смежных ячеек называется связным диапазоном. При задании адреса связного диапазона указывают его начальную и конечную ячейки — ячейки левого верхнего и правого нижнего углов (например, A1:A10, B2:C2, B2:D10). Чтобы указать адрес несвязного диапазона ячеек, надо через точку с запятой указать адреса его связных частей.

Строка и столбец также являются диапазонами ячеек. Например, адрес 5:5 — адрес диапазона ячеек, в который входят все ячейки пятой строки, а адрес C:C — адрес диапазона ячеек, в который входят все ячейки столбца C. Соответственно 10:12 — это адрес диапазона ячеек, который включает все ячейки строк с номерами 10, 11 и 12, а C:E — адрес диапазона ячеек, в который входят все ячейки столбцов C, D и E.

В таблице 1.1 приведены объекты табличного процессора, а также их основные свойства, которые далее будут рассмотрены более подробно.

Таблица 1.1

Объекты табличного процессора и их свойства

Объект	Свойства объекта
Рабочая книга	Имя, количество листов
Лист	Имя, количество размещённых объектов и их вид, наличие защиты
Электронная таблица	Общее количество строк и столбцов; количество строк и столбцов, содержащих данные
Столбец	Имя, ширина, количество заполненных данными ячеек
Строка	Номер, высота, количество заполненных данными ячеек
Ячейка	Адрес, имя, содержимое, тип данных, формат отображения данных, примечание, границы, заливка
Диапазон ячеек	Адрес, количество ячеек
Диаграмма	Тип, вид, название, размер области диаграммы, цветовая гамма

Операции создания новой книги, открытия книги, созданной ранее и сохраненной на внешнем носителе, сохранения книги в файле выполняются в табличных процессорах так же, как и аналогичные операции в текстовых процессорах. Стандартным типом файла в Microsoft Excel является Книга Excel, а стандартным расширением имени файла — расширение xlsx.

Какой тип файлов является стандартным при сохранении данных в табличном процессоре, имеющемся в вашем распоряжении? Какое расширение имени файла ему соответствует?

Выполнение команды **Файл** → **Печать** → **Печать** приводит к печати части текущего листа книги, которая заполнена данными, а также других объектов (например, диаграмм), расположенных на этом листе. Сетка, разделяющая лист электронной таблицы на ячейки, по умолчанию не печатается. Если данными заполнена область листа, которая физически не умещается на одной странице (одном листе бумаги), то табличный процессор автоматически распределяет её на несколько страниц.

Для установки определённых значений параметров печати можно использовать управление настройками окна **Печать** или группу инструментов **Параметры страницы** на вкладке **Разметка страницы**.

В чём состоит принципиальное отличие операции печати книги в среде табличного процессора от операции печати документа в текстовом процессоре?

1.2. Некоторые приёмы ввода и редактирования данных

Вся информация заносится пользователем в ячейки ЭТ. Для того чтобы вводить или редактировать данные в той или иной ячейке ЭТ, в неё следует поместить табличный курсор, т. е. сделать ячейку активной.

Обратите внимание на то, какую форму принимает указатель мыши, перемещаемый над рабочей областью листа в ЭТ.

Вспомните, как выглядит табличный курсор и как его можно поместить в ту или иную ячейку.

Перед началом ввода текстовый курсор в ячейке отсутствует и появляется после ввода первого символа. Во время ввода данных надпись «Готово» в строке состояния изменяется на «Ввод», данные отображаются как в текущей ячейке, так и в поле **Строка формул**.

Адрес активной ячейки и вводимые в неё данные отражаются в строке формул. В поле **Строка формул** можно редактировать информацию, хранящуюся в активной ячейке. При этом в строке формул появляются кнопки **Ввод** (✓) и **Отмена** (✗).

Для удаления данных из ячейки нужно выделить её и нажать клавишу Delete. Таким же способом можно очистить содержимое ячеек выделенного диапазона.

Содержимым ячейки может быть число, текст или формула. Возможности работы с данными определяются их типом. ЭТ работают с данными следующих типов:

- числовые значения (например, 143; 51,1; 4/5; 1,23E+02);
- дата и время суток (например, Май 1945; 20.12.2012; 15:00; 3:00 PM);
- формулы (например, =(A1+B1)/2 или =СУММ(A1:A5));
- текстовые значения (например, Время, Стоимость, Всего, Фамилия);

- примечания;
- гиперссылки;
- различные графические изображения.

Табличный процессор самостоятельно пытается распознать тип вводимых данных. По умолчанию числа выравниваются по правому краю ячейки.

Дробную часть числа от целой отделяют запятой или точкой, в зависимости от установок операционной системы. В русскоязычных версиях Windows в качестве разделителя целой и дробной частей числа по умолчанию используется запятая, а при употреблении точки число интерпретируется как дата.

Посмотрите, как будут отображены в ячейках ЭТ данные 17.05 и 05.17. Как вы можете это объяснить?

Ввод формулы начинается со знака равенства, который указывает табличному процессору на необходимость выполнения вычислений в соответствии со следующим за ним выражением.

Формула начинается со знака «=» и может содержать скобки, числа, тексты, ссылки на ячейки, знаки операций и функции.

При вводе формул необходимо соблюдать следующие правила:

- для обозначения арифметических действий используются операторы: «+» — для сложения, «–» — для вычитания, «*» — для умножения, «/» — для деления;
- для обозначения действия возведения в степень используется оператор «^»; например 5^3 будет записано как 5^3;
- для обозначения действия нахождение процентов используется оператор %; например формула нахождения 25% от числа 240 будет выглядеть так: =240*25%;
- нельзя опускать оператор умножения;
- порядок выполнения (приоритет) операций совпадает с порядком (приоритетом), принятым в математике;
- для изменения порядка выполнения действий используют круглые скобки;
- формула должна быть записана линейно, т. е. в виде строки символов.

Как правило, в формулах используются не сами исходные данные, а **ссылки** на ячейки, в которых эти данные находятся.

Ссылка на ячейку состоит из адреса ячейки. Ячейка, в которую вводится формула, и ячейка, ссылка на которую используется в формуле, могут находиться на разных листах и даже в

разных книгах. В таких случаях в ссылках к адресу ячейки добавляется указание на её месторасположение. Например, Лист2!C4 является ссылкой на ячейку C4 листа Лист2.

При изменении данных в каких-либо ячейках происходит автоматический пересчёт значений всех формул, содержащих ссылки на эти ячейки.

Возможность **автоматического пересчёта** формул при изменении исходных данных — одна из ключевых идей электронных таблиц. Благодаря этому электронные таблицы называют динамическими.

При создании формулы входящие в неё ссылки можно ввести, набрав адреса ячеек на клавиатуре. Однако лучше их вводить, помещая табличный курсор с помощью мыши или клавиатуры в соответствующую ячейку ЭТ. В этом случае вы точно не спутаете похожие по начертанию русские и латинские буквы и сможете контролировать правильность ввода формул, обращая внимание на выделенные цветом ссылки в формуле и границы соответствующих им ячеек.

По умолчанию в ячейках с формулами отображаются не сами формулы, а результаты их вычислений. При этом сама формула отображается в строке формул. Это так называемый режим отображения значений.

Выясните, как можно установить режим отображения формул в ЭТ, имеющихся в вашем распоряжении.

При использовании формул в ячейках электронной таблицы могут появляться сообщения об ошибках (табл. 1.2)

Таблица 1.2

Некоторые сообщения об ошибках

Сообщение	Причина ошибки
#####	Столбец недостаточно широкий для отображения числа
#ДЕЛ/0!	Попытка деления на ноль
#ЗНАЧ!	В формуле для математических вычислений содержится ссылка на ячейку с текстом
#ССЫЛКА!	Ячейка, ссылка на которую используется в формуле, не существует

Ввод текста в ячейку ЭТ также имеет некоторые особенности. По умолчанию текст выравнивается по левому краю. Если длина текста больше ширины ячейки, то текст на экране может отобразиться полностью, перекрыв свободные ячейки, расположенные правее. Если справа нет свободных ячеек, то видимая часть текста будет обрезана.

Чтобы ввести данные в новой строке ячейки, вставляют разрыв строки, нажав клавиши Alt + Enter.

Иногда требуется сохранить в виде текста числа, даты или формулы. Для этого их ввод в ячейку надо начинать с апострофа.

Сравните то, что будет отображено в ячейке при вводе в неё 2017 и '2017.

Для привлечения внимания к наиболее важной информации или ввода пояснений можно снабдить ячейки таблицы примечаниями.

Выясните, как можно создать примечание в табличном процессоре, имеющемся в вашем распоряжении.

1.3. Копирование и перемещение данных

Для выполнения операций копирования и перемещения данных в ЭТ соответствующие ячейку или диапазон ячеек сначала следует выделить, а затем можно воспользоваться командами **Копировать**, **Вырезать**, **Вставить** группы **Буфер обмена** вкладки **Главная**.

Для выделения несвязного диапазона ячеек можно выделить первую связную часть, а затем нажать клавишу Ctrl и, удерживая её, выделить следующие связные диапазоны.

Данные из одной ячейки можно вставить в другую ячейку или в диапазон ячеек, выделенный перед вставкой. Данные из связного диапазона можно вставить в один или несколько связных диапазонов ячеек того же размера. В последнем случае перед вставкой нужно выделить левую верхнюю ячейку каждого связного диапазона.

По умолчанию при вставке новые данные заменяют данные, имеющиеся в ячейках.

Если содержимым ячеек, которое копируется или перемещается, являются формулы, то можно вставить в выделенные ячейки не сами формулы, а вычисленные по ним значения. Для этого необходимо скопировать содержимое исходных ячеек в буфер обмена, выделить позицию вставки и выполнить команду **Главная** → **Вставить** → **Вставить значения**.

Данные из буфера обмена можно вставить в выделенные ячейки таким образом, что они не заменят имеющиеся там данные, а, например, суммируются с ними. Эту и ряд других возможностей можно исследовать в окне **Специальная вставка** (рис. 1.2), вызываемом командой **Главная** → **Вставить** → **Специальная вставка**.

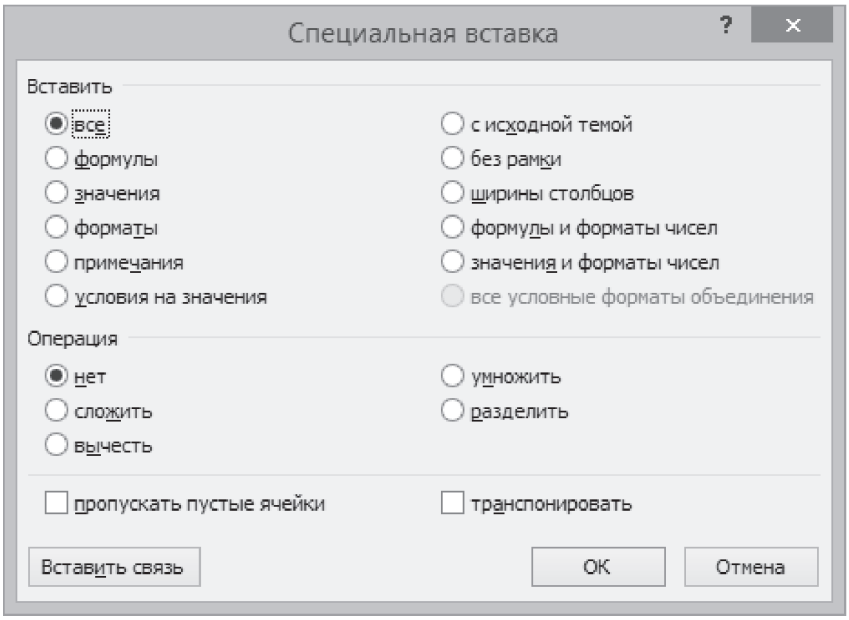


Рис. 1.2. Окно **Специальная вставка**

Если необходимо в несколько расположенных подряд ячеек ввести повторяющиеся данные или изменяющиеся по определённым закономерностям последовательности данных, то можно воспользоваться **функцией автозаполнения**. Для этого нужно:

1) внести данные в две первые ячейки;
2) выделить эти ячейки;
3) установить указатель мыши над **маркером заполнения** — маленьким чёрным квадратиком в правом нижнем углу выделенной ячейки (вид указателя мыши при наведении его на маркер заполнения меняется на знак +);

4) нажать левую кнопку мыши, протянуть (перетащить) указатель по заполняемым ячейкам и отпустить кнопку мыши.

Примените описанный выше приём для получения следующих рядов:
1) 1, 1, 1, ...;
2) 5, 10, 15, ...;
3) 100, 200, 300, ...;
4) урок, урок, урок, ...;
5) январь, февраль, март, ...;
6) товар 1, товар 2, товар 3, ...

Попробуйте добиться такого же результата с помощью команды **Заполнить** → **Прогрессия...** (вкладка **Главная**, группа **Редактирование**).

Любым способом введите следующие ряды:
1) 2, 4, 8, 16, ...;
2) 0, 5, 10, 15, ...

Вспомните, как называются такие последовательности.

Если содержимым ячейки является формула, включающая ссылки, то при копировании этой ячейки в формуле может происходить автоматическое изменение ссылок.

Ссылка, которая изменяется при копировании формулы, называется относительной.
Ссылка, которая не изменяется при копировании формулы, называется абсолютной.
Ссылка, в которой при копировании формулы изменяется только номер строки или только имя столбца, называется смешанной.

Большинство ссылок в формулах относительные. При копировании в составе формулы в другую ячейку они автоматически изменяются в соответствии с новым положением скопированной формулы, т. е. они изменяются относительно месторасположения формул. В этом состоит суть **принципа относительной адресации.**

При копировании формулы с относительными ссылками [столбец] [строка] на *n* строк ниже (выше) и на *m* столбцов правее (левее) ссылка заменяется на [столбец ± *n*] [строка ± *m*].
При копировании формулы в пределах одного столбца (одной строки) обозначения столбцов (номера строк) в формулах не изменяются.

Иногда нужно, чтобы при копировании формул адреса ячеек не менялись. В этом случае используют абсолютные ссылки.

Для создания абсолютной ссылки служит знак $. С помощью этого знака можно зафиксировать весь адрес ($A$1), только столбец ($A1) или только строку (A$1). В двух последних случаях говорят о смешанных ссылках.

Чтобы быстро преобразовать ссылку из относительной в абсолютную и наоборот, можно выделить её в строке ввода и нажать клавишу F4 (Microsoft Excel) или Shift+F4 (OpenOffice Calc).

Вспомните, как можно быстро получить смешанные ссылки.

Если в формуле для ссылки на ячейку использовать её имя, то при копировании формулы эта ссылка изменяться не будет. Иначе говоря, имя ячейки в формуле является абсолютной ссылкой.

При перемещении формулы имеющиеся в ней ссылки не изменяются.

Пример 1. В ячейке B1 записана формула =2*$A1.

Выясним, какой вид приобретёт формула, после того как содержимое ячейки B1 скопируют в ячейку C2.

В формуле используется смешанная ссылка: при копировании формулы имя столбца останется неизменным, а номер строки увеличится на 1. Таким образом, после копирования в ячейке C2 окажется формула =2*$A2.

Пример 2. Дан фрагмент электронной таблицы:

	A	B	C
1	15	13	
2	14	12	=($A2+B2)/2

Выясним, чему станет равным значение ячейки C1, если в неё скопировать формулу из ячейки C2.

Так как копирование формулы происходит внутри одного столбца, имена столбцов в формуле не изменятся, а номер строки в ссылках уменьшится на единицу.

Формула примет вид: =($A1+B1)/2. В ячейке C1 отобразится число 14.

Пример 3. Дан фрагмент электронной таблицы:

	A	B	C	D
1	1	1	1	1
2	2	2	2	=B$3+$C2
3	3	3	3	3
4	4	4	4	4

Выясним, чему будет равна сумма значений диапазона ячеек E1:E4 после копирования в него формулы из ячейки D2.

Формулы копируются в ячейки соседнего столбца. Поэтому буквенное обозначение столбца в относительной ссылке изменится на следующее по алфавиту. Следовательно, первое слагаемое в формуле примет вид: C$3 (ссылка на номер строки здесь абсолютная, она останется неизменной). Во втором слагаемом неизменным является обозначение столбца. А номер строки при копировании формулы в ячейки E1, E2, E3 и E4 соответственно: уменьшится на единицу, останется неизменным, увеличится на единицу, увеличится на 2.

	A	B	C	D	E
1	1	1	1	1	=C$3+$C1
2	2	2	2	=B$3+$C2	=C$3+$C2
3	3	3	3	3	=C$3+$C3
4	4	4	4	4	=C$3+$C4

После вычисления значений по формулам ячеек E1, E2, E3 и E4 (4, 5, 6 и 7) находим сумму значений диапазона ячеек E1:E4, равную 22.

Пример 4. Требуется с помощью формул в ЭТ построить таблицу двузначных чисел, фрагмент которой представлен на рисунке:

	A	B	C	D	E	F	G	H	I	J	K
1		**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
2	**1**	10	11	12	13	14	15	16	17	18	19
3	**2**	20	21	22	23	24	25	26	27	28	29
4	**3**	30	31	32	33	34	35	36	37	38	39
5	**4**	40	41	42	43	44	45	46	47	48	49

Возможный алгоритм действий может быть таким:

1) в диапазон B1:K1 ввести числа от 0 до 9 (можно воспользоваться маркером заполнения);
2) в диапазон A2:A10 ввести числа от 1 до 9;
3) в ячейку B2 записать формулу двузначного числа: =A2*10+B1 (A2 — число десятков; B1 — число единиц);
4) внести изменения в формулу с учётом следующего:
   - при копировании формулы вниз должен изменяться номер строки, «отвечающей» за количество десятков, а имя столбца A, «отвечающего» за разряд десятков, должно оставаться неизменным ($A2);
   - при копировании формулы вправо должно изменяться имя столбца, «отвечающего» за количество единиц, а номер строки 1, «отвечающей» за разряд единиц, должен оставаться неизменным (B$1);
5) скопировать отредактированную формулу (=$A2*10+B$1) во все ячейки диапазона B2:K10.

САМОЕ ГЛАВНОЕ

Прикладные программы, предназначенные для работы с данными, представленными в таблицах, называются табличными процессорами (ТП) или просто электронными таблицами (ЭТ).

Основными объектами табличного процессора являются рабочая книга, лист, электронная таблица, строка, столбец и ячейка.

Ячейка — это наименьшая структурная единица электронной таблицы, которая образуется на пересечении столбца и строки. Адрес ячейки определяется именем столбца и номером строки, на пересечении которых она находится. Содержимым ячейки может быть число, текст или формула.

Формула начинается со знака «=» и может содержать скобки, числа, тексты, ссылки на ячейки, знаки операций и функции.

В формулах используются не сами исходные данные, а ссылки на ячейки, в которых эти данные находятся. При изменении данных в ячейках происходит автоматический пересчёт значений всех формул, содержащих ссылки на эти ячейки.

Различают относительные, абсолютные и смешанные ссылки. Ссылка, которая изменяется при копировании формулы, называется относительной. Ссылка, которая не изменяется при копировании формулы, называется абсолютной. Ссылка, в которой при копировании формулы изменяется только номер строки или только имя столбца, называется смешанной.

Вопросы и задания

1. Что понимают под табличным процессором и электронными таблицами?
2. Сравните интерфейс известных вам текстового и табличного процессоров. Что у них общего? Чем они различаются?
3. Что такое адрес (имя) ячейки ЭТ? Как задаётся адрес ячейки, адрес диапазона ячеек?
4. Выясните, куда в табличном процессоре перемещается табличный курсор при нажатии клавиш Home, End, PageUp, PageDown. Куда перемещается табличный курсор при нажатии комбинации клавиш: Ctrl + →, Ctrl + ↓, Ctrl + ←, Ctrl + ↑, Ctrl + Home, Ctrl + End? Проведите аналогию с перемещениями текстового курсора в текстовом процессоре.
5. Какие типы данных могут быть занесены в ячейку ЭТ?
6. Какие существуют особенности ввода числовых значений в ЭТ?
7. Вспомните основные правила ввода формул в ЭТ. Где вы уже встречались с аналогичными правилами ввода арифметических выражений?
8. В чём суть принципа относительной адресации в ЭТ? Что происходит при копировании формул, содержащих относительные ссылки?
9. В каких случаях в формулах используются абсолютные ссылки?
10. В чём заключается преимущество использования ссылок в формулах?
11. На основании чего можно судить о том, что табличный процессор интерпретировал введённые в ячейку данные как текст? Как число?
12. Сравните приёмы копирования и вставки данных в текстовом и табличном процессорах. Что у них общего? Чем они различаются?
13. Как осуществляется автозаполнение ячеек?
14. Как ввести следующее четверостишие А. Ерикеева в одну ячейку электронной таблицы?

Наступила осень,
Пожелтел наш сад.
Листья на берёзе
Золотом горят.

15. Значение переменной x находится в ячейке A1, значение переменной y — в ячейке A2, значение переменной z — в ячейке A3. Запишите формулы для вычисления в электронных таблицах значений выражений:
    1) $(x + y + z) : 3$;
    2) $5x^3 + 4y^2 - 3z$.
16. Только путём ввода последовательностей составьте таблицу умножения:

	A	B	C	D	E	F	G	H	I
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81

    За сколько операций вам удалось это сделать?
17. Дан фрагмент электронной таблицы:

	A	B	C
1	40	10	=A1+B$1
2	30	20	

    Чему будет равно значение ячейки C2, если скопировать в неё формулу из ячейки C1?
18. В ячейке B3 записана формула =C$2+$D3+2. Какой вид приобретёт формула после копирования её в ячейку B2?
19. Измерьте длину, ширину и высоту кухни, прихожей и жилых комнат вашей квартиры. Создайте в табличном процессоре таблицу с результатами измерений. Вычислите площадь пола, площадь стен и объём каждого из помещений, а также общую площадь всех помещений.
20. В табличном процессоре создайте таблицу вида:

Страна	Площадь, кв. км	Население, млн чел.	Плотность	Проценты

Занесите в таблицу информацию о десяти странах, имеющих самую большую численность населения. Введите в соответствующие ячейки формулы для вычисления:

1) общей площади и общего количества населения этих десяти стран (предусмотрите соответствующие ячейки под созданной таблицей с данными);
2) плотности населения в каждой из этих стран;
3) процентов, которые составляет население каждой из этих стран по отношению к общему количеству населения в мире.

21. В табличном процессоре вычислите значения функции $y = x^2 + x - 12$ на промежутке $[-5; 5]$ с шагом 0,5.
22. Подготовьте краткое сообщение о первых электронных таблицах.

§ 2 Редактирование и форматирование в табличном процессоре

2.1. Редактирование книги и электронной таблицы

В процессе работы с табличным процессором в книгу часто приходится добавлять (вставлять) новые листы, удалять, переименовывать, перемещать или копировать существующие листы.

Самостоятельно найдите не менее двух способов вставки нового листа в книгу и удаления существующего листа из книги.

Исследуйте контекстное меню ярлычка листа. Выясните, какие операции с листом можно выполнить с его помощью.

Зачастую возникает необходимость внести в уже созданную электронную таблицу строки, столбцы или диапазоны ячеек.

Для вставки в таблицу новых столбцов (строк) необходимо выделить столбцы (строки), перед которыми нужно вставить новые, вызвать контекстное меню и выбрать в нём команду **Вставить**.

Если выделить один столбец (строку), то перед ним вставится один новый столбец (строка). Если же выделить несколько строк (столбцов) подряд, то перед ними вставится столько же новых столбцов (строк), сколько было выделено.

Существует несколько вариантов вставки ячейки или диапазона ячеек (рис. 1.3).

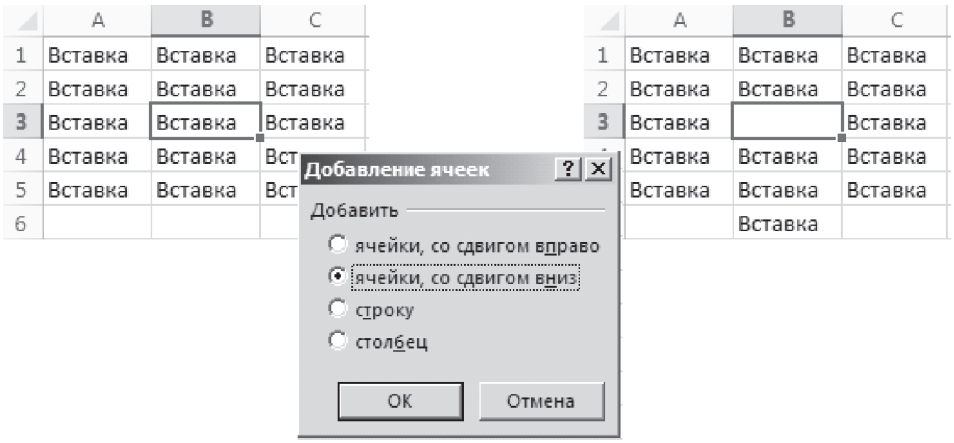


Рис. 1.3. Вставка пустой ячейки

Удаление строк, столбцов, ячейки или диапазона ячеек происходит аналогично.

Самостоятельно исследуйте возможности вставки (удаления) ячейки или диапазона ячеек, имеющиеся в окне **Добавление ячеек** (**Удаление ячеек**), вызываемом командой **Вставить...** (**Удалить...**) контекстного меню.

2.2. Форматирование объектов электронной таблицы

Основными операциями форматирования являются: форматирование данных, форматирование ячеек, изменение ширины столбцов и высоты строк.

Форматирование числовых данных. Формат отображения данных позволяет представлять их в наиболее подходящем для пользователя виде. При вводе любых данных используется формат **Общий**. Тип данных при этом определяется автоматически.

Для форматирования содержащихся в ячейке (ячейках) данных:

1) эту ячейку (ячейки) выделяют;
2) правой кнопкой мыши вызывают её контекстное меню;
3) вызывают диалоговое окно **Формат ячеек**;
4) задают необходимый формат на вкладке **Число**.

При форматировании данных сами данные не изменяются, изменяется лишь их внешний вид.
Реальное значение числовых данных можно увидеть в строке формул, сделав соответствующую ячейку текущей.

Пример. Введём в диапазон ячеек A1:A9 названия основных форматов данных. В ячейку B1 введём число 12,3 и скопируем его в диапазон B2:B9 (рис. 1.4).

Будем поочерёдно выделять ячейки диапазона B2:B9 и применять форматы, указанные в соответствующих ячейках столбца A, сравнивая при этом вид числа в ячейке и в строке формул.

Формат **Общий** является форматом по умолчанию. Он используется для представления чисел в большинстве случаев так, как они были введены. Если ширина ячейки недостаточна для отображения введённого с клавиатуры числа, оно автоматически представляется в экспоненциальном виде.

Формат **Числовой** используется для представления числа в виде десятичной дроби с заданным количеством десятичных знаков. Если число в ячейке имеет меньше десятичных знаков, чем предусмотрено форматом, то при отображении в ячейке оно будет дополнено нулями справа, а если больше — будет округлено.

Формат **Денежный** используется для установки значений тех же свойств, что и для формата **Числовой**, с добавлением к числу обозначения денежной единицы, которое выбирается из списка.

Формат **Дата** используется для представления числа в виде даты определённого типа. В Microsoft Excel в формате **Дата** все даты сохраняются как натуральные числа. Отсчёт начинается с

	A	B
1	Общий	12,3
2	Числовой, 2 десятич.знака	12,3
3	Денежный	12,3
4	Дата	12,3
5	Время	12,3
6	Процентный	12,3
7	Дробный	12,3
8	Экспоненциальный	12,3
9	Текстовый	12,3

	A	B
1	Общий	12,3
2	Числовой, 2 десятич.знака	12,30
3	Денежный	12,30р.
4	Дата	12.01.1900
5	Время	7:12:00
6	Процентный	1230,00%
7	Дробный	12 1/3
8	Экспоненциальный	1,23E+01
9	Текстовый	12,3

Рис. 1.4. Формат числа

01.01.1900, и этой дате соответствует число 1. Каждой следующей дате соответствует следующее натуральное число: 02.01.1900 — 2, 03.01.1900 — 3, ..., 06.06.2006 — 38 874, ..., 01.09.2010 — 40 422. Такое представление дат позволяет выполнять операции над ними. Так, количество дней между двумя датами определяется разностью чисел, которые соответствуют этим датам. Например, разность 01.09.2016 – 01.01.2016 будет вычисляться таким образом: 42 614 – 42 370 = 244.

В формате **Процентный** данные представляются числом, которое является результатом умножения содержимого ячейки на 100, со знаком % в конце.

Над данными, представленными в формате **Текстовый**, можно выполнять операции и как над числами, и как над текстами.

По умолчанию числа в формате **Текстовый** выравниваются в ячейке по левому краю, в других форматах — по правому краю.

Форматирование ячеек. При форматировании ячейки (ячеек) электронной таблицы можно устанавливать:

- границы ячейки, их цвет, тип линий и др.;
- цвет фона ячейки, цвет и стиль узора, способы заливки и др.;
- защиту ячейки, режим скрытия формул;
- формат числовых данных (числовой формат);
- значения свойств символов в ячейке: шрифт, начертание, размер, подчеркивание, горизонтальное и вертикальное выравнивание, ориентацию и др.

Для этого можно использовать элементы управления групп **Шрифт**, **Выравнивание**, **Число**, **Стили**, **Ячейки** вкладки **Главная** на ленте или элементы управления, расположенные на вкладках окна **Формат ячеек**. Окно **Формат ячеек** можно вызвать, используя кнопки открытия диалоговых окон групп **Шрифт**, **Выравнивание** или **Число**, а также с помощью контекстного меню ячейки (ячеек).

Операции изменения шрифта, цвета, размера и начертания символов в ячейках электронной таблицы аналогичны соответствующим операциям форматирования символов в текстовом процессоре.

Как и текстовые процессоры, и редакторы презентаций, табличные процессоры предоставляют возможности стилевого форматирования. В них включён стандартный набор стилей (вкладка **Главная**, группа **Стили**), которые можно использовать для оформления объектов электронной таблицы. Этот набор можно дополнять собственноручно разработанными стилями.

Формат ячейки можно применить к другим ячейкам, используя инструмент **Формат по образцу** (вкладка **Главная**, группа **Буфер обмена**).

Для очистки всех установленных форматов, т. е. для возврата к формату по умолчанию, нужно выделить ячейки и выполнить команду **Очистить → Очистить форматы** (вкладка **Главная**, группа **Редактирование**).

На вкладке **Защита** окна **Формат ячеек** можно установить или отменить режимы защиты ячеек и скрытия формул. Режим защиты ячеек запрещает несанкционированное изменение данных, а режим скрытия формул — отображение данных в строке формул. Для включения этих режимов нужно установить соответствующие флажки (**Защищаемая ячейка** и **Скрыть формулы**), щёлкнуть по кнопке **ОК**, после чего выполнить команду **Защитить лист** (вкладка **Рецензирование**, группа **Изменения**). Откроется окно **Защита листа**, в котором можно установить пароль для снятия режимов защиты и скрытия, а также разрешить выполнение определённых операций для данных режимов.

Форматирование электронной таблицы. Иногда ширины столбцов (высоты строк), установленной по умолчанию, не хватает, чтобы полностью отобразить содержимое ячеек, или наоборот, для более компактного вида заполненной части таблицы целесообразно уменьшить ширину некоторых столбцов (высоту некоторых строк).

Для быстрого автоподбора ширины всех столбцов листа нужно нажать кнопку **Выделить всё** и выполнить двойной щелчок мышью на границе между заголовками двух любых столбцов. Если дважды щёлкнуть на любой границе между заголовками двух строк, то выполнится автоподбор высоты всех строк выделенного диапазона.

Предложите ещё не менее двух способов изменения ширины столбцов (высоты строк).

В некоторых случаях удобно несколько ячеек, образующих связный диапазон, объединить в одну ячейку. В такую объединенную ячейку можно ввести, например, текст заголовка таблицы или нескольких столбцов. Для этого ячейки нужно выделить, а затем воспользоваться инструментом **Объединить и поместить в центре** (вкладка **Главная**, группа **Выравнивание**).

После такого объединения все эти ячейки будут рассматриваться как одна ячейка, адресом которой будет считаться адрес верхней левой ячейки соответствующего диапазона. Данные, которые были в ячейках до объединения (кроме тех, что были в

верхней левой ячейке), при объединении будут утеряны. Поэтому рекомендуется сначала объединить ячейки, а затем вводить данные. Редактирование и форматирование объединённой ячейки и её содержимого выполняются так же, как и обычной ячейки. Отменить объединение ячеек можно, выбрав эту ячейку и воспользовавшись инструментом объединения повторно.

Предложите ещё один способ объединения ячеек.

Если заполнено много столбцов (строк) таблицы, причём некоторые из них временно не нужны для работы, то их можно скрыть. Для этого следует выделить такие столбцы (строки) и отдать команду **Скрыть** контекстного меню выделенного диапазона. Для отображения скрытых объектов нужно выполнить команду **Показать**.

Попробуйте временно скрыть один из листов книги. Как вы это сделали?

Если заполненные данными ячейки не помещаются на экране, а некоторые из них необходимо постоянно держать «перед глазами», то можно установить режим закрепления областей. В этом режиме при прокрутке электронной таблицы некоторые столбцы (строки) не будут исчезать с экрана. Для этого нужно выделить определённую часть таблицы (табл. 1.3) и воспользоваться инструментом **Закрепить области** (вкладка **Вид**, группа **Окно**).

Таблица 1.3

Закрепление областей

Объект выделения	Область закрепления
Столбец	Вертикальная область левее выделенного столбца
Строка	Горизонтальная область над выделенной строкой
Ячейка	Область левее и выше выделенной ячейки

Команды **Закрепить верхнюю строку** и **Закрепить первый столбец** позволяют закрепить указанные объекты таблицы без их выделения.

Для отказа от закрепления областей предназначена команда **Снять закрепление областей**.

САМОЕ ГЛАВНОЕ

Редактирование книги состоит в добавлении в неё новых листов, удалении, перемещении или копировании существующих листов.

Редактирование электронной таблицы — вставка или удаление строк, столбцов или диапазонов ячеек.

Основными операциями форматирования являются: форматирование данных, форматирование ячеек, изменение ширины столбцов и высоты строк.

При форматировании данных сами данные не изменяются, изменяется лишь их внешний вид. Реальное значение данных можно увидеть в строке формул, сделав соответствующую ячейку текущей.

При форматировании ячеек электронной таблицы можно устанавливать:

- границы ячейки, их цвет, тип линий и др.;
- цвет фона ячейки, цвет и стиль узора, способы заливки и др.;
- защиту ячейки, режим скрытия формул;
- формат числовых данных (числовой формат);
- значения свойств символов в ячейке: шрифт, начертание, размер, подчеркивание, горизонтальное и вертикальное выравнивание, ориентацию и др.

В некоторых случаях удобно несколько ячеек, образующих связный диапазон, объединить в одну ячейку.

Если заполнено много столбцов (строк) таблицы, причём некоторые из них временно не нужны для работы, то их можно скрыть.

Если заполненные данными ячейки не помещаются на экране, а некоторые из них необходимо постоянно держать «перед глазами», то можно установить режим закрепления областей.

Вопросы и задания

1. Какие операции можно отнести к операциям редактирования данных? К операциям редактирования книги? К операциям редактирования электронной таблицы?
2. Перечислите основные операции, выполняемые с листами книги.
3. Как вставить в электронную таблицу пустые строки (столбцы)?
4. Как удалить из электронной таблицы строки (столбцы)?
5. Как можно изменить размеры ячеек, столбцов, строк электронной таблицы? Назовите несколько способов.
6. Для чего предназначено скрытие строк (столбцов)? Как это можно сделать? Как отобразить скрытые объекты?

7. Какие способы вызова окна **Формат ячеек** вам известны?
8. Дайте краткую характеристику форматам **Общий**, **Числовой**, **Денежный**, **Дата**, **Процентный**, **Текстовый**. Как их можно установить?
9. Исследуйте вкладку **Выравнивание** окна **Формат ячеек**. Значения каких свойств данных в ячейках можно установить с её помощью? Каким ещё способом можно это сделать?
10. Введите в электронную таблицу необходимые данные и оформите их по образцу:

	A	B	C	D	E
1	Текст	Текст	Текст	Текст	Текст
2	Текст	Текст	Текст	Текст	Текст
3	Текст	Текст	Текст	Текст	Текст
4	Текст	Текст	Текст	Текст	Текст

11. Значения каких свойств символов в ячейках можно установить на вкладке **Шрифт** окна **Формат ячеек**? Каким ещё способом можно это сделать?
12. Значения каких свойств ячеек можно установить на вкладке **Границы** окна **Формат ячеек**? Каким ещё способом можно это сделать?
13. Значения каких свойств ячеек можно установить на вкладке **Заливка** окна **Формат ячеек**? Каким ещё способом можно это сделать?
14. Введите в электронную таблицу необходимые данные и оформите их по образцу (название цвета определяет цвет его шрифта и цвет фона ячейки справа от него):

	A	B	C	D
1				
2		Цвет ячейки		
3		красный		
4		зелёный		
5		синий		
6				

15. Значения каких свойств ячеек можно установить на вкладке **Защита** окна **Формат ячеек**? Каким ещё способом можно это сделать?
16. Как можно скопировать формат ячейки на другие ячейки?

§ 3
Встроенные функции и их использование

3.1. Общие сведения о функциях

В любом табличном процессоре используются встроенные функции.

> Встроенная функция — это заранее написанная процедура преобразования данных.

Всё многообразие встроенных в табличные процессоры функций принято делить на категории по их назначению, выделяя среди них математические, статистические, логические, текстовые, финансовые и другие типы функций.

Каждая встроенная функция имеет имя — как правило, это сокращённое название производимого ею действия. Функции вызываются с некоторыми аргументами и возвращают единственное значение — результат обработки.

Аргументом функции может быть число, текст, выражение, ссылка на ячейку или диапазон ячеек, результат другой функции.

Можно выделить функции:

- с одним аргументом, например КОРЕНЬ;
- с несколькими аргументами, количество которых фиксировано, например ОКРУГЛ;
- с нефиксированным количеством аргументов, например МАКС;
- с некоторыми необязательными аргументами, например РАНГ;
- без аргументов, например ТДАТА.

При использовании функции в формуле сначала указывается её имя, а затем в скобках указывается список аргументов через точку с запятой (табл. 1.4).

Таблица 1.4

Примеры записи функций в Microsoft Excel

Функция	Запись в Microsoft Excel
Квадратный корень	КОРЕНЬ(A1)
Округление числа до заданного количества десятичных разрядов	ОКРУГЛ(G13;2)
Среднее значение	СРЗНАЧ(A3:B10)
Максимальное значение	МАКС(A3:B10; C8:C12; M6)
Текущие дата и время	ТДАТА()

Назначение каждой функции, наличие аргументов, их количество и тип можно посмотреть в **Справке** или в комментариях при вводе функции в формулу.

Вставить функцию в формулу можно несколькими способами:

1) использовать кнопки категорий функций в группе **Библиотека функций** вкладки **Формулы** на ленте;
2) воспользоваться инструментом **Вставить функцию** в группе **Библиотека функций** или в строке формул;
3) ввести функцию непосредственно в ячейку или в поле **Строка формул**.

Рассмотрим более подробно второй способ.

Если щёлкнуть на кнопке **Вставить функцию** строки формул, то откроется окно **Мастер функций** (рис. 1.5), а в текущую ячейку автоматически вставится знак «=» (если в этой ячейке ввод формулы ещё не начинался). В окне **Мастер функций** в списке поля **Категория** можно выбрать нужную категорию, после чего в списке поля **Выберите функцию** выбрать нужную функцию.

После выбора имени функции в текущую ячейку автоматически вставляется имя функции и пара круглых скобок, а также

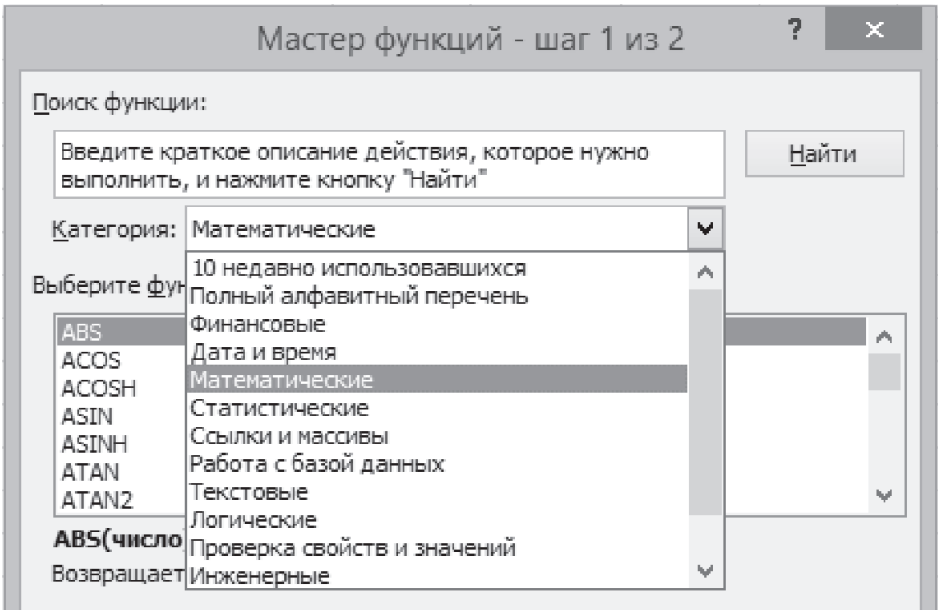


Рис. 1.5. Окно **Мастер функций**

открывается окно **Аргументы функции** с полями для ввода аргументов этой функции (рис. 1.6).

Если функция имеет фиксированное количество аргументов, то в окне **Аргументы функции** сразу отображается соответствующее количество полей для их ввода. Если функция имеет нефиксированное количество аргументов, то в окне сначала появляется несколько полей, а следующие поля появляются уже в процессе ввода аргументов.

Если аргументом является число или текст, то его нужно вводить в поле с клавиатуры. Если аргументом является ссылка на ячейки, то её также можно ввести с клавиатуры, но лучше выделить соответствующие ячейки с помощью мыши. Для этого:

1) выберите кнопку **Свернуть** соответствующего поля для ввода аргумента функции (после этого окно **Аргументы функции** изменит свой вид: в нём кроме строки заголовка останется только это поле, а вместо кнопки **Свернуть** появится кнопка **Развернуть**);

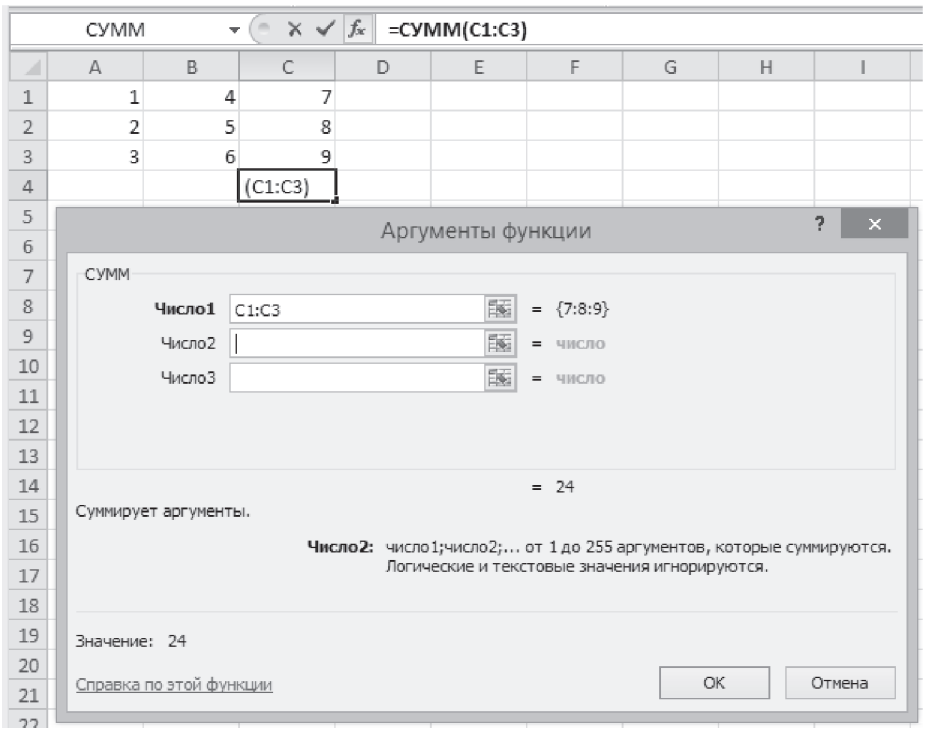


Рис. 1.6. Окно **Аргументы функции** функции СУММ

2) выделите нужные ячейки (ссылки на них автоматически вставятся в соответствующее поле и в формулу);
3) выберите кнопку **Развернуть** (после этого окно **Аргументы функции** примет свой предыдущий вид);
4) при необходимости повторите шаги 1–3 для других аргументов функции;
5) после ввода в поля всех нужных аргументов функции выберите кнопку **ОК**.

Для некоторых функций Microsoft Excel автоматически предлагает первый аргумент. Например, для функции СУММ предлагается найти сумму чисел диапазона ячеек, заполненных числовыми данными, которые находятся над ячейкой с формулой (см. рис. 1.6) или слева от неё, если верхний диапазон ячеек пуст. Это предложение можно принять (если оно соответствует плану проводимых вычислений) или ввести вместо автоматически предложенного аргумента другой.

Два других способа вставки функции в формулу исследуйте самостоятельно. Назовите их основные отличия друг от друга. Что у них общего? Какой из способов будете применять вы?

В электронных таблицах используется большое количество встроенных функций. Познакомимся более подробно с теми из них, которые могут пригодиться вам в учебной и исследовательской деятельности, а также в повседневной жизни.

3.2. Математические и статистические функции

Для решения математических задач (решения уравнений, построения графиков функций) вам могут быть полезны функции, представленные в таблице 1.5. Здесь же представлены некоторые из статистических функций, позволяющих автоматизировать статистическую обработку данных. С их помощью можно вычислить наименьшее значение, подсчитать количество ячеек, содержащих заданную информацию, и т. д.

Рассмотрим более детально работу статистической функции РАНГ, имеющую формат:

РАНГ(число; ссылка на список; [порядок])

Здесь:

- число — это число, для которого определяется ранг (порядок);

Таблица 1.5

Некоторые математические и статистические функции

Функция	Количество аргументов	Запись	Результат
ABS(число)	1	ABS(F5)	Модуль (абсолютная величина) числа
SIN(число)	1	SIN(D4)	Синус числа (угла в радианах)
РАДИАНЫ(число)	1	РАДИАНЫ(A10)	Перевод из градусной меры угла в радианную
ГРАДУСЫ(число)	1	ГРАДУСЫ(C6)	Перевод радианной меры угла в градусы
ПИ()	0	ПИ()	Значение числа π
СТЕПЕНЬ(число; степень)	2	СТЕПЕНЬ(A2; 5)	Число, возведённое в степень

Окончание табл. 1.5

Функция	Количество аргументов	Запись	Результат
СУММ(число1; [число2]; ...)	От 1 до 255; все, кроме первого, необязательные	СУММ(A3:B10)	Сумма чисел, указанных в скобках
ОКРУГЛ(число; число_разрядов)	2	ОКРУГЛ(G13; 2)	Число, округлённое до заданного количества десятичных разрядов
СЧЁТ(значение1; [значение2]; ...)	От 1 до 255; все, кроме первого, необязательные	СЧЁТ(A3:B10; G13)	Количество чисел в указанных ячейках
МИН(число1; [число2]; ...)	От 1 до 255; все, кроме первого, необязательные	МИН(A3:B10; G13:G23)	Наименьшее среди указанных в скобках чисел
РАНГ(число; ссылка на список; [порядок])	3; третий необязательный	РАНГ(A1; $A$1:$A$5; 1)	Ранг числа в списке чисел

- ссылка на список — ссылка на список, которому принадлежит число (нечисловые значения в ссылке игнорируются);
- порядок — способ упорядочения значений списка:
  — 0 или отсутствие параметра — определяет ранг (позицию, место) числа в списке так, как если бы список был отсортирован в порядке убывания (т. е. максимальному значению присваивается ранг равный 1, чуть меньшему числу — ранг 2 и т. д.);
  — число, не равное 0, — определяет ранг числа так, как если бы список сортировался в порядке возрастания (т. е. минимальному числу присваивается ранг 1, чуть большему числу — ранг 2 и т. д.).

Функция РАНГ присваивает повторяющимся числам одинаковый ранг. При этом наличие повторяющихся чисел влияет на ранг последующих чисел.

В ячейку B1 введена и скопирована в B2:B6 одна из двух следующих формул:
1) =РАНГ(A1; $A$1:$A$6; 1);
2) =РАНГ(A1; $A$1:$A$6; 0).

По какой из формул представлены результаты вычислений в столбце B?

Как вы можете объяснить отсутствие числа 2 среди значений ячеек диапазона C1:C6, если это — результаты вычислений по другой из приведённых выше формул?

	A	B	C
1	45	5	1
2	12	4	3
3	8	3	4
4	45	5	1
5	6	2	5
6	3	1	6

3.3. Логические функции

Функция, результатом которой является ИСТИНА или ЛОЖЬ, называется логической.

К категории логических относятся функции ЕСЛИ, И, ИЛИ, ИСТИНА, ЛОЖЬ, НЕ.

Функции И, ИЛИ, НЕ позволяют создавать составные логические выражения. Формат этих функций:

И(логическое_значение1; [логическое_значение2]; ...)

ИЛИ(логическое_значение1; [логическое_значение2]; ...)

НЕ(логическое_значение)

Аргументами функций И, ИЛИ, НЕ могут быть логические выражения или ссылки на ячейки, содержащие логические значения.

Функция ЕСЛИ имеет формат:

ЕСЛИ(лог_выражение; значение_если_истина; значение_если_ложь)

Значение этой функции определяется так:

- если лог_выражение имеет значение ИСТИНА, то значение функции равно значению выражения значение_если_истина;
- если лог_выражение имеет значение ЛОЖЬ, то значение функции равно значению выражения значение_если_ложь.

Табличные процессоры имеют и такие функции, которые вычисляют сумму, среднее арифметическое, количество не всех значений из диапазонов ячеек, а только тех, которые удовлетворяют определённому условию:

- функция СУММЕСЛИ вычисляет сумму тех чисел из указанного диапазона, которые удовлетворяют заданному условию;
- функция СРЗНАЧЕСЛИ вычисляет среднее арифметическое тех чисел из указанного диапазона, которые удовлетворяют заданному условию;
- функция СЧЁТЕСЛИ подсчитывает количество ячеек из указанного диапазона, содержимое которых удовлетворяет заданному условию.

Пример 1. Выясним, сколько решений имеет логическое уравнение $((x_1 \to x_2) \to (x_3 \to x_4)) = 1$.

Преобразуем исходное уравнение, выразив импликацию через инверсию и дизъюнкцию:

$$\overline{(\overline{x_1} \vee x_2)} \vee (\overline{x_3} \vee x_4) = (x_1 \,\&\, \overline{x_2}) \vee \overline{x_3} \vee x_4 = 1.$$

Запишем формулу для вычисления логического выражения с помощью логических функций Microsoft Excel:

=ИЛИ(И(X1;НЕ(X2)); НЕ(X3); X4).

Внесём данные в таблицу и выполним расчёты — рис. 1.7.

E18 | fx | =СЧЁТЕСЛИ(E2:E17;ИСТИНА)

	A	B	C	D	E	F
1	X1	X2	X3	X4	Результат	
2	0	0	0	0	ИСТИНА	
3	0	0	0	1	ИСТИНА	
4	0	0	1	0	ЛОЖЬ	
5	0	0	1	1	ИСТИНА	
6	0	1	0	0	ИСТИНА	
7	0	1	0	1	ИСТИНА	
8	0	1	1	0	ЛОЖЬ	
9	0	1	1	1	ИСТИНА	
10	1	0	0	0	ИСТИНА	
11	1	0	0	1	ИСТИНА	
12	1	0	1	0	ИСТИНА	
13	1	0	1	1	ИСТИНА	
14	1	1	0	0	ИСТИНА	
15	1	1	0	1	ИСТИНА	
16	1	1	1	0	ЛОЖЬ	
17	1	1	1	1	ИСТИНА	
18					13	

Рис. 1.7. Решение логического уравнения (пример 1)

Итак, исходное уравнение имеет 13 решений — столько раз встречается значение ИСТИНА в диапазоне E2:E17. Для подсчёта этого значения можно воспользоваться функцией СЧЁТЕСЛИ.

Вспомните другой способ решения этого уравнения.

3.4. Финансовые функции

Финансовые функции используются для вычисления размеров выплат при погашении кредитов, банковских процентов на вклады, для определения процентной ставки и др.

Рассмотрим несколько финансовых функций, которыми полезно уметь пользоваться каждому человеку, планирующему взять в банке кредит[1)] или сделать вклад[2)]. Аргументами этих функций являются:

1) Кредит — это ссуда, предоставленная кредитором (в данном случае банком) заёмщику под определённые проценты за пользование деньгами.

2) Вклад — денежные средства, внесённые физическим или юридическим лицом в финансовое учреждение на хранение, в рост или для участия в получении прибыли.

- ставка — процентная ставка за период;
- плт — выплата, производимая в каждый период (месяц, квартал, год и т. п.);
- пс — приведённая (нынешняя) стоимость инвестиции;
- кпер — общее число периодов платежей по кредиту;
- бс — будущая стоимость инвестиции;
- тип — число 0, если оплата в конце периода; число 1, если оплата в начале периода (по умолчанию — 0).

Пример 2. Пусть ставка кредита в некотором банке составляет 18% годовых. Клиент хочет взять кредит на сумму 100 000 руб. и может выплачивать банку по 4000 руб. ежемесячно. Нужно определить, за сколько периодов клиент сможет погасить этот кредит.

Функция КПЕР(ставка; плт; пс; [бс]; [тип]) возвращает количество периодов платежей для инвестиции на основе периодических постоянных выплат и постоянной процентной ставки.

Обязательные аргументы функции:

- ставка — годовая ставка в процентах, разделённая на количество периодов платежей за год (в нашем примере это 18%/12);
- плт — сумма, которую клиент ежемесячно должен возвращать банку (в нашем примере это –4000, т. к. эти деньги отдаются);
- пс — размер кредита (в нашем примере это 100 000).

