ИТОГОВЫЙ КОНТРОЛЬ

Информатика

Учебно-справочные материалы для 9 класса

Экзамен с «Просвещением»

ИТОГОВЫЙ КОНТРОЛЬ: ГИА

Информатика

ГОСУДАРСТВЕННАЯ ИТОГОВАЯ АТТЕСТАЦИЯ

Учебно-справочные материалы для 9 класса

Москва
Санкт-Петербург
«ПРОСВЕЩЕНИЕ»
2011

УДК 002(03)
ББК 32.81я2
И74

Проект «Итоговый контроль»
Серия «Итоговый контроль: ГИА» основана в 2010 году

Руководитель проекта *М. А. Поляков*
Научный руководитель проекта к. п. н. *Г. С. Ковалёва*

Авторы:
С. М. Авдошин, Р. З. Ахметсафина, О. В. Максименкова, И. Н. Лесовская, М. В. Курак, Н. П. Липкин, С. А. Семикина

Информатика: ГИА: Учебно-справочные материа-
И 74 лы для 9 класса (Серия «Итоговый контроль: ГИА») / С. М. Авдошин, Р. З. Ахметсафина, О. В. Максименкова, И. Н. Лесовская, М. В. Курак, Н. П. Липкин, С. А. Семикина.— М.; СПб.: Просвещение, 2011.— 252 с.: ил.

ISBN 978-5-09-024862-4.

Пособие предназначено для отработки основных знаний и умений выпускников, необходимых для успешной сдачи ГИА. Оно поможет систематизировать знания по предметам, сконцентрировать внимание на наиболее важных вопросах курсов, выносимых на экзамен, а также правильно выстроить стратегию и тактику подготовки к ГИА. Пособие содержит краткий теоретический курс среднего (полного) общеобразовательного уровня, представленный на основе кодификатора, разработанного Федеральным институтом педагогических измерений (ФИПИ). Каждый раздел сопровождается примерами типовых заданий разных уровней сложности.

УДК 002(03)
ББК 32.81я2

ISBN 978-5-09-024862-4

Предисловие

Книга, которую вы держите в руках, представляет собой учебно-справочное пособие для подготовки к Государственной (итоговой) аттестации (ГИА) по информатике и ИКТ и предназначена как для учеников 8—9-х классов средних школ, так и для их педагогов.

Настоящее пособие призвано помочь систематизировать материал, изученный при подготовке к аттестации, а также выявить и устранить пробелы в знаниях.

В настоящее время в России изданы учебники разных авторов по информатике и ИКТ для 8—9-х классов средних общеобразовательных и специализированных школ. Последовательность изложения материала и его объём в разных учебниках зачастую существенно различаются. Соответственно различаются и учебные программы по информатике и ИКТ в разных школах. Например, в одних учебниках делается упор на моделирование, в других — на программирование, в третьих — на информационные технологии и т. д. Данное пособие не заменяет учебники по информатике и ИКТ, но делает их использование более эффективным при подготовке к ГИА.

При создании учебно-справочного пособия авторы опирались на «Кодификатор элементов содержания экзаменационной работы и требований к уровню подготовки выпускников для проведения в 2010 году государственной (итоговой) аттестации (в новой форме) по информатике и ИКТ обучающихся, освоивших основные общеобразовательные программы основного общего образования».

Элементы содержания кодификатора представлены в пособии с разной степенью детализации. Некоторые элементы содержания, не нашедшие отражения в заданиях ГИА, не включены в пособие. В то же время авторы предлагают материал, не включённый в разделы кодификатора, но необходимый для решения заданий ГИА.

Поскольку задания ГИА, связанные с программированием, опираются на школьный алгоритмический язык (АЯ), то в главах, посвящённых алгоритмизации и программированию, авторы используют именно школьный АЯ.

В данном пособии приведён справочный теоретический материал, на который можно опираться при выполнении тренировочных заданий ГИА, а также рассмотрены примеры решения этих заданий.

В конце каждой главы предлагаются задания разных типов для их самостоятельной проработки, требующие знания теории, описанной в текущей и предыдущих главах. Все задачи снабжены ответами и дифференцированы по уровню сложности, а именно задачи повышенного уровня отмечены знаком *, задачи базового уровня без отличительных отметок.

Кроме того, пользователь данного пособия и пособия, содержащего контрольные тренировочные материалы по информатике, может развить практические навыки решения заданий ГИА.

Важные термины и определения в тексте выделены жирным шрифтом или оформлены в рамки, разделы для дополнительного чтения — боковой вертикальной линейкой.

Авторы благодарят за помощь в подготовке учебно-справочных материалов ***Галину Васильевну Этину***, заместителя директора по информационным технологиям лицея № 126 Санкт-Петербурга.

Надеемся, что материал данного пособия окажется полезным для вас и поможет успешно пройти государственную (итоговую) аттестацию.

Приятного и плодотворного чтения!

ГЛАВА 1

ИНФОРМАЦИЯ И ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ

Во все времена информация играла важную роль в жизни людей: её получали древние охотники, идущие по следу; секреты мастерства ремесленников передавались от отца к сыну; она сохранялась в песнях и сказаниях, а также в наскальных рисунках и пергаментных свитках.

Понятия «вещество», «время», «энергия», «пространство» долгое время были основой научной картины мира. В XX веке к этим понятиям добавилось понятие «информация». Появилась вычислительная техника, начали бурно развиваться науки, изучающие информацию и информационные процессы.

Строгого определения информации нет так же, как нет определения точки в геометрии. Латинское слово informatio (информация) означает «сведения, разъяснение, изложение».

В учебниках, словарях и других источниках можно встретить следующие определения слова «информация»:

- сведения об окружающем мире и протекающих в нём процессах, воспринимаемые человеком или специальным устройством (толковый словарь русского языка С. И. Ожегова);
- любые данные или сведения, которые кого-либо интересуют (толковый словарь русского языка С. И. Ожегова);
- сведения (сообщения, данные) независимо от формы их представления (Федеральный закон Российской Федерации от 27 июля 2006 г. N 149-ФЗ «Об информации, информационных технологиях и о защите информации»);
- отражение внешнего мира с помощью знаков и сигналов;
- совокупность данных, зафиксированных на материальном носителе, сохранённых и распространённых во времени и пространстве;
- осознанные сведения об окружающем мире, которые являются объектом хранения, преобразования, передачи и использования;
- сведения об объектах и явлениях окружающей среды, их параметрах, свойствах и состоянии, которые воспринимают живые организмы, технические системы и др. в процессе жизнедеятельности и работы, позволяющие им реагировать на окружающую среду, обеспечивая целенаправленную деятельность.

Информация не является ни веществом, ни энергией. В отличие от них она может появляться и исчезать. Особенность информации заключается в том, что проявляется она только при взаимодействии объектов.

Информация существует в виде текстов, рисунков, электромагнитных волн, жестов, запахов, хромосом и т. д. Человек получает информацию из окружающего мира с помощью своих органов чувств в форме сообщений. Содержащиеся в них сведения должны быть понятными получателю. Например, сообщение на иностранном языке, которого получатель не знает, не несут информации, так как человек не поймёт

сообщения. Новой информации не несёт сообщение, если его содержание уже известно получателю.

К основным свойствам информации относят:

- понятность — информация выражена языком, понятным получателю;
- достоверность — информация отражает истинное положение дел;
- полнота — информация достаточна для принятия решения;
- актуальность — информация важна и существенна в настоящее время;
- ценность (полезность) — определяется задачами, которые можно решить с её помощью и т. д.

Информацию, зафиксированную каким-либо способом, называют **информационным объектом**.

В современной жизни информация играет очень важную роль. «Кто владеет информацией, тот владеет миром!» — эта крылатая фраза может считаться девизом современного информационного общества.

Информация является предметом изучения науки «Информатика». Французское слово informatique образовано из двух слов: information (информация) и automatique (автоматика) — и означает «автоматизированная обработка информации». Аналог этого термина на английском языке — computer science (наука о компьютерной технике).

Информатика состоит из множества различных дисциплин — из теории информации, программирования, теории алгоритмов и др.

Информатика — это наука, изучающая структуру и свойства информации, принципы, законы и методы её поиска, хранения, передачи и обработки с использованием компьютерной техники.

Информация не существует сама по себе, она проявляется в информационных процессах. **Информационные процессы** — это процессы поиска, хранения, передачи и обработки информации. Информационные процессы изменяют содержание информации или форму её представления.

Для хранения информации необходим физический носитель информации. Информация может храниться в памяти людей, в книгах, справочниках, на магнитных и оптических дисках и других носителях.

В передаче информации всегда участвуют две стороны — *источник* и *приёмник*. Передача информации от источника к приёмнику происходит по каналам связи в форме сигналов.

Обработка информации заключается либо в получении новой информации из имеющейся, либо в изменении формы представления информации. Новую информацию получают путём вычислений или логических рассуждений. Форма представления информации меняется, например, при переводе текста с одного языка на другой. Одним из основных средств обработки информации является компьютер.

Компьютер — это программно-управляемое электронное устройство автоматической обработки информации.

Различают два основных класса компьютеров:

— *аналоговые компьютеры*, обрабатывающие непрерывно меняющиеся физические величины, которые являются аналогами вычисляемых величин;

— *цифровые компьютеры*, обрабатывающие данные в виде числовых двоичных кодов.

В школьном курсе информатики изучаются только цифровые компьютеры.

Информационно-коммуникационные технологии (ИКТ) — совокупность технических и программных средств и приёмов хранения, передачи, обработки и поиска информации.

ГЛАВА 2
ПРЕДСТАВЛЕНИЕ ИНФОРМАЦИИ

Язык как способ представления и передачи информации

Информация часто передаётся в устной или письменной форме на **естественном языке** (*русском, английском* и др.). Язык должен быть известен всем людям, участвующим в общении. Кроме естественных языков существуют **формальные**, или **искусственные, языки**. Формальные языки широко используются в науке и технике. Они являются средством более точного обмена информацией между людьми, чем естественный язык. Например, *математические символы и формулы* — формальный язык математики, *ноты и правила их записи* — формальный язык музыки. К формальным языкам относятся также *азбука Морзе, системы счисления, языки программирования, обозначения логических схем* и т. д.

Язык — знаковый способ представления информации. С помощью языка информация передаётся в знаковой форме.

Знаковая система состоит из упорядоченного набора знаков (символов), который называется **алфавитом**. Полное количество символов алфавита называется **мощностью алфавита**. Например, алфавит русского языка состоит из 33 букв, латинского — из 26 букв.

Минимально возможное количество символов в алфавите равно двум. Существующие технические электронные устройства надёжно сохраняют и распознают только два различных состояния, поэтому именно такой алфавит используется в компьютере. Он называется **двоичным алфавитом**, его символы — цифры 0 и 1. С помощью этих двух символов можно представить любую информацию в компьютере.

Если для сообщения используется двоичный алфавит и длина сообщения — один знак, можно составить два различных сообщения (0 и 1). Если длина сообщения — два знака, можно сформировать $2 \times 2 = 2^2 = 4$ разных комбинации (00, 01, 10, 11). При длине сообщения три знака получим $2 \times 2 \times 2 = 2^3 = 8$ различных комбинаций (000, 001, 010, 011, 100, 101, 110, 111). И т. д.

Комбинации символов двоичного алфавита называют **двоичными кодами**. Количество знаков в коде называется **длиной кода**.

Если мощность алфавита равна 2, длина кода равна L, можно составить $K = 2^L$ различных двоичных кодов.

Пример 2.1. Какой должна быть минимальная длина двоичного кода, если требуется составить 18 различных комбинаций?

Решение. Известно, что если длина двоичного кода равна трём, можно составить 8 различных комбинаций. При длине кода 4 символа можно составить $2^4 = 16$ различных комбинаций, при длине кода 5 символов — $2^5 = 32$ комбинации.

Ответ: минимальная длина кода равна 5.

Пример 2.2. Световое табло состоит из лампочек. Каждая лампочка может находиться в одном из трёх состояний («включено», «выключено» или «мигает»). Какое наименьшее количество лампочек должно находиться на табло, чтобы с его помощью можно было передать 18 различных сигналов?

Решение. Нам нужно найти количество лампочек, которое обеспечит возможность передачи 18 различных сигналов. Сколько сигналов можно передать с помощью одной лампочки? Три сигнала. А сколько различных сигналов можно передать двумя лампочками? В этом случае первая лампочка может находиться в одном из трёх состояний, при этом вторая также может находиться в одном из трёх состояний. Это даёт $3 \times 3 = 9$ комбинаций, т. е. можно передать 9 различных сигналов. Если взять 3 лампочки, то можно составить уже $3 \times 3 \times 3 = 27$ комбинаций. Двух лампочек явно не достаточно, чтобы закодировать 18 сигналов, однако трёх более чем достаточно.

Ответ: минимальное количество лампочек равно 3.

Пример 2.3 *. Сформулируем задачу несколько иначе (см. пример 2.2). Мощность алфавита равна трём (знаки 0, 1, 2). Какой должна быть длина троичного кода, чтобы закодировать 18 комбинаций?

Решение. Каждая цифра в коде соответствует состоянию лампочки из предыдущего примера. Рассуждаем аналогично первому варианту решения. Одна цифра позволит закодировать 3 сигнала. Две цифры дадут 9 различных сигналов, а три цифры — 27 различных сигналов. Значит, на 18 сигналов потребуется код длины не меньше трёх.

Ответ: минимальная длина троичного кода равна 3.

Если мощность алфавита равна N, длина кода L, можно составить $K = N^L$ различных кодовых комбинаций.

Моделирование объектов и процессов

Моделирование является мощным инструментом познания окружающей нас действительности.

Человек постоянно сталкивается с моделями тех или иных объектов, процессов или явлений, иногда даже не подозревая об этом. Моделью человеческой фигуры являются манекены в портновских мастерских и магазинах. Игрушечные коллекционные машинки представляют собой модели полноразмерных действующих автомобилей. Глобус — это модель Земли. Все эти *модели предметные*, они похожи на людей, машины, планету.

Маленькие дети любят играть в «дочки-матери», «магазин», «в школу». В игре *моделируются отношения*, которые складываются в определённых обстоятельствах.

Моделями являются скульптуры, театральные постановки, литературные произведения.

Для изучения силы трения на уроках физики проводятся опыты: к бруску прикрепляется динамометр, затем брусок заставляют двигаться равномерно. При изменении условий опыта (площади и массы бруска, угла наклона поверхности, по которой он движется) снимают показания, анализ которых позволяет вывести формулу, описывающую закон трения.

При проектировании новых двигателей их проверяют на испытательных стендах, которые позволяют смоделировать разные условия работы двигателя. Это дешевле и безопаснее, чем проводить реальные испытания.

Любая экономическая реформа (повышение налогов, сокращение рабочих мест, вложения в развитие предприятия или отрасли) затрагивает интересы многих людей. Проведение реальных экспериментов может потребовать больших затрат и привести к нежелательным результатам. Для прогнозирования результатов реформ используется *имитационное моделирование* — один из методов исследования сложных экономических систем.

Основные понятия

Человечество постоянно создаёт и использует модели объектов окружающего мира. Объект-оригинал иногда сложно или даже невозможно изучать: он может быть слишком маленьким (атом, электрон) или слишком большим (Солнечная система), медленным (процесс развития общества) или очень быстрым (химическая реакция), иметь неизвестные свойства и взаимосвязи, быть недоступным, эксперименты с ним могут быть опасными или дорогостоящими. Без использования моделей невозможно проектирование и создание технических устройств, машин и механизмов, зданий и сооружений. Развитие науки и получение образования невозможно без *теоретических моделей* (теорий, законов, гипотез), отражающих строение, свойства и поведение реальных объектов. Модели используются для представления объектов, объяснения

известных фактов, построения гипотез, получения новых знаний об исследуемых объектах, прогнозирования, управления и т. д.

Объектом моделирования может быть материальный предмет, явление, событие, процесс, который требуется изучить или описать.

Субъектом моделирования является человек, который занимается моделированием.

Объекты моделирования могут быть **естественными** (Солнечная система, животные и растения, круговорот воды в природе) и **искусственными** (автомобиль, самолёт, книга, математическая или химическая формула). Модели естественных объектов всегда проще, чем оригинал. Естественные объекты имеют множество свойств и признаков, о них далеко не всё известно, к тому же они могут быть очень сложными. При моделировании искусственных объектов принципиально возможно создание точной модели оригинала, но чаще при моделировании таких объектов всё же выделяют некоторые их существенные свойства.

Модели всегда не сложнее объектов, для которых они созданы. Некоторые свойства объектов не учитывают в соответствии с целями моделирования, но бывает и так, что о некоторых свойствах просто не знают.

Что общего имеют приведённые выше примеры?

1. Во всех примерах есть **объект моделирования** (человек, Земля, двигатель, отношения людей, сила трения, экономические отношения).
2. Любая модель соответствует объекту, **подобна** ему по внешнему виду, по структуре или по поведению.
3. Любая модель строится с какой-либо **целью**, определённой субъектом моделирования.
4. Модель отражает лишь некоторые, наиболее **существенные** свойства и признаки объекта. Свойства и признаки объекта выделяются в зависимости от целей моделирования и решаемых задач.
5. Модель создаётся для **получения** или **распространения информации** об объекте моделирования, необходимой для решения поставленных задач и достижения целей.

Описание **внешнего вида объекта** сводится к перечислению его признаков и необходимо для идентификации (распознавания) объекта, долговременного хранения образа объекта (фотография, портрет).

Структурой объекта называют совокупность его элементов и связей между ними. Моделирование структуры используется для наглядного представления, выявления значимых связей и т. д.

Поведение объекта — это изменения, происходящие с ним во времени. Описание поведения сводится к описанию внешнего вида и структуры с течением времени в результате взаимодействия с другими объектами и используется для прогнозирования, управления и т. д.

Свойство объекта, которое можно выразить относительно постоянным показателем, называют **параметром модели**. Параметры указывают, чем данный объект отличается от других. Параметры могут быть не только количественными, но и качественными (например, словесное описание объекта).

Важно пояснить, что существенность и несущественность свойств объекта — понятия относительные. Например, для кассира по продаже авиабилетов модель самолёта — это план салона, а существенные признаки — количество рядов кресел и мест в ряду, наличие свободных мест и т. д. Для авиадиспетчера модель самолёта — это светящаяся точка на экране радара, существенные признаки — скорость и высота полёта, направление и вид движения (взлёт, посадка и т. п.). Для инженера авиастроительного предприятия модель самолёта — это конструкторские чертежи, перечень деталей, технологическая карта сборки, а существенные признаки — наименование и количество деталей, квалификация специалистов, выполняющих сборку, и т. п.

Разные объекты могут описываться одной моделью. Например, если размер объекта значительно меньше его перемещений (движение планет, автомобиля, мяча), можно использовать модель движения материальной точки.

Для описания и исследования одного и того же объекта могут использоваться разные модели.

Для описания и исследования разных объектов может использоваться одна и та же модель.

Термин «модель» имеет множество значений. Моделями могут быть:

- уменьшенные копии предметов;
- формулы (математические, физические, химические и т. д.);
- схемы физических явлений (модель движения планет Солнечной системы, модель получения изображения при использовании линз);
- описание последовательности действий (разбор предложения по составу, последовательность изготовления швейного изделия, порядок зачисления абитуриентов в вузы);
- эталонные модели (метра, килограмма).

Существуют разные определения понятия «модель». Приведём одно из них:

Модель — это новый объект, который отражает наиболее существенные (с точки зрения целей моделирования) свойства изучаемого объекта.

Моделирование — это построение моделей для исследования и изучения объектов окружающей действительности.

В результате моделирования человек получает информацию, которую может использовать для определения и улучшения характеристик реальных объектов и процессов, анализа явлений, создания новых объектов, принятия решений.

При моделировании изучение свойств объекта заменяется исследованием свойств его модели, что существенно облегчает, а в ряде случаев делает принципиально возможным проведение исследования.

Моделирование используется в познании, общении и практической деятельности и является неотъемлемым элементом любой целенаправленной деятельности.

Информационные модели

Модели можно классифицировать по разным признакам:

- по областям использования — опытные, игровые, имитационные;
- отраслям знаний — биологические, химические и т. д.;
- способам представления — материальные и информационные.

Материальные (натурные) модели — это уменьшенные или увеличенные копии, воспроизводящие внешний вид моделируемого объекта, его структуру или поведение (макет нефтяной вышки, модель атома, глобус);

В информатике рассматриваются информационные модели.

Информационная модель — это описание моделируемого объекта на каком-либо языке (словесное описание, схемы, чертежи, карты, формулы, рисунки), отражающее наиболее существенные с точки зрения цели моделирования свойства, признаки и состояния объекта, его взаимосвязь с внешним миром.

Например, модель поваренной соли можно описать словами (поваренная соль — это соединение одного атома натрия и одного атома хлора) или представить в виде химической формулы NaCl.

Информационная модель, как и любой другой вид информации, имеет материальный носитель. Им может быть бумага, классная доска, экран монитора и т. п. Модель может быть зафиксирована с помощью типографской краски, чернил и т. п. Для информационной модели важен смысл, который она передаёт, а не материальный носитель.

Приведём ещё одно определение информационной модели.

Модель, представляющая объект набором параметров и связей между ними, называется **информационной моделью**.

Информационные модели делятся на *описательные* и *формальные*.

Описательные информационные модели — это модели, созданные на естественном языке в устной или письменной форме. Их также называют *вербальными*. Они, как правило, отображают объекты моделирования качественно, не используя количественных характеристик. Например, модель мира Коперника на естественном языке: Земля вращается вокруг Солнца, Луна вращается вокруг Земли. Все планеты вращаются вокруг Солнца.

Формальные информационные модели — это модели, созданные на формальном языке. Их также называют *знаковыми*. Примерами знаковых моделей являются формулы, таблицы, карты, блок-схемы и т. д.

Процесс построения информационных моделей с помощью формальных языков называется **формализацией**.

Этапы разработки информационной модели

Перед разработкой модели должны быть обязательно чётко сформулированы цели моделирования. Это определит результат всей дальнейшей работы.

Разработка модели выполняется в несколько этапов:

- анализ объекта моделирования и выделение его свойств;
- анализ выделенных свойств с точки зрения цели моделирования и определение существенных свойств;
- выбор формы представления модели;
- формализация;
- анализ модели на непротиворечивость;
- анализ адекватности модели целям и задачам моделирования.

На первом этапе разработки модели в результате анализа выявляются структура объекта, т. е. элементы, их свойства и связи между ними, и при необходимости поведение объекта.

Далее определяют наиболее существенные элементы, свойства и связи реальных объектов. Информации об объектах может быть очень много, не вся она важна для целей моделирования. Например, если создаётся информационная модель «Успеваемость учащихся», важны такие сведения, как фамилия, имя ученика, класс, в котором он учится, предметы, которые он изучает, оценки. Несущественными для модели являются сведения о том, какие фильмы нравятся ученику, его вес, размер обуви, цвет его глаз и т. д. От того, насколько правильно и полно выделены существенные признаки, зависит соответствие модели поставленной цели, т. е. её адекватность цели моделирования.

На следующем этапе выбирается форма представления информационной модели. При этом следует учитывать поставленные задачи, возможности средств и методов моделирования. От формы представления существенных признаков объекта зависит адекватность модели объекту моделирования. При компьютерном моделировании следует учитывать возможности программного обеспечения, которое будет использоваться.

Формализация как этап разработки модели — это процесс представления информации о существенных свойствах объекта в выбранной форме. Например, математические модели строятся с использованием математических понятий и формул.

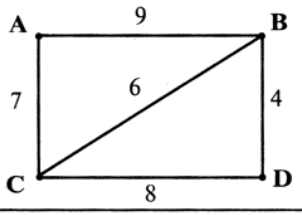
Информационная модель может быть построена с использованием *графов*.

Граф — это средство наглядного представления элементов объекта и связей между ними. Граф состоит из вершин, связанных дугами или рёбрами. Вершины изображают точками, кругами, овалами, прямоугольниками и т. д., связи между вершинами — линиями. Если линия направленная, т. е. со стрелкой, она называется *дугой*. Если линия ненаправленная, т. е. без стрелок, она называется *ребром*. Одно ребро заменяет две дуги между противоположными вершинами. Две вершины, соединённые дугой или ребром,

называются *смежными*. Граф, у которого каждому ребру или дуге сопоставлено некоторое число, называется *нагруженным*. Это число может обозначать расстояние между вершинами, время перехода от одной вершины к другой и т. д. *Путь в графе* — это конечная последовательность вершин графа, где каждая из вершин соединена со следующей в последовательности вершиной как минимум одним ребром (дугой).

Граф может быть задан разными способами: *списком дуг, графически* или *таблицей*. С помощью графа удобно представить модель населённых пунктов, связанных дорогами (см. ниже).

Формы представления информационных моделей

Список дуг	Графическая форма	Табличная форма
(AB; 9), (BD; 4), (DC; 8), (AC; 7), (CB; 6)		

	A	B	C	D
A		9	7	
B	9		6	4
C	7	6		8
D		4	8	

В табличной форме на пересечении строк и столбцов, обозначающих населённые пункты, показаны расстояния между ними. Такая таблица называется *таблицей смежности*. Исследовать подобную модель можно с использованием теории графов и разработанных на её основе алгоритмов.

Построенную модель необходимо проверить на непротиворечивость и проанализировать, насколько она адекватна объекту и цели моделирования.

Приведём пример. Цель моделирования — составить расписание занятий в школе, т. е. определить для каждого класса порядок проведения занятий на каждый день и представить его в наглядной форме. Объект моделирования — организация учебного процесса. Субъект моделирования — завуч. Задачи моделирования: организовать учебный процесс так, чтобы выполнить учебный план, чтобы количество уроков в день было не больше шести и т. д.

Объект моделирования включает элементы: список классов школы; список учителей; список дисциплин для каждого класса и количество часов в неделю на каждую дисциплину; перечень учебных кабинетов; максимальное количество уроков в день.

Если ограничиться перечисленными выше элементами модели, то можно получить такое расписание, где урок литературы в 9-м классе, например, будет проводить учитель начальных классов в кабинете английского языка или в каком-либо классе может быть 6 уроков алгебры

в день и т. д. Почему это возможно? Модель неадекватна объекту. В модели не выделены связи между элементами.

Следует вернуться к этапу выделения свойств и структуры объекта. Требуется указать, какие дисциплины и в каких классах ведёт каждый учитель, в каких кабинетах проводятся занятия по каждой дисциплине и т. д.

Составление расписания — очень трудоёмкий и сложный процесс. При проверке модели расписания на противоречивость следует выяснить, нет ли у разных классов занятий в одно и то же время в один день в одном и том же кабинете; нет ли у одного учителя одновременно занятий в двух и более классах и т. д.

Таким образом, модель организации учебного процесса в форме расписания неоднократно уточняется и изменяется.

Использование компьютеров расширило возможности моделирования как инструмента изучения объектов. Это связано с высоким быстродействием компьютеров; использованием алгоритмов и языков программирования; применением мультимедийных средств для организации диалоговых режимов взаимодействия пользователя и компьютера в процессе моделирования; удобством представления, визуализации, хранения и передачи по компьютерным сетям результатов моделирования и т. д.

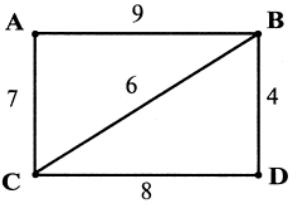
При проведении компьютерного моделирования на основе разработанной модели составляют план исследований, в который обязательно включают *разработку тестов* и *тестирование модели*. Тест предназначен для проверки правильности реализации модели. Если для исследований разработана программа, её запускают на выполнение, вводят исходные данные и получают результаты. Если модель исследуется в приложении, например в электронной таблице, то можно построить диаграммы, провести сортировку и поиск необходимых сведений, использовать другие методы обработки информации. Для тестирования модели следует ввести такие исходные данные, где результат известен заранее. Если полученные результаты не совпадают с ожидаемыми, следует уточнить модель. Возможно, не были выделены некоторые существенные свойства объекта или при разработке модели были допущены ошибки.

Результаты исследования модели анализируются, при этом считается, что объект обладает теми же свойствами, которые выявлены в модели. Анализ результатов моделирования позволяет делать выводы, строить прогнозы, принимать решения и т. д.

Использование электронных таблиц для моделирования будет рассмотрено в главе 10. Здесь мы рассмотрим решение задач с использованием графических моделей.

Пример 2.4. Задание с выбором одного ответа

На схеме нарисованы дороги между четырьмя населёнными пунктами A, B, C, D и указаны протяжённости данных дорог.



Передвигаться можно только по указанным на схеме дорогам. Кратчайшее расстояние между двумя наиболее удалёнными друг от друга пунктами равно

1) 9
2) 13
3) 15
4) 17

Решение. На графической модели — схеме дорог — изображены 4 населённых пункта. Существует 6 пар пунктов и протяжённостей дорог между ними. Все пункты, кроме A и D, попарно соединены дорогами. Заметим, что для всех пар связанных пунктов схемы выполняется неравенство треугольника (длина любой стороны треугольника не превосходит сумму длин двух его других сторон). Поэтому кратчайшее расстояние между любыми двумя населёнными пунктами, связанными дорогой, равно протяжённости этой дороги.

A и D — единственная, не соединённая дорогой пара пунктов. Вычислим минимальную протяжённость дорог между этими пунктами. Для пути A—C—D расстояние равно $7 + 8 = 15$. Для пути A—B—D расстояние равно $9 + 4 = 13$. Другие возможные пути (A—B—C—D, A—C—B—D) не рассматриваются, так как выполняется неравенство треугольника.

Запишем все минимальные расстояния между пунктами:

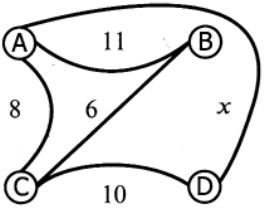
1) из A в B — 9;
2) из A в C — 7;
3) из A в D — 13;
4) из B в C — 6;
5) из B в D — 4;
6) из C в D — 8.

Таким образом, наиболее удалёнными друг от друга пунктами являются A и D, а кратчайшее расстояние между ними равно 13.

Ответ: 2.

Пример 2.5. Задание с выбором одного ответа

На схеме нарисованы дороги между четырьмя населёнными пунктами A, B, C, D и указаны протяжённости данных дорог.



Известно, что кратчайшее расстояние между наиболее удалёнными друг от друга пунктами составляет 15. Определите значение x, при котором это возможно.

1) 15
2) 6
3) 5
4) 4

Решение. На графической модели — схеме дорог — изображены 4 населённых пункта. Существует 6 пар пунктов и протяжённостей дорог между ними. Все пункты, кроме B и D, попарно соединены дорогами. Запишем все минимальные расстояния между пунктами:

1) из A в B — 11;
2) из A в C — 8;
3) из A в D — 18 или x;
4) из B в C — 6;
5) из B в D — 16 или $11 + x$;
6) из C в D — 10.

Если x будет равно 15, то минимальное расстояние A—D равно 15, но при этом расстояние B—D станет равно $11 + 15 = 26$, что не соответствует условию задачи.

Аналогично рассуждаем при $x = 6$ и $x = 5$.

При $x = 4$ расстояние A—D равно 4, расстояние B—D станет равно $11 + 4 = 15$, что и является минимальным расстоянием между наиболее удалёнными пунктами.

Ответ: 4.

Измерение информации

Вещество и энергия имеют числовые характеристики, выраженные в единицах измерения. Можно измерить вес, длину, температуру, силу тока, количество тепла. Информацию тоже можно измерить.

Информация может быть представлена в *аналоговой (непрерывной)* или *дискретной* форме. При **аналоговой форме** физическая величина, используемая для **представления информации**, может принимать бесконечное множество значений, как, например, звуковые и электромагнитные волны.

При **дискретной форме представления информации** физическая величина может принимать конечное множество значений.

Пример аналогового представления информации — картина художника, написанная маслом. Пример дискретного представления информации — та же картина на экране компьютера, где она представляется конечным числом точек экрана, каждая точка имеет свой цвет из ограниченного набора цветов.

Примером аналогового хранения звуковой информации является виниловая пластинка (звуковая дорожка изменяет свою форму непрерывно), а дискретного — аудио-компакт-диск, звуковая дорожка которого содержит участки с различной отражающей способностью.

Аналоговая форма представления информации может быть преобразована в дискретную форму. Пример дискретизации приводится в главе 4 «Передача информации. Кодирование информации» при описании кодирования звука. Возможность дискретизации непрерывного сигнала принципиально важна с точки зрения информатики.

В компьютерах внутреннее представление информации дискретно. Дискретизация аналоговой информации позволяет сделать её пригодной для компьютерной обработки.

Минимальное количество информации, которое можно получить, содержит ответ на вопрос, допускающий два варианта ответа: «да» (1) или «нет» (0).

Минимальная единица измерения информации называется **бит**. Термин bit происходит от сокращения английских слов **bi**nary digi**t** (двоичная цифра). Один бит (одна из двух цифр) соответствует одному разряду двоичного кода. Получение информационного сообщения в один бит уменьшает неопределённость нашего знания о чём-либо в два раза.

В вычислительной технике битом называют наименьшую порцию памяти, необходимую для хранения одного из двух знаков — **0** или **1**, используемых для представления информации (данных и команд) в компьютере.

Бит — очень маленькая единица информации, поэтому существует величина **байт**, представляющая собой последовательность из 8 бит (1 байт — 2^3 бит).

Более крупные единицы измерения информации обозначаются с использованием префиксов (приставок), известных нам из физики (кило, мега и т. д.). В международной системе единиц эти приставки означают кратность степени числа 10, поэтому их называют десятичными. При измерении информации каждый следующий префикс соответствует увеличению значения не в 1000, а в $\mathbf{1024 = 2^{10}}$ раз. Приставки называют двоичными и записывают, начиная с прописной буквы *. Например:

килобит: 1 Кбит = 1024 бит = 2^{10} бит;
мегабит: 1 Мбит = 2^{10} Кбит = 2^{20} бит;
гигабит: 1 Гбит = 2^{10} Мбит = 2^{20} Кбит = 2^{30} бит;
терабит: 1 Тбит = 2^{10} Гбит = 2^{20} Мбит = 2^{30} Кбит = 2^{40} бит;
петабит: 1 Пбит = 2^{10} Тбит = 2^{20} Гбит = 2^{30} Мбит = 2^{40} Кбит = 2^{50} бит.

При решении задач вам зачастую будет необходимо проводить операции над различными единицами измерения информации (складывать, делить, умножать). В таких задачах следует переходить к одной единице измерения, как в физике. А решение сведётся к устному счёту, если представлять килобайты, мегабиты и т. д. просто степенями двойки. Их будет удобно умножать и делить. Со сложением будет чуть сложнее, но всё равно вполне комфортно. Вам пригодится таблица соответствия единиц измерения информации.

---

* По международным стандартам для обозначения кратности степеням двойки должны использоваться новые обозначения двоичных приставок. Они начинаются на те же слоги, что десятичные, но второй слог у всех двоичных приставок **би** (от англ. binary (двоичный),— т. е. килобит становится кибибитом, мегабит — мебибитом и т. д. Эти обозначения пока не прижились, и на практике чаще используются привычные десятичные приставки.

1 байт	2^3 бит					
1 Кбайт	2^{13} бит	2^{10} байт				
1 Мбайт	2^{23} бит	2^{20} байт	2^{10} Кбайт			
1 Гбайт	2^{33} бит	2^{30} байт	2^{20} Кбайт	2^{10} Мбайт		
1 Тбайт	2^{43} бит	2^{40} байт	2^{30} Кбайт	2^{20} Мбайт	2^{10} Гбайт	
1 Пбайт	2^{53} бит	2^{50} байт	2^{40} Кбайт	2^{30} Мбайт	2^{20} Гбайт	2^{10} Тбайт

Пример 2.6. Задание с кратким ответом

Сколько килобайт информации содержит сообщение объёмом 2^{16} бит? В ответе укажите одно число.

Решение. Воспользуемся приведённой выше таблицей: 1 Кбайт = $= 2^{13}$ бит, отсюда, 2^{16-13} Кбайт = 2^3 Кбайт = 8 Кбайт.

Ответ: 8.

Пример 2.7. Задание с выбором одного ответа

Выберите вариант ответа, в котором объёмы памяти расположены в порядке убывания:

1) 1010 байт, 2 байт, 1 Кбайт, 20 бит, 10 бит
2) 1010 байт, 1 Кбайт, 20 бит, 2 байт, 10 бит
3) 1010 байт, 1 Кбайт, 2 байт, 20 бит, 10 бит
4) 1 Кбайт, 1010 байт, 20 бит, 2 байт, 10 бит

Решение. Приведём все величины к одной единице измерения, например, выразим все значения в битах:

1010 байт = 1010 · 8 бит;

2 байт = 2 · 8 бит = 16 бит;

1 Кбайт = 1024 байт = 1024 · 8 бит.

Расположим значения по убыванию:

1024 · 8 бит; 1010 · 8 бит; 20 бит; 16 бит; 10 бит или 1 Кбайт; 1010 байт; 20 бит; 2 байт; 10 бит.

Ответ: 4.

Системы счисления

Системы счисления — это способы представления числовой информации. Они созданы человеком, поэтому относятся к искусственным системам.

В любой системе счисления для представления чисел выбирают некоторые символы — **цифры**. В результате выполнения операций над цифрами получают числа. Цифры можно сравнить с буквами алфавита (знаками), а числа — со словами, составленными из этих букв и обозначающими предмет, понятие и т. д.

Системой счисления называют совокупность символов (цифр) и правил их использования для построения, записи и наименования чисел.

Существуют *позиционные* и *непозиционные системы счисления*. В непозиционных системах счисления количественное значение цифры числа не зависит от её позиции в записи числа. В позиционной системе счисления вклад каждой цифры в величину числа зависит от её позиции в записи числа.

Непозиционные системы счисления

В непозиционной системе счисления каждой цифре соответствует величина, не зависящая от её места в записи числа. К таким системам относится **римская** система счисления.