Формула для вычисления количества периодов выплат для погашения взятого кредита будет иметь вид:

=КПЕР(18%/12; –4000; 100000).

Получаем приблизительно 32 периода (месяца), т. е. более 2,5 лет.

Пример 3. Выясним, на какую сумму клиент может взять кредит, если ставка 19% годовых, а выплачивать он может по 12 000 руб. на протяжении двух лет (24 периода).

Функция ПС(ставка; кпер; плт; [бс]; [тип]) возвращает приведённую (к текущему моменту) стоимость инвестиции, представляющую собой общую сумму, которая на данный момент равноценна ряду будущих выплат.

Обязательные аргументы функции:

- ставка (19%/12);
- кпер — общее количество периодов выплаты платежей по кредиту (24);
- плт (–12 000).

Формула для вычисления размера кредита будет иметь вид:

$$=\text{ПС}(19\%/12;\ 24;\ -12000).$$

Получаем приблизительно 238 054 руб.

Пример 4. Пусть клиент хочет взять кредит 100 000 руб. на 2 года. При этом выплачивать он может по 5000 руб. ежемесячно. Может ли он воспользоваться предложением банка, ставка по кредитам в котором составляет 20%?

Функция СТАВКА(кпер; плт; пс; [бс]; [тип]; [предположение]) вычисляет процентную ставку за период (а не за год).

Обязательные аргументы функции:

- кпер (24);
- плт (–5000);
- пс (100 000).

Формула для вычисления ставки будет иметь вид:

$$=\text{СТАВКА}(24;\ -5000;\ 100000).$$

В результате вычислений получаем процентную ставку за месяц 1,51308%. Соответственно, процентная ставка за год составит 18,157% (1,51308 · 12).

Таким образом, клиенту не рекомендуется брать кредит в банке, ставка по кредитам в котором составляет 20%.

Пример 5. Клиент хочет сделать вклад на 3 года на сумму 300 000 руб. под 11% годовых с ежемесячным начислением процентов. Выясним, какую сумму он получит по окончании срока вклада.

Функция БС(ставка; кпер; плт; [пс]; [тип]) возвращает будущую стоимость инвестиции при условии периодических равных платежей и постоянной процентной ставки. Иначе говоря, с её помощью можно вычислить сумму, которую выплатят клиенту за вклад под определённые проценты по окончании срока вклада.

Аргументы функции:

- ставка — годовая ставка в процентах, разделённая на количество периодов начисления процентов за год (в нашем примере это 11%/12);
- кпер — количество периодов начисления процентов (3 · 12 = 36);
- плт — сумма, которая добавляется к вкладу каждый период времени: 0 или отрицательное число (в нашем примере это 0, т. к. пополнение вклада клиентом не предусмотрено);
- пс — начальная сумма вклада (в нашем примере это 300 000).

Формула для вычисления суммы, которую клиент получит за вклад по окончании срока вклада, будет иметь вид:

=БС(11%/12; 36; 0; –300000).

В результате вычислений получаем 416 663,58 руб.

Пример 6. Клиент хочет сделать вклад на 2 года на сумму 100 000 руб. под 10,5% годовых с ежемесячным начислением процентов. При этом он имеет возможность ежемесячно пополнять вклад ещё на 2000 рублей. Выясним, какую сумму клиент получит по окончании срока вклада.

Для нахождения результата мы воспользуемся той же функцией, что и в примере 5. Отличие состоит в том, что аргумент плт в этом случае примет значение –2000.

Формула для вычисления суммы, которую клиент получит за вклад по окончании срока вклада, будет иметь вид:

=БС(10,5%/12; 24; –2000; –100000).

В результате вычислений получаем 176 409,84 руб.

Как изменится формула в примере 6, если клиент ежемесячно будет не пополнять счёт на 2000 руб., а снимать со счёта по 1000 руб.?

3.5. Текстовые функции

В основном табличные процессоры используются для работы с числами, но в них предусмотрена и возможность работы с текстом. Например, в электронные таблицы заносятся наименования товаров и услуг, фамилии, имена и отчества сотрудников, партнёров и клиентов, их адреса, телефоны и многое другое.

Для обработки текста в табличных процессорах имеется набор функций, которые можно использовать для определения длины текста, номера позиции первого вхождения символа в текст, части текста, который удовлетворяет определённому условию и др.

Аргументами текстовых функций могут быть текстовые данные (их нужно заключать в кавычки), ссылки на ячейки с текстом, ссылки на ячейки с числами.

Рассмотрим примеры некоторых текстовых функций Microsoft Excel.

Функция СТРОЧН преобразует все буквы обрабатываемого текста в строчные, а функция ПРОПИСН, наоборот, — в прописные. Функция ПРОПНАЧ делает прописной первую букву каждого слова, а все остальные буквы — строчными.

Функция СОВПАД позволяет сравнить две текстовые строки в Microsoft Excel. Если они в точности совпадают, то возвращается значение ИСТИНА, в противном случае — ЛОЖЬ (функция учитывает регистр, но игнорирует различие в форматировании).

Какое значение появится в ячейке C1, если в неё записать формулу =СОВПАД(A1; B1)? Какое значение появится в ячейке C2, если в неё скопировать формулу из ячейки C1?

	A	B	C	D	E
1	Строка	строка			
2	Строка	*Строка*			

Объясните следующий результат сравнения двух текстов:

C1 | f_x =СОВПАД(A1;B1)

	A	B	C	D	E
1	Строка	Строка	ЛОЖЬ		

Функция СЖПРОБЕЛЫ удаляет из текста все лишние пробелы, кроме одиночных между словами. Эту функцию полезно применять к данным, которые импортируются в рабочие листы Microsoft Excel из внешних источников.

Вспомните, как можно удалить все лишние пробелы из документа с помощью инструментов текстового процессора.

Кроме лишних пробелов импортируемые данные могут содержать и различные непечатаемые символы. Для удаления из текста всех непечатаемых символов предназначена функция ПЕЧСИМВ.

Выскажите свои предположения о назначении текстовых функций ДЛСТР, ЛЕВСИМВ, ПРАВСИМВ, ПСТР по результатам их работы:

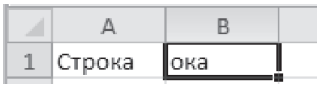
1) B1=ДЛСТР(A1)

	A	B
1	Строка	6

2) B1=ЛЕВСИМВ(A1;3)

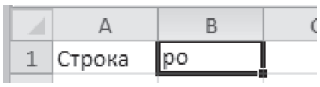
	A	B
1	Строка	Стр

3) B1=ПРАВСИМВ(A1;3)



	A	B
1	Строка	ока

4) B1=ПСТР(A1;3;2)



	A	B
1	Строка	ро

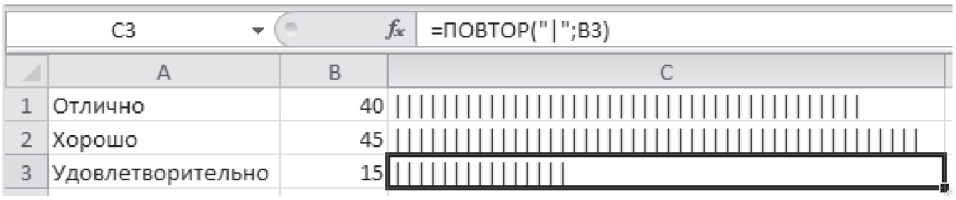
Функция СЦЕПИТЬ последовательно объединяет значения указанных аргументов в одну строку.

Функция ПОВТОР повторяет текстовую строку указанное количество раз. Строка задаётся как первый аргумент функции, а количество повторов — как второй.

Чему равен результат вычисления по формуле ячейки C2, если результат вычисления по формуле ячейки A2 равен 6?

	A	B	C
1	Строка	=ПОВТОР(A1;3)	
2	=ДЛСТР(A1)	=СЦЕПИТЬ(A1;B1;6)	=ДЛСТР(B2)

Функцию ПОВТОР можно применить и для «графического» представления числовых значений. Например, с её помощью можно визуализировать информацию об успеваемости некоторого ученика, получившего в текущем триместре 40 отметок «отлично», 45 — «хорошо» и 15 — «удовлетворительно».



C3 | fx | =ПОВТОР("|";B3)

	A	B	C
1	Отлично	40	||||||||||||||||||||||||||||||||||||||||
2	Хорошо	45	||||||||||||||||||||||||||||||||||||||||||
3	Удовлетворительно	15	|||||||||||||||

Функции НАЙТИ и ПОИСК очень похожи. Они находят вхождение одной строки в другую и возвращают положение первого символа искомой фразы относительно начала текста. Различие в том, что первая учитывает регистр, а вторая — нет.

Какие значения будут отображены в ячейках A2 и B2?

	A	B
1	Microsoft Office	Of
2	=НАЙТИ(B1;A1)	=ПОИСК(B1;A1)

Функция ПОДСТАВИТЬ заменяет определённый текст или символ на новое значение. Её применяют, когда заранее известно, какой текст необходимо заменить, а не его местоположение.

Функция ЗАМЕНИТЬ заменяет символы в заранее известном месте строки на новые. Функцию применяют, когда известно, где располагается текст, при этом сам он не важен.

1. Что будет отображено в ячейках B1 и B2?

	A	B
1	Excel 2010 Word 2010	=ПОДСТАВИТЬ(A1;10;13)
2		=ЗАМЕНИТЬ(A1; 9;2;"13")

2. С помощью какой из двух последних рассмотренных функций можно удалить все пробелы из текстовой строки? Как это сделать?
3. В ячейке содержится текст «колокол» (без кавычек). Что будет результатом вычислений по формуле:

   =ДЛСТР(A1)–ДЛСТР(ПОДСТАВИТЬ(A1;"о";""))?
4. Сформулируйте алгоритм подсчёта количества вхождений определённого символа в заданную строку.

САМОЕ ГЛАВНОЕ

В любом табличном процессоре используются встроенные функции — заранее написанные процедуры преобразования данных.

Каждая встроенная функция имеет имя — как правило, это сокращённое название производимого ею действия. Функции вызываются с некоторыми аргументами и возвращают единственное значение — результат обработки.

Аргументом функции может быть число, текст, выражение, ссылка на ячейку или диапазон ячеек, результат другой функции.

Всё многообразие встроенных в табличные процессоры функций принято делить на категории по их назначению, выделяя среди них математические, статистические, логические, текстовые, финансовые и другие типы функций.

Для решения уравнений, построения графиков функций и т. д. могут быть полезны математические функции.

Для автоматизации статистической обработки данных предназначены статистические функции. С их помощью можно вычислить наибольшее, наименьшее или среднее значение, подсчитать количество ячеек, содержащих заданную информацию, и т. д.

Функция, результатом которой является ИСТИНА или ЛОЖЬ, называется логической.

Финансовые функции используются для вычисления размеров выплат при погашении кредитов, банковских процентов на вклады, для определения процентной ставки и др.

Для обработки текста в табличных процессорах имеется набор функций, которые можно использовать для определения длины текста, номера позиции первого вхождения символа в текст, части текста, который удовлетворяет определённому условию и др.

Вопросы и задания

1. Раскройте суть математического понятия «функция». Что такое аргумент функции? Какие функции вы знаете из курса алгебры?
2. Что представляют собой функции в электронных таблицах? На какие категории они подразделяются?
3. Выясните, чему равен результат функции ОКРУГЛ, если заданное число разрядов больше нуля, меньше нуля, равно нулю.
4. Сколько аргументов могут иметь функции в электронных таблицах? Приведите примеры.
5. Данные каких типов могут быть аргументами функций? Приведите примеры.
6. Какие функции относятся к категории логических?
7. Какие значения будут в ячейках диапазона A2:B5 в результате вычисления по соответствующим формулам?

	A	B
1	-10	10
2	=И(A1>5;A1<0)	=НЕ(B1<20)
3	=ИЛИ(B1<10;B1>=20)	=И(ИЛИ(B1>5;B1<-5);НЕ(B1>10))
4	=НЕ(И(A1>-2;B1>0))	=ИЛИ(И(A1>2;A1<=10);B1<>0)
5	=НЕ(И(A2>-2;B2>0))	=НЕ(И(A1<100;B1=0))

8. Прочитайте формулу:

=ЕСЛИ(A1=100; "Всегда"; ЕСЛИ(И(A1>=80; A1<100); "Обычно"; ЕСЛИ(И(A1>=60; A1<80); "Иногда"; "Никогда"))).

Постройте фрагмент блок-схемы, соответствующий формуле.

9. Какие формулы надо использовать, чтобы для заданных значений переменной x вычислить соответствующие значения функции:

$$y=\begin{cases}\sin x, & x\le -5;\\ x^2, & -5<x<5;\\ \dfrac{1}{x^2-4x}, & x\ge 5.\end{cases}$$

10. Десять спортсменов-многоборцев принимают участие в соревнованиях по пяти видам спорта: бег на 60 м с барьерами, прыжок в высоту, толкание ядра, прыжок в длину, бег на 800 м. На квалификационном этапе по каждому виду спорта спортсмен может набрать от 0 до 30 очков. Спортсмен проходит в группу финалистов, если он набирает в сумме 100 и более очков. Создайте электронную таблицу следующего вида:

	A	B	C	D	E	F	G	H
1	Фамилия	Бег на 60 м с барьерами	Прыжок в высоту	Толкание ядра	Прыжок в длину	Бег на 800 м	Сумма баллов	Результат
2								
3								
				...				
10								
11								
12	Количество финалистов							

Введите данные и выполните необходимые расчёты.

11. Как изменится цена некоторого товара, если сначала её увеличить на 25%, а затем уменьшить на 25%?
12. Клиент хочет выяснить, какие условия вклада в банк выгоднее ему: 10,5% годовых с начислением процентов ежемесячно или 12% годовых с начислением процентов каждые полгода. Какая функция нужна для решения этой задачи?
13. Для чего в табличный процессор включены текстовые функции?

§ 4
Инструменты анализа данных

4.1. Диаграммы

Как правило, электронные таблицы содержат большое количество числовых данных, которые требуется сравнивать, оценивать их изменение с течением времени, определять соотношение между ними т. д. Проводить подобный анализ большого количества числовых данных значительно легче, если изобразить их графически (визуализировать). Для графического представления числовых данных используются диаграммы.

Диаграмма — это графическое представление числовых данных, позволяющее быстро оценить соотношение нескольких величин.

Табличные процессоры позволяют строить диаграммы следующих типов:

- гистограмма;
- линейчатая диаграмма;
- круговая диаграмма;
- график;
- диаграмма с областями;
- поверхностная диаграмма;
- лепестковая диаграмма и др.

Чтобы в Microsoft Excel просмотреть все доступные типы диаграмм, изучите группу **Диаграммы** на вкладке **Вставка** (рис. 1.8).

Выясните, какие типы диаграмм можно создавать в табличном процессоре, имеющемся в вашем распоряжении.

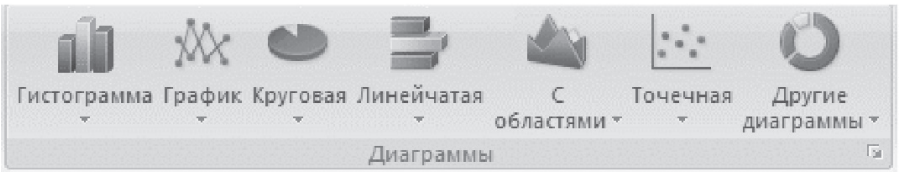


Рис. 1.8. Типы диаграмм в Microsoft Excel

В диаграмме любого типа можно выделить следующие объекты (рис. 1.9):

1 — область диаграммы (в ней размещаются все объекты диаграммы);

2 — название диаграммы, чётко описывающее то, что представлено на диаграмме;

3 — область построения диаграммы (непосредственно в ней располагается сама диаграмма);
4 — ось значений (вертикальная, ось *Y*). На ней находится шкала с определённым шагом, устанавливаемым автоматически, в зависимости от наименьшего и наибольшего значений данных, изображённых на диаграмме. Именно по этой шкале можно оценить данные, представленные на диаграмме;
5 — ряды данных — наборы числовых данных, некоторым образом связанных между собой и размещённых в электронной таблице в одной строке или столбце. На диаграмме ряд данных изображается геометрическими фигурами одного вида и цвета;
6 — ось категорий (горизонтальная, ось *X*). На ней отображаются значения определённого свойства данных;
7 — легенда, поясняющая соответствие между названиями рядов и используемыми на диаграмме цветами. По умолчанию названия рядов являются названиями строк (или столбцов) диапазона данных, по которым построена диаграмма;
8 — названия осей.

Воспроизведите в табличном процессоре диаграмму, представленную на рисунке 1.9. С помощью контекстного меню исследуйте свойства каждого объекта этой диаграммы.

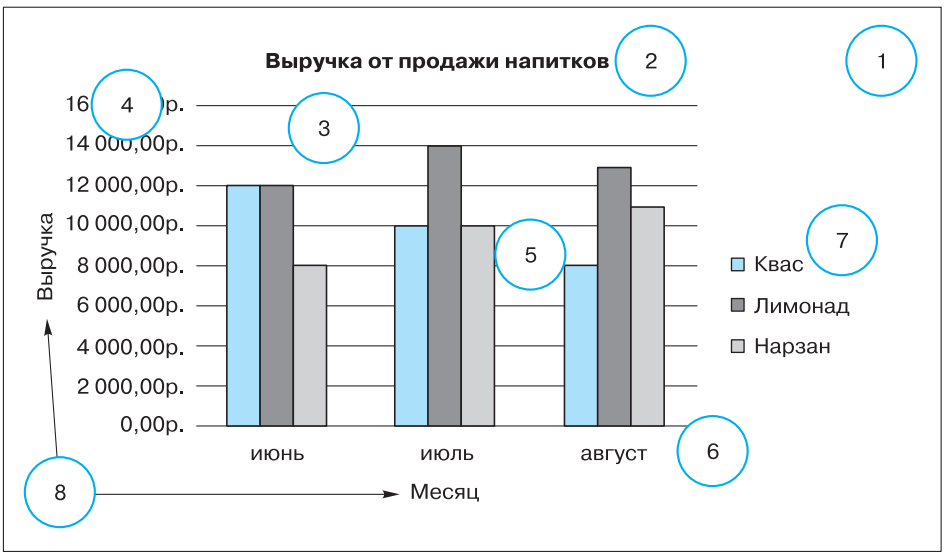


Рис. 1.9. Основные элементы диаграммы

На диаграммах разных типов числовые данные могут быть представлены точками, отрезками, прямоугольниками, секторами круга, прямоугольными параллелепипедами, цилиндрами, конусами и другими геометрическими фигурами. При этом размеры геометрических фигур или расстояния от них до осей пропорциональны числовым данным, которые они отображают.

Диаграммы, создаваемые в электронных таблицах, динамические — при редактировании данных в таблице размеры или количество фигур, обозначающих эти данные, автоматически изменяются.

Вспомните основные приёмы построения диаграмм, известные вам из курса информатики основной школы.

Рассмотрим самые распространённые типы диаграмм.

Гистограммы целесообразно создавать тогда, когда нужно сравнить значения нескольких наборов данных, графически изобразить отличия значений одних наборов данных от других, показать изменения данных с течением времени.

Различают следующие виды гистограмм:

- гистограмма с группировкой;
- гистограмма с накоплением;
- нормированная гистограмма с накоплением;
- объёмная гистограмма.

В гистограмме с группировкой прямоугольники, которые являются графическими изображениями числовых данных из разных наборов, располагаются рядом друг с другом (см. рис. 1.9). В гистограмме с накоплением прямоугольники, изображающие числовые данные, располагаются друг над другом (рис. 1.10). Это даёт возможность оценить суммарные данные и вклад каждой составляющей в общую сумму.

В нормированной гистограмме с накоплением вертикальная ось имеет шкалу в процентах. Это даёт возможность оценить долю (процентную часть) данных в общей сумме (рис. 1.11).

Подумайте, по какой из трёх диаграмм проще всего определить:

1) продажа каких напитков неуклонно возрастала;
2) продажа каких напитков принесла наибольшую прибыль в июле;
3) динамику изменений суммарной выручки от продажи всех трёх напитков;
4) вклад от продажи каждого напитка в общую выручку.

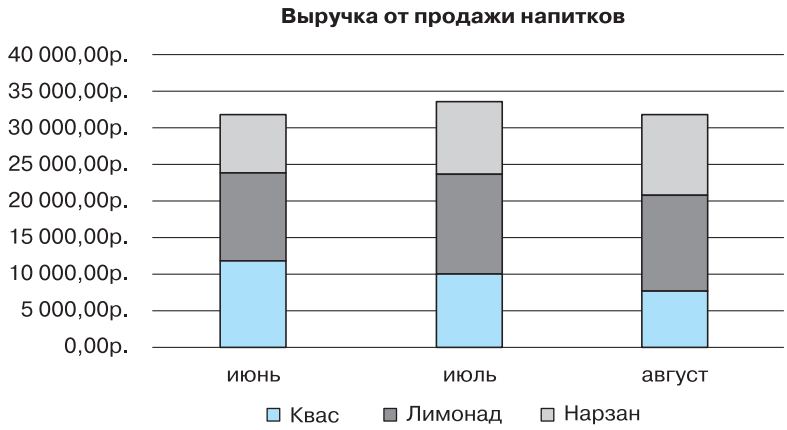


Рис. 1.10. Пример гистограммы с накоплением

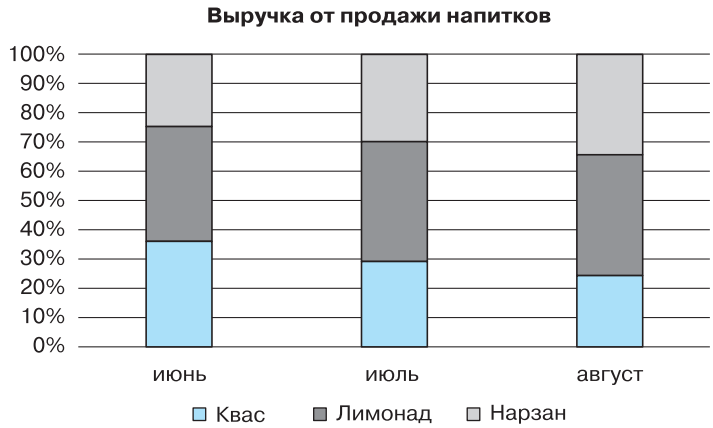


Рис. 1.11. Нормированная гистограмма с накоплением

Линейчатые диаграммы аналогичны гистограммам и отличаются от них лишь горизонтальным расположением геометрических фигур.

К типу диаграмм **Круговая** относятся плоские и объёмные круговые диаграммы. Их целесообразно использовать тогда, когда нужно отобразить части одного целого, сравнить соотношение частей между собой и отношение частей к целому.

Круговые диаграммы позволяют отобразить только один ряд данных. Они теряют наглядность, если содержат много элементов данных. Несколько круговых диаграмм можно заменить, например, одной нормированной гистограммой с накоплением.

Подумайте, сколько разных круговых диаграмм можно построить по информации, содержащейся в диаграмме, представленной на рисунке 1.9.

Сколько круговых диаграмм потребуется для того, чтобы изобразить информацию, представленную на гистограмме с накоплением (см. рис. 1.10)?

Диаграммы типа **График** целесообразно использовать, если количество данных в наборе достаточно большое, если нужно отобразить динамику изменения данных во времени, сравнить изменения нескольких рядов данных (рис. 1.12).

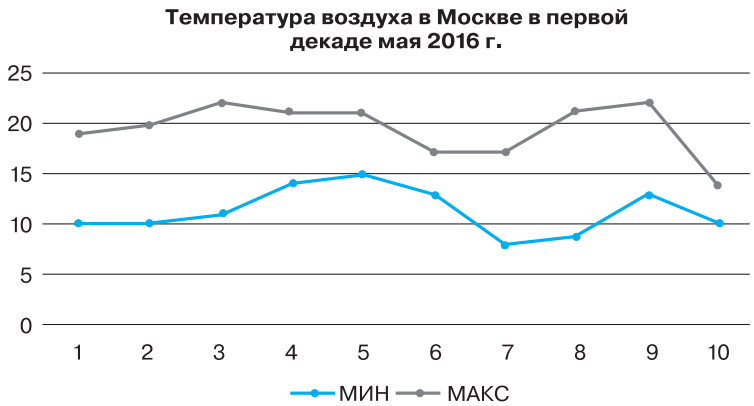


Рис. 1.12. Пример диаграммы **График с маркерами**

Точечные диаграммы с гладкими кривыми можно использовать для построения графиков функций, предварительно заполнив диапазон ячеек значениями аргумента и соответствующими значениями функции. Можно построить на одной диаграмме графики двух функций и использовать их для приближённого решения уравнения.

Пример. Найдём на отрезке [0; 1,2] корень уравнения $\cos(x) = \sqrt{x}$, построив в табличном процессоре графики функций, соответствующих левой и правой частям равенства. Для этого:

1) используя стандартные функции COS и КОРЕНЬ, построим таблицу значений функций для x, изменяющегося с шагом 0,1:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	x	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1	1,1	1,2
2	cos(x)	1	1	0,98	0,96	0,92	0,88	0,83	0,76	0,7	0,62	0,54	0,45	0,36
3	корень(x)	0	0,32	0,45	0,55	0,63	0,71	0,77	0,84	0,89	0,95	1	1,05	1,1

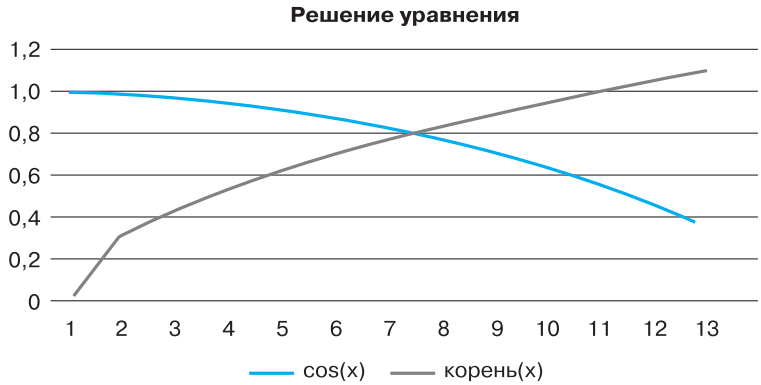


Рис. 1.13. Графики функций

2) по значениям диапазона B2:N3 построим графики функций COS(x) и КОРЕНЬ(x) (рис. 1.13);
3) заменим номера точек, проставленные по горизонтальной оси, на значения аргумента x рассматриваемых функций. Для этого вызовем контекстное меню горизонтальной оси и выберем пункт **Выбрать данные**. Появится окно **Выбор источника данных** (рис. 1.14).
В открывшемся окне нажмём на кнопку изменения подписей горизонтальной оси и выберем диапазон со значениями аргумента (рис. 1.15).

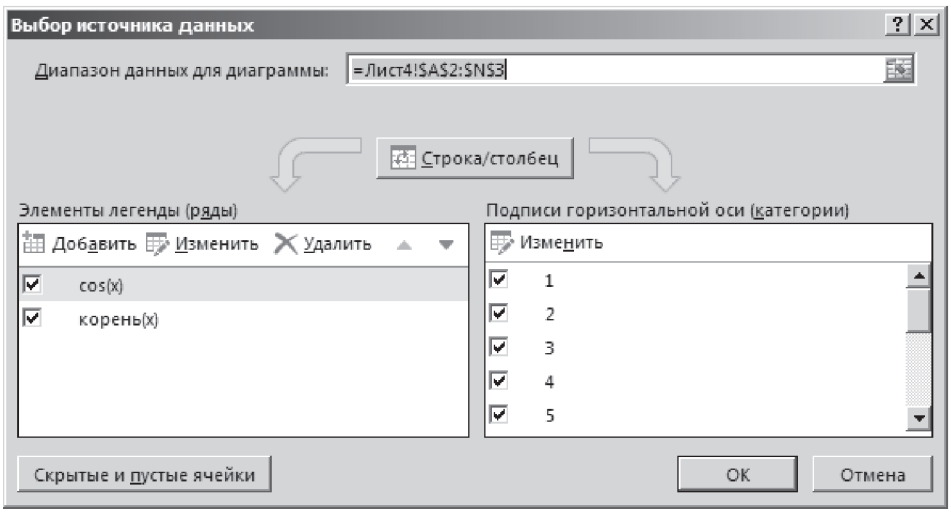


Рис. 1.14. Окно **Выбор источника данных**

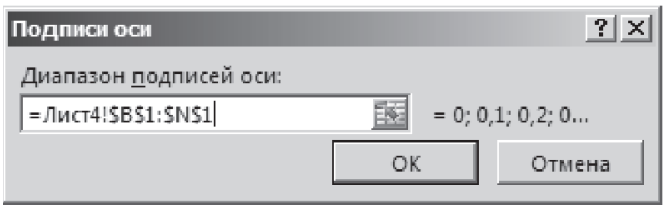


Рис. 1.15. Окно **Подписи оси**

После редактирования (совмещения) точки пересечения осей и добавления вертикальных линий сетки график приобретёт вид, представленный на рисунке 1.16.

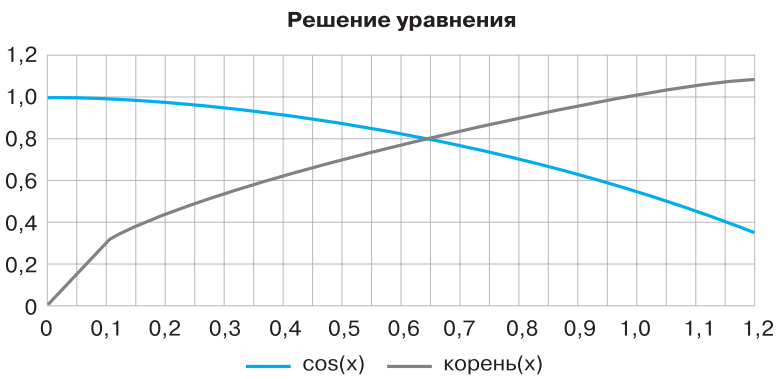


Рис. 1.16. График после редактирования

В результате построения графиков функций видно, что корень уравнения приблизительно равен 0,64.

Построенную диаграмму можно редактировать:

- изменять способ формирования ряда данных: из данных строки или из данных столбца;
- изменять диапазон ячеек, по которым строится диаграмма;
- изменять тип, вид или макет диаграммы;
- вставлять, перемещать, удалять или изменять название диаграммы, легенды, подписей данных;
- изменять отображение осей и линий сетки и др.

Построенную диаграмму можно форматировать. При этом можно применить стилевое форматирование сразу ко всей диаграмме, воспользовавшись одним из стилей оформления диаграмм. Кроме того, можно форматировать отдельные объекты диаграммы, которые предварительно надо выделить.

Некоторые объекты диаграммы, например ряд, состоят из нескольких частей. Чтобы выделить только одну часть, например отдельную точку ряда, необходимо сначала выделить весь объект, а затем выбрать нужную его часть.

4.2. Сортировка данных

Данные в электронной таблице можно сортировать, т. е. изменять порядок их расположения в строках или столбцах. В отсортированных данных легче найти необходимые значения, осуществить их анализ, выявить имеющиеся закономерности и др.

Сортировка — это упорядочение данных в таблице.

Сортировка данных может проводиться по возрастанию (от наименьшего к наибольшему) или по убыванию (от наибольшего к наименьшему). В Microsoft Excel соответствующие инструменты размещены на вкладке **Данные** в группе **Сортировка и фильтр**.

В Microsoft Excel сортировка данных по возрастанию заключается в следующем:

- символы упорядочиваются в порядке размещения их кодов в кодовой таблице Unicode;
- числа и даты упорядочиваются от наименьшего значения к наибольшему и располагаются перед текстовыми данными, причём сначала располагаются числа;
- текстовые данные сначала упорядочиваются по их первым символам; если первые символы в текстах совпали, то они упорядочиваются по вторым символам; тексты, в которых совпали первые два символа, упорядочиваются по их третьим символам и т. д.;
- логическое значение ЛОЖЬ размещается перед значением ИСТИНА;
- пустые ячейки всегда располагаются последними.

При сортировке данных по убыванию порядок расположения будет обратный, за исключением пустых ячеек, которые всегда располагаются последними.

Далее представлены введённые данные (диапазон A1:A15) и результаты их сортировки. Определите, где представлены данные, отсортированные по возрастанию, где — по убыванию. Объясните полученные результаты.

	A	B	C
1		-11	ИСТИНА
2	Петрозаводск	11	ЛОЖЬ
3		27.03.2017	Петрозаводск
4	11	25.02.2020	Петербург
5	11А	11А	Москва
6	Москва	Best	Best
7		Москва	11А
8	Петербург	Петербург	25.02.2020
9		Петрозаводск	27.03.2017
10	-11	ЛОЖЬ	11
11	ИСТИНА	ИСТИНА	-11
12	ЛОЖЬ		
13	25.02.2020		
14	27.03.2017		
15	Best		

Сортировка в выделенном связном диапазоне ячеек из нескольких столбцов выполняется по данным первого из выделенных столбцов. Иначе говоря, данные во всех других столбцах выделенного диапазона ячеек не сортируются, а расставляются по строкам электронной таблицы в соответствии с перестановкой данных первого столбца.

Если установить курсор в одну из ячеек связного диапазона и воспользоваться инструментом сортировки, то данные всего связного диапазона будут отсортированы в выбранном порядке по данным текущего столбца этого диапазона.

Если перед сортировкой данных выделить только часть связного диапазона, то при вызове инструмента сортировки откроется окно **Обнаружены данные вне указанного диапазона.** В этом окне при необходимости можно расширить выбранный диапазон ячеек до всего связного диапазона или подтвердить то, что сортировать данные следует только в пределах выделения.

В произвольном выделенном диапазоне ячеек отсортировать данные можно по значениям не одного, а нескольких столбцов (по нескольким уровням сортировки). Сортировка данных по значениям нескольких столбцов выполняется так:

- сначала данные сортируются по значениям первого из выбранных столбцов;
- сортировка данных по значениям каждого следующего из выбранных столбцов происходит лишь для тех строк элек-

тронной таблицы, в которых значения во всех предыдущих выбранных для сортировки столбцах совпадают.

Уровни такой сортировки и её порядок задаются в окне **Сортировка (Данные → Сортировка и фильтр → Сортировка)**. На рисунке 1.17 приведены введённые данные и результат их сортировки по значениям двух столбцов, а соответствующие настройки окна **Сортировка** приведены на рисунке 1.18.

В окне **Сортировка** можно:

- указать требуемые уровни сортировки;
- удалить любой уровень сортировки из списка для сортировки;
- переместить любой уровень сортировки выше или ниже в списке для сортировки;
- задать требуемые режимы сортировки.

В этом же окне можно установить параметры сортировки (кнопка **Параметры...**): сортировать по столбцам (по умолчанию) или по строкам; сортировать без учёта регистра (по умолчанию) или с учётом регистра.

	A	B	C	D	E	F	G
1	**Фамилия**	**Имя**	**Класс**		**Фамилия**	**Имя**	**Класс**
2	Васечкин	Дима	10		Васечкин	Юра	5
3	Иванов	Саша	9		Иванов	Петя	5
4	Петров	Коля	11		Сидоров	Денис	5
5	Сидоров	Денис	10		Петров	Руслан	6
6	Васечкин	Андрей	8		Васечкин	Андрей	8
7	Иванов	Андрей	11		Иванов	Саша	9
8	Петров	Руслан	6		Васечкин	Дима	10
9	Сидоров	Денис	5		Иванова	Ирина	10
10	Васечкин	Юра	5		Сидоров	Денис	10
11	Иванов	Петя	5		Иванов	Андрей	11
12	Иванова	Ирина	10		Петров	Коля	11

Рис. 1.17. Пример сортировки по значениям двух столбцов

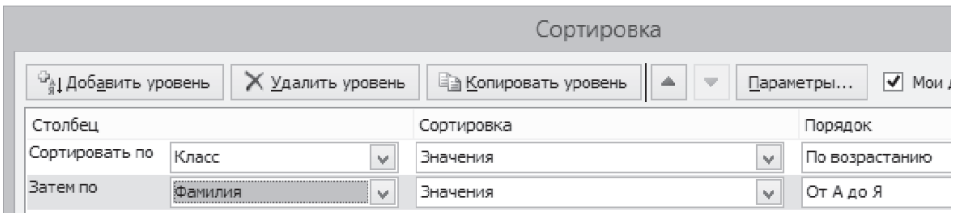


Рис. 1.18. Фрагмент окна **Сортировка**

4.3. Фильтрация данных

Ещё одним инструментом эффективного анализа данных в электронных таблицах является фильтрация.

Фильтрация — это выбор в электронной таблице данных, соответствующих определённым условиям.

Операция фильтрации, в отличие от операции сортировки, не меняет порядок строк. В отфильтрованном списке отображаются только строки, отвечающие условиям отбора данных, а остальные строки временно скрываются.

Если установить табличный курсор в произвольную ячейку заполненного данными диапазона (причём некоторые ячейки этого диапазона могут быть пустыми) и вызвать инструмент **Фильтр (Данные → Сортировка и фильтр → Фильтр)**, то около правой границы каждой ячейки первой строки этого диапазона появятся кнопки открытия списков, в которых находятся:

- команды сортировки данных по значениям данного столбца;
- команда **Фильтр по цвету**;
- команда **Снять фильтр с**;
- команда открытия меню команд для установки условий фильтрации:
  - — числовые фильтры (если в столбце числовые данные);
  - — текстовые фильтры (если в столбце текстовые данные);
  - — фильтры по дате (если в столбце даты).

	A	B	C
1	Фамил	Имя	Класс
4	Петров	Коля	11
7	Иванов	Андрей	11

Рис. 1.19. Пример фильтрации с помощью числового фильтра

Результаты выбора с помощью фильтрации из списка (см. рис. 1.17) всех учеников 11 класса представлены на рисунке 1.19.

Отфильтрованную таблицу можно редактировать, форматировать, выводить на печать. Для неё можно создавать диаграммы, не изменяя порядок строк и не перемещая их.

4.4. Условное форматирование

Ещё одним способом выбора в таблице данных, удовлетворяющих определённым условиям, является так называемое условное форматирование.

Условное форматирование автоматически изменяет формат ячейки на заданный, если для значения в данной ячейке выполняется определённое условие.

Для установки условного форматирования необходимо:

1) выделить нужный диапазон ячеек;
2) вызвать инструмент **Условное форматирование (Главная → Стили → Условное форматирование);**
3) выбрать в списке кнопки **Условное форматирование** необходимый тип правил (Правила выделения ячеек, Правила отбора первых и последних значений, Гистограммы, Цветовые шкалы, Наборы значков);
4) в списке правил указанного типа выбрать нужное правило;
5) в открывшемся окне задать условие и выбрать из списка форматов тот, который будет установлен;
6) завершить операцию щелчком по кнопке **ОК.**

Результат применения операции условного форматирования к диапазону C2:C12 списка учеников (см. рис. 1.17) представлен на рисунке 1.20 (тип правил — **Правила выделения ячеек**; условие выбора — **Больше 8**; формат — **Зелёная заливка и тёмно-зелёный текст** (рис. 1.21)).

	A	B	C
1	**Фамилия**	**Имя**	**Класс**
2	Васечкин	Дима	10
3	Иванов	Саша	9
4	Петров	Коля	11
5	Сидоров	Денис	10
6	Васечкин	Андрей	8
7	Иванов	Андрей	11
8	Петров	Руслан	6
9	Сидоров	Денис	5
10	Васечкин	Юра	5
11	Иванов	Петя	5
12	Иванова	Ирина	10

Рис. 1.20. Пример условного форматирования

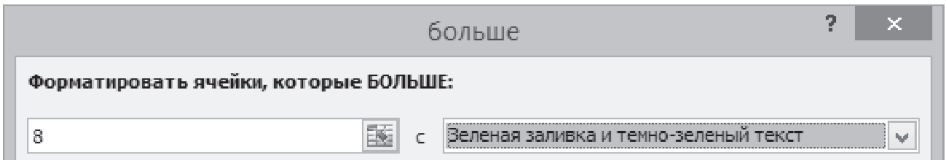


Рис. 1.21. Пример условного форматирования

В отличие от фильтрации условное форматирование не скрывает ячейки со значениями, не удовлетворяющими определённому условию, а лишь выделяет заданным образом те ячейки, для значений которых условие выполняется.

4.5. Подбор параметра

Если известны параметры (исходные данные) и формула, по которой они должны быть преобразованы, то пользователь вводит их в ячейки электронной таблицы и получает некоторый результат. В электронных таблицах есть и обратная возможность: подобрать такие параметры, которые при подстановке их в известную формулу будут приводить к желаемому заранее известному результату. В Microsoft Excel это можно сделать с помощью одной из функций специального инструмента **Анализ «что-если»**.

Рассмотрим эту возможность на примере решения квадратного уравнения $x^2 - 5x + 6 = 0$.

Нам известна формула для вычислений ($x^2 - 5x + 6$) и желаемый результат (0). Внесём эту информацию в ячейки таблицы:

	A	B
1	x	0
2	F(x)	=B1^2-5*B1+6

При подборе параметра в Microsoft Excel используется итерационный (циклический) процесс. Количество итераций и точность вычислений можно установить в окне **Параметры Excel (Файл → Параметры → Формулы → Параметры вычислений)**.

Вызовем окно подбора параметра (**Данные → Анализ «что-если» → Подбор параметра**) и заполним в нём поля ввода (рис. 1.22).

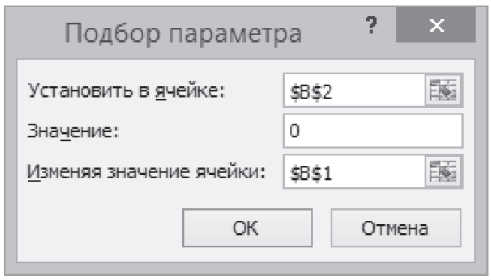


Рис. 1.22. Окно **Подбор параметра**

Ниже представлен результат подбора параметра:

	A	B
1	x	2
2	F(x)	0

Получен один из двух корней квадратного уравнения. Инструмент подбора параметра устроен так, что он возвращает одно решение, причём то, которое ближе к начальному значению. Напомним, что мы в качестве начального значения параметра приняли $x = 0$.

Самостоятельно поэкспериментируйте с другими начальными значениями параметра и найдите второй корень квадратного уравнения.

САМОЕ ГЛАВНОЕ

Проводить анализ большого количества числовых данных значительно легче, если изобразить их графически (визуализировать). Для графического представления числовых данных используются диаграммы.

Табличные процессоры позволяют строить гистограммы, линейчатые диаграммы, круговые диаграммы, графики, диаграммы с областями, поверхностные диаграммы, лепестковые диаграммы и др.

В диаграмме любого типа можно выделить следующие объекты: область диаграммы, название диаграммы, область построения диаграммы, ось категорий, ось значений, названия осей, ряды данных, легенду.

Построенную диаграмму можно редактировать и форматировать.

Диаграммы, создаваемые в электронных таблицах, динамические — при редактировании данных в таблице размеры или количество фигур, обозначающих эти данные, автоматически изменяются.

Данные в электронной таблице можно сортировать, т. е. изменять порядок их расположения в строках или столбцах. В отсортированных данных легче найти необходимые значения, осуществить их анализ, выявить имеющиеся закономерности и др.

Ещё одним инструментом эффективного анализа данных в электронных таблицах является фильтрация, позволяющая из многочисленных данных отобрать только те, которые соответствуют заданным условиям. Операция фильтрации, в отличие от операции сортировки, не меняет порядок строк. В отфильтрованном списке отображаются только строки, отвечающие условиям отбора данных, а остальные строки временно скрываются.

Условное форматирование автоматически изменяет формат ячейки на заданный, если для значения в данной ячейке выполняется определённое условие. В отличие от фильтрации условное форматирование не скрывает ячейки со значениями, не удовлетворяющими определённому условию, а лишь выделяет заданным образом те ячейки, для значений которых условие выполняется.

Для анализа данных может быть полезна имеющаяся в электронных таблицах возможность подобрать такие параметры, которые при подстановке их в известную формулу будут приводить к желаемому заранее известному результату.

Вопросы и задания

1. Для чего предназначены диаграммы? Какой анализ числовых данных можно выполнить с их помощью?
2. Назовите основные типы диаграмм, которые могут быть построены в электронных таблицах.
3. Назовите основные объекты диаграмм и их свойства.
4. Опишите виды гистограмм. Для чего предназначен каждый из этих видов?
5. Для чего предназначены круговые диаграммы?
6. Для чего предназначены графики?
7. Перечислите основные операции редактирования диаграмм.

8. Перечислите основные операции форматирования диаграмм.
9. По представленной ниже информации составьте таблицу распределения суши и воды на поверхности земного шара.

   Площадь поверхности Земли — 510 072 тыс. кв. км, в том числе площадь суши — 148 940 тыс. кв. км (29,2%), площадь водной поверхности — 361 132 тыс. кв. км (70,8%). При этом суша большей частью лежит в Северном полушарии, а водная поверхность — наоборот. В Северном полушарии водная поверхность занимает 61%, а поверхность суши — 39%; для Южного полушария эти соотношения таковы: 81% воды и 19% суши.

   По данным полученной таблицы постройте следующие диаграммы:
   1) гистограмму с группировкой;
   2) гистограмму с накоплением;
   3) нормированную гистограмму с накоплением;
   4) объёмную гистограмму с накоплением;
   5) круговую;
   6) линейчатую с группировкой.
10. Дан фрагмент электронной таблицы:

	A	B	C
1	2	1	
2	=C1-B1*4	=(B1+C1)/A1	=C1-4

   Какое целое число должно быть записано в ячейке C1, чтобы после выполнения вычислений диаграмма, построенная по значениям диапазона ячеек A2:C2, соответствовала рисунку?
11. Можно ли построить круговые диаграммы для данных, содержащих отрицательные числа? Подкрепите свой ответ примерами.
12. В табличном процессоре постройте график функции $y = \frac{1}{x^2+1}$ на отрезке $[-2;\ 2]$ с шагом 0,2.
13. В табличном процессоре на одной диаграмме постройте графики трёх функций $y = \sin x$, $y = 2\sin x$, $y = \sin 2x$ на отрезке $[-2\pi;\ 2\pi]$ с шагом $\frac{\pi}{8}$.

14. На интервале [–1; 1] с шагом 0,1 решите графически систему уравнений:

$$\begin{cases} y = 2x + 7; \\ y = 2x^2 + 9. \end{cases}$$

15. Что называют сортировкой? Для чего она используется?
16. Сформулируйте правила, определяющие порядок сортировки данных разных типов по убыванию.
17. Какой порядок сортировки можно задать для числовых данных? Для текстовых данных?
18. Что называют фильтрацией? Для чего она используется?
19. Сравните операции сортировки и фильтрации. Что у них общего? Чем они различаются?
20. Используя возможность подбора параметра, решите квадратное уравнение $x^2 + 2x - 15 = 0$.

Дополнительные материалы к главе смотрите в авторской мастерской.

Глава 2
АЛГОРИТМЫ И ЭЛЕМЕНТЫ ПРОГРАММИРОВАНИЯ

Предположим, перед вами поставлена задача, для решения которой необходимо написать программу. Из курса информатики основной школы вам известно, что решение задачи имеет определённые этапы.

1. **Постановка задачи**. Необходимо определить исходные данные (что подаётся «на вход») и требуемые результаты (что следует получить «на выходе»), указать связи между исходными данными и результатами.
2. **Формализация**. На этом этапе определяется класс, к которому принадлежит задача, связи между исходными данными и результатами записываются с помощью математических соотношений.
3. **Выбор метода решения и разработка алгоритма**. Это один из важнейших этапов решения задачи; от правильности выбора метода зависит эффективность алгоритма, определяющая быстродействие программы и размер необходимой для её выполнения памяти.
4. **Составление программы** (запись алгоритма на языке программирования) и её ввод в память компьютера. Сейчас этот этап принято называть кодированием.
5. **Отладка программы**. Созданная программа может содержать ошибки, допущенные как в процессе кодирования, так и на любом из предыдущих этапов. Ошибки могут быть выявлены в процессе тестирования — выполнения программы с заранее подготовленными наборами исходных данных, для которых известен результат.
6. **Вычисление и обработка результатов**.

Задачи, которые мы будем рассматривать в данной главе, достаточно чётко поставлены и формализованы. Основное внимание при их решении мы будем уделять этапам 3–6.

§ 5
Основные сведения об алгоритмах

5.1. Понятие алгоритма. Свойства алгоритма

Каждый из нас ежедневно решает задачи различной сложности: как быстрее добраться в школу или на работу в условиях недостатка времени, в каком порядке выполнить дела, намеченные на текущий день, и т. д. Некоторые задачи настолько сложны, что требуют длительных размышлений для нахождения решения (которое иногда так и не удаётся найти). Другие задачи мы решаем автоматически, т. к. выполняем их ежедневно на протяжении многих лет (выключить звенящий будильник; почистить утром зубы; вскипятить воду в чайнике; позвонить другу по телефону; открыть или закрыть входную дверь ключом). В большинстве случаев в решении каждой задачи можно выделить отдельные шаги.

Пример 1. Решение задачи «Закрыть входную дверь ключом» предполагает выполнение следующих шагов.

1. Вставить ключ в замочную скважину.
2. Повернуть ключ несколько раз на 180 градусов против(по) часовой стрелки(е).
3. Вынуть ключ из замочной скважины.

Последовательность шагов, приведённая в примере 1, является алгоритмом решения задачи «Закрыть входную дверь ключом». Исполнитель этого алгоритма — человек. Объекты этого алгоритма — ключ, дверь.

Для решения любой задачи надо знать, что дано и что следует получить, т. е. у задачи есть исходные данные (объекты) и искомый результат. Для получения результатов необходимо знать способ решения задачи — располагать алгоритмом.

Из курса информатики основной школы вам известны понятия алгоритма и исполнителя.

Исполнитель алгоритма — это субъект или устройство, способные правильно интерпретировать описание алгоритма и выполнить содержащийся в нём перечень действий.

Исполнители бывают двух типов: формальные и неформальные. Далее мы будем говорить преимущественно о формальных или «неразмышляющих» исполнителях. Формальный исполнитель не размышляет над выполняемыми командами, а строго следует пошаговым инструкциям алгоритма. Одну и ту же команду формальный исполнитель всегда выполняет одинаково. За действия формального исполнителя отвечает управляющий им объект.

Алгоритм — это точная конечная система предписаний, определяющая содержание и порядок действий исполнителя над некоторыми объектами (исходными и промежуточными данными) для получения (после конечного числа шагов) искомого результата.

Приведённое определение не является определением в строгом смысле этого слова, это — описание понятия алгоритма, раскрывающее его сущность. Оно не является формальным, потому что в нём используются такие неуточняемые понятия, как «система предписаний», «действия исполнителя», «объект».

Понятие алгоритма, являющееся фундаментальным понятием математики и информатики, возникло задолго до появления вычислительных машин.

Первоначально под словом «алгоритм» понимали способ выполнения арифметических действий над десятичными числами. В дальнейшем это понятие стали использовать для обозначения любой последовательности действий, приводящей к решению поставленной задачи.

Пример 2. Простые числа — это натуральные числа, бóльшие единицы, которые имеют только два делителя: единицу и само это число (рис. 2.1).

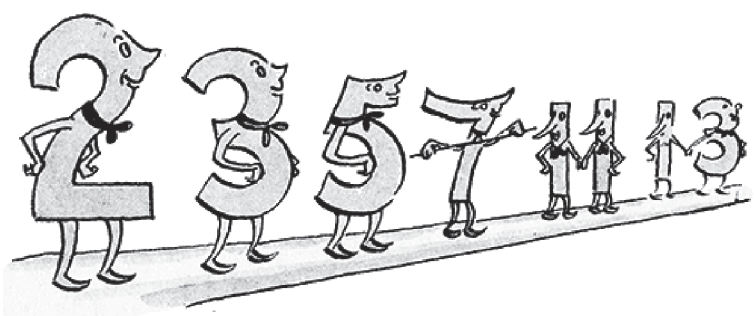


Рис. 2.1. Примеры простых чисел (иллюстрация Н. П. Антокольской)

Метод нахождения простых чисел в интервале [2; n] предложил греческий математик Эратосфен (275–194 гг. до н. э.). По одной из версий, он выписал все числа от 2 до 10 000 на папирусе, натянутом на рамку, и прокалывал составные числа. Папирус стал подобен решету, которое «просеивало» составные числа, а простые оставляло. Поэтому метод Эратосфена называют ещё Эратосфеновым решетом.

Во все времена люди хотели найти как можно большее простое число.

Пока люди считали только при помощи карандаша и бумаги, им нечасто удавалось обнаружить новые простые числа. До 1952 г. самое большое известное простое число состояло из 39 цифр.

В 2013 г. открыто простое число, запись которого в десятичной системе счисления состоит из 17 425 170 знаков. На проверку простоты нового числа ушло 39 дней работы персонального компьютера в Университете Центрального Миссури, США.

Для нахождения всех простых чисел не больше заданного числа n, следуя методу Эратосфена, нужно выполнить следующие шаги:

1) выписать подряд все целые числа от 2 до n (2, 3, 4, ..., n);
2) присвоить переменной p значение 2 (2 — первое простое число);
3) зачеркнуть в списке числа, кратные p: $2p$, $3p$, $4p$, ...;
4) найти первое незачёркнутое число в списке, большее чем p, и присвоить переменной p соответствующее значение;
5) повторять шаги 3 и 4, пока возможно.

Числа, остающиеся незачёркнутыми после завершения работы алгоритма, и есть все простые числа от 2 до n.

На практике, алгоритм можно улучшить следующим образом. На шаге 3 числа можно зачёркивать начиная сразу с числа p^2 ($p \cdot p$), потому что все составные числа меньше него к этому времени уже будут зачёркнуты. И соответственно, останавливать алгоритм можно, когда p^2 станет больше, чем n.

Любой алгоритм существует не сам по себе, он всегда предназначен для определённого исполнителя. Алгоритм описывается в командах исполнителя, который этот алгоритм будет выполнять. Объекты, над которыми исполнитель может совершать действия, образуют так называемую среду исполнителя. Исходные данные и результаты любого алгоритма всегда принадлежат среде того исполнителя, для которого предназначен алгоритм.

Значение слова «алгоритм» очень похоже по значению на слова «рецепт», «метод», «способ». Но, в отличие от рецепта или способа, любой алгоритм обязательно обладает следующими свойствами.

1. **Дискретность.** Выполнение алгоритма разбивается на последовательность законченных действий-шагов. Только выполнив одно действие, можно приступать к выполнению следующего. Произвести каждое отдельное действие исполнителю предписывает специальное указание в записи алгоритма, называемое командой.
2. **Детерминированность.** Каждая команда алгоритма определяет однозначное действие исполнителя и недвусмысленно указывает, какая команда должна выполняться следующей. Если алгоритм многократно применяется к одному и тому же набору входных данных, то каждый раз получаются одинаковые промежуточные и выходной результаты.
3. **Понятность.** Алгоритм не должен содержать предписаний, смысл которых может восприниматься исполнителем неоднозначно, т. е. запись алгоритма должна быть настолько чёткой и полной, чтобы у исполнителя не возникло потребности в принятии каких-либо самостоятельных решений. Стоит помнить, что алгоритм всегда рассчитан на выполнение «неразмышляющим» исполнителем.
4. **Результативность.** Под этим свойством понимается содержательная определённость результата каждого шага и алгоритма в целом. При точном исполнении команд алгоритма процесс должен прекратиться за конечное число шагов, и при этом должен быть получен ответ на вопрос задачи. В качестве одного из возможных ответов может быть и установление того факта, что задача решений не имеет. Свойство результативности содержит в себе свойство **конечности** — завершение работы алгоритма за конечное число шагов.
5. **Массовость.** Алгоритм пригоден для решения любой задачи из некоторого класса задач, т. е. алгоритм правильно работает на некотором множестве исходных данных, которое называется областью применимости алгоритма.

Докажите, что рассмотренный в примере 2 метод Эратосфена является алгоритмом.

Не любой расписанный по шагам способ решения задачи является алгоритмом. Убедимся в этом на примере.

Пример 3. Опишем метод построения перпендикуляра к прямой MN, проходящего через заданную точку A этой прямой, с помощью линейки и циркуля (рис. 2.2):

1) отложить в обе стороны от точки A на прямой MN циркулем отрезки равной длины с концами B и C;
2) увеличить раствор циркуля до радиуса, в полтора-два раза больше длины отрезков AB и AC;
3) провести указанным раствором циркуля дуги окружностей с центрами в точках B и C так, чтобы они охватили точку A и образовали две точки пересечения друг с другом (D и E);
4) взять линейку, приложить её к точкам D и E и соединить их отрезком; при правильном построении отрезок пройдёт через точку A и будет являться перпендикуляром к прямой MN.

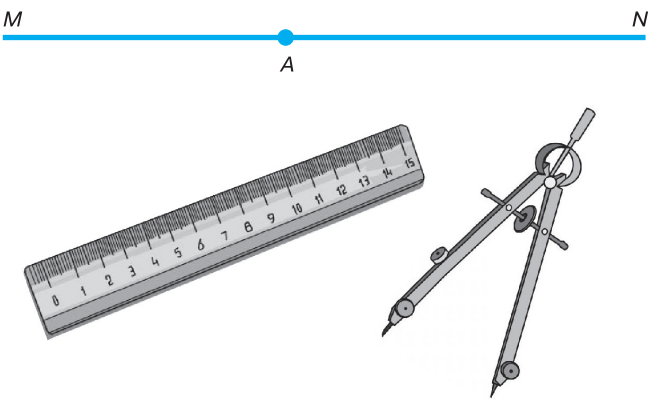


Рис. 2.2. Построение перпендикуляра к прямой

Почему этот способ построения перпендикуляра к прямой не является алгоритмом? Какое свойство алгоритмов здесь нарушено?

Есть задачи, которые человек успешно решает, но при этом не может представить процесс их решения в виде последовательности шагов.

Например, если перед человеком положить фотографии собак и кошек и попросить определить, на каких фотографиях изображены собаки, а на каких кошки, то человек практически мгновенно и безошибочно решит эту задачу. Написать же формальный алгоритм решения этой и подобных ей задач пока что не удалось, хотя определённые успехи в этой области, называемой теорией распознавания образа, уже достигнуты.

В супермаркетах вы видите системы распознавания штрих-кодов, используете системы оптического распознавания символов, знаете о распознавании автомобильных номеров, слышали о распознавании изображений лиц, речи и т. д.

Технические системы распознавания образов находят сегодня широкое применение в самых разных областях — от военного дела и систем безопасности до оцифровки аналоговых сигналов. Их разработка базируется на теории распознавания образов — разделе информатики и смежных дисциплин, развивающем основы и методы классификации и идентификации предметов, явлений, процессов, сигналов, ситуаций и тому подобных объектов, характеризующихся конечным набором некоторых свойств и признаков.

С учётом рассмотренных свойств алгоритма можно дать более формальное определение этого понятия, которое также не является строгим, т. к. в нём всё ещё используются не определяемые точно термины.

Алгоритм — это конечная система правил, сформулированных на языке исполнителя, которая определяет последовательность перехода от допустимых исходных данных к конечному результату и обладает свойствами дискретности, детерминированности, понятности, результативности, конечности и массовости.

Математики достаточно долго пользовались интуитивным (нестрогим) понятием алгоритма, записывая алгоритмы примерно так, как в примере 3. Ими были сформулированы многие успешно применяемые на практике алгоритмы решения таких задач, как выполнение арифметических действий «столбиком», нахождение корней квадратных и кубических уравнений, решение систем линейных уравнений и др. Постепенно математики подходили к постановке и решению всё более сложных задач, требовавших построения формального определения алгоритма.

Попытки разработать формальное определение алгоритма привели в 20–30-х годах XX века к возникновению теории алгоритмов — науки, изучающей общие свойства и закономерности алгоритмов и разнообразные формальные модели их представления. Вместе с математической логикой теория алгоритмов образует теоретическую основу вычислительных наук.

Математики (А. Тьюринг, Э. Пост, А. Н. Колмогоров, А. А. Марков и др.) предложили несколько подходов к формальному определению алгоритма: нормальный алгоритм Маркова, машина Тьюринга, машина Поста и т. д. Дальнейшее показало, что все эти определения эквивалентны. На основании этих определений учёные пришли к выводу о существовании алгоритмически неразрешимых задач — задач, для которых невозможно построить процедуру решения.

5.2. Способы записи алгоритма

Из курса информатики основной школы вам известны разные способы записи одного и того же алгоритма:

- словесная запись алгоритма на естественном языке;
- запись алгоритма псевдокодом — частично формализованным естественным языком с элементами языка программирования и общепринятыми математическими обозначениями;
- запись алгоритма в виде блок-схемы — подробного графического представления логической структуры программы или алгоритма с помощью стандартных блоков (условных символов), соединённых линиями;
- запись алгоритма на языке программирования и т. д.

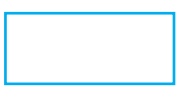
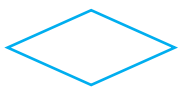
Выбор способа записи алгоритма зависит от ряда причин. Небольшой алгоритм можно записать и в словесной форме. Если для вас наиболее важна наглядность, то разумно использовать блок-схему. Алгоритм же, готовый к реализации, должен быть записан на языке, понятном исполнителю.

Правила выполнения блок-схем, внешний вид графических блоков и их назначение определяются стандартом ГОСТ 19.701–90 (ИСО 5807–85) «Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения».

Напомним основные условные графические обозначения (символы), которые мы будем использовать в дальнейшем (табл. 2.1).

Таблица 2.1

Основные символы блок-схем и отображаемые ими функции

Символ	Функция
	Пуск/останов. Начало, конец, прерывание процесса обработки данных или выполнения программы
	Ввод/вывод. Преобразование данных в форму, пригодную для обработки (ввод) или отображения результатов обработки (вывод)
	Процесс. Выполнение операций или группы операций, в результате которых изменяется значение, форма представления или расположение данных
	Решение. Выбор направления выполнения алгоритма или программы в зависимости от некоторых переменных условий

Окончание табл. 2.1

Символ	Функция
	Модификация. Выполнение операций, меняющих команды или группу команд, изменяющих программу
	Предопределённый процесс. Использование ранее созданных и отдельно описанных алгоритмов или программ

Пример 4. Вспомним детскую игру «Угадай-ка». Первый игрок задумывает целое число и сообщает второму игроку, из какого оно диапазона. Второй игрок должен как можно быстрее угадать загаданное число. Второй игрок может называть числа, а первый должен говорить, меньше или больше названное число задуманного. Игра заканчивается, если названо число, равное задуманному.

Пусть задумано число $X \in [A, B]$. Представим в виде блок-схемы алгоритм решения этой задачи — известный вам метод половинного деления (рис. 2.3).

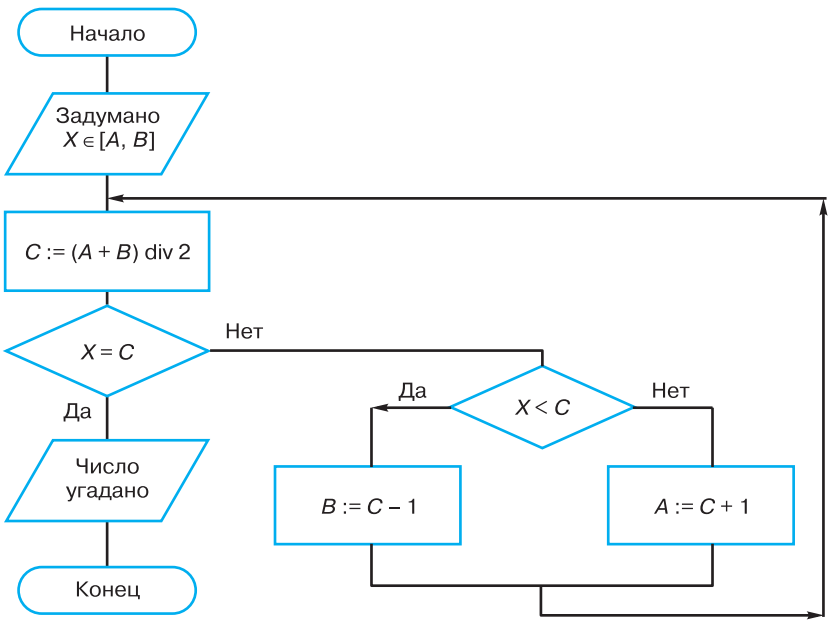


Рис. 2.3. Метод половинного деления

Подсчитайте, какое наибольшее число шагов может понадобиться для угадывания по этому алгоритму числа $X \in [0, 100]$.

5.3. Понятие сложности алгоритма

В теории алгоритмов установлено, что для задачи, имеющей алгоритмическое решение, можно придумать множество различных способов её решения, т. е. алгоритмов.

Какой же алгоритм лучше подходит для решения конкретной задачи? По каким критериям его следует выбирать из множества возможных?

Человек может назвать алгоритм сложным и запутанным из-за того, что тот обладает разветвлённой логической структурой, содержащей много проверок условий и переходов. Однако для компьютера выполнение программы, реализующей такой алгоритм, не составит труда, т. к. он выполняет одну команду за другой, и для компьютера неважно — операция ли это умножения или проверка условия.

Теория алгоритмов предоставляет аппарат анализа различных алгоритмов решения одной и той же задачи, на основе которого можно выбрать самый эффективный (наилучший) алгоритм.

Вычислительным процессом, порождённым алгоритмом, называется последовательность шагов алгоритма, пройденных при его исполнении.

Сложность алгоритма — количество элементарных шагов (действий) в вычислительном процессе этого алгоритма.

Обратите внимание, в определении сложности алгоритма речь идёт именно о вычислительном процессе, а не о самом алгоритме. Алгоритм состоит из команд. Команда — это отдельная инструкция в описании алгоритма. Шаг алгоритма — это отдельное действие, которое исполнитель выполняет по команде. В циклических алгоритмах за счёт повторного выполнения одних и тех же команд число шагов при выполнении алгоритма может быть значительно больше числа команд в алгоритме.

Очевидно, что в наибольшей степени число операций при выполнении алгоритма зависит от количества обрабатываемых данных. Действительно, для упорядочивания по алфавиту списка из 100 фамилий требуется существенно меньше операций, чем для упорядочивания списка из 100 000 фамилий. Поэтому сложность алгоритма выражают в виде функции от объёма входных данных.

Так, например, алгоритм, выполняющий только операции чтения данных и занесения их в оперативную память, имеет линейную сложность $O(n)$ (читается «о большое от эн»). Существуют алгоритмы, имеющие квадратичную и кубическую сложности.

Для решения задачи могут быть разработаны алгоритмы, имеющие разную сложность. Лучшим среди них считается алгоритм, имеющий наименьшую сложность.

Наряду со сложностью важной характеристикой алгоритма является эффективность. Эффективность оценивается количеством элементарных операций, которые необходимо выполнить для решения задачи, а также количеством памяти, требующейся для выполнения алгоритма.

Пример 5. Известно, что во многих языках программирования нет операции возведения в степень, и поэтому такой алгоритм программисту надо писать самостоятельно. Операция возведения в степень реализуется через операции умножения. С ростом показателя степени растёт количество операций умножения, которые выполняются достаточно долго. Следовательно, актуален вопрос о создании эффективного алгоритма возведения в степень.

Рассмотрим метод быстрого вычисления натуральной степени n вещественного числа x, описанный в Древней Индии ещё до нашей эры.

1. Запишем n в двоичной системе счисления.
2. Заменим в этой записи каждую единицу парой букв КХ, а каждый ноль — буквой К.
3. Вычеркнем крайнюю левую пару КХ.
4. Полученная строка, читаемая слева направо, даёт правило быстрого вычисления x^n, если букву К рассматривать как операцию возведения результата в квадрат, а букву Х — как операцию умножения результата на x. Вначале результат равен x.

Воспользуемся этим алгоритмом для того, чтобы возвести x в степень $n = 100$.

1. $100 = 1100100_2$.
2. Строим последовательность: КХКХКККХКК.
3. Вычёркиваем крайнюю левую пару КХ: КХККККХКК.
4. Вычисляем искомое значение:
   К: возвести x в квадрат (x^2);
   Х: умножить результат на x (x^3);
   К: возвести результат в квадрат (x^6);

К: возвести результат в квадрат (x^{12});
К: возвести результат в квадрат (x^{24});
Х: умножить результат на x (x^{25});
К: возвести результат в квадрат (x^{50});
К: возвести результат в квадрат (x^{100}).

Мы вычислили сотую степень числа x за 8 умножений. Это значительно эффективнее «прямолинейного» алгоритма возведения в степень, требующего 99 операций умножения.

САМОЕ ГЛАВНОЕ

Алгоритм — это конечная система правил, сформулированных на языке исполнителя, которая определяет последовательность перехода от допустимых исходных данных к конечному результату и обладает свойствами дискретности, детерминированности, понятности, результативности, конечности и массовости.

Исполнитель алгоритма — это субъект или устройство, способные правильно интерпретировать описание алгоритма и выполнить содержащийся в нём перечень действий.

Один и тот же алгоритм может быть записан разными способами: на естественном языке, псевдокодом, с помощью блок-схем, на языке программирования и т. д.

Для задачи, имеющей алгоритмическое решение, можно придумать множество различных способов её решения, т. е. алгоритмов. Теория алгоритмов предоставляет аппарат анализа различных алгоритмов решения одной и той же задачи, на основе которого можно выбрать самый эффективный (наилучший) алгоритм.

Алгоритм состоит из команд. Команда — это отдельная инструкция в описании алгоритма. Шаг алгоритма — это отдельное действие, которое исполнитель выполняет по команде. Вычислительным процессом, порождённым алгоритмом, называется последовательность шагов алгоритма, пройденных при его исполнении.

Сложность алгоритма — количество элементарных шагов (действий) в вычислительном процессе этого алгоритма. Наряду со сложностью важной характеристикой алгоритма является эффективность. Эффективность оценивается количеством элементарных операций, которые необходимо выполнить для решения задачи, а также количеством памяти, требующейся для выполнения алгоритма.

Вопросы и задания

1. Перечислите основные свойства алгоритмов и проиллюстрируйте их примерами.
2. Почему кулинарный рецепт приготовления торта нельзя считать алгоритмом? Какими свойствами алгоритма он не обладает?
3. Переформулируйте описание способа проведения перпендикуляра к прямой в заданной точке так, чтобы оно стало алгоритмом.
4. Есть двое песочных часов: на 3 и на 8 минут. Для приготовления эликсира бессмертия его надо варить ровно 7 минут. Как это сделать?

   Придумайте систему команд исполнителя Колдун. Запишите с их помощью план действий исполнителя по приготовлению эликсира.
5. Исполнитель Вычислитель получает на вход целое число x и может выполнять с ним преобразования по алгоритму, состоящему из любого количества команд: 1) прибавить 5; 2) вычесть 2.

   Сколько разных алгоритмов, состоящих из пяти команд, можно составить для этого исполнителя? Сколько из них будут приводить к одинаковым результатам для заданного числа x?
6. Как известно, для каждого исполнителя набор допустимых действий всегда ограничен, иначе говоря, не может существовать исполнителя, для которого любое действие является допустимым. Докажите это утверждение, предположив, что такой исполнитель существует.
7. Перечислите известные вам способы записи алгоритмов.
8. Приведите примеры задач и оптимальных способов записи алгоритмов их решения.
9. Исполнитель Автомат получает на вход четырёхзначное число. Это число он преобразует по следующему алгоритму:
   1) вычисляется сумма первой и второй цифр числа;
   2) вычисляется сумма второй и третьей цифр числа;
   3) вычисляется сумма третьей и четвёртой цифр числа;
   4) из полученных трёх чисел (сумм) выбирается и отбрасывается одно — не превышающее двух других чисел;

5) оставшиеся два числа записываются друг за другом в порядке неубывания без разделителей.

Так, если исходное число 9575, то, преобразуя его, автомат создаст суммы: $9 + 5 = 14$, $5 + 7 = 12$, $7 + 5 = 12$. Сумма, не превышающая двух других, 12. Оставшиеся суммы: 14, 12. Результат: 1214.

Опишите систему команд этого исполнителя.

Могут ли результатом работы этого исполнителя быть числа 1610, 1010, 1019?

Укажите минимальное и максимальное значения результата работы этого исполнителя.

При обработке некоторого числа x автомат выдаёт результат 1418. Укажите наименьшее и наибольшее значения x, при которых возможен такой результат.

10. Подготовьте краткое сообщение об одном из учёных (А. Тьюринг, Э. Пост, А. Н. Колмогоров, А. А. Марков и др.), внёсших вклад в развитие теории алгоритмов.
11. В чём отличие шага алгоритма от команды алгоритма? Приведите пример.
12. Что такое сложность алгоритма? От чего она зависит в наибольшей степени?
13. Подсчитайте сложность алгоритма перемножения двух натуральных чисел «столбиком» при условии, что одно из них состоит из n, а второе — из m десятичных цифр.
14. Какой алгоритм считается эффективным?
15. Постройте эффективный алгоритм возведения числа x в степень $n = 152$.

§ 6
Алгоритмические структуры

Вне зависимости от выбранной формы записи элементарные шаги алгоритма объединяются в алгоритмические конструкции (структуры): последовательные, ветвящиеся, циклические, вспомогательные и рекурсивные. Для записи любого алгоритма достаточно трёх основных алгоритмических структур: последовательной, ветвящейся, циклической.

6.1. Последовательная алгоритмическая конструкция

Алгоритм реализован через **последовательную алгоритмическую конструкцию**, если все команды алгоритма выполняются один раз, причём в том порядке, в котором они записаны в тексте программы.

Пример 1. Алгоритм, реализованный через последовательную алгоритмическую конструкцию, представлен блок-схемой на рисунке 2.4.

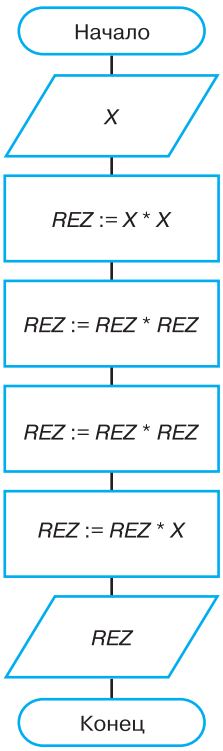


Рис. 2.4. Последовательная алгоритмическая конструкция

Выясните, какую задачу решает этот алгоритм. Чему равен результат работы алгоритма при $x = 2$?

Пример 2. Из курса информатики основной школы вам знаком исполнитель Вычислитель, выполняющий программы линейной структуры. Его система команд может содержать две и более команды с использованием арифметических операций.

Пусть исполнитель Вычислитель может выполнять следующие команды:

1) прибавь 2;
2) умножь на 3.

Подсчитаем, сколько разных программ, состоящих из трёх команд, можно составить для этого исполнителя, и выясним число различных значений, которые будут получены в результате их исполнения при начальном значении 2.

Так как каждую из команд вы можете выбрать одним из двух вариантов, а всего команд в программе три, общее число программ находится как $N = 2^3$.

Для удобства поиска числа разных вариантов значений, которые будут получены по этим программам, составим дерево решений (рис. 2.5). В его корневой вершине (0-й уровень) записывается начальное значение. Ветви дерева соответствуют командам: левые — первой, правые — второй. В каждой вершине дерева 1-го, 2-го и т. д. уровней записывается результат, полученный после применения команды к числу из вершины-предка. Путь к каждой из вершин соответствует последовательности команд (программе) для получения значения, находящегося в вершине. Таким образом в вершинах N-го уровня будут записанны результаты программ, состоящих из N команд.

В нашем примере все значения в вершинах третьего уровня различны. Иногда значения в вершинах одного уровня совпадают. Можно сказать, что в общем случае число различных значений на уровне не превышает числа его вершин.

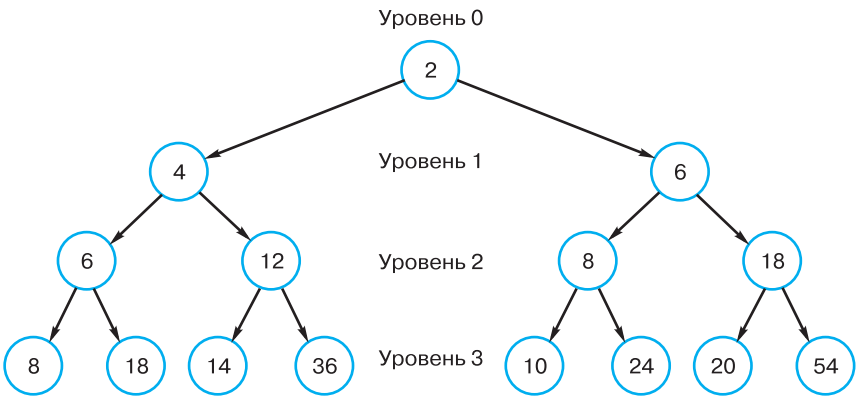


Рис. 2.5. Пример дерева решений

Номер уровня, на котором впервые появляется необходимое число, равен минимальному количеству команд для его получения. В нашем примере число 8 записано в вершинах 2-го и 3-го уровней, следовательно, минимальное количество команд для его получения равно двум.

Для определения количества программ, с помощью которых требуемое число получается из заданного, можно подсчитать количество вершин дерева, в которых записано требуемое число.

Подумайте, почему при заданных условиях существует только две программы для получения из числа 2 числа 8.

Если количество уровней вершин в дереве решений превышает четыре, то строить такое дерево неудобно. Так же как и в случае, когда система команд исполнителя состоит из трёх и более команд.

Для некоторых задач удобнее строить обратное дерево решений: команды исполнителя меняются на «обратные», пути строятся от результата к начальному значению. На рисунке 2.6 показан пример построения обратного дерева решений (из числа 8 получаем число 2; команды: 1) вычти 2; 2) раздели на 3).

Этот приём удобно использовать, когда обратные команды не применимы ко всем вершинам.

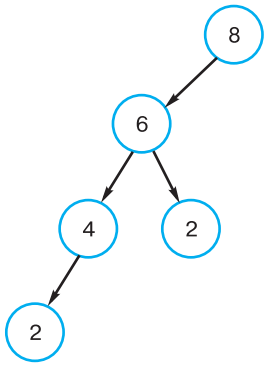


Рис. 2.6. Пример обратного дерева решений

6.2. Ветвящаяся алгоритмическая конструкция

Алгоритм реализован через **ветвящуюся алгоритмическую конструкцию**, если от входных данных зависит, какие команды алгоритма будут выполняться. При каждом конкретном наборе входных данных ветвящаяся алгоритмическая конструкция сводится к выполнению последовательной алгоритмической конструкции.

Пример 3. Алгоритм, реализованный через ветвящуюся алгоритмическую конструкцию, представлен блок-схемой на рисунке 2.7.

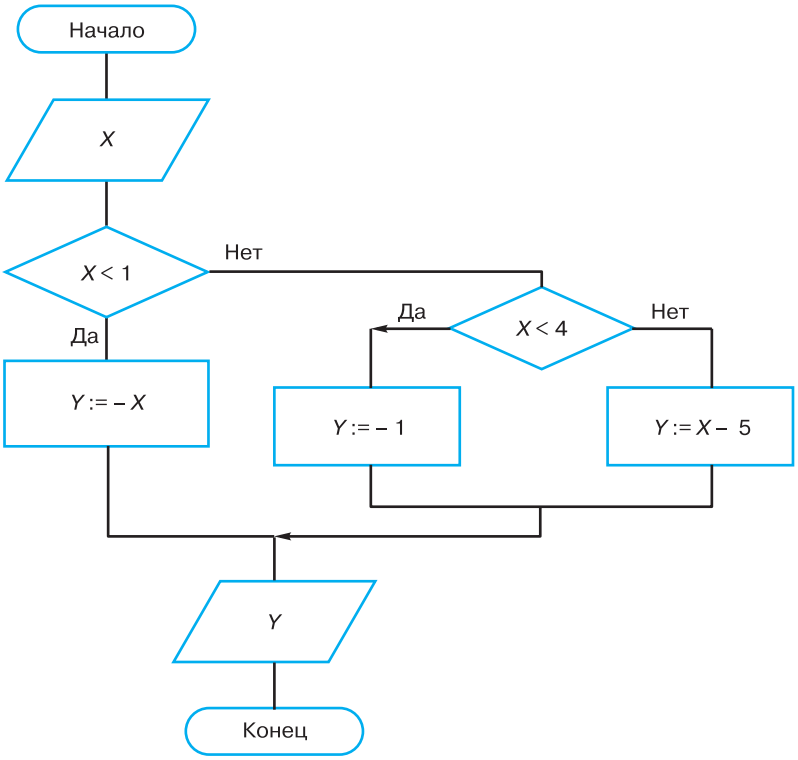


Рис. 2.7. Ветвящаяся алгоритмическая конструкция

Выясните, какую задачу решает этот алгоритм. К какой последовательной алгоритмической конструкции сводится эта ветвящаяся конструкция при:
1) $x = -10$;
2) $x = 2$;
3) $x = 10$?

6.3. Циклическая алгоритмическая конструкция

Алгоритм реализован с использованием **циклической алгоритмической конструкции**, если некая группа подряд идущих шагов алгоритма может выполняться многократно в зависимости от входных данных. Любая циклическая алгоритмическая конструкция содержит в себе элементы ветвящейся алгоритмической конструкции.

Циклическая структура (цикл) обеспечивает многократное выполнение одних и тех же команд. Существует несколько разновидностей циклических структур: цикл с предусловием (цикл-пока), цикл с постусловием (цикл-до), цикл с параметром. Любая циклическая структура состоит из двух частей — заголовка и тела цикла. Последовательность команд, повторяющуюся при выполнении цикла, называют телом цикла. Заголовок определяет количество повторений тела цикла. На рисунке 2.8 представлены блок-схемы цикла с предусловием и цикла с параметром.

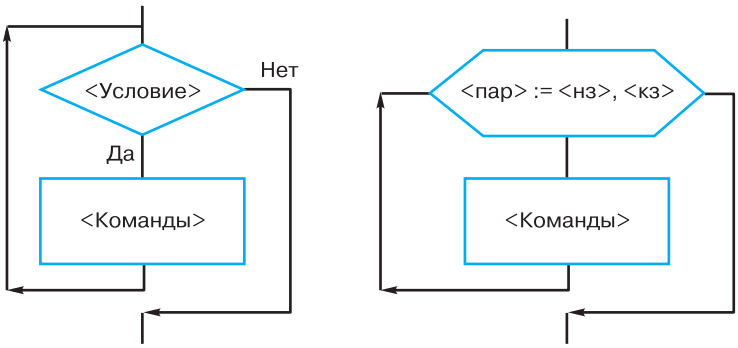


Рис. 2.8. Блок-схемы циклов

Пример 4. Исполнитель Редактор получает на вход строку цифр и преобразует её. Редактор может выполнять две команды, в которых параметры *v* и *w* обозначают цепочки цифр.

Команда **нашлось (*v*)** проверяет, встречается ли цепочка *v* в строке исполнителя Редактор. Если она встречается, то команда возвращает логическое значение «истина», в противном случае возвращает значение «ложь». Строка при этом не изменяется.

Команда **заменить (*v*, *w*)** заменяет в строке первое слева вхождение цепочки *v* на цепочку *w*.

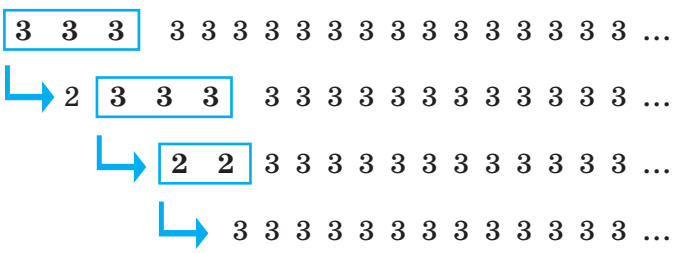
Дана программа для исполнителя Редактор:

```
НАЧАЛО
ПОКА нашлось (333) ИЛИ нашлось (22)
    ЕСЛИ нашлось (333)
        ТО заменить (333, 2)
        ИНАЧЕ заменить (22, 3)
    КОНЕЦ ЕСЛИ
КОНЕЦ ПОКА
КОНЕЦ
```

Выясним, какая строка получится в результате применения приведённой выше программы к строке, состоящей из N подряд идущих цифр 3 при:

1) $N = 21$;
2) $N = 25$.

Пусть $N = 21$. Исполнитель Редактор начинает работать со строкой, состоящей из одних только цифр 3: первые три цифры 3 сразу же заменяются цифрой 2, следующие три цифры 3 тоже заменяются цифрой 2, полученные две цифры 2 заменяются цифрой 3. После этого мы фактически возвращаемся к исходной задаче, которую должны решить для строки из 16 (21 – 3 – 3 + 1) подряд идущих троек:



Можно сказать, что при каждом повторении описанных выше действий из последовательности вычёркивается по пять цифр 3:

$21[3] \to 16[3] \to 11[3] \to 6[3] \to 1[3]$ или 3.

Пусть $N = 25$. Тогда:

$25[3] \to 20[3] \to 15[3] \to 10[3] \to 5[3] \to 1[2]2[3]$ или 233.

Итак, можно вычёркивать из последовательности по пять цифр 3, но только при условии, что в ней есть шесть и более идущих подряд троек.

Самостоятельно определите, какая строчка получится в результате применения приведённой выше программы к строке, состоящей из 22, 23 и 24 подряд идущих цифр 3.

Таким образом, можно сформулировать следующее правило преобразования строки из N подряд идущих цифр 3, соответствующее приведённому выше алгоритму:

1) если $N \bmod 5 = 0$, то $N := 5$, иначе $N := N \bmod 5$;
2) исполнить исходный алгоритм для строки, состоящей из N подряд идущих цифр 3.

Определите, какая строчка получится в результате применения приведённой выше программы к строке, состоящей из 2017, 12 345 подряд идущих цифр 3.

Определите, какая строчка получится в результате применения приведённой выше программы к строке, состоящей из 2015, 12 347 подряд идущих цифр 2.

Пример 5. Алгоритмы, реализованные через циклическую алгоритмическую конструкцию, представлены блок-схемами на рисунке 2.9.

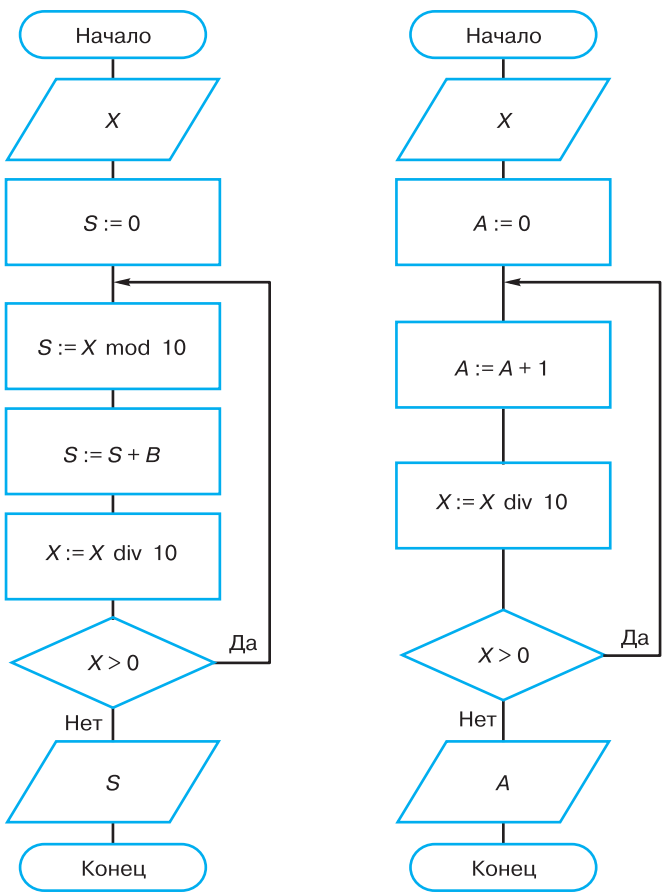


Рис. 2.9. Циклическая алгоритмическая конструкция

Известно, что X, A, B, S — целые положительные числа. Выясните, какую задачу решает каждый из алгоритмов на рисунке 2.9.

Известно, что при некотором X результатом работы и первого, и второго алгоритмов является число 3. Укажите все значения X, при которых возможен такой результат.

САМОЕ ГЛАВНОЕ

Вне зависимости от выбранной формы записи элементарные шаги алгоритма объединяются в алгоритмические конструкции (структуры): последовательные, ветвящиеся, циклические, вспомогательные и рекурсивные. Для записи любого алгоритма достаточно трёх основных алгоритмических структур: последовательной, ветвящейся, циклической.

Алгоритм реализован через последовательную алгоритмическую конструкцию, если все команды алгоритма выполняются один раз, причём в том порядке, в котором они записаны в тексте программы.

Алгоритм реализован через ветвящуюся алгоритмическую конструкцию, если от входных данных зависит, какие команды алгоритма будут выполняться.

Алгоритм реализован с использованием циклической алгоритмической конструкции, если некая группа подряд идущих шагов алгоритма может выполняться многократно в зависимости от входных данных.

Вопросы и задания

1. Какая алгоритмическая конструкция называется последовательной?
2. Петя приглашён в гости к однокласснику Васе, живущему в квартире № 362 шестнадцатиэтажного десятиподъездного дома. Петя забыл, в каком подъезде и на каком этаже живёт Вася, но знает, что в доме на каждой лестничной площадке по 4 квартиры. Помогите Пете узнать, в каком подъезде и на каком этаже находится нужная ему квартира.
3. Какая алгоритмическая конструкция называется ветвящейся? Как она связана с последовательной?
4. Как на блок-схемах изображается полное ветвление? Неполное ветвление?
5. Автомат по продаже напитков имеет только две кнопки (A и B), но должен продавать 4 напитка: горячий кофе, горячий чай, холодный яблочный сок и холодную газировку. Представьте в форме блок-схемы алгоритм работы такого автомата.

6. Разработайте и составьте в словесной форме инструкцию для школьного охранника: в какой последовательности и что он должен проверять (наличие пропуска, соответствие фотографии, есть ли сменная обувь и т. п.) и как реагировать на выявленные нарушения (вызвать милицию, отправить домой, сделать замечание, но пропустить, и т. д.).
7. Какая алгоритмическая конструкция называется циклической? Как она связана с ветвящейся?
8. Водитель автобуса, в котором K мест, продаёт билеты и по одному пропускает пассажиров в автобус. Он должен завершить посадку и уехать либо когда в автобус войдут все желающие, либо когда все места будут заняты. Составьте алгоритм действий водителя.
9. Исполнитель Редактор получает на вход строку цифр и преобразует её. Редактор может выполнять две команды. Команда **нашлось** (***v***) проверяет, встречается ли цепочка v в строке, поданной на вход исполнителя. Команда **заменить** (***v***, ***w***) заменяет в строке первое слева вхождение цепочки v на цепочку w. Дана программа для исполнителя Редактор:

```
НАЧАЛО
ПОКА нашлось (33) ИЛИ нашлось (22)
    ЕСЛИ нашлось (33)
        ТО заменить (33, 2)
        ИНАЧЕ заменить (22, 3)
    КОНЕЦ ЕСЛИ
КОНЕЦ ПОКА
КОНЕЦ
```

Какая строка получится в результате применения приведённой выше программы к строке, состоящей из:

1) 500 идущих подряд цифр 3;
2) 500 идущих подряд цифр 2;
3) 300 идущих подряд цифр 3 и следующих за ними 200 идущих подряд цифр 2.

§ 7 Запись алгоритмов на языках программирования

Язык программирования — формальная знаковая система, предназначенная для записи компьютерных программ.

Компьютерную программу можно считать последовательностью строк символов некоторого алфавита. Современные системы программирования допускают использование визуальных элементов (окон, иконок и др.) для построения программ, в частности для создания интерфейса пользователя. Такое программирование называют визуальным. Тем не менее основная, алгоритмическая часть любой программы строится с использованием символьных средств.

В основной школе вы познакомились со школьным алгоритмическим языком КуМир и языком программирования Pascal (Паскаль). В 11 классе мы продолжим работать с языком Pascal.

www

Желательно установить среду программирования на ваш домашний компьютер.

Все алгоритмы, представленные в этом учебнике на языке Pascal, вы можете записывать и на любом другом интересующем вас языке программирования.

Среда программирования	Сайт с программным обеспечением
Pascal ABC.Net	http://pascalabc.net
Компилятор Free Pascal	http://freepascal.org
Среда разработки Lazarus с компилятором Free Pascal	http://lazarus.freepascal.org
Интерпретатор Python	http://python.org

7.1. Структурная организация данных

Информация, представленная в виде, пригодном для автоматизированной обработки, называется данными. Компьютер оперирует только одним видом данных — отдельными битами, или двоичными цифрами. Причём он работает с этими данными в соответствии с неизменным набором алгоритмов, которые определяются системой команд центрального процессора.

Задачи, которые решаются с помощью компьютера, редко выражаются на языке битов. Как правило, данные имеют форму чисел, символов, текстов и более сложных структур. Алгоритмы, создаваемые для обработки этих данных, учитывают их структуру.

Под структурой данных в общем случае понимают множество элементов данных и множество связей между ними.

Различают простые и сложные структуры данных.

Простые структуры данных не могут быть разделены на составные части больше, чем бит. К ним относятся числовые, символьные, логические и другие данные. Простые структуры данных служат основой для построения сложных структур данных — массивов, списков, графов, деревьев и др.

В языках программирования понятие «структуры данных» тесно связано с понятием «типы данных». Любые данные, т. е. константы, переменные, значения функций или выражения, характеризуются своими типами. Информация по каждому типу однозначно определяет:

1) множество допустимых значений, которые может иметь тот или иной объект описываемого типа;
2) множество допустимых операций, которые применимы к объекту описываемого типа;
3) объём выделенной памяти для хранения данных указанного типа.

Некоторые простые типы данных языка Pascal приведены на рис. 2.10.

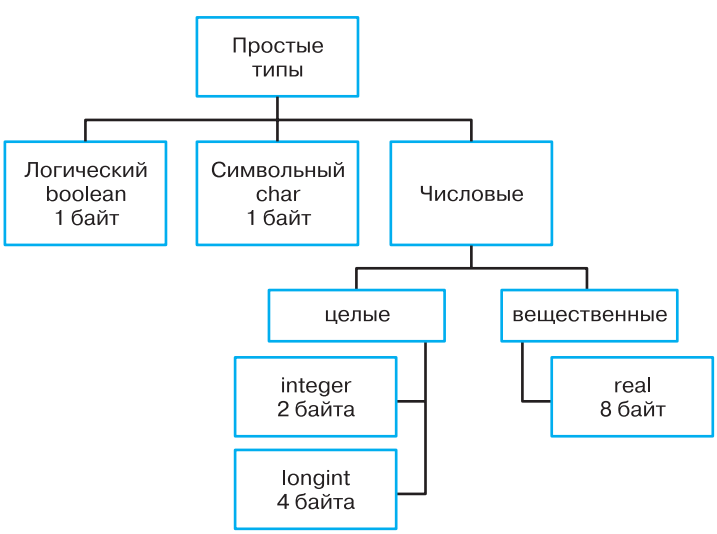


Рис. 2.10. Некоторые простые типы данных языка Pascal

7.2. Некоторые сведения о языке программирования Pascal

Основными элементами языка Pascal являются:

- алфавит языка (латинские буквы, арабские цифры, специальные символы);
- служебные слова, значение которых в языке программирования строго определено;
- постоянные и переменные величины;
- знаки операций (табл. 2.2);
- стандартные функции;
- выражения;
- операторы (языковые конструкции, с помощью которых в программах записываются действия, выполняемые над данными в процессе решения задачи).

Все величины имеют имена (идентификаторы), формируемые по определённым правилам:

- имя может состоять из буквы или последовательности букв латинского алфавита, цифр и символа подчёркивания, но начинаться такая последовательность должна с буквы или символа подчёркивания;
- желательно, чтобы имя отражало смысл величины;
- имя не должно совпадать ни с одним из зарезервированных слов.

Таблица 2.2

Операции в языке Pascal

Арифметические операции		Операции отношения	
+	Сложение	=	Равно
–	Вычитание	<>	Не равно
*	Умножение	>	Больше
/	Деление	<	Меньше
div	Целочисленное деление	<=	Меньше или равно
mod	Остаток от целочисленного деления	>=	Больше или равно

Окончание табл. 2.2

Логические операции		Строковые операции	
not	Логическое отрицание	+	Сцепление (присоединение)
and	Логическое И		
or	Логическое ИЛИ		
xor	Исключающее ИЛИ		

Выражение — это формула, по которой вычисляется значение. Выражение может состоять из операндов (констант, переменных, стандартных функций), знаков операций и круглых скобок. Выражения записываются в строку; знаки операций не пропускаются. Порядок выполнения операций определяется скобками и приоритетом операций (табл. 2.3). Операции одинакового приоритета выполняются слева направо, если порядок выполнения не задан явно круглыми скобками. Вычисление выражения с вложенными скобками начинается с внутренних скобок.

Таблица 2.3

Приоритет операций в языке Pascal

Приоритет	Операция
1	not
2	*, /, div, mod, and
3	+, –, or, xor
4	=, <>, >, <, >=, <=

Программа на языке Pascal имеет следующую структуру:

program <имя программы>;	Заголовок программы
var <переменные с указанием типов>; **const** <постоянные с указанием типов>;	Блок описания используемых данных
begin <последовательность команд>; **end.**	Блок описания действий по преобразованию данных (программный блок)

Обязательными в ней являются два раздела: описания данных и описания действий, которые над этими данными необходимо выполнить.

Данные, обрабатываемые компьютером, хранятся в памяти. С точки зрения языка Pascal она разделена на секции, называемые переменными. Каждая переменная имеет имя, тип и значение; значения переменных могут меняться в ходе выполнения программы.

Блок описания действий начинается со слова **begin**, а заканчивается словом **end** и знаком точки. Действия представляются операторами (табл. 2.4). Операторы языка Pascal разделяются точкой с запятой. Операторы бывают простые и составные (заключённые в операторные скобки **begin ... end**).

Таблица 2.4

Основные операторы языка Pascal

Название	Общий вид
Присваивание	`a:=b`
Ввод с клавиатуры	`read(a)`
Вывод на экран	`write(a)`
Условный	**if** <условие> **then** <оператор 1> **else** <оператор 2>
Цикл с предусловием	**while** <условие> **do** <тело цикла (операторы)>
Цикл с постусловием	**repeat** <тело цикла (операторы)> **until** <условие>
Цикл с увеличивающимся параметром	**for** <целочисленная переменная>:=<начальное значение> **to** <конечное значение> **do** <тело цикла (операторы)>
Цикл с уменьшающимся параметром	**for** <целочисленная переменная>:=<начальное значение> **downto** <конечное значение> **do** <тело цикла (операторы)>

Пример 1. В начале этой главы мы обсуждали алгоритмы нахождения простых чисел. Напишем программу, проверяющую, является ли заданное натуральное число n простым.

Самый простой путь решения этой задачи — проверить, имеет ли данное число n ($n >= 2$) делители в интервале $[2; n - 1]$.

Если делители есть, число n — составное, если — нет, то — простое.

В программе будем использовать логическую переменную *flag*:

- если *flag* = true, то n — простое число;
- если *flag* = false, то n — составное число (если у числа n есть делители, то «флаг выключаем» с помощью оператора присваивания *flag* := false).

```
var
  n, i: longint;
  flag: boolean;
begin
  writeln('Введите n');
  read(n);
  flag:=true;
  for i:=2 to n-1 do
    if n mod i = 0 then flag:=false;
  if flag then writeln('Да') else writeln('Нет')
end.
```

В этой программе мы проверяли, нет ли у числа n делителей из интервала $[2; n - 1]$. Но если $n = a \cdot b$, то меньшее из чисел a, b не больше $\sqrt{n}$ (в противном случае оба числа были бы больше $\sqrt{n}$, а следовательно, их произведение было бы больше n). Кроме того, из делимости числа n на a автоматически следует, что n делится и на n/a.

Усовершенствуйте приведённую выше программу с учётом этих соображений.

Проверку, является ли заданное натуральное число $n >= 2$ простым, мы осуществили методом перебора всех возможных его делителей. Метод перебора используется для решения достаточно широкого круга задач.

Пример 2. Применим метод перебора для поиска наибольшего общего делителя (НОД) двух натуральных чисел a и b.

Начнём перебор с d — наименьшего из чисел a и b. Это первый, очевидный кандидат на роль их наибольшего общего

делителя. И далее, пока не найдём d, на которое оба числа делятся нацело, будем уменьшать его на единицу. Как только такое деление произойдёт, останавливаем уменьшение d. Полученное значение d и будет наибольшим общим делителем чисел a и b.

```
var
  a, b, d: integer;
begin
  write('Введите два числа: ');
  readln(a, b);
  if a<b then d:=a
    else d:=b;
  while (a mod d <> 0) or (b mod d <> 0) do
    d:=d - 1;
  write('НОД = ', d)
end.
```

7.3. Анализ программ с помощью трассировочных таблиц

Для анализа свойств алгоритма и проверки его соответствия решаемой задаче используются трассировочные таблицы. В них фиксируется пошаговое исполнение алгоритма (программы), что позволяет наглядно представлять значения переменных, изменяющиеся при его выполнении. Поэтому трассировочные таблицы иначе называют таблицами значений.

Используются трассировочные таблицы двух видов:

1) таблицы, каждая строка которых отражает результат одного действия;
2) таблицы, каждая строка которых отражает результат выполнения группы действий.

Пример 3. Определим значения переменных a и b, полученные в результате выполнения следующей программы:

```
var a, b: integer;
begin
  a:=5;
  b:=1;
  while b<=a do
    begin
    b:=b + 1;
```

```
      a:=a - 1;
      end;
    writeln(a);
    writeln(b)
end.
```

Составим трассировочную таблицу первого вида. В её заголовке поместим имена всех переменных, используемых в программе. В отдельном столбце будем записывать команды и условия, имеющиеся в программе. Каждая строка таблицы соответствует одному шагу алгоритма. Чтобы не загромождать таблицу, будем записывать в каждой строке только то значение переменной, которое получено на соответствующем шаге.

№ шага	Команда или условие	Значение выражения	a	b
1	a:=5	5	5	
2	b:=1	1		1
3	b<=a	да		
4	b:=b+1	2		2
5	a:=a-1	5	4	
6	b<=a	да		
7	b:=b+1	3		3
8	a:=a-1	3	3	
9	b<=a	да		
10	b:=b+1	4		4
11	a:=a-1	2	2	
12	b<=a	нет		
13	writeln(a)		2	
14	writeln(b)			4

Из таблицы видно, что в результате работы переменные приняли значения: $a = 2$ и $b = 4$.

Пример 4. Определим значение переменной s, полученное в результате выполнения следующей программы:

```
var s, k, d: integer;
begin
  s:=0;
  d:=10;
  for k:=5 to 10 do
    s:=s+d;
  writeln(s)
end.
```

Построим трассировочную таблицу второго вида, отражая в каждой строке результат группы действий. Группу действий ограничим контрольной точкой: выполнение алгоритма продолжается до контрольной точки и приостанавливается после выполнения отмеченной ею строки.

Будем считать, что контрольная точка (КТ) поставлена на строке $s := s + d$.

Результат в КТ	k	s	d
Начальные значения	–	0	10
1	5	10	
2	6	20	
3	7	30	
4	8	40	
5	9	50	
6	10	60	

Итак, в результате работы программы переменная приняла значение $s = 60$.

Каким должно быть значение d, чтобы в результате работы программы переменная приняла значение $s = 186$? Существует ли такое значение d, что в результате работы программы переменная примет значение $s = 212$?

Пример 5. Определим значение переменной *s*, полученное в результате выполнения следующей программы:

```
var s, i, j: integer;
begin
  s:=0;
  for i:=1 to 3 do
    for j:=3 downto i do
      s:=s + i + j;
  writeln(s)
end.
```

Трассировочная таблица может иметь вид:

Результат в КТ	*i*	*j*	*s*
Начальные значения	–	–	0
1	1	3	4
2		2	7
3		1	9
4	2	3	14
5		2	18
6	3	3	24
Результат:	24		

Пример 6. Выясним, для чего предназначена следующая программа:

```
var
  n: integer; nd: string;
begin
  writeln('Введите натуральное число');
  read(n);
  nd:='';
  while n<>0 do
    begin
      if n mod 2=1 then nd:='1'+nd
                   else nd:='0'+nd;
      n:=n div 2
    end;
  writeln(nd);
end.
```

Прежде всего, обратим внимание на то, что в ней кроме переменной n целого типа используется строка nd, для которой символ «+» обозначает операцию сцепления строк. Начальное значение n вводится с клавиатуры, поэтому зададим его по своему усмотрению, например $n = 12$.

Результат в КТ	*nd*	*n*
Начальные значения	''	12
1	'0'	6
2	'00'	3
3	'100'	1
4	'1100'	0
Результат:	1100	

Выполните программу для $n = 25$. Какую задачу, по вашему мнению, решает эта программа?

7.4. Другие приёмы анализа программ

Трассировочная таблица — наглядный, но не универсальный инструмент анализа программ. Например, её затруднительно строить, если в алгоритме много шагов.

Пример 7. Требуется выяснить, какое число будет напечатано в результате выполнения следующей программы:

```
var n, s: integer;
begin
  n:=0;
  s:=400;
  while s<2992 do begin
    s:=s+12;
    n:=n+2
  end;
  write(n)
end.
```

Трассировочная таблица для этой программы будет содержать не одну сотню строк. Попробуем проанализировать программу иначе.

1. Выясним, какую функцию выполняет каждая из переменных, задействованных в программе.
   Начальное значение переменной $s = 400$. При каждом выполнении тела цикла к значению s прибавляется число 12.
   Начальное значение переменной $n = 0$. При каждом выполнении тела цикла значение переменной увеличивается на 2: $n = 2$, если тело цикла выполнено 1 раз; $n = 4$ — если 2 раза; $n = 6$ — если 3 раза и т. д. Таким образом, искомое значение n — это $2 \cdot k$, где k — число выполнений тела цикла.
2. Выясним, при каком условии произойдёт выход из цикла. Цикл выполняется, пока $s < 2992$. Следовательно, цикл завершится при достижении s значения, равного или большего 2992.
3. Выясним, сколько раз выполнится тело цикла, вычислив значение выражения: $(2992 - 400)/12 = 216$. После того как тело цикла выполнится 216 раз, значение переменной s будет равно 2992, что является условием выхода из цикла. При этом $n = 2 \cdot 216 = 432$.

Выясните, каким будет результат работы программы, если в ней условие выхода из цикла будет изменено на:
1) s < 2990; 2) s <= 2992; 3) s <= 300.

Пример 8. Получив на вход некоторое натуральное число x, эта программа выводит два числа — m и n.

```
var x, m, n: integer;
begin
  readln(x);
  m:=0; n:=1;
  while x>0 do
    begin
      m:=m+1;
      n:=n*(x mod 10);
      x:=x div 10;
    end;
  writeln(m); write(n)
end.
```

Известно, что при некотором значении x были выведены числа 5 и 25. Выясним, сколько существует разных значений x, при вводе которых может быть получен такой результат.

Выясним, какие именно данные накапливаются в переменных.

Начальное значение переменной x задаётся пользователем. Тип этой переменной integer, следовательно, она не может превышать 32 767. В цикле значение переменной x изменяется по правилу, заданному командой:

```
x:=x div 10
```

При таком преобразовании значение переменной x уменьшается в 10 раз и дробная часть результата отбрасывается. Можно сказать, что при каждом выполнении тела цикла от значения переменной x «отсекается» одна цифра справа.

Начальное значение переменной $m = 0$. При каждом выполнении цикла значение переменной m увеличивается на единицу. Можно сказать, что в m подсчитывается количество цифр, «отсечённых» от x.

Начальное значение переменной $n = 1$. В цикле значение переменной n изменяется по правилу, заданному командой:

```
n:=n*(x mod 10)
```

Здесь x mod 10 — не что иное, как последняя цифра числа x. Таким образом, в переменной n накапливается произведение цифр числа x, взятых справа налево.

Выход из цикла осуществляется при $x <= 0$, т. е. когда все значащие цифры этого числа будут рассмотрены.

Следовательно, если на экран первой выводится цифра 5, то исходное число пятизначное. Второе число указывает на то, что 25 — это произведение всех цифр исходного числа x.