Цифры римской системы счисления

I — 1	X — 10	C — 100	M — 1000
V — 5	L — 50	D — 500	

Правила составления чисел в римской системе

- меньшее число, стоящее справа от большего, прибавляется к большему: XI = 10 + 1 = 11;
- меньшее число, стоящее слева от большего, вычитается из большего: IX = 10 — 1 = 9.

Как видно из приведённых примеров, цифры I и X не меняют своё количественное значение в зависимости от позиции в записи числа.

Для правильной записи больших чисел римскими цифрами необходимо сначала записать число тысяч, затем сотен, затем десятков и, наконец, единиц.

Например, число 1 998 запишем в римской системе счисления: 1 тысяча — M, 900 — CM, 90 — XC, 8 — VIII. Запишем их вместе: MCMXCVIII.

Приведём примеры вычисления значения числа, записанного в римской системе счисления.

Римское обозначение	Вычисление значения числа	Число
IV	5 – 1	4
VIII	5 + 1 + 1 + 1	8
IX	10 – 1	9
XXIX	10 + 10 + (10 – 1)	29
XLVI	50 – 10 + 5 + 1	46
XCIX	100 –10 + (10 – 1)	99
DCLXVII	500 + 100 + 50 + 10 + 5 + 1 + 1	667
DCCCLXXXVIII	500 + 100 + 100 + 100 + 50 + 10 + 10 + + 10 + 5 + 1 + 1 + 1	888
MCMXCVIII	1000 + (1000 – 100) + (100 – 10) + 5 + + 1 + 1 + 1	1998
MMIX	1000 + 1000 + (10 – 1)	2009

Позиционные системы счисления

В позиционных системах количественное значение цифры («вес» цифры) определяется её позицией в записи числа. Позиции мы будем называть также **разрядами** числа.

Главная характеристика позиционных систем — **основание системы счисления *P***. Основание системы счисления определяет:

- определённое количество цифр, используемых для записи чисел;
- вес каждого разряда (позиции) в записи чисел.

Количество цифр, равное основанию системы, а также совокупность используемых цифр называют **алфавитом системы счисления**.

Разряды целого числа нумеруются справа налево, начиная с нуля. Вес каждого разряда равен основанию системы счисления, возведённому в степень, которая, в свою очередь, соответствует номеру разряда. Вес каждого разряда образует **базис** позиционной системы счисления. В десятичной системе счисления базис целого числа образует последовательность 10^n, ..., 10^4, 10^3, 10^2, 10, 1. Например, в десятичном числе 567 цифра 5 означает пять сотен, цифра 6 — шесть десятков, цифра 7 — семь единиц. Число 567 можно представить в виде $5 \cdot 100 + 6 \cdot 10 + 7 \cdot 1$ или $5 \cdot 10^2 + 6 \cdot 10 + 7$.

Позиционные системы счисления с основанием P называют ***P*-ичными** (далее будем обозначать их **СС*p***).

Запись числа x в P-ичной системе счисления

$$x_p = x_n \ldots x_1 x_{0-1} \ldots x_{-1} x_{-2} \ldots x_{-m}$$

называется **свёрнутой формой** записи числа (это привычная нам форма записи чисел). Здесь $x_n, \ldots x_1, x_0, x_{-1}, x_{-2}, \ldots, x_{-m}$ — цифры в записи числа, n и m — количество целых и дробных разрядов числа, p — основание системы счисления, которая записывается в виде нижнего индекса справа от числа. Для чисел в десятичной системе счисления ($\mathbf{CC_{10}}$) индекс, как правило, не указывают.

x_p можно представить также в виде многочлена:

$$x_p = x_n \cdot P^n + \ldots + x_1 P^1 + x_0 P^0 + x_{-1} P^{-1} + x_{-2} P^{-2} + \ldots + x_{-m} P^{-m}.$$

Такая запись называется **развёрнутой формой** записи числа.

В любой P-ичной системе счисления натуральные числа, меньшие основания, представляются в виде одной цифры. Само число P записывается в виде 10_p ($P = 1 \cdot P + 0$). Число P^2 записывается в виде 100_p ($P = 1 \cdot P^2 + 0 \cdot P + 0$), P^3 записывается в виде 1000_p и т. д., по аналогии с десятичной системой счисления.

> Если для записи целого числа в системе счисления с основанием P используется N разрядов (цифр), то количество различных чисел равно P^N, это числа от **0** до $P^N - 1$.

Например, если для записи десятичных чисел используется три разряда ($x_2 x_1 x_0$), можно записать 1000 разных чисел — от 0 до 999 ($10^3 - 1$).

Основные задачи при изучении систем счисления:

1) перевод числа из одной системы счисления в другую;
2) выполнение арифметических действий в произвольной системе счисления.

Перевод числа из одной позиционной системы счисления в другую

Рассмотрим два правила перевода чисел из одной позиционной системы счисления в другую:

1) из системы счисления с произвольным основанием **(СС*p*)** в десятичную систему счисления **($СС_{10}$)**;
2) из десятичной системы счисления **($СС_{10}$)** в систему счисления с произвольным основанием **(СС*p*)** *.

Основное различие этих правил зависит от того, в какой системе счисления проводятся вычисления — в исходной или в новой.

Для перевода числа $x_n x_{n-1} \dots x_1 x_0 \dots x_{-1} x_{-2} \dots x_{-m}$, записанного в P-ичной системе счисления ($СС_p$), в десятичную систему счисления ($СС_{10}$) применяют формулу разложения числа по основанию системы счисления P:

$$x_n \cdot P^n + \dots + x_1 P^1 + x_0 P^0 + x_{-1} P^{-1} + x_{-2} P^{-2} + \dots + x_{-m} P^{-m}.$$

Вычисления по этой формуле проводятся в «новой» (десятичной) системе счисления. Это значит, что цифры «старой» системы счисления и её основание переводят в десятичную систему счисления, после чего проводят вычисления.

Для перевода целого числа из $СС_{10}$ в СС*p* используется следующий алгоритм: последовательно делят с остатком исходное число на основание «новой» системы счисления P до получения нулевого результата; записывают остатки, начиная с последнего, и получают представление исходного числа в СС*p*. Вычисления проводятся в исходной, или «старой» (десятичной), системе счисления.

Правила выполнения арифметических действий

Правила выполнения арифметических действий для всех позиционных СС одинаковы и совпадают с правилами в $СС_{10}$. В их основе лежат таблицы сложения и умножения однозначных чисел. При выполнении арифметических действий в СС*p* надо помнить о величине P — основании системы, в которой выполняются действия, и о том, что цифры результатов не могут быть больше максимальной цифры.

В информатике в основном рассматриваются следующие системы:

- десятичная ($P = 10 = P_{10}$, цифры: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9);
- двоичная ($P = 2 = P_2$, цифры: 0, 1);
- восьмеричная ($P = 8 = P_8$, цифры: 0, 1, 2, 3, 4, 5, 6, 7);
- шестнадцатеричная ($P = 16 = P_{16}$, цифры: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F).

Двоичная система счисления

Для моделирования работы электронных устройств компьютеров и других устройств обработки информации применяется наиболее под-

---

* Эти правила можно использовать для любых двух систем счисления.

ходящая **двоичная система счисления**. Это связано с надёжностью представления цифр 0 и 1 в электронных устройствах компьютера.

Нули могут стоять как в начале, так и в конце записи числа. Тогда говорят о **значащих** и **незначащих нулях**. Незначащие нули при записи чисел, как правило, отбрасываются:

$$\underbrace{000}_{\text{незначащие нули}}10010110\underbrace{000}_{\text{значащие нули}}{}_2$$

Для двоичных чисел действуют следующие **правила сложения**:

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 10$$

Запишем несколько первых чисел натурального ряда (P_{10}) в двоичной системе счисления (P_2):

P_{10}	0	1	2	3	4	5	6	7	8	9	10
P_2	0	1	10	11	100	101	110	111	1000	1001	1010

Ряд чисел в P_2 получен последовательным прибавлением единицы к предыдущему числу, например:

$$\begin{array}{r} 11 \\ +\quad 1 \\ \hline 100 \end{array}.$$

Рассуждаем так: по правилам сложения для $СС_2$ 1 + 1 = 10; в младшем разряде (самом правом) ноль пишем, один в уме. Для следующего разряда: 1 плюс 1 в уме тоже равно 10, записываем 10. Таким образом, слева добавляется ещё один разряд в записи числа, в нём записана 1.

Перевод двоичного числа в десятичную систему счисления

Воспользовавшись формулой разложения числа по основанию системы счисления, переведём некоторые двоичные числа в их десятичные представления:

1) $11011_2 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 16 + 8 + 2 + 1 = 27_{10}$;
2) $100010_2 = 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 32 + 2 = 34_{10}$.

Важно не забывать, что позиции целого числа нумеруются справа налево, начиная с нуля. Для того чтобы быстро и без ошибок переводить двоичные числа в десятичные, полезно запомнить базис $СС_2$ (веса разрядов) хотя бы до десятой степени двойки:

$СС_2$	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
$СС_{10}$	1024	512	256	128	64	32	16	8	4	2	1

Перевод числа из двоичной СС в десятичную СС сводится к следующему: следует в десятичной системе счисления сложить те степени двоек, которые соответствуют единицам в записи исходного двоичного числа.

Перевод десятичного числа в двоичную систему счисления

Способ 1

Для обратного преобразования числа из $СС_{10}$ в $СС_2$ будем выполнять следующие деления:

1) десятичное число делим на два, записываем остаток от деления;
2) целую часть частного делим на два, приписываем остаток от деления слева к записанным остаткам;
3) если целая часть частного равна нулю, деление прекращаем.

Остатки от деления есть цифры полученного двоичного числа.

Пример 2.8. Задание с кратким ответом

Переведите число 81_{10} в двоичную систему счисления. В ответе запишите только число без указания основания системы счисления.

Решение.

Выражение	Частное	Остаток
81 : 2	40	1
40 : 2	20	0
20 : 2	10	0
10 : 2	5	0
5 : 2	2	1
2 : 2	1	0
1 : 2	0	1

Разряды в записи числа ↑

Выпишем остатки от деления в том порядке, как указывает стрелка (снизу вверх). Получим 1010001.

Часто для перевода числа из десятичной СС в систему счисления с другим основанием деление записывают столбиком. Для примера 2.8 деление выглядит так:

```
_81 | 2
 80 | _40 | 2
  1 |  40 | _20 | 2
    |   0 |  20 | _10 | 2
    |     |   0 |  10 | _5 | 2
    |     |     |   0 |  4 | _2 | 2
    |     |     |     |  1 |  2 | _1 | 2
    |     |     |     |    |  0 |  0 | 0
    |     |     |     |    |    |  1 |
```

Остатки от деления выписываем снизу вверх (от последнего остатка к первому). Обратите внимание, что в заданиях с кратким ответом обычно не требуется указывать в ответе основание системы счисления.

Ответ: 1010001.

Способ 2

Второй способ основан на выделении максимальной степени числа 2 в десятичном числе, его ещё называют последовательным разложением числа по степеням двойки (весам двоичных разрядов).

Если в формуле разложения числа по основанию системы счисления $P = 2$, то каждая цифра x_i может принимать одно из двух значений: 0 или 1. Тогда значение числа представляется в виде суммы степеней двойки тех разрядов, в которых $x_i = 1$. Проще говоря, если степень двойки вошла в разложение десятичного числа, то в разряде двоичного числа с номером, равным степени двойки, ставится единица. Если степень двойки в разложение не вошла, на её месте ставят ноль, как показано на рис. 1.

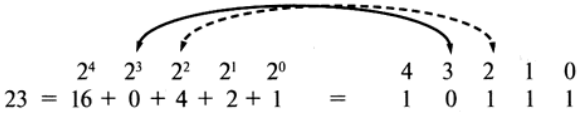


Рис. 1

Для исходного десятичного числа a надо найти такое n, при котором $2^n \le a < 2^{n+1}$. Тогда двоичная запись исходного числа a будет состоять из $n + 1$ цифр, а в позиции n двоичного числа записывается единица.

Далее определим остаток $(a - 2^n)$, обозначим его также a. Если остаток равен нулю, то вычисления прекращаются, цифры оставшихся младших разрядов равны нулю. Если остаток не равен нулю, опять определим для него максимальную степень числа 2, не превышающую a, определим очередной остаток и т. д.

Пример 2.9. Задание с кратким ответом

Перевести число 74_{10} в двоичную систему счисления.

Решение.

1) Определим ближайшую меньшую степень двойки, она равна 2^6 ($2^6 = 64 < 74 < 128 = 2^7$). Значит, в шестом разряде надо записать единицу, а всего разрядов в записи числа семь.

2) Остаётся число $74 - 64\ (2^6) = 10$. Ближайшая меньшая степень двойки — это 2^3 ($2^3 = 8 < 10 < 16 = 2^4$), в третий разряд запишем единицу.

3) Остаётся $10 - 8\ (2^3) = 2$, значит, в первый разряд ($2^1 = 2 \le 2 < 4 = 2^2$) тоже запишем единицу, в остальные разряды — нули.

$$74_{10} = 64_{10} + 8_{10} + 2_{10} = 2^6 + 2^3 + 2^1$$

Таким образом, степени двойки, которые вошли в последовательные вычитания, дают единицы в соответствующих разрядах двоичного числа, в остальных разрядах — нули. Получим 1001010_2.

Ответ: 1001010.

Ещё примеры:

1) $128_{10} = 2^7 = 10000000_2$;
2) $35_{10} = 2^5 + 2^1 + 2^0 = 100011_2$.

Двоичная запись чисел неудобна для восприятия и является очень громоздкой. Даже двузначные десятичные числа в двоичном виде имеют семь разрядов ($99_{10} = 1100011_2$). Поэтому в книгах и программных средах представление двоичных чисел заменяют их представлением в восьмеричной или шестнадцатеричной системе счисления. Эти две системы счисления являются «родственниками» двоичной, так как основания систем связаны степенной зависимостью: $8 = 2^3$, $16 = 2^4$. Это даёт возможность легко переходить от записи числа в $СС_2$ к записям в $СС_8$ и $СС_{16}$ и наоборот.

Восьмеричная система счисления

В восьмеричной системе счисления для записи чисел используются восемь цифр (0, 1, ..., 7). Перевод в десятичную систему осуществляется стандартным способом разложения по основанию системы счисления, например:

$$726_8 = 7 \cdot 8^2 + 2 \cdot 8^1 + 6 \cdot 8^0 = 7 \cdot 64 + 2 \cdot 8 + 6 \cdot 1 = 470_{10}.$$

Для перевода чисел из десятичной системы в восьмеричную систему применим метод последовательных делений.

Пример 2.10. Задание с кратким ответом

Перевести число 127_{10} в восьмеричную систему счисления.

Решение.

Выражение	Частное	Остаток
127 : 8	15	7
15 : 8	1	7
1 : 8	0	1

Запишем остатки от деления в обратном порядке: 177_8.

Ответ: 177.

Восемь — это 2^3. Существует способ перевода двоичных чисел в восьмеричные без промежуточного перевода в десятичную. Каждая цифра восьмеричной системы счисления представляется числом из трёх разрядов в двоичной системе, как показано в таблице.

Основание системы счисления			
2	8	2	8
000	0	100	4
001	1	101	5
010	2	110	6
011	3	111	7

Чтобы перевести число из восьмеричной системы в двоичную, достаточно каждую цифру восьмеричного числа заменить двоичным числом из трёх разрядов (**триадой**). Однозначные и двузначные числа двоичного числа дополняются незначащими нулями до трёх разрядов.

Пример 2.11. Задание с кратким ответом

Перевести число 701_8 в двоичную систему счисления.

Решение. Цифра 7 восьмеричной системы счисления представляется в двоичной как 111_2, цифра 0 как 000_2, цифра 1 как 001_2. Получим $111\ 000\ 001_2$.

Не забывайте, что надо обязательно использовать ровно три двоичных разряда для представления одной восьмеричной цифры! Если не придерживаться этого правила, можно получить неверную запись числа (11101).

Ответ: 111000001.

Для перевода чисел из двоичной системы счисления в восьмеричную необходимо разбить двоичное число на триады справа налево и заменить их на соответствующие восьмеричные числа. Недостающие незначащие нули дописываются.

Пример 2.12. Задание с кратким ответом

Представить число 10011110011001_2 в восьмеричной системе счисления.

Решение. Разобьём двоичную запись числа на триады справа налево. При необходимости слева допишем незначащие нули.

$$\underbrace{010}_{2}\underbrace{011}_{3}\underbrace{110}_{6}\underbrace{011}_{3}\underbrace{001}_{1}{}_2$$

Получим запись восьмеричного числа 23631.

Самая распространённая ошибка при решении подобных задач — разбиение числа на триады слева направо.

Ответ: 23631.

Шестнадцатеричная система счисления

В шестнадцатеричной системе счисления используют 16 цифр (напомним, что количество цифр в позиционной системе счисления равно её основанию). В позиционных системах счисления количественное значение цифры зависит от разряда, в котором она находится, поэтому все цифры должны занимать один разряд. Поскольку десятью цифрами десятичной системы обозначить все цифры шестнадцатеричной системы невозможно, то оставшиеся шесть цифр обозначаются латинскими буквами: **A**, **B**, **C**, **D**, **E**, **F**, которые соответствуют десятичным значениям 10, 11, 12, 13, 14, 15.

Перевод шестнадцатеричного числа в десятичное осуществляется стандартным способом, но при вычислениях все шестнадцатеричные цифры заменяются соответствующим десятичным представлением (вычисления проводятся в новой, десятичной СС):

$$FF1_{16} = 15 \cdot 16^2 + 15 \cdot 16^1 + 1 \cdot 16^0 = 15 \cdot 256 + 15 \cdot 16 + 1 \cdot 1 = 4181_{10}.$$

При переводе десятичных чисел в шестнадцатеричное представление при помощи последовательных делений остатки от 10 до 15 заменяются соответствующими буквами от A до F.

Пример 2.13. Задание с кратким ответом

Перевести число 3556_{10} в шестнадцатеричную систему счисления.

Решение:

Выражение	Частное	Остаток
3556 : 16	222	4
222 : 16	13	14 (E)
13 : 16	0	13 (D)

Выпишем остатки от деления в обратном порядке. Не забудьте, что каждая цифра должна занимать ровно одну позицию в записи числа. Самая распространённая ошибка при решении подобных задач — запись не шестнадцатеричной цифры, а её десятичного значения. Для этого примера запись 13144 неверна. Верной является запись $DE4_{16}$.

Ответ: DE4.

Шестнадцать, как и восемь, является степенью двойки, поэтому существует простой способ взаимного перевода чисел из одной из этих систем в другую. Будем использовать следующую таблицу соответствий чисел:

Основание системы счисления				
2	16		2	16
0000	0		1000	8
0001	1		1001	9
0010	2		1010	A
0011	3		1011	B
0100	4		1100	C
0101	5		1101	D
0110	6		1110	E
0111	7		1111	F

Чтобы перевести число из шестнадцатеричной системы в двоичную, достаточно каждую цифру шестнадцатеричного числа заменить двоичным числом из четырёх разрядов (**тетрадой**). Однозначные, двузначные и трёхзначные числа двоичного числа дополняются незначащими нулями до четырёх разрядов.

Пример 2.14. Задание с кратким ответом

Представить число $\mathbf{F0}_{16}$ в двоичном виде.

Решение. Цифра F_{16} представляется как 1111_2, а цифра 0_{16}, как 0000_2. Получим $F0_{16} = 1111\ 0000_2$.

Ответ: 11110000.

Для перевода целых чисел из двоичной системы в шестнадцатеричную необходимо разбить двоичное число на четвёрки справа налево и заменить их на соответствующие шестнадцатеричные цифры. Недостающие незначащие нули дописываются слева при необходимости.

Пример 2.15. Задание с кратким ответом

Представить число 10111110011001_2 в шестнадцатеричной системе счисления.

Решение. Разобьём двоичную запись числа на тетрады справа налево. При необходимости слева допишем незначащие нули:

$$\underbrace{0010}_{2}\underbrace{1111}_{F}\underbrace{1001}_{9}\underbrace{1001}_{9}{}_2$$

Получим запись шестнадцатеричного числа $2F99_{16}$

Ответ: 2F99.

Связь между двоичной, восьмеричной и шестнадцатеричной системами счисления позволяет «транзитивно» через $СС_2$ переводить числа из $СС_8$ в $СС_{16}$ и наоборот. Например:

1) $56_8 = 101110_2 = 00101110_2 = 2E_{16}$;
2) $1A0_{16} = 000110100000_2 = 110100000_2 = 640_8$.

Номер разряда $СС_2$	3	2	1	0
Вес разряда в $СС_2$	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3 = 2 + 1	0	0	1	1
...	...	...	...	...
12 = 8 + 4	1	1	0	0
13 = 8 + 4 + 1	1	1	0	1
14 = 8 + 4 + 2	1	1	1	0
15 = 8 + 4 + 2 + 1	1	1	1	1

Таблицы соответствий цифр $СС_8$ и $СС_{16}$ с их представлением триадами и тетрадами в $СС_2$ можно не запоминать. Используйте метод последовательного разложения по степеням двойки десятичных чисел от 0 до 15 в двоичную систему счисления, например: $14 = 8 + 4 + 2 = 1110_2$. Если степень двойки вошла в разложение, в соответствующий разряд запишем 1, если нет — 0.

Пример 2.16 *. Задание с выбором одного ответа

Вычислите сумму чисел x и y, если $x = 110111_2$, $y = 135_8$. Результат представьте в двоичной системе счисления.

1) 11010100_2
2) 10100100_2
3) 10010011_2
4) 10010100_2

Решение. Варианты ответа представлены в двоичной системе, поэтому удобнее будет перевести число y в двоичную систему и сложить числа в ней:

$$y = 135_8 = 001011101_2 = 1011101_2.$$

Складываем числа x и y в столбик с использованием таблицы сложения $СС_2$:

$$\begin{array}{r} 110111 \\ +\ 1011101 \\ \hline 10010100 \end{array}.$$

Напомним, что в двоичной системе счисления:

$1 + 1 = 10$;

$1 + 1 + 1 = 11$.

Полученный ответ сравниваем с вариантами ответов.

Ответ: 4.

Пример 2.17 *. Задание с выбором одного ответа

Вычислите сумму чисел x и y, если $x = 110_{16}$, $y = 567_8$. Результат представьте в двоичной системе счисления.

1) 1010000000_2
2) 1010000111_2
3) 110000111_2
4) 1010001000_2

Решение. Переведём оба числа в СС$_2$, представляя каждую цифру первого числа тетрадами, второго числа — триадами:

$$x = 110_{16} = 0001\ 0001\ 0000_2 = 100010000_2;$$
$$y = 567_8 = 101110111_2.$$

Выполним сложение чисел в столбик:

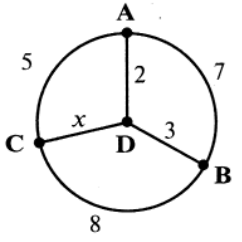
$$\begin{array}{r} 100010000_2 \\ +\ 101110111_2 \\ \hline 1010000111_2 \end{array}.$$

Ответ: 2.

Задания для самостоятельного решения

Задания с выбором одного ответа

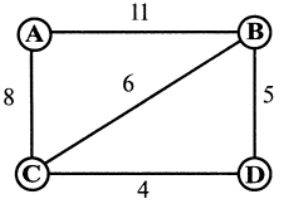
Пример 2.18. На схеме нарисованы дороги между четырьмя населёнными пунктами A, B, C, D и указаны протяжённости данных дорог.



Известно, что кратчайшее расстояние между наиболее удалёнными друг от друга пунктами составляет 7. Определите максимальное значение x, при котором это возможно.

1) 4
2) 5
3) 6
4) 7

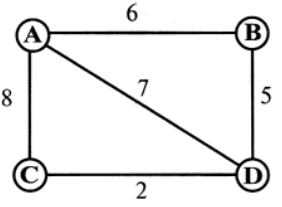
Пример 2.19. На схеме нарисованы дороги между четырьмя населёнными пунктами A, B, C, D и указаны протяжённости данных дорог.



Определите два наиболее удалённых друг от друга пункта (при условии, что передвигаться можно только по указанным на схеме дорогам). В ответе укажите кратчайшее расстояние между этими пунктами.

1) 4
2) 11
3) 12
4) 16

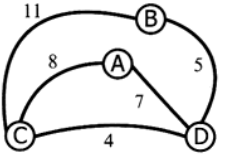
Пример 2.20. На схеме нарисованы дороги между четырьмя населёнными пунктами A, B, C, D и указаны протяжённости данных дорог.



Передвигаться можно только по указанным на схеме дорогам. Кратчайшее расстояние между двумя наиболее удалёнными друг от друга пунктами равно

1) 2
2) 5
3) 7
4) 14

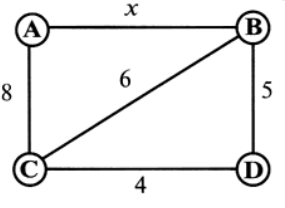
Пример 2.21. На схеме нарисованы дороги между четырьмя населёнными пунктами A, B, C, D и указаны протяжённости данных дорог.



Передвигаться можно только по указанным на схеме дорогам. Кратчайшее расстояние между двумя наиболее удалёнными друг от друга пунктами равно

1) 4
2) 5
3) 12
4) 15

Пример 2.22. На схеме нарисованы дороги между четырьмя населёнными пунктами A, B, C, D и указаны протяжённости данных дорог.



Известно, что кратчайшее расстояние между наиболее удалёнными друг от друга пунктами составляет 10. Определите максимальное значение x, при котором такая ситуация возможна.

1) 4
2) 5
3) 6
4) 7

Пример 2.23. Выберите вариант ответа, в котором объёмы памяти расположены в порядке возрастания.

1) 30 бит, 3 байт, 20 бит, 1 Кбайт, 1040 байт
2) 20 бит, 3 байт, 30 бит, 1 Кбайт, 1040 байт
3) 30 бит, 20 бит, 3 байт, 1040 байт, 1 Кбайт
4) 20 бит, 30 бит, 3 байт, 1 Кбайт, 1040 байт

Пример 2.24. Выберите вариант ответа, в котором объёмы памяти расположены в порядке убывания.

1) 1040 байт, 1 Кбайт, 24 бит, 2 байт, 12 бит
2) 1 Кбайт, 1040 байт, 24 бит, 12 бит, 2 байт
3) 1 Кбайт, 1040 байт, 24 бит, 2 байт, 12 бит
4) 1040 байт, 1 Кбайт, 2 байт, 24 бит, 12 бит

Пример 2.25. Число 1101101101110_2 в восьмеричной системе счисления записывается как

1) 6667_8
2) 66670_8
3) 15556_8
4) 65551_8

Пример 2.26. Число 10110000101_2 в восьмеричной системе счисления записывается как

1) 5064_8
2) 4605_8
3) 2605_8
4) 5411_8

Пример 2.27. Число $1D5_{16}$ в восьмеричной системе счисления записывается как

1) 825_8
2) 13222_8
3) 527_8
4) 725_8

Пример 2.28. Число $2D5_{16}$ в восьмеричной системе счисления записывается как

1) 5521_8
2) 1325_8
3) 1255_8
4) 5231_8

Пример 2.29. Число 25_8 в шестнадцатеричной системе счисления записывается как

1) $A1_{16}$
2) 19_{16}
3) 15_{16}
4) $F9_{16}$

Пример 2.30. Число 73_8 в шестнадцатеричной системе счисления записывается как

1) 311_{16}
2) 143_{16}
3) $3B_{16}$
4) $E3_{16}$

Пример 2.31 *. Вычислите сумму чисел x и y, если $x = 771_{16}$, $y = 1101_8$. Результат представьте в двоичной системе счисления.

1) 100110110101_2
2) 110110110010_2
3) 100100110010_2
4) 100110110010_2

Задания с кратким ответом

Пример 2.32. Количество различных кодовых комбинаций длиной в два символа, которые можно составить, используя алфавит из трёх символов равно _____ .

Пример 2.33. Количество различных последовательностей из символов + и — («плюс» и «минус») длиной ровно в пять символов равно _____ .

Пример 2.34. Количество различных последовательностей длиной в шесть символов из цифр 0 и 1 равно _____ .

Пример 2.35. Сколько различных последовательностей длиной в три символа можно составить из букв Q, W, E, R, T, Y?

Пример 2.36. Световое табло состоит из лампочек, каждая из которых может находиться в двух состояниях («включено» или «выключено»). Какое наименьшее количество лампочек должно находиться на табло, чтобы с его помощью можно было передать 14 различных сигналов?

Пример 2.37. Сколько разных букетов можно составить из трёх цветков, если у вас есть неограниченное количество тюльпанов и нарциссов?

Пример 2.38. Сколько Кбит информации содержит сообщение объёмом 10 Кбайт? В ответе укажите одно число.

Пример 2.39. Сколько Кбит информации содержит сообщение объёмом 2048 бит? В ответе укажите одно число.

Пример 2.40. Сколько Кбайт информации содержит сообщение объёмом 2^{15} байт? В ответе укажите одно число.

Пример 2.41. Сколько байт информации содержит сообщение объёмом 2 Кбит? В ответе укажите одно число.

Пример 2.42. Число 11100101_2 в десятичной системе счисления равно _____ .

Пример 2.43. Число 100000001_2 в десятичной системе счисления равно _____ .

Пример 2.44. Число 1101_2 в десятичной системе счисления равно _____.

Пример 2.45. Число 100000_2 в десятичной системе счисления равно _____ .

Пример 2.46. Число 10001001_2 в десятичной системе счисления равно _____ .

Пример 2.47. Число 255_{10} в двоичной системе счисления записывается как _____ .

Пример 2.48. Число 196_{10} в двоичной системе счисления записывается как _____ .

Пример 2.49. Число 73_{10} в двоичной системе счисления записывается как _____ .

Пример 2.50. Число 64_{10} в двоичной системе счисления записывается как _____ .

Пример 2.51. Число 131_{10} в двоичной системе счисления записывается как _____ .

ГЛАВА 3

КОМПЬЮТЕР КАК УНИВЕРСАЛЬНОЕ УСТРОЙСТВО ОБРАБОТКИ ИНФОРМАЦИИ

Основные компоненты компьютера и их функции

Компьютер — это устройство, предназначенное для автоматического выполнения последовательных действий в соответствии с заложенной программой.

Основные компоненты, обеспечивающие работу компьютера,— это **аппаратное обеспечение (Hardware)** и **программное обеспечение (Software).**

Для описания принципа действия, конфигурации и взаимного соединения основных логических узлов компьютера используется термин **«архитектура»**.

Компьютер включает в себя *четыре основных вида* аппаратных устройств, позволяющих получать, передавать, хранить и обрабатывать информацию:

- устройство обработки и управления **(процессор)**;

- устройство хранения (**внутренняя** и **внешняя память**);
- устройства ввода (**клавиатура, мышь, планшет, сканер**);
- устройства вывода (**дисплей, принтер, плоттер**).

Потоки информации в компьютере

При работе компьютера информация через устройства ввода попадает в память. Процессор извлекает из памяти данные (данные — это информация, представленная в памяти компьютера), обрабатывает их, затем помещает в память результаты обработки. Далее эти результаты сообщаются пользователю через устройства вывода.

Программа — это последовательность команд, предписывающих компьютеру порядок его действий по обработке данных для достижения конкретного результата.

Именно программы управляют аппаратными устройствами компьютера.

Программа — данные, предназначенные для управления конкретными компонентами системы обработки информации в целях реализации определённого алгоритма (ГОСТ 19781-90).

Архитектура фон Неймана

В 1946 году Джон фон Нейман и его коллеги, работавшие над созданием первой в мире ламповой электронно-вычислительной машины (ЭВМ) ENIAC, выдвинули основополагающие принципы построения наиболее эффективной архитектуры компьютера. Сейчас такую архитектуру называют классической, или **архитектурой фон Неймана**. Она предусматривает, что **процессор** состоит из одного *арифметико-логического устройства*, через которое проходит поток данных, и из одного *устройства управления*, через которое проходит поток команд.

Арифметико-логическое устройство (АЛУ) — мозг компьютера, он выполняет логические и арифметические операции над данными, хранящимися в памяти. **Устройство управления (УУ)** управляет и обеспечивает контроль над всеми устройствами, подключёнными к компьютеру, управляет последовательностью выполнения команд в программе.

В классической архитектуре компьютера выделяют два вида памяти — *внутреннюю* и *внешнюю*.

Внутренняя память — это электронное устройство, в котором хранится информация, с которой компьютер работает в данный момент: данные, команды исполняемой программы, управляющие программы. В современных компьютерах внутренняя память пред-

ставлена *оперативным запоминающим устройством* (ОЗУ, другое название — *оперативная память*, ОП), *сверхоперативным запоминающим устройством* (кэш) и *постоянным запоминающим устройством* (ПЗУ). ОЗУ и кэш являются энергозависимыми устройствами, при выключении питания информация из этих видов памяти исчезает.

Наименьший элемент памяти компьютера называется *битом памяти*. В каждом бите памяти в данный момент может храниться одно из двух значений — ноль или единица.

Техническая реализация двоичного кодирования значительно проще, чем, например, десятичного. Такую систему легко реализовать в электронике, так как для неё требуется всего два устойчивых состояния, или два различных сигнала (есть ток — нет тока, намагничен — не намагничен и т. п.). Чем меньше количество состояний у элемента, тем выше помехоустойчивость, надёжность и тем быстрее он может работать. Кроме того, двоичная арифметика является довольно простой. Простыми являются таблицы сложения и умножения — основных действий над числами. Логические операции также легко реализуются в двоичных кодах.

Недостаток двоичного кодирования — длинные коды. Но в технике легче иметь дело с большим количеством простых элементов, чем с небольшим числом сложных.

Использование в компьютере двоичной системы счисления является одним из принципов фон Неймана. Его называют **принципом двоичности**. Данные и программы в памяти компьютера хранятся в виде двоичного кода.

Правило размещения программ и данных в памяти компьютера является другим принципом фон Неймана. Он называется **принципом хранимой программы** или **принципом однородности памяти**.

Внутренняя память компьютера имеет *свойство дискретности*. Она состоит из отдельных частиц — бит. *Второе свойство* внутренней памяти — *адресность*. Все байты (1 байт = 8 бит) внутренней памяти пронумерованы начиная с нуля. Порядковый номер байта называется его **адресом**. **Принцип адресности**, следующий принцип фон Неймана, означает, что запись информации в память и чтение её из памяти производится по адресам.

Исполняемая программа состоит из набора машинных команд, расположенных в ячейках памяти друг за другом. Машинная команда — описание элементарной операции, которую должен выполнить компьютер. Команды хранятся в ячейках памяти в двоичном коде. В общем случае команда содержит: код выполняемой операции; указания по определению операндов (или их адресов); указания по размещению получаемого результата.

В процессоре имеются специальные ячейки памяти — *регистры*. Все регистры имеют имена. В устройстве управления один из регистров — счётчик команд — хранит адрес ячейки выполняемой команды.

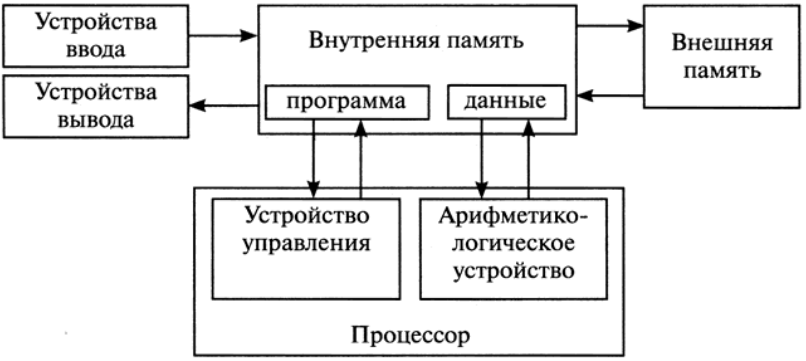
Процесс выполнения команды можно разбить на следующие шаги:

1) Устройство управления по содержимому счётчика команд определяет адрес следующей команды в памяти. Счётчик команд увеличивается и указывает на следующую команду программы.
2) Устройство управления извлекает команду из памяти, определяет тип операции, которую должно выполнить арифметико-логическое устройство (АЛУ), загружает данные, над которыми будет производиться операция (операнды), в регистры АЛУ.
3) Арифметико-логическое устройство выполняет операцию над данными. Результат остаётся в регистрах АЛУ или записывается в память по адресу, указанному в команде.
4) Шаги 1—3 повторяются до завершения выполнения всех команд программы.

Так как и команды, и данные хранятся в двоичных кодах, компьютер не различает, что именно хранится в конкретной ячейке памяти — команда или данные (например, число). Над командами и над данными можно выполнять одинаковые действия. Это означает, что программа в момент выполнения может изменять части самой себя, а некоторые программы могут в результате своего выполнения создавать другие программы.

Именно размещение программы в памяти компьютера обеспечивает автоматический режим выполнения команд. (До этого программы задавались, например, путём установки перемычек на специальной коммутационной панели.) Это позволяет процессору выполнять всю программу в определённой последовательности, названной фон Нейманом **принципом программного управления.**

Архитектура ЭВМ, построенной на принципах фон Неймана



Внешняя память — энергонезависимые устройства для долговременного хранения данных и программ (магнитные и оптические диски, DVD, флэш-память). Отличается от внутренней памяти большим объёмом, меньшей стоимостью в расчёте на один байт хранимой информации, более низкой скоростью доступа к данным.

Устройства ввода/вывода — это любые устройства, которые могут использоваться компьютером для получения и вывода информации. Например, ввести информацию можно с помощью клавиатуры или сканера, а вывести — на принтер или монитор.

Устройство персонального компьютера

Современные ЭВМ бывают разными — от суперкомпьютеров до нетбуков — и используются для разных целей. Самыми распространёнными являются персональные компьютеры (ПК). Они построены по *принципу открытой архитектуры*, разработанному корпорацией IBM и предполагающему:

- наличие общей информационной шины, к которой подключаются дополнительные устройства через разъёмы расширения;
- модульное построение компьютера;
- совместимость всех новых устройств и программных средств с предыдущими версиями.