Рассмотрим варианты пятизначных чисел, произведение цифр которых равно 25. Например, 11551, 51151 и т. д. Очевидно, в записи любого из таких чисел должны быть две пятёрки и три единицы. Применение известной вам формулы из комбинаторики позволяет вычислить число разных чисел, удовлетворяющих такому условию, — это 10.

О какой формуле идёт речь? Приведите эту формулу и выполните соответствующие вычисления.

Укажите наибольшее и наименьшее числа, удовлетворяющие условию задачи.

Выпишите все числа, удовлетворяющие условию задачи.

САМОЕ ГЛАВНОЕ

Компьютерную программу можно считать последовательностью строк символов некоторого алфавита. Современные системы программирования и языки допускают использование визуальных элементов (окон, иконок и др.) для построения программ, в частности для создания интерфейса пользователя. Тем не менее основная, алгоритмическая, часть любой программы строится с использованием символьных средств.

В основной школе вы познакомились со школьным алгоритмическим языком КуМир и языком программирования Pascal (Паскаль). В 11 классе мы продолжаем работать с языком Pascal.

Компьютер оперирует только одним видом данных — отдельными битами, или двоичными цифрами. Задачи, решаемые с помощью компьютера, оперируют данными, имеющими форму чисел, символов, текстов и более сложных структур. Алгоритмы для обработки этих данных создаются с учётом их структуры — множества элементов данных и множества связей между ними.

Различают простые и сложные структуры данных. Простые структуры данных не могут быть разделены на составные части больше, чем бит. К ним относятся числовые, символьные, логические и другие данные. Простые структуры данных служат основой для построения сложных структур данных — массивов, списков, графов, деревьев и др.

Для анализа свойств алгоритма и проверки его соответствия решаемой задаче используются трассировочные таблицы. В них фиксируется пошаговое исполнение алгоритма (программы), что позволяет наглядно представлять значения переменных, изменяющиеся при его выполнении. Используются трассировочные таблицы двух видов:

1) таблицы, каждая строка которых отражает результат одного действия;
2) таблицы, каждая строка которых отражает результат выполнения группы действий.

Вопросы и задания

1. Что такое язык программирования? Опишите состав и интерфейс среды разработки программ на используемом вами языке программирования.
2. Приведите примеры структур данных, используемых в языке программирования Pascal.

3. Кратко охарактеризуйте основные элементы языка программирования Pascal.
4. Опишите структуру программы на языке Pascal.
5. Для чего предназначены трассировочные таблицы?
6. Вещественные числа x, y, z являются исходными данными для следующего алгоритма:

   1) переменной m присвоить значение x;
   2) сравнить значения m и y: если y больше m, переменной m присвоить значение y;
   3) сравнить значения m и z: если z больше m, переменной m присвоить значение z.

   Выясните, какую задачу решает этот алгоритм. Запишите его на языке программирования Pascal. Решите аналогичную задачу для чисел x, y, z и w.
7. Определите значение переменной n, которое будет получено в результате выполнения следующей программы:

```
var s, n: integer;
begin
  s:=0; n:=1;
  while sqr(s+2)<125 do
    begin
      n:=n*2;
      s:=s+2;
    end;
  writeln(n)
end.
```

8. Определите значение переменной s, которое будет получено в результате выполнения следующей программы:

```
var s, i, j: integer;
begin
  s:=0;
  for i:=1 to 3 do
    for j:=i to 4 do
      s:=s+2*i-j;
  writeln (s)
end.
```

9. Требуется выяснить, какое число будет выведено в результате выполнения следующей программы:

```
var n, s: integer;
begin
  n:=0;
  s:=1000;
  while s>=100 do
    begin
      s:=s-2;
      n:=n+1
    end;
  write(n)
end.
```

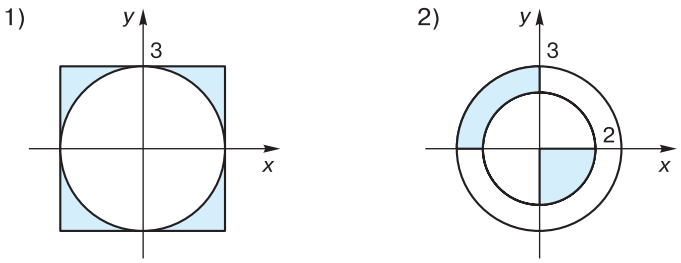
10. Получив на вход число x, приведённая ниже программа выводит два числа — m и n.

```
var x, m, n: integer;
begin
  readln(x);
  m:=0; n:=0;
  while x>0 do
    begin
      if n<x mod 10 then n:=x mod 10;
      m:=m+1;
      x:=x div 10;
    end;
  writeln(m); write(n)
end.
```

Известно, что при некотором значении x были выведены числа 4 и 8. Укажите наибольшее и наименьшее из таких чисел x. Сколько всего существует таких x?

11. Напишите программу, выводящую на экран все чётные трёхзначные числа.
12. Напишите программу, подсчитывающую сумму квадратов всех чисел от 1 до n.
13. Напишите программу, позволяющую определить, входит ли заданная цифра в некоторое целое неотрицательное число.
14. Разработайте программу перевода десятичного натурального числа n в троичную систему счисления.

15. Разработайте программу, которая выводит сообщение «Да», если точка с координатами (x, y) принадлежит закрашенной области, и «Нет» в противном случае.



16. Шифр кодового замка является двузначным числом. Буратино забыл код, но помнит, что сумма цифр этого числа, сложенная с их произведением, равна самому числу. Напишите все возможные варианты кода, чтобы Буратино смог быстрее открыть замок. Решите задачу методом перебора.

§ 8 Структурированные типы данных. Массивы

Мы повторили основные приёмы работы с простыми типами данных. Из элементов простых типов в языке Pascal можно образовывать составные типы данных (структуры данных). Примером таких структур являются одномерные массивы.

Массив — это поименованная совокупность однотипных элементов, упорядоченных по индексам, определяющим положение элемента в массиве.

8.1. Общие сведения об одномерных массивах

Массив в языке Pascal — это набор однотипных данных, причём количество этих данных фиксировано и определяется при описании массива. Все переменные, входящие в массив, имеют одно и то же имя — имя массива, а различаются они по индексу — номеру (месту) в массиве.

Описание массива выглядит так:

```
array [<тип индекса>] of <тип компонент>
```

Здесь:

- **array** и **of** — служебные слова («массив» и «из»);
- <тип индекса> — описание индексации компонент (элементов) массива;
- <тип компонент> — тип величин, составляющих массив.

Например:

- **var** *day*: **array** [1..365] **of** integer — 365 целочисленных элементов пронумерованы от 1 до 365;
- **var** *tem*: **array** [1..12] **of** real — 12 вещественных элементов пронумерованы от 1 до 12;
- **var** *ocenka*: **array** [2..5] **of** integer — 4 целочисленных элемента пронумерованы от 2 до 5;
- **const** *n* = 10; **var** *slovo*: array [1..*n*] **of** string — *n* строковых величин пронумерованы от 1 до *n*.

Вспомним основные приёмы работы с массивами.

Пример 1. Имеются сведения о количестве ежедневных осадков в течение июня месяца в некотором регионе. Требуется найти среднее количество осадков и вывести таблицу, в которой для каждого дня месяца указать количество осадков в этот день и его отклонение от среднемесячного значения.

Для решения этой задачи данные о количестве ежедневных осадков в течение месяца будут просмотрены дважды:

1) при поиске среднего значения;
2) при расчёте отклонения.

Для решения задачи нам понадобится массив из 30 вещественных чисел. Назовём его *osad*. В программе будет два цикла. В первом цикле мы будем вводить значения элементов массива и сразу же подсчитывать их сумму — по завершении цикла мы получим сумму осадков, выпавших в течение месяца. Во втором цикле мы будем выводить строки таблицы и вычислять значения отклонений.

При работе с элементами массива будем пользоваться переменной *osad*[*i*]; значение индекса *i* при этом будет изменяться от 1 до 30 с шагом 1. Для вычисления среднего значения задействуем вещественную переменную *sred*, присвоив ей начальное значение 0 и последовательно накапливая в ней сумму осадков, выпавших в течение месяца. Разделив по завершении цикла

полученное значение на 30, вычислим требуемое среднемесячное количество осадков и присвоим результат этой же переменной.

```
Program osadki;
var osad: array [1..30] of real;
    sred: real; i: integer;
begin
  sred:=0;
  writeln('Введите количество осадков по дням');
  for i:=1 to 30 do
    begin
      write(i, ' июня: ');
      readln(osad[i]);
      sred:=sred + osad[i]
    end;
  sred:=sred/30;
  writeln('День Количество осадков Отклонение');
  for i:=1 to 30 do
    writeln(i:3, osad[i]:15:3, osad[i] - sred:12:2)
end.
```

Найдите в Интернете информацию о количестве ежедневных осадков, выпавших в течение месяца, в вашем регионе. Используя эти данные, выполните программу в среде программирования Pascal.

Выполните аналогичные расчёты с помощью электронных таблиц.

Чаще всего массив обрабатывается в цикле **for**. Но при работе с массивами можно использовать и другие циклы.

Пример 2. Имеется массив символов. Требуется вывести на экран элементы данного массива в обратном порядке.

Элементами массива символов могут быть любые символы, имеющиеся на клавиатуре, причём каждому элементу соответствует именно один символ. Если в качестве элементов нашего массива рассматривать последовательности букв, образующие некоторое слово или фразу на естественном языке, то, решив поставленную задачу, мы научимся строить «перевёртыши» слов.

Будем рассматривать слова и фразы не более чем из 20 символов, задав соответствующую размерность массива:

```
simbol: array [1..20] of char;
```

Если какое-то слово или фраза будут короче, то часть массива окажется не занятой, но это не повлияет на работу программы. Договоримся признаком конца слова считать точку — ввод символов продолжается, пока не введена точка; после ввода точки ввод символов прекращается.

```
program slova;
var simbol: array [1..20] of char;
    i, n: integer;
begin
  writeln('Введите слово - цепочку символов —
           с точкой в конце');
  i:=0;
  repeat
    i:=i+1;
    read(simbol[i]);
  until simbol[i]='.';
  n:=i-1;
  writeln('Перевёрнутое слово: ');
  for i:=n downto 1 do
    write(simbol[i]);
end.
```

Запустите программу в среде программирования Pascal.

Модифицируйте программу так, чтобы в начале её работы пользователю задавался вопрос о количестве символов, которые он будет вводить. Какой цикл при этом лучше использовать?

Как изменить программу, чтобы она выводила на экран в обратном порядке элементы целочисленного массива?

К типовым задачам обработки одномерных массивов, решаемым в процессе их однократного просмотра, относятся:

- задачи поиска элементов с заданными свойствами, в том числе максимумов и минимумов;
- проверка соответствия элементов массива некоторому условию (подсчёт количества или суммы элементов, удовлетворяющих некоторому условию; проверка соответствия всех элементов массива некоторому условию; проверка массива на упорядоченность и др.);
- задачи на удаление и вставку элементов массива;
- задачи на перестановку всех элементов массива в обратном порядке и т. д.

8.2. Задачи поиска элемента с заданными свойствами

Очень часто в реальной жизни нам приходится сталкиваться с задачей поиска информации в большом массиве данных. Например, поиск нужного слова в словаре, поиск времени отправления нужного поезда в расписании, поиск нужного товара в интернет-магазине и т. д.

В программировании поиск — одна из наиболее часто встречающихся задач невычислительного характера.

В алгоритмах поиска существует два возможных варианта окончания их работы: поиск может оказаться удачным — заданный элемент найден в массиве и определено его месторасположение, либо поиск может оказаться неудачным — необходимого элемента в данном объёме информации нет.

Рассмотрим несколько типовых задач поиска, первое знакомство с которыми у вас состоялось ещё в основной школе.

Пример 3. Последовательный поиск в неупорядоченном массиве.

Имеется массив $a[1..n]$; требуется найти элемент массива, равный p.

Алгоритм последовательного поиска в неупорядоченном массиве может быть следующим.

1. Установить $i = 1$.
2. Если $a[i] = p$, алгоритм завершил работу успешно.
3. Увеличить i на 1.
4. Если $i \le n$, то перейти к шагу 2. В противном случае алгоритм завершил работу безуспешно.

Возможная программа, реализующая этот алгоритм на языке Pascal, имеет вид:

```
const n=10;
var a: array [1..n] of integer; i, p: integer;
begin
  writeln('Ввод значений элементов массива:');
  for i:=1 to n do
    read(a[i]);
  write('Ввод p: ');
  readln(p);
  i:=1;
  while (i<=n) and (a[i]<>p) do i:=i+1;
```

```
  if i=n+1
    then writeln('Искомого элемента в массиве нет')
    else writeln('Искомый элемент a[', i, '] = ', a[i])
end.
```

Внимательно рассмотрите условие продолжения цикла. В каком случае выполнение цикла продолжается? В каких случаях осуществляется выход из цикла?

Запустите программу на выполнение в среде программирования Pascal.

Как иначе можно решить эту задачу, например, с использованием цикла **for**? Напишите соответствующую программу.

Оценим сложность рассмотренного алгоритма последовательного поиска, непосредственно зависящую от числа сравнений с искомым элементом. В худшем случае искомый элемент окажется на последнем месте или не будет найден вообще. В таком случае необходимо будет проделать n сравнений, т. е. сложность алгоритма будет равна $O(n)$.

Пример 4. Поиск максимумов и минимумов.

Имеется массив $a[1..n]$; требуется найти значение наибольшего (наименьшего) элемента массива.

Алгоритм поиска значения наибольшего (максимального) элемента в неупорядоченном массиве может быть следующим.

1. Установить значение текущего максимума равным первому исследуемому элементу ($max := a[1]$).
2. Установить счётчик равным 2 ($i := 2$).
3. Если исследованы ещё не все элементы ($i <= n$), то перейти к шагу 4, иначе алгоритм окончен (максимальный элемент равен max).
4. Если рассматриваемый элемент больше, чем текущий максимум ($a[i] > max$), то max присвоить значение $a[i]$.
5. Перейти к следующему элементу (увеличить i на единицу).
6. Перейти к шагу 3.

Возможная программа, реализующая этот алгоритм на языке Pascal, имеет вид:

```
const n=10;
var a: array [1..n] of integer;
    i, max: integer;
begin
  writeln('Ввод значений элементов массива:');
```

```
  for i:=1 to n do
    read(a[i]);
  max:=a[1];
  i:=2;
  while (i<=n) do
    begin
      if a[i] > max then max:=a[i];
      i:=i+1
    end;
  writeln('Max=', max)
end.
```

Запустите программу на выполнение в среде программирования Pascal.

Как иначе можно решить эту задачу, например, с использованием цикла **for**? Напишите соответствующую программу.

Преобразуйте программу так, чтобы с её помощью можно было находить минимальный элемент массива.

Какие изменения надо внести в программу для поиска индекса максимального (минимального) элемента массива?

Самостоятельно оцените сложность рассмотренного алгоритма.

8.3. Проверка соответствия элементов массива некоторому условию

Пример 5. Подсчёт количества элементов, удовлетворяющих некоторому условию.

Зачастую бывает важно выяснить, сколько элементов, обладающих определённым свойством, содержится в массиве.

Для решения этой задачи следует:

1) присвоить нулевое значение переменной, введённой для подсчёта количества элементов, удовлетворяющих заданному условию ($k := 0$);
2) организовать просмотр всех элементов массива: если просматриваемый элемент удовлетворяет заданному условию, значение переменной k увеличивать на 1.

Фрагмент программы подсчёта количества элементов массива, например бóльших некоторого числа p, имеет вид:

```
k:=0;
for i:=1 to n do
    if a[i]>p then k:=k+1;
```

Запишите полный текст программы и выполните её на компьютере для рассматриваемого в примере 8 массива *a*, состоящего из семи элементов, и числа *p* = 15.

Как модифицировать программу, чтобы можно было вычислить сумму элементов массива, бóльших некоторого числа *p*?

Пример 6. Проверка соответствия всех элементов массива некоторому условию.

Для того, чтобы установить факт соответствия всех элементов массива некоторому условию достаточно:

1) подсчитать количество элементов массива, соответствующих заданному условию;
2) сравнить найденное количество с общим числом элементов массива и вывести соответствующий результат.

Самостоятельно разработайте программу, позволяющую определить, все ли элементы массива являются двузначными числами. Выполните её на компьютере для рассматриваемого в примере 8 массива *a*, состоящего из семи элементов.

Пример 7. Проверка массива на упорядоченность.

Рассмотрим алгоритм, позволяющий определить, упорядочены ли элементы массива $a[1..n]$ по неубыванию, т. е. каждый элемент массива с 1-го по $(n-1)$-й не больше последующего.

Самый простой путь решения этой задачи — проверить, есть ли в массиве такие пары элементов, что $a[i] > a[i+1]$. Если подобные пары элементов есть, то массив не упорядочен по неубыванию, а если таких пар нет, то упорядочен.

В программе будем использовать логическую переменную *flag*:

- если *flag* = true, то массив упорядочен;
- если *flag* = false, то массив неупорядочен.

Ниже представлен фрагмент программы, реализующей этот алгоритм:

```
flag:=true;
for i:=1 to n-1 do
    if a[i]>a[i+1] then flag:=false;
```

Запишите полный текст программы и выполните её на компьютере для рассматриваемого в примере 8 массива *a*, состоящего из семи элементов.

Как можно решить эту же задачу путём подсчёта количества пар элементов массива, таких что *a*[*i*] > *a*[*i* + 1] (*a*[*i*] <= *a*[*i* + 1])?

8.4. Удаление и вставка элементов массива

Пример 8. Удаление из массива элемента с индексом *k*.

Имеется одномерный целочисленный массив из семи элементов:

i	1	2	3	4	5	6	7
a[*i*]	10	12	5	8	4	15	20

Удалим из массива элемент с индексом $k = 4$, а все элементы, расположенные справа от него, сдвинем на одну позицию влево. Получим следующий целочисленный массив из шести элементов:

i	1	2	3	4	5	6
a[*i*]	10	12	5	4	15	20

При удалении из массива любого из элементов размерность массива уменьшается на 1.

Мы видим, что элементы с индексами от 1 до $k - 1$ не изменились. На место элемента с индексом k (4) переместился элемент, имевший индекс $k + 1$ (5), на место элемента с индексом $k + 1$ (5) переместился элемент, имевший индекс $k + 2$ (6) и т. д.

В общем случае, фрагмент программы удаления из массива $a[1..n]$ элемента с индексом k и последующим сдвигом всех расположенных справа от него элементов на одну позицию влево имеет вид:

```
for i:=k to n-1 do
  a[i]:=a[i+1];
```

Запишите полный текст программы и выполните её на компьютере для рассмотренного выше массива *a*.

Пример 9. Вставка в массив элемента на место с индексом *k*.

Будем работать с тем же массивом из семи элементов. Но теперь наша задача будет состоять в том, чтобы вставить в массив на место с индексом $k = 4$ (т. е. после элемента с индексом $k - 1$) ещё один элемент, имеющий значение 11.

Получим следующий целочисленный массив из восьми элементов:

i	1	2	3	4	5	6	7	8
$a[i]$	10	12	5	11	8	4	15	20

При вставке в массив ещё одного элемента размерность массива увеличивается на 1. Это надо учесть при описании массива.

Итак, $a[4] := 11$. Элементу $a[5]$ следует присвоить то значение, которое было у $a[4]$, элементу $a[6]$ — значение, которое было у $a[5]$ и т. д. В общем случае, элементу $a[k + 1]$ следует присвоить то значение, которое было у $a[k]$.

Подумайте, что получится в результате выполнения следующих групп операторов присваивания:

1) `a[4]:=11; a[5]:=a[4]; a[6]:=a[5]; a[7]:=a[6]; a[8]:=a[7];`
2) `a[8]:=a[7]; a[7]:=a[6]; a[6]:=a[5]; a[5]:=a[4]; a[4]:=11;`

В общем случае, фрагмент программы вставки в массив $a[1..n - 1]$ элемента на место с индексом k и сдвигом k-го, $(k + 1)$-го, ..., $(n - 1)$-го элементов на одну позицию вправо имеет вид:

```
for i:=n downto k+1 do
  a[i]:=a[i-1];
a[k]:=<значение элемента>;
```

Запишите полный текст программы и выполните её на компьютере для рассмотренного выше массива a. Помните, что при описании массива надо учесть размерность массива, получающегося в результате работы программы.

8.5. Перестановка всех элементов массива в обратном порядке

Пример 10. Перестановка всех элементов массива $a[1..n]$ в обратном порядке сводится к тому, что меняются местами первый и последний элементы, второй и предпоследний элементы и т. д.

Перестановка нашего массива из семи элементов даст такой результат:

i	1	2	3	4	5	6	7
$a[i]$	20	15	4	8	5	12	10

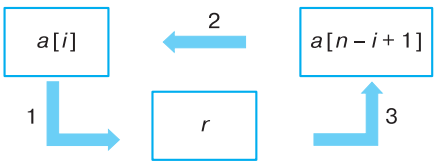
В общем случае, меняются местами элементы $a[i]$ и $a[n - i + 1]$.

Вспомним, как можно произвести обмен значений между двумя переменными. Выполнение операторов:

```
a[1]:=a[n]; a[n]:=a[1];
```

к желаемому результату не приводит. Самый простой вариант — использование вспомогательной переменной:

```
r:=a[i];
a[i]:=a[n-i+1];
a[n-i+1]:=r;
```



Выясним, сколько всего операций обмена следует произвести.

Если произведена перестановка, например, первого и последнего элементов, то одновременно произведена и перестановка последнего и первого элементов. Таким образом, если число элементов массива чётное, то достаточно произвести $n/2$ операций обмена.

Но что происходит, если массив содержит нечётное число элементов? Например, в нашем массиве из семи элементов выполнялось три обмена, а четвёртый элемент, занимающий центральную позицию, оставался на своём месте. В общем случае число операций обмена при перестановке в обратном порядке всех n элементов массива определяется как n div 2.

В общем случае, фрагмент программы по перестановке в обратном порядке всех элементов массива $a[1..n]$ имеет вид:

```
for i:=1 to n div 2 do
  begin
    r:=a[i];
    a[i]:=a[n-i+1];
    a[n-i+1]:=r
  end
```

Запишите полный текст программы и выполните её на компьютере для рассмотренного выше массива *a*, состоящего из семи и из шести элементов.

8.6. Сортировка массива

Сортировка — один из наиболее распространённых процессов современной обработки данных.

Сортировка — это распределение элементов массива в соответствии с определёнными правилами.

Под сортировкой (упорядочением) массива понимают перераспределение значений его элементов в некотором определённом порядке.

Порядок, при котором в массиве первый элемент имеет самое маленькое значение, а значение каждого следующего элемента не меньше значения предыдущего элемента, называют неубывающим.

Порядок, при котором в массиве первый элемент имеет самое большое значение, а значение каждого следующего элемента не больше значения предыдущего элемента, называют невозрастающим.

Цель сортировки — ускорить последующий поиск элементов, т. к. нужный элемент легче искать в упорядоченном массиве.

Рассмотрим и проанализируем несколько алгоритмов сортировки для решения следующей задачи. Дан одномерный массив целых чисел. Требуется отсортировать его так, чтобы все элементы были расположены в порядке неубывания: $a[i] <= a[i + 1]$.

Обменная сортировка методом «пузырька»

Своё название алгоритм получил благодаря следующей ассоциации: если сортировать этим алгоритмом массив по неубыванию, то максимальный элемент «тонет», а «лёгкие» элементы поднимаются на одну позицию к началу массива на каждом шаге алгоритма.

Пусть n — количество элементов в неупорядоченном массиве.

1. Поместим на место n-го элемента ($a[n]$) наибольший элемент массива. Для этого:
   1) положим $i = 1$;
   2) пока не обработана последняя пара элементов, т. е. $(n - 1)$-й и n-й элементы:
      - сравниваем i-й и $(i + 1)$-й элементы массива;
      - если $a[i] > a[i + 1]$ (элементы расположены не по порядку), то меняем элементы местами;
      - переходим к следующей паре элементов, сдвинувшись на один элемент вправо.
2. Повторяем пункт 1, каждый раз уменьшая размерность неупорядоченного массива на 1, до тех пор, пока не будет обработан массив из одной пары элементов (таким образом, на k-м просмотре будут сравниваться первые $(n - k)$ элементов со своими соседями справа).

Пример 11. Есть массив: 5 4 3 2 1. На примере этого массива подсчитаем количество элементарных действий в вычислительном процессе алгоритма сортировки методом «пузырька»:

- 1-я итерация: 4 3 2 1 5 (4 сравнения, 4 обмена);
- 2-я итерация: 3 2 1 4 5 (3 сравнения, 3 обмена);
- 3-я итерация: 2 1 3 4 5 (2 сравнения, 2 обмена);
- 4-я итерация: 1 2 3 4 5 (1 сравнение, 1 обмен).

Алгоритм закончил работу. Было сделано 10 сравнений и 10 обменов (4 + 3 + 2 + 1).

Этот алгоритм легко запоминается, но на практике он используется достаточно редко из-за квадратичной сложности, означающей, что в общем случае количество выполненных сравнений и обменов сопоставимо с n^2, где n — количество элементов массива.

Попытайтесь самостоятельно запрограммировать алгоритм сортировки методом «пузырька».

Сортировка выбором

Сортировка выбором (в порядке неубывания) осуществляется следующим образом:

1) в массиве выбирается минимальный элемент;
2) минимальный и первый элементы меняются местами (первый элемент считается отсортированным);
3) в неотсортированной части массива снова выбирается минимальный элемент и меняется местами с первым неотсортированным элементом массива;
4) действия, описанные в пункте 3, повторяются с неотсортированными элементами массива до тех пор, пока не останется один неотсортированный элемент (его значение будет максимальным).

Пример 12. Есть массив: 5 4 3 2 1.
1-я итерация: 1 4 3 2 5 (4 сравнения, 1 обмен).
2-я итерация: 1 2 3 4 5 (3 сравнения, 1 обмен).
3-я итерация: 1 2 3 4 5 (2 сравнения, 0 обменов).
4-я итерация: 1 2 3 4 5 (1 сравнение, 0 обменов).

В общем случае алгоритм сортировки выбором имеет квадратичную сложность относительно операций сравнения и линейную сложность относительно операций обменов. Этот алгоритм целесообразно применять, когда операция обмена над элементами массива особенно трудоёмка (например, если элементом массива является запись с большим числом полей).

Приведём фрагмент программы, реализующей описанный выше алгоритм:

```
for i:=1 to n-1 do
  begin
    imin:=i;
    for j:=i+1 to n do
      if a[j]<a[imin] then imin:=j;
    per:=a[i];
    a[i]:=a[imin];
    a[imin]:=per;
  end;
```

Запишите полный текст программы и выполните её на компьютере для рассмотренного выше массива.

САМОЕ ГЛАВНОЕ

Из элементов простых типов в языке Pascal можно образовывать составные типы данных (структуры данных). Примером таких структур являются одномерные массивы.

Массив в языке Pascal — это набор однотипных данных, причём количество этих данных фиксировано и определяется при описании массива. Все переменные, входящие в массив, имеют одно и то же имя — имя массива, а различаются они по индексу — номеру (месту) в массиве.

Перед использованием в программе массив должен быть описан, т. е. должно быть указано имя массива, количество элементов массива и их тип. Это необходимо для того, чтобы выделить в памяти под массив блок ячеек нужного типа.

Чаще всего массив обрабатывается в цикле **for**. Но при работе с массивами можно использовать и другие циклы.

К типовым задачам обработки одномерных массивов, решаемым в процессе их однократного просмотра, относятся:

- задачи поиска элемента с заданными свойствами, в том числе максимумов и минимумов;
- проверка соответствия элементов массива некоторому условию (подсчёт количества или суммы элементов, удовлетворяющих некоторому условию; проверка соответствия всех элементов массива некоторому условию; проверка массива на упорядоченность и др.);
- задачи на удаление и вставку элементов массива;
- задачи на перестановку всех элементов массива в обратном порядке и т. д.

Сортировка — один из наиболее распространённых процессов современной обработки данных. Под сортировкой (упорядочением) массива понимают перераспределение значений его элементов в некотором определённом порядке.

Вопросы и задания

1. Приведите примеры задач поиска информации в больших массивах данных.
2. Почему важно уметь решать задачи, связанные с обработкой массивов, путём однократного просмотра массива?
3. Программист написал программу суммирования элементов массива, но допустил в ней ошибку.

```
Program summa;
const n=10;
var a: array [1..n] of integer; s, i: integer;
begin
  s:=0;
  for i:=1 to n do
    begin
      readln(a[i]);
      s:=s+i
    end;
  writeln('s=', s)
end.
```

1) Что получится в результате выполнения этой программы, если в качестве элементов массива ввести числа: 1, –2, 3, –4, 5, –6, 7, –8, 9, –10?
2) Придумайте пример такого массива, обработка которого с помощью этой программы приводила бы к правильному результату.
3) Найдите ошибку, допущенную программистом.

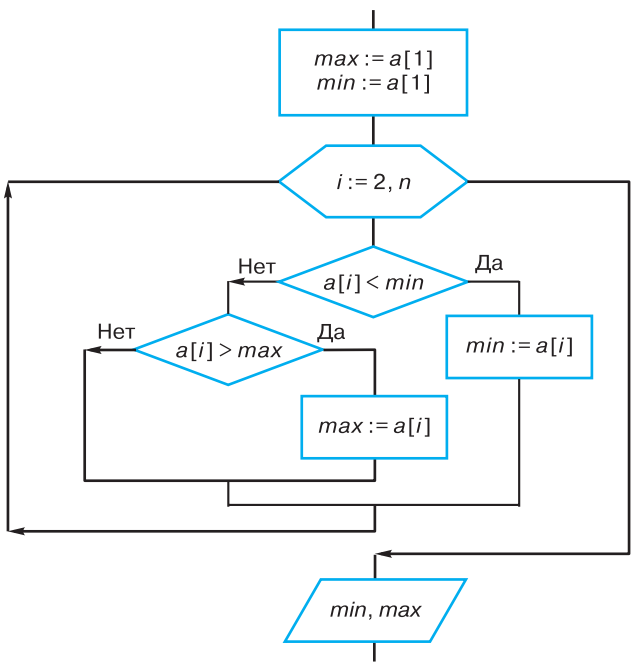
4. Программист написал программу нахождения произведения элементов массива, но допустил в ней ошибку.

```
Program proizv;
const n=10;
var a: array [1..n] of integer; p, i: integer;
begin
  p:=0;
  for i:=1 to n do
```

```
    begin
      readln(a[i]);
      p:=p*a[i]
    end;
  writeln('p=', p)'
end.
```

1) Что получится в результате выполнения этой программы, если в качестве элементов массива ввести числа: 1, –2, 3, –4, 5, –6, 7, –8, 9, –10?
2) Придумайте пример такого массива, обработка которого с помощью этой программы приводила бы к правильному результату.
3) Найдите ошибку, допущенную программистом.

5. На блок-схеме представлен алгоритм одновременного поиска максимального и минимального значений элементов массива:



Реализуйте этот алгоритм на языке программирования и выполните программу для массива из задания 6.

6. Имеется одномерный целочисленный массив из семи элементов:

i	1	2	3	4	5	6	7
a[*i*]	10	12	5	8	4	15	20

Каким будет результат преобразования массива по следующему алгоритму?

```
for i:=k+1 to n do
  a[i-1]:=a[i];
```

7. Имеется ли разница между операциями вставки в массив элемента на место с индексом k и замены значения элемента массива с индексом k? Обоснуйте свой ответ.

8. Имеется одномерный целочисленный массив из семи элементов:

i	1	2	3	4	5	6	7
a[*i*]	10	12	5	8	4	15	20

Каким будет результат преобразования массива по следующему алгоритму?

```
for i:=1 to n div 2 do
  begin
    r:=a[i];
    a[i]:=a[n-i+1];
    a[n-i+1]:=r
  end;
```

9. Дана программа:

```
const n=5;
const a: array[1..n] of integer=(1,2,6,4,6);
var i, max1, max2: integer;
begin
  max1:=a[1];
  max2:=a[2];
  for i:=2 to n do
    if a[i]>max1
    then begin max2:=max1; max1:=a[i]; end
    else if a[i]>max2 then max2:=a[i];
  writeln('max1=', max1, ', max2=', max2);
end.
```

Что получится в результате выполнения этой программы? Какую задачу решает эта программа?

10. Дано натуральное десятичное число $n <= 32\,000$. Напишите программу, в которой:
    1) из цифр данного числа формируется одномерный целочисленный массив;
    2) определяются наибольшая и наименьшая цифры данного числа;
    3) находятся сумма и произведение цифр, образующих данное число.
11. Требуется упорядочить по весу в порядке неубывания n непрозрачных банок с чаем, имея в своём распоряжении только чашечные весы без гирь. Опишите возможный алгоритм решения этой задачи.

§ 9 Структурное программирование

9.1. Общее представление о структурном программировании

Программирование как род занятий и сфера деятельности интенсивно развивается со второй половины прошлого века. За это время сложились определённые технологии, способствующие повышению производительности труда программистов, в том числе сокращению числа ошибок, упрощению отладки, модификации и сопровождения программного обеспечения. Особенно это важно при разработке больших и сложных программных комплексов, осуществляемой усилиями целых коллективов программистов.

Одна из таких технологий — структурное программирование — была разработана ещё в начале 70-х годов прошлого века и связана с именем выдающегося нидерландского ученого Эдсгера Дейкстры (1930–2002).

Структурное программирование — технология разработки программного обеспечения, в основе которой лежит представление программы в виде иерархической структуры логически целостных фрагментов (блоков).

Перечислим некоторые принципы структурного программирования.

1. Любая программа строится из трёх базовых управляющих конструкций: последовательность, ветвление, цикл.
2. В программе базовые управляющие конструкции могут быть вложены друг в друга произвольным образом.
3. Повторяющиеся фрагменты программы можно оформить в виде подпрограмм (процедур и функций). В виде подпрограмм можно оформить логически целостные фрагменты программы, даже если они не повторяются.
4. Все перечисленные конструкции должны иметь один вход и один выход.
5. Разработка программы ведётся пошагово, методом «сверху вниз».

О методе разработки алгоритма «сверху вниз» вы получили представление в курсе информатики основной школы. Напомним его ключевые моменты на примере разработки некоторой программы.

Сначала пишется короткий текст основной программы. В ней вместо каждого логически целостного фрагмента вставляется вызов подпрограммы, которая будет выполнять этот фрагмент. Вместо настоящих, работающих, подпрограмм в программу вставляются так называемые заглушки. Как правило, они удовлетворяют требованиям интерфейса заменяемого фрагмента, но не выполняют его функций.

На следующем шаге следует убедиться, что подпрограммы вызываются в правильной последовательности, т. е. верна общая структура программы.

После этого подпрограммы-заглушки последовательно заменяются на полнофункциональные, причём разработка каждой подпрограммы ведётся тем же методом, что и основной программы. На каждом этапе проверяется, что уже созданная программа правильно работает по отношению к подпрограммам более низкого уровня.

Разработка заканчивается тогда, когда ни на одном уровне не останется ни одной заглушки. Полученная программа проверяется и отлаживается.

Такая последовательность гарантирует, что на каждом этапе разработки программист будет иметь дело с обозримым и понятным ему множеством фрагментов, осознавая, что общая структура всех более высоких уровней программы верна.

9.2. Вспомогательный алгоритм

Пример 1. Применим метод «сверху вниз» для разработки алгоритма нахождения периметра треугольника, заданного координатами своих вершин.

Пусть XA, XB, YA, YB, XC, YC — координаты вершин треугольника ABC. Его периметр — сумма длин отрезков AB, BC и AC.

Из курса геометрии вам известна формула для вычисления длины отрезка AB по координатам его концов (рис. 2.11):

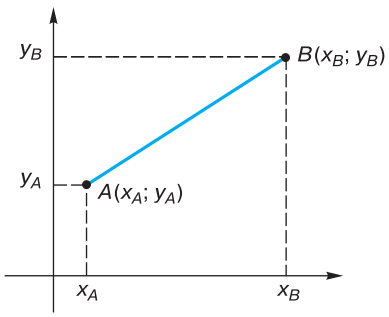


Рис. 2.11. Отрезок AB

$$d = \sqrt{(XA - XB)^2 + (YA - YB)^2}.$$

Действия по вычислению длины отрезка представляют собой логически целостный фрагмент, который целесообразно оформить в виде вспомогательного алгоритма.

Вспомогательный алгоритм — это алгоритм, целиком используемый в составе другого алгоритма.

На рисунке 2.12 представлены:

1) блок-схема алгоритма вычисления периметра треугольника, предполагающая вызов вспомогательного алгоритма Отрезок;
2) блок-схема вспомогательного алгоритма Отрезок.

При вызове вспомогательного алгоритма указываются его параметры (входные данные и результаты). Параметрами вспомогательного алгоритма Отрезок являются величины $X1$, $Y1$, $X2$, $Y2$, D. Это формальные параметры, они используются при описании алгоритма. При конкретном обращении к вспомогательному алгоритму формальные параметры заменяются фактическими параметрами, т. е. именно теми величинами, для которых будет исполнен вспомогательный алгоритм. Типы, количество и порядок следования формальных и фактических параметров должны совпадать.

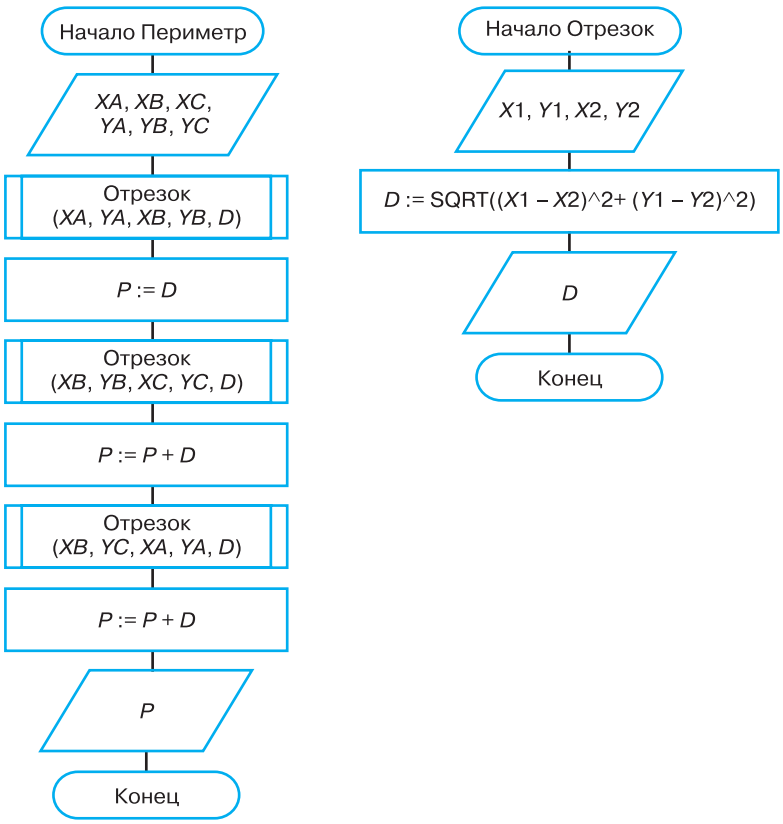


Рис. 2.12. Алгоритм вычисления периметра треугольника и вспомогательный алгоритм Отрезок

Команда вызова вспомогательного алгоритма исполняется следующим образом:

1) формальные входные данные вспомогательного алгоритма заменяются значениями фактических входных данных, указанных в команде вызова вспомогательного алгоритма;
2) для заданных входных данных исполняются команды вспомогательного алгоритма;
3) полученные результаты присваиваются переменным с именами фактических результатов;
4) осуществляется переход к следующей команде основного алгоритма.

Каким будет результат работы алгоритма при следующих исходных данных: $XA = 1$, $XB = 2$, $XC = 3$, $YA = 1$, $YB = 3$, $YC = 1$.

9.3. Рекурсивные алгоритмы

Алгоритм называется **рекурсивным**, если на каком-либо шаге он прямо или косвенно обращается сам к себе.

Пример 2. Как известно, факториал натурального числа n определяется следующим образом: $n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$; $0!$ считается равным единице $(0! = 1)$.

Иначе это можно записать так:

$F(n) = 1$ при $n \le 1$;
$F(n) = F(n - 1) \cdot n$ при $n > 1$.

В определении факториала через рекурсию имеется условие $n \le 1$, при достижении которого вызов рекурсии прекращается.

В рекурсивном определении должно присутствовать ограничение (граничное условие), при выходе на которое дальнейшая инициация рекурсивных обращений прекращается.

Пример 3. Определим функцию $S(n)$, вычисляющую сумму цифр в заданном натуральном числе n:

$S(n) = n$ при $n < 10$;
$S(n) = S(n \text{ div } 10) + n \bmod 10$ при $n \ge 10$.

Самостоятельно определите функцию $K(n)$, которая возвращает количество цифр заданного натурального числа n.

Пример 4. Алгоритм вычисления значения функции $F(n)$, где n — натуральное число, задан следующими соотношениями:

$F(n) = 1$ при $n \le 2$;
$F(n) = F(n - 1) + 3 \cdot F(n - 2)$ при $n > 2$.

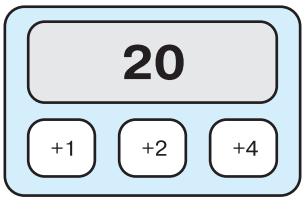
Требуется выяснить, чему равно значение функции $F(7)$.
По условию, $F(1) = F(2) = 1$.

$F(3) = F(2) + 3 \cdot F(1) = 1 + 3 \cdot 1 = 4$.
$F(4) = F(3) + 3 \cdot F(2) = 4 + 3 \cdot 1 = 7$.
$F(5) = F(4) + 3 \cdot F(3) = 7 + 3 \cdot 4 = 19$.
$F(6) = F(5) + 3 \cdot F(4) = 19 + 3 \cdot 7 = 40$.
$F(7) = F(6) + 3 \cdot F(5) = 40 + 3 \cdot 19 = 97$.

Подобные вычисления можно проводить в уме, а их результаты фиксировать в таблице:

n	1	2	3	4	5	6	7
$F(n)$	1	1	4	7	19	40	97

Пример 5. Исполнитель Плюс имеет следующую систему команд:



1) прибавь 1;
2) прибавь 2;
3) прибавь 4.

С помощью первой из них исполнитель увеличивает число на экране на 1, с помощью второй — на 2, с помощью третьей — на 4. Программа для исполнителя Плюс — это последовательность команд. Выясним, сколько разных программ, преобразующих число 20 в число 30, можно составить для этого исполнителя.

Количество программ, с помощью которых можно получить некоторое число n, будем рассматривать как функцию $K(n)$.

Число, меньшее 20, при заданных начальных условиях и системе команд исполнителя Плюс получить невозможно. Следовательно, при $n < 20$ $K(n) = 0$.

Для начального числа 20 количество программ равно 1: существует только одна пустая программа, не содержащая ни одной команды. Можем записать: $K(n) = 1$ при $n = 20$.

Любое число $n > 20$ может быть получено из чисел $n - 1$, $n - 2$ и $n - 4$ одной из трёх команд, входящих в систему команд исполнителя — «прибавь 1», «прибавь 2» и «прибавь 4» соответственно. При этом каждая программа получения из исходного числа чисел $n - 1$, $n - 2$ и $n - 4$ удлинится на одну команду и будет приводить к числу n. Следовательно, $K(n) = K(n - 1) + K(n - 2) + K(n - 4)$.

Запишем все соотношения, определяющие функцию $K(n)$:

$K(n) = 0$ при $n < 20$;

$K(n) = 1$ при $n = 20$;

$K(n) = K(n - 1) + K(n - 2) + K(n - 4)$ при $n > 20$.

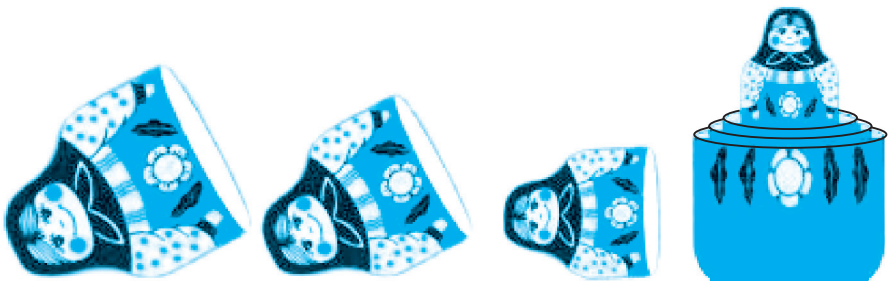
Заполним по этой формуле таблицу для всех значений n от 20 до 30:

n	20	21	22	23	24	25	26	27	28	29	30
$K(n)$	1	1	2	3	6	10	18	31	55	96	169

Итак, существует 169 различных программ, с помощью которых исполнитель Плюс может преобразовать число 20 в 30.

Любой объект, который частично определяется через самого себя, называется рекурсивным. Нас окружает множество рекурсивных объектов. Приведём примеры только некоторых из них.

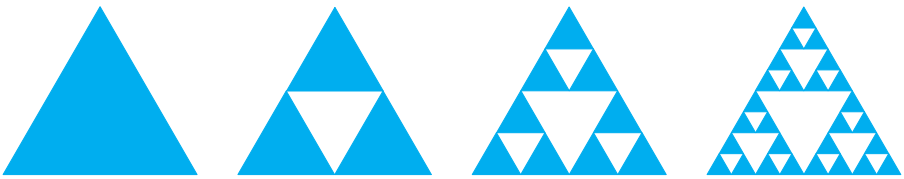
1. Матрёшка — русская деревянная игрушка в виде расписной куклы, внутри которой находятся подобные ей куклы меньшего размера.



2. Два зеркала, поставленные друг напротив друга, — в них образуются два коридора из затухающих отражений. Это, например, можно наблюдать в спальном железнодорожном вагоне.

3. Примером рекурсивной структуры является замечательное стихотворение Р. Бернса «Дом, который построил Джек» в переводе С. Маршака.

4. Рекурсивную природу имеют геометрические фракталы. На рисунке представлено построение одного из геометрических фракталов — треугольника Серпинского. Чтобы его получить, нужно взять равносторонний треугольник с внутренней областью, провести в нём средние линии и «выкинуть» центральный из четырёх образовавшихся маленьких треугольников. Дальше эти же действия нужно повторить с каждым из оставшихся трёх треугольников, и т. д.



9.4. Запись вспомогательных алгоритмов на языке Pascal

Запись вспомогательных алгоритмов в языках программирования осуществляется с помощью подпрограмм. В Паскале различают два вида подпрограмм: процедуры и функции.

Процедура — подпрограмма, имеющая произвольное количество входных и выходных данных.

Описание процедуры имеет вид:

```
procedure <имя_процедуры>(<описание параметров-значений>;
          var: <описание параметров-переменных>);
begin
  <операторы>
end;
```

В заголовке процедуры после её имени приводится перечень формальных параметров и их типов. Для вызова процедуры достаточно указать её имя со списком фактических параметров. При этом между фактическими и формальными параметрами должно быть полное соответствие по количеству, порядку следования и типу.

Пример 6. Запишем на языке Pascal программу нахождения периметра треугольника, заданного координатами его вершин. Вспомогательный алгоритм оформим с помощью процедуры.

```
program perimetr;
var xa,ya,xb,yb,xc,yc,d,p: real;
procedure otrezok(x1,y1, x2,y2: real; var d: real);
begin
  d:=sqrt(sqr(x1-x2)+sqr(y1-y2));
end;
begin
  p:=0;
  writeln('Ввод координат концов отрезка:');
  writeln('xa, ya');
  read(xa,ya);
  writeln('xb, yb');
  read(xb,yb);
  writeln('xc, yc');
  read(xc,yc);
  otrezok(xa,ya,xb,yb,d);
  p:=p+d;
  otrezok(xa,ya,xc,yc,d);
  p:=p+d;
  otrezok(xc,yc,xb,yb,d);
  p:=p+d;
  writeln('p=',p)
end.
```

Выполните программу на компьютере.

Подумайте, каким образом можно модифицировать программу, чтобы вычислять с её помощью периметр *n*-угольника. Каким образом при решении этой задачи можно использовать массивы?

Функция — подпрограмма, имеющая единственный результат, записываемый в ячейку памяти, имя которой совпадает с именем функции.

Описание функции имеет вид:

```
function  <имя_функции>(<описание входных данных>):
          <тип_функции>);
begin
  <операторы>
end;
```

В заголовке функции после её имени приводится описание входных данных — указывается перечень формальных параметров

и их типов. Там же указывается тип самой функции, т. е. тип результата. В блоке функции обязательно должен присутствовать оператор

```
<имя_функции> := <результат>;
```

Для вызова функции достаточно указать её имя со списком фактических параметров в любом выражении, в условиях (после слов if, while, until) или в операторе write главной программы.

Пример 7. Запишем на языке Pascal программу нахождения периметра треугольника, заданного координатами его вершин. Вспомогательный алгоритм оформим с помощью функции.

```
program perimetr;
  var xa,ya,xb,yb,xc,yc,p: real;
function d(x1,y1, x2,y2: real): real;
begin
  d:=sqrt(sqr(x1-x2)+sqr(y1-y2));
end;
begin
  writeln('Ввод координат концов отрезка:');
  writeln('xa, ya');
  read(xa,ya);
  writeln('xb, yb');
  read(xb,yb);
  writeln('xc, yc');
  read(xc,yc);
  p:=p+d(xa,ya,xb,yb)+d(xa,ya,xc,yc)+d(xc,yc,xb,yb);
  writeln('p=',p)
end.
```

Выполните программу на компьютере.

На основе этой программы напишите функцию, вычисляющую площадь треугольника по целочисленным координатам его вершин. Используйте эту функцию для вычисления площади n-угольника.

САМОЕ ГЛАВНОЕ

Структурное программирование — технология разработки программного обеспечения, в основе которой лежит представление программы в виде иерархической структуры логически целостных фрагментов (блоков).

Основные принципы структурного программирования заключаются в том, что:

1) любая программа строится из трёх базовых управляющих конструкций: последовательность, ветвление, цикл;
2) в программе базовые управляющие конструкции могут быть вложены друг в друга произвольным образом;
3) повторяющиеся фрагменты программы можно оформить в виде подпрограмм (процедур и функций). В виде подпрограмм можно оформить логически целостные фрагменты программы, даже если они не повторяются;
4) все перечисленные конструкции должны иметь один вход и один выход;
5) разработка программы ведётся пошагово, методом «сверху вниз».

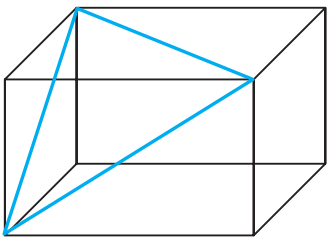
Вспомогательный алгоритм — это алгоритм, целиком используемый в составе другого алгоритма.

Алгоритм называется рекурсивным, если на каком-либо шаге он прямо или косвенно обращается сам к себе.

Запись вспомогательных алгоритмов в языках программирования осуществляется с помощью подпрограмм. В Паскале различают два вида подпрограмм: процедуры и функции.

Вопросы и задания

1. В чём заключается сущность структурного программирования? Какие преимущества обеспечивает эта технология?
2. Какой алгоритм называется вспомогательным?
3. Вспомните, в чём состоит суть метода последовательного построения (уточнения) алгоритма. Как он называется иначе?
4. Опишите основные шаги разработки программы методом «сверху вниз».
5. Дан прямоугольный параллелепипед, длины рёбер которого равны a, b и c. Требуется определить периметр треугольника, образованного диагоналями его граней. Какой алгоритм целесообразно использовать при решении этой задачи в качестве вспомогательного?



6. Какой вспомогательный алгоритм называется рекурсивным? Что такое граничное условие и каково его назначение в рекурсивном алгоритме?

7. Алгоритм вычисления значения функции $F(n)$, где n — натуральное число, задан следующими соотношениями:

$$F(n) = 2 \text{ при } n \le 0;$$
$$F(n) = F(n-2) + F(n-1) + F(n \text{ div } 2) \text{ при } n > 0.$$

Требуется выяснить, чему равно значение функции $F(10)$.

8. Исполнитель Калькулятор имеет следующую систему команд:
   1) прибавь 1;
   2) умножь на 2.

   С помощью первой из них исполнитель увеличивает число на экране на 2, с помощью второй — в 2 раза.
   1) Выясните, сколько разных программ, преобразующих число 1 в число 20, можно составить для этого исполнителя.
   2) Сколько среди них таких программ, у которых в качестве промежуточного результата обязательно получается число 15?
   3) Сколько среди них таких программ, у которых в качестве промежуточного результата никогда не получается число 12?

9. Попробуйте найти рекурсивные синтаксические структуры:
   1) в поэме А. Блока «Двенадцать»;
   2) в стихотворении М. Лермонтова «Сон»;
   3) в романе М. Булгакова «Мастер и Маргарита»;
   4) в фольклоре.

10. Найдите информацию о таких геометрических фракталах, как Снежинка Коха, Т-квадрат, Н-фрактал, кривая Леви, Драконова ломаная.

11. Напишите программу вычисления значения функции $F(n)$, рассмотренной в примере 4 этого параграфа. Вычислите с её помощью значение функции F(7).

12. Напишите программу вычисления $C_n^k = \dfrac{n!}{(n-k)! \cdot k!}$. Используйте подпрограмму.
13. Дана программа:

```
program rek;
procedure F(n: integer);
```

```
begin
  if n>0 then
  begin
    F(n-4);
    writeln(n);
    F(n div 3)
  end;
end;
begin
  F(9)
end.
```

Не выполняя программу на компьютере, выясните, что получится в результате работы этой программы.

Проверьте свой результат, выполнив программу на компьютере.

Дополнительные материалы к главе смотрите в авторской мастерской.

Глава 3
ИНФОРМАЦИОННОЕ МОДЕЛИРОВАНИЕ

§ 10
Модели и моделирование

10.1. Общие сведения о моделировании

Человек стремится познать объекты (предметы, процессы или явления) окружающего мира, т. е. понять, как устроен конкретный объект, каковы его структура, основные свойства, законы развития и взаимодействия с другими объектами. При этом зачастую исследуются не сами объекты, а их модели.