Принцип открытой архитектуры позволяет пользователю комплектовать нужную ему конфигурацию компьютера и производить при необходимости её модернизацию.

Функциональная схема персонального компьютера

В ПК используются специальные схемы управления работой внешних устройств — контроллеры (адаптеры, платы, карты). Существуют контроллеры дисковода, монитора, принтера, сетевая карта и т. д. Универсальный контроллер позволяет подключать через разъём USB разные виды устройств.

Пересылка данных и программ от одного устройства к другому осуществляется по **системной шине (системной магистрали)**, которая обеспечивает высокую скорость передачи информации.

Шина — это кабель, состоящий из множества проводников. По одной группе проводников — **шине данных** — передаётся обрабатываемая информация, по другой — **шине адреса** — адреса памяти или внешних устройств, к которым обращается процессор. Третья часть магистрали — **шина управления**, по ней передаются управляющие сигналы (например, сигнал к началу работы устройства).

К магистрали могут подключаться дополнительные периферийные устройства, одни модели устройств могут заменяться другими.

Каждое подключаемое устройство получает номер — адрес устройства. Информация от процессора подаётся на контроллер, управляющий работой устройства.

Магистраль, процессор и внутренняя память располагаются на материнской плате компьютера. Помимо основных устройств, материнская плата содержит множество слотов для подключения контроллеров устройств ввода/вывода и других компонентов (звуковая, видео, сетевая карты и т. д.).

Основные характеристики персонального компьютера

Процессор характеризуется тактовой частотой и разрядностью *.

Тактовая частота — это количество тактов процессора в секунду, а **такт** — промежуток времени, за который выполняется элементарная операция (например, сложение). Некоторые операции выполняются процессором за несколько тактов.

Задаётся тактовая частота специальной микросхемой — генератором тактовой частоты, который вырабатывает периодические импульсы. *Тактовая частота* — это число вырабатываемых за секунду импульсов, синхронизирующих работу узлов компьютера. Она измеряется в мегагерцах (МГц) и гигагерцах (ГГц). Именно тактовая частота определяет быстродействие компьютера.

Разрядность процессора — максимальная длина двоичного кода, который может обрабатываться или передаваться процессором целиком. Современные ПК обычно работают с 32-разрядными или 64-разрядными процессорами, существуют процессоры с разрядностью 128 бит.

Для внутренней памяти самой важной характеристикой является её объём. Для работы современных программ требуется оперативная память 128 Мбайт, 256 Мбайт и больше.

Системная шина характеризуется тактовой частотой и разрядностью. Количество одновременно передаваемых по шине бит называется **разрядностью шины**. **Тактовая частота шины** — это частота, с которой происходит обмен данными между процессором и системной шиной компьютера, измеряется в мегагерцах и гигагерцах.

Более подробную информацию о составе персонального компьютера, характеристиках его устройств можно найти в учебниках и других источниках.

Программное обеспечение

Все программы, хранящиеся в памяти компьютера, составляют его **программное обеспечение (ПО)**.

Некоторые программы служат для того, чтобы управлять устройствами компьютера, программами и файлами. Совокупность таких

* Современные процессоры характеризуются также количеством ядер, количеством логических процессов на ядро, уровнями и размером кэша процессора и т. д.

программ компьютера называется **системным программным обеспечением**. Наиболее важной составляющей системного ПО является **операционная система**, обеспечивающая взаимодействие памяти с процессором, поддерживающая диалог с пользователем и управляющая устройствами компьютера. К операционным системам относятся, например, UNIX, MS DOS, MS Windows, MAC OS, Debian и др.

Программы, позволяющие решать задачи пользователя, называются **прикладными программами (прикладным ПО)**. К прикладным программам относятся, например, текстовые и графические редакторы, электронные таблицы, игры и т. д.

Выделяют также **инструментальное программное обеспечение** — программы и среды, при помощи которых программисты разрабатывают другие программы-приложения. К инструментальному программному обеспечению относятся, например, Borland C, Borland Pascal, MS Visual Studio, Java Eclipse, Borland Delphi и др.

Взаимодействие пользователя с компьютером

Для использования компьютера надо не только понимать его возможности, но и знать основные приёмы и правила взаимодействия с ним. Доступные пользователю способы взаимодействия с программами и устройствами компьютера называют *пользовательским интерфейсом*.

Пользовательский интерфейс — совокупность способов организации диалога «человек — компьютер». Он включает возможности задания пользователем команд, например запуска программы на выполнение; виды и способы вывода сообщений компьютера в ответ на команды пользователя; виды сообщений о состоянии устройств и т. д. При взаимодействии с компьютером необходимо строго соблюдать принятые формальные правила.

Вид и особенности пользовательского интерфейса задаёт операционная система.

Интерфейс командной строки

Одним из самых старых является **интерфейс командной строки** (консоль) — разновидность текстового интерфейса. При таком способе взаимодействия пользователь вводит (в основном с клавиатуры) в специальную командную строку текстовые команды, являющиеся инструкциями, понятными операционной системе. Для командных строк характерно наличие **приглашения** — символа в начале строки, указывающего, что система ждёт от пользователя ввода команд. Приглашением, например, может быть символ $ или @. В командной строке Windows приглашением к вводу команды служит мигающий символ нижнего подчёркивания. При помощи команд можно управлять файлами и устройствами, запускать приложения и т. д. Пользователь должен знать команды и их формат. Выводится информация в консоль обычно также в текстовой форме.

Интерфейс командной строки требует мало памяти, но он не является дружественным для пользователя.

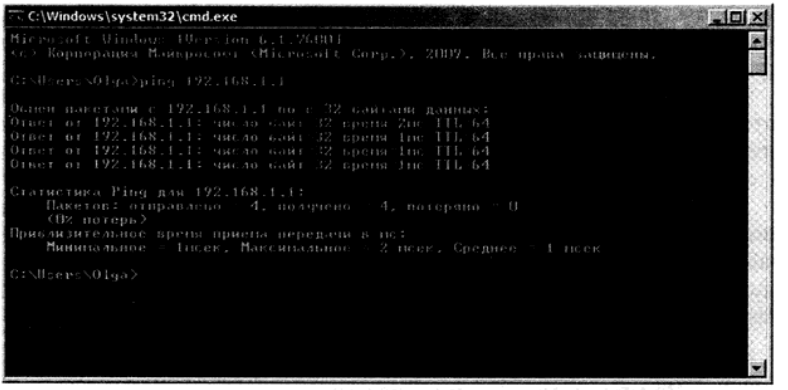


Рис. 2. Вид экрана интерфейса командной строки.

Графический интерфейс

В настоящее время практически все пользователи компьютеров ведут диалог с операционной системой при помощи **графического интерфейса**. Графический интерфейс сделал управление файлами и устройствами проще, увеличив число пользователей и количество областей применения персональных компьютеров. Рассмотрим некоторые элементы оконного интерфейса операционных систем семейства Microsoft Windows.

Окно — это прямоугольная область экрана, в которой выводится определённая информация: содержимое дисков, программы, создаваемые пользователем документы, запросы и сообщения графической операционной системы (рис. 3). Окна можно открывать (разворачивать), закрывать, сворачивать, перемещать, упорядочивать, менять размеры. Открытое окно может занимать целый экран или его часть.

Закрыть окно — значит полностью убрать его с экрана. Закрытие программного окна означает удаление программы из оперативной памяти.

Свёрнутые окна отображаются в виде кнопок в Панели задач. Программа, окно которой свёрнуто, остаётся в оперативной памяти, с ней в любой момент можно возобновить работу. Чтобы вновь раскрыть свёрнутое окно, нужно щёлкнуть на кнопке в Панели задач.

При работе в Windows пользователи сталкиваются с разными типами окон.

1. В **окнах дисков и папок** отображается их содержимое. Любую папку Windows можно открыть в своём окне. С помощью окон папок можно просмотреть всю файловую структуру дисков. В строке заголовка указывается имя папки, ниже располагаются меню и панель инструментов.

Рис. 3. Пример окна MS Windows.

2. **Программные окна (окна приложений)** появляются при работе с загруженными в оперативную память Windows-программами. В окнах приложений открываются окна документов. В приложении может одновременно открываться только один документ или несколько документов. Пример окна приложения приведён в главе 10 «Электронные таблицы».

3. **Диалоговые окна запросов** появляются во время работы с операционной системой и приложениями, располагаясь поверх всех остальных окон на экране. В них содержится запрос информации от пользователя или подтверждение его действий. Окна запросов нельзя изменять в размерах, сворачивать и разворачивать, их можно только закрывать. Диалоговое окно может быть *модальным* или *немодальным*.

3.1. **Модальное окно** блокирует работу приложения. Пользователь должен завершить все операции с этим окном и закрыть его, чтобы вернуться в окно приложения (папки, документа). Различают **три вида модальных окон**:

3.1.1. **Окно диалога** используется для ввода параметров, необходимых для работы с программой.

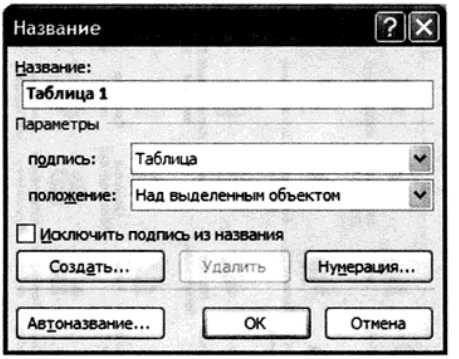


Рис. 4. Окно диалога.

3.1.2. **Окно сообщения** ставит в известность пользователя или о свершившейся операции, или о завершении каких-либо действий.

Рис. 5. Окно сообщения.

3.1.3. **Окно запроса** предоставляет пользователю право совершить действия, в результате которых будет завершена либо продолжена работа программы.

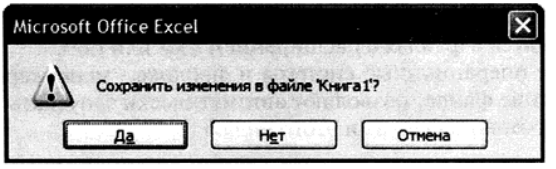


Рис. 6. Окно запроса.

3.2. **Немодальное окно** не останавливает работу приложения. Пользователь может щелчком мыши, не закрывая окна, переходить в окно приложения (документа), работать с ним, ещё одним щелчком возвращаться в диалоговое окно (например, окна справочной системы).

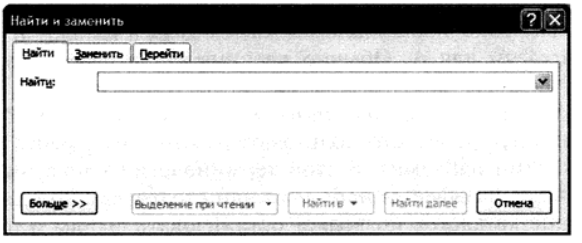


Рис. 7. Немодальное окно.

Файлы и файловая система

Файлы

Файлом называют именованную область на диске или другом носителе информации. В файлах хранятся исполнимые (готовые к запуску) программы, текстовые документы, музыкальные фрагменты, видеозаписи, изображения и множество других данных.

Для того чтобы операционная система могла обращаться к файлам, пользователь мог управлять файлами, а файлы могли взаимодействовать друг с другом, каждому файлу требуется обозначение. Таким обозначением является пара: `<имя>.<расширение>`, например: `save.c3; история.doc; word.exe`.

Последовательность символов, состоящую из букв, цифр, символов подчёркивания и некоторых других символов, допустимых в конкретной операционной системе, называют именем файла. Современные операционные системы, например семейства MS Windows, поддерживают длинные имена файлов, которые могут содержать до 255 символов.

Расширение является необязательной частью обозначения файла. Как правило, расширение представляет собой короткую, от одного до пяти символов, комбинацию букв и цифр, указывающую на содержимое файла или на создавшую его программу. Расширение отделяется от имени файла точкой. Например, программа MS Word позволяет сохранять

созданные в ней документы в файлах с расширением doc; исполнимые файлы хранятся в файлах с расширением exe или com.

Многие операционные системы и файловые менеджеры, опираясь на расширение файла, позволяют автоматически запускать программы, которые работают с файлами этого вида.

Каталоги

Каталог — это область на диске или другом носителе, в которой хранится информация о файлах: имена, размеры, дата и время создания и модификации и т. д. Если каталог хранит имя файла, говорят, что этот файл находится в данном каталоге. В реальности каталоги являются файлами специального вида, т. е. имеют имена, которые могут храниться в других каталогах. Имя каталога задаётся по тем же правилам, что и имя файла. В случае если имя каталога A сохранено в каталоге B, каталог A называют *подкаталогом* каталога B, а каталог B *родительским каталогом* для A. Обычно расширения в названиях каталогов опускают.

Часто каталоги называют *папками*. Термин «папка» был введён с появлением графического пользовательского интерфейса (по аналогии с офисными папками). В этой терминологии папка, находящаяся в другой папке, называется *подпапка* или *вложенная папка*.

Вложение каталогов позволяет организовать файлы и папки в иерархическую древовидную структуру, которую называют *деревом каталогов*. Каталог, который не является подкаталогом ни одного другого каталога, называется *корневым*. Это значит, что этот каталог находится на самом верхнем уровне иерархии всех каталогов.

Каждый диск имеет один главный (корневой) каталог, в котором сохраняются имена каталогов первого уровня. Каталоги, зарегистрированные в каталогах первого уровня, называются каталогами второго уровня и т. д.

Активный каталог, с файлами и папками которого в данный момент работает пользователь, называется *текущим каталогом*. Смена текущего каталога в Windows происходит автоматически при открытии новой папки.

Устройства хранения файлов

Для хранения файлов используются устройства внешней памяти. К ним относятся жёсткие диски, компакт-диски (CD) и цифровые видеодиски (DVD), флэш-память. Дисководы позволяют прочитать информацию со сменных носителей — CD и DVD.

Для того чтобы обратиться к жёсткому диску или послать команду для работы с носителем, нужно указать имя устройства. Традиционно дискам и сменным устройствам хранения присваивают имена, состоящие из заглавных латинских букв и двоеточия: A:, B: и т. д. Имена A:, B: зарезервированы для дисководов гибких дисков, которые в настоящее время уже практически не используются. Имя C: обычно соответствует жёсткому диску, на который установлена операционная система. Такой диск называют *системным*.

Обозначения `C:`, `D:` не всегда соответствуют физически разным устройствам, носитель информации может быть «разбит» на несколько частей — логических разделов, с которыми можно работать как с отдельными дисками. Такие диски называют *логическими*. Например, один жесткий диск может быть разбит на разделы `C:` и `D:`, где `C:` будет системным диском, а `D:` — диском для хранения данных.

Путь к файлу. Полное имя файла

Обратиться к файлу, находящемуся не в текущем каталоге, можно напрямую, используя путь к нему. **Путь к файлу** — это последовательность из имён каталогов или символов «..», разделённых символом «\» (обратный слэш), задающий маршрут от текущего или корневого каталога к тому, в котором находится нужный файл.

Путь, начинающийся с «\», воспринимается компьютером как путь от корневого каталога. Каждое имя каталога, указанное в пути, соответствует входу в каталог с этим именем. Символ «..» обозначает переход в родительский каталог. Путь к файлу от корневого каталога называют *абсолютным путём*, а от текущего — *относительным путём*.

Путь к каталогу файла и имя файла, разделённые «\», перед которыми указано имя диска, представляет собой **полное имя файла**. Если имя диска опущено, подразумевается текущий дисковод.

При решении задач, связанных с переходом по дереву каталога, воспользуемся следующим фрагментом дерева:

Файлы `program.pas`, находящиеся в каталогах `Документы` и `Школа`,— это разные файлы, они могут содержать разную информацию.

Пример полного имени файла: `C:\Я\Школа\program.pas`. Относительный путь к этому файлу из каталога `Документы` будет таким: `..\Я\Школа\program.pas`.

Пример 3.1. Задание с выбором одного ответа

Пользователь работал в каталоге C:\Я\Школа (см. фрагмент дерева на с. 47). Он поднялся на два уровня вверх. Укажите каталоги, находящиеся на этом уровне.

1) Школа, Photo, Документы
2) 2009
3) Мама, Я
4) C:, D:

Решение. Пользователь находился внутри каталога Школа. Поднявшись на один каталог, он оказался на уровне каталогов второго уровня в папке Я. Поднявшись ещё на один каталог, он оказался на уровне каталогов первого уровня. Значит, ему видны все каталоги, лежащие в корневом каталоге,— Я и Мама.

Ответ: 3.

Пример 3.2. Задание с выбором одного ответа

Пользователь работал с каталогом C:\Мама\Photo\2009\ (см. фрагмент дерева на с. 47). Он поднялся на два уровня вверх. Укажите полные имена каталогов, находящихся в каталоге, в котором оказался пользователь.

1) C:\Мама\Photo\2009
2) Photo, Документы
3) C:\Мама\Photo, C:\Мама\Документы, C:\Я\Школа
4) C:\Мама\Photo, C:\Мама\Документы

Решение. Поднявшись на один уровень, пользователь оказался в каталоге Photo, поднявшись ещё на один уровень, пользователь оказался в каталоге Мама, в котором находятся каталоги Photo и Документы.

Ответ: 4.

Файловая система

Набор правил, определяющих, как хранится и как организована информация о файлах и дисках, называется **файловой системой**.

Для использования дисков, записанных с помощью некоторой файловой системы, операционная система или специальная программа (файловый менеджер) должна поддерживать эту файловую систему. Существуют различные файловые системы, например FAT и NTFS.

Операционные системы позволяют работать с файлами, т. е. выполнять над ними некоторые операции, например:

- создание файла/каталога;
- уничтожение файла/каталога;
- открытие файла/каталога;
- закрытие файла/каталога;
- чтение файла/каталога;
- запись в файл/каталог;
- дополнение файла/каталога;
- поиск в файле/каталоге;

- переименование файла/каталога;
- архивирование файла/каталога;
- разархивирование файла/каталога и т. д.

Для упрощения работы с файлами существуют специальные программы — *файловые менеджеры*. К ним относятся Norton Commander, FAR, Total Commander и др. Операционные системы с графическим интерфейсом позволяют работать с файлами и папками не только в файловых менеджерах, но и в окнах.

Поиск файлов

Поиск как одного, так и множества файлов или каталогов может быть осуществлён при помощи файловых менеджеров или специальных средств системы.

Зная имя и расширение файла, можно найти все файлы с таким именем и расширением на диске или дисках компьютера.

В случае, когда требуется выборка файлов, имена которых отвечают некоторому условию, или когда имя файла полностью неизвестно, в поисковых запросах используются **шаблоны**, или **маски имён файлов**.

В масках, кроме символов, которые могут быть в имени файла, используются специальные символы:

***** (звёздочка) — обозначает любое количество (в том числе и ноль) произвольных символов в имени или расширении файла;

? (вопросительный знак) — обозначает ровно один произвольный символ в имени или расширении файла.

Например, поисковому запросу `*.tiff` соответствуют файлы с любым именем, имеющие расширение `tiff`, а запросу `?.tiff` соответствуют файлы с именем из одного символа и расширением `tiff`. По запросу `wind.b*` могут быть найдены файлы `wind.bmp`, `wind.bat`, `wind.bas`, `wind.b`.

Пример 3.3. Задание с выбором одного ответа

Для групповых операций с файлами используются маски имён файлов. Маска представляет собой последовательность букв, цифр и прочих допустимых в именах файлов символов, в которых также могут встречаться символы «?» и «*» ,в том числе «*» может задавать и пустую последовательность. Определите, какое из указанных имён файлов не удовлетворяет маске `??pri*.?*`.

1) `caprika.wow`
2) `weprik.cpp`
3) `otopri.c`
4) `reprint.be`

Решение. Проверим, все ли из указанных имён файлов удовлетворяют первой части маски `??pri`. Перед `pri` в имени должно находиться ровно два символа. Имя файла `otopri.c` не удовлетворяет условию. Это ответ. Чтобы убедиться, что не допустили ошибки, проверим соответствие имён оставшихся файлов маске. Между `pri` и точкой, отделяющей имя от расширения файла, может находиться любое количество символов. Все имена файлов удовлетворяют

этому условию. Расширение файла должно состоять хотя бы из одной буквы. И это условие выполняется для всех имён файлов.

Ответ: 3.

Пример 3.4. Задание с кратким ответом

Пользователь работал с каталогом `D:\Program\Basic\Circle`. Сначала он поднялся на один уровень выше, затем спустился в каталог `Condition`. Запишите полный путь к каталогу, в котором оказался пользователь.

1) `D:\Program\Basic\Circle\Condition`
2) `D:\Program\Condition`
3) `D:\Program\Basic\Condition`
4) `D:\Program\Basic`

Решение. Поднявшись на один уровень, пользователь оказался в каталоге `D:\Program\Basic\`. После спуска на один уровень, он попал в каталог `Condition`, этот каталог является подкаталогом каталога `D:\Program\Basic\`, и полный путь к нему описывается: `D:\Program\Basic\Condition`, т. е. ответ — 3.

Ответ: 3.

Архивирование информации

Архивирование — это процедура перекодирования данных, производимая с целью уменьшения их объёма. В процессе архивации информация шифруется при помощи специальных алгоритмов, что и приводит к сокращению её информационного объёма. Архивирование требуется, например, для обеспечения долгосрочного сохранения информации, ускорения пересылки информации по каналам связи и т. п. Процедура восстановления данных из архивов называется **разархивированием** или **извлечением**.

Для архивации файлов и папок используются специальные программы — **архиваторы**, например, WinRAR, 7-Zip, QuickZip и проч.

Задания для самостоятельного решения

Задания с выбором одного ответа

Пример 3.5. Каталог содержит файлы:

а) `z1.pas`;
б) `z21.ppt`;
в) `z4.p`;
г) `z33.p`;
д) `zad.pas`;
е) `zom.pp`.

При выделении файлов с использованием маски `z??.p*` список всех выделенных файлов

1) а, б, д
2) а, б, д, е
3) б, г, д, е
4) б, д, е
5) в, г, е

Пример 3.6. Каталог содержит файлы:

а) z1.pas;
б) z21.ppt;
в) z4.p;
г) z33.p;
д) zad.pas;
е) zom.pp.

При выделении файлов с использованием маски z*.p?? список всех выделенных файлов

1) а, б, д
2) а, б, д, е
3) б, г, д, е
4) б, д
5) в, г, е

Пример 3.7. Определите, какое из указанных имён файлов НЕ удовлетворяет маске ?*oo*.*t.

1) bdfoo.net
2) nmfoot.ppt
3) saloon.dat
4) door.etc

Пример 3.8. Определите, какое из указанных имён файлов НЕ удовлетворяет маске *un?*.d?*.

1) funds.da
2) uncle.td
3) round.doc
4) fortune.dpi

Пример 3.9. Определите, какое из указанных имён файлов удовлетворяет маске re*t?.?ft.

1) repeat.tft
2) premty.oft
3) renty.rft
4) reprint.eft

Пример 3.10. Определите, какое из указанных имён файлов удовлетворяет маске *ad??e*q?.*y.

1) pleadweqr.why
2) adilenqu.by
3) radeprix.fly
4) scadnerq.yt

Пример 3.11. Определите, по какой из масок будет выбрана указанная группа файлов: cake.doc, paste.txt, page.pas, name.bat.

1) *a*e.?
2) a?e*.??
3) ?a*e.??*
4) ?a?e.*

Пример 3.12. Пользователь работал с каталогом D:\Программы\Кодеки\xDiv. Сначала он поднялся на два уровня вверх, затем спустился в каталог Игры и после этого спустился в каталог Квесты. Запишите полный путь к каталогу, в котором оказался пользователь.

1) D:\Программы\Кодеки\Игры\Квесты
2) D:\Программы\Игры\Квесты
3) D:\Программы\Игры\Кодеки\Квесты
4) D:\Игры\Квесты

Пример 3.13. Пользователь работал с каталогом D:\Program. Сначала он спустился в каталог Pascal, затем спустился в каталог Array, после чего поднялся на один уровень и спустился в каталог String.

Запишите полный путь к каталогу, в котором оказался пользователь.

1) D:\Program\String
2) D:\Program\Pascal\String
3) D:\Program\Pascal\Array\String
4) D:\String

Пример 3.14. Пользователь работал с каталогом C:\Авиация. Сначала он переместился в корень другого локального диска, затем спустился в каталог Вертолёты. После этого пользователь спустился на два уровня вниз. Запишите полный путь к каталогу, в котором мог оказаться пользователь.

1) E:\Вертолёты\Двухвинтовые\ЯК-40
2) D:\Вертолёты\КА-50
3) C:\Авиация\ Вертолёты\МИ-26\Фото
4) A:\Авиация\Вертолёты\Винтокрылы

Пример 3.15. Файл Иванов.doc был расположен в некотором каталоге на локальном диске C. В этом каталоге пользователь создал два подкаталога и переместил файл в один из них. Полное имя файла после перемещения стало C:\Кадры\Резюме\Иванов.doc. Запишите полный путь к файлу Иванов.doc до перемещения.

1) C:\Кадры\Резюме\Иванов
2) C:\Кадры\Заявления\Иванов.doc
3) C:\Кадры\Иванов.doc
4) C:\Кадры\Иванов

Пример 3.16. Файл Россия.mp3 был расположен в некотором каталоге на локальном диске D. В этом каталоге пользователь создал подкаталог Гимны и переместил в него файл. Полное имя файла после перемещения стало D:\Чемпионат\Гимны\Россия.mp3. Запишите полный путь к файлу Россия.mp3 до перемещения.

1) D:\Чемпионат\Гимны
2) D:\Чемпионат
3) D:\Россия.mp3
4) D:\Чемпионат\Россия.mp3

ГЛАВА 4

ПЕРЕДАЧА ИНФОРМАЦИИ. КОДИРОВАНИЕ ИНФОРМАЦИИ

Рассмотрим подробнее один из информационных процессов — процесс **передачи информации**.

Процесс передачи информации

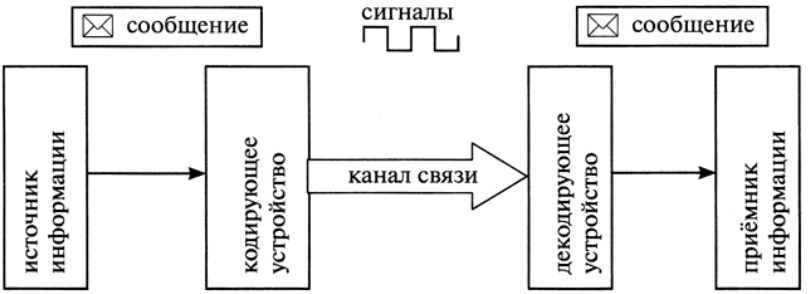
Информация передаётся в виде **информационных сообщений** (для краткости будем называть их **сообщениями**) от **источника** к **приёмнику** по **каналам связи**.

Источниками и приёмниками информации могут быть люди, животные, технические устройства (например, компьютеры). В устной и письменной речи сообщениями могут быть слова, фразы, условные знаки, изображения; каналами связи могут быть, например, звуковые и световые волны. Важно, чтобы и отправитель, и получатель информации понимали, что означает то или иное сообщение (знак, жест, звук).

Первой технической системой передачи информации был телеграф. Затем появились телефон, радио, телевидение, Интернет. Все эти виды связи основаны на передаче физического (электрического или электромагнитного) сигнала.

В начале XX века возникла **теория связи**, математический аппарат которой был разработан американским учёным Клодом Шенноном. Он предложил модель процесса передачи информации по техническим каналам связи (см. схему).

Схема передачи информации



Источник посылает сообщение, которое кодируется в передаваемый сигнал. **Кодирующее устройство** преобразует исходное сообщение источника информации в форму, пригодную для передачи по каналу связи (сигнал). Полученный сигнал декодируется и поступает в виде сообщения приёмнику информации. На канал связи действуют помехи, поэтому при передаче информации необходимо принять меры для защиты от помех.

Сигнал — это изменение некоторой физической величины во времени. Характеристика сигнала, которая используется для представления сообщения, называется **параметром сигнала**. Сигналы могут быть *непрерывными* и *дискретными*. Если параметр сигнала может принимать лишь конечное число значений и существует в конечном числе моментов времени, сигнал называется **дискретным**. В цифровой вычислительной технике используются дискретные сигналы, которые могут принимать два значения, например уровень напряжения 0 вольт и 3,3 вольт, они условно обозначаются 0 и 1.

В качестве примера передачи информации по техническим каналам связи рассмотрим передачу SMS-сообщения по сотовой связи. Сообщение в форме набранного текста преобразуется в последовательность электромагнитных сигналов, которые передаются по каналам связи и затем в телефоне-приёмнике декодируются в текст *.

Компьютеры обмениваются информацией по каналам связи различной физической природы: кабельным, оптоволоконным, радиоканалам и т. д. Кабельные каналы обычно используют внутри зданий, радиоканалы — в пределах прямой видимости, оптоволоконные каналы могут иметь протяжённость до нескольких тысяч километров. При использовании телефонных линий в компьютерных сетях функции кодирования/декодирования выполняет устройство, которое называется *модемом*.

Основная характеристика канала передачи — **пропускная способность**, или **скорость передачи информации**. Она определяется объёмом информации, который может быть передан по каналу в единицу времени. Пропускная способность канала измеряется в битах в секунду (бит/с), в килобитах в секунду (Кбит/с) и в других кратных единицах. Иногда используют единицу измерения байт в секунду (байт/с) и кратные ему единицы (Кбайт/с, Мбайт/с и т. д.).

Напомним, что префикс **кило** в информатике соответствует множителю $2^{10} = 1024$ и 1 Кбит = 1024 бит.

Пример 4.1. Задание с кратким ответом

Скорость передачи данных через ADSL-соединение ** равна 1 024 000 бит/с. Передача файла через данное соединение заняла 5 с. Определите размер файла в килобайтах.

Решение. Задачи такого вида очень похожи на задачи, которые вы решали на уроках физики и алгебры. В задаче фигурируют аналоги

---

* Текстовая информация в техническом устройстве тоже кодируется для хранения и вывода на экран, как будет показано ниже в этой главе.

** Передача данных по технологии ADSL реализуется через обычную аналоговую телефонную линию при помощи абонентского устройства — модема ADSL (англ. asymmetric digital subscriber line — асимметричная цифровая абонентская линия). Пропускная способность в сетях ADSL до 24 Мбит/с. Скорость получения информации из сети значительно выше скорости передачи информации в сеть.

известных величин скорости, времени и расстояния. Введём обозначения:

v — количество бит, которое может быть передано за секунду;

t — время передачи;

V — размер передаваемого файла.

Эти величины связаны известным из физики и алгебры соотношением

$$V = v \cdot t.$$

Важно!

При выполнении вычислений необходимо учитывать единицы измерения. Все вычисления рекомендуем выполнять, выделяя степени двойки.

Итак, для определения размера переданного файла необходимо умножить скорость передачи информации на время, в течение которого происходила передача:

$1\,024\,000$ бит/с $\cdot$ 5 с $= (2^{10} \cdot 10^3 \cdot 5)$ бит $= 2^{10} \cdot (2^3 \cdot 5^4)$ бит $= 2^{13} \cdot 5^4$ бит.

Переведём биты в килобайты, для этого разделим полученный результат на 2^{13}:

$$(2^{13} \cdot 5^4) : 2^{13} = 5^4 = 625 \text{ Кбайт}.$$

Ответ: 625.

Шифрование информации

Во все времена наиболее уязвимым местом при передаче секретной информации был канал связи. Поэтому ещё древние люди начали изобретать способы **шифрования** для защиты информации. При шифровании сообщение изменяется способом, известным источнику и приёмнику, и передаётся по каналу связи. Однако узнать истинный смысл сообщения могут только те приёмники, которые знают, как именно, по каким правилам источник исказил сообщение. Способы шифрования информации изучает **криптология**.

В настоящее время известно большое количество самых разных шифров, отличающихся сложностью, стойкостью к дешифрованию и т. д.

Среди древнейших шифров одним из самых известных является *шифр Цезаря*. Для того чтобы закодировать сообщение шифром Цезаря, необходимо произвести замену буквы алфавита на другую, отстоящую от неё, на данное количество букв. Длина сдвига называется *ключом* шифра Цезаря. На с. 56 представлен пример шифра Цезаря с ключом 3. В нижней строке показан исходный алфавит, в верхней — алфавит для зашифрованного сообщения. При кодировании и передаче исходного сообщения «ABS» к получателю поступит сообщение «DEV».

Пример шифра Цезаря

X Y Z A B C ... U V W X Y Z
A B C D E F ... X Y Z A B C

Восстановление исходного сообщения из полученного называется **дешифровкой**. Для восстановления исходного сообщения надо применить шифр Цезаря с ключом −3 к полученному сообщению.

Термины «шифрование» и «кодирование» близки по смыслу. В информатике принято их различать в зависимости от поставленных целей. Шифрование используется для обеспечения информационной безопасности при передаче сообщений, ключ шифрования держится в тайне. Кодирование используется для обеспечения представления одной и той же информации разными техническими средствами или в разных форматах. При кодировании информация переводится из одной формы представления в другую по некоторому известному алгоритму (методу).

Кодирование информации

Решать задачу кодирования информации человечество начало задолго до появления компьютеров: великие достижения человечества — письменность и арифметика — не что иное, как системы кодирования речи и числовой информации. Задачи кодирования решались и для передачи информации с помощью технических устройств.

Написанное на бумаге предложение невозможно передать по телеграфу в виде букв, цифр и символов. Устройство телеграфной установки таково, что оно позволяет лишь посылать или не посылать сигнал, причём сигнал может иметь разную длительность. Таким образом, привычные для нас буквы должны быть закодированы при помощи алфавита телеграфа. Напомним, что алфавитом называется упорядоченный набор знаков, при помощи которых можно составлять слова.

Каждый символ исходного алфавита при кодировании представляется последовательностью символов кодового алфавита, которая называется **кодовым словом**. Иногда кодовое слово называют кратко **кодом**.

Кодовый алфавит может быть, например, двоичным, т. е. состоять из двух знаков — 0 и 1.

Азбука Морзе

Для первой телеграфной машины Самуэля Морзе использовался именно такой способ знакового кодирования.

1) В **исходный алфавит** входили буквы латинского алфавита, цифры и знаки препинания.
2) **Кодовый алфавит Морзе** состоял из трёх символов:
   тире — длинный сигнал,
   точка — короткий сигнал,
   пауза — отсутствие сигнала.

3) Каждый символ (знак) исходного алфавита Морзе обозначил уникальной комбинацией из длинных и коротких сигналов — кодовым словом.

Кодовые слова однозначно определяли каждый символ исходного алфавита. Впоследствии к латинскому алфавиту добавились шифры для знаков национальных алфавитов, например русского.

Принцип кодирования азбуки Морзе исходит из того, что буквы, которые чаще употребляются в английском языке, кодируются более короткими сочетаниями точек и тире. Это делает передачи компактнее. Такие коды называются **неравномерными**.

Примеры кодов Морзе некоторых символов:

Символ исходного алфавита	Кодовое слово	Символ исходного алфавита	Кодовое слово	Символ исходного алфавита	Кодовое слово
A	• —	I	• •	J	• — — —
M	— —	L	• — • •	W	• — —

Заметим, что началом кодовых слов символов J, L, W является кодовое слово символа A. Поэтому невозможно однозначно декодировать полученное сообщение, если не использовать паузы между кодовыми словами. Например, требуется расшифровать сообщение, закодированное азбукой Морзе и переданное без пауз между кодами символов (используются только приведённые в таблице выше кодовые слова):

• — — — • — • •

Для декодирования сообщения будем последовательно слева направо выделять коды символов. Получим варианты декодирования: AMAI (• — — — • — • •), AML (• — — — • — • •), AI (• — — — • — • •), JL (• — — — • — • •).

Для однозначного декодирования сообщения, закодированного азбукой Морзе, используют паузы, разделяющие кодовые слова. За единицу времени принимается длительность одной точки. Длительность тире равна длительности трёх точек. Пауза между элементами одного знака равна длительности одной точки, между знаками в слове — длительности трёх точек, между словами — длительности семи точек.

Без пауз между символами по телеграфу подаётся только всемирно известный сигнал **SOS**. Символ S обозначается тремя точками, символ O — тремя тире:

SOS • • • — — — • • •.

Равномерные коды

Для того чтобы однозначно прочитать сообщение, не разделяя коды символов специальными знаками, можно использовать **равномерные коды**. В этом случае кодовые слова всех символов исходного алфавита имеют одинаковую длину. Расшифровать такое сообщение не составит труда, но сообщение становится длиннее, чем при использовании

неравномерных кодов. Например, для кодирования 26 символов латинского алфавита двоичными числами потребуется выделить по 5 бит на один символ, тогда можно закодировать $2^5 = 32$ символа. Если код символа A будет иметь вид 00000, символа B — 00001, символа S — 10010, то сообщение ABS выглядит как последовательность 000000000110010. Для декодирования достаточно разделить последовательность на группы по пять двоичных цифр.

Немецкий изобретатель Бодо предложил код, в котором, как и в азбуке Морзе, использовались сигналы двух типов, но не использовались паузы. В отличие от азбуки Морзе все символы кодировались последовательностью из одинакового количества сигналов, т. е. код Бодо был равномерным. Использование кода Бодо позволило сделать телеграф массовым средством передачи сообщений.

Префиксные коды

Для того чтобы можно было однозначно декодировать сообщение, закодированное неравномерным кодом, без разделения кодов символов, используют так называемые *префиксные коды*.

Префиксный код — это код со словами переменной длины, в котором ни одно кодовое слово не является началом другого кодового слова.

Азбука Морзе — пример непрефиксного кода.

Пример префиксного кода: 0, 10, 11. Сообщение 100111011010 однозначно декодируется: 10 0 11 10 11 0 10.

Набор 0, 10, 100, 11 образует непрефиксный код. Приведённое выше сообщение можно декодировать несколькими способами:

10 0 11 10 11 0 10;
100 11 10 11 0 10.

Сообщения, закодированные префиксными кодами, можно декодировать «на лету», не дожидаясь получения всего сообщения целиком. Префиксные коды используются для кодирования аудио- и видеофайлов, поэтому можно слушать музыку или смотреть видео до того, как файл загрузится целиком.

Пример 4.2. Задание с выбором одного ответа

От разведчика была получена следующая шифрованная радиограмма, переданная с использованием азбуки Морзе:

— — • — • — • • — • — — — —

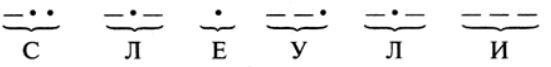
При передаче радиограммы было потеряно разбиение на буквы, но известно, что в радиограмме использовались только следующие буквы:

Е	Л	И	С	У
•	— • —	— — —	— — •	— • •

Определите текст радиограммы. В ответе укажите, сколько букв было в исходной радиограмме.

1) 6
2) 7
3) 8
4) 9

Решение. Заметим, что коды использованных в сообщении символов образуют префиксный код. Это значит, что сообщение однозначно декодируется. Будем последовательно слева направо определять подходящие символы в полученном коде:



В радиограмме содержалось 6 букв.

Ответ: 1.

Кодирование информации в компьютере

Процессор берёт команды программ и данные для обработки из памяти. Память является электронным устройством и состоит из микросхем, которые, в свою очередь, состоят из тысяч более мелких электронных компонентов. Подобные электронные компоненты могут находиться только в двух состояниях — «включено» или «выключено», что соответствует двум цифрам двоичной системы счисления 1 или 0 или одному биту.

Таким образом, любая информация в памяти компьютера представляется в виде последовательности битов, каждый из которых находится в одном из допустимых состояний.

Кодирование текстовой информации

При использовании одного бита можно представить в памяти компьютера только два различных символа. Одному из них будет сопоставлен двоичный код — ноль, а второму — единица.

Если мы увеличим длину кодовой комбинации символа до двух цифр, то получим следующие коды: 00, 01, 10, 11. Таким образом, в памяти компьютера можно будет представить четыре различных символа. При последовательном наращивании длины двоичной кодовой комбинации увеличивается количество символов, которые могут быть закодированы. Кодом длиной в три символа представляются 8 различных символов (000, 001, 010, 011, 100, 101, 110, 111) и т. д.

При длине кодовой комбинации L количество кодовых комбинаций K определяется по формуле:

$$K = 2^L,$$

где 2 — количество символов кода.

Текстовая информация состоит из букв, цифр, знаков препинания, специальных символов, таких, как пробел, символ перевода строки и др. Для кодирования текстовой информации в компьютере используются равномерные коды. В случае, когда код каждого символа занимает в памяти компьютера 1 байт, или 8 бит, общее количество символов, которые можно закодировать, равно $2^8 = 256$. Если кодовое слово состоит из двух байтов, можно закодировать $2^{16} = 65\,536$ символов.

Двоичные кодовые комбинации символов можно считать своеобразным шифром. Только зная соответствие двоичных комбинаций

и символов, можно понять сообщение, закодированное с помощью двоичных последовательностей. В компьютерах используются специальные **кодовые таблицы**, которые ставят в соответствие двоичные последовательности (коды) и символы.

Кодовые таблицы символов

Существуют стандартные таблицы кодов. Они могут использовать один или два байта для кодирования одного символа.

Широко используется таблица кодов, известная как **стандарт ASCII** (**American Standart Code for Information Interchange** — Американский стандартный код для обмена информацией), использующая один байт для кодирования одного символа. ASCII представляет собой кодировку для представления десятичных цифр, символов латинского и национального алфавитов, знаков препинания, символов арифметических операций и управляющих символов. Управляющие символы называют непечатаемыми символами, к ним относятся такие, как «перевод строки» (код символа 10), «возврат каретки» (код 13) и др.

Первая половина кодовой таблицы содержит **стандартные** символы ASCII (символы с кодами 0 — 127), они одинаковые во всех странах. В таблице приведены печатаемые символы первой части кодовой таблицы ASCII.

Первая часть кодовой таблицы ASCII

Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ
20	пробел	30	0	40	@	50	P	60	'	70	p
21	!	31	1	41	A	51	Q	61	a	71	q
22	«	32	2	42	B	52	R	62	b	72	r
23	#	33	3	43	C	53	S	63	c	73	s
24	$	34	4	44	D	54	T	64	d	74	t
25	%	35	5	45	E	55	U	65	e	75	u
26	&	36	6	46	F	56	V	66	f	76	v
27	'	37	7	47	G	57	W	67	g	77	w
28	(	38	8	48	H	58	X	68	h	78	x
29	)	39	9	49	I	59	Y	69	i	79	y
2A	*	3A	:	4A	J	5A	Z	6A	j	7A	z
2B	+	3B	;	4B	K	5B	[	6B	k	7B	{
2C	,	3C	<	4C	L	5C	\	6C	l	7C	|
2D	-	3D	=	4D	M	5D	]	6D	m	7D	}
2E	.	3E	>	4E	N	5E	^	6E	n	7E	~
2F	/	3F	?	4F	O	5F	_	6F	o	7F	DEL

Коды в таблице записаны в шестнадцатеричной системе счисления, как принято в информатике. Код символа A, например, $41_{16} = 65_{10}$. Таблицу кодов не надо запоминать, но следует помнить последовательность символов:

1) знаки препинания и арифметических операций;
2) цифры от 0 до 9;
3) прописные символы латинского алфавита;
4) строчные символы латинского алфавита.

Вторая часть кодовой таблицы (символы с кодами 128 — 255) называют **расширенными кодами** ASCII. В расширенные коды ASCII включают символы национальных алфавитов, например символы кириллицы. Но даже с учётом этих дополнительных знаков алфавиты многих языков не удаётся охватить при помощи 256 знаков. По этой причине существуют различные варианты кодировки ASCII, включающие символы разных языков.

Отсутствие согласованных стандартов привело к появлению различных кодовых таблиц (вернее, различных вторых частей кодовых таблиц) для кодирования символов кириллицы, среди которых

- международный стандарт ISO 8859;
- кодовая таблица фирмы Microsoft CP-1251 (кодировка Windows);
- кодовая таблица, применяемая в ОС Unix KOI8R и др.

По этой причине тексты на русском языке, набранные с использованием одной кодовой таблицы, невозможно прочитать при использовании другой кодовой таблицы.

В настоящее время в компьютерах широко применяется стандарт кодирования **Unicode (Юникод)**, в котором для кодирования одного символа отводятся один байт, два байта или четыре байта. Первые 128 символов Юникода совпадают с символами ASCII. Остальная часть кодовой таблицы включает символы, используемые в основных языках мира.

Пример 4.3. Задание с выбором одного ответа

В кодировке Unicode каждый символ кодируется двумя байтами. Определите информационный объём слова из 24 символов в этой кодировке:

1) 384 бит
2) 192 бит
3) 256 бит
4) 48 бит

Решение. Если на каждый символ отводится 2 байт, а слово состоит из 24 символов, то для кодирования этого слова необходимо $2 \cdot 24 = 48$ байт. Переведём 48 байт в биты: $48 \cdot 8 = 384$ бит.

Ответ: 1.

Кодирование графической информации

Изображение на экране монитора формируется набором экранных точек — **пикселей**. Каждая экранная точка имеет свой цвет. Картинка на экране — это отображение информации из памяти компьютера.

Первые мониторы были монохромными. Точка на экране монохромного монитора может быть только светлой (белой) или тёмной (чёрной). Для кодирования цвета пикселя используется один бит памяти, значение 1 соответствует белому цвету, 0 — чёрному. Подобные экраны используются в недорогих сотовых телефонах, системах видеонаблюдения и других устройствах.

Каждый пиксель современного дисплея определяется компонентами трёх основных цветов: красного (Red, R), зелёного (Green, G) и синего (Blue, B). В памяти необходимо сохранять информацию о состоянии каждой точки изображения, т. е. о состоянии каждой из её трёх составляющих. Управление яркостью каждой составляющей позволяет влиять на цвет экранной точки.

Цветовой моделью называется правило представления цвета в виде наборов чисел (обычно трёх-четырёх). В компьютерной графике используется несколько видов цветовых моделей.

Рассмотрим цветовую модель, связанную с представлением пикселя составляющими красного, зелёного и синего цветов. Она называется **RGB (Red-Green-Blue)-моделью**.

Цветовая модель RGB

В RGB-модели происходит сложение цветов и добавление их к чёрному цвету экрана, поэтому она называется *аддитивной* (additive). Разные цвета образуются смешиванием трёх основных цветов в разных пропорциях, т. е. с разными яркостями.

Глубина цвета (color depth) — это число бит, используемых для представления каждого пикселя изображения.

В модели RGB каждый цвет может кодироваться тремя байтами (режим **TrueColor**). Каждый байт отвечает за яркость красной, зеленой и синей составляющей пикселя соответственно. Таким образом, глубина цвета в режиме TrueColor составляет 24 бита. Изображения, пиксели которых закодированы таким способом, называются 24-битными изображениями.

Чтобы указать цвет пикселя в модели RGB, достаточно перечислить разделённые точками яркости каждой составляющей, например: 255.255.0 — код жёлтой точки, записанный при помощи десятичных кодов яркостей. Значения яркости варьируются от 0 («выключено») до 255 («включено на максимум»). Если значения яркостей всех трёх составляющих равны, получим оттенки серого цвета.

> В языке разметки HTML для обозначения цвета используется шестнадцатеричная запись вида "#XXXXXX". Каждый цвет записывается в виде двух шестнадцатеричных цифр без пробелов. Первая пара цифр — интенсивность красного цвета, вторая — зелёного, последняя — синего. Напомним, что $FF_{16} = 1111\ 1111_2 = 255$. Например, белый цвет обозначается "#FFFFFF".

Значения некоторых цветов в числовой модели RGB

Цвет	R	G	B	R	G	B
	Десятичная запись			Шестнадцатеричная запись		
Красный (red)	255	0	0	FF	00	00
Зелёный (green)	0	255	0	0	FF	0
Синий (blue)	0	0	255	00	00	FF
Фуксин (magenta)	255	0	255	FF	00	FF
Голубой (cyan)	0	255	255	00	FF	FF
Жёлтый (yellow)	255	255	0	FF	FF	00
Белый (white)	255	255	255	FF	FF	FF
Чёрный (black)	0	0	0	00	00	00

На рис. 8 представлена модель RGB. Отметим, что на рисунке не показан чёрный фон.

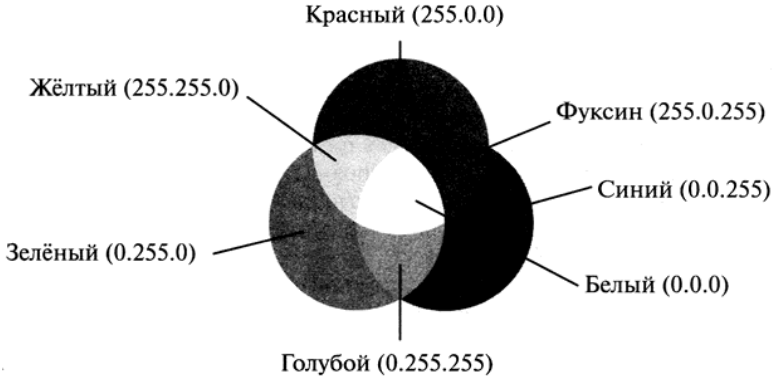


Рис. 8. Цветовая модель RGB.

Если изменять интенсивность каждого цвета для смешанных цветов, например задать цвет 127.127.0, то мы получим на экране болотный цвет, а не более тёмный оттенок жёлтого цвета, как можно было ожидать. Это связано с тем, что человеческий глаз более чувствителен к зелёному цвету. Чем ниже интенсивности составляющих, тем темнее цвет на экране. И наоборот — чем выше интенсивности цветов, тем светлее оттенки.

Цветовые модели CMY и CMYK

Модель CMY использует также три основных цвета: голубой (Cyan), фуксин (Magenta, иногда его называют «пурпурный» или «малиновый») и жёлтый (Yellow). Эти цвета описывают отражённый от белой бумаги свет трёх основных цветов RGB-модели.

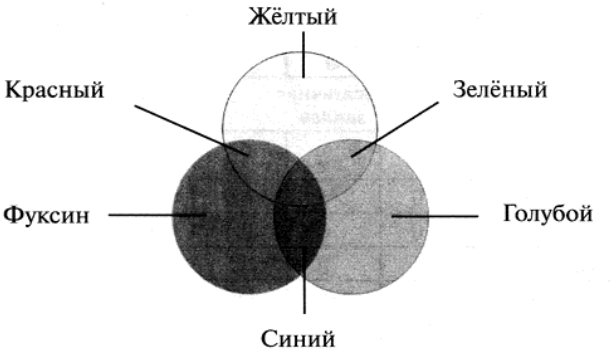


Рис. 9. Цветовая модель CMY.

Модель CMY является *субтрактивной* (основанной на вычитании) цветовой моделью. Краситель, нанесённый на белую бумагу, вычитает часть спектра из падающего белого света. Например, на поверхность бумаги нанесли жёлтый (Yellow) краситель. Теперь синий свет, падающий на бумагу, полностью поглощается. Таким образом, жёлтый носитель вычитает синий свет из падающего белого.

При смешении двух субтрактивных составляющих результирующий цвет затемняется, а при смешении всех трёх должен получиться чёрный цвет. Но при использовании реальных полиграфических красок получается не чёрный, а неопределённый тёмный цвет. Поэтому к трём основным цветам CMY-модели добавляют чёрный (Black) и получают новую цветовую модель CMYK.

Цветовая модель CMYK используется в основном в полиграфии при выводе изображения на печать.

Глубина цвета и объём памяти

Количество различных цветов K и количество битов для их кодирования (глубина цвета) L связаны формулой $K = 2^L$. При $L = 24$ бита можно закодировать $2^{24} = 16\,777\,216$ различных цветов.

Если известно разрешение экрана (количество точек по горизонтали и вертикали) и глубина цвета, можно определить объём видеопамяти для хранения одного кадра (одной страницы) изображения. Например, при разрешении экрана 640 × 480 и использовании 24 бит на точку объём видеопамяти равен $640 \cdot 480 \cdot 24 = 7\,372\,800$ бит = 900 Кбайт.

Все компьютерные изображения делятся на два больших класса — **растровые** и **векторные**. Различие между ними определяет способ хранения изображений в памяти компьютера.

Растровая графика

Растровые изображения используются для хранения реалистичных изображений, например цифровых фотографий. В памяти компьютера хранится значение цвета каждого пикселя изображения. Однако изме-

нение размеров таких изображений серьёзно отражается на их качестве.

На рис. 10а показано изображение окружности, нарисованной в растровом графическом редакторе. На рис. 10б показан результат, полученный после двух действий — уменьшения исходного изображения, а затем его увеличения. Казалось бы, должна получиться исходная окружность. Но этого не произошло, потому что при уменьшении изображения происходит потеря пикселей (а вместе с ними и цветов для цветных изображений). Когда мы увеличили предварительно уменьшенное изображение, потеря точек стала отчётливо видна.

При увеличении растрового изображения каждая точка исходного изображения представляется прямоугольным набором точек, появляется «лестничный эффект» (рис. 10в).

Рис. 10. Потеря качества растрового изображения при изменении размера.

Файлы растровых изображений имеют большой размер. Расширения имён файлов, хранящих растровые изображения: JPG, GIF, BMP и др. Программами обработки и создания растровых изображений являются MS Paint, Adobe PhotoShop.

Векторная графика

Векторные изображения хранятся в виде математических формул, описывающих объекты, из которых состоят изображения, толщину и цвет линий, цвета заливки сплошных областей и др. Векторные изображения имеют не такую богатую палитру, как растровые, однако легко переносят увеличение и уменьшение размеров. Все характеристики картинки просто пересчитываются по формулам для новых размеров изображения, и качество не ухудшается. Так как в текстовых процессорах используются векторные шрифты, то можно легко изменять их размер без изменения качества изображения символов.

Векторные изображения создаются при помощи специальных векторных редакторов, например, таких, как Adobe Illustrator, CorelDraw, 3DMax. Файлы векторных изображений имеют небольшой размер по сравнению с файлами растровых изображений.

Кодирование звуковой информации

Звук представляет собой звуковую волну с непрерывно меняющейся амплитудой и частотой. Чем больше амплитуда сигнала, тем громче звук; чем больше частота сигнала (число колебаний в секунду), тем выше тон.

В настоящее время существует **два основных способа записи звука** — *аналоговый (непрерывный)* и *цифровой (дискретный)*. Виниловая пластинка является примером аналогового хранения звуковой информации, так как звуковая дорожка изменяет свою форму непрерывно. Компакт-диски являются примером цифрового хранения звуковой информации, так как звуковая дорожка компакт-диска содержит участки с различной отражающей способностью.

Для того чтобы записать звук на какой-нибудь носитель, его нужно преобразовать в электрический сигнал. Это делается с помощью **микрофона**. Микрофоны имеют мембрану, которая колеблется под воздействием звуковых волн. К мембране присоединена катушка, перемещающаяся синхронно с мембраной в магнитном поле. В катушке возникает переменный электрический ток. Так звуковые волны преобразуются микрофоном в электрический ток переменного напряжения, который представляет собой аналоговый сигнал. Применительно к электрическому сигналу термин «аналоговый» обозначает, что этот сигнал непрерывен по времени и амплитуде (см. рис. 11а).

Для того чтобы компьютер мог обрабатывать звук, непрерывный сигнал должен быть превращён в последовательность электрических импульсов (двоичных нулей и единиц). В процессе кодирования непрерывного звукового сигнала производится его дискретизация по времени. **Дискретизация** — это преобразование непрерывных сигналов в набор дискретных значений, каждому из которых присваивается число — кодовое слово.

Для дискретизации надо несколько раз в секунду измерять величину аналогового сигнала и кодировать её, например, с помощью 256 значений.

Фактически плоскость, на которой изображён непрерывный сигнал, разбивается вертикальными и горизонтальными линиями (см. рис. 11б), и считается, что график проходит строго через узлы полученной сетки, непрерывная плавная линия заменяется ломаной.

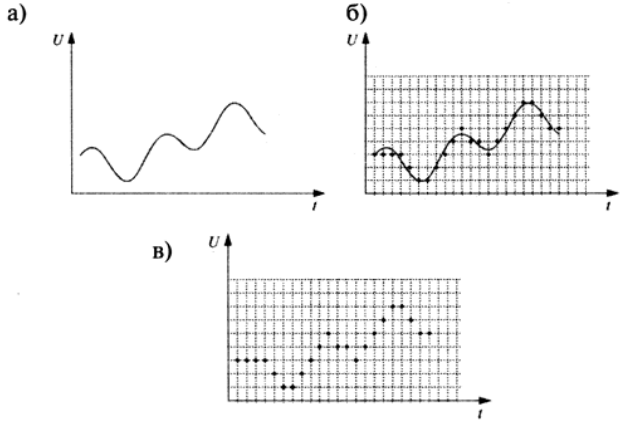


Рис. 11. Пример дискретизации аналогового сигнала.

Дискретизация по времени соответствует разбиению вертикальными линиями. Она характеризуется **частотой дискретизации**. Частота дискретизации звукового компакт-диска 44,1 кГц, DVD — примерно 96 кГц. Это значит, что величина аналогового сигнала измеряется 44 100 и 96 000 раз в секунду соответственно. Если кодируется стереозвук, отдельно кодируются два канала.

Горизонтальное разбиение также важно: чем меньше расстояние между горизонтальными линиями сетки, тем качественнее будет цифровой звук. Количество линий сетки определяет количество уровней звука, поэтому горизонтальное разбиение называется **квантованием по уровню**. Для кодирования полученных значений уровней используют двоичные числа. Количество используемых для кодирования бит называется **глубиной звука**. Если глубина звука 8 бит или 16 бит, можно закодировать соответственно $2^8 = 256$ уровней или $2^{16} = 65\,536$ уровней сигналов. Это значит, что интервал от нулевого до максимального напряжения аналогового сигнала разбивается на 256 или 65 536 уровней, что соответствует количеству высот звука (тонов).

Пример 4.4. Задание с кратким ответом

Если частота дискретизации равна 44,1 кГц, для кодирования звука используется 16 бит. Сколько памяти потребуется для хранения одной минуты стереозвуковой записи?

Решение. Измерение производится 44 100 раз в секунду, в минуту $44\,100 \cdot 60$ раз. Для сохранения результата каждого измерения требуется 16 бит = 2 байт. Всего $44\,100 \cdot 60 \cdot 2$ байт. Для стереозвука записываются два сигнала, следовательно, потребуется

$$44\,100\ 1/\text{с} \cdot 60\ \text{с} \cdot 2\ \text{байт} \cdot 2 = 1058\,4000\ \text{байт} = 10{,}1\ \text{Мбайт}.$$

Из рис. 11 видно, что преобразование аналогового сигнала в цифровой всегда происходит с некоторыми искажениями.

Преобразование непрерывной звуковой волны в последовательность звуковых импульсов различной амплитуды производится с помощью аналого-цифрового преобразователя (АЦП), размещённого на звуковой плате.

С помощью специальных программных средств (редакторов звукозаписей) открываются широкие возможности по созданию, редактированию и прослушиванию звуковых файлов. Но, как видно из примера, звуковые файлы занимают очень много места в памяти. Поэтому используются методы сжатия звуковых файлов. Качество музыки после сжатия несколько ухудшается, но это практически незаметно, так как при разработке алгоритмов сжатия учитываются законы восприятия музыки человеком.

Наиболее популярные форматы сжатых музыкальных файлов — MP3, WMA и др.

Задания для самостоятельного решения

Задания с выбором одного ответа

Пример 4.5. От разведчика была получена следующая шифрованная радиограмма, переданная с использованием азбуки Морзе:

•— —•••••—•— —•••—•—

При передаче радиограммы было потеряно разбиение на буквы, но известно, что в радиограмме использовались только следующие буквы:

А	К	О	Р	Ф
—	•—	••—	•••—	••••

Определите текст радиограммы. В ответе укажите, сколько букв было в исходной радиограмме.

1) 7
2) 8
3) 9
4) 12

Пример 4.6. От разведчика была получена следующая шифрованная радиограмма, переданная с использованием азбуки Морзе:

•— —••—•—•••— —••— — —

При передаче радиограммы было потеряно разбиение на буквы, но известно, что в радиограмме использовались только следующие буквы:

Ф	И	В	А	Н
—	•—•	••	•— — —	•— —•

Определите текст радиограммы. В ответе укажите, сколько букв было в исходной радиограмме.

1) 5
2) 6
3) 7
4) 10

Пример 4.7. От разведчика была получена следующая шифрованная радиограмма, переданная с использованием азбуки Морзе:

••—•••—•— — —••—•— — — —•—

При передаче радиограммы было потеряно разбиение на буквы, но известно, что в радиограмме использовались только следующие буквы:

Л	И	Т	К	О
•—	•••	••—	—•—	——

Определите текст радиограммы. В ответе укажите, сколько букв было в исходной радиограмме.

1) 8
2) 9
3) 10
4) 11

Пример 4.8. От разведчика была получена следующая шифрованная радиограмма, переданная с использованием азбуки Морзе:

•— —•—••••— — — —•— —•••

При передаче радиограммы было потеряно разбиение на буквы, но известно, что в радиограмме использовались только следующие буквы:

В	Р	Н	Я	И
— — —	—•—	••	—•••	•—

Определите текст радиограммы. В ответе укажите, сколько букв Н было в исходной радиограмме.

1) 1
2) 2
3) 3
4) 7

Пример 4.9. От разведчика была получена следующая шифрованная радиограмма, переданная с использованием азбуки Морзе:

—••— —• — — — ••— • — • — —• — •• — —••

При передаче радиограммы было потеряно разбиение на буквы, но известно, что в радиограмме использовались только следующие буквы:

Л	С	Ю	О	Е
— —••	—•—•	•—••	—•—	•— —

Определите текст радиограммы. В ответе укажите, сколько гласных букв было в исходной радиограмме.

1) 2
2) 3
3) 4
4) 5

Пример 4.10. В кодировке Unicode каждый символ кодируется двумя байтами. Определите информационный объём сообщения из 45 символов в этой кодировке.

1) 45 байт
2) 380 бит
3) 90 байт
4) 480 бит

Пример 4.11. В кодировке ASCII каждый символ кодируется 8 бит. Определите информационный объём сообщения в этой кодировке. **Длина этого текста 17 бит.**

1) 17 бит
2) 136 бит
3) 17 байт
4) 25 байт

Пример 4.12. В кодировке ASCII каждый символ кодируется одним байтом. Определите информационный объём сообщения в этой кодировке: **В кодировке ASCII на каждый символ отводится 1 байт.**

1) 43 байт
2) 51 бит
3) 51 байт
4) 42 байт

Пример 4.13. В кодировке Unicode каждый символ кодируется 16 битами. Определите информационный объём сообщения из 47 символов в этой кодировке.

1) 752 бит
2) 832 бит
3) 47 байт
4) 96 байт

У следующих пяти заданий имеется общее начало:

Для кодирования цвета фона web-страницы используется атрибут bgcolor="#XXXXXX", где в кавычках задаются шестнадцатеричные значения интенсивности цветовых компонентов в 24-битной RGB-модели.

Пример 4.14. Какой цвет будет у страницы, заданной тэгом < body bgcolor="#00FF00" > ?

1) белый
2) серый
3) зелёный
4) синий

Пример 4.15. Какой цвет будет у страницы, заданной тэгом < body bgcolor="#FFFFFF" > ?

1) белый
2) серый
3) чёрный
4) синий

Пример 4.16. Какой цвет будет у страницы, заданной тэгом < body bgcolor="#000000" > ?

1) белый
2) серый
3) чёрный
4) синий

Пример 4.17. Какой цвет будет у страницы, заданной тэгом < body bgcolor="#FFFF00" > ?

1) жёлтый
2) фуксин
3) голубой
4) красный

Пример 4.18. Какой цвет будет у страницы, заданной тэгом < body bgcolor="#C8C8C8" > ?

1) жёлтый
2) серый
3) белый
4) чёрный

Задания с кратким ответом

Пример 4.19. Скорость передачи данных через ADSL-соединение равна 6 144 бит/с. Передача файла через данное соединение заняла 20 с. Определите размер файла в килобайтах.

Пример 4.20. Скорость передачи данных через ADSL-соединение равна 128 Кбит/с. Размер файла составляет 368 Кбайт. Определите время передачи файла в секундах.

Пример 4.21. Файл размером 36 Кбайт был передан через ADSL-соединение за 9 с. Определите скорость передачи данных через ADSL-соединение в Кбит/с.

Пример 4.22. Скорость передачи данных через ADSL-соединение равна 256 000 бит/с. Передача файла через это соединение заняла 2 мин. Определите размер файла в килобайтах.

Пример 4.23*. Известно, что длительность непрерывного подключения к сети Интернет с помощью модема для некоторых АТС не превышает 7 мин. Определите максимальный размер файла (в Кбит), который может быть передан за время такого подключения, если модем передаёт информацию в среднем со скоростью 3 Кбайт/мин.

Пример 4.24. Автоматическое устройство осуществило перекодировку информационного сообщения на русском языке, первоначально записанного в 8-битой кодировке КОИ-8, в 16-битный Unicode. В результате преобразования информационное сообщение увеличилось на 256 бит. Какова длина сообщения в символах?

ГЛАВА 5
ЛОГИЧЕСКИЕ ОСНОВЫ ОБРАБОТКИ ИНФОРМАЦИИ

Логика

Логика (греч. logikē < logos — довод, доказательство; разумное основание; наука о рассуждении, искусство рассуждения) — наука о формах, методах и законах правильного мышления.

Мышление изучают и психология, и педагогика, и многие другие науки. По содержанию человеческое мышление бесконечно многообразно, ведь думать можно о чём угодно. Но мысли возникают и строятся по одним и тем же законам, подчиняются одним и тем же принципам, имеют одни и те же схемы или формы.

Рассмотрим пример. Следующие высказывания:

— все ананасы — это фрукты;

— все тигры — это хищные животные;

— все автомобили — это транспортные средства —

различаются по содержанию, но сходны по форме: «все *A* — это *B*», где *A* и *B* — это какие-либо предметы. Само высказывание «все *A* — это *B*» не имеет содержания, оно представляет собой форму, которую можно наполнить любым содержанием, например: «все учебники — это книги».

Рассмотрим другой пример. Три различных по содержанию высказывания:

— если наступает весна, то тает снег;

— если не готовиться к ГИА, то можно получить двойку;

— если стоит сильный туман, то самолёты не могут совершить посадку —

строятся по одной и той же форме: «если *A*, то *B*». И к этой форме можно подобрать множество различных содержательных высказываний.

Логика не интересуется содержанием мышления, она изучает только формы мышления. Логику интересует не что мы мыслим, а как мы мыслим, поэтому она также часто называется *формальной логикой*.

Например, если по содержанию высказывание «все комары — это насекомые» является понятным, осмысленным, а высказывание «все крокодилы — это птицы» является бессмысленным, то для логики эти два высказывания равноценны: ведь она занимается формами мышления, а форма у этих двух высказываний одна и та же — «все *A* — это *B*».

Таким образом, **форма мышления** — это способ, которым выражаются мысли, или схема, по которой они строятся. Существует *три формы мышления*.

1) *Понятие* — это форма мышления, которая обозначает какой-либо объект или признак объекта, отличающий его от других объектов. Примеры понятий: «собака», «растение», «планета», «химический элемент», «смелость», «трудолюбие» и т. п.

2) *Высказывание (суждение, утверждение)* — это форма мышления, в которой что-либо утверждается или отрицается о свойствах понятий и отношениях между ними, например: «Солнце не является планетой»; «некоторые вещества — это металлы»; «все цифры — это знаки»; «$2 \cdot 2 = 4$» и т. п.). Высказывание может быть истинным или ложным.

3) *Умозаключение* — это форма мышления, в которой из двух или нескольких исходных высказываний получают новое высказывание или вывод. Пример умозаключения: «Все металлы электропроводны. Железо — это металл. Железо электропроводно».

Помимо форм мышления, логика также занимается законами мышления, т. е. такими правилами, соблюдение которых приводит рассуждение к истинным выводам при условии истинности исходных суждений.

Основная цель логики — исследование того, как из одних утверждений можно выводить другие. При этом предполагается, что вывод зависит только от способа связи входящих в него утверждений и их строения, а не от их конкретного содержания. Отсюда ещё одно определение логики. **Логика** — наука, изучающая методы установления истинности или ложности одних высказываний (утверждений) на основе истинности или ложности других высказываний.

Исторически логика изучалась как часть философии. Сейчас логика изучается ещё и как часть математики и информатики.

Логика появилась примерно в IV веке до н.э. в Древней Греции, её создателем считается Аристотель. *Аристотелевская*, или *традиционная, логика* для анализа правильного мышления использует *естественный язык*, а *символическая логика*, появившаяся в XIX веке, пользуется *искусственным языком* символов, подобным языку математики.

В конце XIX—начале XX века были заложены основы математической, или символической, логики. Её суть заключается в том, что для обнаружения истинного значения выражений естественного языка можно применять математические методы.

Большой вклад в развитие символической логики внесли такие учёные, как Дж. Буль, О. де Морган, Г. Фреге, Ч. Пирс и др. В XX веке математическая логика оформилась в качестве самостоятельной дисциплины.

В середине XX века развитие вычислительной техники привело к появлению логических элементов, логических блоков и устройств вычислительной техники, что было связано с дополнительной разработкой таких областей логики, как проблемы логического синтеза, логическое проектирование и логическое моделирование логических устройств и средств вычислительной техники.

Алгебра логики

Алгебра логики — раздел математической логики, изучающий логические высказывания и методы установления их истинности или ложности с помощью алгебраических методов.

Основоположником алгебры логики является английский математик Джордж Буль (1815—1864). Он изучал логику мышления математическими методами и разработал алгебраические методы решения традиционных логических задач, поэтому алгебру логики называют ещё *булевой алгеброй*.

Логическое высказывание — это повествовательное предложение, о котором можно сказать, истинно оно или ложно.

Например, высказывание «сумма углов треугольника равна 180 градусам» — истинно, а высказывание «Рим — столица Греции» — ложно. Не всякое повествовательное предложение является логическим высказыванием. Определить, истинны или ложны предложения «ученик восьмого класса» и «очень жаркое лето» нельзя. Вопросительные предложения, предложения в повелительной форме также не являются высказываниями.

Истинность или ложность высказывания определяется не алгеброй логики, а конкретными науками, практикой, наблюдениями. Для алгебры логики важен не смысл высказывания, важна лишь его истинность или ложность.

Из заданных высказываний можно строить новые высказывания. Для этого используются слова и словосочетания «и», «или», «не», «либо..., либо», «тогда и только тогда» и др. Такие слова и словосочетания называются **логическими связками**.

Высказывания, образованные из других высказываний, называются *составными (сложными)*. Высказывания, не являющиеся составными, называются *простыми* или *элементарными*. Например, из простых высказываний «Сергей футболист», «Сергей пловец» можно получить составное высказывание «Сергей футболист и пловец». Истинность этого высказывания означает, что Сергей занимается двумя видами спорта. Если высказывание ложно, то Сергей либо не занимается обоими видами спорта, либо не занимается хотя бы одним из них.

Другое составное высказывание «Сергей футболист или пловец» означает в алгебре логики, что при истинности высказывания Сергей или футболист, или пловец, или футболист и пловец одновременно. Если же высказывание ложно, значит, Сергей не занимается ни футболом, ни плаванием.

Алгебра логики позволяет определять истинность или ложность составных высказываний, не вникая в их содержание.

В алгебре логики высказывания для формализации работы обозначают символическими именами, например: *A, B, C*. Тогда если обозначить простые высказывания «Денис сделал уроки» именем *A*, «Денис пошёл в кино» именем *B*, то составное высказывание «Денис сделал уроки и пошёл в кино» можно записать как «*A* и *B*». Здесь «и» — *логическая связка*, *A, B — логические переменные*, которые могут принимать логические значения «истина» или «ложь».

Логические значения «истина» и «ложь» могут обозначаться иначе:

Истина	**Ложь**
true	false
да	нет
1	0

Истинность или ложность составных высказываний зависит от истинности или ложности простых высказываний.

В алгебре логики логические связки рассматриваются как логические операции, имеющие название и обозначение.

Логические операции

В данной ниже таблице приведены логические операции и их обозначения, используемые как в алгебре логики, так и при записи алгоритмов и программ на некоторых языках программирования. Жирным шрифтом выделены обозначения операций, используемые в заданиях ГИА.

№	**Операция**	**Обозначение**	**Соответствующие речевые обороты**
1	2	3	4
1	отрицание (инверсия, логическое НЕ)	$\neg A$ $\bar{A}$ **не** ***A*** not *A*	не *A*; неверно, что *A*
2	конъюнкция (логическое умножение, логическое И)	$A \wedge B$ ***A* & *B*** $A \cdot B$ ***A* и *B*** *A* and *B*	*A* и *B*; как *A*, так и *B*; *A* вместе с *B*

Продолжение таблицы

1	2	3	4
3	дизъюнкция (логическое сложение, логическое ИЛИ, включающее ИЛИ)	$A \vee B$ $A + B$ ***A* или *B*** *A* or *B*	*A* или *B*; *A* или *B* или оба вместе
4	исключающая дизъюнкция (исключающее ИЛИ, сложение по модулю 2, строгая дизъюнкция)	$A \oplus B$ *A* хог *B*	либо *A*, либо *B*; *A* или *B*, но не оба вместе; только *A* или только *B*
5	импликация (следование)	$A \rightarrow B$ $A \Rightarrow B$ *A* — посылка *B* — следствие	если *A*, то *B*; из *A* следует *B*; *A* влечёт *B*; для *A* необходимо *B*; для *B* достаточно *A*; *A* только тогда, когда *B*; *B* тогда, когда *A*; все *A* есть *B*
6	эквиваленция (равнозначность)	$A \leftrightarrow B$ $A \equiv B$ $A \sim B$	*A* эквивалентно *B*; *A* равносильно *B* ; *A* необходимо и достаточно для *B*; *A* тогда и только тогда, когда *B*; либо *A* и *B*, либо ни *A*, ни *B*

Операция отрицания выполняется над одним операндом. Такие операции называются *одноместными* или *унарными*. Правило выполнения этой операции: если значение логического операнда *A* истинно, то значение $\neg A$ ложно, если *A* ложно, то $\neg A$ истинно. Если высказывание «Денис сделал уроки» обозначить переменной *A*, то $\neg A$ соответствует высказыванию «Денис не сделал уроки» или «неверно, что Денис сделал уроки».

Все остальные логические операции, перечисленные в таблице, выполняются над двумя операндами и называются *двуместными* или *бинарными*.

При выполнении операции конъюнкции результатом будет истина только в том случае, когда оба операнда истинны. Во всех остальных случаях результатом выполнения конъюнкции будет ложь. Например, высказывание *A* «число 7 простое» истинно, высказывание *B* «число 10 чётное» истинно. Высказывание «число 7 простое и число 10 чётное»

истинно. Высказывания «число 7 не простое и число 10 чётное», «число 7 простое и число 10 нечётное», «число 7 не простое и число 10 нечётное» ложны.

Результатом выполнения операции дизъюнкции будет ложь только в том случае, когда оба операнда ложны. Для получения истины хотя бы один операнд должен быть истинным. Например, три составных высказывания «число 7 простое или число 10 чётное», «число 7 не простое или число 10 чётное», «число 7 простое или число 10 нечётное» истинны, высказывание «число 7 не простое или число 10 нечётное» ложно.

Если операнды имеют разные логические значения, результатом операции исключающей дизъюнкции (исключающего ИЛИ) будет истина, а результатом операции эквиваленции — ложь. Если оба простые высказывания ложны или оба простые высказывания истинны, результатом операции исключающей дизъюнкции будет ложь, а операции эквиваленции — истина. Можно сделать вывод, что операция эквиваленции есть отрицание операции исключающей дизъюнкции и, наоборот: операция исключающей дизъюнкции есть отрицание операции эквиваленции.