Из курса информатики основной школы вам известно, что:

- модель — это новый объект, который имеет свойства данного объекта, существенные для определённого исследования;
- моделирование — метод познания, заключающийся в создании и исследовании моделей;
- натурная (материальная) модель — реальный предмет, в уменьшенном или увеличенном виде воспроизводящий внешний вид, структуру или поведение моделируемого объекта;
- информационная модель — описание объекта-оригинала на одном из языков кодирования информации;
- по форме представления можно выделить знаковые, образные и смешанные информационные модели;
- для создания информационной модели объекта необходимо:
  1) выяснить цель моделирования;
  2) выделить свойства объекта-оригинала, существенные с точки зрения цели моделирования;

3) установить взаимосвязи между значениями выбранных свойств и выразить их в некоторой форме (словесно, таблицей, графиком, функцией, уравнением, неравенством, системой и т. п.).

Модель — общенаучное понятие; моделирование имеет место в любых областях знания и сферах человеческой деятельности.

Приведите примеры моделей, с которыми вы встречались на уроках физики, химии, биологии, истории, математики, обществознания, литературы.

В информатике рассматриваются общие подходы к созданию и использованию информационных моделей, связанные с использованием компьютерной техники.

10.2. Компьютерное моделирование

Информационные модели, реализованные с помощью систем программирования, электронных таблиц, специализированных математических пакетов или программных средств для моделирования, называются **компьютерными моделями**.

Компьютерное моделирование включает в себя процесс реализации информационной модели на компьютере и исследование с помощью этой модели объекта моделирования — проведение вычислительного эксперимента.

С помощью компьютерного моделирования решаются многие научные и производственные задачи: прогнозирование погоды и климатических изменений; конструирование транспортных средств и дизайн лекарственных препаратов; стратегическое управление организациями и прогнозирование цен на финансовых рынках; прогнозирование прочности конструкций и исследование поведения зданий, конструкций и деталей под механической нагрузкой; многие другие задачи.

Рассмотрим основные этапы компьютерного моделирования более подробно (рис. 3.1).

На первом этапе в результате анализа условия задачи определяется объект моделирования и цель создания модели. После этого в объекте моделирования выделяются параметры (свойства,

Рис. 3.1. Основные этапы компьютерного моделирования

основные части), существенные с точки зрения поставленной цели. Далее уточняется, какие результаты и в каком виде должны быть получены, а также какие исходные данные для этого нужны.

На втором этапе определяются параметры модели и связи между ними; приводится математическое описание зависимостей между параметрами модели.

На третьем этапе выбирается или разрабатывается алгоритм получения из исходных данных результатов, подбираются программные средства реализации алгоритма на компьютере и создаётся компьютерная модель.

На четвёртом этапе осуществляется работа непосредственно с полученной компьютерной моделью. Сначала на заранее разработанных тестах (наборах исходных данных, для которых заранее известны результаты) осуществляется проверка правильности (тестирование) модели, и при необходимости модель дорабатывается. После тестирования, когда есть уверенность в правильности функционирования модели, переходят непосредственно к компьютерному эксперименту — целенаправленным действиям пользователя над компьютерной моделью. В ходе такого экспериментирования сознательно изменяются условия функционирования модели и накапливаются данные о её «поведении». В процессе проведения эксперимента может выясниться, что нужно усовершенствовать или изменить используемый алгоритм, уточнить информационную модель или внести изменения в постановку задачи. В таких

случаях происходит возвращение к соответствующему этапу, и процесс начинается снова.

На пятом этапе результаты эксперимента анализируются, на их основе делаются выводы о моделируемом объекте. На основе всестороннего анализа полученных результатов принимается некоторое решение, что и является конечной целью моделирования.

Компьютерное моделирование даёт возможность:

- существенно расширить круг исследуемых объектов (моделирование прошлого и будущего, несуществующего или невоспроизводимого в реальных условиях);
- исследовать процессы в развитии, при необходимости ускоряя или замедляя их и проводя эксперименты многократно;
- находить оптимальные решения без затрат на изготовление пробных экземпляров;
- проводить эксперименты без риска негативных последствий для здоровья человека или окружающей среды;
- визуализировать получаемые результаты.

10.3. Списки, графы, деревья и таблицы

Между данными, используемыми в той или иной информационной модели, всегда существуют некоторые связи, определяющие ту или иную структуру данных.

Вспомните, как мы определяли структуру данных при рассмотрении алгоритмов и программ. О каких информационных моделях тогда шла речь? С какими структурами данных вы сталкивались в программировании?

Различают линейные и нелинейные структуры данных.

В курсе информатики основной школы вы познакомились с **линейным односвязным списком** — последовательностью линейно связанных элементов, для которых разрешены операции добавления элемента в произвольное место списка и удаление любого элемента. Связь элементов списка осуществляется за счёт того, что каждый элемент списка содержит кроме данных адрес элемента, следующего за ним в списке. В линейном списке для каждого элемента, кроме первого, есть предыдущий элемент; для каждого элемента, кроме последнего, есть следующий элемент.

Частным случаем линейного односвязного списка является **стек** — последовательность, в которой включение и исключение

элементов осуществляются с одной и той же стороны этой последовательности.

Ещё одним частным случаем линейного односвязного списка является **очередь** — последовательность, у которой включение элементов производится с одной стороны последовательности, а исключение — с другой. Сторона, где происходит включение элементов, называется хвостом; сторона, где происходит исключение — головой. Понятие очереди как структуры данных очень близко к бытовому понятию «очередь» (рис. 3.2).

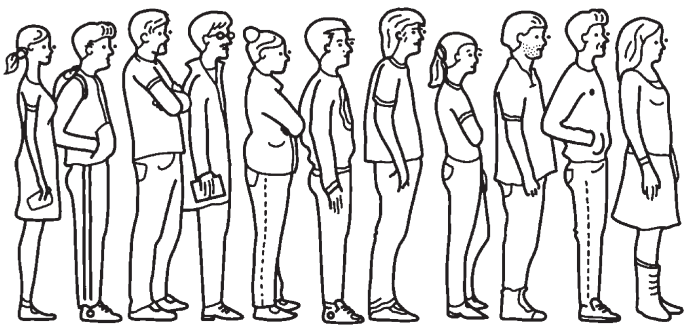
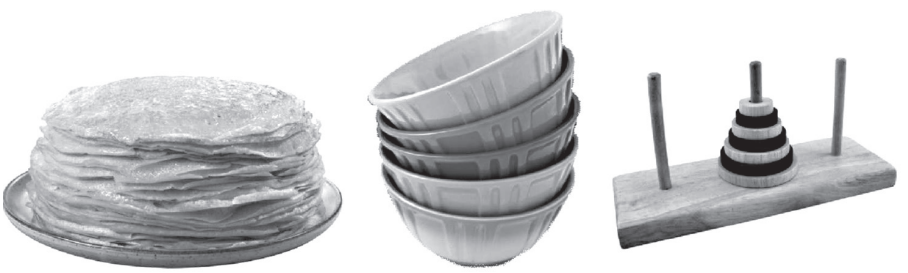


Рис. 3.2. Иллюстрация понятия «очередь»

Подумайте, какая связь между стеком и следующими объектами:



Почему стек является структурой типа LIFO (от англ. *Last In, Firts Out* — последним пришёл, первым ушёл)?

Почему очередь является структурой типа FIFO (от англ. *First In, First Out* — первым пришёл, первым ушёл)?

Примеры нелинейных структур данных вам также хорошо известны — это графы и деревья (рис. 3.3).

Граф — это множество элементов (вершин графа) вместе с набором отношений между ними.

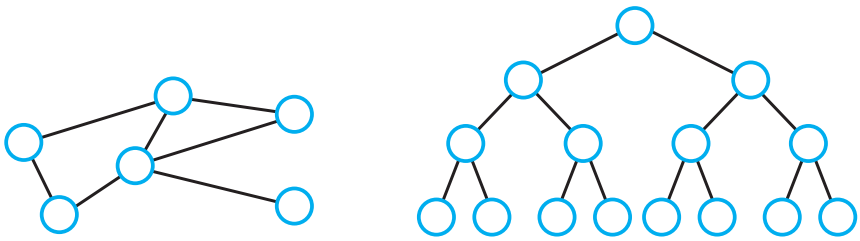


Рис. 3.3. Примеры графовых структур

Граф является многосвязной структурой, обладающей следующими свойствами:

1) на каждый элемент может быть произвольное количество ссылок;
2) каждый элемент может иметь связь с любым количеством других элементов;
3) каждая связка может иметь направление и вес.

Ненаправленная (без стрелки) линия, соединяющая вершины графа, называется **ребром**. Линия направленная (со стрелкой) называется **дугой**. При этом вершина, из которой дуга исходит, называется начальной, а вершина, куда дуга входит, — конечной. Граф называется **неориентированным**, если его вершины соединены рёбрами. Вершины **ориентированного** графа соединены дугами. Граф называется **взвешенным**, если его вершины или рёбра характеризуются некоторой дополнительной информацией — **весами** вершин или рёбер.

Графы являются основным средством для описания структур сложных объектов. С их помощью можно описать вычислительную сеть, транспортную систему, схему авиалиний и другие объекты.

Одной из разновидностей графа является дерево.

Дерево — это совокупность элементов (вершин), в которой выделен один элемент (корень), а остальные элементы разбиты на непересекающиеся множества (поддеревья). Каждое поддерево является деревом, а его корень является потомком корня дерева, т. е. все элементы связаны между собой отношением «предок — потомок». В результате образуется иерархическая структура вершин.

Частным случаем дерева является **бинарное дерево**, в котором каждая вершина может иметь не более двух потомков.

Деревья используются для представления родственных связей (генеалогическое дерево), для определения выигрышной стратегии в играх и т. д.

Ещё одной знакомой вам структурой данных являются **таблицы**, состоящие из строк и граф (столбцов, колонок), пересечение которых образуют ячейки. Таблицы применяют для наглядности и удобства сравнения показателей.

Оформляют таблицы в соответствии с рисунком 3.4.

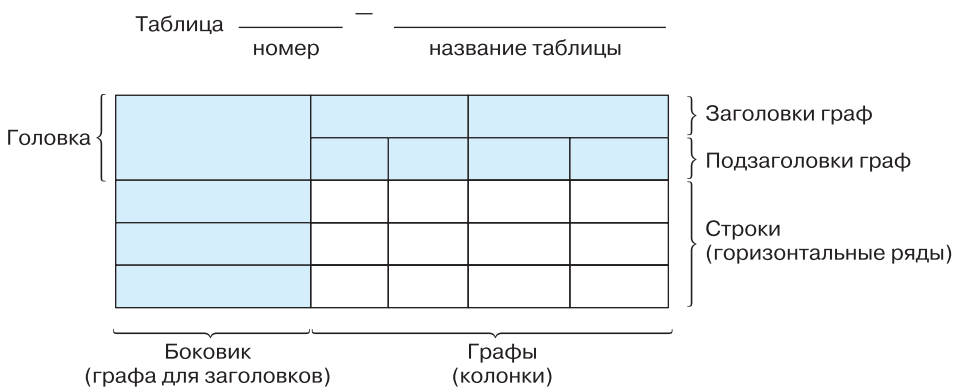


Рис. 3.4. Оформление таблицы

Название таблицы, при его наличии, должно отражать её содержание, быть точным, кратким. Название следует помещать над таблицей.

Заголовки граф и строк таблицы следует писать с прописной буквы, а подзаголовки граф — со строчной буквы, если они составляют одно предложение с заголовком, или с прописной буквы, если они имеют самостоятельное значение. В конце заголовков и подзаголовков таблицы точки не ставят. Заголовки и подзаголовки граф указывают в единственном числе.

Если все показатели, приведенные в графах таблицы, выражены в одной и той же единице физической величины, то её обозначение необходимо помещать над таблицей справа. Если в графе таблицы помещены значения одной и той же физической величины, то обозначение единицы физической величины указывают в заголовке (подзаголовке) этой графы.

Эти и другие требования к оформлению таблиц содержатся в ГОСТ 2.105–95 «ЕСКД. Общие требования к оформлению текстовых документов».

В курсе информатики основной школы вы познакомились с таблицами типа:

- «объект — свойство», содержащими информацию о свойствах отдельных объектов, принадлежащих одному классу;

- «объект — объект», содержащими информацию о некотором одном свойстве пар объектов, принадлежащих одному или разным классам.

Таблицы, в которых отражено наличие или отсутствие связей между отдельными элементами некоторой системы, называются двоичными матрицами.

Вспомните и приведите примеры таблиц типа «объект — свойство», «объект — объект», отражающих не только количественные, но и качественные характеристики свойств (двоичные матрицы).

Табличный способ представления данных является универсальным — любую структуру данных, в том числе и представленную в форме графа, можно свести к табличной форме. Это тем более важно в связи с тем, что для компьютерной обработки табличное представление данных является предпочтительным.

Пример 1. Построим таблицу, соответствующую неориентированному графу (рис. 3.5), отражающему схему дорог между некоторыми населёнными пунктами.

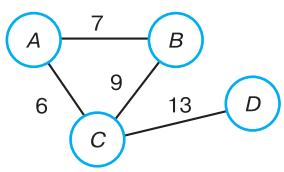


Рис. 3.5. Граф схемы дорог

Строки и столбцы таблицы будут соответствовать вершинам графа. Если две вершины являются смежными (соединены ребром), то в ячейку на пересечении соответствующих столбца и строки будем записывать вес этого ребра. В противном случае (вершины не являются смежными) в ячейку будем записывать 0. Получится таблица типа «объект — объект».

Такую таблицу называют матрицей смежности. Часто в матрицах смежности вместо нуля ставят знак минус, что обеспечивает бóльшую наглядность.

	A	*B*	*C*	*D*
A	0	7	6	0
B	7	0	9	0
C	6	9	0	13
D	0	0	13	0

	A	*B*	*C*	*D*
A	–	7	6	–
B	7	–	9	–
C	6	9	–	13
D	–	–	13	–

Матрица смежности неориентированного графа симметрична относительно главной диагонали, идущей от левого верхнего угла к правому нижнему углу. У матрицы смежности неориентированного графа такая симметрия отсутствует.

Пример 2. Обед в школьной столовой состоит из двух блюд и напитка. На первое можно выбрать щи или окрошку, на второе — плов или пельмени, на третье — сок или компот. Все возможные варианты представлены с помощью дерева на рисунке 3.6.

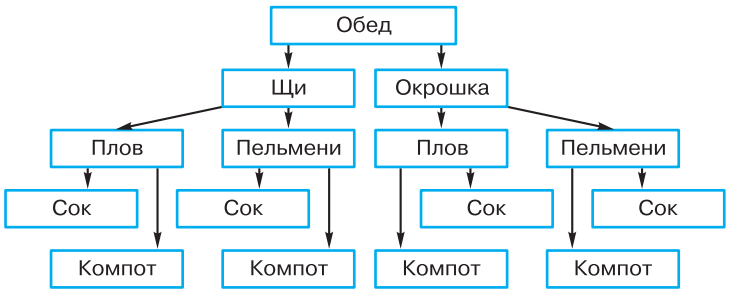


Рис. 3.6. Дерево вариантов обеда

Для того чтобы представить эту же информацию в таблице, будем двигаться по дереву от листьев к корню, описывая все возможные варианты обеда.

Обед	Напиток	2-е блюдо	1-е блюдо
Вариант 1	Сок	Плов	Щи
Вариант 2	Компот	Плов	Щи
Вариант 3	Сок	Пельмени	Щи
Вариант 4	Компот	Пельмени	Щи
Вариант 5	Компот	Плов	Окрошка
Вариант 6	Сок	Плов	Окрошка
Вариант 7	Компот	Пельмени	Окрошка
Вариант 8	Сок	Пельмени	Окрошка

Получилась таблица типа «объект–свойства»: объектами в ней являются варианты обеда, а свойствами — составляющие его блюда. При этом число граф в полученной таблице соответствует числу уровней в дереве.

При решении класса задач, связанного с нахождением кратчайшего пути в ориентированном графе, можно:
1) от исходного графа перейти к матрице смежности;
2) по матрице смежности построить дерево решений;
3) по дереву решений выбрать подходящий вариант.

Пример 3. Найдём кратчайший путь от вершины *A* до вершины *F* в графе, приведённом на рисунке 3.7.

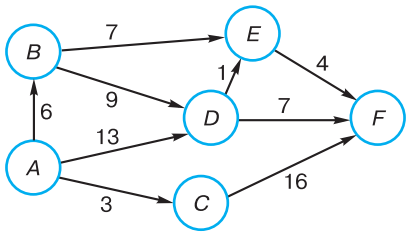


Рис. 3.7. Ориентированный граф

Составим матрицу смежности, соответствующую данному ориентированному графу:

	A	*B*	*C*	*D*	*E*	*F*
A	–	6	3	13	–	–
B	–	–	–	9	7	–
C	–	–	–	–	–	16
D	–	–	–	–	1	7
E	–	–	–	–	–	4
F	–	–	–	–	–	–

По матрице смежности построим полное дерево перебора решений — рисунок 3.8.

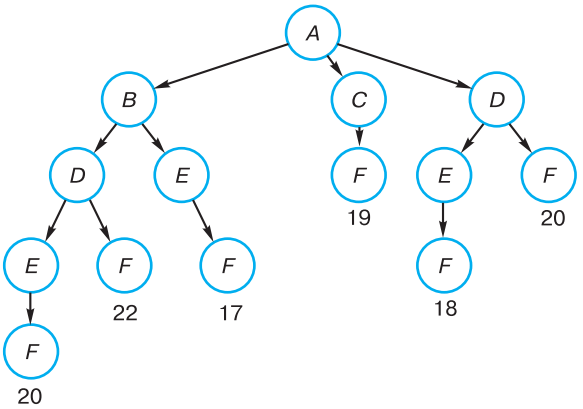


Рис. 3.8. Полное дерево перебора решений

На рисунке 3.8 видно, что кратчайший путь из вершины A в вершину F равен 17 и имеет вид A–B–E–F.

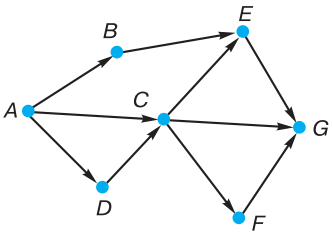


Рис. 3.9. Схема дорог

Пример 4. На рисунке 3.9 представлена схема дорог, связывающих города A, B, C, D, E, F, G. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько разных путей существует из города A в город G?

Существует несколько способов решения этой задачи. Рассмотрим их.

Вариант 1. По графу можно построить матрицу смежности, а на её основе построить дерево, корнем которого будет служить вершина A. Число листьев построенного дерева будет равно числу дорог из города A в город G.

Постройте дерево и подсчитайте число дорог из города A в город G самостоятельно.

Вариант 2. Пусть K_X — число путей из города A в город X. Начнем считать число путей с конца маршрута. Так как в город G есть дороги из городов C, E, F, то $K_G = K_C + K_E + K_F$.

В свою очередь $K_C = 1 + K_D = 1 + 1 = 2$,

$K_E = K_B + K_C = 1 + 2 = 3$, $K_F = K_C = 2$.

Таким образом, $K_G = 2 + 3 + 2 = 7$.

Вариант 3. Можно считать число путей с начала маршрута. При этом процесс подсчёта удобно изображать на самом графе — рисунок 3.10.

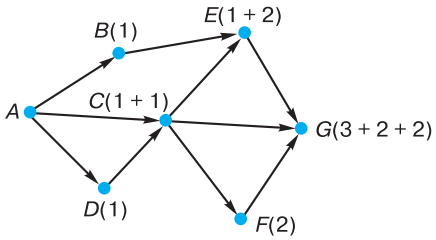
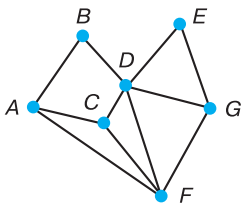


Рис. 3.10. Схема дорог с подсчётом числа путей

Пример 5. На рисунке 3.11 представлена схема дорог, связывающих населённые пункты A, B, C, D, E, F, G. В таблице содержатся сведения о длинах этих дорог (в километрах). Схему и таблицу создавали независимо друг от друга, поэтому в них используются разные обозначения. Необходимо выяснить длину пути в километрах из пункта D в пункт F.



	Г1	Г2	Г3	Г4	Г5	Г6	Г7
Г1		15	10	20			25
Г2	15			25			10
Г3	10					30	20
Г4	20	25			15		
Г5				15			10
Г6			30				15
Г7	25	10	20		10	15	

Рис. 3.11. Схема дорог и таблица их длин

Рассмотрим имеющийся граф и выясним степень каждой вершины — число рёбер соединяющих некоторую вершину с другими вершинами. Получим:

A	B	C	D	E	F	G
3	2	3	5	2	4	3

На основании имеющейся таблицы мы также можем сделать выводы о том, сколькими дорогами соединён тот или иной населённый пункт с другими населёнными пунктами:

Г1	Г2	Г3	Г4	Г5	Г6	Г7
4	3	3	3	2	2	5

Сопоставив полученную информацию, можем сказать, что через Г1 в таблице обозначен населённый пункт F, а через Г7 — D. Согласно таблице, расстояние между этими пунктами равно 25 км.

САМОЕ ГЛАВНОЕ

Модель — это новый объект, который имеет свойства данного объекта, существенные для определённого исследования. Моделирование — метод познания, заключающийся в создании и исследовании моделей. Информационная модель — описание объекта-оригинала на одном из языков кодирования информации.

В информатике рассматриваются общие подходы к созданию и использованию информационных моделей, связанные с использованием компьютерной техники.

Информационные модели, реализованные с помощью систем программирования, электронных таблиц, специализированных математических пакетов или программных средств для моделирования, называются компьютерными моделями. Компьютерное моделирование включает в себя процесс реализации информационной модели на компьютере и исследование с помощью этой модели объекта моделирования — проведение вычислительного эксперимента.

Между данными, используемыми в той или иной информационной модели, всегда существуют некоторые связи, определяющие ту или иную структуру данных. Различают линейные и нелинейные структуры данных.

Линейный односвязный список — последовательность линейно связанных элементов, для которых разрешены операции добавления элемента в произвольное место списка и удаление любого элемента. Частным случаем линейного односвязного списка является стек — последовательность, в которой включение и исключение элементов осуществляются с одной стороны этой последовательности. Ещё один частный случай линейного односвязного списка — очередь, представляющая собой последовательность,

у которой включение элементов производится с одной стороны последовательности, а исключение — с другой.

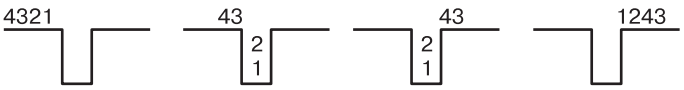
Примерами нелинейных структур данных являются графы и деревья. Граф — это множество элементов (вершин графа) вместе с набором отношений между ними, называемых рёбрами (дугами) графа. Дерево — это совокупность элементов (вершин), в которой выделен один элемент (корень), а остальные элементы разбиты на непересекающиеся множества (поддеревья). Каждое поддерево является деревом, а его корень является потомком корня дерева, т. е. все элементы связаны между собой отношением «предок — потомок». Частным случаем дерева является бинарное дерево, в котором каждая вершина может иметь не более двух потомков.

Таблица — это структура данных, состоящая из строк и граф (столбцов, колонок), пересечение которых образуют ячейки. Таблицы применяют для наглядности и удобства сравнения показателей. Табличный способ представления данных является универсальным — любую структуру данных, в том числе и представленную в форме графа, можно свести к табличной форме. Это тем более важно в связи с тем, что для компьютерной обработки табличное представление данных является предпочтительным.

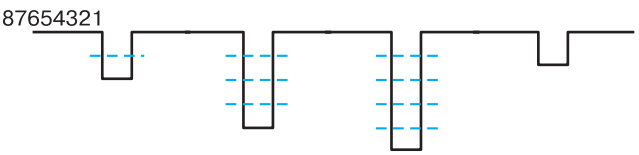
Вопросы и задания

1. Что такое модель? Что такое моделирование? В каких областях науки и техники оно применяется?
2. Какие модели называются натурными? Приведите примеры натурных моделей.
3. Какие модели называются информационными? Приведите примеры информационных моделей. Какова роль информатики в информационном моделировании?
4. Создайте информационную модель одной из комнат вашей квартиры с целью оклейки её обоями. Представьте информационную модель в знаковой и графической формах.
5. Какие модели называются компьютерными информационными моделями?
6. Опишите основные этапы компьютерного моделирования.
7. Приведите примеры линейных структур данных. Чем очередь отличается от стека?
8. Муравьи идут друг за другом по неровной лесной тропе. На их пути встречаются ямки, в которые могут провалиться несколько муравьёв. Когда ямка заполняется муравьями,

остальные муравьи проходят через неё, а затем по одному вытаскивают провалившихся. Например, вот как четыре муравья проходят через ямку, вмещающую двух муравьёв:



Пусть по тропе идут 8 муравьёв. В каком порядке они будут идти после преодоления участка с четырьмя ямками, вмещающими 2, 4, 5 и 1 муравья соответственно?



Какую структуру данных иллюстрирует данный пример?[1)]

9. Выясните, что представляет собой обратная польская запись, и вычислите значение записанного с её помощью выражения: 1 2 + 3 × 4 5 × +.
10. Что такое граф? Какой граф называется ориентированным? Какой граф называется неориентированным? Какой граф называется взвешенным? Приведите примеры.
11. Что такое дерево? Какое дерево называется бинарным? Приведите примеры.
12. Почему графы и деревья считаются многоуровневыми структурами данных?
13. Информация о родственных связях в некоторой семье представлена следующим образом:

    parent(Юрий, Пётр); parent(Анна, Ева);
    parent(Ирина, Георгий); parent(Маргарита, Анна);
    parent(Анна, Николай); parent(Пётр, Георгий);
    parent(Михаил, Николай); parent(Маргарита, Пётр);
    parent(Юрий, Анна); parent(Маргарита, Александр);
    parent(Дарья, Руслан); parent(Александр, Руслан);
    parent(Михаил, Ева); parent(Юрий, Александр).

    Запись parent(*A*, *B*) означает, что *A* является родителем *B*. Нарисуйте генеалогическое древо этой семьи. Сколько у Ирины племянников и племянниц?

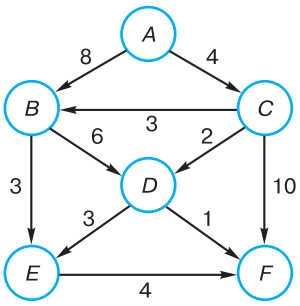
---

1) По материалам международного конкурса по информатике «Бобёр» (**bebras.ru**).

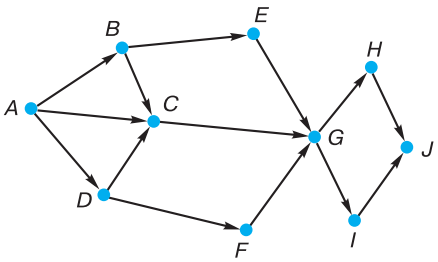
14. В кладовке хранятся ёлочные игрушки — большие и маленькие красные и золотые шары и звёзды. При этом игрушки разного размера, цвета и формы хранятся в отдельных коробках. Например, в одной коробке — большие красные звёзды, в другой — маленькие красные звёзды и т. д. Известно, что среди игрушек нет ни маленьких шаров, ни маленьких золотых звёзд. Всего звёзд 25, а шаров — 17. Всего больших игрушек — 32; красных игрушек — 28. Золотых звёзд на 2 больше, чем золотых шаров. В скольких коробках хранятся игрушки? Сколько игрушек в каждой коробке?

    Постройте граф, представляющий состав игрушек. Используйте его для решения задачи. Представьте эту же информацию в табличной форме.
15. Что с вашей точки зрения более наглядно представляет структуру системы: граф или таблица? Какая форма представления информации предпочтительна для компьютерной обработки данных?
16. Решите следующую задачу, составив двоичную матрицу.

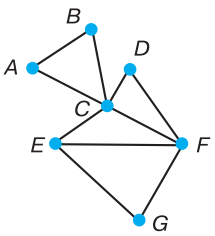
    Ваня, Кирилл, Петя и Саша учатся в 5, 6, 7 и 8 классах. Как-то они отправились в лес за белыми грибами. Шестикласснику не повезло — он не нашёл ни одного гриба, а Петя с пятиклассником нашли много грибов. Ваня и семиклассник нашли куст малины и позвали Кирилла полакомиться ягодами. Восьмиклассник, шестиклассник и Кирилл объясняли Саше, как ориентироваться на местности. В каком классе учится каждый из мальчиков?
17. Как осуществляется переход от ориентированного графа к дереву решений?
18. Найдите кратчайший путь от вершины A до вершины F в ориентированном графе:



19. На рисунке представлена схема дорог, связывающих города A, B, C, D, E, F, G, H, I, J. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько разных путей существует из города A в город J?



20. На рисунке представлена схема дорог, связывающих населённые пункты A, B, C, D, E, F, G. В таблице содержатся сведения о длинах этих дорог (в километрах). Схему и таблицу создавали независимо друг от друга, поэтому в них используются разные обозначения. Необходимо выяснить длину пути в километрах из пункта E в пункт F.



	Г1	Г2	Г3	Г4	Г5	Г6	Г7
Г1		9		2			
Г2	9			8		11	
Г3					3	12	
Г4	2	8				4	7
Г5			3			11	
Г6		11	12	4	11		9
Г7				7		9	

§ 11
Моделирование на графах

11.1. Алгоритмы нахождения кратчайших путей между вершинами графа

Графы как информационные модели находят широкое применение во многих сферах нашей жизни. Например, с их помощью

можно планировать оптимальные транспортные маршруты, кратчайшие объездные пути, расположение торговых точек и других объектов. Необходимость решения задач, связанных с поиском кратчайшего пути на графе, возникает при проектировании инженерных сетей и линий электропередач, в микроэлектронике и во многих других случаях.

Путь между вершинами A и B графа считается кратчайшим, если:

- эти вершины соединены минимальным числом ребер (в случае, если граф не является взвешенным);
- сумма весов рёбер, соединяющих эти вершины, минимальна (для взвешенного графа).

Есть множество алгоритмов определения кратчайшего пути между вершинами графа, в том числе:

1) алгоритм построения дерева решений;
2) алгоритм Дейкстры;
3) метод динамического программирования.

Алгоритм построения дерева решений, как правило, используется для нахождения кратчайшего пути в ориентированном графе. Его мы рассмотрели в предыдущем параграфе.

Алгоритм Дейкстры служит для нахождения кратчайшего пути между одной конкретной вершиной (источником) и всеми остальными вершинами графа.

Суть алгоритма состоит в следующем. Каждой вершине графа ставится в соответствие метка — минимальное известное расстояние от источника до этой вершины. Метка самого источника полагается равной 0. Алгоритм работает пошагово — на каждом шаге он «посещает» одну вершину и пытается уменьшать метки.

На первом шаге расстояние от источника до всех остальных вершин неизвестно. Метки вершин (кроме источника) считаются равными бесконечности, все вершины считаются непосещёнными.

Далее, из всех непосещённых вершин выбирается вершина, имеющая минимальную метку. Для каждого из соседей этой вершины (кроме отмеченных как посещённые) рассчитывается новая длина пути, как сумма значений текущей метки этой вершины и длины ребра, соединяющего её с соседом. Если полученное значение длины меньше значения метки соседа, то значение метки заменяется полученным значением длины. После рассмотрения всех соседей вершина помечается как посещённая. Этот шаг алгоритма повторяется, пока есть непосещённые вершины. Работа алгоритма завершается, когда все вершины посещены.

Рассмотрим работу алгоритма на примере. На рисунке 3.12 кружками обозначены вершины графа, в кружки вписаны имена вершин. Вершины соединены линиями — рёбрами графа. Около каждого ребра обозначен его «вес» — длина пути. Рядом с каждой вершиной дана метка — длина кратчайшего пути в эту вершину из вершины A: для вершины A — это 0, для всех других вершин она неизвестна и обозначена знаком «бесконечность».

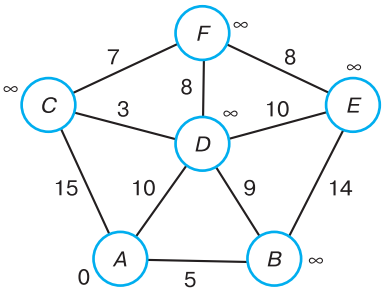


Рис. 3.12. Алгоритм Дейкстры. Начальное состояние

Минимальную метку (0) имеет вершина A. Её соседи — вершины B, C, D. Очерёдность рассмотрения соседей: B, D, C. После изменения их меток получим результат, представленный на рисунке 3.13.

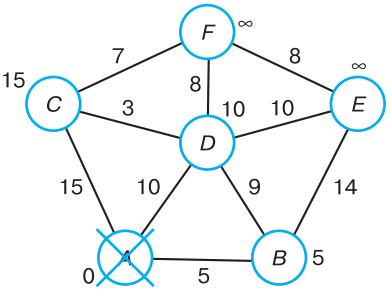


Рис. 3.13. Алгоритм Дейкстры. Шаг 1

После изменения меток всех соседей вершины A она помечается как просмотренная. Теперь минимальная метка из непросмотренных вершин у вершины B. Её соседи — вершины D и E. Так как $5 + 9 > 10$, метка вершины D не изменяется. Вершина E получает метку 19 (рис. 3.14).

Теперь минимальная метка из непросмотренных вершин у вершины D. Её соседи — вершины C, E и F. Так как $10 + 3 < 15$, метка вершины C изменяется. Вершина F получает метку 18. Метка вершины E не изменяется (рис. 3.15).

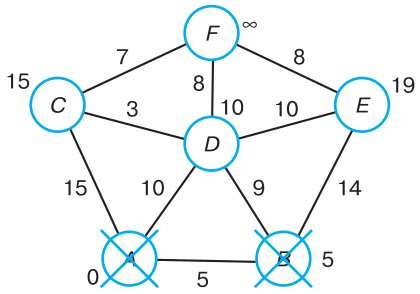


Рис. 3.14. Алгоритм Дейкстры. Шаг 2

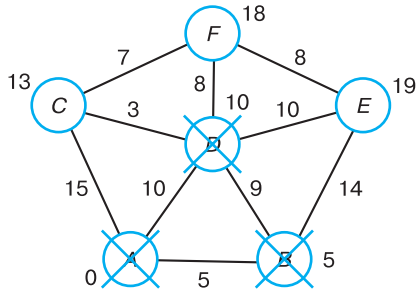


Рис. 3.15. Алгоритм Дейкстры. Шаг 3

Далее в качестве вершин с минимальными метками будут поочерёдно рассматриваться вершины C, F и E. К изменению меток соседних с ними вершин это не приведёт (рис. 3.16).

Полученные в результате работы алгоритма метки вершин графа — это и есть кратчайшие расстояния от вершины A до каждой из этих вершин.

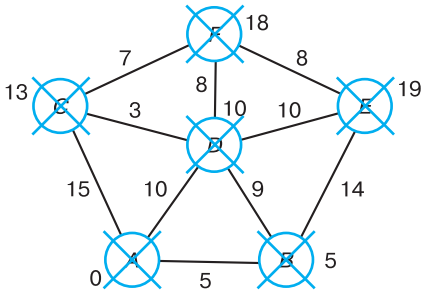


Рис. 3.16. Алгоритм Дейкстры. Результат работы

Метод динамического программирования основан на том, что процесс решения задачи разбивается на стадии (шаги), на каждой из которых принимаются решения, приводящие к достижению поставленной цели.

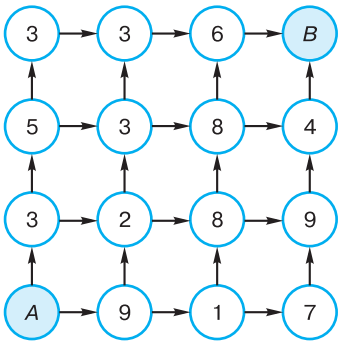


Рис. 3.17. Лабиринт

Предположим, персонажу некоторой игры необходимо пройти по лабиринту из пункта A в пункт B, набрав при этом как можно меньше штрафных баллов, количество которых указано в клетках лабиринта, причём перемещаться можно только вверх или вправо. С помощью графа начальные условия могут быть заданы так, как показано на рисунке 3.17.

Составим таблицу, в которой каждая ячейка будет соответствовать определённой клетке лабиринта. Числа в ячейках будут равны минимальному числу штрафных баллов, которое можно получить, пройдя путь от начала до соответствующей клетки.

Заполнять таблицу будем снизу вверх и слева направо. При этом для заполнения каждой новой ячейки будем рассматривать числа двух соседних с ней заполненных ячеек, находящихся слева от неё и под ней. Будем выбирать наименьшее из этих двух чисел, прибавлять к ним число текущей ячейки и результат записывать в неё.

3			
A	9		

3	5		
A	9		

8	8		
3	5	13	
A	9	10	

11			
8	8	16	
3	5	13	
A	9	10	17

11	11		
8	8	16	
3	5	13	22
A	9	10	17

11	11	17	
8	8	16	20
3	5	13	22
A	9	10	17

11	11	17	17
8	8	16	20
3	5	13	22
A	9	10	17

Ответ равен числу в правом верхнем углу таблицы.

Давайте считать, что числа, обозначающие веса вершин рассмотренного графа, — это призовые баллы, которые можно получить, пройдя по соответствующим клеткам лабиринта. Самостоятельно подсчитайте, какое максимальное число призовых баллов можно набрать, пройдя этот лабиринт.

11.2. Знакомство с теорией игр

Рассмотрим несколько примеров.

Пример 1. Это задачка из учебника информатики для 4 класса[1)].

Алёша Попович и Добрыня Никитич воюют с девятиглавым змеем. По очереди богатыри ходят к его пещере и срубают 1, 2 или 3 головы. Как начавшему бой Алёше обрести славу победителя змея (срубить последнюю голову), если и Добрыня готов приложить все усилия, чтобы стать победителем в этой битве?

Изобразим на числовой линейке текущее число голов змея:

9	8	7	6	5	4	3	2	1	0

Здесь: 9 — начальное значение; 0 — конечное значение (победа).

Алёша обретёт славу победителя, если после его последнего удара у змея останется 0 голов. Для этого нужно, чтобы после очередного удара Добрыни у змея осталось 3, 2 или 1 голова. Иначе говоря, позиции 3, 2 и 1 являются для Алёши выигрышными (как, впрочем, для любого из богатырей, кому они достаются в качестве исходных при последнем ударе). Выигрышные и проигрышные позиции на числовой линейке будем помечать буквами «В» и «П» соответственно:

						В	В	В	
9	8	7	6	5	4	3	2	1	0

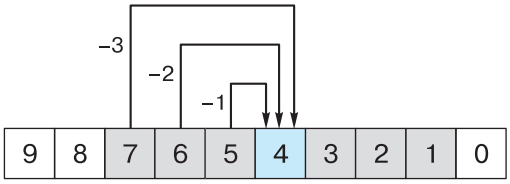
Если Добрыня выйдет на бой с черырёхголовым змеем (будет находиться в позиции 4), то любым своим ударом он создаст

1) *Семёнов А. Л.* Информатика: Учеб. пособие для 4 кл. нач. шк. В 2 ч. Ч. 2 / А. Л. Семёнов, Т. А. Рудченко. — М.: Просвещение, 2008. — 48 с. : ил.

Алёше условия для выигрыша (переведёт Алёшу в выигрышную позицию). Следовательно, задача Алёши на предыдущем шаге состоит в том, чтобы перевести Добрыню в эту заведомо проигрышную для него позицию:

					П	В	В	В	
9	8	7	6	5	4	3	2	1	0

Четырёхголового соперника Алёша сможет обеспечить Добрыне, если сам будет находиться в одной из позиций 7, 6 или 5:



Любой удар Добрыни приведёт к благоприятному для Алёши результату только в том случае, если Добрыня выйдет на бой с восьмиголовым змеем:

	П	В	В	В	П	В	В	В	
9	8	7	6	5	4	3	2	1	0

Следовательно, первым своим ударом Алёша должен срубить змею одну голову.

Выигрышная стратегия — это правило, следуя которому игрок выигрывает независимо от того, как играет противник.
Игрок имеет выигрышную стратегию, если он может выиграть при любых ходах противника. Выигрышная стратегия может быть только у одного игрока.
Описать стратегию игрока — значит описать, какой ход он должен сделать в любой ситуации при различной игре противника.

На рисунке 3.18 в форме дерева представлена выигрышная стратегия для Алёши. Поэтому для Алёши всегда указывается один ход («Ход А»), обеспечивающий требуемый результат. А вот для Добрыни, фактически выступающего в качестве соперника, рассматриваются все возможные варианты («Ход Д»).

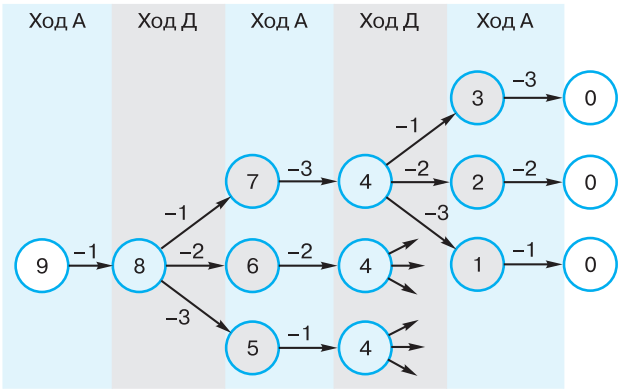


Рис. 3.18. Дерево выигрышной стратегии для Алёши

Пример 2. А эта задача из открытого банка заданий ЕГЭ по информатике **(fipi.ru)**.

Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежит куча камней. Игроки ходят по очереди, первый ход делает Петя.

За один ход игрок может выполнить одно из следующих действий:

- добавить в кучу один камень (+ 1);
- добавить в кучу два камня (+ 2);
- увеличить количество камней в куче в 3 раза (× 3).

Например, имея кучу из 5 камней, за один ход можно получить кучу из 6, 7 или 15 камней.

У каждого игрока, чтобы делать ходы, есть неограниченное количество камней. Игра завершается в тот момент, когда количество камней в куче превышает 45. Победителем считается игрок, сделавший последний ход, т. е. первым получивший кучу, в которой будет 46 или больше камней. Будем считать, что в начальный момент в куче S камней, $1 \le S \le 45$.

Выясним, при каких значениях числа S Петя может выиграть первым ходом.

Если $S = 45$, то, добавив в кучу один камень (+ 1), два камня (+ 2) или утроив количество камней в ней (× 3), Петя становится победителем.

Если $S = 44$, то стать победителем можно, если добавить в кучу два камня (+ 2) или утроить количество камней в ней (× 3).

Если $S = 43$, то Петя становится победителем, утроив количество камней в куче (× 3). Также можно действовать для любого $S \ge 16$ $(16 \times 3 = 48,\ 15 \times 3 = 45)$.

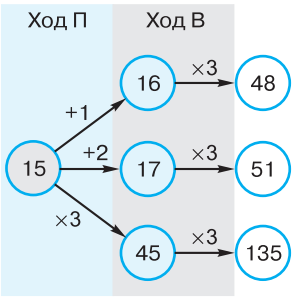


Рис. 3.19. Позиция 15 — выигрышная для Вани

Итак, Петя может выиграть, если $S = 16, ..., 45$ — это его выигрышные позиции. Для выигрыша Пете достаточно увеличить количество камней в 3 раза. При меньших значениях S за один ход нельзя получить кучу, в которой будет 46 или более камней.

Если же в куче будет 15 камней, то после любого хода Пети своим первым ходом может выиграть Ваня. Действительно, при $S = 15$ после первого хода Пети («Ход П») в куче будет 16, 17 или 45 камней. Любой из этих случаев является выигрышным для делающего ход Вани («Ход В»), которому для победы достаточно увеличить количество камней в 3 раза (рис. 3.19).

Теперь попробуем определить значения S, при которых у Пети будет выигрышная стратегия, причём Петя не сможет выиграть первым ходом, но сможет выиграть своим вторым ходом, независимо от того, как будет ходить Ваня.

Мы выяснили, что $S = 15$ — проигрышная позиция для любого игрока. Если Петя своим первым ходом сможет перевести в неё Ваню, то что бы ни делал последний, сам он выиграть не сможет, но переведёт в выигрышную позицию своего соперника. 15 камней Петя может получить при $S = 14$ (+ 1), $S = 13$ (+ 2) или $S = 5$ (× 3). Других вариантов для S нет (рис. 3.20).

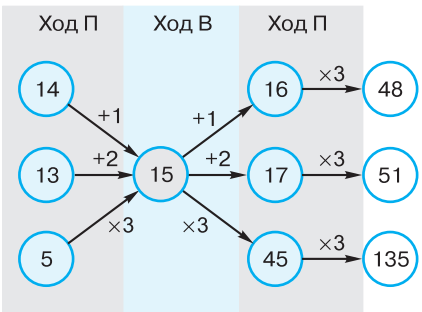


Рис. 3.20. Позиции 5, 13, 14 — выигрышные для Пети

Представим всю информацию на числовой линейке:

				В								В	В	П	В	В	В	В	
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	...	45	46

Найдём на ней такое значение S, при котором у Вани есть выигрышная стратегия, позволяющая ему выиграть первым или вторым ходом при любой игре Пети, и при этом у Вани нет стратегии, которая позволит ему гарантированно выиграть первым ходом.

Здесь речь идёт о проигрышной позиции для первого игрока. Следовательно, искать значение S надо среди позиций, не отмеченных как выигрышные.

Пусть $S = 12$. Каким бы ни был ход Пети, им он переведёт своего соперника в выигрышную позицию: 13 (12 + 1), 14 (12 + 2) или 36 (12 × 3). В последнем случае Ваня имеет возможность выиграть своим первым же ходом (36 × 3), а в первых двух случаях он должен перевести соперника в проигрышную позицию $S = 15$, что обеспечит ему выигрыш вторым ходом. Следовательно, позиция $S = 12$ — проигрышная для Пети. На дереве решений наши рассуждения могут быть представлены так, как показано на рисунке 3.21.

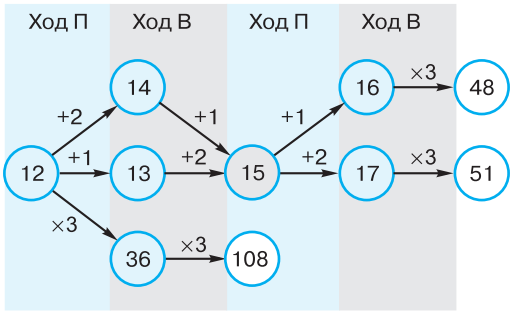


Рис. 3.21. Позиция 12 — проигрышная для Пети

Если вместо $S = 12$ взять $S = 11$, то приведёт ли любой ход Пети Ваню к выигрышу? Подойдёт ли для этой цели $S = 10$? Обоснуйте свой ответ.

Примеры, которые мы рассмотрели, имеют самое непосредственное отношение к теории игр — разделу современной математики, связанному с решением многих задач экономики, социологии, политологии, биологии, искусственного интеллекта и ряда других областей, где необходимо изучение поведения человека и животных в различных ситуациях.

Игра выступает в качестве математической модели некоторой ситуации и понимается как процесс, в котором участвуют две и

более стороны, ведущие борьбу за реализацию своих интересов. При этом игра характеризуется такими признаками, как:

1) присутствие нескольких игроков;
2) неопределённость поведения игроков, связанная с имеющимися у каждого из них несколькими вариантами действий;
3) различие (несовпадение) интересов игроков;
4) взаимосвязанность поведения игроков (результат, получаемый каждым из них, зависит от поведения всех игроков);
5) наличие правил поведения, известных всем игрокам.

Игра может быть представлена в виде дерева, каждая вершина которого соответствует ситуации выбора игроком своей стратегии.

Примеры, которые мы рассмотрели, относятся к так называемым играм с полной информацией. В играх с полной информацией участники знают все ходы, сделанные до текущего момента, равно как и возможные стратегии противников, что позволяет им в некоторой степени предсказать последующее развитие игры.

Тем, кто хочет получить более полное представление о теории игр рекомендуем познакомиться с книгой Александра Шеня «Игры и стратегии с точки зрения математики». Электронная версия книги является свободно распространяемой и доступна по адресу **www.mccme.ru/free-books/shen/shen-games.pdf**.

САМОЕ ГЛАВНОЕ

Графы как информационные модели находят широкое применение во многих сферах нашей жизни. С их помощью можно планировать оптимальные транспортные маршруты, кратчайшие объездные пути, расположение торговых точек и других объектов.

Путь между вершинами A и B графа считается кратчайшим, если эти вершины соединены минимальным числом рёбер (в случае, если граф не является взвешенным) или если сумма весов рёбер, соединяющих эти вершины, минимальна (для взвешенного графа).

Для определения кратчайшего пути между вершинами графа используются алгоритм построения дерева решений, алгоритм Дейкстры, метод динамического программирования и другие алгоритмы.

Важную роль в решении многих задач экономики, социологии, политологии, биологии, искусственного интеллекта и ряда других областей, где необходимо изучение поведения человека и животных в различных ситуациях, играет теория игр.

Игра, выступающая в качестве математической модели некоторой ситуации, характеризуется такими признаками, как:

1) присутствие нескольких игроков;
2) неопределённость поведения игроков, связанная с имеющимися у каждого из них несколькими вариантами действий;
3) различие (несовпадение) интересов игроков;
4) взаимосвязанность поведения игроков (результат, получаемый каждым из них, зависит от поведения всех игроков);
5) наличие правил поведения, известных всем игрокам.

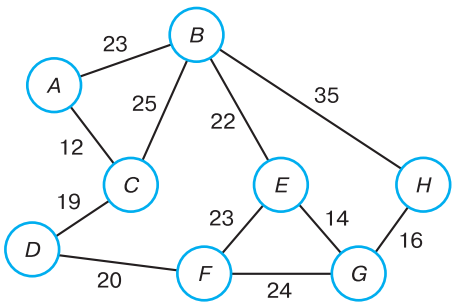
Игра может быть представлена в виде дерева, каждая вершина которого соответствует ситуации выбора игроком своей стратегии.

В играх с полной информацией участники знают все ходы, сделанные до текущего момента, равно как и возможные стратегии противников, что позволяет им в некоторой степени предсказать последующее развитие игры.

Выигрышная стратегия — это правило, следуя которому игрок выигрывает независимо от того, как играет противник.

Вопросы и задания

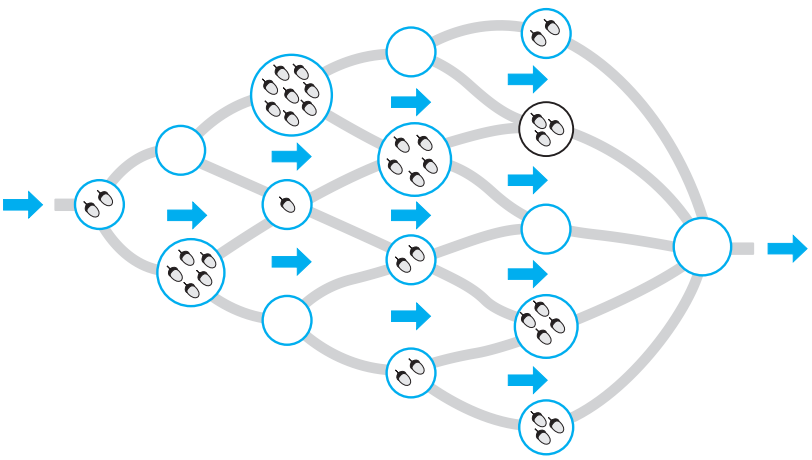
1. В решении каких прикладных задач используются алгоритмы нахождения кратчайшего пути между заданными вершинами в графе?
2. С помощью алгоритма Дейкстры найдите кратчайший путь между вершинами A и G следующего графа:



3. В материалах международного конкурса по информатике «Бобёр» есть такая задача, предложенная разработчиками из Нидерландов.

   Бобёр Билли любит жёлуди. Он хочет поплыть по течению и собрать все жёлуди на островах, мимо которых будет проплывать. Увы, течение реки настолько сильное, что он мо-

жет плыть только вниз по течению. Какое максимальное количество желудей он сможет собрать?



Решите эту задачу, воспользовавшись методом динамического программирования.

4. На столе лежит 25 спичек. Играют двое. Игроки по очереди могут взять от одной до четырёх спичек. Кто не может сделать ход (т. к. спичек не осталось), проигрывает. Другими словами, выигрывает взявший последнюю спичку. Выясните, у кого из игроков есть выигрышная стратегия.
5. Выясните, у кого из двух игроков есть выигрышная стратегия в такой игре: начальная позиция — на столе лежит 107 спичек, за один ход можно брать 1 или 2 спички. Выигрывает тот, кто взял последнюю спичку.
6. Два игрока играют в следующую игру. Перед ними лежат две кучки камней, в первой из которых 2, во второй — 3 камня. У каждого игрока неограниченное количество камней. Игроки ходят по очереди. Ход состоит в том, что игрок или увеличивает число камней в какой-то куче в 3 раза, или добавляет 3 камня в любую из куч. Выигрывает игрок, после хода которого общее число камней в двух кучах становится не менее 35. Кто выигрывает — игрок, делающий ход первым, или игрок, делающий ход вторым?
7. Два игрока, Петя и Ваня, играют в следующую игру[1]. Перед игроками лежит куча камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить

[1] Большое количество подобных заданий вы можете найти в открытом банке заданий ЕГЭ по информатике на сайте **fipi.ru**.

в кучу 1 камень или 5 камней. Например, имея кучу из 10 камней, за один ход можно получить кучу из 11 или 15 камней. У каждого игрока, чтобы делать ходы, есть неограниченное количество камней. Игра завершается в тот момент, когда количество камней в куче становится не менее 47. Победителем считается игрок, сделавший последний ход, т. е. первым получивший кучу, в которой будет 47 или больше камней.
В начальный момент в куче было S камней, $1 \le S \le 46$.
Выполните следующие задания, в каждом случае обосновывая свой ответ.