В операции импликации $A \to B$ операнд A называют *посылкой*, операнд B — *следствием* или *заключением*. Импликация ложна лишь тогда, когда посылка истинна, а следствие ложно. Во всех остальных случаях импликация истинна.

Тот факт, что когда A ложно, импликация $A \to B$ истинна при любом B, соответствует принципу «ex falso quod libet» (лат. «из ложного (высказывания) — всё что угодно»).

Составим таблицу, иллюстрирующую правило выполнения операции импликации для высказываний A «идёт дождь», B «на небе тучи».

Пример составного высказывания «Если..., то...»

A «идёт дождь»	B «на небе тучи»	Составное высказывание	$A \to B$
ложь	ложь	«если не идёт дождь, то на небе нет туч»	истина
ложь	истина	«если не идёт дождь, то на небе тучи»	истина
истина	ложь	«если идёт дождь, то на небе нет туч»	ложь
истина	истина	«если идёт дождь, то на небе тучи»	истина

Из всех логических операций импликация вызывает больше всего вопросов. В разговорной речи связка «если..., то...» описывает причинно-следственную связь между высказываниями. Напомним, что в алгебре логики смысл высказываний не учитывается, рассматривается только их истинность или ложность. Высказывания

A и *B* могут быть даже не связанными по содержанию, например: «если в огороде бузина, то в Киеве дядька». Надо понимать, что результат операции импликации, как и других логических операций, определяется истинностью или ложностью переменных, а не наличием причинно-следственных связей между высказываниями. Приведённый в таблице пример лишь помогает понять и запомнить правило выполнения импликации.

Таблицы истинности

Для задания логических операций используются **таблицы истинности**. В таблицах истинности перечисляются все возможные сочетания значений логических переменных (операндов) и результаты выполнения соответствующих логических операций. Как правило, используются обозначения логических значений 0 (ложь) и 1 (истина).

Таблицы истинности могут быть записаны так же, как таблицы умножения. Такая форма записи называется *картой Карно*.

Таблицы истинности логических операций

отрицание

A	$\neg A$
0	1
1	0

конъюнкция

A \ *B*	0	1
0	0	0
1	0	1

дизъюнкция

A \ *B*	0	1
0	0	1
1	1	1

исключающая дизъюнкция

A \ *B*	0	1
0	0	1
1	1	0

импликация $A \to B$

A \ *B*	0	1
0	1	0
1	1	1

эквиваленция

A \ B	0	1
0	1	0
1	0	1

В первой строке каждой таблицы записаны значения логической переменной *B*, в первом столбце — значения логической переменной *A*. На пересечении строки и столбца — результат выполнения логической операции при соответствующих значениях *A* и *B*.

Другой вид таблиц истинности показан на с. 78. Наборы значений аргументов и соответствующие им значения функций записаны в строках сводной таблицы истинности.

Сводная таблица истинности логических операций

Логические переменные		Логические операции					
		отрицание	конъюнкция	дизъюнкция	исключающая дизъюнкция	импликация	эквиваленция
A	B	$\neg A$	$A \,\&\, B$	$A \vee B$	$A \oplus B$	$A \to B$	$A \leftrightarrow B$
0	**0**	1	0	0	0	1	1
0	**1**	1	0	1	1	1	0
1	**0**	0	0	1	1	0	0
1	**1**	0	1	1	0	1	1

Совокупность значений переменных называют *набором*, например: $A = 0$, $B = 1$. Если количество переменных N, то количество различных наборов 2^N. Для двух переменных количество различных наборов $2^2 = 4$. Для всех четырёх наборов логических переменных A и B записаны результаты применения к ним соответствующих логических операций.

Логические выражения

Составное логическое высказывание можно представить в виде *логического выражения (формулы)*, состоящего из логических констант «истина» или «ложь», логических переменных (операндов), знаков логических операций и скобок. Логические выражения принимают значения «истина» или «ложь».

Логическое выражение строится по правилам:

1) всякая логическая переменная, а также логические константы «истина» и «ложь» есть выражение;
2) если A — выражение, то $\neg A$ — выражение;
3) если A и B — выражения, то $(A \,\&\, B)$, $(A \vee B)$, $(A \oplus B)$, $(A \to B)$, $(A \leftrightarrow B)$ — выражения.

В соответствии с этими правилами $\neg A \,\&\, B \vee A \,\&\, \neg B$ — выражение, $\neg A \,\&\, \vee B$ — не выражение.

В логических выражениях операции выполняются в соответствии с их приоритетами:

1) отрицание;
2) конъюнкция;
3) дизъюнкция, исключающая дизъюнкция (исключающее ИЛИ);
4) импликация, эквиваленция.

Самый высокий приоритет имеет операция отрицания, она выполняется в первую очередь. Далее, как и в арифметике, логическое умно-

жение, затем логическое сложение. Операции одного приоритета выполняются слева направо. Скобки меняют порядок выполнения операций. Например, при вычислении двух выражений последовательность выполнения операций будет следующей:

$$\begin{array}{lllll} 1 & 3 & 5 & 4 & 2 \\ \neg A & \& B & \vee A & \& \neg & B \end{array} \qquad \begin{array}{lllllll} 2 & 4 & & 1 & & 5 & 3 \\ \neg A & \& & (B & \vee & A) & \& & \neg B \end{array}$$

Если $A = 1$, $B = 0$, то результат вычисления первого выражения будет равен 1 (истина), второго — 0 (ложь).

Операции импликации, исключающего ИЛИ, эквиваленции можно выразить через отрицание, конъюнкцию и дизъюнкцию — по формулам:

$$\begin{array}{l} A \rightarrow B = \neg A \vee B; \\ A \oplus B = \neg A \,\&\, B \vee A \,\&\, \neg B; \\ A \leftrightarrow B = A \,\&\, B \vee \neg A \,\&\, \neg B. \end{array} \qquad (*)$$

Именно по этой причине операции отрицания, конъюнкции и дизъюнкции называют основными: этих трёх операций достаточно, чтобы описывать и обрабатывать логические выражения *.

Логические выражения, у которых для всех наборов входящих в них переменных значения в таблицах истинности совпадают, называются ***равносильными*** или ***эквивалентными***. Равносильность выражений обозначается знаком равенства «=».

Логические выражения, принимающие значение «истина» при некоторых наборах входящих в них переменных и значение «ложь» при других наборах, называются ***выполнимыми***. Пример: $A \vee B$.

Логические выражения, принимающие значение «истина» при любых значениях входящих в них переменных, называются ***тождественно-истинными*** выражениями или ***тавтологиями***. Пример: $A \vee \neg A$.

Логические выражения, принимающие значение «ложь» при любых значениях входящих в них переменных, называются ***тождественно-ложными*** выражениями или ***противоречиями***. Пример: $A \,\&\, \neg A$.

Логическая функция — это функция логических переменных $X_1, X_2, X_3, ..., X_N$:

$$Y = F(X_1, X_2, X_3, ..., X_N),$$

которая может принимать только два значения — истина (1) или ложь (0).

Логическая функция может быть задана таблицей истинности. Число строк в таблице — это число возможных наборов значений аргументов. Оно равно $S = 2^N$, где N — количество переменных.

Для каждого набора функция может принимать два значения, поэтому количество различных функций N переменных равно 2^S. Приведём пример различных функций при $N = 2$. В этом случае $S = 2^2 = 4$, а количество разных функций равно $2^4 = 16$.

---

* На самом деле описать и обработать логические выражения можно, используя и другие наборы основных операций, но при изучении информатики в школе они не рассматриваются.

Переменные		Функции							
A	B	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8
0	**0**	0	0	0	0	0	0	0	0
0	**1**	0	0	0	0	1	1	1	1
1	**0**	0	0	1	1	0	0	1	1
1	**1**	0	1	0	1	0	1	0	1

Переменные		Функции							
A	B	Y_9	Y_{10}	Y_{11}	Y_{12}	Y_{13}	Y_{14}	Y_{15}	Y_{16}
0	**0**	1	1	1	1	1	1	1	1
0	**1**	0	0	0	0	1	1	1	1
1	**0**	0	0	1	1	0	0	1	1
1	**1**	0	1	0	1	0	1	0	1

Некоторые из этих функций соответствуют известным вам логическим операциям, например: Y_2 — конъюнкция, Y_8 — дизъюнкция.

Построение таблиц истинности

Убедиться в равносильности двух выражений можно, построив для них таблицы истинности. Количество строк в таблице будет равно числу наборов 2^N, где N — количество логических переменных, входящих в выражения.

Для проверки равносильности выражений $A \to B$ и $\neg A \vee B$ построим таблицу истинности выражения $\neg A \vee B$. Операции будем выполнять в соответствии с их приоритетами: первой выполняется операция отрицания, второй — дизъюнкции. В таблице количество строк равно четырём. Количество столбцов определяется количеством логических переменных и логических операций в выражении.

A	B	$\neg A$	$\neg A \vee B$	$A \to B$
0	0	1	**1**	**1**
0	1	1	**1**	**1**
1	0	0	**0**	**0**
1	1	0	**1**	**1**

Результат в четвёртом столбце получен дизъюнкцией значений третьего и второго столбцов. Он совпадает с таблицей истинности операции импликации, записанной в последнем столбце.

Для построения таблицы истинности логического выражения необходимо:

1) определить количество строк таблицы по формуле 2^N, где N — количество используемых логических переменных;
2) определить порядок выполнения операций в формуле с учётом приоритетов и скобок;
3) найти значения промежуточных формул и конечного результата в соответствии с таблицами истинности.

Построим таблицу истинности для выражения $\neg A \,\&\, B \vee A \,\&\, \neg B$, чтобы убедиться в том, что оно равносильно исключающему ИЛИ $A \oplus B$.

Переменные		Промежуточные логические выражения				Результат	
1	2	3	4	5	6	7	
A	B	$\neg A$	$\neg B$	$\neg A \,\&\, B$	$A \,\&\, \neg B$	$\neg A \,\&\, B \vee A \,\&\, \neg B$	$A \oplus B$
0	0	1	1	0	0	**0**	**0**
0	1	1	0	1	0	**1**	**1**
1	0	0	1	0	1	**1**	**1**
1	1	0	0	0	0	**0**	**0**

В первую очередь выполняется операция отрицания (столбцы 3 и 4). Затем конъюнкция (логическое умножение) столбцов 3 и 2, столбцов 1 и 4. Результат в столбце 7 получен дизъюнкцией (логическим сложением) значений пятого и шестого столбцов, он совпадает с таблицей истинности операции исключающего ИЛИ (последний столбец).

Пример 5.1. Задание с выбором одного ответа

Для какого из указанных значений числа X истинно выражение $\neg (X >= 7) \,\&\, (X > 4)$?

1) 4
2) 6
3) 8
4) 10

Решение. Первой выполняется операция отрицания. Отрицанием высказывания $(X >= 7)$ является $(X < 7)$. Получим выражение $(X < 7) \,\&\, (X > 4)$. Для того чтобы выражение было истинным, оба неравенства должны одновременно выполняться, т. е. быть истинными. Этому условию соответствуют значения $X = 5$ и $X = 6$.

Ответ: 2.

Пример 5.2 Задание с выбором одного ответа

Для какого из указанных значений числа X истинно выражение $(X \cdot 2 > 10) \,\&\, \neg (X + 3 < 8) \vee \neg (X > 0)$?

1) 0
2) 2
3) 4
4) 7

Решение. Преобразуем неравенства так, чтобы слева оставалась только переменная X. Получим $(X > 5)$ & $\neg (X < 5) \vee \neg (X > 0)$. Далее выполним операции отрицания, получим $(X > 5)$ & $(X >= 5) \vee (X <= 0)$. Затем выполняется операция конъюнкции $(X > 5)$ & $(X >= 5)$, результатом выполнения которой будет истина только в том случае, если оба неравенства будут выполняться. Это возможно только при $X > 5$. Наконец, последней выполняется операция дизъюнкции. Для получения истины необходимо, чтобы хотя бы один из операндов был истинным: $X > 5$ или $X <= 0$. В предложенных ответах все числа положительные, значит, ответ $X = 7$.

Эту задачу можно решить составлением таблицы истинности. Пусть переменная A — неравенство $(X \cdot 2 > 10)$, переменная $B - (X + 3 < 8)$, переменная $C - (X > 0)$. Тогда выражение можно записать в виде A & $\neg B \vee \neg C$. Порядок выполнения операций: отрицание B, отрицание C, логическое умножение, логическое сложение.

X	A (X > 5)	B (X < 5)	C (X > 0)	$\neg B$	A & $\neg B$	$\neg C$	A & $\neg B \vee \neg C$
0	0	1	1	0	0	0	0
2	0	1	1	0	0	0	0
4	0	1	1	0	0	0	0
7	1	0	1	1	1	0	1

Ответ: 4.

Основные законы алгебры логики

Способ определения истинности сложного выражения путём построения таблиц истинности становится громоздким при возрастании количества логических переменных, так как число наборов резко увеличивается. В таких случаях используют способы упрощения формул путём применения равносильных преобразований. Ниже в таблице приведены основные законы алгебры логики.

Основные законы алгебры логики

Закон	Дизъюнкция	Конъюнкция
1	2	3
коммутативности	$A \vee B = B \vee A$	A & $B = B$ & A
ассоциативности	$(A \vee B) \vee C = B \vee (A \vee C)$	A & B & $C = B$ & A & C
дистрибутивности	A & $(B \vee C) = A$ & $B \vee A$ & C	$A \vee B$ & $C = (A \vee B)$ & $(A \vee C)$
законы де Моргана	$\neg(A \vee B) = \neg A$ & $\neg B$	$\neg(A$ & $B) = \neg A \vee \neg B$

Продолжение таблицы

1	2	3
идемпотентности	$A \vee A = A$	$A \,\&\, A = A$
поглощения	$A \vee A \,\&\, B = A$ $A \vee \neg A \,\&\, B = A \vee B$	$A \,\&\, (A \vee B) = A$ $A \,\&\, (\neg A \vee B) = A \,\&\, B$
склеивания	$A \,\&\, B \vee \neg A \,\&\, B = B$	$(A \vee B) \,\&\, (\neg A \vee B) = B$
закон исключённого третьего	$A \vee \neg A = 1$	—
закон непротиворечия	—	$A \,\&\, \neg A = 0$
законы исключения констант	$A \vee 0 = A$ $A \vee 1 = 1$	$A \,\&\, 1 = A$ $A \,\&\, 0 = 0$
двойного отрицания	$\neg(\neg A) = A$	

Выполняется **закон двойственности**: если конъюнкцию заменить дизъюнкцией, дизъюнкцию заменить конъюнкцией, 0 заменить на 1, а 1 на 0, то равносильность, записанная для дизъюнкции, перейдёт в равносильность, записанную для конъюнкции.

Под упрощением формулы понимают равносильное преобразование, приводящее к нормальной форме.

Выражение имеет **нормальную форму**, если в нём используются только операции дизъюнкции, конъюнкции и отрицания переменных. При этом не используются двойное отрицание и отрицание выражений.

Для приведения выражения к нормальной форме в первую очередь избавляются от отрицаний выражений, операций импликации, эквиваленции и исключающего ИЛИ. Затем используют законы алгебры логики для уменьшения количества вхождений переменных.

Применение законов алгебры логики, похожих на законы преобразований в обычной алгебре, не вызывает затруднений. Рассмотрим примеры применения законов склеивания, поглощения, идемпотентности, законов де Моргана, не имеющих аналогов в обычной алгебре.

Закон склеивания $A \,\&\, B \vee \neg A \,\&\, B = B$ и $(A \vee B) \,\&\, (\neg A \vee B) = B$.

1) Упростить выражение $X \,\&\, Y \,\&\, Z \vee \neg X \,\&\, Y \,\&\, Z$. В выражении $X \,\&\, Y \,\&\, Z \vee \neg X \,\&\, Y \,\&\, Z$ переменной A соответствует переменная X, переменной B соответствует выражение $Y \,\&\, Z$. Тогда в результате упрощения получим

$$X \,\&\, Y \,\&\, Z \vee \neg X \,\&\, Y \,\&\, Z = Y \,\&\, Z.$$

2) Упростить выражение $(X \vee Y \vee Z) \,\&\, (X \vee \neg Y \vee Z)$. В выражении $(X \vee Y \vee Z) \,\&\, (X \vee \neg Y \vee Z)$ переменной A соответствует переменная Y, переменной B соответствует выражение $X \vee Z$. Получим

$$(X \vee Y \vee Z) \,\&\, (X \vee \neg Y \vee Z) = X \vee Z.$$

3) Упростить выражение $X \,\&\, Y \,\&\, Z \vee \neg X \,\&\, Y \,\&\, Z \vee X \,\&\, \neg Y \,\&\, Z \vee \vee X \,\&\, Y \,\&\, \neg Z$. В этом примере, на первый взгляд, закон склеивания можно использовать только один раз — для первого слагаемого и одного из трёх других:

$$\begin{gathered} X \,\&\, Y \,\&\, Z \vee \neg X \,\&\, Y \,\&\, Z \vee X \,\&\, \neg Y \,\&\, Z \vee X \,\&\, Y \,\&\, \neg Z = \\ = Y \,\&\, Z \vee X \,\&\, \neg Y \,\&\, Z \vee X \,\&\, Y \,\&\, \neg Z = \\ = X \,\&\, Z \vee \neg X \,\&\, Y \,\&\, Z \vee X \,\&\, Y \,\&\, \neg Z = \\ = X \,\&\, Y \vee \neg X \,\&\, Y \,\&\, Z \vee X \,\&\, \neg Y \,\&\, Z. \end{gathered}$$

Но если использовать закон идемпотентности $A = A \vee A$, можно в исходную формулу добавить ещё два слагаемых, совпадающих с первым. Получим

$$\begin{gathered} (X \,\&\, Y \,\&\, Z \vee \neg X \,\&\, Y \,\&\, Z) \vee (X \,\&\, Y \,\&\, Z \vee X \,\&\, \neg Y \,\&\, Z) \vee \\ \vee (X \,\&\, Y \,\&\, Z \vee X \,\&\, Y \,\&\, \neg Z) = Y \,\&\, Z \vee X \,\&\, Z \vee X \,\&\, Y. \end{gathered}$$

Закон поглощения (см. табл. на с. 82—83)

1) Упростить выражение $X \vee X \,\&\, Y \vee X \,\&\, Z \vee X \,\&\, Y \,\&\, Z$. Для всех слагаемых общим сомножителем является X. Последовательно применяя закон поглощения $A \vee A \,\&\, B = A$, получим

$$\begin{gathered} X \vee X \,\&\, Y \vee X \,\&\, Z \vee X \,\&\, Y \,\&\, Z = X \vee X \,\&\, Z \vee X \,\&\, Y \,\&\, Z = \\ = X \vee X \,\&\, Y \,\&\, Z = X. \end{gathered}$$

2) Упростить выражение $X \vee \neg X \,\&\, Y \vee \neg X \,\&\, Z \vee \neg X \,\&\, Y \,\&\, Z$. Последовательно применяя закон поглощения $A \vee \neg A \,\&\, B = A \vee B$, получим

$$\begin{gathered} X \vee \neg X \,\&\, Y \vee \neg X \,\&\, Z \vee \neg X \,\&\, Y \,\&\, Z = \\ = X \vee Y \vee \neg X \,\&\, Z \vee \neg X \,\&\, Y \,\&\, Z = \\ = X \vee Y \vee Z \vee \neg X \,\&\, Y \,\&\, Z = \\ = X \vee Y \vee Z \end{gathered}$$

В предпоследней строке к выражению $Z \vee \neg X \,\&\, Y \,\&\, Z$ был применён закон поглощения $A \vee A \,\&\, B = A$.

3) Упростить выражение $(X \vee Y) \,\&\, (X \vee Y \vee Z) \,\&\, (X \vee Y \vee \neg Z)$. Для всех сомножителей общим слагаемым является $X \vee Y$, это и есть результат упрощения.

> **Совет:** В операциях конъюнкции и дизъюнкции имена переменных записывайте в алфавитном порядке, причём сначала переменную, а затем её отрицание.

Пример 5.3. Задание с выбором одного ответа

Какое из перечисленных выражений является тождественно ложным?

1) $A \vee B \,\&\, (C \vee \neg A \vee A \,\&\, C) \,\&\, \neg B$
2) $A \,\&\, B \,\&\, (C \vee \neg A \vee A \,\&\, C) \vee \neg B$
3) $A \,\&\, B \vee (C \vee \neg A \vee A \,\&\, C) \,\&\, \neg B$
4) $A \,\&\, B \,\&\, (C \vee \neg A \vee A \,\&\, C) \,\&\, \neg B$

Решение. Выполним преобразования выражений. Выделим шрифтом фрагменты, к которым применяются законы алгебры логики.

1) $A \vee B \,\&\, (C \vee \neg A \vee A \,\&\, C) \,\&\, \neg B = A \vee \boldsymbol{B \,\&\, \neg B}\, (C \vee \neg A \vee \vee A \,\&\, C) = A + 0 = A$. Формула выполнимая.

2) $A \,\&\, B \,\&\, (\boldsymbol{C} \vee \neg A \vee \boldsymbol{A \,\&\, C}) \vee \neg B = A \,\boldsymbol{\&}\, \boldsymbol{B} \,\&\, (C \vee \neg A) \vee \vee \boldsymbol{\neg B = A \,\&\, (C \vee \neg A)} \vee \neg B = A \,\&\, C \vee \neg B$. Формула выполнимая.

3) $A \,\&\, B \vee (\boldsymbol{C} \vee \neg A \vee \boldsymbol{A \,\&\, C}) \,\&\, \neg B = A \,\&\, B \vee (C \vee \neg A) \,\&\, \neg B = = A \,\&\, B \vee \neg B \,\&\, C \vee \neg A \,\&\, \neg B$. Формула выполнимая.

4) $A \,\&\, \boldsymbol{B} \,\&\, (C \vee \neg A \vee A \,\&\, C) \,\&\, \boldsymbol{\neg B} = A \,\&\, \boldsymbol{B} \,\&\, \boldsymbol{\neg B} \,\&\, (C \vee \vee \neg A \vee A \,\&\, C) = \boldsymbol{A \,\&\, 0 \,\&\, (C \vee \neg A \vee A \,\&\, C)} = 0$. Формула тождественно ложная.

Ответ: 4.

Пример 5.4. Задание с выбором одного ответа

Логическое выражение $\neg(A \,\&\, (B \vee \neg C) \vee \neg A \,\&\, B)$ равносильно выражению

1) 1
2) $\neg A \,\&\, \neg B \vee \neg B \,\&\, C$
3) $\neg A \vee \neg B \,\&\, C$
4) $\neg A \,\&\, C \vee \neg B$

Решение. Выполним преобразования, используя законы де Моргана, закон двойного отрицания и формулу поглощения:

$$\neg(A \,\&\, (B \vee \neg C) \vee \neg A \,\&\, B) = \neg(A \,\&\, (B \vee \neg C)) \,\&\, \neg(\neg A \,\&\, B) = = (\neg A \vee \neg(B \vee \neg C)) \,\&\, (A \vee \neg B) = (\neg A \vee \neg B \,\&\, C) \,\&\, (A \vee \neg B) = = \neg A \,\&\, \neg B \vee \neg B \,\&\, C.$$

Ответ: 2.

Задания для самостоятельного решения

Задания с выбором одного ответа

Пример 5.5. Таблица истинности логической функции $F = A \,\&\, B \vee \neg A \,\&\, \neg B$ имеет вид

1)

A	*B*	*F*
0	0	1
0	1	1
1	0	1
1	1	0

2)

A	*B*	*F*
0	0	0
0	1	1
1	0	1
1	1	0

3)

A	*B*	*F*
0	0	1
0	1	0
1	0	0
1	1	1

4)

A	*B*	*F*
0	0	1
0	1	0
1	0	1
1	1	0

Пример 5.6. Таблица истинности логической функции $F = \neg A \,\&\, B \,\&\, \neg A \,\&\, \neg B$ имеет вид

A	*B*	*F*
0	0	0
0	1	0
1	0	0
1	1	0

1)

A	*B*	*F*
0	0	0
0	1	1
1	0	1
1	1	0

2)

A	*B*	*F*
0	0	1
0	1	0
1	0	0
1	1	1

3)

A	*B*	*F*
0	0	1
0	1	1
1	0	1
1	1	1

4)

Пример 5.7. Таблица истинности логической функции $F = \neg A \,\&\, \neg B \vee A \,\&\, \neg B$ имеет вид

A	*B*	*F*
0	0	1
0	1	0
1	0	1
1	1	0

1)

A	*B*	*F*
0	0	0
0	1	1
1	0	1
1	1	0

2)

A	*B*	*F*
0	0	1
0	1	1
1	0	0
1	1	0

3)

A	*B*	*F*
0	0	0
0	1	0
1	0	0
1	1	1

4)

Пример 5.8. Таблица истинности логической функции $F = \neg A \,\&\, B \vee \neg A \,\&\, \neg B$ имеет вид

A	*B*	*F*
0	0	1
0	1	0
1	0	1
1	1	0

1)

A	*B*	*F*
0	0	0
0	1	1
1	0	1
1	1	0

2)

A	*B*	*F*
0	0	1
0	1	1
1	0	0
1	1	0

3)

A	*B*	*F*
0	0	1
0	1	0
1	0	0
1	1	1

4)

Пример 5.9. Таблица истинности логической функции $F = (A \,\&\, B) \,\&\, \neg A \,\&\, \neg B$ имеет вид

A	*B*	*F*
0	0	1
0	1	1
1	0	1
1	1	1

1)

A	*B*	*F*
0	0	0
0	1	1
1	0	1
1	1	0

2)

A	*B*	*F*
0	0	1
0	1	1
1	0	0
1	1	0

3)

A	*B*	*F*
0	0	0
0	1	0
1	0	0
1	1	0

4)

Пример 5.10. Для какого из указанных значений числа X истинно выражение $\neg(X < 2) \,\&\, (X < 3)$?

1) 1
2) 2
3) 3
4) 4

Пример 5.11. Для какого из указанных значений числа X истинно выражение $\neg(X > 2) \vee (X > 6)$?

1) 2
2) 3
3) 4
4) 5

Пример 5.12. Для какого из указанных значений числа X истинно выражение $\neg(X > 5)\ \&\ \neg(X < 2)$?

1) 0
2) 1
3) 3
4) 6

Пример 5.13. Для какого из указанных значений числа X истинно выражение $\neg((X - 2 > 6) \vee (X - 4 > 8))\ \&\ \neg(X \cdot 3 > 25)$?

1) 7
2) 9
3) 11
4) 12

Пример 5.14. Для какого из указанных значений числа X истинно выражение $\neg(X : 2 < 5)\ \&\ \neg(X + 17 > 30)\ \&\ (X \cdot 3 < 39)$?

1) 8
2) 9
3) 12
4) 13

Пример 5.15. Укажите тождественно истинное выражение.

1) $\neg A \vee \neg B\ \&\ (\neg A \vee A\ \&\ B) \vee \neg B$
2) $A\ \&\ (\neg A\ \&\ B \vee A\ \&\ \neg B) \vee \neg A$
3) $A\ \&\ (\neg A\ \&\ B \vee A\ \&\ \neg B)\ \&\ \neg A$
4) $A \vee B\ \&\ (\neg A \vee A\ \&\ B) \vee \neg B$

Пример 5.16. Укажите тождественно ложное выражение.

1) $\neg(A \vee B)\ \&\ (\neg A \vee B)$
2) $\neg(A\ \&\ B)\ \&\ (\neg A \vee B)$
3) $\neg(A \vee B)\ \&\ (\neg A\ \&\ B)$
4) $\neg(A\ \&\ B)\ \&\ (\neg A\ \&\ B)$

Пример 5.17. Логическое выражение $(\neg A \vee C)\ \&\ \neg(A\ \&\ C)\ \&$ $\&\ (B \vee \neg C)\ \&\ \neg(B\ \&\ C)$ равносильно выражению

1) $(\neg A\ \&\ \neg C) \vee (\neg B\ \&\ \neg C)$
2) $\neg A \vee \neg$C
3) $\neg A \vee \neg B \vee \neg C$
4) $\neg A\ \&\ C \vee \neg A\ \&\ \neg$C $\vee \neg B\ \&\ \neg C$

Пример 5.18. Логическое выражение $\neg(A\ \&\ B \vee \neg C)) \vee \neg A\ \&\ B)$ равносильно выражению

1) 0
2) $\neg A\ \&\ B \vee \neg A\ \&$ C $\vee \neg B\ \&\ C$
3) $\neg A \vee \neg B \vee \neg C$
4) $\neg A\ \&\ B \vee \neg A\ \&\ \neg$C $\vee \neg B\ \&\ \neg C$

Пример 5.19. Логическое выражение $\neg(A \vee B)\ \&\ (A \vee C)\ \&\ (C \vee B)$ равносильно выражению

1) $\neg A \vee B \vee C$
2) $\neg A\ \&\ B\ \&\ C$
3) $\neg A\ \&\ B\ \&$ C $\vee A\ \&\ B \vee B\ \&\ C$
4) $A\ \&\ B \vee \neg A\ \&$ C $\vee B\ \&\ C$

ГЛАВА 6
ОСНОВЫ АЛГОРИТМИЗАЦИИ

Понятие алгоритма является одним из центральных в программировании. Существует множество его определений, данных известными учёными в разные времена.

Слово «алгоритм» происходит от имени математика Аль Хорезми, разработавшего в IX веке правила (алгоритм) для выполнения четырёх арифметических действий над десятичными числами. Впоследствии это слово стало собирательным названием для отдельных правил (последовательности действий) определённого вида, причём не только арифметических. В течение длительного времени его употребляли только математики, обозначая словом «алгоритм» пошаговые правила решения различных задач.

Будем считать, что **алгоритм** — это строго определённая последовательность действий для некоторого исполнителя, приводящая к конкретному результату за конечное число шагов.

Понятие алгоритма связано с понятием *исходных (входных) данных* и *искомого результата (выходных данных)*, поскольку назначением любого алгоритма является преобразование исходных данных в искомый результат.

Входные данные задаются до начала работы алгоритма или определяются динамически во время его работы. Число входных данных может быть равным нулю.

В результате выполнения алгоритма получают **выходные данные** — величины, имеющие определённую алгоритмом связь с входными данными.

Свойства алгоритмов

В данном пособии рассматриваются только свойства, описанные в большинстве учебников и учебных пособий, важные для решения задач итоговой аттестации.

- **Дискретность** — алгоритм представляет собой последовательность отдельных шагов, каждый из которых называется *командой*. Каждая команда начинает выполняться только после завершения выполнения предыдущей команды.
- **Понятность** — каждая команда алгоритма должна быть понятна тому, кто его исполняет (исполнителю). Необходимо знать, какие команды исполнителю известны, а какие — нет. Полный список команд, которые может выполнить исполнитель, называется *системой команд исполнителя* (СКИ). При составлении алгоритма для определённого исполнителя можно использовать лишь те команды , которые имеются в его СКИ.
- **Определённость (детерминированность)** — каждая команда алгоритма должна быть точно и однозначно определена, также должно быть однозначно определено, какая команда будет выполняться на следующем шаге. Результат выполнения команды

не должен зависеть от какой-либо дополнительной информации. Алгоритм не должен оставлять места для произвола исполнителя.
- **Конечность (результативность)** — алгоритм всегда должен заканчиваться после выполнения конечного числа шагов, при этом должен быть получен результат.
- **Массовость** — один и тот же алгоритм может применяться к разным наборам входных данных.

Разработка алгоритма — один из основных этапов решения различных задач на компьютере. Изучением общих свойств и закономерностей алгоритмов, их разработкой и анализом, формальными моделями их представления занимается теория алгоритмов.

Формальные исполнители алгоритма

Исполнителями алгоритма могут быть живые существа или технические устройства:
- машины: станки, роботы, бытовые приборы, компьютеры;
- животные: дрессированная собака, тигры и медведи в цирке;
- люди: ученик, токарь, кассир и т. д.

Исполнитель алгоритма способен выполнять команды, заданные алгоритмом. Животные и человек как исполнители отличаются тем, что могут понимать команды в различных вариантах, одни и те же команды выполнять по-разному, отказаться исполнять команду. Поэтому их называют *неформальными исполнителями*.

Формальный исполнитель может не понимать смысла алгоритма, но всё равно правильно выполнить его и получить нужный результат.

Формальный исполнитель алгоритма выполняет в строгой последовательности все предписанные алгоритмом команды, не вникая в содержание поставленной задачи, не задумываясь о её цели, результате и необходимости. Формальный исполнитель не привносит в алгоритм ничего нового и не отбрасывает никаких действий.

К формальным исполнителям относят автоматы, роботы и другие технические устройства, исполняющие инструкции (алгоритмы). На уроках информатики изучаются формальные исполнители Черепашка, Паркетчик, Робот, Вычислитель и др.

При записи алгоритмов для формального исполнителя используют команды, входящие в систему команд данного исполнителя. **Система команд формального исполнителя** (СКИ) — это совокупность команд, понятных и выполнимых конкретным исполнителем.

Среда — это «место обитания» исполнителя. Например, исполнитель Черепашка имеет свою определённую систему координат, а исполнитель Робот имеет клеточное поле. Расположение стен и закрашенных клеток, положение самого Робота задают конкретное состояние среды. **Среда исполнителя** — это совокупность объектов и связей между ними, над которыми данный исполнитель может выполнять команды.

Компьютер тоже можно считать формальным исполнителем, он имеет систему команд и может исполнять алгоритмы, записанные на языке программирования, т. е. программы.

Исполнение алгоритмов в среде формального исполнителя

В среде формального исполнителя алгоритмы, составленные с использованием СКИ, совершаются по шагам.

В следующих задачах в качестве примера исполнителя рассмотрим Графопостроитель, чертящий пером на бумаге.

Система команд Графопостроителя:

Команда	Действия
`поднять`	поднимает перо Графопостроителя
`опустить`	опускает перо Графопостроителя
`вперёд <шагов>`	проходит вперёд указанное число шагов
`поворот <градусов>`	осуществляет поворот на заданное число градусов против часовой стрелки
`повтор <раз> [<команда_1> ... <команда_N>]`	повторяет команды, стоящие в квадратных скобках, заданное количество раз

Исходное положение Графопостроителя: перо поднято.

Пример 6.1. Что будет нарисовано Графопостроителем в результате выполнения следующего фрагмента программы?

```
опустить
повтор 5 [вперёд 10 поднять вперёд 5]
```

Решение. При решении задач, требующих пошагового исполнения, в первую очередь нужно обратить внимание на ключевые моменты, изменяющие состояние пера: поднятие-опускание, поворот. В данном фрагменте программы присутствует команда `поднять`. Это значит, что следует внимательно отслеживать интервалы поднятого и опущенного пера и что, вероятно, линия не будет сплошной.

Первая команда опускает перо на бумагу, следующая команда вынуждает исполнителя пять раз повторить действия в квадратных скобках: пройти вперёд на 10, поднять перо, снова пройти вперёд на 5. В результате однократного исполнения этих действий на бумаге появится линия длиной 10. Повторения не изменят результата, поскольку в цикле отсутствует команда `опустить`. После выполнения алгоритма исполнитель будет находиться на расстоянии 75 от исходной точки.

✍ Модифицируйте алгоритм самостоятельно так, чтобы в результате его исполнения на бумаге появлялась:

а) пунктирная линия со штрихами длиной 10 и интервалами длиной 5;

б) штрихпунктирная линия с чередующимися штрихами длиной 10 и 2 и интервалами 4 между ними.

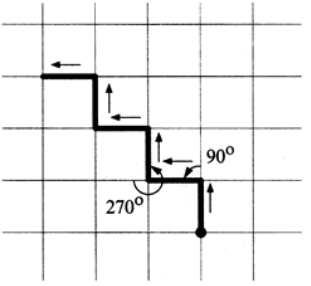
Постарайтесь рационально использовать команды исполнителя.

Пример 6.2. Что будет нарисовано Графопостроителем в результате выполнения следующего фрагмента программы?

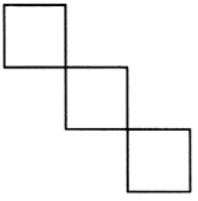
```
опустить
повтор 3 [вперёд 5 поворот 90 вперёд 5 поворот 270]
```

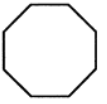
Решение. Выделим ключевые моменты. Перо единожды опущено и не поднимается, следовательно, перед нами предстанет сплошная линия. В циклической конструкции присутствуют повороты, за которыми мы будем внимательно следить. Линия, вероятно, будет ломаной.

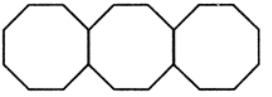
Как и в предыдущем примере, изучим, что именно происходит за один шаг цикла: из начального положения проводится линия длиной 5, осуществляется поворот на 90°, снова проводится линия длиной 5 и осуществляется поворот на 270°. Выполним алгоритм полностью, рисуя на бумаге изображение. То есть за один шаг перо Графопостроителя изобразит на бумаге два отрезка под углом 90° друг к друг. Три повторения цикла приведут к построению «лестницы» из трёх ступеней:

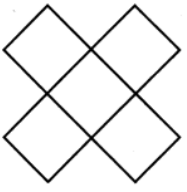


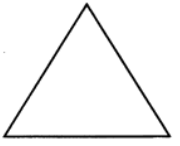
✍ Модифицируйте алгоритм самостоятельно, чтобы в результате его выполнения на бумаге появлялись следующие рисунки:

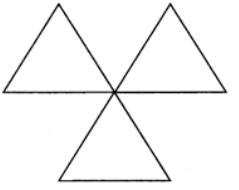
а) 

б) 

в) 

г) 

д) 

е) 

Пример 6.3. Задание с выбором одного ответа

Исполнитель Черепашка перемещается на экране компьютера, оставляя след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существуют две команды:

`Вперёд n` (где n — целое число), вызывающая передвижение Черепашки на n шагов в направлении движения.