1) Укажите все такие значения числа S, при которых Петя может выиграть в один ход. Обоснуйте, что найдены все нужные значения S, и укажите выигрывающие ходы.
2) Укажите такое значение S, при котором Петя не может выиграть за один ход, но при любом ходе Пети Ваня может выиграть своим первым ходом. Опишите выигрышную стратегию Вани.
3) Укажите два значения S, при которых у Пети есть выигрышная стратегия, причём Петя не может выиграть за один ход, но может выиграть своим вторым ходом независимо от того, как будет ходить Ваня. Для указанных значений S опишите выигрышную стратегию Пети.
4) Укажите значение S, при котором у Вани есть выигрышная стратегия, позволяющая ему выиграть первым или вторым ходом при любой игре Пети, однако у Вани нет стратегии, которая позволит ему гарантированно выиграть первым ходом. Для указанного значения S опишите выигрышную стратегию Вани. Постройте дерево всех партий, возможных при этой выигрышной стратегии Вани.

§ 12 База данных как модель предметной области

12.1. Общие представления об информационных системах

Способность накапливать информацию и обеспечивать эффективный доступ к ней является сегодня определяющим фактором развития и поддержания жизнеспособности человеческого общества.

Объёмы накопленных человечеством данных неуклонно растут. В 2011 году общий мировой объём сгенерированных человечеством данных составил более 1,8 зеттабайт (1,8 трлн Гбайт). Столько же «весят» 200 млн двухчасовых фильмов в формате высокой чёткости, которые можно просматривать ежедневно без перерыва в течении 47 млн лет. Ожидается, что количество данных на планете будет как минимум удваиваться каждые два года вплоть до 2020 года. (По материалам журнала «Суперкомпьютеры», № 1(17), 2014.)

Развитие средств вычислительной техники и информационных технологий открыло новые способы хранения, представления и поиска информации, сделав возможным создание информационных систем, обеспечивающих практически мгновенное получение каждым требуемой ему информации в той или иной предметной области (части реального мира). Центральной частью любой информационной системы является база данных.

База данных (БД) — совокупность данных, организованных по определённым правилам, отражающая состояние объектов и их отношений в некоторой предметной области (транспорт, медицина, образование, право и т. д.), предназначенная для хранения во внешней памяти компьютера и постоянного применения.

Информационная система — это совокупность содержащейся в базах данных информации и обеспечивающих её обработку информационных технологий и технических средств[1)].

В наше время информационные системы находят широкое применение в производственной, маркетинговой, финансовой, кадровой и многих других сферах деятельности.

Примерами информационных систем (рис. 3.22), доступными каждому пользователю, в том числе и с помощью мобильных устройств, являются:

- информационные системы, обеспечивающие возможность получения справочной информации о расписании самолётов, поездов дальнего следования, электричек и автобусов, а также покупку железнодорожных и авиабилетов;
- информационные системы, позволяющие узнать наличие и цены на медикаменты в аптеках, места в отелях города (региона) и др.;

1) Закон Российской Федерации «Об информации, информационных технологиях и о защите информации».

- разнообразные поисково-информационные картографические службы;
- порталы, содержащие информацию нормативно-правового характера и др.

Узнайте, для чего предназначены информационные системы, представленные на рисунке 3.22.

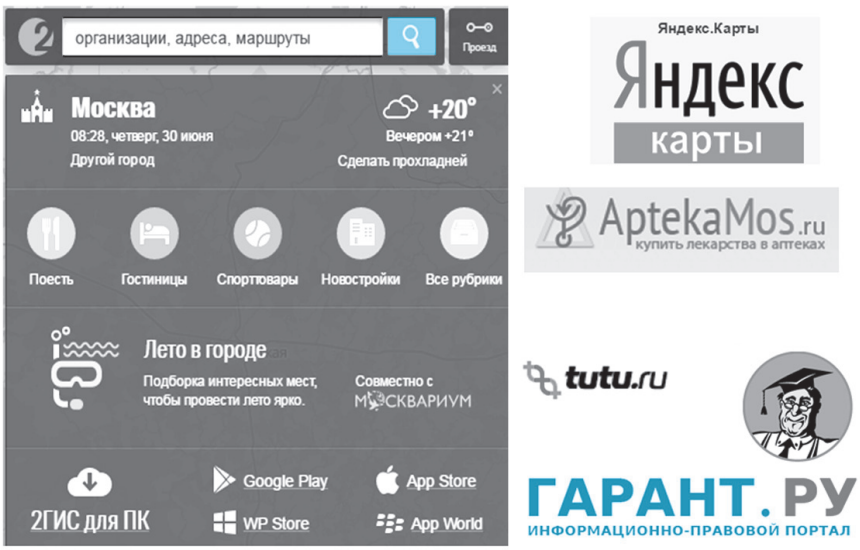


Рис. 3.22. Примеры информационных систем

12.2. Предметная область и её моделирование

Предметная область — это часть реального мира, рассматриваемая в рамках определённой деятельности. Например, можно рассматривать такие предметные области, как школа, библиотека, поликлиника, кинотеатр, склад и т. д.

В предметной области можно выделить некоторые объекты (классы объектов) и зафиксировать их свойства (атрибуты).

Объект предметной области — это факт, лицо, событие, предмет, о котором могут быть собраны данные.

Информационный объект или сущность — это описание некоторого класса реальных объектов в виде совокупности свойств.

Сущность предметной области — это класс объектов предметной области; по сути, это совокупность однотипных объектов.

Примерами объектов (с точки зрения внешнего мира) или сущностей (с точки зрения БД) являются ученик, класс, кабинет, время занятий и т. д. Сущность УЧЕНИК может быть представлена в БД с помощью следующих атрибутов: номер личного дела, фамилия, имя, отчество, год рождения. Это можно записать так:

УЧЕНИК (НОМЕР ЛИЧНОГО ДЕЛА, ФАМИЛИЯ, ИМЯ, ОТЧЕСТВО, ГОД РОЖДЕНИЯ).

Между объектами, а следовательно, и между соответствующими им сущностями могут существовать связи одного из следующих типов:

- «один к одному» (обозначается 1 : 1);
- «один ко многим» (обозначается 1 : М);
- «многие к одному» (обозначается М : 1);
- «многие ко многим» (обозначается М : М).

Связь 1 : 1 имеет место, когда одному экземпляру одной сущности соответствует один экземпляр другой сущности. Такая связь может быть установлена между сущностями ВЫПУСКНИК_ШКОЛЫ и АТТЕСТАТ: каждый выпускник школы получает аттестат о среднем образовании и каждый аттестат принадлежит одному выпускнику.

Связь 1 : М имеет место, когда одному экземпляру одной сущности может соответствовать несколько экземпляров другой сущности. Например, у матери может быть несколько детей; в одном кинотеатре может быть несколько залов; в одном классе, как правило, множество учеников.

Связь М : 1 является противоположной к связи 1 : М; она имеет место, когда нескольким экземплярам одной сущности соответствует один экземпляр другой сущности. Например, несколько учеников учатся в одном классе.

Связь М : М имеет место, когда нескольким экземплярам одной сущности соответствует несколько экземпляров другой сущности. Например, многие ученики получают много разных оценок; каждый учитель, преподающий в 11 классе, обучает многих учащихся, а каждый учащийся 11 класса обучается у нескольких учителей; один автор может написать несколько книг, и, в то же время, одна книга может быть написана несколькими авторами.

Существуют связи, которыми каждый экземпляр одной сущности обязательно связан с одним или несколькими экземплярами другой сущности. Например, связь между сущностями КЛАСС и УЧЕНИК такова, что каждый ученик принадлежит к определённому классу, и каждый класс состоит из определённой группы учеников. Возможны связи, при которых каждый экземпляр одной сущности не обязательно связан хотя бы с одним экземпляром другой сущности.

Для создания БД необходимо, прежде всего, построить модель её предметной области, определив, данные о каких объектах будут в ней храниться и какие связи между этими данными необходимо учесть.

Модель предметной области, включающую в себя сущности, их атрибуты и связи между сущностями называют моделью «сущность–связь», или ER-моделью (от англ. *Entity–Relationship* — сущность–связь).

Для большей наглядности при создании моделей «сущность–связь» пользуются условными графическими обозначениями: сущности изображаются в виде прямоугольников, атрибуты — в виде эллипсов, связи — в виде ромбов.

Построим модель «сущность–связь» для предметной области «Авиаперелёты», в которой рассмотрим две сущности: ПАССАЖИР и БИЛЕТ (рис. 3.23).

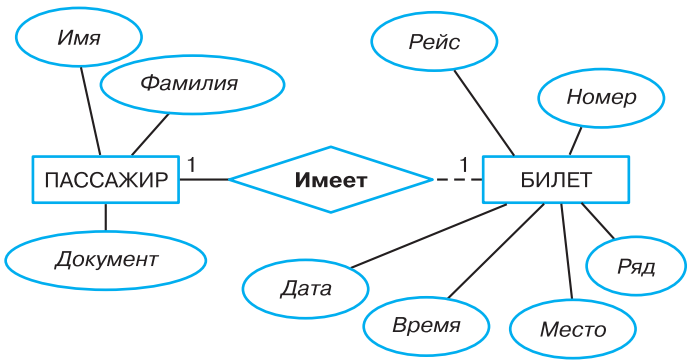


Рис. 3.23. Модель «сущность–связь» предметной области «Авиаперелёты»

Каждый пассажир, собирающийся лететь, например, в Париж, имеет билет. Двух одинаковых билетов, как и двух одинаковых пассажиров, не существует. Сущность ПАССАЖИР в данной модели характеризуется свойствами: ФАМИЛИЯ, ИМЯ и ДОКУМЕНТ, удостоверяющий личность. Атрибуты сущности БИЛЕТ — РЕЙС, ДАТА, ВРЕМЯ, РЯД, МЕСТО и НОМЕР БИЛЕТА. Между сущностями ПАССАЖИР и БИЛЕТ существует связь **Имеет**. Это связь «один к одному»; соответствующие обозначения находятся над линиями связи возле прямоугольников сущностей. Эта связь является обязательной (сплошная линия на схеме) для сущности ПАССАЖИР (для того чтобы быть пассажиром, человек должен иметь билет) и необязательной (пунктирная линия на схеме) для сущности БИЛЕТ, поскольку не все билеты на рейс могут быть проданы.

12.3. Представление о моделях данных

Модель данных — это совокупность структур данных и операций их обработки.

С помощью модели данных могут быть представлены сущности и взаимосвязи между ними. Выделяют три основных типа моделей данных: иерархическую, сетевую и реляционную.

Иерархическая модель данных определяет организацию данных об объектах в виде дерева. В иерархической модели данных у каждого объекта есть только один объект высшего уровня, которому он подчинен (родительский), и может быть несколько подчиненных объектов (потомков). Исключение составляет только наивысший по иерархии объект — у него нет родительского объекта. В иерархической модели данных каждый родительский объект в совокупности с подчиненными объектами (потомками) можно рассматривать как отдельное дерево.

Пример иерархической организации данных представлен на рисунке 3.24. Информация БД «Школа» структурирована в виде иерархических деревьев, количество которых равно количеству подразделений (НАЧАЛЬНАЯ ШКОЛА, ОСНОВНАЯ ШКОЛА, СТАРШАЯ ШКОЛА). На первом уровне находится сущность ПОДРАЗДЕЛЕНИЕ (НОМЕР, НАЗВАНИЕ, РУКОВОДИТЕЛЬ). Сущности второго уровня — КЛАССЫ (КОД КЛАССА, КЛАССНЫЙ

РУКОВОДИТЕЛЬ), сущности третьего уровня — УЧЕНИКИ (ЛИЧНОЕ ДЕЛО, ФАМИЛИЯ). Подчёркиванием выделен атрибут, который однозначно определяет каждый экземпляр сущности.

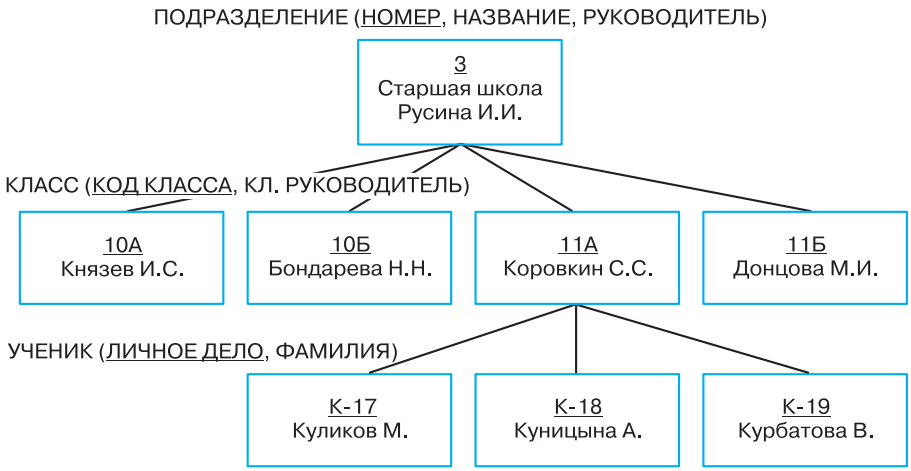


Рис. 3.24. Пример иерархической организации данных

Для обработки данных в иерархической модели данных используется следующий набор команд:

- найти указанное дерево (например, дерево 1);
- перейти от одного дерева к другому (например, от дерева 11А к дереву 10А);
- перейти от родительского объекта к объекту-потомку внутри дерева (например, от объекта 11А к объекту К-18);
- перейти от одного объекта к другому в порядке, предусмотренном иерархической структурой (например, от объекта 9А к объекту 10А);
- вставить новый объект в указанном месте;
- удалить текущий объект и др.

Модель данных должна обеспечивать целостность данных, иначе говоря, в представленных с её помощью данных не должно быть противоречий. Свойство целостности должно сохраняться при любых действиях с данными.

Основное правило обеспечения целостности в иерархической модели данных состоит в том, что ни один подчиненный объект (потомок) не может существовать без родительского объекта, за исключением одного основного родительского объекта.

При значительном количестве данных в БД, построенных по иерархической модели, поиск нужных данных может занять много времени. Например, поиск файла, содержащего определённый фрагмент текста, на всех жёстких дисках персонального компьютера может длиться несколько минут.

В Интернете подобный поиск будет длиться максимум несколько секунд, при этом будут обработаны значительно большие объёмы данных.

Постарайтесь вспомнить, за счёт чего так быстро происходит поиск в Интернете. Как это связано с индексацией данных? Используются ли аналогичные возможности в современных операционных системах?

Иерархическую модель данных удобно использовать для предметной области, объекты которой также имеют между собой иерархическую зависимость. Для предметной области, в которой объекты связаны между собой более сложной зависимостью, чем иерархия, может быть использована сетевая модель данных.

Сетевая структура данных предусматривает, что у каждого объекта может быть как несколько объектов-потомков, так и несколько родительских объектов. Пример связей между объектами при использовании сетевой модели данных изображен на рисунке 3.25.

Для обработки данных в сетевой модели данных используются команды:

- найти указанный объект среди однотипных объектов, например объект с данными об ученике Кучеренко М.;

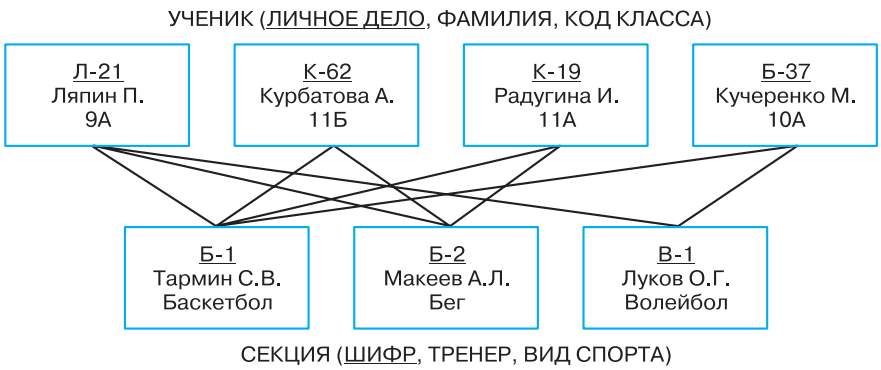
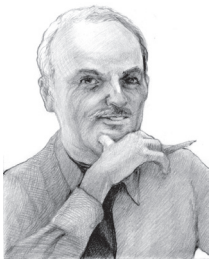


Рис. 3.25. Сетевая структура данных

- перейти от родительского объекта к первому потомку, используя определённую связь;
- перейти от объекта-потомка к родительскому объекту, используя определённую связь;
- вставить новый объект в указанном месте;
- удалить текущий объект;
- изменить объект;
- включить объект в определённую связь;
- разорвать связь и др.

Использование сетевой модели данных осложняется при значительном увеличении количества объектов предметной области и усложнении связей между ними.

Основой структуры реляционной модели данных является таблица, каждая строка которой содержит набор значений свойств одного из объектов предметной области, а каждый столбец — набор значений определённого свойства объектов предметной области. Таблица реляционной БД состоит из элементов определённых множеств, что позволяет для обработки данных этой таблицы использовать операции над множествами.



Эдгар Франк Кодд (1923–2003) — британский учёный, внёсший существенный вклад во многие области информатики. Кодд является создателем реляционной модели данных и основоположником теории реляционных БД. Математик по образованию, он ввёл в теорию БД математический подход, основывающийся на теории множеств.

Целостность[1)] в реляционной модели данных обеспечивается соблюдением двух принципов:

1) обязательная возможность идентификации объекта (экземпляра сущности) за счёт уникальности набора значений его свойств, указанных в строке реляционной таблицы;
2) обязательная корректность связей между таблицами.

В соответствии с моделью данных, лежащей в основе БД, различают иерархические, сетевые и реляционные БД. Последние мы рассмотрим более подробно.

1) Более подробно этот вопрос будет раскрыт в следующем параграфе.

12.4. Реляционные базы данных

Итак, основным объектом реляционной БД является таблица. Каждая такая таблица, называемая реляционной таблицей или отношением, обладает следующими свойствами:

- все столбцы в таблице однородные, т. е. все элементы в одном столбце имеют одинаковый тип и максимально допустимый размер;
- каждый столбец имеет уникальное имя;
- одинаковые строки в таблице отсутствуют;
- порядок следования строк и столбцов в таблице не имеет значения.

Основными структурными элементами реляционной таблицы являются поле и запись (рис. 3.26).

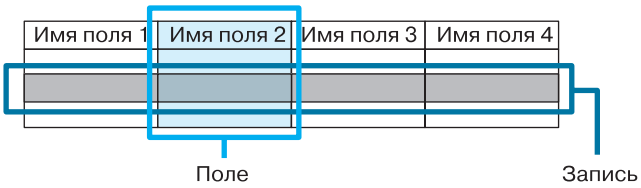


Рис. 3.26. Основные структурные элементы реляционной таблицы

Поле (столбец реляционной таблицы) — элементарная единица логической организации данных, которая соответствует конкретному атрибуту сущности.

Запись (строка реляционной таблицы) — совокупность логически связанных полей, соответствующая конкретному экземпляру сущности. Например, информация о крупнейших озёрах мира в виде реляционной таблицы представлена на рисунке 3.27.

Для наглядности представления связей между таблицами переходят к представлению структур таблиц, указывая только имена полей:

НАЗВАНИЕ ТАБЛИЦЫ

ИМЯ ПОЛЯ 1
ИМЯ ПОЛЯ 2
…

Название	Материк	Макс. глубина, м	Площадь, тыс. кв. км
Байкал	Евразия	1620	31,5
Большое Невольничье	Северная Америка	150	28,6
Гурон	Северная Америка	228	59,6
Гэрднер	Австралия	8	4,7
Женевское	Евразия	310	0,5
Маракайбо	Южная Америка	250	16,3
Мёртвое море	Евразия	356	1
Онежское	Евразия	110	9,6
Онтарио	Северная Америка	237	19,5
Рудольф	Африка	73	8,6
Танганьика	Африка	1435	32,9
Чудское	Евразия	14	3,5

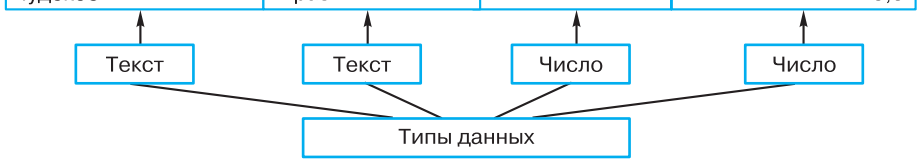


Рис. 3.27. Таблица реляционной БД

Например, структура таблицы, представленной на рисунке 3.27, будет иметь вид:

КРУПНЕЙШИЕ ОЗЕРА

НАЗВАНИЕ
МАТЕРИК
МАКСИМАЛЬНАЯ ГЛУБИНА
ПЛОЩАДЬ

Первичный ключ (идентификатор) реляционной таблицы — это поле или совокупность полей, которые однозначно определяют каждую строку (запись) в таблице.

Основные свойства первичного ключа:

1) однозначная идентификация записи (запись должна однозначно определяться значением ключа);
2) отсутствие избыточности (удаление любого поля первичного ключа приведёт к нарушению свойства однозначной идентификации записи).

Ключ, состоящий из одного поля, называется простым ключом (ключевым полем). Ключ называется составным, если он включает в себя несколько полей.

В таблице БД, представленной на рисунке 3.27, в качестве ключевого можно использовать поле НАЗВАНИЕ: значения в этом поле являются уникальными для каждой записи, потому что крупных озёр с одинаковыми названиями не существует.

Для хранения данных о сущностях некоторой предметной области может использоваться несколько связанных между собой таблиц. Связь между таблицами устанавливается с помощью ключевых полей.

Можно связать две реляционные таблицы, если ключ одной связываемой таблицы ввести в состав ключа другой таблицы (возможно совпадение ключей). Ключевое поле одной связываемой таблицы можно ввести в структуру другой таблицы, при этом оно уже не будет ключевым; такое поле называется внешним ключом.

Между таблицами *A* и *B* установлена связь «один к одному», если каждая запись в таблице *A* может иметь не более одной связанной с ней записи в таблице *B*, и наоборот — каждая запись в таблице *B* может иметь не более одной связанной с ней записи в таблице *A*.

Связь между таблицами имеет тип «один к одному», если она установлена по совпадающим первичным ключам.

Таблицы УЧЕНИК и УЧЕБНЫЙ ГОД связаны по типу «один к одному» (рис. 3.28). В этом случае имеет место совпадение ключей.

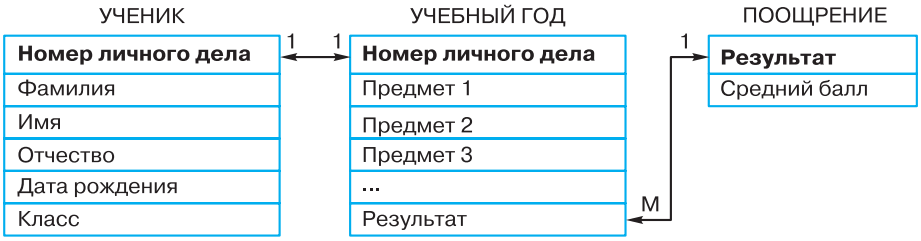


Рис. 3.28. Пример связей «один к одному» и «один ко многим» между таблицами

Между таблицами *A* и *B* установлена связь «один ко многим», если каждая запись в таблице *A* может быть связана с несколькими записями таблицы *B*, но каждая запись в таблице *B* не может быть связана более чем с одной записью таблицы *A*.

Между таблицами ПООЩРЕНИЕ и УЧЕБНЫЙ ГОД связь «один ко многим». В этом случае ключевое поле одной связываемой таблицы (ПООЩРЕНИЕ) введено в структуру другой таблицы (УЧЕБНЫЙ ГОД) так, что там оно уже не является ключевым и рассматривается как внешний ключ. В рамках этой связи таблица, содержащая первичный ключ, считается главной, а таблица, содержащая внешний ключ, считается подчинённой.

Между таблицами *A* и *B* установлена связь «многие ко многим», если каждой записи таблицы *A* может соответствовать несколько записей в таблице *B*, и наоборот — каждой записи таблицы *B* может соответствовать несколько записей в таблице *A*.

Такая связь всегда реализуется с помощью третьей связующей таблицы *C*. Связь «многие ко многим» представляет собой комбинацию двух связей типа «один ко многим»: между таблицами *A* и *C* и между таблицами *B* и *C*.

Например, связь между таблицами ЧИТАТЕЛЬ и КНИГА базы данных БИБЛИОТЕКА может быть реализована с помощью таблицы АБОНЕМЕНТ[1] (рис. 3.29).

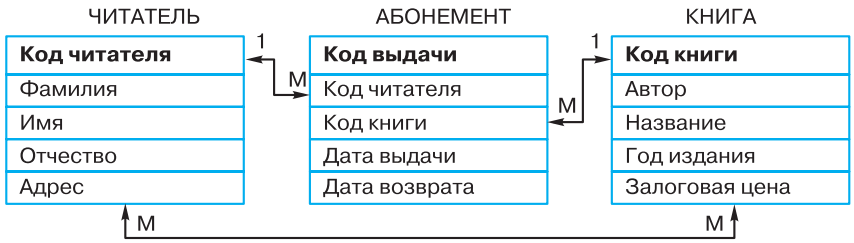


Рис. 3.29. Пример связей «один ко многим» и «многие ко многим» между таблицами

1) Самостоятельно уточните значение слова «абонемент» в дополнительных источниках информации.

САМОЕ ГЛАВНОЕ

База данных (БД) — совокупность данных, организованных по определённым правилам, отражающая состояние объектов и их отношений в некоторой предметной области, предназначенная для хранения во внешней памяти компьютера и постоянного применения.

Информационная система — это совокупность содержащейся в базах данных информации и обеспечивающих её обработку информационных технологий и технических средств.

Модель данных — это совокупность структур данных и операций их обработки. Выделяют три основных типа моделей данных: иерархическую, сетевую и реляционную.

Основным объектом реляционной БД является таблица, обладающая следующими свойствами: все столбцы в таблице однородные, т. е. все элементы в одном столбце имеют одинаковый тип и максимально допустимый размер; каждый столбец имеет уникальное имя; одинаковые строки в таблице отсутствуют; порядок следования строк и столбцов в таблице не имеет значения.

Основными структурными элементами реляционной таблицы являются поле и запись.

Первичный ключ (ключевое поле, идентификатор) реляционной таблицы — это поле или совокупность полей, которые однозначно определяют каждую строку (запись) в таблице. Ключ, состоящий из одного поля, называется простым ключом. Ключ называется составным, если он включает в себя несколько полей.

Для хранения данных о сущностях некоторой предметной области может использоваться несколько связанных между собой таблиц. Связь между таблицами устанавливается с помощью ключевых полей.

Между таблицами A и B установлена связь «один к одному», если каждая запись в таблице A может иметь не более одной связанной с ней записи в таблице B, и наоборот — каждая запись в таблице B может иметь не более одной связанной с ней записи в таблице A.

Между таблицами A и B установлена связь «один ко многим», если каждая запись в таблице A может быть связана с несколькими записями таблицы B, но каждая запись в таблице B не может быть связана более чем с одной записью таблицы A.

Между таблицами A и B установлена связь «многие ко многим», если каждой записи таблицы A может соответствовать несколько записей в таблице B, и наоборот — каждой записи таб-

лицы *B* может соответствовать несколько записей в таблице *A*. Такая связь всегда реализуется с помощью третьей связующей таблицы *C*. Связь «многие ко многим» представляет собой комбинацию двух связей типа «один ко многим»: между таблицами *A* и *C* и между таблицами *B* и *C*.

Можно связать две реляционные таблицы, если ключ одной связываемой таблицы ввести в состав ключа другой таблицы (возможно совпадение ключей). Ключевое поле одной связываемой таблицы можно ввести в структуру другой таблицы, при этом оно уже не будет ключевым; такое поле называется внешним ключом.

Вопросы и задания

1. Для чего нужно упорядоченное хранение данных?
2. Что такое информационная система? Каково основное назначение информационных систем?
3. Имеете ли вы опыт использования каких-либо информационных систем?
4. Что такое база данных? Как связаны информационная система и база данных?
5. Что такое предметная область? Как представляются объекты предметной области и их свойства в информационной модели предметной области?
6. Что такое сущность? Что такое экземпляр сущности? Приведите примеры.
7. Что называют моделью «сущность–связь»?
8. Постройте модель «сущность–связь» для предметной области «Концертный зал».
9. Назовите типы связей между сущностями предметной области.
10. Определите тип связей между сущностями:
    1) КЛИЕНТ и ЗАКАЗ в интернет-магазине;
    2) МАШИНА и ЧАСТИ МАШИНЫ;
    3) УЧИТЕЛЬ и УЧЕНИК в школе;
    4) КОМНАТА и ГОСТЬ в отеле;
    5) ГРАЖДАНИН и ПАСПОРТ.
11. Что такое модель данных? Для чего она создаётся?
12. Опишите иерархическую модель данных.
13. Опишите сетевую модель данных.
14. Опишите реляционную модель данных.

15. Опишите таблицу реляционной БД.
16. Что такое ключевое поле? Каковы требования к ключевому полю?
17. Какого типа связи могут быть установлены между таблицами реляционной БД? Охарактеризуйте каждый тип связи.
18. Во фрагменте БД представлены сведения об участниках выставки:

Таблица 1

Страна	Участник
Великобритания	Стив
Германия	Мейер
США	Кинкейд
Россия	Сафронов
Канада	Селби
Германия	Рихард
Великобритания	Дейв
Германия	Гюнтер
Россия	Глазунов
Германия	Зив

Таблица 2

Участник	Жанр
Глазунов	Натюрморт
Селби	Пейзаж
Сафронов	Портрет
Мейер	Пейзаж
Кинкейд	Пейзаж
Дейв	Портрет
Рихард	Натюрморт
Гюнтер	Пейзаж
Зив	Натюрморт
Стив	Портрет

1) Охарактеризуйте связь между представленными таблицами БД.
2) Художники из скольких стран представили на выставке пейзажи?
3) Представьте всю имеющуюся информацию о выставке в одной таблице.
4) Представьте всю имеющуюся информацию о выставке в форме графа.

19. Во фрагменте БД представлены сведения о родственных отношениях:

Таблица 1

ID	Фамилия И. О.	Пол
2272	Диковец А. Б.	Ж
2228	Диковец Б. Ф.	М
2299	Диковец И. Б.	М
2378	Диковец П. И.	М
2356	Диковец Т. И.	Ж
2331	Тесла А. П.	М
1217	Тесла П. А.	М
1202	Ландау М. А.	Ж
2227	Решко Д. А.	Ж
2240	Решко В. А.	Ж
2322	Друк Г. Р.	Ж

Таблица 2

ID Родителя	ID Ребёнка
2227	2272
2227	2299
2228	2272
2228	2299
2272	2240
2272	1202
2272	1217
2299	2356
2299	2378
2322	2356
2322	2378

Представьте имеющуюся информацию в форме графа и ответьте на следующие вопросы.

1) Сколько внуков у Решко Д. А.?
2) Информация о скольких супружеских парах представлена в таблицах?
3) Какой идентификационный номер (ID) у дяди Решко В. А.?

§ 13
Системы управления базами данных

13.1. Этапы разработки базы данных

Процесс разработки БД состоит из нескольких этапов.

1. **Постановка задачи.** На этом этапе определяется цель создания БД, уточняется предметная область, перечисляются виды работ, которые предполагается осуществлять в этой БД (отбор, изменение данных, печать отчёта и т. д.), определяются потенциальные пользователи. К постановке задачи привлекаются не только специалисты по БД, но и специалисты из той предметной области, для которой она создаётся. Чем полнее будут представления специалиста-предметника о принципах создания БД, тем конструктивнее будет его взаимодействие со специалистом в области информационных технологий и тем качественнее будет конечный результат.
2. **Проектирование БД.** На этом этапе определяется, из каких сущностей (информационных объектов) должна состоять БД, какими атрибутами будет описываться каждая сущность. Затем определяется структура реляционных таблиц с указанием свойств полей и связей между таблицами, а именно:
   1) составляется общий список полей, отражающий атрибуты таблиц БД;
   2) поля общего списка распределяются по базовым таблицам;
   3) в соответствии со свойствами данных определяются свойства каждого поля;
   4) в каждой таблице выделяется ключевое поле;
   5) определяются связи между таблицами.
3. **Создание БД** с использованием одного из языков программирования или специального программного обеспечения — систем управления базами данных (СУБД). Первый способ применяется для создания уникальных БД и требует высокой квалификации от программиста. Для работы с СУБД достаточно базовых пользовательских навыков и понимания основ разработки БД, которые мы рассмотрели в предыдущем параграфе. Далее мы будем говорить только об этом способе.
   Создание БД в СУБД предполагает:
   1) запуск СУБД и создание нового файла БД;
   2) создание таблиц и установление связей между ними;

3) тестирование БД и её коррекцию;
4) разработку различных элементов управления данными (экранных форм для ввода, редактирования и просмотра данных в таблицах; запросов для сортировки, поиска и отбора данных; отчётов для вывода данных на печать), а также установку средств защиты БД, например разграничение прав доступа для различных пользователей с помощью паролей;
5) заполнение таблиц данными (как правило, непосредственно разработчик БД вводит в неё только тестовые данные, необходимые для проверки правильности структур таблиц, связей между таблицами и т. д.; в готовую БД информацию может вводить кто-то из пользователей БД).

4. **Эксплуатация созданной БД**, в том числе:
- сортировка, фильтрация и поиск записей в таблицах;
- отбор данных из таблиц в соответствии с заданными критериями отбора;
- выполнение обработки данных (удаление, добавление, изменение данных, выполнение вычислений);
- подготовка отчётов.

В ходе эксплуатации БД, как правило, данные регулярно обновляются, могут изменяться связи между сущностями и т. п.

При проектировании БД, больших по объёму и ориентированных на разные группы пользователей, выделяют концептуальный, внешний и внутренний уровни представления данных.

Модель «сущность–связь» следует рассматривать как концептуальный уровень представления данных.

Внешний (пользовательский) уровень предусматривает представление данных в виде, требуемом конкретному пользователю БД.

Внутренний (физический) уровень представления данных определяет особенности хранения данных, методов доступа к ним и т. д.

В каждой школе нашей страны используется БД «Электронный журнал». Схему уровней представления данных для этой БД вы видите на рисунке 3.30.

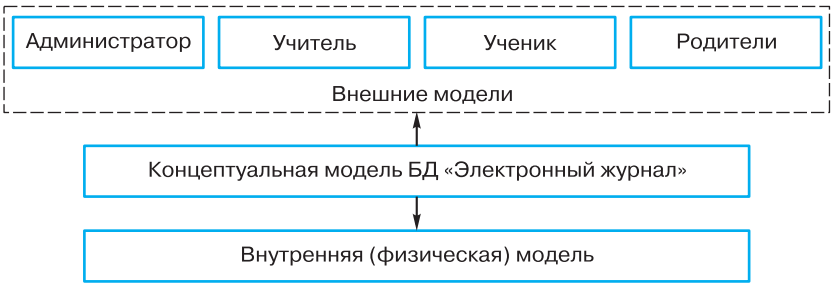


Рис. 3.30. Уровни представления данных БД «Электронный журнал»

13.2. СУБД и их классификация

Программное обеспечение для создания БД, хранения и поиска в них необходимой информации называется СУБД (системой управления базами данных).

Именно наличие СУБД превращает огромный объём хранимых в компьютерной памяти сведений в мощную информационную систему, способную быстро производить поиск и отбор необходимой нам информации.

В настоящее время существует множество СУБД. Они могут быть классифицированы по моделям данных, по размещению или по способу доступа к БД (рис. 3.31).

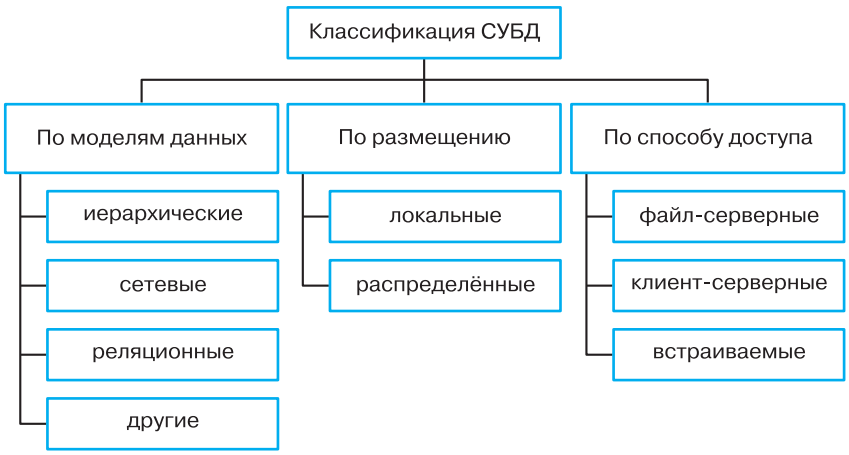


Рис. 3.31. Классификация СУБД

В зависимости от модели данных, которая используется в СУБД, их разделяют на иерархические, сетевые, реляционные и другие. С помощью дополнительных источников информации выясните, какие модели данных скрываются под словом «другие».

СУБД считается локальной, если все её части, обеспечивающие хранение и обработку данных, размещаются на одном компьютере. В распределённых СУБД данные могут храниться и обрабатываться на разных компьютерах в локальной или глобальной сети.

В файл-серверных СУБД файлы с данными размещаются централизованно на сервере. На каждом клиентском компьютере устанавливается полная версия СУБД, а доступ СУБД к данным осуществляется через локальную сеть. Одним из вариантов использования файл-серверных СУБД является размещение и СУБД, и данных на одном клиентском компьютере. Примерами файл-серверных СУБД являются Microsoft Access и dBase. В настоящее время файл-серверная технология в крупных информационных системах не используется.

В клиент-серверных СУБД на сервере, обладающем высокими техническими характеристиками, устанавливается серверная версия СУБД и БД. На клиентских компьютерах устанавливаются небольшие по объёму клиентские версии СУБД, позволяющие формировать запросы на обработку данных и выводить результаты обработки, полученные с сервера, а все операции с данными осуществляются на сервере. Примерами клиент-серверных СУБД являются Oracle, MySQL.

Встраиваемые СУБД используются в качестве составляющих других программных продуктов, например электронных энциклопедий, словарей, поисковых систем и т. п.

В школе зачастую ученики работают с реляционной СУБД Microsoft Access. Эта файл-серверная СУБД предусматривает работу с небольшими по объёму БД для личных нужд и для небольших организаций. В качестве альтернативы СУБД Microsoft Access может рассматриваться СУБД Base, входящая в состав свободного пакета офисных приложений Apache OpenOffice.

13.3. Работа в программной среде СУБД

Работа в программной среде СУБД начинается с того, что создаётся файл новой БД. Далее, на основе ранее разработанной модели «сущность–связь», необходимо создать структуру БД, определяемую:

1) количеством таблиц БД и их структурой;
2) типами связей между таблицами, если используется несколько таблиц;
3) видами и количеством других объектов БД: форм, запросов, отчётов.

В свою очередь, структура таблицы определяется набором и свойствами полей (столбцов таблицы), перечнем её ключевых полей.

Основными свойствами полей являются:

- **Имя поля** — определяет, как следует обращаться к данным этого поля при операциях с БД (должно быть уникальным в рамках таблицы);
- **Тип поля** — определяет тип данных, которые могут содержаться в данном поле;
- **Размер поля** — определяет предельную длину данных, которые могут размещаться в данном поле;
- **Формат поля** — определяет способ форматирования данных в ячейках таблицы, принадлежащих полю;
- **Подпись** — определяет заголовок столбца таблицы для данного поля (если подпись не указана, то в качестве заголовка столбца используется **Имя поля**);
- **Значение по умолчанию** — значение, которое вводится в ячейки поля автоматически при формировании очередной записи таблицы;
- **Условие на значение** — ограничение, используемое для проверки правильности ввода данных.

При создании структуры таблицы указываются типы данных, которые будут храниться в таблице. Для хранения данных определённого типа в памяти компьютера используется разная длина двоичного кода. Чем меньше объём данных, тем быстрее происходит их обработка. Основные типы данных для СУБД Microsoft Access представлены в таблице 3.1.

После того как таблицы созданы, между ними устанавливаются связи. Важно убедиться, что поля таблиц и связи описаны правильно; для этого в БД необходимо ввести несколько тестовых записей.

Таблица 3.1

Основные типы данных СУБД Microsoft Access

Тип данных	Примечание
Текстовый	Используется для хранения текста, в том числе комбинаций алфавитно-цифровых знаков (например, кода товара); максимальная длина поля 255 знаков
Поле MEMO	Используется для хранения обычного текста или комбинаций алфавитно-цифровых знаков длиной более 255 знаков
Числовой	Служит для хранения числовых значений (целых или дробных), предназначенных для вычислений (за исключением денежных значений)
Дата/время	Используется для хранения значений даты между годами 100 и 9999 и времени между 00:00:00 и 23:59:59
Денежный	Используется для хранения значений в денежных единицах (рубли, доллары и т. д.), предназначенных для вычислений
Счётчик	Используется для хранения уникальных данных (целых чисел), генерируемых программой автоматически при добавлении записи
Логический	Применяется для хранения логических значений, которые могут содержать одно из двух значений: Да/Нет, Истина/Ложь или Вкл/Выкл
Поле объекта OLE	Используется для хранения изображений, документов, диаграмм и других объектов из приложений Microsoft Office и других программ Windows в виде растровых изображений
Гиперссылка	Применяется для хранения ссылок на файлы, находящиеся на данном компьютере, или на веб-страницы в Интернете

СУБД обеспечивает автоматический контроль ввода данных. Система не допустит, чтобы поля, по которым таблицы связанны между собой, имели разные значения. Режим каскадной замены, установленный для связанных таблиц, при изменении значения поля в главной таблице обеспечивает автоматическое изменение соответствующих значений в подчинённой таблице. Аналогично действует режим каскадного удаления: достаточно удалить запись из главной таблицы, чтобы связанные записи исчезли из всех подчинённых таблиц. Всё это обеспечивает целостность данных — одно из важнейших свойств БД.

На всех этапах работы таблицы можно редактировать:

- изменять их структуру, добавляя и удаляя поля;
- изменять типы и свойства полей;
- исправлять неточные данные;
- добавлять записи.

На этом процесс создания структуры БД завершается. Теперь в созданные таблицы можно заносить данные.

Данные можно заносить непосредственно в таблицы, хотя это и не очень удобно, например из-за того, что текст в ячейке таблицы располагается в одну, иногда достаточно длинную, строку. Более удобным средством ввода и просмотра данных являются формы.

Формы — это вспомогательные объекты БД, обеспечивающие удобный для пользователя интерфейс при вводе, просмотре или редактировании данных в БД.

Формы создаются на основе одной или нескольких таблиц и содержат поля, выбранные пользователем из этих таблиц. Простая форма включает в себя поля одной таблицы. Составная форма включает в себя поля из нескольких таблиц, связанных отношением «один ко многим». Кроме полей в форму могут быть включены рисунки, текстовые надписи, диаграммы, различные элементы управления (кнопки, флажки, переключатели и т. п.). Данные, введённые пользователем в поля формы, сохраняются в тех таблицах, на основе которых эта форма была создана. Кроме того, в формах можно создавать поля для вычислений и вывода на экран новых значений на основе имеющихся значений других полей.

Дизайн (структура и оформление) формы выбирается в зависимости от того, с какой целью она создаётся.

В СУБД имеются специальные инструменты для создания форм. В Microsoft Access для этого предназначены такие инструменты, как:

- **Автоформа** — автоматизированное средство для создания форм трёх стандартных типов (в столбец, ленточная, табличная); при этом в форму вставляются все поля источника данных;
- **Мастер форм** — инструмент, позволяющий создавать структуру одного из трёх стандартных типов формы в режиме диалога с разработчиком формы; при этом в форму вставляются только поля, выбранные пользователем из источника данных;
- **Конструктор форм** — инструмент, с помощью которого пользователь конструирует форму в окне конструктора форм.

13.4. Манипулирование данными в базе данных

Действия, выполняемые над данными, хранящимися в БД, называются манипулированием данных.

Над данными, хранящимися в БД, могут выполняться следующие действия:

- сортировка данных;
- обновление, удаление и добавление данных;
- выборка данных по некоторым условиям.

Выполнение этих действий производится с помощью инструментов сортировки, фильтров и запросов.

Сортировка данных. Как и в электронных таблицах, в таблицах БД данные можно сортировать.

По умолчанию в Microsoft Access 2010 при открытии таблицы данные сортируются по возрастанию значений ключевого поля. Для изменения порядка сортировки следует:

1) открыть таблицу БД, данные в которой нужно отсортировать;
2) установить курсор в пределах поля, по данным которого будет выполнена сортировка записей;
3) выполнить команду **Главная → Сортировка и фильтр → По возрастанию (По убыванию).**

Для сортировки по данным нескольких полей с одинаковыми значениями параметров сортировки следует выделить эти поля (выделить можно лишь соседние поля) и выполнить команду

Главная → Сортировка и фильтр → По возрастанию (По убыванию). При этом в первую очередь сортировка происходит по данным полей, размещённых слева.

Таблица «Крупнейшие озёра» (см. рис. 3.27), отсортированная по возрастанию значений полей «Материк» и «Макс. глубина», приведена на рисунке 3.32.

Название	Материк	Макс. глубина, м	Площадь, тыс. кв. км
Гэрднер	Австралия	8	4,7
Рудольф	Африка	73	8,6
Танганьика	Африка	1435	32,9
Чудское	Евразия	14	3,5
Онежское	Евразия	110	9,6
Женевское	Евразия	310	0,5
Мёртвое море	Евразия	356	1
Байкал	Евразия	1620	31,5
Большое Невольничье	Северная Америка	150	28,6
Гурон	Северная Америка	228	59,6
Онтарио	Северная Америка	237	19,5
Маракайбо	Южная Америка	250	16,3

Рис. 3.32. Результат сортировки по данным двух полей

Для того чтобы выполнить сортировку по данным нескольких полей, не являющихся соседними, достаточно последовательно выполнить сортировку для каждого из них. При этом можно использовать разные значения параметров сортировки. Например, сначала отсортировать по убыванию поле «Площадь», а затем — по возрастанию поле «Материк».

Поиск и замена данных в БД осуществляются так же, как в электронных таблицах.

Фильтрация данных. Для отбора записей, данные в которых соответствуют определённым условиям, используют фильтры.

Фильтр — это условие, по которому производится поиск и отбор записей.

Фильтр «пропускает» записи, соответствующие заданным условиям, и «задерживает» (скрывает) записи, не соответствующие им.

Запросы являются одним из основных инструментов обработки данных в БД. Они могут обеспечивать не только поиск данных, соответствующих определённым критериям, подобно тому как это осуществляется во время фильтрации, но и одновременное выполнение операций над данными, и сохранение результатов.

К числу основных операций, которые могут быть осуществлены с использованием запросов, относятся:

- создание новых таблиц на основе анализа данных уже существующих таблиц БД;
- проведение вычислений — вычисление обобщённых данных (суммы, максимального или минимального значения и т. п.) для заданных полей и нахождение значений новых свойств, используя данные из разных таблиц или запросов;
- внесение изменений в уже существующие таблицы (обновление данных, вставка и удаление записей и т. п.).

В структуре запроса можно выделить:

1) поля, выводимые по запросу; они могут совпадать с полями имеющихся таблиц или вычисляться по формулам;
2) условие выбора записей, представляющее собой логическое выражение, которому удовлетворяют отбираемые записи;
3) подлежащее сортировке поле (последовательность полей) и порядок сортировки.

Так, можно создать запрос к БД «Крупнейшие озёра» на вывод названий озёр и соответствующих им материков, для которых выполняется условие:

((Площадь > 30) ИЛИ (Глубина > 300)) И (Материк <> "Африка")

Если отсортировать выводимые поля по возрастанию значений поля «Название», то получим:

Название	Материк
Байкал	Евразия
Гурон	Северная Америка
Женевское	Евразия
Мёртвое море	Евразия

Отчёт. Одним из достоинств БД является возможность создания различных форм представления выходной информации — так называемых отчётов.

Отчёт — это готовый к печати электронный документ. Отчёты можно использовать для заполнения бланков документов, например счетов на покупку товаров, сертификатов об участии в конкурсе или олимпиаде, графика посещения фитнес-центра и т. д.

Отчёт может содержать данные из разных таблиц и запросов. При создании отчёта имеется возможность применить средства обобщения, сортировки и группировки данных, выполнить расчёты с использованием встроенных функций и данных из разных полей. Также в отчёт допускается включать надписи, поясняющие данные, диаграммы и графики, рисунки и т. п.

САМОЕ ГЛАВНОЕ

Процесс разработки БД является примером решения задачи с использованием компьютера и предусматривает такие этапы, как: 1) постановка задачи; 2) проектирование БД; 3) создание БД; 4) эксплуатация созданной БД. Последние два этапа реализуются с помощью специального программного обеспечения — систем управления базами данных (СУБД).

В настоящее время существует множество СУБД. Они могут быть классифицированы по моделям данных (иерархические, сетевые, реляционные и др.), по размещению (локальные, распределённые) или по способу доступа к БД (файл-серверные, клиент-серверные, встраиваемые).

Работа в программной среде СУБД начинается с того, что создаётся файл новой БД. Далее, на основе модели «сущность–связь», создаётся структура БД, определяемая: 1) количеством

таблиц БД и их структурой; 2) типами связей между объектами таблиц, если используется несколько таблиц; 3) видами и количеством других объектов БД (форм, запросов, отчётов).

Структура таблицы определяется набором и свойствами полей (столбцов таблицы), перечнем её ключевых полей.

Связь между таблицами осуществляется через общие поля. Связь «один к одному» — через общий первичный ключ; связь «один ко многим» — через первичный ключ в одной таблице и соответствующее поле, называемое внешним ключом, в другой таблице.

СУБД обеспечивает автоматический контроль согласованности взаимосвязанных данных в разных таблицах, что гарантирует целостность данных — одно из важнейших свойств БД.

Формы — это вспомогательные объекты БД, обеспечивающие удобный для пользователя интерфейс при вводе, просмотре или редактировании данных в БД.

Действия, выполняемые над данными, хранящимися в БД, называются манипулированием данных. К ним относятся такие действия, как: сортировка данных; обновление, удаление и добавление данных; выборка данных по некоторым условиям. Выполнение этих действий производится с помощью инструментов сортировки, фильтров и запросов.

Фильтр — это условие, по которому производится поиск и отбор записей. Фильтр «пропускает» записи, соответствующие заданным условиям, и «задерживает» (скрывает) записи, не соответствующие им.

Запросы являются одним из основных инструментов обработки данных в БД. Они могут обеспечивать не только поиск данных, соответствующих определённым критериям подобно тому, как это осуществляется во время фильтрации, но и одновременное выполнение операций над данными, и сохранение результатов.

Отчёты предназначены для вывода данных на экран или на принтер. В них предусмотрены специальные элементы оформления, характерные для печатных документов, а также средства обобщения, сортировки и группировки данных, выполнения расчётов.

Вопросы и задания

1. Вспомните основные этапы решения задачи на компьютере и этапы компьютерного моделирования. Сопоставьте их с этапами разработки БД. Какие выводы вы можете сделать?
2. Охарактеризуйте суть каждого из этапов разработки БД.

3. Как взаимодействуют специалисты в области разработки БД и специалисты из предметной области, для которой разрабатывается БД, а также предполагаемые пользователи этой БД? Попытайтесь представить схему этого взаимодействия графически.
4. Какие данные могут получить ученики и родители в БД «Электронный журнал»? Одинаковые ли права доступа к данным имеют учителя, ученики и родители?
5. Недостатками каких СУБД являются необходимость устанавливать иногда достаточно дорогие полные версии программ на каждый компьютер, высокая загруженность сети во время передачи данных, необходимость в достаточно мощных компьютерах на рабочих местах клиентов?
6. СУБД какого типа (файл-серверные или клиент-серверные) обеспечивают более высокую надёжность, доступность и безопасность при работе с данными?
7. Чем отличаются локальные СУБД от распределённых СУБД?
8. Подготовьте небольшое сообщение об использовании в мобильных устройствах встраиваемых СУБД. Используйте ресурсы сети Интернет.
9. На протяжении многих лет одной из самых популярных в мире является СУБД Microsoft Access. Найдите информацию о том, когда была выпущена первая версия этой программы.
10. Что понимается под структурой БД?
11. Что понимается под структурой таблицы БД?
12. Перечислите основные типы данных СУБД Microsoft Access.
13. Определите тип данных для следующих полей некоторых БД: номер дома, возраст человека, номер телефона, количество учеников в классе, наличие у ученика персонального компьютера, наименование товара, дата изготовления товара.
14. Что вы понимаете под целостностью данных? Почему целостность данных является одним из важнейших свойств БД?
15. Для чего в БД используются формы?
16. Как вы можете объяснить многообразие типов форм и инструментов их создания?
17. С помощью имеющейся в вашем распоряжении СУБД создайте БД «Мои учебники», содержащую две таблицы:
    1) таблицу «Форма», состоящую из одного поля и содержащую список форм (печатная, электронная);

2) таблицу «Фонд», имеющую поля: «Код» (П-<порядковый номер> для учебников в печатной форме и Э-<порядковый номер> для учебников в электронной форме), «Наименование учебника», «Автор», «Форма», «Год издания», «Титульная страница».

Определите и установите типы полей для обеих таблиц; установите связь между таблицами. Введите в БД данные обо всех учебниках, которыми вы пользуетесь в 11 классе.

18. Что такое манипулирование данными? Какие инструменты манипулирования данными имеются в СУБД?
19. Что такое фильтр?
20. Что такое запрос?
21. БД «Страны» содержит сведения по различным странам мира: название; численность населения; дата переписи; процент населения страны от всего населения Земли; площадь в км^2; название материка, на котором расположена.

№	Страна	Население	Дата	Процент	Площадь	Материк
1	Бангладеш	142 319 000	15.03.2011	2,04	144 000	Евразия
2	Бразилия	196 763 000	13.07.2012	2,82	8 514 877	Ю. Америка
3	Вьетнам	87 840 000	01.06.2011	1,26	331 210	Евразия
4	Германия	81 751 602	01.01.2011	1,17	357 021	Евразия
5	Египет	81 623 000	13.07.2012	1,17	1 001 450	Африка
6	Индия	1 229 055 000	13.07.2012	17,41	3 287 590	Евразия
7	Индонезия	237 641 326	01.05.2010	3,4	1 919 440	Евразия
8	КНР	1 352 250 000	13.07.2012	19,37	9 596 960	Евразия
9	Мексика	112 336 538	12.06.2010	1,61	1 972 550	С. Америка
10	Нигерия	166 629 383	01.07.2012	2,39	923 768	Африка
11	Пакистан	176 210 000	13.07.2012	2,52	803 940	Евразия
12	Россия	143 098 100	01.05.2012	2,05	17 098 246	Евразия
13	США	313 329 000	13.07.2012	4,5	9 518 900	С. Америка
14	Филиппины	92 337 852	01.05.2010	1,32	299 764	Евразия
15	Эфиопия	84 320 987	01.01.2012	1,21	1 104 300	Африка
16	Япония	127 960 000	01.10.2011	1,83	377 944	Евразия

Укажите количество записей, удовлетворяющих условиям:

1) (Процент > 2) И (Процент < 5);
2) (Материк = "С. Америка") ИЛИ (Материк = "Ю. Америка");
3) ((Население > 80 000 000) И (Дата > 01.01.2012)) ИЛИ (Площадь < 500 000).