`Направо m` (где m — целое число), вызывающая изменение направления движения на `m` градусов по часовой стрелке.

Запись `Повтори k [Команда 1 Команда 2 Команда 3]` означает, что последовательность команд в скобках повторится k раз.

Черепашке был дан для исполнения следующий алгоритм:

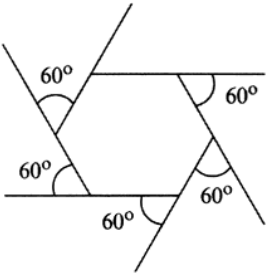
```
Повтори 6 [Направо 60 Вперёд 15]
```

Какая фигура появится на экране?

1) правильный пятиугольник
2) правильный треугольник
3) правильный шестиугольник
4) правильный двенадцатиугольник

Решение. Вспомним, что выпуклый многоугольник называется правильным, если у него все стороны равны и все углы равны.

Черепашка шесть раз повторяет группу команд, указанную в скобках. При этом она каждый раз поворачивает в одном направлении — направо, на один и тот же угол и прочерчивает отрезки одинаковой длины. После исполнения алгоритма Черепашка совершит поворот на $6 \cdot 60 = 360$ градусов. Известно, что сумма внешних углов многоугольника есть 360°. Следовательно, должна получиться замкнутая ломаная, образующая правильный многоугольник, состоящий из шести равных сторон.



Можно решить эту задачу иначе — выполнить все команды по шагам, как показано на рисунке. Будем считать, что исходное направление Черепашки совпадает с осью абсцисс, т. е. Черепашка смотрит на Восток (направо). Надо быть внимательными при выполнении поворотов. Поворот вправо на 60 градусов выполняется относительно текущего направления движения Черепашки, а не относительно осей координат.

Ответ: 3.

Формы записи алгоритма

Основные формы записи алгоритма:
1) словесная, на естественном языке;
2) графическая, например в виде блок-схем;
3) на алгоритмическом языке (псевдокоде).

1) Самым простым способом является *словесная запись* алгоритма *на естественном языке*. Если формальным исполнителем алгоритма является человек, то запись алгоритма естественным языком может быть приемлема. Но при изложении алгоритма на естественном языке появляется опасность неточного понимания команд исполнителем, т. е. нарушается свойство определённости. В приведённых ниже фрагментах текста один и тот же глагол **«кликать»** имеет различные значения. Формальный исполнитель мог бы истолковать стихотворение Пушкина, как команду работы с мышью.

Вот пошёл он к синему морю;
Видит,— море слегка разыгралось.
Стал он **кликать** золотую рыбку,
Приплыла к нему рыбка и спросила:
«Чего тебе надобно, старче?»
(А. С. Пушкин.
Сказка о рыбаке и рыбке)

Для того чтобы развернуть окно, дважды **кликните** левой кнопкой мыши на заголовке.

Ещё одним примером неточности понимания является история о программисте. Однажды программист пошёл в магазин с заданием: «Купи батон колбасы, если будут яйца — возьми десяток». Программист вернулся домой с десятью батонами колбасы, потому что в магазине были яйца.

Если речь идёт о формальном исполнителе, например роботе или автомате, то естественного языка он может не понимать вовсе.

Рассмотрим пример задания, в котором алгоритм описан на естественном языке.

Пример 6.4. Задание с кратким ответом

Алгоритм из одной цепочки символов получает новую цепочку следующим образом. Сначала в обратном порядке записываются буквы исходной цепочки символов, потом две последние буквы исходной цепочки символов в прямом порядке и, наконец, первая буква исходной цепочки символов. Получившаяся цепочка является результатом работы алгоритма. Например, если исходная цепочка символов была **СОН**, то результатом алгоритма будет цепочка **НОСОНС**.

Дана цепочка символов **ДНО**. Какая цепочка символов получится, если к данной цепочке применить алгоритм дважды? (Т. е. алгоритм применяется к данной цепочке, а затем к результату его работы.)

Русский алфавит:
А Б В Г Д Е Ё Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я

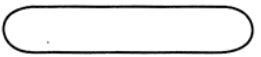
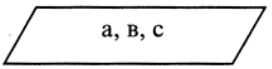
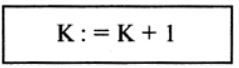
Решение. Выполним алгоритм, строго соблюдая последовательность действий.

№	Действие	Результат
	Первое выполнение алгоритма. Входные данные	**ДНО**
1	Записать буквы в обратном порядке	**ОНД**
2	Дописать две последние буквы исходной цепочки в исходном порядке	**ОНДНО**
3	Дописать первую букву исходной цепочки	**ОНДНОД**
	Второе выполнение алгоритма. Входные данные	**ОНДНОД**
1	Записать буквы в обратном порядке	**ДОНДНО**
2	Дописать две последние буквы исходной цепочки в исходном порядке	**ДОНДНООД**
3	Дописать первую букву исходной цепочки	**ДОНДНООДО**
	Результат	**ДОНДНООДО**

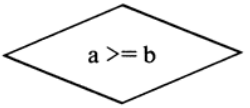
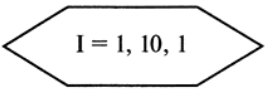
Ответ: ДОНДНООДО.

2) Для *графического представления* алгоритмов в виде **блок-схем** используются стандартные обозначения элементов, некоторые из них приведены в таблице.

Обозначения элементов блок-схем

Обозначение и пример заполнения	Пояснение
1	2
	начало или конец алгоритма
	блок ввода / вывода данных
	блок обработки информации

Продолжение таблицы

1	2
	логический блок (проверка истинности или ложности логического выражения (условия) и выбор следующего блока)
	организация циклического процесса — заголовок цикла с параметром

Между собой блоки соединяются линиями переходов. По умолчанию блоки выполняются сверху вниз слева направо. Если последовательность выполнения должна быть другой, используются стрелки.

3) Для представления алгоритмов были разработаны *специальные алгоритмические языки* — разновидности формальных языков. Они позволяют наглядно, просто и в удобной форме отображать содержательный смысл алгоритма и проводить его анализ. По алгоритмам, представленным на алгоритмическом языке, можно составить программу на любом языке программирования.

Алгоритмический язык — это система обозначений, предназначенных для записи алгоритмов.

В алгоритмическом языке используются формальные конструкции, но нет строгих синтаксических правил для записи команд.

В алгоритмическом языке есть служебные слова. Они имеют вполне определённый смысл и в печатном тексте выделяются жирным шрифтом, а в рукописном — подчёркиванием. Можно встретить различные алгоритмические языки, отличающиеся набором служебных слов и формой записи основных конструкций. В данном пособии мы будем использовать школьный алгоритмический язык (школьный АЯ).

Основные служебные слова школьного АЯ

Общие	`алг` (алгоритм); `арг` (аргумент); `рез` (результат); `нач` (начало алгоритма); `кон` (конец алгоритма)
Операции ввода и вывода	`ввод; вывод`
Описание типов данных	`цел` (целый); `вещ` (вещественный); `сим` (символьный); `лит` (литерный); `лог` (логический)

Продолжение таблицы

Операции целочисленного деления	`div; mod`
Логические операции	`и; или; не`
Логические значения	`да; нет`
Организация ветвления	`если; то; иначе; выбор; при; все`
Организация цикла	`пока; для; от; до; нц; кц`

При описании алгоритмического языка и его конструкций будем использовать следующие соглашения:

- если что-то записано в угловых скобках, например `<команда>`, `<аргумент>`, значит, при составлении алгоритма на это место следует записать конкретную команду, аргумент и т. п.
- если в записи используется конструкция в квадратных скобках [ ], то эта конструкция необязательна.

В общем виде алгоритм обычно выглядит так:

```
алг <название алгоритма> (<аргументы>, <результаты>)
нач
        <команды>
кон
```

Здесь `<аргументы>` — это входные данные, `<результаты>` — выходные данные алгоритма. Для их описания используются служебные слова `арг` и `рез`, после которых записывается тип значения входных (выходных) данных (`цел`, `вещ`, `лог` и т. д.) и наименования переменных. Например:

алг `Площадь_круга (арг вещ R, рез вещ S);`

алг `Задача (арг цел a, b, арг вещ c, рез вещ d).`

В первом примере входная переменная — радиус окружности — и результат вычислений — площадь круга — имеют вещественный тип.

Во втором примере на вход алгоритму подаются два аргумента целого типа, один аргумент вещественного типа. В результате работы алгоритма получаем одно выходное значение вещественного типа.

Переменные в алгоритмах

Для хранения результатов промежуточных вычислений в процессе выполнения алгоритма входных и выходных данных и другой информации используются *переменные*. Если провести аналогию с повседневностью, можно представить переменную в виде ящика, в котором хранятся какие-то вещи (см. рис. 12). Сами «ящики» — это аналоги **переменных**, для их различения используют «этикетки» — **имена переменных**.

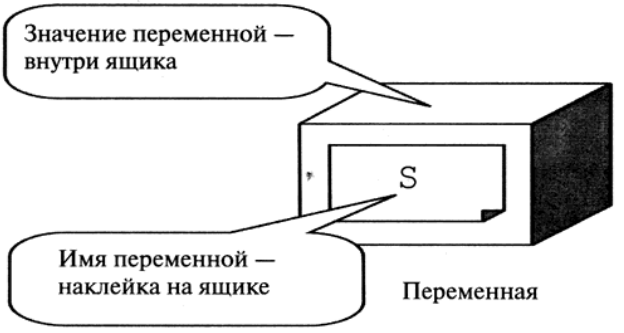


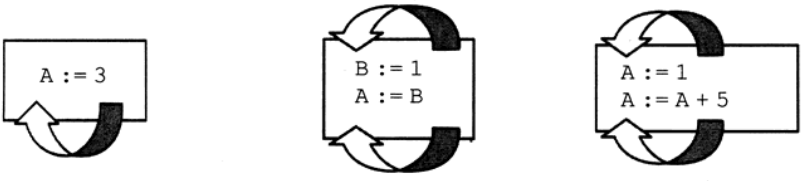
Рис. 12.

Команда присваивания

Для того чтобы сохранить в переменную («положить в ящик») некоторое значение, применяют **команду присваивания**. В данном пособии в качестве команды присваивания мы будем использовать конструкцию школьного АЯ «:=». Однако следует понимать, что в системах команд других формальных исполнителей команда присваивания может записываться иначе.

В школьном АЯ в левой части команды присваивания может находиться только имя переменной. В правой части может быть записано значение (константа), переменная или выражение. (Константа и переменная являются частным случаем выражения.)

Команда присваивания действует справа налево, т. е. значение, стоящее справа от знака «:=», сохраняется в переменную, имя которой указано слева от знака «:=». Значение переменной справа будет скопировано в переменную слева. В случае, когда справа стоит выражение, вычисляется его значение и результат присваивается переменной, имя которой указано слева.



а) б) в)

Рис. 13.

На рис. 13а переменной `A` присваивается значение числовой константы 3. На рис. 13б сначала переменной `B` присваивается значение 1, а затем переменной `A` присваивается значение переменной `B`. На рис. 13в переменной `A` присваивается значение 1, затем результат вычисления выражения `A` + 5, т. е. 1 + 5 = 6.

В общем случае можно сказать, что команда присваивания используется для вычисления выражения, записанного в правой части, и присваивания результата вычислений переменной, имя которой записано в левой части:

```
<имя_переменной> := <выражение>
```

Выражение — это формула, по которой вычисляется значение. Выражение может состоять из операндов, знаков операций и круглых скобок. В алгоритмических языках выражение записывается в строку.

Операндами являются константы, переменные и обращения к функциям.

Для обозначения операций в школьном АЯ используют следующие символы и слова:

+	операция сложения
−	операция вычитания
*	операция умножения
/	операция деления
div	операция целочисленного деления, результатом является целая часть частного
mod	операция получения остатка целочисленного деления
не, и, или	логические операции, выполняются в соответствии с таблицами истинности
операции сравнения, или операции отношения:	
>	больше
<	меньше
>=	больше или равно
<=	меньше или равно
=	равно
< >	неравно

Довольно часто школьники ошибаются при использовании операций целочисленного деления `div` и `mod` – вместо целочисленного остатка от деления используют десятичную часть частного. Запомнить, как выполняются эти операции, вам поможет простой пример. Если вы покупаете конфеты по 17 рублей за одну штуку, даёте продавцу 40 рублей и просите дать конфеты «на всю сумму», то получите `div (40, 17) = 2` конфеты. Ваша сдача составит `mod (40, 17) = 6` рублей. Если же вы попытаетесь купить конфеты, имея 15 рублей, то получите `div (15, 17) = 0` конфет, а сдача составит `mod (15, 17) = 15` рублей.

Операции в выражении выполняются в соответствии с приоритетами, определёнными правилами языка. В школьном АЯ приняты следующие приоритеты выполнения операций:

1	не
2	*, /, div, mod, и
3	+, −, или
4	>, <, >=, <=, =, <>

В первую очередь выполняются операции с меньшим приоритетом. Операции с одинаковым приоритетом выполняются слева направо, если порядок выполнения не указан явно круглыми скобками. Если операции записаны в скобках, они выполняются в первую очередь с учётом приоритетов. Вычисление значения выражения с вложенными скобками начинается с внутренних скобок.

Примеры вычислений выражений

Выражение	Вычисления
Y := 8 + 3 · 5	8 + 3 · 5 = 8 + 15 = 23
A := 12 : 2 · 3	12 : 2 · 3 = 6 · 3 = 18
B := 32 : 4 : 2	32 : 4 : 2 = 8 : 2 = 4
C := 35 : (3 + 12 : 3)	35 : (3 + 12 : 3) = 35 : (3 + 4) = 35 : 7 = 5
D := ((1 + 2) · 5 + 2) × × 5 + 3	((1 + 2) · 5 + 2) · 5 + 3 = (3 · 5 + 2) · 5 + 3 = = 17 · 5 + 3 = 88

Команды ввода и вывода

Для ввода и вывода данных в школьном АЯ используются команды:
ввод <список_ввода>;
вывод <список_вывода>.

Списки ввода и вывода состоят из элементов, которые перечисляются **через запятую**.

В списке ввода могут присутствовать только имена переменных. При выполнении команды ввод алгоритм получает данные, которые записываются в соответствующие переменные.

В списке вывода могут быть перечислены константы, имена переменных и выражения.

Текстовые константы записываются в списке вывода **в кавычках**, а выводятся (например, на экран или принтер) **без кавычек**. Числовые константы выводятся без изменений. Если в списке вывода указана переменная, то выводится её значение. Если в списке вывода указано выражение, выводится результат его вычисления.

Пример записи алгоритма:

```
алг  гипотенуза (арг вещ a, b, рез вещ c)
нач
     ввод a, b
     c := a*a+b*b
     вывод "катеты" ,a,", ",b," гипотенуза ",c
кон
```

Если при выполнении команды `ввод` ввести, например, числа 3 и 4, они сохранятся соответственно в переменных `a` и `b`. При выполнении команды `вывод` будет выведено следующее:

```
катеты 3, 4  гипотенуза 5
```

Если вместо переменной `c` в список вывода записать `a * a + b * b`, будет сформирован тот же вывод.

Как бы ни был записан алгоритм — на языке формального исполнителя, на естественном языке или при помощи блок-схемы,— он составляется с использованием *базовых (основных) алгоритмических структур*.

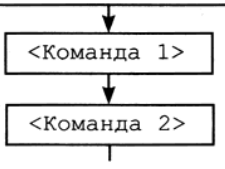
Базовые алгоритмические структуры

Выделяют три базовые алгоритмические структуры (конструкции) — *линейная (следование)*, *ветвление* и *цикл*, из которых можно построить любой алгоритм. **Каждая алгоритмическая структура имеет одну точку входа и одну точку выхода.**

Будем записывать структуры в виде блок-схем и школьного АЯ.

1. **Линейная структура** является самой простой организацией алгоритмов — команды выполняются последовательно одна за другой.

Линейная структура

Блок-схема	Школьный АЯ
	... <Команда 1> <Команда 2> ...

2. **Структура «ветвление»**. Решение некоторых задач требует различных действий в зависимости от выполнения некоторых условий. В таких случаях говорят о ветвлении алгоритма.

Для реализации структуры «ветвление» используются две структурированные команды * школьного АЯ — `если` и `выбор`, каждая из которых может быть полной и неполной. Полная команда `если` имеет вид `если...то...иначе`, неполная — `если...то`. Полная команда `выбор` имеет вид `выбор...иначе`.

В блок-схемах и школьном АЯ `<условие>` — это логическое выражение, результатом которого может быть одно из двух возможных значений — **истина** или **ложь**. В школьном АЯ эти значения записывают как `да` и `нет`. В языках программирования часто используются значения True и False. В компьютере эти значения хранятся как 1 и 0.

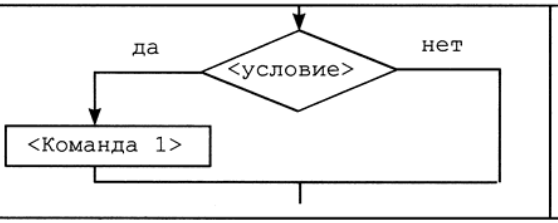
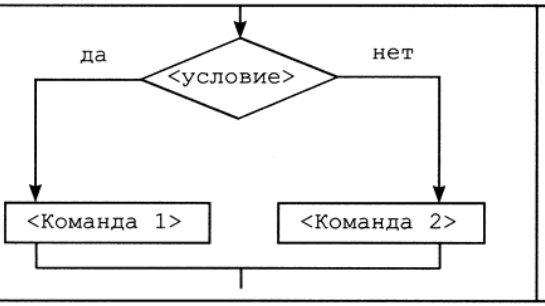
---

* Структурированная команда включает в себя несколько команд.

Полная форма команды если определяет две ветви команд: первая выполняется в случае истинности условия, вторая — в случае его ложности. В каждой ветви может выполняться не одна команда, а серия команд.

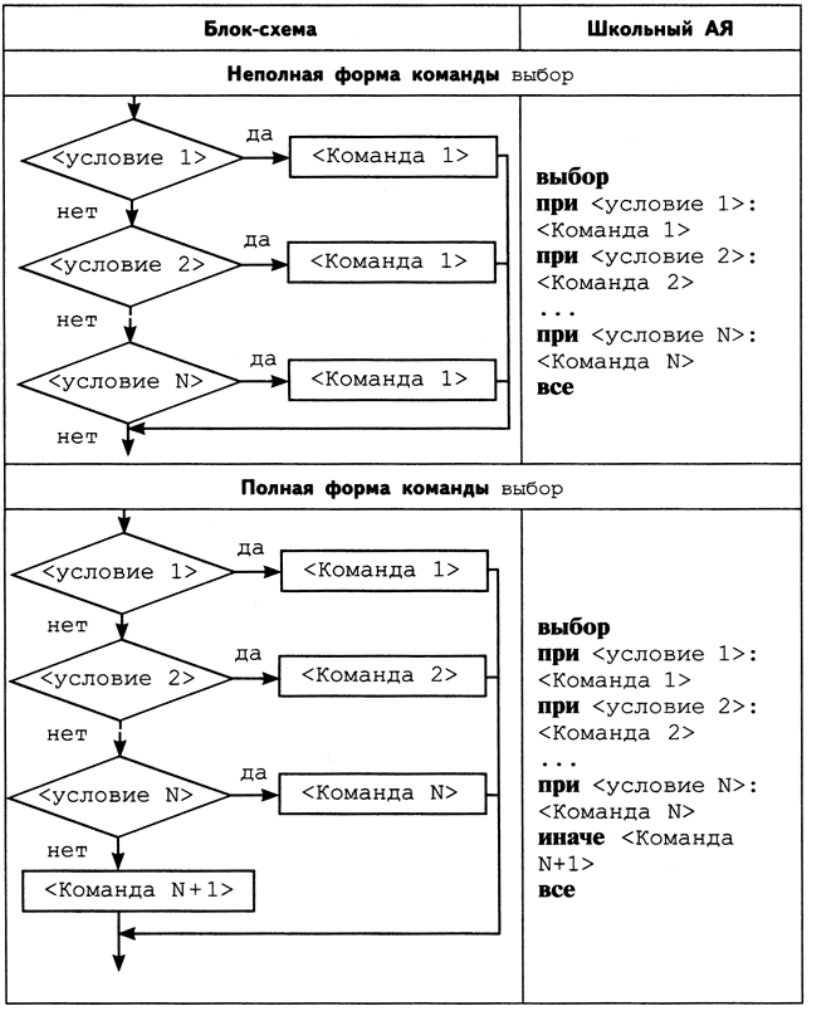
Неполная форма команды если предполагает, что при истинности условия выполняется команда 1 (серия команд), в противном случае никакие действия не выполняются.

Структура «ветвление». Команда если

Блок-схема	Школьный АЯ
Неполная форма команды если	
	**если** <условие> **то** <Команда 1> **все**
Полная форма команды если	
	**если** <условие> **то** <Команда1 > **иначе** <Команда 2> **все**

Команда выбор используется для организации множественного ветвления.

Структура «ветвление». Команда выбор

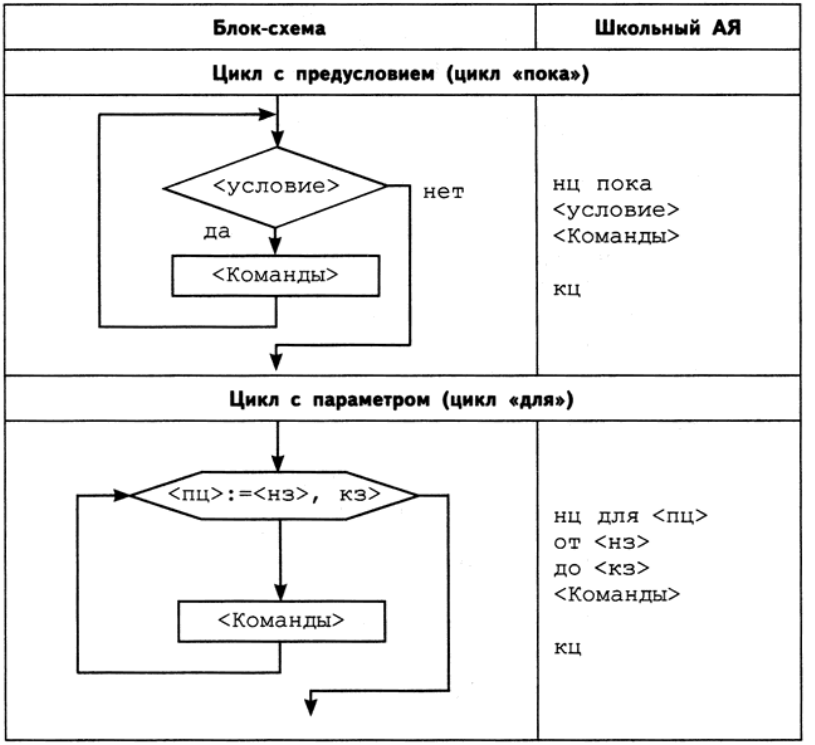


3. **Циклическая структура (цикл)** обеспечивает многократное выполнение одних и тех же команд. Существует несколько разновидностей циклических структур. Рассмотрим две: циклическую структуру **с предусловием**, её ещё называют циклом **«пока»**, и циклическую структуру **с параметром**, её называют циклом **«для»**.

Любая циклическая структура состоит из двух частей — *заголовка* и *тела цикла*. Набор команд, повторяющихся при выполнении цикла, называют **телом цикла**. **Заголовок** определяет количество повторений тела цикла.

В цикле с предусловием заголовок `пока <условие>`. Выполнение тела цикла (`<Команды>`) будет повторяться до тех пор, пока условие истинно.

Циклические структуры



В цикле с параметром заголовок цикла — `для <пц> от <нз> до <кз>`. Здесь `<пц>` — параметр цикла (это, как правило, переменная целого типа), `<нз>` — начальное значение параметра цикла, `<кз>` — конечное значение параметра цикла.

Цикл с параметром выполняется следующим образом:

1) параметр цикла принимает начальное значение;
2) если параметр цикла не превышает конечного значения, выполняется тело цикла, иначе — выход из цикла, переход к следующей команде алгоритма.
3) параметр цикла увеличивается на единицу;
4) переход к пункту 2.

Тело цикла с параметром выполняется (`<кз> - <нз> + 1`) раз.

Пошаговое выполнение алгоритмов. Трассировочные таблицы

Наиболее простым способом отслеживания действий, предписанных алгоритмом, является его пошаговое исполнение. Например, на бумаге можно последовательно записать результаты каждого действия.

Для того чтобы наглядно представлять значения переменных, изменяющихся при выполнении алгоритма в сложных алгоритмических структурах, например циклах, составляют **трассировочные таблицы,** которые называют также **таблицами значений**.

Трассировочные таблицы могут быть двух видов:

1) пошаговое выполнение каждого действия с записью результата в строку таблицы;
2) выполнение группы действий с записью результатов в одну строку для всей группы.

Рассмотрим составление трассировочной таблицы первого вида. В заголовке таблицы помещают имена всех переменных, используемых в алгоритме. В отдельном столбце записывают команды и логические выражения (условия), которые выполняются. Каждая строка таблицы соответствует одному шагу алгоритма, при котором изменяются значения переменных или выражений.

Как правило, в таблицу заносят только те значения, которые получены на очередном шаге. Если значение какой-либо переменной не меняется, его в таблицу не записывают, чтобы не загромождать её. Подразумевается, что в переменной находится значение, записанное в ближайшей сверху строке соответствующего столбца.

Составление трассировочных таблиц для линейных структур

Попытаемся решить задачу: поменять значения двух переменных `A` и `B`. Запишем команды и результаты их выполнения в трассировочную таблицу:

Команда	A	B
`A := 3`	3	
`B := 5`		5
`A := B`	5	
`B := A`		5
Результат	5	5

Как видно из таблицы, задача не решена, значение 3, находившееся в переменной `A`, утеряно. Надо его сохранить, прежде чем присваивать переменной `A` значение переменной `B`. Введём дополнительную переменную `C`. Такую переменную, предназначенную для временного хранения значения, называют *буферной*.

Команда	A	B	C
A := 3	3		
B := 5		5	
C := A			3
A := B	5		
B := C		3	
Результат	5	3	

Требуемый результат получен.

Решим эту же задачу без использования дополнительной переменной.

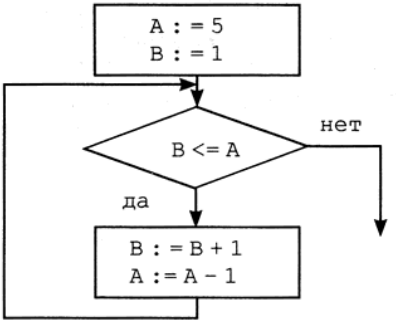
```
A := 3
B := 5
A := A + B
B := A - B
A := A - B
```

№ шага	Команда	Вычисление правой части команды присваивания	A	B
1	A := 3	3	3	
2	B := 5	5		5
3	A := A + B	A + B = 3 + 5 = 8	8	
4	B := A - B	A - B = 8 - 5 = 3		3
5	A := A - B	A - B = 8 - 3 = 5	5	
Результат			5	3

Задача решена. Из этого несложного примера видно, что для одной задачи можно составить несколько алгоритмов, которые отличаются количеством используемых переменных, команд и операций.

Составление трассировочных таблиц для циклических конструкций

Задан фрагмент алгоритма, содержащего базовую структуру цикла с предусловием.

Школьный АЯ	Блок-схема
A := 5 B := 1 пока B <= A нц B := B + 1 A := A − 1 кц	

Определим,

- сколько раз выполняется тело цикла;
- значения переменных A и B после его завершения.

Построим трассировочную таблицу первого вида.

№ шага	Команда или условие (логическое выражение)	Вычисление правой части команды присваивания или условия (логического выражения)	A	B
1	A : = 5	5	5	
2	B : = 1	1		1
3	B <= A	(1 <= 5) = да		
4	B : = B + 1	1 + 1 = 2		2
5	A : = A − 1	5 − 1 = 4	4	
6	B <= A	(2 <= 4) = да		
7	B : = B + 1	2 + 1 = 3		3
8	A : = A − 1	4 − 1 = 3	3	
9	B <= A	(3 <= 3) = да		
10	B : = B + 1	3 + 1 = 4		4
11	A : = A − 1	3 − 1 = 2	2	
12	B <= A	(4 <= 2) = нет		
Результат			2	4

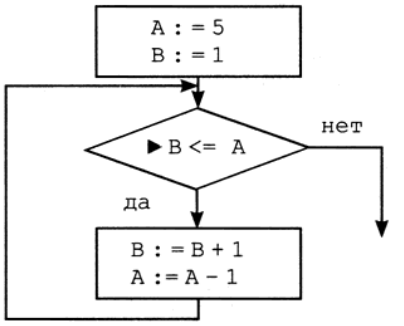
Из таблицы видно, что тело цикла выполнилось трижды (строки 4—5, 7—8, 10—11), переменные приняли значения A = 2, B = 4.

Даже новичок легко построит трассировочную таблицу первого вида, но она может получиться довольно длинной. Более компактный вид имеют трассировочные таблицы второго вида, но для их построения требуется некоторый опыт трассировки алгоритмов, навык строгого соблюдения последовательности действий.

При составлении таблиц второго вида используются **контрольные точки**.

Контрольную точку устанавливают на выбранной пользователем строке алгоритма. Выполнение алгоритма продолжается до контрольной точки и приостанавливается на отмеченной строке. В трассировочную таблицу записываются текущие значения переменных и выражений. Если значение переменной не изменилось, его можно не вносить в таблицу. При необходимости можно поставить несколько контрольных точек.

Поставим **контрольную точку** (►) на заголовке цикла пока B <= A.

Школьный АЯ	Блок-схема
A := 5 B := 1 ► пока B <= A нц B := B + 1 A := A - 1 кц	

Строки составленной таблицы отражают состояние переменных и выражений программы в контрольных точках (кт):

№ кт	B <= A	A	B
1	да	5	1
2	да	4	2
3	да	3	3
4	нет	2	4
Результат		2	4

Из таблицы видно, что цикл выполнится три раза (выражение, определяющее вход в цикл, три раза принимает значение да), и значения переменных равны A = 2, B = 4.

Пример разработки алгоритма с использованием команд ветвления и цикла

Постановка задачи: найти наибольший общий делитель D двух натуральных чисел X и Y.

Входные данные: натуральные числа X и Y.

Выходные данные: натуральное число D.

Методы решения:

Метод 1 — разложение чисел X и Y на простые множители.

Шаг 1. Представим каждое число как произведение его простых множителей, например:

$$420 = 2 \cdot 2 \cdot 3 \cdot 5 \cdot 7,$$

$$162 = 2 \cdot 3 \cdot 3 \cdot 9.$$

Шаг 2. Запишем степени всех простых множителей:

$$420 = 2 \cdot 2 \cdot 3 \cdot 5 \cdot 7 = 2^2 \cdot 3^1 \cdot 5^1 \cdot 7^1,$$

$$162 = 2 \cdot 3 \cdot 3 \cdot 9 = 2^1 \cdot 3^2 \cdot 9^1.$$

Шаг 3. Выпишем все общие делители (множители) этих чисел. Это числа 2 и 3.

Шаг 4. Выберем наименьшую степень каждого из них, встретившуюся во всех произведениях. Это 2^1 и 3^1.

Шаг 5. Перемножим эти степени: НОД $= 2^1 \cdot 3^1 = 6$.

Этот метод достаточно сложный для алгоритмической реализации, так как не все шаги являются элементарными действиями. Шаги 1 и 2 являются отдельными задачами, состоящими из многих простых действий.

Метод 2 — алгоритм Евклида. Пусть A и B одновременно не равные нулю целые неотрицательные числа, и пусть $A >= B$. Тогда если $B = 0$, то НОД $(A, B) = A$, а если $B \neq 0$, то для чисел A, B и R, где R — остаток от деления A на B, выполняется равенство НОД (A, B) = НОД (B, R). Алгоритм выполняется по шагам следующим образом:

Шаг 1. Задать числа X и Y.

Шаг 2. Присвоить большее из этих чисел переменной A, меньшее — переменной B.

Шаг 3. Пока $B \neq 0$ выполнить:
найти остаток R от деления A на B;
заменить A на B, B на R.

Шаг 4. Присвоить переменной D значение переменной A.

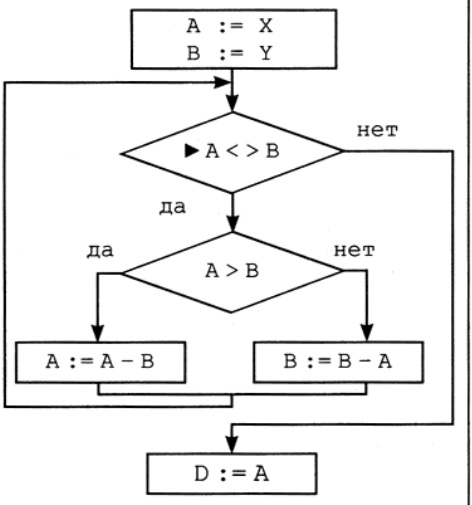
Шаг 5. Конец.

Метод 3 также является реализацией алгоритма Евклида. Операция определения остатка от деления заменяется последовательным вычитанием.

Шаг 1. Задать числа X и Y.
Шаг 2. Присвоить $A := X$, $B := Y$.
Шаг 3. Пока $A \neq B$ выполнить:
если $A > B$, то присвоить $A := A - B$,
иначе присвоить $B := B - A$.
Шаг 4. Присвоить переменной D значение переменной A.
Шаг 5. Конец.

Составление алгоритма решения

Составим алгоритм решения задачи в виде блок-схемы и на школьном АЯ для третьего метода.

Школьный АЯ	Блок-схема
ввод X, Y A := X B := Y ▶ пока A < > B если A > B то A := A – B иначе B := B – A все кц D := A	

Проверка работоспособности программы, включающая её тестирование

Пусть $X = 48$, а $Y = 18$, НОД (48 18) = 6.

Составим трассировочную таблицу первого типа и найдём наибольший общий делитель.

№ шага	Команда или условие (логическое выражение)	Вычисление правой части команды присваивания или условия (логического выражения)	A	B
1	A := X	48	48	
2	B := Y	18		18
3	A <> B	(48 <> 18) = да		
4	A > B	(48 > 18) = да		
5	A := A – B	48 – 18 = 30	30	
6	A <> B	(30 <> 18) = да		
7	A > B	(30 > 18) = да		
8	A := A – B	30 – 18 = 12	12	
9	A <> B	(12 <> 18) = да		
10	A > B	(12 > 18) = нет		
11	B := B – A	18 – 12 = 6		6
12	A <> B	(12 <> 6) = да		
13	A > B	(12 > 6) = да		
14	A := A – A	12 – 6 = 6	6	
15	A <> B	(6 <> 6) = нет		
16	D := A	6		
Результат: D = 6				

Покажем пример составления трассировочной таблицы второго типа для этого примера. Положение контрольной точки показано в блок-схеме и в записи алгоритма на школьном АЯ.

№ кт	A <> B	A > B	A	B	D
1	да		48	18	
2	да	да	30		
3	да	да	12		
4	да	нет		6	
5	нет	да	6		
6					6

✍ Самостоятельно составьте трассировочные таблицы для других исходных данных.

Разработайте блок-схему и алгоритм на школьном АЯ для второго метода решения задачи нахождения НОД и составьте трассировочные таблицы для различных исходных данных.

Пример 6.5. Задание с кратким ответом

В алгоритме, записанном ниже, используются переменные a и b и следующие операции:

:= — присваивание;

mod — остаток от деления первого операнда на второй.

Определите значение переменной a после выполнения данного алгоритма:

```
a := 44
b := mod (a, 10)
a := a - b
```

В ответе укажите одно число — значение переменной a.

Решение. Проследим изменения значения переменной:

1) a = 44;
2) b = mod (a, 10) = mod (44, 10) = 4;
3) a = a - b = 44 - 4 = 40.

Ответ: 40.

Пример 6.6. Задание с кратким ответом

Во фрагменте алгоритма, записанном ниже, используются переменные a и b и операция присваивания :=. Определите начальное значение переменной a, если после выполнения данного фрагмента алгоритма её значение стало равно 18.

```
b := a * 2 - 10;
a := a - b.
```

В ответе укажите одно число — начальное значение переменной a.

Решение. Трассировочные таблицы можно составлять для конкретных значений исходных данных и в общем виде. Обозначим x — начальное значение переменной a. Проследим изменения значения переменных в трассировочной таблице:

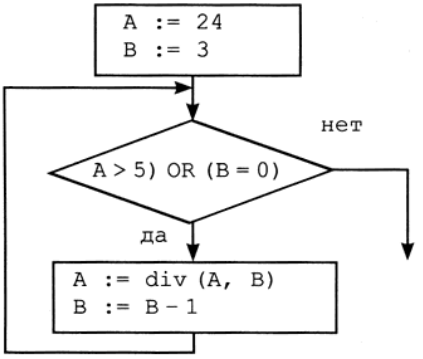
Команда	a	b
b := a * 2 - 10	x	2 * x - 10
a := a - b	x - (2 * x - 10) = 10 + x	

По условию задачи в результате выполнения фрагмента алгоритма значение переменной a стало равно $10 + x = 18$. Тогда $x = 8$.

Ответ: 8.

Пример 6.7. Задание с кратким ответом

Определите значение переменной B после выполнения фрагмента алгоритма, представленного блок-схемой:



Знаком := обозначена операция присваивания;

OR — операция логического ИЛИ;

div – операция целочисленного деления первого операнда на второй.

В ответе укажите одно число — значение переменной B.