22. Что такое отчёт?

23. С помощью имеющейся в вашем распоряжении СУБД создайте БД «ОТДЫХ», содержащую две таблицы следующей структуры:
    - Тур (Страна, Вид отдыха, Продолжительность, Стоимость, Название фирмы);
    - Фирма (Название фирмы, Адрес, Телефон, Наличие системы скидок, Процент скидок).

    В первой таблице должно быть не менее 20 записей; во второй — не менее 5 записей.

    Создайте запрос для отображения информации о фирмах (название, адрес телефон), предлагающих пляжный отдых. Создайте на его основе отчёт.

Дополнительные материалы к главе смотрите в авторской мастерской.

Глава 4
СЕТЕВЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

§ 14
Основы построения компьютерных сетей

14.1. Компьютерные сети и их классификация

Компьютерная сеть — это группа (два и более) компьютеров, соединённых каналами передачи данных.

Компьютерные сети обеспечивают:

- быстрый обмен данными между отдельными компьютерами сети;
- совместное использование вычислительных ресурсов, принтеров, модемов, сканеров, устройств внешней памяти и т. п.;
- совместное использование программного обеспечения и баз данных;
- совместную работу пользователей над некоторым заданием или проектом;
- возможность удалённого управления компьютерами (диагностику, настройку и/или установку на них программного обеспечения, оказание других видов удалённой поддержки пользователям и т. п.).

В зависимости от выполняемых в сети функций различают компьютеры-серверы и компьютеры-клиенты:

- сервер (от англ. *server* — обслуживающий) — компьютер, предоставляющий доступ к собственным ресурсам другим компьютерам и/или управляющий распределением ресурсов сети;

- клиент (рабочая станция) — компьютер, использующий ресурсы сервера.

Компьютерные сети могут быть классифицированы по разным основаниям: по территориальной распространённости, по архитектуре, по скорости передачи данных, по назначению, по типу среды передачи данных и др. Рассмотрим некоторые из этих классификаций.

По территориальной распространённости выделяют:

- локальные сети или LAN (англ. Local Area Network) — сети, состоящие из близко расположенных компьютеров;
- глобальные сети или WAN (англ. Wide Area Network) — сети, охватывающие большие территории и включающие большое число компьютеров.

По архитектуре различают:

- одноранговые сети, в которых все компьютеры имеют равные права — каждый компьютер может предоставлять собственные ресурсы другим компьютерам сети и использовать ресурсы остальных. Такая организация позволяет сохранять работоспособность сети при любом количестве и любом сочетании её участников. В одноранговой сети все компьютеры работают независимо друг от друга, у них нет единого центра. Такую сеть сложно обслуживать — руководить доступом к ресурсам, устанавливать и обновлять программное обеспечение на отдельных компьютерах, защищать от вмешательства посторонних пользователей, от вирусных атак и т. п.;
- сети с выделенным сервером — сети, в которых один или несколько компьютеров являются серверами, а все остальные — клиентами. Как правило, сервер мощнее и защищён лучше большинства клиентов. На сервере проще организовать доступ к данным только клиентам с соответствующими правами. Основной недостаток таких сетей в том, что неработоспособность сервера может привести к неработоспособности всей сети.

Скорость передачи данных по сети — это количество бит данных, которые могут быть переданы за одну секунду. Пропускная способность — это максимальная скорость передачи данных. По скорости передачи данных различают:

- низкоскоростные сети (до 10 Мбит/с);
- среднескоростные сети (до 100 Мбит/с);
- высокоскоростные сети (свыше 100 Мбит/с).

14.2. Аппаратное и программное обеспечение компьютерных сетей

Объединение компьютеров в сеть осуществляется с использованием каналов передачи данных — среды передачи данных и оборудования, обеспечивающего передачу данных в этой среде.

По типу среды передачи данных различают сети:

- проводные (кабельные) — средой передачи данных являются кабели (телефонный провод, коаксиальный кабель, витая пара, оптоволоконный кабель);
- беспроводные — средой передачи являются радиоволны в определённом частотном диапазоне.

Сетевые адаптеры — устройства, выполняющие функцию сопряжения компьютера со средой передачи данных.

Какой бы природы ни был сигнал (электрический, оптический, радиосигнал), при передаче по сети на большое расстояние он слабеет. Чтобы сигнал не искажался и не пропадал, его необходимо усиливать. Делается это с помощью специального оборудования, так называемых повторителей, увеличивающих расстояние сетевого соединения путём повторения сигнала «один в один».

Концентраторы и коммутаторы служат для объединения нескольких компьютеров в требуемую конфигурацию локальной вычислительной сети.

Для соединения подсетей (логических сегментов) и различных вычислительных сетей в качестве межсетевого интерфейса применяются коммутаторы, мосты, маршрутизаторы и шлюзы.

Для организации обмена данными между компьютерами сети используется несколько видов программного обеспечения: сетевые компоненты операционной системы, служебные и прикладные программы. Сетевая операционная система связывает все компьютеры и периферийные устройства в сети, координирует их функции, обеспечивает защищённый доступ к данным. Прикладные программы, используемые для получения сетевых услуг, как правило, построены по клиент-серверной технологии и состоят из двух частей:

1) клиентской, предоставляющей возможность обратиться с запросом к ресурсам других компьютеров;
2) серверной, отвечающей на запросы клиентской части.

Чтобы обмен данными между компьютерами сети проходил без потерь и искажений, разнообразные компьютеры, сетевое оборудование и программное обеспечение должны взаимодействовать

по одинаковым чётко определённым правилам. Такие правила называют сетевыми протоколами.

Сетевой протокол — это совокупность особых соглашений, а также технических процедур, которые регулируют порядок и способ осуществления связи между компьютерами, объединёнными в сеть.

Большинство современных компьютерных сетей осуществляет передачу данных на основе стека (набора) протоколов под названием TCP/IP (англ. Transmission Control Protocol/Internet Protocol — протокол управления передачей/межсетевой протокол).

При передаче данные разделяют на отдельные небольшие пакеты, дополняют служебными данными (адресами компьютеров получателя и отправителя, номером пакета и контрольным битом) и передают последовательно друг за другом.

Маршрут передачи определяют маршрутизаторы, которые также следят и за доставкой пакетов. Разные пакеты одного сообщения могут передаваться разными маршрутами. Пакет, по какой-то причине не попавший к адресату, отправляется повторно. Повторно передаются и пакеты, в которых во время передачи возникают искажения данных. В пункте назначения все пакеты соединяются, и данные приобретают первоначальный вид. Благодаря разделению данных на отдельные пакеты их передача по сети происходит быстро и надёжно — она возможна даже при выходе из строя части сети. В такой ситуации маршрутизаторы определят новый маршрут для прохождения пакета в обход повреждённого участка.

Правила разбивки данных на пакеты, их доставки к адресату и объединения пакетов в единое целое определяет протокол TCP.

Пересылка пакетов между компьютерами, которые могут иметь разную архитектуру, использовать разные операционные системы и относиться к разным сетям, осуществляется на основе протокола IP.

14.3. Работа в локальной сети

Локальная сеть — это сеть, состоящая из близко расположенных компьютеров, чаще всего находящихся в одной комнате, в одном или нескольких близко расположенных зданиях.

Локальные сети предназначены для ограниченного круга пользователей.

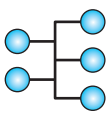
Одной из важнейшей характеристик локальных сетей является скорость передачи данных, поэтому компьютеры соединяются с помощью высокоскоростных адаптеров и высокоскоростных линий связи. Кроме того, локальные сети должны обладать открытостью и гибкостью: пользователи должны иметь возможность добавлять в сеть или перемещать компьютеры и другие устройства, при необходимости отключать их без прерываний в работе сети и т. д. Эти характеристики во многом определяются конфигурацией или топологией сети.

Топология — это конфигурация сети, способ соединения её элементов друг с другом.

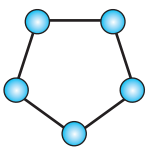
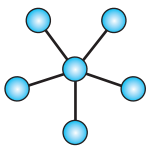
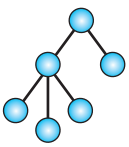
Топологию сети удобно представлять с помощью графа, вершинам которого соответствуют компьютеры (иногда — другое оборудование), а рёбрам — физические связи между ними. Чаще всего используются шинная, кольцевая, радиальная и древовидная топологии. Их описание, основные достоинства и недостатки представлены в табл. 4.1.

Таблица 4.1

Топологии локальных сетей

Шинная топология (общая шина)	
Описание	
Все компьютеры сети подключаются к одному кабелю. Этот кабель используется совместно всеми рабочими станциями по очереди. При таком типе соединения все сообщения, посылаемые каждым отдельным компьютером, принимаются всеми остальными компьютерами в сети	
Достоинства	**Недостатки**
Выход из строя отдельных компьютеров не влияет на работоспособность всей сети	При обрыве кабеля (единого для всей сети) нарушается работа всей сети

Окончание табл. 4.1

<table>
<tr><th colspan="2">Кольцевая топология (кольцо)</th></tr>
<tr><th colspan="2">Описание</th></tr>
<tr><td colspan="2">Данные передаются по кольцу от одного компьютера к другому. Если один компьютер получает данные, предназначенные для другого компьютера, то он передаёт их дальше (по кольцу). Если компьютер распознаёт данные как свои, то копирует их себе во внутренний буфер  </td></tr>
<tr><th>Достоинства</th><th>Недостатки</th></tr>
<tr><td>Балансировка нагрузки, возможность и удобство прокладки кабеля</td><td>Физические ограничения на общую протяжённость сети</td></tr>
<tr><th colspan="2">Радиальная топология (звезда)</th></tr>
<tr><th colspan="2">Описание</th></tr>
<tr><td colspan="2">Каждый компьютер через специальный сетевой адаптер подключается отдельным кабелем к объединяющему устройству  </td></tr>
<tr><th>Достоинства</th><th>Недостатки</th></tr>
<tr><td>Большая надёжность</td><td>Сравнительно высокая стоимость оборудования и ограниченное количество узлов в сети</td></tr>
<tr><th colspan="2">Древовидная топология (иерархическая звезда)</th></tr>
<tr><th colspan="2">Описание</th></tr>
<tr><td colspan="2">Образуется соединением между собой нескольких звездообразных топологий. В настоящее время самый распространённый способ построения как локальных, так и глобальных сетей  </td></tr>
<tr><th>Достоинства</th><th>Недостатки</th></tr>
<tr><td>Большая надёжность, соответствие реальной структуре информационных потоков</td><td>Необходимость в дополнительном сетевом оборудовании</td></tr>
</table>

В одноранговой сети компьютеры, как правило, объединяют в рабочую группу. Рабочая группа — это группа компьютеров локальной сети, пользователи которых выполняют похожие задания и осуществляют регулярный обмен данными. Например, в локальной сети школы в одну рабочую группу могут быть объединены компьютеры кабинета информатики. Рабочим группам дают имена, например Class или Administration.

В сетях с выделенным сервером компьютеры, как правило, объединяют в домены. Домен (от англ. *domain* — владение) — это группа компьютеров, централизованно обслуживаемых общим сервером, который руководит распределением прав доступа пользователей к ресурсам сети. Как и рабочей группе, домену дают имя. В большой локальной сети может быть несколько доменов.

Каждому компьютеру в составе рабочей группы или домена дают уникальное имя. Например, компьютер учителя в кабинете информатики может иметь имя Teacher, а имена компьютеров учеников могут быть PC1, PC2 и т. д.

Часто на одном компьютере работает несколько пользователей, например на уроках информатики ученики разных классов работают на одних и тех же компьютерах, имеющихся в кабинете информатики. Для того чтобы обеспечить пользователям возможность работы с индивидуальными настройками операционной системы, упростить доступ к их собственным документам и ограничить доступ к чужим файлам и папкам, используют учётные записи.

Предоставление общего доступа к папкам даёт возможность пользователям работать с файлами, хранящимися на дисках других компьютеров локальной сети. Кроме того, общий доступ можно установить, например, к принтеру или сканеру.

Ещё одним преимуществом локальной сети является возможность подключения к глобальной сети всех компьютеров через один из них, имеющий в неё выход. Этот компьютер выполняет функцию шлюза — устройства, обеспечивающего соединение двух сетей: локальной и глобальной.

14.4. Как устроен Интернет

Глобальная сеть — это сеть, предназначенная для объединения большого числа отдельных компьютеров и локальных сетей, расположенных на значительном удалении (сотни и тысячи километров) друг от друга.

Глобальные сети ориентированы на обслуживание неограниченного круга пользователей. Самый впечатляющий пример глобальной сети — Интернет.

Интернет — это глобальная компьютерная сеть, в которой многочисленные научные, корпоративные, государственные и другие сети, а также персональные компьютеры отдельных пользователей соединены между собой каналами передачи данных.

Основой аппаратной структуры сети Интернет можно считать мощные компьютеры (узлы) и связывающие их высокоскоростные магистральные каналы передачи данных. Компьютерный узел, как правило, представляет собой несколько мощных компьютеров, постоянно подключённых к сети. Организации, имеющие в собственности и обслуживающие это оборудование, являются первичными провайдерами (от англ. *provider* — поставщик) услуг Интернета. Это так называемый, первый уровень доступа к Интернету. К первичным провайдерам присоединяются провайдеры следующих уровней, которые, в свою очередь, обеспечивают доступ к каналам Интернета своим клиентам — провайдерам более низкого уровня, локальным сетям и отдельным пользователям. Надёжность функционирования Интернета обеспечивается наличием большого количества каналов связи между входящими в него сетями.

Интернет является совокупностью сетей, имеющих различную географическую и организационную принадлежность. У каждой из этих сетей может быть владелец, но в целом Интернет не принадлежит никому.

Так как Интернет не имеет единого внешнего управления, его нельзя единовременно выключить целиком.

Координирует развитие Интернета общественная организация Интернет Сообщество (Internet Society, ISOC).

За каждым компьютерным узлом в Интернете закреплён постоянный адрес, называемый IP-адресом. IP-адреса получают и компьютеры пользователей сети Интернет, но в отличие от адресов узловых компьютеров их адреса действуют лишь во время подключения пользователя к сети и изменяются при каждом новом сеансе связи.

IP-адрес представляет собой 32-битный идентификатор, например: 01010101.10001110.00010011.00011110.

Точками 32-битная цепочка разделена только для более удобного её восприятия человеком, которому в отличие от технических устройств трудно работать с длинными последовательностями нулей и единиц. Именно поэтому в большинстве случаев мы используем запись IP-адреса в виде четырёх разделённых точками десятичных чисел — от 0 до 255 каждое.

Например, десятичная запись представленного выше адреса будет иметь вид: 85.142.19.30.

Интернет является сетью сетей, и система IP-адресации учитывает эту структуру. IP-адрес состоит из двух частей, одна из которых определяет адрес сети, а вторая — адрес самого узла в этой сети. При этом деление адреса на части определяется маской — 32-битным числом, в двоичной записи которого сначала стоят единицы, а потом — нули. Первая часть IP-адреса, соответствующая единичным битам маски, относится к адресу сети. Вторая часть IP-адреса, соответствующая нулевым битам маски, определяет числовой адрес узла в сети. Адрес сети получается в результате применения поразрядной конъюнкции к IP-адресу узла и маске.

Пример 1. Пусть IP-адрес узла равен 231.165.215.131, а маска равна 255.255.110.0. Требуется выяснить адрес сети.

Чтобы найти адрес сети, применим к IP-адресу узла и маске поразрядную конъюнкцию:

$$\begin{array}{l} 231.165.215.131 \\ \& \; 255.255.110.\ \ 0 \\ \hline ? \end{array}$$

Вспомним, что десятичный ноль может быть представлен цепочкой из восьми нулей, а $255_{10} = 11111111_2$.

Что касается операции конъюнкции (логического умножения), то для неё справедливы следующие равенства: $A \,\&\, 1 = A$, $A \,\&\, 0 = 0$, где A — некоторая логическая переменная.

На этом основании, пропустив этап преобразования операндов в двоичную систему счисления, можем заключить:

1) результатом поразрядной конъюнкции любого целого числа A (от 0 до 255_{10}) и числа 255_{10} будет само A;
2) результатом поразрядной конъюнкции любого целого числа A (от 0 до 255_{10}) и числа 0 будет число 0.

Таким образом:

	231	.	165	.	215	.	131
&	255	.	255	.	110	.	0
	231	.	165	.	?	.	0

Для выполнения поразрядной конъюнкции чисел 215_{10} и 110_{10} переведём их в двоичную систему счисления.

Вспомнить возможные способы перевода целых десятичных чисел вам помогут следующие записи.

215	107	53	26	13	6	3	1
1	1	1	0	1	0	1	1

$215_{10} = 11010111_2$.

$110_{10} = 64 + 32 + 8 + 4 + 2 = 1101110_2 = 01101110_2$.

Выполним поразрядную конъюнкцию:

$$\begin{array}{r} 11010111 \\ \& \; 01101110 \\ \hline 01000110 \end{array}$$

Выполним перевод двоичного числа 01000110 в десятичную систему счисления:

Степени основания (2)

$$\begin{array}{cccccccc} 6 & 5 & 4 & 3 & 2 & 1 & 0 & \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & = 1 \cdot 2^6 + 1 \cdot 2^2 + 1 \cdot 2^1 = 70. \end{array}$$

Запишем искомый адрес сети: 231.165.70.0

Пример 2. Для узла с IP-адресом 227.195.208.12 адрес сети равен 227.195.192.0. Какой в этом случае может быть маска?

Так как адрес сети получается в результате применения поразрядной конъюнкции к IP-адресу узла и маске, то мы можем записать:

IP-адрес узла:	227.195.208.12		227.195.208.12
Маска:	& *xxx. xxx. xxx.xxx*	→	& 255.255. *xxx.*0
Адрес сети:	227.195.192.0		227.195.192.0

Мы видим, что 1-й и 2-й байты маски — единичные, а 4-й байт — нулевой. Будем «реконструировать» 3-й байт маски.

$208_{10} = 11010000_2$, $192_{10} = 11000000_2$.

Если 3-й байт маски представить в виде $xxxxxxxx_2$, то можно записать:

$$\begin{array}{r} 1\,1\,0\,1\,0\,0\,0\,0 \\ \&\ x\,x\,x\,x\,x\,x\,x\,x \\ \hline 1\,1\,0\,0\,0\,0\,0\,0 \end{array}$$

Первая, вторая и четвёртая слева цифры, принадлежащие рассматриваемому байту маски, определяются однозначно и равны соответственно 1, 1 и 0:

$$\begin{array}{r} 1\,1\,0\,1\,0\,0\,0\,0 \\ \&\ 1\,1\,x\,0\,x\,x\,x\,x \\ \hline 1\,1\,0\,0\,0\,0\,0\,0 \end{array}$$

Из того, что маска — 32-битное число, в двоичной записи которого сначала стоят единицы, а потом — нули, следует, что после нуля, стоящего на четвёртом месте, могут следовать только нули:

$$\begin{array}{r} 1\,1\,0\,1\,0\,0\,0\,0 \\ \&\ 1\,1\,x\,0\,0\,0\,0\,0 \\ \hline 1\,1\,0\,0\,0\,0\,0\,0 \end{array}$$

Так как $0\ \&\ x = 0$ при любом x, то для третьего байта маски возможны два варианта:

1) $11000000_2 = 192_{10}$, вся маска: 255.255.192.0;
2) $11100000_2 = 224_{10}$, вся маска: 255.255.224.0.

Мы рассмотрели структуру адреса по так называемому протоколу IPv4, согласно которому IP-адрес имеет длину 32 бита. Таких адресов достаточно много — более 4 миллиардов ($2^{32} - 1 = 4\ 294\ 967\ 295$).

По данным Международного союза электросвязи (пресс-релиз 26 мая 2015 года) при населении Земли в 7,2 миллиарда человек 3,2 миллиарда из них являются пользователями Интернета. Это говорит о том, что запас четырёхбайтовых адресов уже фактически исчерпан.

В связи с этим разработан протокол IPv6, согласно которому IP-адрес имеет длину 128 бит. Возможное пространство адресов при этом столь огромно, что может обеспечить 300 миллионов IP-адресов на каждого жителя Земли!

Согласно протоколу IPv6, адрес представляет собой цепочку из 128 нулей и единиц, разделённую на области по 16 бит. Например: 0010000111011010.0000000011010011.0000000000000000.0000000000000000.0000001010101010.0000000011111111.1111111000101000.1001110001011010.

Каждая 16-разрядная область двоичного кода преобразуется в шестнадцатеричный код (вспомните «быстрый» перевод целых двоичных чисел в шестнадцатеричную систему счисления с помощью тетрад). Полученные группы из четырёх шестнадцатеричных цифр разделяются двоеточиями. Шестнадцатеричная запись рассмотренного выше адреса будет иметь вид:

21DA:00D3:0000:0000:02AA:00FF:FE28:9C5A.

Если одна или более групп подряд равны 0000, то они могут быть опущены и заменены на двойное двоеточие:

21DA:00D3::02AA:00FF:FE28:9C5A.

Запись адреса в новом стандарте также можно представить восьмью целыми десятичными числами в диапазоне от 0 до 65 535 каждое, разделёнными двоеточием.

Наряду с цифровыми IP-адресами в Интернете действуют более удобные и понятные для пользователей символьные адреса.

Например, IP-адресу 87.242.99.97 соответствует символьный адрес metodist.lbz.ru. В отличие от числового этот символьный адрес говорит пользователю о его принадлежности российскому сегменту сети (ru); возможно, некоторые пользователи узнают в нём адрес издательства «БИНОМ. Лаборатория знаний» (lbz) и поймут, что речь идёт о методической поддержке учебного процесса (metodist).

Адрес, представляющий собой символьную строку, составленную из разделённых точками слов или их сокращений, называется **доменным именем**.

Доменные имена имеют серверы Интернета. Каждый компьютер, подключаемый к Интернету, получает IP-адрес, но при этом он может не иметь доменного имени.

Система доменных имён DNS (Domain Name System) имеет древовидную структуру. Узлы этой структуры называются доменами.

Домéн (от фр. *dominion* — область) — узел в дереве имён, вместе со всеми подчинёнными ему узлами, иначе говоря, это именованная ветвь или поддерево в дереве имён.

Часть доменного имени, записанная после последней точки, является доменом верхнего уровня. Домены верхнего уровня

определены международным соглашением. Они делятся на два вида:

1) административные (по типу организации), например: gov, edu, org, com;
2) географические, например: ru, by, cy, uk.

Владельцем домена может быть страна, регион, организация или отдельный человек. Обычный пользователь не может зарегистрировать домен верхнего уровня, но может зарегистрировать домен, например, второго или третьего уровня. Каждый домен любого уровня может содержать множество подчинённых доменов.

Структура доменного имени отражает порядок следования узлов в иерархии: доменное имя читается слева направо от доменов низшего уровня к доменам высшего уровня. Чем «выше» уровень домена, тем правее он записывается в имени.

Для преобразования доменного имени в IP-адрес и наоборот служит распределённая база данных DNS, функционирующая на основе иерархии DNS-серверов, каждый из которых является «держателем» некоторой доменной зоны и отвечает на касающиеся её запросы. Каждый сервер, отвечающий за доменную зону, может делегировать ответственность за некоторую часть домена другому серверу, что позволяет возложить ответственность за актуальность информации на серверы различных организаций (людей), отвечающих только за «свою часть» доменного имени.

14.5. История появления и развития компьютерных сетей

История появления и развития компьютерных сетей тесно связана с развитием вычислительной техники и коммуникаций. В ней можно выделить несколько этапов.

- 1950–1960 гг. Компьютеры представляют собой громоздкие устройства, требующие длительного времени для обработки информации. Создаются отдельные терминалы с собственными устройствами ввода-вывода, напрямую работающие с общим компьютером (мэйнфреймом). Терминалы, физически удалённые от мэйнфрейма, — первый прообраз компьютерной сети.
- 1960–1970 гг. Разрабатываются технические принципы компьютерной сети. В 1969 г. появляется ARPANET — первая глобальная сеть невоенного назначения, объединяющая суперкомпьютеры нескольких научно-исследовательских центров США, использующая для передачи данных телефонные сети.

- 1970–1980 гг. Появляются большие интегральные схемы, первые мини-компьютеры, первые нестандартные, настраиваемые вручную локальные сети. Появляются первые сетевые стандарты. Начинает функционировать электронная почта.
- 1980–1990 гг. Создаются персональные компьютеры. Принимается протокол TCP/IP, вводится система доменных имён DNS. Появляется Интернет в виде, близком к современному. Появляются стандартные технологии локальных сетей (Ethernet — 1980 г., Token Ring, FDDI — 1985 г.). Начинается коммерческое использование Интернета.
- 1990–2000 гг. Появляются первые интернет-сайты. Интернет объединяет локальные сети и становится средством массовой коммуникации. Телетехнологии (телемосты, видеоконференции) встраиваются в глобальную сеть.
- 2000–2010 гг. Производится массовое подключение отдельных пользователей и локальных сетей к Интернету. Используются беспроводные сети, резко снижается стоимость передачи единицы информации. Доступ к сети Интернет и электронной почте встраивается в мобильные телефоны. Создаются и получают широкое распространение сетевые средства массовой информации, интернет-магазины, цифровые библиотеки, дистанционное образование, социальные сети.
- 2010–2015 гг. Активно разворачиваются цифровые услуги населению, создаются облачные ресурсы и действующие на их основе мобильные сервисы, разворачивается глобальная сеть онлайн-обучения.

САМОЕ ГЛАВНОЕ

Компьютерная сеть — это группа (два и более) компьютеров, соединённых каналами передачи данных, обеспечивающая:

- быстрый обмен данными между отдельными компьютерами сети;
- совместное использование вычислительных ресурсов и внешних (периферийных) устройств;
- совместное использование программного обеспечения и баз данных;
- совместную работу пользователей над некоторым заданием или проектом;
- возможность удалённого управления компьютерами.

В зависимости от выполняемых в сети функций различают компьютеры-серверы и компьютеры-клиенты.

Компьютерные сети могут быть классифицированы по разным основаниям: по территориальной распространённости, по архитектуре, по скорости передачи данных, по назначению, по типу среды передачи данных и др.

Объединение компьютеров в сеть осуществляется с использованием каналов передачи данных — среды передачи данных и оборудования, обеспечивающего передачу данных в этой среде.

Аппаратные компоненты компьютерных сетей — это сетевые адаптеры, повторители, концентраторы, коммутаторы, мосты, маршрутизаторы, шлюзы и другое оборудование.

Для организации обмена данными между компьютерами сети используются сетевые компоненты операционной системы, служебные и прикладные программы.

Сетевой протокол — это совокупность особых соглашений, а также технических процедур, которые регулируют порядок и способ осуществления связи между компьютерами, объединёнными в сеть. Большинство современных компьютерных сетей осуществляет передачу данных на основе стека (набора) протоколов под названием TCP/IP (англ. Transmission Control Protocol/Internet Protocol — протокол управления передачей/межсетевой протокол).

Локальная сеть — это сеть, состоящая из близко расположенных компьютеров, чаще всего находящихся в одной комнате, в одном или нескольких близко расположенных зданиях.

Топология — это конфигурация сети, способ соединения её элементов друг с другом. В настоящее время самым распространённым способом построения компьютерных сетей является древовидная топология.

Безусловным преимуществом локальной сети является возможность подключения к глобальной сети всех компьютеров через один из них, имеющий в неё выход.

Глобальная сеть — это сеть, предназначенная для объединения большого числа отдельных компьютеров и локальных сетей, расположенных на значительном удалении (сотни и тысячи километров) друг от друга. Интернет — это глобальная компьютерная сеть, в которой многочисленные научные, корпоративные, государственные и другие сети, а также персональные компьютеры отдельных пользователей соединены между собой каналами передачи данных.

Каждый компьютер, подключаемый к Интернету, получает свой уникальный 32-битный идентификатор, называемый IP-адресом.

Наряду с цифровыми IP-адресами в Интернете действуют более удобные и понятные для пользователей символьные адреса, называемые доменными именами. Система доменных имён DNS (Domain Name System) имеет древовидную структуру. Узлы этой структуры называются доменами.

Вопросы и задания

1. Что такое компьютерная сеть? Какие возможности она предоставляет?
2. Какие функции выполняет компьютер-сервер в сети? Какой компьютер называют клиентом?
3. По каким основаниям можно классифицировать компьютерные сети?
4. Какую сеть называют одноранговой? Что представляет собой сеть с выделенным сервером?
5. Назовите виды компьютерных сетей по территориальной распространённости.
6. Кроме LAN и WAN по территориальной распространённости выделяют также сети BAN, PAN, CAN и MAN. Найдите в дополнительных источниках информацию об этих сетях и подготовьте о них краткое сообщение.
7. Какие среды передачи данных могут использоваться в компьютерных сетях? Приведите примеры.
8. Выясните, каковы максимальные скорость и расстояние передачи данных, обеспечиваемые в беспроводной сети Wi-Fi. Используйте дополнительные источники информации.
9. Какие аппаратные компоненты компьютерных сетей вам известны?
10. Найдите в дополнительных источниках информацию о функциях, внешнем виде и характеристиках сетевых адаптеров, повторителей, концентраторов, коммутаторов, мостов и маршрутизаторов. Представьте найденную информацию в форме презентации.
11. Музыкальный фрагмент был записан в формате стерео (двухканальная запись), затем оцифрован и сохранён в виде файла без использования сжатия данных. Получившийся файл был передан в город А по каналу связи за 60 секунд. Затем тот же музыкальный фрагмент был повторно записан в формате моно и оцифрован с разрешением в 2 раза выше и

частотой дискретизации в 2 раза меньше, чем в первый раз. Сжатие данных не производилось. Полученный файл был передан в город Б. Пропускная способность канала связи с городом Б в 3 раза ниже, чем канала связи с городом А. Сколько секунд длилась передача файла в город Б?

12. Какое программное обеспечение используют в компьютерных сетях? В чём суть клиент-серверного программного обеспечения?
13. Что представляют собой сетевые протоколы? Для чего они нужны?
14. На основе какого стека (набора) протоколов осуществляется передача данных в современных сетях? Назовите его составляющие и опишите их функции.
15. Какая сеть называется локальной?
16. Что такое топология сети? Какие бывают топологии локальной сети? Какая топология является наиболее распространённой в наше время?
17. Исследуйте локальную сеть кабинета информатики в вашей школе. Эта сеть одноранговая или с выделенным сервером? Какая у неё топология? Как организовано подключение к сети Интернет?
18. Какие сети называются глобальными?
19. Что такое Интернет?
20. Составьте «Топ–10» стран по числу пользователей Интернета. Как вы можете объяснить полученные результаты?
21. Что представляет собой IP-адрес в стандарте IPv4? Почему каждое из фигурирующих в нём четырёх десятичных чисел заключено в диапазоне от 0 до 255?
22. Восстановите IP-адрес по его фрагментам:

2.132	20	.82	2.19

23. Чему равен адрес сети, если IP-адрес узла равен 211.64.254.139, а маска равна 255.255.240.0?
24. Для узла с IP-адресом 117.191.84.37 адрес сети равен 117.191.80.0. Какой в этом случае может быть маска?
25. Что называется доменным именем? Приведите примеры доменных имён.
26. Назовите виды и приведите примеры доменов верхнего уровня.

27. Объясните назначение DNS-серверов.
28. Каковы основные вехи в истории появления и развития компьютерных сетей? Подготовьте презентацию на эту тему.
29. Найдите в дополнительных источниках информацию и подготовьте небольшое сообщение о Всемирном дне Интернета.

§ 15
Службы Интернета

Интернет играет важную роль в жизни современного человека, являясь:

1) средством доступа к общим информационным ресурсам;
2) средством коммуникации (общения) между удалёнными пользователями.

Благодаря Интернету, ставшему в наше время мобильным, пользователь, где бы он ни находился, всегда имеет доступ к важным документам, почте, может получить информацию по любому интересующему его вопросу.

Средства обеспечения определённых услуг для пользователей сети Интернет принято называть **службами** (сервисами).

Для каждой службы Интернета существует своя программа-сервер, клиентская программа и свой протокол, обеспечивающий взаимодействие программы-клиента с сервером. Воспользоваться какой-либо службой Интернета можно только в том случае, если на компьютере установлено соответствующее программное обеспечение (клиентская программа).

15.1. Информационные службы

Информационные службы предоставляют пользователям возможность доступа к разнообразным информационным ресурсам (файлам, документам), хранящимся в Интернете.

Основными информационными службами Интернета являются Всемирная паутина, служба передачи файлов, служба файлообменников.

Всемирная паутина (англ. World Wide Web, WWW) представляет собой распределённую по всему миру информационную

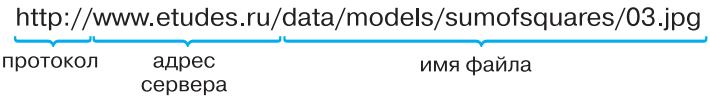
систему, состоящую из миллиардов взаимосвязанных электронных документов — веб-страниц.

Каждый ресурс (страница, документ, файл) в Интернете имеет свой уникальный адрес или URL (англ. Universal Resource Locator — универсальный указатель ресурсов).

Адрес документа в Интернете (URL) состоит из следующих частей:

1) название протокола со знаками :// в конце названия;
2) доменное имя сервера со знаком / в конце имени;
3) полное имя файла на сервере, где он находится.

Например:



Ещё одной информационной службой является **служба передачи файлов**, предоставляющая пользователям услуги по хранению и обеспечению доступа к большому количеству файлов: системному и прикладному программному обеспечению, электронным книгам, музыке, видео и т. п. Передача файлов осуществляется по протоколу FTP (англ. File Transfer Protocol — протокол передачи файлов), именем которого названа и сама служба.

FTP-сервер — программа, позволяющая хранить файлы и передавать их по протоколу FTP.

FTP-клиент — программа, позволяющая подключаться к удалённому FTP-серверу и получать/передавать файлы по протоколу FTP. Существует множество бесплатных и платных FTP-клиентов. Встроенными возможностями для работы по протоколу FTP обладают браузеры, менеджеры файлов и некоторые другие программы.

На серверах службы FTP хранятся FTP-архивы — большие хорошо структурированные коллекции объединённых общей тематикой файлов. Среди них можно выделить: 1) коммерческие серверы и серверы ограниченного доступа, которые доступны только для зарегистрированных пользователей; 2) серверы с открытым доступом для всех желающих.

Если вы хотите поделиться файлами (например, фотографиями, сделанными во время летнего отдыха) со своими друзьями, то можно воспользоваться **файлообменником** — службой, предоставляющей пользователю место под его файлы и круглосуточный доступ к ним, как правило, по протоколу http. Загрузив свои файлы на сервер файлообменника, вы получите ссылку, которую

сможете переслать кому-то конкретно или опубликовать для широкого круга лиц.

В последнее время всё большую популярность среди пользователей приобретают **облачные хранилища** данных (англ. cloud storage). С точки зрения пользователя данные хранятся и обрабатываются на одном большом виртуальном сервере, в так называемом «облаке», которое физически представляет собой многочисленные серверы, удалённые друг от друга географически, вплоть до расположения на разных континентах.

15.2. Коммуникационные службы

К основным коммуникационным службам относятся электронная почта, форум, чат, IP-телефония.

Электронная почта — e-mail (от англ. *electronic mail*) — одна из самых первых коммуникационных служб Интернета. Она предоставляет возможность передавать электронные письма (текстовые сообщения и прикреплённые к ним файлы) от пользователя-отправителя одному или группе адресатов.

Почтовый клиент — программа, помогающая составлять и посылать электронные сообщения, получать и отображать письма на компьютере пользователя.

Почтовый сервер — программа, пересылающая сообщения из почтовых ящиков на другие серверы или на компьютер пользователя по запросу его почтового клиента. На почтовом сервере создают почтовые ящики для пользователей с определённым именем и паролем для доступа. Адрес электронной почты имеет вид:

<имя_пользователя>@<имя_сервера>

Почтовая служба основана на следующих протоколах:

- SMTP (англ. Simple Mail Transfer Protocol — простой протокол передачи почты) — почтовый протокол, служащий для отправки сообщений с компьютера-клиента на почтовый сервер, а также для пересылки почты между серверами;
- POP3 (англ. Post Office Protocol — протокол почтового офиса версия 3) — почтовый протокол для получения доступа к почтовому ящику на сервере и пересылки сообщений на компьютер-клиент;
- IMAP (англ. Internet Message Access Protocol — протокол доступа к сообщениям Интернета) — протокол для доступа к почтовому ящику на сервере, позволяющий управлять корреспонденцией на сервере.

Коммуникационные службы предоставляют пользователям сети возможность обмениваться новостями, обсуждать проблемы, проводить дискуссии и т. п.

Обсуждение определённой темы группой собеседников, находящихся на значительном расстоянии, называют **телеконференцией**. Видеоконференция предусматривает использование средств передачи видеоизображений.

Долгосрочные (постоянно действующие) телеконференции, в ходе которых собеседники посылают и читают текстовые сообщения в удобное для них время, называют **форумами**.

Форум предлагает набор разделов для обсуждения. Зарегистрированные на сайте пользователи, посылая свои сообщения, могут создавать внутри разделов темы и вести обсуждения в рамках этих тем. Сообщение и все ответы на него образуют «ветку» форума. Незарегистрированные пользователи получают статус гостей, которые могут просматривать ветки форума, но не имеют права принимать участие в обсуждениях. За соблюдением правил следят модераторы, имеющие право редактировать, перемещать и удалять чужие сообщения в определённом разделе или теме.

Службы интерактивного общения предоставляют возможность двум или группе пользователей обмениваться текстовыми сообщениями через Интернет в реальном времени. Первой такой службой была служба IRC (англ. Internet Relay Chat — ретранслируемый интернет-разговор) или просто **чат** (от англ. *chat* — болтать). В отличие от телеконференций, где обсуждение темы открыто всем, в системе IRC общение происходит только в пределах одного канала, в работе которого принимают участие обычно лишь несколько человек. Подключившись к каналу, пользователь видит на экране сообщения других участников. Сообщение, введённое любым участником, практически немедленно появляется на канале, и все остальные участники обсуждения могут «высказаться» в ответ. Для работы службы разработан одноимённый протокол IRC. IRC-сервер — программа, обеспечивающая работоспособность системы IRC и хранящая информацию о каналах и подключённых пользователях. IRC-клиент — программа для подключения к IRC-серверу и ведения беседы.

Некоторые службы интерактивного общения дают возможность вести обсуждение только двум собеседникам. Каждому пользователю при регистрации в таких службах предоставляется индивидуальный код, и каждый сам создаёт список контактов — лиц, с которыми он желает общаться.

IP-телефония — служба, обеспечивающая передачу телефонных разговоров абонентов по сети Интернет. Передача данных в IP-телефонии осуществляется на основе набора протоколов VoIP (англ. Voice over Internet Protocol — голос поверх протокола Интернета). Skype (скайп) — одна из самых популярных программ, позволяющих обмениваться мгновенными сообщениями, использовать голосовую связь и видеозвонки. С помощью Skype можно: общаться с одним или сразу с несколькими людьми, пересылать файлы, вместо изображения с веб-камеры передавать изображение с экрана монитора. Часто эта программа используется для проведения видеоконференций.

В последнее время среди пользователей Интернета широкое распространение получили **социальные сети** — интерактивные многопользовательские веб-сайты, содержание (контент) которых создаётся самими участниками сети. Такие сайты представляют собой автоматизированные социальные среды, позволяющие общаться группам пользователей, объединённых общими интересами.

15.3. Сетевой этикет

В сети Интернет существуют негласные правила поведения, так называемый сетевой этикет. Кратко, суть сетевого этикета может быть выражена одной фразой: «Уважайте своих невидимых партнёров по Сети!».

Приведём основные правила сетевого этикета, которых следует придерживаться в почтовой переписке, а также при использовании других сервисов сети Интернет.

1. Ясно идентифицируйте себя.
2. Знайте и уважайте своего адресата.
3. Указывайте тему сообщения.
4. Пишите грамотно, кратко. Давайте чёткий ответ на поставленный вопрос.
5. В текстовых сообщениях можете выражать эмоции с помощью небольших рисунков, называемых смайликами.
6. Не запрашивайте подтверждения получения сообщения без необходимости.
7. Не допускайте спама — бессодержательных, навязчивых или грубых сообщений в адрес другого лица или группы лиц.
8. Не надейтесь на полную конфиденциальность переписки.

САМОЕ ГЛАВНОЕ

Средства обеспечения определённых услуг для пользователей сети Интернет принято называть службами (сервисами). Для каждой службы Интернета существует своя программа-сервер, клиентская программа и свой протокол, обеспечивающий взаимодействие программы-клиента с сервером. Воспользоваться какой-либо службой Интернета можно только в том случае, если на компьютере установлено соответствующее программное обеспечение (клиентская программа).

Информационные службы предоставляют пользователям возможность доступа к разнообразным информационным ресурсам (файлам, документам), хранящимся в Интернете. Основными информационными службами Интернета являются Всемирная паутина, служба передачи файлов, служба файлообменников.

К основным коммуникационным службам относятся электронная почта, форум, чат, IP-телефония.

В последнее время среди пользователей Интернета широкое распространение получили социальные сети — интерактивные многопользовательские веб-сайты, содержание (контент) которых создаётся самими участниками сети. Такие сайты представляют собой автоматизированные социальные среды, позволяющие общаться группам пользователей, объединённых общими интересами.

В сети Интернет существуют негласные правила поведения, так называемый сетевой этикет. Кратко, суть сетевого этикета может быть выражена одной фразой: «Уважайте своих невидимых партнёров по Сети!».

Вопросы и задания

1. Что понимается под службой Интернета?
2. В чём различие между информационными и коммуникационными службами Интернета?
3. Назовите известные вам информационные службы Интернета и объясните их назначение.
4. Как устроен универсальный указатель ресурса в Интернете? Для чего он предназначен?
5. Доступ к файлу http.txt, находящемуся на сервере www.net, осуществляется по протоколу ftp. Запишите URL этого ресурса.
6. Исследуйте достоинства и недостатки облачных хранилищ данных. Подготовьте небольшое сообщение на эту тему.

7. Назовите известные вам коммуникационные службы Интернета.
8. Сравните электронную почту и обычную (бумажную) почту по составу элементов и принципам работы.
9. Сравните возможности доступа к почте по протоколам POP3 и IMAP. Укажите достоинства и недостатки каждого из них.
10. Выясните происхождение слов «телеконференция» и «форум».
11. Сравните понятия «телеконференция» и «видеоконференция».
12. Что представляют собой социальные сети?
13. Что такое сетевой этикет? Каковые его основные правила?

§ 16 Интернет как глобальная информационная система

16.1. Всемирная паутина

Напомним основные понятия, касающиеся Всемирной паутины.

Веб-страница может содержать текст, мультимедийные объекты (графическую, аудио- и видеоинформацию), гиперссылки на файлы или другие веб-страницы, а также всевозможные активные компоненты, например формы, позволяющие установить обратную связь между пользователем и веб-страницей посредством типовых элементов управления (текстовых полей, кнопок и т. п.).

Гиперссылка — некоторое ключевое слово или объект в документе, с которым связан переход к другому документу. Текст, в котором используются гиперссылки, называется **гипертекстом**.

Веб-сайт — группа веб-страниц, связанных единой темой, общим стилем оформления и взаимными гиперссылками.

Браузер (от англ. *browse* — просматривать) — специальная программа для просмотра веб-страниц.

Современные браузеры обладают возможностями загрузки и отображения веб-страниц, сохранения веб-страниц на носителях

данных, сохранения истории посещения веб-страниц, создания каталога избранных ресурсов, поиска на веб-странице фрагмента текста, просмотра HTML-кода веб-страницы, печати содержимого веб-страницы и др.

Веб-страница с точки зрения её разработчика — это файл, содержащий собственно текст, несущий определённую информацию для пользователя, и служебную информацию для браузера (тэги разметки) на языке HTML (англ. HyperText Markup Language — язык разметки гипертекста). Тэги разметки представляют собой определённые стандартом HTML последовательности символов, являющиеся инструкциями для программы просмотра. Согласно этим инструкциям, браузер располагает текст на экране, включает в него рисунки, хранящиеся в отдельных графических файлах, и формирует гиперсвязи с другими документами или ресурсами Интернета.

Веб-страницы предназначены для воспроизведения на самых разных экранах самых разных компьютеров. Поэтому они не имеют «жёсткого» форматирования. Оформление веб-страницы выполняется непосредственно во время её воспроизведения на компьютере клиента в соответствии с настройками используемого браузера.

HTML — один из так называемых «веб-стандартов» («стандартов Web»), по которым разрабатываются сайты во всём мире. Ещё одним из таких стандартов является технология CSS (англ. Cascading Style Sheets — каскадные таблицы стилей) — формальный язык описания внешнего вида документа, составленного с использованием языка разметки. Эта технология позволяет принципиально разделить содержание и представление документа:

- описание содержания и логической структуры веб-страницы производится с помощью HTML или других языков разметки;
- описание внешнего вида веб-страницы производится с помощью CSS.

Такое разделение позволяет применять единый заранее разработанный стиль оформления для многих схожих документов, а также быстро изменять оформление документов за счёт изменения этого стиля, хранящегося в отдельном CSS-файле. Перенос правил представления данных в отдельный файл ведёт к уменьшению времени загрузки страниц сайта — описание представления данных загружается браузером только один раз, а далее, при переходе с одной страницы сайта на другую, браузер загружает только структуру страницы и хранимые на ней данные. Можно

предусмотреть несколько дизайнов страницы, применяемых в зависимости от характеристик устройства (размера и разрешения экрана), используемого для просмотра.

Веб-сайты и веб-страницы хранятся на так называемых веб-серверах — компьютерах, на которых установлено специальное программное обеспечение, обладающее соответствующим функционалом. Программа, позволяющая хранить и пересылать веб-страницы, также называется веб-сервером. Пользователи, имеющие доступ к сети, просматривают веб-документы при помощи программ-клиентов — веб-браузеров (рис. 4.1).

Взаимодействие клиент–сервер происходит по протоколу HTTP (англ. HyperText Transfer Protocol — протокол передачи гипертекста).

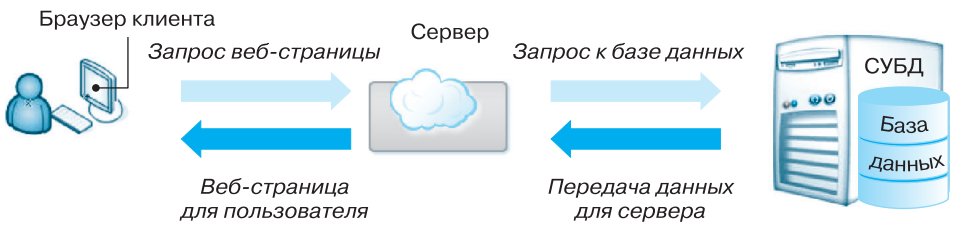


Рис. 4.1. Схема обработки запросов пользователя

Активные компоненты веб-страниц содержат программный код, позволяющий выполнять действия в соответствии с заложенной в них программой. Для написания таких программ используются языки веб-программирования. Один из них — JavaScript, позволяющий описывать правила, определяющие реагирование веб-страницы на действия пользователя. Эти правила представляются в форме веб-сценариев (скриптов) — программ, записываемых непосредственно в html-код страниц или в отдельный файл.

Что касается языков веб-программирования в целом, то их можно разделить на две пересекающиеся группы: клиентские и серверные. Программы на клиентских языках выполняются на стороне пользователя, и результат их выполнения зависит от используемого браузера. Серверные языки программирования, соответственно, выполняются на стороне сервера. Получая запрос с адресом веб-документа от браузера, серверные программы связываются с базой данных. БД передаёт информацию о веб-странице скриптам сервера, и те, обработав её, отсылают для интерпретации браузеру клиента, который и выводит результат совместной работы на монитор.

16.2. Поиск информации в сети Интернет

Поиск нужного документа во Всемирной паутине может происходить разными способами:

- указанием адреса документа;
- путём использования поисковых систем.

Поисковая система — это программно-аппаратный комплекс, предназначенный для поиска информации во Всемирной паутине.
Поисковая машина — программная часть поисковой системы; комплекс программ, предназначенный для поиска информации.

Поисковые системы располагаются на специально выделенных компьютерах с мощными каналами связи. Ежеминутно они обслуживают огромное количество поисковых запросов клиентов.

По принципу действия различают несколько типов поисковых систем, а именно:

- поисковые каталоги, управляемые человеком;
- системы, использующие поисковых роботов;
- гибридные поисковые системы.

Поисковые каталоги (веб-каталоги или тематические каталоги) содержат базу данных ссылок на веб-сайты, распределённых по отдельным тематическим рубрикам. Такие каталоги заполняются специалистами вручную. Поиск в них осуществляется спуском по дереву каталога:

- определив тему, по которой будет выполняться поиск, пользователь выбирает соответствующую рубрику тематического каталога;
- прочитав описания ссылок на открывшейся странице, пользователь может перейти по ссылке, соответствующей его ожиданиям; если же нужных ссылок не обнаруживается, то можно уточнить тему и повторить поиск в этой же или другой поисковой службе.

В 1994 году Дэвид Фило и Джерри Янг из Стэнфордского университета (США) предприняли попытку упорядочить большое количество накопившихся у них ссылок на разнообразные информационные источники. Так появилась идея использования специализированной базы данных для эффективного поиска информации в сети. Очень скоро созданная ими система Yahoo! стала самым популярным и полным иерархическим предметно-ориентированным путеводителем по Интернету. В наши дни — это одна из наиболее известных поисковых систем.

Действие поисковых систем, использующих поисковых роботов, основано на постоянном, последовательном изучении всех страниц всех сайтов Всемирной паутины. Для каждого документа составляется его поисковый образ — набор ключевых слов, отражающих содержание этого документа. В связи с постоянным обновлением информации поисковые системы периодически возвращаются к ранее изученным страницам, чтобы обнаружить и зарегистрировать изменения. Информация о ключевых словах исследованных таким образом страниц сохраняется в поисковой системе.

При поступлении запроса от пользователя поисковая система на основании имеющейся в ней информации формирует список страниц, соответствующих критериям поиска. Найденные документы, как правило, упорядочиваются в зависимости от местоположения ключевых слов (в заголовке, в начале текста), частоты их появления в тексте и других характеристик.

Различные поисковые системы, использующие поисковых роботов, имеют схожую структуру, включающую:

1) **модуль индексирования**, состоящий из трёх программ-роботов (Spider или «паук» — скачивает веб-страницы; Crawler или «путешествующий паук» — переходит по всем ссылкам, имеющимся на странице, и ищет новые документы, ещё не известные поисковой системе; Indexer или «робот-индексатор» — разбивает на фрагменты страницы, которые скачали «пауки», анализирует их и составляет некоторое описание этих страниц);
2) **базу данных** — хранилище представленных в определённом формате всех скачанных и обработанных модулем индексирования документов;
3) **поисковый сервер** — система выдачи результатов поиска, определяющая, какие страницы и в какой степени удовлетворяют запросу пользователя.

Поисковая система, получив запрос на поиск, анализирует ту информацию, которая была ею проиндексирована. С одной стороны, это позволяет существенно повысить скорость обработки поискового запроса. С другой стороны, результаты поиска нельзя считать полными, т. к. ни одна поисковая система не может загрузить в свою базу данных информацию обо всех без исключения ресурсах. Кроме того, результаты поиска могут быть отчасти устаревшими — ситуация в сети Интернет меняется быстрее, чем происходит обновление сведений в базах данных поисковых систем.

Гибридные поисковые системы сочетают в себе функции систем, управляемых человеком, и систем, использующих поисковых роботов.

Несмотря на общие принципы работы, поисковые системы различаются по таким характеристикам, как: язык запроса, зона поиска, глубина поиска внутри документа, метод упорядочивания информации и др. На данный момент самой популярной в мире поисковой системой является Google, а крупнейшей отечественной поисковой системой — Яндекс.

В большинстве поисковых систем реализовано три основных типа поиска:

1) поиск по любому из слов — результатом является огромный список всех страниц, содержащих хотя бы одно из ключевых слов; применяется, когда пользователь не уверен в ключевых словах;
2) поиск по всем словам — в этом режиме формируется список всех страниц, содержащих все ключевые слова в любом порядке;
3) поиск точно по фразе — в результате составляется список всех страниц, содержащих фразу, точно совпадающую с ключевой (знаки препинания игнорируются).

Чтобы поиск стал более продуктивным, во всех поисковых системах предусмотрены специальные языки формирования запросов со своим синтаксисом. Эти языки во многом похожи. Выяснить особенности определённого языка можно с помощью справочной системы, входящей в состав поисковой машины.

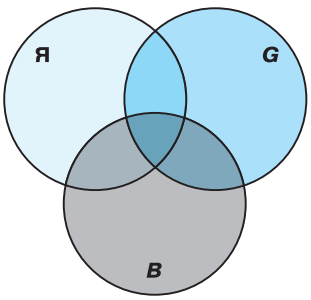
Найдите информацию о правилах формирования поисковых запросов в поисковых системах Яндекс и Google. Сравните их между собой. Укажите общее и различия.

Пример. В языке запросов некой поисковой машины для обозначения логической операции ИЛИ используется символ «|», а для логической операции И — символ «&». Известны запросы и количество найденных по ним страниц некоторого сегмента сети Интернет:

Запрос	Найдено страниц (тыс.)
Яндекс & Google	145
Bing & Google	580
Яндекс & Bing & Google	85
(Яндекс | Bing) & Google	x

Чему равно x, если все запросы выполнялись практически одновременно, т. е. во время выполнения запросов набор страниц, содержащих все искомые слова, не изменился?

С подобными задачами вы встречались в курсе информатики основной школы и знаете, что их условие может быть представлено с помощью кругов Эйлера и записано на языке теории множеств.



Пусть Я, G и B — множества страниц, содержащих слова Яндекс, Google и Bing соответственно.

Тогда $Я \cap G$ — множество страниц, соответствующих запросу «Яндекс & Google», и его мощность равна 145 (тыс.): $|Я \cap G| = 145$.

Рассуждая аналогично, можем записать: $|B \cap G| = 580$, $|Я \cap B \cap G| = 85$.

Требуется найти мощность множества $(Я \cup B) \cap G$.

Преобразуем это выражение:

$$(Я \cup B) \cap G = (Я \cap G) \cup (B \cap G).$$

В справедливости такого преобразования вы можете убедиться, изобразив левую и правую части равенства с помощью кругов Эйлера.

Согласно принципу включений-исключений, для двух множеств имеем: $|X \cup Y| = |X| + |Y| - |X \cap Y|$.

В нашем случае получаем:

$$|(Я \cap G) \cup (B \cap G)| = |Я \cap G| + |B \cap G| - |(Я \cap G) \cap (B \cap G)| =$$
$$= |Я \cap G| + |B \cap G| - |Я \cap G \cap B| = 145 + 580 - 85 = 640.$$

Итак, $x = 640$.

Как бы точно ни был составлен пользователем запрос к информационной системе, только малая часть из полученных по этому запросу документов будет релевантной, т. е. соответствующей зафиксированной в запросе информационной потребности.

Полнота поиска — это отношение числа выданных релевантных документов к общему числу релевантных документов, имеющихся в базе данных поисковой системы. В идеале это число должно равняться 1; на практике может достигать значений 0,7–0,9.

Точность поиска — это отношение числа выданных релевантных документов к общему числу документов, выданных системой по данному запросу. Значение этого параметра колеблется от 0,1 до 1.

Полнота и точность определяют качество или эффективность поиска.

16.3. О достоверности информации, представленной на веб-ресурсах

Поиск информации в сети Интернет удобен и прост, а самое главное — он занимает гораздо меньше времени, чем поход в библиотеку или работа с другими источниками информации. Благодаря этому в наши дни Интернет является самым популярным источником информации: им пользуются не только рядовые граждане, но и учёные, бизнесмены, государственные служащие, специалисты разных областей и сфер деятельности для решения самого широкого круга профессиональных задач. И это при том, что Интернет является зоной свободного доступа, в которой абсолютно каждый может не только искать ту или иную информацию, но и размещать в ней всё, что сочтёт возможным. Эти данные никем не контролируются и не проверяются, а поэтому они могут быть недостоверными (содержать устаревшие данные, ошибочные или заведомо ложные утверждения) и субъективными (отражать личную точку зрения автора).

К данным, которые вы получили в результате поиска в Интернете, следует относиться критически и предпринимать шаги для того, чтобы убедиться в достоверности информации.

Рассмотрим некоторые способы проверки информации, полученной в результате поиска в сети Интернет.

1. **Выяснение репутации сайта**, на котором размещена информация, представляющая для вас интерес. Проверенные данные публикуют официальные сайты государственных, коммерческих, научных и других структур, являющиеся первоисточниками информации. Ответственность за любую опубликованную ими информацию несут ресурсы, имеющие свидетельство о регистрации средства массовой информации. Избегают недостоверной информации известные ресурсы, занимающие высокие места в соответствующих рейтингах. Представление о репутации сайта можно получить в том числе и по имеющимся в сети отзывам об этом ресурсе.

Если веб-сайт не обладает широкой известностью, то следует обратить внимание на следующие моменты:

- указано ли, для кого предназначен ресурс и какова цель его создания;
- насколько регулярно обновляются данные на веб-сайте; не устарела ли информация (узнать дату размещения материалов);
- не требуют ли разработчики веб-страницы ввода ваших личных данных.

2. **Получение информации об авторе** представляющего интерес материала. Следует убедиться, что на веб-странице приведены данные об авторе, в том числе описание его квалификации и контактная информация. Можно попытаться найти и ознакомиться с другими работами этого автора, комментариями и отзывами читателей на его работы.
3. **Проверка фактического материала**. Любые фактические и статистические данные имеют источник. Хорошо, если ссылки на авторитетные источники имеются на страницах заинтересовавшего вас сайта. Если таких ссылок нет, то данные можно выборочно сверить с официальными источниками самостоятельно. Если обнаружится, что какие-то данные не согласуются с данными официальных источников, то и остальному материалу также не стоит доверять. Хорошо, если данные подаются с разных точек зрения, если они согласуются с тем, что вы изучали в школе или узнали из других источников.

Необходимо чётко представлять себе, что именно вы ищете, и проверять все важные данные, найденные в Интернете. Если в результате поиска вы не нашли ни одного подходящего документа, нужно:

- проверить правильность написания ключевых слов;
- проверить правильность использования логических связок;
- подобрать более удачные синонимы;
- изменить логику запроса.

САМОЕ ГЛАВНОЕ

Веб-страница с точки зрения её разработчика — это файл, содержащий собственно текст, несущий определённую информацию для пользователя, и служебную информацию для браузера (тэги разметки) на языке HTML (англ. HyperText Markup Language — язык разметки гипертекста).

HTML — один из веб-стандартов, по которым разрабатываются сайты во всём мире. Ещё одним из таких стандартов является технология CSS (англ. Cascading Style Sheets — каскадные таблицы стилей) — формальный язык описания внешнего вида документа, составленного с использованием языка разметки. Технология CSS позволяет принципиально разделить содержание и представление документа: описание содержания и логической структуры веб-страницы производится с помощью HTML или других языков разметки, а описание внешнего вида веб-страницы производится с помощью CSS.

Веб-страницы предназначены для воспроизведения на самых разных экранах самых разных компьютеров. Поэтому они не имеют «жёсткого» форматирования. Оформление веб-страницы выполняется непосредственно во время её воспроизведения на компьютере клиента в соответствии с настройками используемого браузера.

Поисковая система — это программно-аппаратный комплекс, предназначенный для поиска информации во Всемирной паутине. Поисковая машина — программная часть поисковой системы; комплекс программ, предназначенный для поиска информации.

По принципу действия различают такие типы поисковых систем, как: поисковые каталоги, управляемые человеком; системы, использующие поисковых роботов; гибридные поисковые системы.

Необходимо чётко представлять себе, что именно вы ищете, и проверять все важные данные, найденные в Интернете. Основными способами проверки найденной информации являются: выяснение репутации сайта; получение информации об авторе материала; проверка фактического материала по данным из авторитетных источников.

Вопросы и задания

1. Что представляет собой веб-страница с точки зрения пользователя и с точки зрения её разработчика?
2. В чём, по вашему мнению, состоит одно из основных отличий веб-страницы от обычного текстового документа?
3. Кому принадлежит идея гипертекста? Подготовьте краткое сообщение на эту тему.
4. Назовите два основных веб-стандарта. Для чего предназначен каждый из них?
5. Какие способы поиска документа во Всемирной паутине вам известны?

6. Что такое поисковая система? Что такое поисковая машина?
7. Какие типы поисковых систем можно выделить в зависимости от принципа их действия?
8. Почему, если мы не можем найти что-либо в одной поисковой системе, имеет смысл поискать нужный материал в другой?
9. Может ли случиться так, что поисковая система найдёт документ, который не существует?
10. В таблице приведены запросы к поисковому серверу. Для обозначения логической операции ИЛИ в запросе используется символ «|», а для логической операции И — «&». Расположите номера запросов в порядке возрастания количества страниц, которые найдёт поисковый сервер по каждому запросу.

1	принтер | сканер | монитор
2	монитор & принтер
3	принтер & сканер & монитор
4	принтер & сканер & монитор & колонки
5	принтер | сканер
6	принтер | сканер | монитор | колонки
7	(монитор | принтер) & (принтер | сканер)
8	(монитор | сканер) & принтер

11. Известны запросы и количество найденных по ним страниц некоторого сегмента сети Интернет:

Запрос	**Найдено страниц (тыс.)**			
	1	**2**	**3**	**4**
Яндекс | Google	900	1300	750	*x*
Bing | Google	700	1400	*x*	2000
Яндекс | Bing | Google	1200	*x*	450	2500
(Яндекс & Bing) | Google	*x*	1600	1100	500

Чему равно x, если все запросы выполнялись практически одновременно, т. е. во время выполнения запросов набор страниц, содержащих все искомые слова, не изменился?

12. Зная, что такое точность информации, дайте определение парному понятию «информационный шум».
13. Зная, что такое полнота информации, дайте определение парному понятию «потери информации».
14. Какая информация называется релевантной? Как связаны полнота и точность с качеством (эффективностью) поиска?
15. Можно ли безоговорочно доверять такому популярному ресурсу, как Википедия?
16. Кто такие блогеры? Можно ли безоговорочно доверять публикуемой ими информации?
17. В чём суть основных способов проверки достоверности информации, найденной в сети Интернет?
18. Найдите в сети Интернет не менее трёх авторитетных источников, содержащих информацию по одной из следующих тем:
    - «Системы искусственного интеллекта и машинное обучение»;
    - «Принципы построения и редактирования трёхмерных моделей»;
    - «Представление о системах автоматизированного проектирования».

    Почему вы считаете, что этим источникам можно доверять?

    На основе найденных материалов подготовьте небольшое сообщение по выбранной теме.

Дополнительные материалы к главе смотрите в авторской мастерской.

Глава 5
ОСНОВЫ СОЦИАЛЬНОЙ ИНФОРМАТИКИ

Информатика изучает информационные процессы, протекающие в системах самой разной природы: технических, биологических, социальных. В последние десятилетия всё большее значение приобретают информационные процессы, характеризующиеся широкомасштабным применением информационно-коммуникационных технологий (ИКТ) во всех сферах социально-экономической, политической и культурной жизни общества. Широкое и повсеместное применение ИКТ оказывает всё более сильное влияние на жизнь каждого отдельного человека и общества в целом.

Социальная информатика — это наука, изучающая комплекс проблем, связанных с информационными процессами в обществе (социуме).

Мы рассмотрим ключевые понятия социальной информатики — информационное общество, информационные ресурсы, информационное право, а также тесно связанное с ними понятие информационной безопасности.

§ 17
Информационное общество

17.1. Понятие информационного общества

Из курсов истории и обществознания вам известно, что человеческая цивилизация в своём развитии прошла через несколько социально-экономических стадий (рис. 5.1).

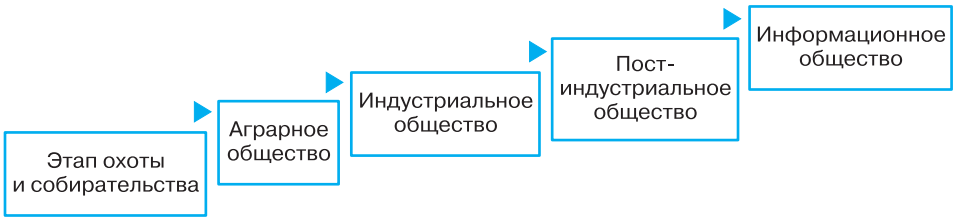


Рис. 5.1. Социально-экономические стадии развития общества

Одним из критериев, определяющим стадию общественного развития, является характер трудовой деятельности населения. Так, на этапе аграрного общества большая часть населения занята в сельском хозяйстве; в индустриальном обществе более половины населения занято в сфере промышленного производства; постиндустриальная стадия развития общества характеризуется тем, что более 50% населения занято в сфере услуг. О переходе на стадию информационного общества можно говорить тогда, когда более половины населения задействовано в сфере информационно-интеллектуального производства и услуг.

Информационное общество — новая историческая фаза развития цивилизации, в которой главными продуктами производства являются информация и знания.

В 2000 г. в Окинаве восемь крупнейших промышленно развитых стран мира (Великобритания, Германия, Италия, Канада, Россия, Соединённые Штаты Америки, Франция, Япония) приняли Хартию глобального информационного общества, в которой определены приоритетные направления деятельности, способствующие вхождению государств в информационное общество.

Так, в Хартии подчёркивается, что:

- «все люди повсеместно, без исключения должны иметь возможность пользоваться преимуществами глобального информационного общества»;
- «информационное общество, как мы его представляем, позволяет людям шире использовать свой потенциал и реализовывать свои устремления. Для этого мы должны сделать так, чтобы информационные технологии служили достижению взаимодополняющих целей обеспечения устойчивого экономического роста, повышения общественного благосостояния, стимулирования социального согласия и полной реализации их потенциала в

области укрепления демократии, транспарентного[1)] и ответственного управления, международного мира и стабильности».

Ознакомьтесь с Окинавской Хартией (**www.iis.ru/library/okinawa/charter.ru.html**). Назовите изложенные в ней приоритетные направления деятельности, способствующие процессу развития информационного общества.

В рамках Всемирной встречи (Всемирного саммита) на высшем уровне по вопросам информационного общества (Женева, 2003 г. — Тунис, 2005 г.) были сформулированы принципы построения информационного общества.

Ознакомьтесь с Декларацией принципов построения информационного общества (**www.un.org/ru/events/pastevents/pdf/dec_wsis.pdf**). Подготовьте презентацию, раскрывающую суть каждого из изложенных в ней принципов.

Можно выделить следующие основные черты информационного общества:

- увеличение роли информации и знаний в жизни общества;
- возрастание числа людей, занятых в сфере информационных и коммуникационных технологий, рост доли информационных продуктов и услуг в валовом внутреннем продукте;
- широкомасштабное использование ИКТ во всех сферах социально-экономической, политической и культурной жизни общества;
- создание глобального информационного пространства, обеспечивающего: эффективное информационное взаимодействие людей; их доступ к мировым информационным ресурсам; удовлетворение их потребностей в информационных продуктах и услугах;
- развитие информационной экономики, электронного правительства, электронных социальных сетей и др.

И развитые, и развивающиеся страны в разной степени продвинулись по пути к информационному обществу. Принято считать, что США завершат в целом переход к информационному обществу к 2020 году, Япония и большинство стран Западной Европы — к 2030–2040 гг.

17.2. Информационные ресурсы, продукты и услуги

Человеческое общество по мере своего развития овладевало не только веществом и энергией, но и информацией. С течением

1) Транспаре́нтный — прозрачный, открытый, не содержащий недомолвок и секретов.

времени накапливались информационные ресурсы, являющиеся сегодня стратегическими, аналогичными по значимости ресурсам материальным, сырьевым, энергетическим, трудовым и финансовым. Всё возрастающая роль информационных ресурсов — одна из важнейших тенденций общественного развития, сопровождающая переход к информационному обществу.

Совокупность всей информации, накопленной человечеством в процессе развития науки, культуры, образования и практической деятельности людей, называют **информационными ресурсами**.

Информация, содержащаяся в государственных информационных системах, а также иные имеющиеся в распоряжении государственных органов сведения и документы являются государственными информационными ресурсами[1)].

Государственные информационные ресурсы используются для решения задач государственного управления, обеспечения прав и безопасности граждан, поддержки социально-экономического развития страны, развития культуры, науки, образования и т. д. Значительная часть государственных информационных ресурсов сконцентрирована в федеральных государственных информационных системах, реестр которых размещен на сайте **rkn.gov.ru/it/register**.

Многие государственные информационные системы ориентированы на внешнего пользователя. К таким системам можно отнести:

- библиотечную сеть Российской Федерации;
- архивный фонд Российской Федерации;
- государственную систему статистики;
- государственную систему научно-технической информации;
- государственную систему правовой информации;
- систему информационных ресурсов органов государственной власти и др.

Рассмотрим более подробно первые две из них.

Библиотечная сеть РФ включает свыше 150 тыс. библиотек федерального, регионального и муниципального уровней. В библиотеках хранится информация, представленная в виде различных отечественных и зарубежных изданий, охватывающая все направления, темы и отрасли знаний. Это сотни миллионов единиц хранения. В последние годы в библиотеках активно создаются

1) Федеральный закон «Об информации, информационных технологиях и о защите информации».

электронные каталоги, другие библиографические и реферативные базы данных, организуется доступ читателей к электронным изданиям и ресурсам сети Интернет.

Архивный фонд Российской Федерации — это исторически сложившаяся и постоянно пополняющаяся совокупность подлежащих постоянному хранению документов. Объём Архивного фонда Российской Федерации составляет более 500 млн. единиц хранения, представленных на разных носителях.

Оцифровка архивных документов и создание соответствующих информационных систем обеспечивают доступ к ним широкого круга пользователей. Например, оцифровка документов о Великой Отечественной войне, хранящихся в Центральном архиве Минобороны РФ, Центральном военно-морском архиве Минобороны РФ, Российском государственном военном архиве, Государственном архиве РФ и его региональных отделениях, Управлении Минобороны РФ по увековечению памяти погибших при защите Отечества, позволила собрать в единую информационную систему донесения боевых частей о безвозвратных потерях, другие архивные документы, уточняющие потери (похоронки, документы госпиталей и медсанбатов, трофейные карточки советских военнопленных и т. д.), а также паспорта захоронений советских солдат и офицеров. В рамках проекта отсканировано более 16,8 млн листов архивных документов и свыше 45 тыс. паспортов воинских захоронений. В результате был создан Обобщённый банк данных «Мемориал» **(www.obd-memorial.ru)**. Вы можете ознакомиться с реальными документами, самостоятельно провести поиск пропавших без вести или уточнить информацию о погибших в годы войны (рис. 5.2).

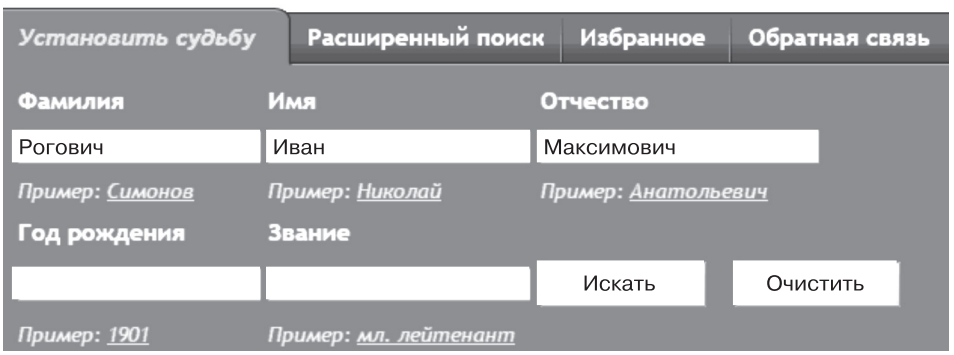


Рис. 5.2. Поиск в Обобщённом банке данных «Мемориал»

Перенесённые на электронные носители информационные ресурсы приобретают качественно новое состояние, позволяющее создавать на их основе разнообразные информационные продукты.

Информационный продукт — информация всех видов, программные продукты, базы данных, представленные в форме товара, т. е. созданные с целью продажи за деньги или обмена на другие продукты.

Информационные продукты отличаются от других продуктов тем, что:

1) сохраняют содержащуюся в них информацию, независимо от того, сколько раз она была использована;
2) со временем могут потерять свою ценность, например, в связи с потерей актуальности содержащейся в них информации;
3) они могут быть представлены в разной форме, с учётом возможностей или предпочтений потребителей;
4) требуют значительных затрат на производство и незначительных по сравнению с ними затрат на тиражирование.

Информационная услуга — действия, направленные на удовлетворение информационной потребности пользователя путём предоставления информационного продукта.

Характерная черта информационного общества — появление развитого рынка информационных ресурсов и услуг (рис. 5.3).

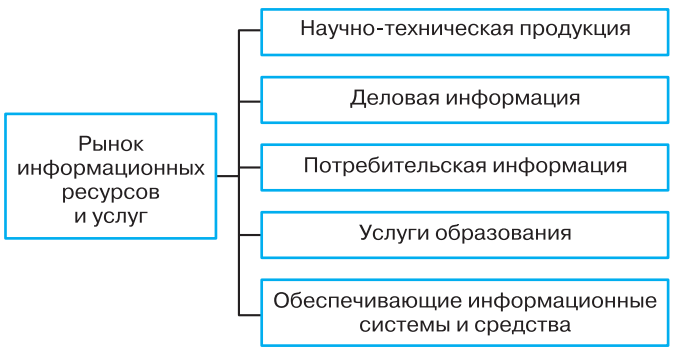


Рис. 5.3. Структура рынка информационных ресурсов и услуг

Соотнесите следующие информационные ресурсы и услуги с секторами информационного рынка:

- системное ПО;
- прикладное ПО;
- электронные учебники;
- развивающие компьютерные игры;
- компьютерные обучающие и контролирующие системы;
- биржевая и финансовая информация;
- статистическая информация;
- коммерческая информация;
- профессиональная информация;
- научно-техническая информация;
- новостная информация;
- всевозможные расписания;
- резервирование билетов и мест в гостиницах;
- заказ товаров и услуг;
- проведение банковских операций;
- произведения художественной литературы;
- кинофильмы;
- музыка;
- игры.

17.3. Информатизация образования

По мере продвижения к информационному обществу всё большие возможности, связанные с использованием информационных и коммуникационных технологий, появляются в сфере образования, способствуя повышению его доступности и качества, созданию системы непрерывного образования.

Ключевым условием построения современного образовательного процесса является наличие в образовательном учреждении информационно-образовательной среды — «системы инструментальных средств и ресурсов, обеспечивающих условия для реализации образовательной деятельности на основе информационно-коммуникационных технологий»[1)].

Охарактеризуйте информационно-образовательную среду вашей школы: опишите имеющееся техническое оснащение, программное обеспечение и их использование учителями и школьниками.

Сегодня мы говорим об электронном обучении — «организации образовательной деятельности с применением содержащейся в базах данных и используемой при реализации образовательных программ информации и обеспечивающих её обработку информационных технологий, технических средств, а также информационно-телекоммуникационных сетей, обеспечивающих передачу

1) ГОСТ Р 53620–2009 «Информационно-коммуникационные технологии в образовании. Электронные образовательные ресурсы. Общие положения».

по линиям связи указанной информации, взаимодействие обучающихся и педагогических работников»[1)].

Информационные образовательные ресурсы, представленные в электронной форме, принципиально отличаются от печатных учебных материалов, обеспечивая:

1) моделирование и визуализацию информации об изучаемых объектах;
2) интерактивное взаимодействие пользователя и средства ИКТ;
3) хранение больших объёмов информации с возможностью лёгкого доступа к ним;
4) автоматизацию процессов вычислительной, информационно-поисковой деятельности, а также обработки результатов учебного эксперимента с возможностью многократного повторения фрагмента или самого эксперимента;
5) автоматизацию организационного управления учебной деятельностью и контроля результатов усвоения;
6) информационное взаимодействие между участниками образовательного процесса с помощью локальных и глобальной компьютерных сетей.

Был ли у вас опыт работы с электронными образовательными ресурсами, обладающими одной или несколькими из перечисленных выше возможностей? Приведите примеры.

В настоящее время в сети Интернет имеется множество открытых образовательных ресурсов (ООР) — разнообразных материалов, предоставляющих доступ к знаниям (полные курсы, учебные материалы, модули, учебники, видео, тексты, программное обеспечение и т. д.), размещённых в свободном доступе, либо выпущенных под лицензией, разрешающей их свободное использование или переработку.

Каждый желающий может воспользоваться порталом «Универсариум» — открытой площадкой дистанционного образования от лучших преподавателей ведущих вузов страны (**universarium.org**). «Универсариум» предоставляет средство получения качественного и доступного образования всем желающим, обеспечивает возможность профориентационного, дополнительного и профессионального образования для граждан Российской Федерации, проживающих на удалённых и труднодоступных территориях, делает доступным образование для граждан с ограниченными возможностями здоровья и выполняет ряд других социальных миссий.

1) Федеральный закон «Об образовании в Российской Федерации».

17.4. Россия на пути к информационному обществу

Исследования в области информационных проблем развития современного общества ведутся сегодня во всём мире. Их основой являются ноосферная концепция эволюции биосферы, созданная трудами таких всемирно известных российских учёных, как К. Э. Циолковский, В. И. Вернадский, А. Л. Чижевский и др. Эта концепция получила дальнейшее развитие в трудах Н. Н. Моисеева, А. Д. Урсула, А. И. Ракитова и др.

Рассмотрим ключевые события, определяющие развитие информационного общества в России (рис. 5.4).

Рис. 5.4. Развитие информационного общества в России

Россия была одной из стран, подписавших в 2000 г. Хартию глобального информационного общества.

Следующим значительным шагом на пути к созданию информационного общества можно считать принятую в 2002 году Федеральную целевую программу «Электронная Россия (2002–2010 годы)», этапы и ход реализации которой представлены на сайте Министерства связи и массовых коммуникаций Российской Федерации. Там же подведены итоги реализации этой программы и зафиксированы её основные результаты:

- повышение качества взаимоотношений государства и общества путём расширения возможности доступа граждан к информации о деятельности органов государственной власти, повышения оперативности предоставления государственных и муници-

пальных услуг, внедрения единых стандартов обслуживания населения;

- повышение эффективности межведомственного взаимодействия и внутренней организации деятельности органов государственной власти на основе организации межведомственного информационного обмена и обеспечения эффективного использования органами государственной власти информационных и телекоммуникационных технологий, повышения эффективности управления внедрением информационных и телекоммуникационных технологий в деятельность органов государственной власти;
- повышение эффективности государственного управления, обеспечение оперативности и полноты контроля за деятельностью органов государственной власти.

Одним из зримых результатов реализации программы «Электронная Россия» является разработка федеральной государственной информационной системы «Единый портал государственных и муниципальных услуг (функций)» (рис. 5.5).

Перейдите на портал **www.gosuslugi.ru** и выясните, какие категории госуслуг могут быть получены гражданами.

В 2008 году была принята Стратегия развития информационного общества в Российской Федерации. Она определила цель формирования и развития информационного общества в РФ — «повышение качества жизни граждан, обеспечение конкурентоспособности России, развитие экономической, социально-политической, культурной и духовной сфер жизни общества,

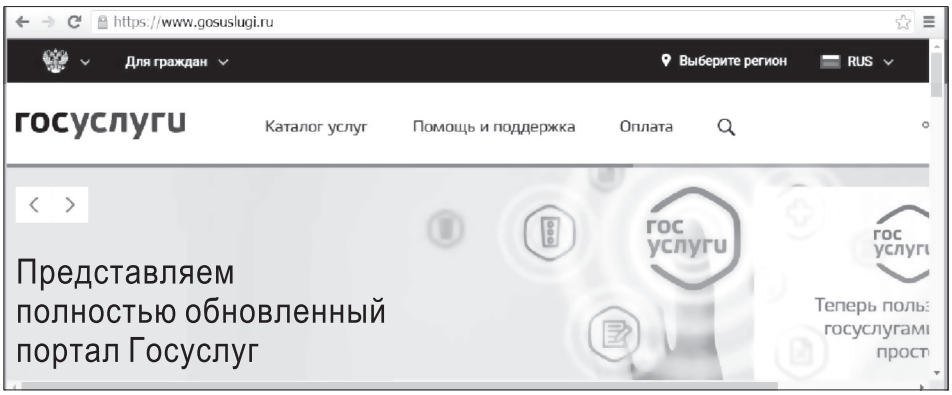


Рис. 5.5. Единый портал государственных услуг

совершенствование системы государственного управления на основе использования информационных и телекоммуникационных технологий».

Ознакомьтесь с документом «Стратегия развития информационного общества в Российской Федерации» (**rg.ru/2008/02/16/informacia-strategia-dok.html**). Каковы основные задачи и принципы развития информационного общества в нашей стране?

Докажите, что в Стратегии учтены основные положения Окинавской Хартии глобального информационного общества и Декларации принципов построения информационного общества.

В настоящее время в нашей стране действует государственная программа «Информационное общество (2011–2020)». Цель этой программы – получение гражданами и организациями преимуществ от применения информационных и телекоммуникационных технологий, создание на основе их использования условий для оперативного и эффективного взаимодействия государства с гражданами и бизнесом.

Программа включает следующие направления:

- построение электронного правительства и повышение эффективности государственного управления;
- повышение качества жизни граждан и улучшение условий развития бизнеса;
- развитие российского рынка информационных и коммуникационных технологий, обеспечение перехода к цифровой экономике;
- обеспечение безопасности в информационном обществе;
- развитие цифрового контента и сохранение культурного наследия.

Выясните, какие именно проекты (мероприятия) предусмотрены в рамках программы «Информационное общество». Укажите 3–4 проекта, являющихся, на ваш взгляд, наиболее интересными. (**www.tadviser.ru/index.php/Статья:Информационное_общество_ГП**)

Ожидается, что в результате реализации программы «Информационное общество» к 2020 г. вырастет индекс Российской Федерации в международном рейтинге стран по уровню развития информационных и телекоммуникационных технологий и до 85% увеличится доля россиян, пользующихся электронными госуслугами в повседневной жизни (в 2010 г. этот показатель составлял 11%).

САМОЕ ГЛАВНОЕ

Информационное общество — новая историческая фаза развития цивилизации, в которой главными продуктами производства являются информация и знания. Это общество характеризуется: широкомасштабным использованием информационно-коммуникационных технологий (ИКТ); ростом доли информационных продуктов и услуг в валовом внутреннем продукте; созданием глобальной информационной инфраструктуры, обеспечивающей информационное взаимодействие людей, их доступ к информации и удовлетворение их социальных и личностных потребностей в информационных продуктах и услугах.

Совокупность всей информации, накопленной человечеством в процессе развития науки, культуры, образования и практической деятельности людей, называют информационными ресурсами. Государственные информационные ресурсы используются для решения задач государственного управления, обеспечения прав и безопасности граждан, поддержки социально-экономического развития страны, развития культуры, науки, образования и т. д.

Информационный продукт — информация всех видов, программные продукты, базы данных, представленные в форме товара, т. е. созданные с целью продажи за деньги или обмена на другие продукты. Информационная услуга — действия, направленные на удовлетворение информационной потребности пользователя путём предоставления информационного продукта.

По мере продвижения к информационному обществу всё большие возможности, связанные с использованием информационных и коммуникационных технологий, появляются в сфере образования, способствуя повышению его доступности и качества, созданию системы непрерывного образования.

Основными документами, определяющими путь России к информационному обществу, являются «Стратегия развития информационного общества в Российской Федерации» и государственная программа «Информационное общество (2011–2020)». Ожидается, что в результате реализации этой программы к 2020 г. существенно вырастет индекс Российской Федерации в международном рейтинге стран по уровню развития информационных и телекоммуникационных технологий и до 85% увеличится доля россиян, пользующихся электронными госуслугами в повседневной жизни.

Вопросы и задания

1. Какие социально-экономические стадии прошло человечество в своём развитии? Какой критерий может быть использован для определения стадии общественного развития?
2. Что такое информационное общество? Назовите его основные черты.
3. Работая в группе, вспомните и дайте краткую характеристику основных этапов информационного развития общества. Подготовьте презентацию, иллюстрирующую эти этапы.
4. Поясните смысл термина «ресурс». Какие бывают ресурсы?
5. Что называют информационными ресурсами?
6. Что представляют собой государственные информационные ресурсы? Выясните, что такое информационный кризис. Используйте дополнительные источники информации.
7. Выясните, что такое компьютерная зависимость и каковы её основные симптомы. Используйте дополнительные источники информации.
8. Выясните, что такое информационное неравенство. Используйте дополнительные источники информации.
9. Что такое информационный продукт? Чем информационные продукты отличаются от других продуктов?
10. Что такое информационная услуга? Приведите примеры. Пользовались ли вы информационными услугами лично?
11. Опишите структуру рынка информационных ресурсов и услуг.
12. На рынке информационных продуктов и услуг, как и на любом другом рынке, есть поставщики (продавцы) и потребители (покупатели). Кто может быть поставщиком информационных продуктов и услуг? Кто может быть покупателем информационных продуктов и услуг? Приведите примеры.
13. Подготовьте и проиллюстрируйте мультимедийными материалами сообщение на одну из следующих тем.
    1) Мир ИТ-профессий.
    2) «Поход» за покупками в онлайн-магазин.
    3) «Умный дом» — будущее или реальность?

    Укажите адреса сайтов, где вы нашли информацию по выбранной вами теме. На основании чего вы считаете возможным доверять этой информации?

14. Знаете ли вы о том, что ещё 50 лет тому назад выпускнику среднего или высшего учебного заведения было достаточно полученных им знаний для успешной профессиональной деятельности практически на протяжении всей жизни? Согласны ли вы с тем, что в наше время это не так? Будьте готовы обосновать свою точку зрения.
15. Опишите идеальный электронный учебник, с которым было бы интересно работать современному школьнику.
16. Какие образовательные ресурсы называются открытыми?
17. Назовите ключевые события, определяющие развитие информационного общества в России.
18. Назовите основные направления государственной программы «Информационное общество (2011–2020)».
19. Выясните, что представляет собой индекс готовности регионов России к информационному обществу **(eregion.ru)**. Что учитывается при его расчёте? Назовите пять регионов-лидеров в рейтинге по готовности к информационному обществу. Какое место в этом рейтинге занимает ваш регион?
20. Проанализируйте адреса сайтов, имеющиеся в разделе 17.4 этого параграфа. Почему информацию на этих сайтах можно считать достоверной?

§ 18 Информационное право и информационная безопасность

18.1. Правовое регулирование в области информационных ресурсов

Информационные ресурсы, признанные в качестве одного из важнейших ресурсов страны, являются сегодня объектом особого внимания, контроля и управления со стороны государства. В нашей стране разработан ряд законов, обеспечивающих правовое регулирование в информационной сфере.

Согласно Гражданскому кодексу Российской Федерации (ГК РФ), собственнику принадлежат права владения, пользования и распоряжения своим имуществом[1)]. Собственнику инфор-

1) ГК РФ. Статья 209. Содержание права собственности.

мационного объекта, как и собственнику материального объекта, принадлежат:

- право распоряжения, состоящее в том, что только собственник информации имеет право определять, кому эта информация может быть предоставлена;
- право владения, обеспечивающее собственнику информации хранение информации в неизменном виде; никто, кроме владельца информации, не может её изменять;
- право пользования, предоставляющее собственнику информации право её использования в своих интересах.

Владелец информации — субъект, осуществляющий владение и пользование информацией и реализующий полномочия распоряжения в пределах прав, установленных законом и/или собственником информации.

Пользователь (потребитель) информации — субъект, пользующийся информацией, полученной от её собственника, владельца или посредника в соответствии с установленными правами и правилами доступа к информации либо с их нарушением.

Отношения, возникающие при осуществлении права на поиск, получение, передачу, производство и распространение информации, применении информационных технологий, обеспечении защиты информации регулируются Федеральным законом «Об информации, информационных технологиях и о защите информации». Рассмотрим некоторые статьи этого закона.

В статье 3 среди принципов правового регулирования отношений наряду со свободой поиска, получения, передачи, производства и распространения информации любым законным способом провозглашена неприкосновенность частной жизни, недопустимость сбора, хранения, использования и распространения информации о частной жизни лица без его согласия.

В статье 5 этого закона отмечается, что информация, в зависимости от порядка её предоставления или распространения, подразделяется на: 1) информацию, свободно распространяемую; 2) информацию, предоставляемую по соглашению лиц, участвующих в соответствующих отношениях; 3) информацию, которая в соответствии с федеральными законами подлежит предоставлению или распространению; 4) информацию, распространение которой в Российской Федерации ограничивается или запрещается.

В статье 8 утверждается право граждан на доступ без ограничений к: 1) нормативным правовым актам, затрагивающим права, свободы и обязанности человека и гражданина; 2) информации

о состоянии окружающей среды; 3) информации о деятельности государственных органов и органов местного самоуправления, а также об использовании бюджетных средств; 4) информации, накапливаемой в открытых фондах библиотек, музеев и архивов, а также в государственных, муниципальных и иных информационных системах, созданных или предназначенных для обеспечения граждан (физических лиц) и организаций такой информацией.

В статье 12 указывается, что государственное регулирование в сфере применения информационных технологий предусматривает развитие информационных систем различного назначения, а также создание условий для эффективного использования в Российской Федерации информационно-телекоммуникационных сетей, в том числе сети Интернет.

В статье 16 отмечается, что защита информации представляет собой принятие правовых, организационных и технических мер, направленных на: 1) обеспечение защиты информации от неправомерного доступа, уничтожения, модифицирования, блокирования, копирования, предоставления, распространения, а также от иных неправомерных действий в отношении такой информации; 2) соблюдение конфиденциальности информации ограниченного доступа; 3) реализацию права на доступ к информации.

В правовой информационной системе «КонсультантПлюс» найдите Федеральный закон № 149-ФЗ «Об информации, информационных технологиях и о защите информации». Используйте текст закона для ответов на следующие вопросы.

1. Сравните определения основных понятий, приведённые в статье 2 закона № 149-ФЗ с определениями этих понятий в учебнике информатики. Чем вы можете объяснить имеющиеся расхождения?
2. На каких принципах основывается правовое регулирование отношений, возникающих в сфере информации, информационных технологий и защиты информации?
3. Какие права и обязанности имеет обладатель информации?
4. Любая ли информация может быть доступна гражданам? С чем связаны ограничения на доступ граждан к информации?
5. Распространение какой информации запрещено законом?
6. С какой целью создаётся реестр российского программного обеспечения?
7. Где могут размещаться технические средства информационных систем, используемых государственными органами, органами местного самоуправления, государственными и муниципальными учреждениями?

8. Распространение какой информации в нашей стране запрещено?
9. Какой механизм ограничения доступа к сайтам в сети Интернет, содержащим запрещённую информацию, предусмотрен законом № 149-ФЗ?
10. Вы, являясь правообладателем некоторой информации, обнаружили, что она размещена в сети Интернет без вашего разрешения или иного законного основания. Какие действия вам следует предпринять?
11. Лица, права и законные интересы которых были нарушены в связи с разглашением информации ограниченного доступа, вправе обратиться в установленном порядке за судебной защитой своих прав. В каком случае их иск не будет удовлетворён?

Особого внимания заслуживают такие информационные объекты, как программы для электронных вычислительных машин (программы для ЭВМ) и базы данных. Наравне с произведениями науки, литературы или искусства, они считаются результатами интеллектуальной деятельности. Права на результаты интеллектуальной деятельности (авторское право, исключительное право, право собственности и др.) регулируются четвёртой частью ГК РФ[1)].

К сожалению, многие частные лица и даже организации нарушают закон, устанавливая на свои компьютеры программное обеспечение, полученное путём незаконного копирования. Такая практика затрудняет становление цивилизованного рынка информационных ресурсов и услуг.

18.2. Правовые нормы использования программного обеспечения

Использование программного обеспечения (ПО) является законным только тогда, когда на это есть согласие владельца авторских прав (компании-производителя или независимого разработчика). Основой правовых отношений между пользователем и собственником ПО является лицензия.

Лицензия (лицензионное соглашение) **на программное обеспечение** — это документ, определяющий порядок использования и распространения программного обеспечения, защищённого авторским правом.

1) ГК РФ. Часть четвёртая. Раздел VII. Права на результаты интеллектуальной деятельности и средства индивидуализации.

Существует множество разнообразных лицензий на программное обеспечение. Рассмотрим некоторые из них (рис. 5.6).

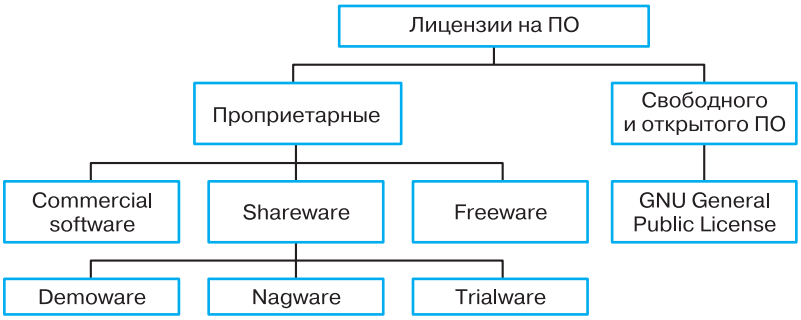


Рис. 5.6. Виды лицензий на использование ПО

Лицензии на программное обеспечение в целом можно разделить на две большие группы: проприетарные и лицензии свободного и открытого ПО.

В проприетарных лицензиях издатель ПО даёт разрешение её получателю использовать одну или несколько копий программы, но при этом сам остаётся правообладателем всех этих копий. Проприетарное программное обеспечение может быть коммерческим (Commercial software), условно-бесплатным (Shareware) и бесплатным (Freeware).

Коммерческое программное обеспечение создаётся с целью получения прибыли от его использования другими, например, путём продажи экземпляров.

Условно-бесплатное программное обеспечение представляет собой ограниченную по возможностям версию программы. Наиболее распространёнными видами ограничений являются:

- ограничение времени работы программы, количества её запусков, количества уровней в ней (в играх);
- отсутствие ряда функциональных возможностей;
- блокирование доступа к важным функциям, например сохранению файла;
- добавление дополнительной информации к сохраняемым файлам, например водяных знаков в программах редактирования изображения и видео.

В соответствии с лицензией, по окончании тестового периода необходимо приобрести или удалить программу. После покупки

программы пользователь получает код, снимающий ограничения, либо отдельную полноценную версию программы.

К условно-бесплатному ПО относятся:

- Demoware — демонстрационные версии коммерческого программного обеспечения, лицензионное соглашение которого предусматривает множество ограничений в функциональности по сравнению с основной версией;
- Nagware — модель лицензирования компьютерных программ, использующая навязчивое напоминание о необходимости регистрации программы за определённую плату;
- Trialware — программное обеспечение, лицензионное соглашение которого предусматривает бесплатное использование программы без каких-либо ограничений в функциональности, но только в течение пробного периода.

Freeware — вид лицензии на программное обеспечение, который предусматривает бесплатное пользование программой. При этом разработчик может уточнять, какое именно использование может быть бесплатным (использование в личных целях, коммерческое использование).

В отличие от проприетарных свободные и открытые лицензии не оставляют права на конкретную копию программы её издателю, а передают самые важные из них конечному пользователю, который и становится владельцем. Примером свободной лицензии является GNU General Public License, которая даёт пользователю право самому распространять ПО под этой лицензией и изменять его любым способом. При этом любые изменения программы, сделанные пользователем и распространённые дальше, должны сопровождаться исходным кодом этих изменений.

18.3. О наказаниях за информационные преступления

Уголовный кодекс Российской Федерации (УК РФ) содержит главу 28 «Преступления в сфере компьютерной информации», в которой определена мера наказания за некоторые виды преступлений в области информационных технологий:

1) неправомерный доступ к охраняемой законом компьютерной информации, если это деяние повлекло уничтожение, блокирование, модификацию либо копирование компьютерной информации;

2) создание, распространение или использование компьютерных программ либо иной компьютерной информации, заведомо предназначенных для несанкционированного уничтожения, блокирования, модификации, копирования компьютерной информации или нейтрализации средств защиты компьютерной информации;
3) нарушение правил эксплуатации средств хранения, обработки или передачи охраняемой компьютерной информации либо информационно-телекоммуникационных сетей и оконечного оборудования, а также правил доступа к информационно-телекоммуникационным сетям, повлекшее уничтожение, блокирование, модификацию либо копирование компьютерной информации.

Данные деяния в зависимости от тяжести последствий, в том числе размеров нанесённого ущерба, наказываются крупными денежными штрафами, ограничением или лишением свободы на срок до семи лет.

18.4. Информационная безопасность

Информационная безопасность — защищённость информации и поддерживающей инфраструктуры информационной системы от случайных или преднамеренных воздействий естественного или искусственного характера, способных нанести ущерб субъектам информационных отношений (владельцам и пользователям информации) в рамках данной информационной системы.

Информационная безопасность достигается обеспечением доступности, целостности, и конфиденциальности обрабатываемых данных, а также доступности и целостности информационных компонентов и ресурсов системы.

Доступность информации — это состояние информации, при котором субъекты, имеющие права доступа, могут реализовывать их беспрепятственно и в течение приемлемого времени. К правам доступа относятся: право на чтение, изменение, копирование, уничтожение информации, а также права на изменение, использование, уничтожение ресурсов.

Целостность информации — это свойство информации сохранять свои структуру и/или содержание в процессе передачи и хранения. Целостность информации обеспечивается в том случае,

если данные в системе не отличаются в смысловом отношении от данных в исходных документах, т. е. если не произошло их случайного или намеренного искажения, или разрушения.

Конфиденциальность информации — это статус, предоставленный информации или данным и определяющий требуемую степень их защиты. К конфиденциальным данным можно отнести, например, личную информацию пользователей, учётные записи (имена и пароли), данные о кредитных картах, данные о разработках и различные внутренние документы, бухгалтерские сведения. Конфиденциальная информация должна быть известна только допущенным и прошедшим проверку (авторизованным) субъектам системы (пользователям, процессам, программам). Для остальных субъектов системы это информация должна быть недоступна.

Совокупность официальных взглядов на цели, задачи, принципы и основные направления обеспечения информационной безопасности представлены в **Доктрине информационной безопасности Российской Федерации**, согласно которой под информационной безопасностью Российской Федерации понимается состояние защищённости её национальных интересов в информационной сфере, определяющихся совокупностью сбалансированных интересов личности, общества и государства.

В доктрине выделены четыре основные составляющие национальных интересов Российской Федерации в информационной сфере:

1) соблюдение конституционных прав и свобод человека и гражданина в области получения информации и пользования ею, обеспечение духовного обновления России, сохранение и укрепление нравственных ценностей общества, традиций патриотизма и гуманизма, культурного и научного потенциала страны;
2) информационное обеспечение государственной политики Российской Федерации, связанное с доведением до российской и международной общественности достоверной информации о государственной политике Российской Федерации, её официальной позиции по социально значимым событиям российской и международной жизни, с обеспечением доступа граждан к открытым государственным информационным ресурсам;
3) развитие современных информационных технологий, отечественной индустрии информации, в том числе индустрии

средств информатизации, телекоммуникации и связи, обеспечение потребностей внутреннего рынка её продукцией и выход этой продукции на мировой рынок, а также обеспечение накопления, сохранности и эффективного использования отечественных информационных ресурсов;

4) защита информационных ресурсов от несанкционированного доступа, обеспечение безопасности информационных и телекоммуникационных систем, как уже развёрнутых, так и создаваемых на территории России.

В доктрине описаны правовые, организационно-технические и экономические методы обеспечения информационной безопасности Российской Федерации, приведены основные положения государственной политики и представлены организационные основы обеспечения информационной безопасности нашей страны.

18.5. Защита информации

Защита информации — деятельность, направленная на предотвращение утечки защищаемой информации, несанкционированных и непреднамеренных воздействий на защищаемую информацию[1)].

Утечка информации (её неконтролируемое распространение) происходила во все времена. Многие реальные эпизоды, связанные с утечкой информации или её предотвращением, положены в основу кинофильмов о спецслужбах. С развитием информационных и коммуникационных технологий ещё одним каналом утечки информации стали компьютерные сети.

Различают несанкционированное и непреднамеренное воздействие на информацию.

Несанкционированным является воздействие на защищаемую информацию с нарушением установленных прав и (или) правил доступа, приводящее к утечке, искажению, подделке, уничтожению, блокированию доступа к информации, а также к утрате, уничтожению или сбою функционирования носителя информации. Такого рода воздействие на информацию или ресурсы информационной системы может осуществляться с помощью вредоносных программ (вирусов).

1) ГОСТ Р 50922–2006 «Защита информации. Основные термины и определения».

Непреднамеренное воздействие на информацию происходит вследствие ошибок пользователя, сбоя технических и программных средств информационных систем, природных явлений или иных нецеленаправленных на изменение информации событий.

Меры, принимаемые для защиты информации, прежде всего зависят от уровня её использования.

Пользователь, являющийся частным лицом, может обеспечить защиту информации на своём компьютере, если будет:

- постоянно использовать блок бесперебойного питания;
- периодически выполнять резервное копирование файлов на внешние носители;
- постоянно использовать антивирусную программу, регулярно обновлять антивирусные базы и осуществлять антивирусную проверку компьютера.

Возможности для защиты информации от случайной потери или удаления предусмотрены и в программном обеспечении компьютера. Например:

- предусмотрены предупреждения о наличии в электронном документе макроопределений, под которые могут быть замаскированы вирусы;
- имеется возможность отменять последние действия;
- предусмотрены запросы на подтверждение команд, приводящих к изменению содержания или удалению файла или группы файлов;
- возможности изменения файлов ограничиваются установкой атрибутов (например, атрибута «только для чтения») и т. д.

Для обеспечения более высокого уровня защиты информации в дополнение к техническим средствам применяются системы шифрования.

Криптография — наука, занимающаяся методами шифрования и защиты целостности информации. Благодаря её современным достижениям разработана и получает всё более широкое применение технология электронной подписи, позволяющая отказаться от необходимости передачи подлинников документов только в бумажном виде.

САМОЕ ГЛАВНОЕ

В нашей стране разработан ряд законов, обеспечивающих правовое регулирование в информационной сфере.

Отношения, возникающие при осуществлении права на поиск, получение, передачу, производство и распространение ин-

формации, применении информационных технологий, обеспечении защиты информации регулируются Федеральным законом «Об информации, информационных технологиях и о защите информации».

Особого внимания заслуживают такие информационные объекты, как программы для электронных вычислительных машин (программы для ЭВМ) и базы данных. Наравне с произведениями науки, литературы или искусства, они считаются результатами интеллектуальной деятельности. Права на результаты интеллектуальной деятельности (авторское право, исключительное право, право собственности и др.) регулируются четвёртой частью ГК РФ.

Использование ПО является законным только тогда, когда на это есть согласие владельца авторских прав (компании-производителя или независимого разработчика). Основой правовых отношений между пользователем и собственником ПО является лицензия.

УК РФ содержит главу 28 «Преступления в сфере компьютерной информации», в которой определена мера наказания за следующие виды преступлений в области информационных технологий: неправомерный доступ к компьютерной информации; создание, распространение или использование вредоносных компьютерных программ; нарушение правил эксплуатации компьютерного оборудования и информационно-телекоммуникационных сетей.

Информационная безопасность — защищённость информации и поддерживающей инфраструктуры информационной системы от случайных или преднамеренных воздействий естественного или искусственного характера, способных нанести ущерб субъектам информационных отношений (владельцам и пользователям информации) в рамках данной информационной системы. Информационная безопасность достигается обеспечением доступности, целостности и конфиденциальности обрабатываемых данных, а также доступности и целостности информационных компонентов и ресурсов системы.

Совокупность официальных взглядов на цели, задачи, принципы и основные направления обеспечения информационной безопасности представлены в Доктрине информационной безопасности Российской Федерации.

Защита информации — деятельность, направленная на предотвращение утечки защищаемой информации, несанкционированных и непреднамеренных воздействий на защищаемую информацию.

Вопросы и задания

1. Какие права принадлежат собственнику информационного объекта?
2. Кто считается владельцем информации?
3. Кто считается потребителем информации?
4. Каким законом регулируются отношения, возникающие при осуществлении права на поиск, получение, передачу, производство и распространение информации?
5. К каким объектам приравнены компьютерные программы и базы данных? Чем регулируются права на эти информационные объекты?
6. В каком случае использование программного обеспечения является законным?
7. Что такое лицензия на программное обеспечение? Какие виды лицензий вам известны? Приведите примеры известных вам продуктов, имеющих лицензии разных видов.
8. Какие деяния Уголовный кодекс РФ классифицирует как преступления в сфере компьютерной информации?
9. Зачем нужны законодательные акты в информационной сфере?
10. Что такое информационная безопасность информационной системы? За счёт чего она достигается?
11. Что понимается под доступностью информации? Приведите пример, когда это условие нарушается.
12. Что понимается под целостностью информации? Приведите пример, когда это условие нарушается.
13. Что понимается под конфиденциальностью информации? Приведите пример, когда это условие нарушается.
14. В чём, на ваш взгляд, проявляются доступность, целостность и конфиденциальность при взаимодействии:
    1) между поликлиникой и пациентом;
    2) между школой и родителями (законными представителями) ученика;
    3) между банком и его клиентом?
15. В чём заключаются интересы личности, общества и государства в информационной сфере? Для ответа на вопрос используйте Доктрину информационной безопасности Российской Федерации.

16. Что относится к национальным интересам Российской Федерации в информационной сфере?
17. Найдите в электронных словарях и проанализируйте определения понятий «концепция», «парадигма», «хартия», «доктрина». Что общего в этих понятиях? В чём основное различие между ними?
18. В чём заключается защита информации?
19. Чем отличается несанкционированное воздействие на информацию от непреднамеренного воздействия на информацию? В чём их опасность?
20. Какие меры следует принимать для защиты информации на своём личном компьютере?
21. Какие меры по защите информации принимаются в вашей школе?

Дополнительные материалы к главе смотрите в авторской мастерской.

Заключение

Уважаемые выпускники!

Важная и всё возрастающая роль информации в современном мире требует от человека наличия информационной культуры, основными компонентами которой являются информационное мировоззрение и информационная грамотность.

Формирование основ информационной культуры как необходимого условия вашей жизни и деятельности в современном высокотехнологичном информационном обществе было одной из основных идей базового уровня изучения информатики в 10 и 11 классах.

Весь материал, предлагавшийся вам для изучения, был структурирован по следующим темам:

1) информация и информационные процессы;
2) компьютер и его программное обеспечение;
3) представление информации в компьютере;
4) элементы теории множеств и алгебры логики;
5) современные технологии создания и обработки информационных объектов;
6) обработка информации в электронных таблицах;
7) алгоритмы и элементы программирования;
8) информационное моделирование;
9) сетевые информационные технологии;
10) основы социальной информатики.

Мы надеемся, что вы смогли расширить и систематизировать свои представления, развить умения и навыки по каждой из этих содержательных линий, тем самым:

- сформировав устойчивые представления о роли информатики, о влиянии информационных и коммуникационных технологий на жизнь человека в обществе;

- развив основы логического и алгоритмического мышления;
- научившись различать факты и оценки, сравнивать оценочные выводы, проверять на достоверность и обобщать информацию;
- приняв правовые и этические аспекты информационных технологий; осознав ответственность людей, вовлечённых в создание и использование информационных систем, распространение информации;
- получив базу для последующей учебной, творческой и профессиональной деятельности, для непрерывного образования на протяжении всей жизни.

Успехов вам!

Оглавление