Решение. Составим трассировочную таблицу первого вида:

№ шага	Команда или условие (логическое выражение)	Вычисление правой части команды присваивания или условия (логического выражения)	A	B
1	A := 24	24	24	
2	B := 3	3		3
3	(A > 5) OR (B = 0)	(24 > 5) OR (3 = 0) = да OR нет = да		
4	A := div (A, B)	div (24, 3) = 8	8	
5	B := B - 1	3 - 1 = 2		2
6	(A > 5) OR (B = 0)	(8 > 5) OR (2 = 0) = да OR нет = да		
7	A := div (A, B)	div (8, 2) = 4	4	
8	B := B - 1	2 - 1 =1		1
9	(A > 5) OR (B = 0)	(4 > 5) OR (1 = 0) = нет OR нет = нет		
Результат			4	1

Ответ: 1.

Пример 6.8. Задание с кратким ответом

Сколько раз выполнится тело цикла во фрагменте алгоритма, представленном блок-схемой в предыдущем примере (см. пример 6.7):

Решение. Установим контрольную точку на логическом выражении `(A > 5) OR (B = 0)` и составим трассировочную таблицу второго вида:

№ шага	`(A > 5) OR (B = 0)`	A	B
1		24	3
2	да	8	2
3	нет	4	**1**

Тело цикла выполнилось дважды, так как условие входа в цикл `(A > 5) OR (B = 0)` два раза приняло значение да.

Ответ: 2.

Задания для самостоятельного решения

Задания с выбором одного ответа

Пример 6.9. Исполнитель Черепашка перемещается на экране компьютера, оставляя след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существуют две команды:

`Вперёд n` (где *n* — целое число), вызывающая передвижение Черепашки на *n* шагов в направлении движения.

`Направо m` (где *m* — целое число), вызывающая изменение направления движения на *m* градусов по часовой стрелке.

Запись `Повтори k [Команда 1 Команда 2 Команда 3]` означает, что последовательность команд в скобках повторится *k* раз.

Черепашке был дан для исполнения следующий алгоритм:

```
Повтори 8 [Направо 45 Вперёд 18 Направо 45]
```

Какая фигура появится на экране?

1) восьмиконечная звезда
2) правильный восьмиугольник
3) незамкнутая ломаная линия
4) квадрат

Пример 6.10. Исполнитель Черепашка перемещается на экране компьютера, оставляя след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существуют две команды:

`Вперёд n` (где *n* — целое число), вызывающая передвижение Черепашки на *n* шагов в направлении движения.

`Направо m` (где *m* — целое число), вызывающая изменение направления движения на *m* градусов по часовой стрелке.

Запись `Повтори k [Команда 1 Команда 2 Команда 3]` означает, что последовательность команд в скобках повторится *k* раз.

Черепашке был дан для исполнения следующий алгоритм:

`Повтори 5 [Вперёд 36 Направо 36 Вперёд 36 Направо 36]`

Какая фигура появится на экране?

1) правильный треугольник
2) равнобедренная трапеция
3) правильный пятиугольник
4) правильный десятиугольник

Пример 6.11. Исполнитель Черепашка перемещается на экране компьютера, оставляя след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существуют две команды:

`Вперёд n` (где *n* — целое число), вызывающая передвижение Черепашки на *n* шагов в направлении движения.

`Направо m` (где *m* — целое число), вызывающая изменение направления движения на *m* градусов по часовой стрелке.

Запись `Повтори k [Команда 1 Команда 2 Команда 3]` означает, что последовательность команд в скобках повторится *k* раз.

Черепашке был дан для исполнения следующий алгоритм:

`Повтори 4 [Направо 45 Вперёд 25 Направо 75]`

Какая фигура появится на экране?

1) ромб
2) правильный треугольник
3) незамкнутая ломаная линия
4) правильный восьмиугольник

Пример 6.12. Исполнитель Черепашка перемещается на экране компьютера, оставляя след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существуют две команды:

`Вперёд n` (где *n* — целое число), вызывающая передвижение Черепашки на *n* шагов в направлении движения.

`Направо m` (где *m* — целое число), вызывающая изменение направления движения на *m* градусов по часовой стрелке.

Запись `Повтори k [Команда 1 Команда 2 Команда 3]` означает, что последовательность команд в скобках повторится *k* раз.

Черепашке был дан для исполнения следующий алгоритм:

`Повтори 7 [Направо 33 Вперёд 10 Направо 39]`

Какая фигура появится на экране?

1) правильный семиугольник
2) правильный треугольник
3) правильный пятиугольник
4) правильный четырнадцатиугольник

Пример 6.13*. Исполнитель Черепашка перемещается на экране компьютера, оставляя след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существуют две команды:

`Вперёд n` (где n — целое число), вызывающая передвижение Черепашки на n шагов в направлении движения.

`Направо m` (где m — целое число), вызывающая изменение направления движения на m градусов по часовой стрелке.

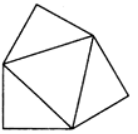
Запись `Повтори k [Команда 1 Команда 2 Команда 3]` означает, что последовательность команд в скобках повторится k раз.

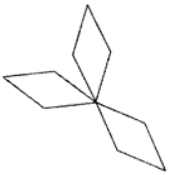
Черепашке был дан для исполнения следующий алгоритм:

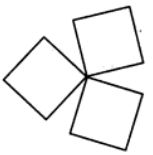
```
Повтори 3 [Повтори 4 [Вперёд 30 Направо 50 Вперёд 30
Направо 130] Направо 130]
```

Какая фигура появится на экране?

1) 

3) 

2) 

4) 

Задания с кратким ответом

Пример 6.14. В алгоритме, записанном ниже, используются целочисленные переменные `a` и `b`, а также следующие операции:

`:=` — присваивание;

`+` — сложение;

`*` — умножение;

`/` — целочисленное деление.

Определите значение переменной `a` после исполнения данного алгоритма:

```
a := 12
b := 15 + a / 4
a := b * a / 9
```

Порядок действий соответствует правилам арифметики.

В ответе укажите одно число — значение переменной `a`.

Пример 6.15. В алгоритме, записанном ниже, используются целочисленные переменные `a` и `b`, а также следующие операции:

`:=` — присваивание;

`-` — вычитание;

`*` — умножение;

`/` — целочисленное деление.

Определите значение переменной `a` после исполнения данного алгоритма:

```
a := 7
b := 36 - a * 4
a := 40 / b * a
```

Порядок действий соответствует правилам арифметики.

В ответе укажите одно число — значение переменной a.

Пример 6.16. В алгоритме, записанном ниже, используются целочисленные переменные a и b, а также следующие операции:

:= — присваивание;

– — вычитание;

* — умножение;

/ — целочисленное деление.

Определите значение переменной a после исполнения данного алгоритма:

```
a := 9
b := a * 12 - 10
a := b / 6 * a/3
```

Порядок действий соответствует правилам арифметики.

В ответе укажите одно число — значение переменной a.

Пример 6.17. В алгоритме, записанном ниже, используются целочисленные переменные a и b, а также следующие операции:

:= — присваивание;

+ — сложение;

– = вычитание;

* — умножение;

/ — целочисленное деление.

Определите значение переменной a после исполнения данного алгоритма:

```
a := 14
b := 4
b := b * a / 7
a := b + a * 10 - 49
```

Порядок действий соответствует правилам арифметики.

В ответе укажите одно число — значение переменной a.

Пример 6.18. В алгоритме, записанном ниже, используются целочисленные переменные a и b, а также следующие операции:

:= — присваивание;

+ — сложение;

– — вычитание;

* — умножение;

/ — целочисленное деление.

Определите значение переменной a после исполнения данного алгоритма:

```
a := 11
a := 112 - a * 9
b := 51 - 33 + a * 2
a := a - 2
a := a - b / a + 39
```

Порядок действий соответствует правилам арифметики.

В ответе укажите одно число — значение переменной a.

Для решения следующих заданий напомним русский алфавит:
А Б В Г Д Е Ё Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я.

Пример 6.19. Алгоритм из одной цепочки символов получает новую цепочку следующим образом. Сначала записывается буква, следующая в алфавите после первой буквы исходной цепочки, затем две последние буквы исходной цепочки в обратном порядке и, наконец, первая буква исходной цепочки. Получившаяся цепочка является результатом работы алгоритма. Например, если исходная цепочка символов была **СОН**, то результатом алгоритма будет цепочка **ТНОС**.

Дана цепочка символов **КОШ**. Какая цепочка символов получится, если к данной цепочке применить алгоритм трижды (т. е. алгоритм применяется к данной цепочке, затем к результату его работы, а затем ко второму результату работы алгоритма)?

Пример 6.20. Алгоритм из одной цепочки символов получает новую цепочку следующим образом. Сначала записываются первые две буквы исходной цепочки в обратном порядке, затем буква, следующая в алфавите за последней буквой исходной цепочки и, наконец, исходная цепочка символов, записанная в обратном порядке. Получившаяся цепочка является результатом работы алгоритма. Например, если исходная цепочка символов была **СОН**, то результатом алгоритма будет цепочка **ОСОНОС**.

Дана цепочка символов **НИЛ**. Какая цепочка символов получится, если к данной цепочке применить алгоритм дважды (т. е. алгоритм применяется к данной цепочке, а затем к результату его работы)?

Пример 6.21. Алгоритм из одной цепочки символов получает новую цепочку следующим образом. Сначала записывается буква, предшествующая в алфавите первой букве исходной цепочки, затем исходная цепочка символов, записанная в обратном порядке. Получившаяся цепочка является результатом работы алгоритма. Например, если исходная цепочка символов была **СОН**, то результатом алгоритма будет цепочка **РНОС**.

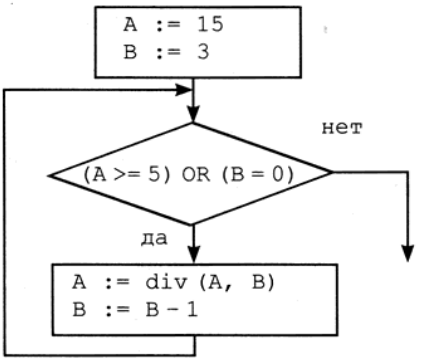
Дана цепочка символов **РОК**. Какая цепочка символов получится, если к данной цепочке применить алгоритм трижды (т. е. алгоритм применяется к данной цепочке, затем к результату его работы, а затем ко второму результату работы алгоритма)?

Пример 6.22. Алгоритм из одной цепочки символов получает новую цепочку следующим образом. Сначала записывается буква, предшествующая в алфавите последней букве исходной цепочки, затем первые две буквы исходной цепочки и, наконец, буква, следующая в алфавите за первой буквой исходной цепочки символов. Получившаяся цепочка является результатом работы алгоритма. Например,

если исходная цепочка символов была **СОН**, то результатом алгоритма будет цепочка **МСОТ**.

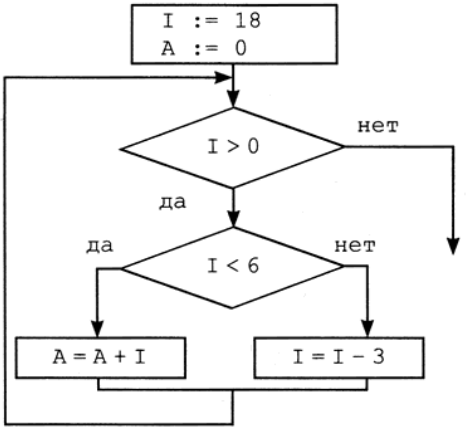
Дана цепочка **АРТ**. Какая цепочка символов получится, если к данной цепочке применить алгоритм дважды (т. е. алгоритм применяется к данной цепочке, а затем к результату его работы)?

Пример 6.23. Сколько раз выполнится тело цикла в следующем фрагменте алгоритма, представленном блок-схемой:



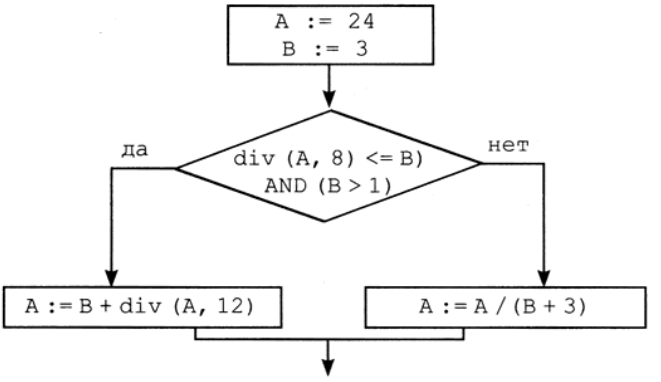
Знаком `:=` обозначена операция присваивания;
`OR` — операция логического ИЛИ;
`div` — операция целочисленного деления.
В ответе укажите одно число — количество тела цикла.

Пример 6.24. Определите значение переменной `A` после выполнения фрагмента алгоритма, представленного блок-схемой:



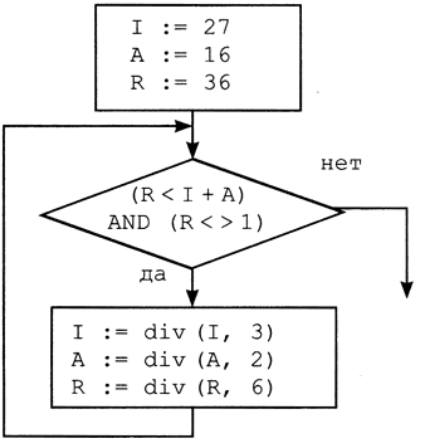
Знаком `:=` обозначена операция присваивания.
В ответе укажите одно число — значение переменной `A`.

Пример 6.25. Определите значение переменной A в результате выполнения фрагмента алгоритма, представленного блок-схемой:



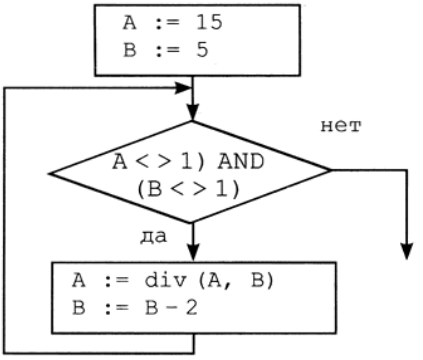
Знаком := обозначена операция присваивания;
AND — операция логического И;
div — операция целочисленного деления.
В ответе укажите одно число — значение переменной A.

Пример 6.26. Сколько раз выполнится тело цикла в следующем фрагменте алгоритма, представленном блок-схемой:



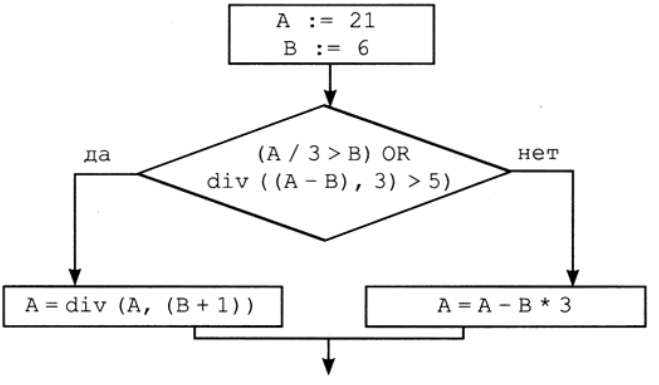
Знаком := обозначена операция присваивания;
< > — операция проверки на неравенство;
AND — операция логического И;
div — операция целочисленного деления.
В ответе укажите одно число — количество тела цикла.

Пример 6.27. Сколько раз выполнится тело цикла в следующем фрагменте алгоритма, представленном блок-схемой:



Знаком := обозначена операция присваивания;
< > — операция проверки на неравенство;
AND — операция логического И;
div — операция целочисленного деления;
В ответе укажите одно число — количество тела цикла.

Пример 6.28. Чему равно значение выражения A + B, если переменные A и B получили свои значения в результате выполнения фрагмента алгоритма, представленного блок-схемой:



Знаком := обозначена операция присваивания;
OR — операция логического ИЛИ;
div — операция целочисленного деления.
В ответе укажите одно число — значение A + B.

ГЛАВА 7
ОСНОВЫ ПРОГРАММИРОВАНИЯ

Алгоритм, записанный на языке, который понимает исполнитель, называется **программой**. Программы пишутся на языках программирования. **Язык программирования** — это фиксированная система обозначений для описания алгоритмов и структур данных. Далее в этом разделе под исполнителем будем понимать компьютер.

Первые программы для компьютеров составляли в машинных кодах. **Машинные коды** — совокупность машинных команд компьютера, отличающихся для разных компьютеров некоторыми параметрами, например набором операций. В программе, составленной в машинных кодах, программистом должны быть явно заданы не только определённые команды для выполнения каждой операции, но и адреса оперативной памяти, где должны храниться данные. Требуется также указать, как пересылать и обрабатывать данные, где хранить результаты вычислений. Программы в машинных кодах трудно отлаживать и модифицировать. Программы, составленные для одного компьютера, могут быть непригодны для другого, поэтому язык машинных кодов называют **машинно-зависимым**.

Появление *языка ассемблера* в 1950 году существенно упростило процесс написания программ. **Язык ассемблера** позволяет программировать на уровне машинных команд, но при этом в отличие от машинных кодов:

1) вместо кода команды записывается её мнемоническое обозначение с использованием букв латинского алфавита;
2) было введено понятие переменной, что позволило использовать вместо адресов ячеек памяти имя переменной.

Языки ассемблерного типа позволили повысить скорость программирования и уменьшить количество ошибок по сравнению с программированием в машинных кодах. Так как команды ассемблера однозначно соответствуют командам машинного языка, все языки ассемблерного типа являются машинно-зависимыми.

В 1954 году появился первый **язык программирования высокого уровня** — ФОРТРАН. В языках высокого уровня используются слова естественного языка (в основном американского английского). Впоследствии были разработаны языки высокого уровня Алгол (1958), Бейсик (1963, Томас Куртц и Джон Кемени), Паскаль (1970, Никлаус Вирт), Си (1972, Кен Томпсон и Денис Ритчи). Новые языки программирования, как универсальные, так и специализированные, появляются и в настоящее время.

Языки программирования высокого уровня — **машинно-независимые языки**. Программы на языках высокого уровня преобразуются в понятную компьютеру форму (в команды машинного языка) при помощи специальных программ — **трансляторов**. Существует два принципиально различных метода трансляции — **компиляция** и **интерпретация**. Программа-компилятор переводит целиком текст программы на языке высокого уровня на язык машинных команд, только после этого

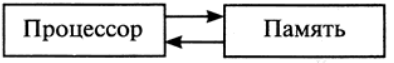
программа выполняется. Интерпретатор переводит на язык машинных команд очередной оператор программы, после чего он выполняется. Откомпилированная программа выполняется быстрее, чем интерпретируемая.

В школах изучают разные языки программирования. В заданиях ЕГЭ для записи программ используются школьный АЯ, языки Бейсик, Паскаль и Си, в заданиях ГИА — школьный АЯ.

Алгоритмы, составленные на школьном АЯ, могут быть выполнены на компьютере, поэтому для записи программ будем использовать школьный АЯ. Основные команды школьного АЯ описаны в предыдущей главе. В языках программирования высокого уровня часто вместо термина «команда» используется термин «оператор».

Переменные и константы в программах

Напомним, что **программы представляют собой запись алгоритмов на языках программирования**. Программы выполняются на реальном вычислительном устройстве — компьютере универсального назначения или специализированном компьютере, управляющем оборудованием. Каждый компьютер, как любой исполнитель, имеет ряд ограничений: по быстродействию, размеру памяти и т. п. При исполнении программ вычислительные действия выполняет процессор компьютера, а данные и команды программ хранятся в памяти:



Данные в программе представляются в виде *переменных* и *констант*. Переменные и константы имеют три основных атрибута: *имя, значение* и *тип*. **Переменная** в программе — это именованная область памяти, в которой хранится её значение. Значение переменной может изменяться в ходе выполнения программы. **Константа**, как и переменная, хранится в памяти и отличается тем, что её значение не может быть изменено в ходе выполнения программы. Переменные и константы, используемые в программах, занимают в памяти некоторое количество байт.

Переменная или константа обозначается *идентификатором* в соответствии с правилами языка программирования. В школьном АЯ именами (идентификаторами) переменных могут быть, например, `A`, `B`, `СЛОВО`, `степ`, `чис1` и проч.

Для записи в переменную некоторого значения используется оператор присваивания, описанный в главе 6 «Основы алгоритмизации». В операторе присваивания имя переменной, записанной слева, может встретиться и в правой части, например,

```
A := A + 7
```

В этом операторе переменной `A` присваивается результат вычисления выражения `A + 7`. Такая запись возможна именно потому, что переменные хранятся в памяти компьютера, а вычисления проводятся про-

цессором. Значение из ячейки памяти, в которой хранится переменная A, суммируется в процессоре со значением ячейки памяти, в которой хранится константа 7. Полученный результат записывается в ячейку памяти переменной A.

В процессоре есть специальные ячейки памяти, которые называются **регистрами**. Каждый регистр имеет закреплённое за ним имя, например: AX, BX, DX. Регистры используются для хранения значений, которые в данный момент обрабатывает процессор, для хранения адресов команд в памяти и т. д. На рис. 14 показаны один регистр процессора, назовем его R, и возможная схема выполнения оператора присваивания языка высокого уровня A := A + B.

1) Значение из ячейки памяти, в которой хранится переменная A, передаётся в регистр R процессора.
2) Затем из памяти в процессор передаётся значение переменной B и суммируется со значением в регистре R.
3) Полученный результат записывается в ячейку памяти переменной A.

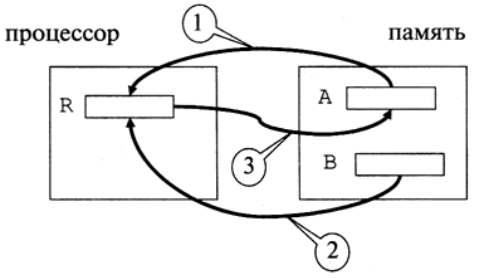


Рис. 14. Схема выполнения оператора присваивания.

Реализация оператора присваивания, записанного на языке высокого уровня A := A + B, на языке ассемблера может иметь вид:

№	Команда	Описание
1	MOV R, A	В регистр R процессора заносится значение переменной A из памяти
2	ADD R, B	Содержимое регистра R суммируется со значением переменной B, находящимся в памяти
3	MOV A, R	В переменную A, находящуюся в памяти, переносится значение из регистра R процессора

Приведённые команды ассемблера состоят из имени команды (MOV, ADD) и двух имён операндов, разделённых запятыми. По правилам языка команда MOV заносит в первый операнд значение второго операнда, команда ADD суммирует значение двух операндов, результат записывается в первый операнд — регистр R.

Для составления программ на языке высокого уровня программисту не требуется знать систему команд и архитектуру процессора (количество регистров, их значение и т. п.). Установление соответствия между программой и системой команд с учётом архитектуры процессора выполняет транслятор.

Типы данных

Из школьного курса математики известны множества натуральных, целых, рациональных и действительных чисел. В компьютерах эти числа записываются в ячейки памяти, состоящие из конечного числа битов, поэтому могут быть записаны не все возможные значения, а только их часть. В зависимости от способа хранения и команд, при помощи которых компьютер обрабатывает двоичные значения ячеек памяти, различают *типы данных*.

Для того чтобы транслятор мог перевести программу в последовательность машинных команд, нужно указать в программе типы данных. В языках программирования для этого используются операторы описания типов переменных.

Тип данных в языках программирования определяет:

- множество возможных значений;
- совокупность операций, допустимых над этими значениями;
- способ хранения константы или переменной в памяти компьютера (формат хранения).

Переменные целочисленных типов можно складывать, умножать, сравнивать и т. д. Переменные литерного (строкового) типа невозможно умножать и делить, их можно сцеплять и сравнивать между собой. Таким образом, для разных типов данных допустимы разные наборы операций.

Переменная целого типа, занимающая в памяти один байт и принимающая только положительные значения (беззнаковый целый тип), может иметь значения из интервала [0; 255] ($255 = 2^8 - 1 - 11111111_2$). Если же тип целочисленной переменной знаковый, то возможные значения лежат в интервале [−128; 127]. Количество различных значений в обоих случаях $256 = 2^8$, так как для хранения переменной выделяется 8 бит.

При разработке алгоритма нужно помнить об ограничениях выполняющего его исполнителя. Если не задумываться об этом, можно получить неожиданные результаты.

Предположим, целочисленные беззнаковые переменные `A` и `B` занимают в памяти по одному байту. Выполним операцию сложения `A + B` при разных значениях `A` и `B`.

`A`	56	$0011\ 1000_2$
`B`	15	$0000\ 1111_2$
`A+B`	71	$0100\ 0111_2$

`A`	250	$1111\ 1010_2$
`B`	15	$0000\ 1111_2$
`A+B`	265	$1\ 0000\ 1001_2$

В первом примере будет получен верный результат 56 + 15 = 71. Во втором примере единица в старшем бите результата сложения будет от-

брошена — ей не хватает места в выделенном байте памяти, получим результат 9, а не 265. В таких случаях говорят о **переполнении разрядной сетки**: результат становится больше максимально возможного значения для переменной.

Программист должен учитывать диапазон возможных значений переменных, точность их представления и другие особенности при программной реализации алгоритма. Для этого надо знать, какие типы данных используются в языке программирования.

В школьном АЯ используются следующие простые типы:

- `цел` — целый;
- `вещ` — вещественный;
- `сим` — символьный;
- `лит` — литерный;
- `лог` — логический.

Значением символьной (`сим`) переменной или константы является один символ. Обычно для хранения символа в памяти выделяется один байт, поэтому количество допустимых символов равно 256, символы пронумерованы от 0 до 255. В алгоритме и программе символьные константы записываются в апострофах, например: 'a', '+', '9'.

Значением литерной (`лит`) переменной или константы является последовательность символов. Иногда её называют строкой или цепочкой символов. Количество символов в строке называется длиной строки.

Для указания типа переменной её необходимо `объявить`, или `описать`, в программе. В школьном АЯ объявление переменных производится в начале программы. Несколько переменных одного типа могут быть объявлены после имени типа через запятую. Например,

```
цел A, B
лог BOOL
```

Параметры программ

Программы на школьном АЯ могут иметь **параметры**. Параметры делятся на **входные** — *аргументы* и **выходные** — *результаты*.

В школьном АЯ параметры перечисляются в скобках после имени алгоритма (программы), при этом входные параметры помечаются словом `арг`, а выходные — словом `рез`.

Отметим, что не во всех языках высокого уровня программы имеют параметры.

Часть программы от слова `алг` до слова `нач` называется заголовком программы, а часть, заключённая между словами `нач` и `кон`,— телом программы.

Пример: алгоритм (программа) подсчёта среднего арифметического (выходной параметр) двух целых чисел (входные параметры):

```
алг СУММА (арг цел A, арг цел B, рез вещ C)
нач
    ввод A, B
    C := (A+B)/2
    вывод C
кон
```

Типы входных и выходных параметров указываются перед именем параметра.

Независимо от того, какой язык программирования используется, программу можно составить из операторов:

- присваивания;
- ввода/вывода;
- ветвления;
- цикла;
- обращения к вспомогательному алгоритму.

Программы, использующие ветвление

Использование в программах ветвления позволяет задавать различные линии поведения программы в зависимости от выполнения некоторых условий. Для организации ветвлений в программах используется **условный оператор** и **оператор выбора**. Условия задаются в форме логических выражений.

Для записи логических выражений в программах будем использовать:

1) логические функции И, ИЛИ и НЕ, обозначаемые в школьном АЯ `и`, `или`, `не` соответственно;
2) операции сравнения:
   - `=` равно;
   - `>` больше;
   - `<` меньше;
   - `>=` больше или равно;
   - `<=` меньше или равно;
   - `<>` не равно.

В качестве примера рассмотрим программу поиска максимального из трёх заданных пользователем чисел:

```
алг МАКСИМУМ (арг вещ A, арг вещ B, арг вещ C, рез
вещ МАКС)
нач
   ввод A, B, C
       если B > A то
       если B>C то МАКС := B
           иначе МАКС := C
       все
   иначе
       если A > C то МАКС := A
           иначе МАКС := C
       все
   все
   вывод МАКС
кон
```

Обратите внимание на то, что в приведённой программе для поиска максимального из трёх чисел используются два вложенных условных оператора.

В этом варианте решения использованы три операции сравнения и четыре оператора присваивания (хотя не все они выполняются). Обратите внимание на наличие в программе вложенных условных операторов: первый вложенный оператор выполняется по ветви `то`, второй — по ветви `иначе` внешнего условного оператора. Вложенные условные операторы должны полностью завершиться в той ветви, где они используются.

Для повышения быстродействия программы обычно минимизируют количество операторов и операций, использованных в ней. Попытаемся улучшить программу:

```
алг МАКСИМУМ (арг вещ A, арг вещ B, арг вещ C, рез вещ МАКС)
нач
   ввод A, B, C
   если B > A то МАКС := B иначе МАКС := A все
   если C > МАКС то МАКС := C все
   вывод МАКС
кон
```

Теперь в программе два сравнения и три оператора присваивания.

Построим трассировочную таблицу первого вида для следующих входных данных: `A = 15, B = 3, C = 9`.

№ шага	Оператор или условие (логическое выражение)	Вычисление выражения	A	B	C	МАКС
1	**ввод** A, B, C		15	3	9	
2	B > A	(3 > 15) = нет				
3	МАКС := A	15				15
4	C > МАКС	(9 > 15) = нет				
5	**вывод** МАКС					**15**

Рассмотрим ещё один пример на разработку программы, использующей условные операторы. Функция

$$y = f(x) = a \cdot x^2 + b \cdot x + c$$

задаётся коэффициентами a, b и c. Если заданная функция описывает параболу, необходимо определить направление её ветвей и координаты вершины. В противном случае вывести сообщение «Это не парабола. Вершина не определена».

```
алг СВОЙСТВА_ПАРАБОЛЫ (арг вещ A, арг вещ B, арг вещ C)
нач
    вещ X, Y
    ввод A, B, C
    если A = 0 то
        вывод "Это не парабола. Вершина не определена"
      иначе
        если A > 0 то
              вывод "Ветви параболы направлены вверх"
            иначе
              вывод "Ветви параболы направлены вниз"
        все
        X = -B / (2 * A)
        Y = A * X * X + B * X + C
        вывод "Координаты вершины: ", X, Y
    все
кон
```

Программы, использующие циклы

Многие программы составляются с использованием циклов. Например, при составлении программы для возведения целого числа A в натуральную степень K без использования циклических конструкций для каждого значения K потребуется написать отдельную программу с фиксированным количеством операторов.

Для того чтобы избавиться от подобного рутинного труда, в программах применяют **операторы цикла**.

Напишем программу, возводящую произвольное целое число A в целую неотрицательную степень K : $S = A^K$. Воспользуемся формулой $A^K = A^{K-1} \cdot A$.

Решение подобных задач разбивается на два этапа:

1) на нулевом шаге выполняется инициализация переменных ($S = 1$, так как $A^0 = 1$);
2) в цикле K раз повторяется вычисление по формуле $S = S \cdot A$.

Для составления программы будем использовать входные переменные A и K, выходную переменную S и переменную организации цикла СЧЕТЧИК:

```
алг СТЕПЕНЬ (арг цел A, арг цел K, рез цел S)
нач
    ввод A, K
    цел S := 1
    цел СЧЕТЧИК := 0
    нц пока СЧЕТЧИК < K
        S := S * A
        СЧЕТЧИК := СЧЕТЧИК + 1
    кц
    вывод S
кон
```

Проверим, верно ли работает программа. Построим трассировочную таблицу первого вида для входных данных A=2 и K=4. В результате выполнения программы число 2 должно быть возведено в степень 4.

№ шага	Оператор или условие (логическое выражение)	Вычисление выражения	A	K	S	СЧЕТЧИК	вывод
1	ввод A, K		2	4			
2	S := 1	1			1		
3	СЧЕТЧИК := 0	0				0	
4	СЧЕТЧИК < K	(0 < 4) = да					
5	S := S * A	1 * 2 = 2			2		
6	СЧЕТЧИК := := СЧЕТЧИК + 1	0 + 1 = 1				1	
7	СЧЕТЧИК < K	(1 < 4) = да					
8	S := S * A	2 * 2 = 4			4		
9	СЧЕТЧИК := := СЧЕТЧИК + 1	1 + 1 = 2				2	
10	СЧЕТЧИК < K	(2 < 4) = да					
11	S := S * A	4 * 2 = 8			8		
12	СЧЕТЧИК := := СЧЕТЧИК + 1	2 + 1 = 3				3	
13	СЧЕТЧИК < K	(3 < 4) = да					
14	S := S * A	8 * 2 = 16			16		
15	СЧЕТЧИК := := СЧЕТЧИК + 1	3 + 1 = 4				4	
16	СЧЕТЧИК < K	(4 < 4) = нет					
17	вывод S						**16**

Трассировочная таблица второго типа (контрольная точка на строке пока СЧЕТЧИК < K) имеет вид:

№ кт	СЧЕТЧИК < K	A	K	S	СЧЕТЧИК	вывод
1	да	2	4	1	0	
2	да			2	1	
3	да			4	2	
4	да			8	3	
5	нет			16	4	
вывод S						**16**

Запишем программу с использованием оператора цикла с параметром:

```
алг СТЕПЕНЬ (арг цел A, арг цел K, рез цел S)
нач
   ввод A, K
   цел S := 1
   цел СЧЕТЧИК
   нц для СЧЕТЧИК от 1 до K
       S := S * A
   кц
   вывод S
кон
```

В программировании параметры цикла традиционно называют латинскими буквами из середины алфавита: `I, J, K` и т. д.

Трассировочная таблица второго типа для программы при контрольной точке на заголовке цикла:

№ кт	A	K	СЧЕТЧИК	S	вывод S
1	2	4	1	1	
2			2	2	
3			3	4	
4			4	8	
5				16	**16**

Сравнение программ показывает, что использование оператора цикла с параметром позволяет записать программу более компактно. Оператор цикла с параметром используется, когда количество повторений тела цикла заранее известно.

В следующем примере приведена программа, вычисляющая *факториал целого неотрицательного числа K*. **Факториал** — это произведение натурального ряда K чисел. В математике обозначается $K!$ и вычисляется по формуле:

$$K! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (K-1) \cdot K,$$
$$0! = 1$$

Формулу вычисления факториала можно записать в виде

$$K! = (K-1)! \cdot \mathrm{K},$$
$$0! = 1$$

что даёт возможность составить программу вычисления факториала с использованием оператора цикла. Если обозначить переменную, в которой последовательно будут вычисляться значения факториалов, именем `ФАКТ`, то надо K раз повторить вычисления по формуле `ФАКТ = ФАКТ * СЧЕТ`, где значение счётчика цикла `СЧЕТ` будет изменяться от 1 до K. До цикла переменная `ФАКТ` должна быть инициирована значением 1 ($0! = 1$).

Программа с использованием оператора цикла с предусловием:

```
алг ФАКТОРИАЛ (арг цел К, рез цел ФАКТ)
нач
   цел ФАКТ := 1, СЧЕТ := 1
   ввод К
   нц пока СЧЕТ<= К
           ФАКТ := ФАКТ * СЧЕТ
      СЧЕТ := СЧЕТ + 1
   кц
   вывод ФАКТ
кон
```

При сравнении программ вычисления степени числа и факториала можно увидеть, что они отличаются одним оператором в теле цикла: в первой программе переменная S умножается на значение заданного числа A, во второй — переменная ФАКТ умножается на значение счетчика цикла СЧЕТ. В обеих программах тело цикла выполняется К раз.

Для проверки работоспособности программы построим трассировочную таблицу первого вида для вычисления факториала числа 4 (в программу вводится значение входного параметра К = 4).

№ шага	Оператор или условие (логическое выражение)	Вычисление выражения	К	ФАКТ	СЧЕТ	вывод
1	ФАКТ := 1	1		1		
2	СЧЕТ := 1	1			1	
3	ввод К	4	4			
4	СЧЕТ <= К	(1 <= 4) = да				
5	ФАКТ := ФАКТ * СЧЕТ	1 * 1 = 1		1		
6	СЧЕТ := СЧЕТ + 1	1 + 1 = 2			2	
7	СЧЕТ <= К	(2 <= 4) = да				
8	ФАКТ := ФАКТ * СЧЕТ	1 * 2 = 2		2		
9	СЧЕТ := СЧЕТ + 1	2 + 1 = 3			3	
10	СЧЕТ <= К	(3 <= 4) = да				
11	ФАКТ := ФАКТ * СЧЕТ	2 * 3 = 6		6		
12	СЧЕТ := СЧЕТ + 1	3 + 1 = 4			4	
13	СЧЕТ <= К	(4 <= 4) = да				
14	ФАКТ := ФАКТ * СЧЕТ	6 * 4 = 24		24		
15	СЧЕТ := СЧЕТ + 1	4 + 1 = 5				
16	СЧЕТ <= К	(5 <= 4) = нет				
17	вывод ФАКТ					**24**

При К = 4 программа работает верно.

Будем считать, что целочисленный тип данных, который используется в этой программе, является беззнаковым и занимает в памяти 2 байт. Максимальное значение, которое может принимать переменная

такого типа, равно $2^{16} - 1 = 65\,535$. Построим трассировочную таблицу второго типа для вычисления значения 9!. Контрольную точку установим на строке

```
пока СЧЕТ <= К
```

и будем записывать результат в десятичной и в двоичной форме:

№ кт	СЧЕТ <= К	СЧЕТ	К	ФАКТ	ФАКТ	ВЫВОД
1	да	1	9	1	0000 0000 0000 0001	
2	да	2		2	0000 0000 0000 0010	
3	да	3		6	0000 0000 0000 0110	
4	да	4		24	0000 0000 0001 1000	
5	да	5		120	0000 0000 0111 1000	
6	да	6		720	0000 0010 1101 0000	
7	да	7		5040	0001 0011 1011 0000	
8	да	8		40320	1001 1101 1000 0000	
9	да	9		35200	0101 1000 1001 1000 0000	
10	нет	10				
Результат						**35200**

Вместо ожидаемого результата 362880 программа выведет 35200. Причина — переполнение разрядной сетки, три старших бита 101 отбрасываются, так как им не хватило места в памяти.

Для сравнения приведём программу вычисления факториала с использованием оператора цикла с параметром:

```
алг ФАКТОРИАЛ (арг цел К, рез цел ФАКТ)
нач
   цел ФАКТ := 1, СЧЕТ
   ввод К
   нц для СЧЕТ от 1 до К
         ФАКТ := ФАКТ * СЧЕТ
   кц
   вывод ФАКТ
кон
```

Пример 7.1. Задание с выбором одного ответа

Значение переменной S после выполнения фрагмента программы

```
цел X := 0; S := 5
нц пока X < 2
  S := S + X
  X := X + 1
кц
```

будет равно

1) 4
2) 6
3) 8
4) 7

Решение. Составим трассировочную таблицу второго типа с контрольной точкой на строке `пока X < 2`.

№ шага	x < 2	x	s
1	да	0	5
2	да	1	5
3	нет	2	**6**

Ответ: 2.

Программы, использующие вложенные циклы

Тело цикла может состоять не только из линейных конструкций, условных операторов, но и из циклических операторов. В последнем случае говорят о **вложенных циклах**. Внутренний, или вложенный, цикл должен полностью содержаться во внешнем, т. е. внутренний цикл должен начаться и завершиться внутри внешнего. На рис. 15а—в показаны допустимые вложенные циклические конструкции, на рис. 15г — недопустимые.

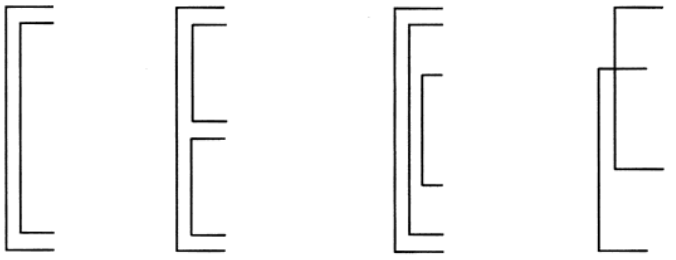


Рис. 15.

Рассмотрим программу с вложенными циклами. Внешний цикл выполняется до тех пор, пока значение переменной `I` строго меньше трёх. Внутренний цикл полностью выполняется на каждом теле внешнего цикла. Он повторяется до тех пор, пока значение переменной `J` строго меньше трёх. Для внутреннего цикла значение `I` неизменно.

Оператор цикла с предусловием	Оператор цикла с параметром
алг ВЛОЖ_ЦИКЛЫ_1 (рез цел J) нач цел I := 0, J нц пока I < 3 J := 0 нц пока J < 3 вывод "I = ", I, "J = ", J ПЕРЕВОД_СТРОКИ J := J + 1 кц I := I + 1 кц кон	алг ВЛОЖ_ЦИКЛЫ_1 (рез цел J) нач цел I, J нц для I от 0 до 2 нц для J от 0 до 2 вывод "I = ", I, "J = ", J ПЕРЕВОД_СТРОКИ кц кц кон

В результате выполнения программы на экран будет выведено:

```
I = 0    J = 0
I = 0    J = 1
I = 0    J = 2
I = 1    J = 0
I = 1    J = 1
I = 1    J = 2
I = 2    J = 0
I = 2    J = 1
I = 2    J = 2
```

Количество повторений внутреннего цикла может зависеть от значения счётчика внешнего цикла. В следующем примере программы внутренний цикл выполняется до тех пор, пока значение переменной J строго меньше текущего значения переменной I:

```
алг ВЛОЖ_ЦИКЛЫ_2 (рез цел J)
нач
   цел I := 0, J
   нц пока I < 3
      J := 0
      нц пока J < I
         вывод J
         J := J + 1
      кц
      I := I + 1
   кц
кон
```

Построим трассировочную таблицу первого вида для данной программы.

№ шага	Оператор или условие (логическое выражение)	Вычисление оператора присваивания или условия	I	J	вывод
1	I := 0	0	0		
2	I < 3	(0 < 3) = да			
3	J := 0	0		0	
4	J <= I	(0 < 0) = нет			
5	I := I + 1	0 + 1 = 1	1		
6	I < 3	(1 < 3) = да			
7	J := 0	0		0	
8	J <= I	(0 < 1) = да			
9	вывод J				0
10	J := J + 1	0 + 1 = 1		1	
11	J <= I	(1 < 1) = нет			
12	I := I + 1	1 + 1 = 2	2		
13	I < 3	(2 < 3) = да			
14	J := 0	0		0	
15	J <= I	(0 < 2) = да			
16	вывод J				0
17	J := J + 1	0 + 1 = 1		1	
18	J <= I	(1 < 2) = да			
19	вывод J				1
20	J := J + 1	1 + 1 = 2		2	
21	J <= I	(2 < 2) = нет			
22	I := I + 1	2 + 1 = 3	3		
23	I < 3	(3 < 3) = нет			

Построим трассировочную таблицу второго типа для этой программы, контрольная точка на строке программы `пока J < I`.

№ шага	Внешний цикл		Внутренний цикл		
	I	I < 3	J	J < I	вывод
1	0	да	0	нет	
2	1	да	0	да	
3			1	нет	0
4	2	да	0	да	
5			1	да	0
6			2	нет	1

В результате выполнения программы на экран будет выведено **001**.

Пример 7.2. Задание с выбором одного ответа

Значение переменной P, вычисленной в соответствии с фрагментом программы

```
цел P := 1, I = 1
нц пока I <= 3
  J := 1
  нц пока J <= 3
      P := P + I + J
      J := J + 1
  кц
  I := I + 1
кц
```

равно:

1) 18
2) 36
3) 19
4) 37

Решение. При решении задач такого типа необходимо обращать внимание на присвоение начальных значений переменным, управляющих циклом, и на выражения, по которым они изменяются в цикле. Выполним фрагмент алгоритма по шагам, не строя трассировочной таблицы:

```
I = 1
    J = 1
        P = 1 + 1 + 1 = 3
    J = 2
        P = 3 + 1 + 2 = 6
    J = 3
        P = 6 + 1 + 3 = 10
I = 2
    J = 1
        P = 10 + 2 + 1 = 13
    J = 2
        P = 13 + 2 + 2 = 17
    J = 3
        P = 17 + 2 + 3 = 22
I = 3
    J = 1
        P = 22 + 3 + 1 = 26
    J = 2
        P = 26 + 3 + 2 = 31
    J = 3
        P = 31 + 3 + 3 = 37
```

Ответ: 4.

Вспомогательные алгоритмы и подпрограммы

В повседневной жизни мы используем множество глаголов и выражений, позволяющих нам не описывать последовательности действий по шагам. Например, «накрыть на стол» может означать: постелить скатерть на стол, расставить тарелки, разложить приборы и салфетки. Однако, сказав «накрой на стол», мы не перечисляем явно все эти действия, они подразумеваются. Если считать последовательность действий при накрывании стола алгоритмом, то можно сказать, что словосочетание «накрыть на стол» — это «синоним», определяющий эту последовательность. В алгоритмизации такие «синонимы» называют **вспомогательными алгоритмами**, в программировании вспомогательные алгоритмы реализуются **подпрограммами**.

Ценность вспомогательных алгоритмов в том, что они представляют собой самостоятельные алгоритмы с входами и выходами, к которым можно обращаться, не указывая описанную в них последовательность действий. Выделение вспомогательных алгоритмов позволяет сократить размер основного алгоритма и сделать его более понятным.

В качестве примера рассмотрим упрощённый алгоритм ГОСТИ, описывающий некоторые действия хозяина, готовящегося к приёму гостей (см. рис. 16).

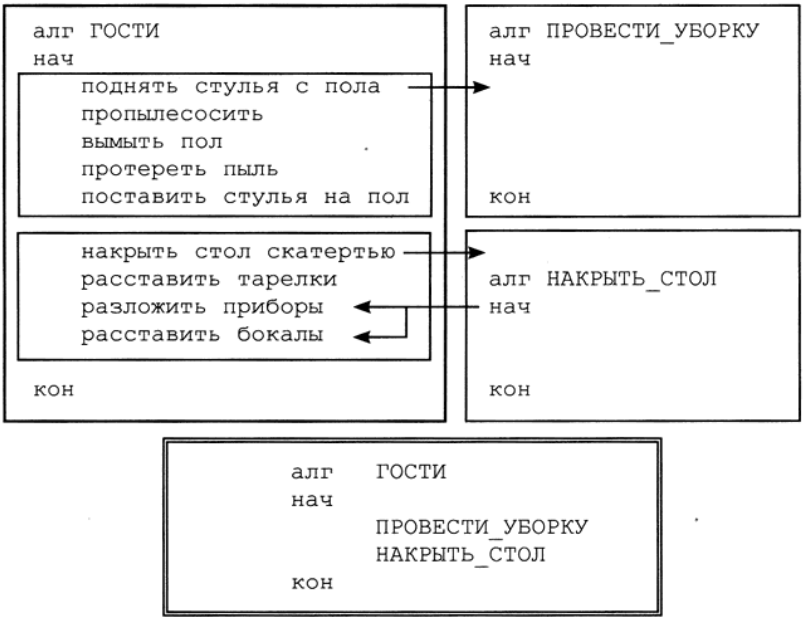


Рис. 16. Вспомогательный алгоритм.

На рисунке 16 слева приведён алгоритм ГОСТИ, содержащий все мелкие детали уборки дома и сервировки стола. Справа показаны два вспомогательных алгоритма, где выделены действия по уборке и сервировке соответственно. Ниже показан алгоритм ГОСТИ, использующий выделенные вспомогательные алгоритмы.

Рассмотрим ещё пример. Требуется написать программу, вычисляющую сумму степеней двойки, значение показателя степени очередного слагаемого вводится пользователем с клавиатуры. Ввод прекращается, если пользователем введено отрицательное число. Результат вывести на экран.

Первый вариант программы (без использования подпрограмм):

```
алг СУММ_СТЕП (арг цел А, рез цел РЕЗУЛЬТАТ)
нач
    цел РЕЗУЛЬТАТ, СТЕП
    РЕЗУЛЬТАТ := 0
    СТЕП := 1
    ввод А
    нц пока А > 0
        I := 0
        нц пока I < А
          СТЕП := СТЕП * 2
          I := I + 1
        кц
          РЕЗУЛЬТАТ := РЕЗУЛЬТАТ + СТЕП
          ввод А
        кц
        вывод РЕЗУЛЬТАТ
кон
```

Вложенный цикл выполняет отдельную задачу — вычисляет заданную степень двойки. Он может быть вынесен в отдельную подпрограмму, которая будет вызываться в основном цикле.

Напишем второй вариант программы (с использованием подпрограмм):

```
алг СТЕП_ДВОЙКИ (арг цел ПОКАЗАТЕЛЬ)
нач
    цел I
    I := 0
    нц пока I < ПОКАЗАТЕЛЬ
        РЕЗУЛЬТАТ := РЕЗУЛЬТАТ * 2
        I := I + 1
    кц
    вернуть РЕЗУЛЬТАТ
кон
```

```
алг СУММ_СТЕП (арг цел А, рез цел РЕЗУЛЬТАТ)
нач
    цел РЕЗУЛЬТАТ
    РЕЗУЛЬТАТ := 0
    ввод А
    нц пока А > 0
        РЕЗУЛЬТАТ := РЕЗУЛЬТАТ + СТЕП_ДВОЙКИ(А)
        ввод А
    кц
    вывод РЕЗУЛЬТАТ
кон
```

✍ Напишите программу с использованием подпрограмм самостоятельно *.

Напишите программу, вычисляющую произведение степеней целых неотрицательных чисел. Числа и степени вводятся пользователем с клавиатуры до тех пор, пока не вводится число ноль. Результат вывести на экран. Используйте оператор цикла с параметром.

Составление программ в среде формального исполнителя

Задачи государственной итоговой аттестации включают в себя вопросы, требующие умения не только исполнять, но и составлять программы на языке произвольного формального исполнителя.

При **составлении** программ на языке формального исполнителя важно помнить, что формальный исполнитель «понимает» только те команды, которые входят в систему его команд. Программист, заставляя устройство работать, должен мыслить элементарными командами и наборами команд, которые «понимает» исполнитель.

На Государственной итоговой аттестации могут быть предложены различные виды задач, например:

- задана система команд формального исполнителя, требуется составить алгоритм, приводящий к конкретному результату;
- задана система команд формального исполнителя, приведён некоторый алгоритм, требуется составить алгоритм с меньшим числом шагов, приводящий к такому же результату и др.

Исполнитель Вычислитель

Пример 7.3. Задание с кратким ответом

Исполнитель Вычислитель имеет следующую систему пронумерованных команд:

1. умножь на два;
2. прибавь единицу.

Первая умножает число на два, а вторая прибавляет к числу единицу. Алгоритм, преобразующий число 3 в число 26, записывается в виде последовательности команд 1121, что соответствует:

1. умножь на два	$3 \cdot 2 - 6$
1. умножь на два	$6 \cdot 2 - 12$
2. прибавь единицу	$12 + 1 - 13$
1. умножь на два	$13 \cdot 2 - 26$

Запишите порядок команд алгоритма, преобразующего число 3 в число 21, содержащего не более пяти команд, указывая лишь номера команд.

Решение. Задача преобразования числа 3 в число 21 имеет не одно решение. Например, можно на каждом шаге добавлять по единице, тогда надо будет выполнить 18 команд прибавь единицу. Можно семь раз прибавить единицу, полученное число 10 умножить на два и прибавить единицу. Существуют и другие решения.

Для наглядного представления вариантов решений подобной задачи удобно использовать дерево.

Дерево — это структура, состоящая из узлов и ветвей. Каждый узел может иметь потомков, с которыми он связан ветвями. По отношению к потомкам такой узел называется предком. Узел, не имеющий предка, называется **корнем** или **вершиной дерева**. Узлы, не имеющие потомков, называют **листьями**. Ветви дерева могут соответствовать выполнению какой-либо операции, приводящей к результату, который сохраняется в узле-потомке.

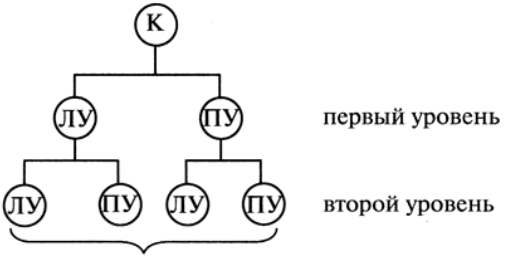


Рис. 17. Пример двоичного дерева.

На рис. 17 представлено двоичное дерево. Каждый узел может иметь двух потомков, на рисунке они обозначены ЛУ — левый узел и ПУ — правый узел. Вершина дерева обозначена символом К. Прямые потомки корня дерева образуют первый уровень, в свою очередь их потомки — второй уровень и т. д. Если N — это номер уровня, то количество узлов на этом уровне двоичного дерева равно 2^N. Например, на пятом уровне двоичного дерева может находиться 32 узла.

Для поиска вариантов решений будем строить двоичное дерево, потому что система команд Вычислителя содержит только две команды.

Левые ветви будут соответствовать выполнению команды 1. умножь на два, правые — выполнению команды 2. прибавь единицу. Пометим некоторые ветви для наглядности. В узлах дерева будем записывать полученный после выполнения соответствующей команды результат. В корень дерева запишем заданное число 3.

По условию задачи можно использовать не более пяти команд, поэтому дерево должно иметь не более пяти уровней. Дерево будем строить по уровням — от каждого узла верхнего уровня порождаются два узла следующего уровня.

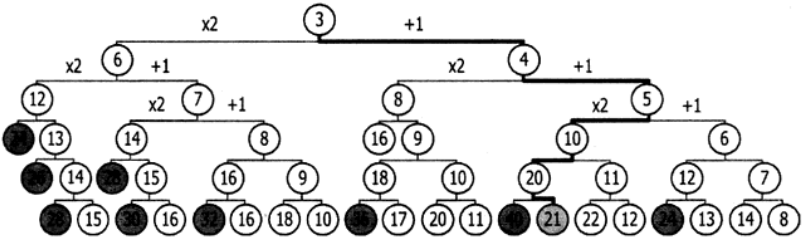


Рис. 18. Фрагмент двоичного дерева поиска вариантов решений.

В некоторых узлах значение превышает заданное число 21, построение дерева от этих узлов прекращается. Результатом является путь от вершины дерева к узлу с заданным числом, состоящий из последовательности команд формального исполнителя. На рисунке этот путь выделен жирной линией. Последовательность ветвей решения — правая, правая, левая, левая, правая, что соответствует командам 22112.

Описанный метод решения приводит к верному результату, но является весьма трудоёмким.

Подобные задачи принято решать «от ответа». Решим обратную задачу: получить из числа 21 число 3. Команды, которые мы будем применять, также должны быть обратными командам исполнителя, заданным выше в условии задачи:

1) `дели на два;`
2) `вычти единицу.`

Операция `дели на два` может выполняться только для чисел, кратных двойке, иначе эта команда не будет обратной исходной команде `умножь на два`.

Для скорейшего получения из числа 21 числа 3 следует по возможности применять операцию деления. Если это невозможно — применять операцию вычитания.

Результат решения обратной задачи — последовательность команд — в ответе на исходную задачу надо записать в обратном порядке.

«Обратная» команда	Выполнение команды
`2. вычти единицу`	21 – 1 = 20
`1. дели на два`	20 : 2 = 10
`1. дели на два`	10 : 2 = 5
`2. вычти единицу`	5 – 1 = 4
`2. вычти единицу`	4 – 1 = 3

Из таблицы видно, что алгоритм получения числа 3 из числа 21 — это последовательность команд 21122. Алгоритм преобразования числа 3 в число 21 — 22112.

Покажем, как можно использовать построение дерева для поиска решения обратной задачи. Дерево строим по уровням, как и в прямой задаче, до пятого уровня. В корень дерева запишем число 21. Построение дерева прекратим, как только на каком-либо уровне получим узел со значением 3. Отметим, что у некоторых узлов будет только один потомок, так как не ко всем числам можно применить операцию `дели на два`. Поэтому правой ветви соответствует операция `дели на два`, левой ветви и ветви, направленной вниз,— операция `вычти единицу`. Решение выделено жирной линией. В ответе номера операций следует записать от листа к корню.

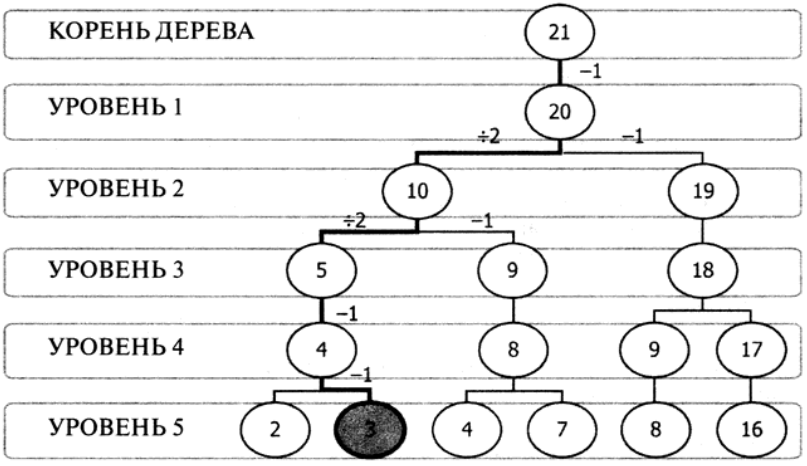


Рис. 19. Фрагмент дерева поиска вариантов решений обратной задачи.

Из рис. 19 видно, что решение обратной задачи менее трудоёмко. Это объясняется тем, что операцию целочисленного деления не всегда можно выполнить и количество ветвей дерева резко сокращается.

Из построенного дерева видно, что имеется два узла, содержащих значение 4. Если потребуется из числа 4 получить число 21 не менее чем за пять команд, то существуют два решения. Если же потребуется получить число 21 из числа 6 не более чем за пять команд, решения не существует.

Ответ: 22112.

Исполнитель Робот

В вариантах ГИА по информатике есть задания, где необходимо привести развёрнутый ответ. Примеры таких заданий и рассмотрим ниже.

Необходимо составить программу, применив систему команд некоторого формального исполнителя. Для удобства чтения общую для всех задач часть формулировки не будем повторять многократно.

Общая часть формулировки задания

Исполнитель Робот перемещается по плоскости, разбитой на клетки. Между соседними клетками плоскости может стоять стена, через которую Робот ходить не может. При попытке пройти сквозь стену Робот разрушается.

Система команд исполнителя Робот:

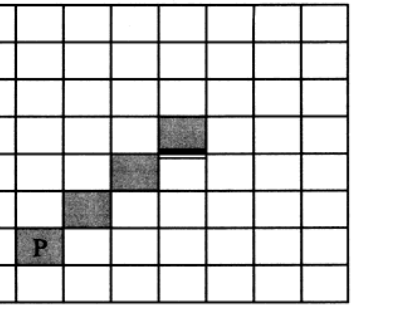
Команда	Описание команды
закрасить	закрасить клетку, в которой стоит Робот
вверх	переместить Робота на клетку вверх
вниз	переместить Робота на клетку вниз
влево	переместить Робота на клетку влево
вправо	переместить Робота на клетку вправо
сверху_свободно	проверка отсутствия стены сверху: стены нет — истина, иначе — ложь
снизу_свободно	проверка отсутствия стены снизу: стены нет — истина, иначе — ложь
слева_свободно	проверка отсутствия стены слева: стены нет — истина, иначе — ложь
справа_свободно	проверка отсутствия стены справа: стены нет — истина, иначе — ложь
если <условие> то <последовательность_команд> все	оператор ветвления: если <условие> верно, то выполнить <последовательность_команд>
нц пока <условие> <последовательность_команд> кц	оператор цикла: пока <условие> верно, выполнять <последовательность_команд>

Команды в условиях можно объединять при помощи логических функций И, ИЛИ и НЕ.

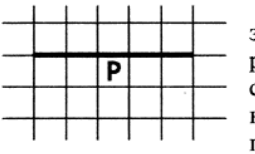
Начальное положение Робота отмечается в клетке на плоскости буквой Р.

Например, следующий алгоритм заставляет Робота закрашивать клетки по диагонали снизу слева вверх вправо до тех пор, пока сверху не появится стена.

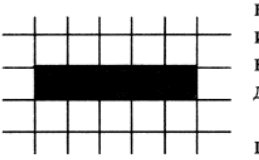
```
нц пока сверху_свободно
        закрасить
        вверх
        вправо
кц
```



Пример 7.4. Задание с развёрнутым ответом



а)



б)

На бесконечном поле клеток имеется горизонтальная стена. Длина стены неизвестна. Робот размещён в одной из клеток непосредственно под стеной. Точная начальная позиция Робота неизвестна. Возможное начальное положение Робота приведено на рис. а).

Напишите для Робота алгоритм, закрашивающий все клетки, расположенные под стеной и прилегающие к ней. Например, для приведённого выше рис. а) Робот должен закрасить следующие клетки (см. рис. б).

Конечное положение Робота может быть произвольным. Алгоритм должен решать задачу для произвольного размера стены и любой допустимой начальной позиции Робота.

Запишите алгоритм в текстовом редакторе и сохраните его на рабочем столе в текстовом файле с именем «алгоритм».

Решение. Разобьём задачу на подзадачи:

1. Привести Робота в начальное положение; пусть это будет крайнее левое положение, где начинается стена.
2. Закрасить все клетки подряд до конца стены.

Запишем фрагмент алгоритма, приводящего Робота в крайнее левое положение:

Команда	Описание команды

`нц пока не сверху_свободно`
`        влево`
`кц`
`вправо`
	В цикле идём до конца стены. После выполнения цикла Робот окажется на одну клетку левее стены. Поэтому необходимо сместить его на позицию вправо.

Теперь запишем фрагмент алгоритма, заставляющий Робота закрашивать клетки под стеной:

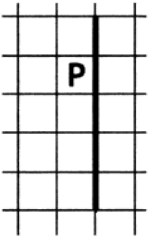
Команда	Описание
нц пока не сверху_свободно закрасить вправо кц	В цикле происходит закраска всех клеток, расположенных под стеной. Цикл завершается, как только достигнута последняя клетка под стеной.

Объединим получившиеся фрагменты в единую программу и сохраним её в файле:

```
нц пока не сверху_свободно
        влево
кц
вправо
нц пока не сверху_свободно
        закрасить
        вправо
кц
```

Пример 7.5. Задание с развёрнутым ответом

На бесконечном поле клеток имеется вертикальная стена. Длина стены неизвестна. Робот размещён в одной из клеток непосредственно слева от стены. Точная начальная позиция Робота неизвестна. Возможное начальное положение Робота приведено на рис. а).

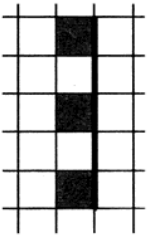


а)

Напишите для Робота алгоритм, закрашивающий клетки, расположенные слева от стены и прилегающие к ней через одну начиная с самой нижней клетки. Например, для приведённого выше рисунка а) Робот должен закрасить следующие клетки (см. рис. б).

Конечное положение Робота может быть произвольным. Алгоритм должен решать задачу для произвольного размера стены и любой допустимой начальной позиции Робота.

Запишите алгоритм в текстовом редакторе и сохраните его на рабочем столе в текстовом файле с именем «алгоритм».


б)

Решение. Разобьём задачу на подзадачи:

1. Привести Робота в начальное положение; пусть это будет крайнее нижнее положение, где начинается стена.
2. Закрасить все клетки через одну до конца стены.

Запишем фрагмент алгоритма, приводящего Робота в крайнее нижнее положение:

Команда	Описание команды
нц пока не справа_свободно вниз кц вверх	В цикле идём до конца стены. После выполнения цикла Робот окажется на одну клетку ниже стены. Поэтому необходимо сместить его на позицию вверх.

Теперь запишем фрагмент алгоритма, заставляющий Робота закрашивать клетки слева от стены через одну:

Команда	Описание команды
нц пока не справа_свободно закрасить вверх если не справа_свободно вверх все кц	В цикле происходит закраска клеток, расположенных слева от стены через одну. В цикл вложен условный оператор, который сдвигает Робота вверх без закрашивания в случае, если справа есть стена. Цикл завершается, как только достигнута последняя клетка слева от стены.

Объединим получившиеся фрагменты в единую программу и сохраним её в файле:

```
нц пока не справа_свободно
    вниз
кц
вверх
нц пока не справа_свободно
    закрасить
    вверх
    если не справа_свободно
        вверх
    все
кц
```

Задания для самостоятельного решения

Задания с выбором одного ответа

Пример 7.6. Значение переменной P, вычисленной в соответствии с фрагментом программы

```
цел P := 1, I := 2
нц пока I <= 4
```

```
      J := I + 1
      нц пока J <= 4
            P := P * (J-I)
            J := J + 1
      кц
      I := I + 1
кц
```

равно

1) 1
2) 2
3) –2
4) 6
5) 8

Пример 7.7. Значение переменной P, вычисленной в соответствии с фрагментом программы

```
цел P := 7, I := 1
нц пока I <= 3
      J := 4-I
      нц пока J >= 1
            P := P + J + I
            J := J-1
      кц
      I := I + 1
кц
```

равно

1) 19
2) 20
3) 27
4) 12
5) 11

Пример 7.8. Значение переменной S после выполнения фрагмента программы

```
цел X := 0, S := 5
нц пока X <= 4
      S := S + X
      X := X + 1
кц
```

равно

1) 9
2) 6
3) 11
4) 15

Пример 7.9. При выполнении фрагмента программы

```
цел X := 1, S := 0
нц пока X >= S
   S := S + X
   X := X + 1
   вывод S," "
кц
```

будет напечатано

1) 13
2) 013
3) 136
4) 0136

Задача 7.10. При выполнении фрагмента программы

```
цел X := 1, S := 0
нц пока X > S
      S := S + X
      X := X + 1
кц
вывод S
```

будет напечатано

1) 3
2) 13
3) 0136
4) 6

Задания с кратким ответом

Пример 7.11. Исполнитель Вычислитель имеет следующую систему пронумерованных команд:

```
1.вычти два;
2.умножь на три.
```

Первая вычитает из числа два, а вторая умножает число на три. Алгоритм, преобразующий число 7 в число 27, записывается в виде последовательности команд `1122`, что соответствует:

```
1. вычти два;
1. вычти два;
2. умножь на три;
2. умножь на три.
```

Запишите порядок команд алгоритма, преобразующего число 4 в число 22, содержащего не более пяти команд.

Пример 7.12. Исполнитель Вычислитель имеет следующую систему пронумерованных команд:

```
1. дели на четыре;
2. прибавь один.
```

Первая делит число на четыре нацело, а вторая прибавляет к числу единицу. Алгоритм, преобразующий число 2 в число 1, записывается в виде последовательности команд `221`, что соответствует:

```
2. прибавь один;
2. прибавь один;
1. дели на четыре.
```

Запишите порядок команд алгоритма, преобразующего число 14 в число 2, содержащего не более пяти команд.

Пример 7.13. Исполнитель Вычислитель имеет следующую систему пронумерованных команд:

```
1. прибавь два;
2. умножь на три.
```

Первая прибавляет к числу два, а вторая умножает число на три. Алгоритм, преобразующий число 5 в число 29, записывается в виде последовательности команд `1121`, что соответствует:

```
1. прибавь два;
1. прибавь два;
2. умножь на три;
1. прибавь два.
```

Запишите порядок команд алгоритма, преобразующего число 2 в число 24, содержащего не более пяти команд.

Пример 7.14. Исполнитель Вычислитель имеет следующую систему пронумерованных команд:

```
1. вычти три;
2. умножь на два.
```

Первая вычитает из числа три, а вторая умножает число на два. Алгоритм, преобразующий число 2 в число 10, записывается в виде последовательности команд `2212`, что соответствует:

```
2. умножь на два;
2. умножь на два;
1. вычти три;
2. умножь на два.
```

Запишите порядок команд алгоритма, преобразующего число 5 в число 11, содержащего не более четырёх команд.

Пример 7.15*. Исполнитель Вычислитель имеет следующую систему пронумерованных команд:

```
1. допиши ноль справа;
2. вычти один.
```

Первая дописывает к числу ноль справа, а вторая вычитает из числа единицу. Алгоритм, преобразующий число 1 в число 890, записывается в виде последовательности команд `12121`, что соответствует:

```
1. допиши ноль справа;
2. вычти один;
1. допиши ноль справа;
2. вычти один;
1. допиши ноль справа.
```

Запишите порядок команд алгоритма, преобразующего число 15 в число 1380, содержащего не более пяти команд.

Пример 7.16*. Исполнитель Вычислитель имеет следующую систему пронумерованных команд:

```
1. допиши ноль справа;
2. вычти три.
```

Первая дописывает к числу ноль справа, а вторая вычитает из числа три. Алгоритм, преобразующий число 7 в число 340, записывается в виде последовательности команд `21221`, что соответствует:

```
2. вычти три;
1. допиши ноль справа;
2. вычти три;
```

```
2. вычти три;
1. допиши ноль справа.
```

Запишите порядок команд алгоритма, преобразующего число 6 в число 5670, содержащего не более пяти команд.

Пример 7.17*. Исполнитель Вычислитель имеет следующую систему пронумерованных команд:

```
1. умножь на два;
2. умножь на три;
3. вычти один.
```

Первая умножает число на два, вторая умножает число на три, а третья вычитает из числа один. Алгоритм, преобразующий число 2 в число 18, записывается в виде последовательности команд 1312, что соответствует:

```
1. умножь на два;
3. вычти один;
1. умножь на два;
2. умножь на три.
```

Запишите порядок команд алгоритма, преобразующего число 4 в число 15, содержащего не более пяти команд.

Пример 7.18*. Исполнитель Вычислитель имеет следующую систему пронумерованных команд:

```
1. умножь на два;
2. умножь на три;
3. прибавь один.
```

Первая умножает число на два, вторая умножает число на три, а третья прибавляет к числу единицу. Алгоритм, преобразующий число 2 в число 26, записывается в виде последовательности команд 2131, что соответствует:

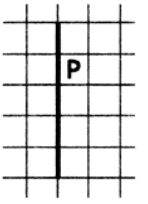
```
2. умножь на три;
1. умножь на два;
3. прибавь один;
1. умножь на два.
```

Запишите порядок команд алгоритма, преобразующего число 3 в число 23, содержащего не более пяти команд.

Задания с развёрнутым ответом

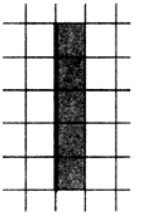
Общая часть формулировки задания приведена на с. 143.

Пример 7.19. На бесконечном поле клеток имеется вертикальная стена. Длина стены неизвестна. Робот размещён в одной из клеток непосредственно справа от стены. Точная начальная позиция Робота неизвестна. Возможное начальное положение Робота приведено на рис. а).



а)

Напишите для Робота алгоритм, закрашивающий все клетки, расположенные справа от стены и прилегающие к ней. Например, для приведённого выше рисунка а) Робот должен закрасить следующие клетки (см. рис. б):

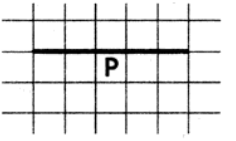


б)

Конечное положение Робота может быть произвольным. Алгоритм должен решать задачу для произвольного размера стены и любой допустимой начальной позиции Робота.

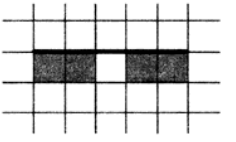
Запишите алгоритм в текстовом редакторе и сохраните на рабочем столе в текстовом файле с именем «алгоритм».

Пример 7.20. На бесконечном поле клеток имеется горизонтальная стена. Длина стены неизвестна. Робот размещён в одной из клеток непосредственно под стеной. Точная начальная позиция Робота неизвестна. Возможное начальное положение Робота приведено на рис. а).



а)

Напишите для Робота алгоритм, закрашивающий каждые две клетки подряд, расположенные под стеной и прилегающие к ней. Например, для приведённого выше рисунка а) Робот должен закрасить следующие клетки (см. рис. б):



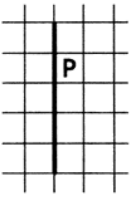
б)

Конечное положение Робота может быть произвольным. Алгоритм должен решать задачу для произвольного размера стены и любой допустимой начальной позиции Робота.

Запишите алгоритм в текстовом редакторе и сохраните на рабочем столе в текстовом файле с именем «алгоритм».

Пример 7.21.

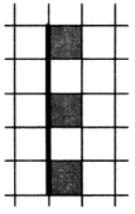
На бесконечном поле клеток имеется вертикальная стена. Длина стены неизвестна. Робот размещён в одной из клеток непосредственно



а)

справа от стены. Точная начальная позиция Робота неизвестна. Возможное начальное положение Робота приведено на рис. а).

Напишите для Робота алгоритм, закрашивающий клетки через одну начиная с самой нижней клетки, расположенные справа от стены и прилегающие к ней. Например, для приведённого выше рис. а) Робот должен закрасить следующие клетки (см. рис. б):



б)

Конечное положение Робота может быть произвольным. Алгоритм должен решать задачу для произвольного размера стены и любой допустимой начальной позиции Робота.

Запишите алгоритм в текстовом редакторе и сохраните на рабочем столе в текстовом файле с именем «алгоритм».

ГЛАВА 8

ТЕКСТОВЫЕ ПРОЦЕССОРЫ

Для работы с текстовыми документами в компьютерах применяются специальные программы — **текстовые редакторы**. Текстовые редакторы позволяют создавать, редактировать, сохранять, выводить на экран и готовить к печати файлы, содержащие текстовую информацию. Примеры текстовых редакторов — блокнот, редактор WordPad, входящие в группу стандартных программ семейства Windows.

Более мощные современные текстовые редакторы — **текстовые процессоры** — имеют значительно больше возможностей по оформлению текстов: подготовка документа к печати, работа с разделами, рисунками и таблицами, автоматическая нумерация рисунков, составление оглавлений и многое другое. Широко известны текстовые процессоры OpenOffice.org Writer, Microsoft Word.

В качестве примера текстового процессора будем рассматривать интерфейс и технологии работы с текстом Microsoft Word 2003.

Любые действия в текстовом процессоре можно выполнить разными способами: с использованием кнопок панели инструментов, горячих клавиш, диалоговых окон, команд меню. В пособии не рассматриваются все технологии оформления текста, приведены лишь некоторые возможности. Более подробно работа с текстовыми процессорами рассматривается в компьютерных практикумах.

В качестве структурных единиц машинописных документов выделяют символы, слова, абзацы, страницы и разделы.

Форматирование текста — процесс оформления текста, изменяющий его внешний вид, оформление, но не содержание. К параметрам форматирования относятся, например, начертание шрифта, между-

строчный интервал, поля и ориентация страницы и др. Для каждой структурной единицы можно установить свои параметры форматирования.

> При оформлении текста можно сначала установить параметры форматирования, а затем набрать текст. Другая возможность — сначала набрать текст, а затем, выделяя его фрагменты, задать параметры форматирования.

Работа с фрагментами текста

Выделение фрагментов текста

Фрагментом называется **непрерывная часть текста**. **Выделить фрагмент** — значит **подсветить его** с помощью мыши или клавиатуры.

Самый простой способ выделения фрагмента — с помощью **протягивания мышью**. Для этого следует установить курсор мыши в начало или конец требуемого фрагмента, нажать и удерживать левую кнопку мыши и перемещать указатель. Протягивать курсор можно как по горизонтали, так и по вертикали, при этом промежуточные строки текста выделяются целиком, а начальная и конечная — по положению курсора.

Выделенная область окрашивается в другой цвет. Для **снятия выделения** достаточно щелкнуть мышью в любой части документа.

Если необходимо выделить **произвольный прямоугольный фрагмент текста**, следует сначала нажать клавишу Alt и затем удерживать её нажатой при выделении фрагмента мышью, как описано выше.

Левое поле документа называют **полосой выделения**. При перемещении курсора мыши в полосу выделения он принимает вид белой стрелки с наклоном вправо (). При этом щелчком мыши можно выделить одну строку. Если же нажать и удерживать левую кнопку мыши, при протягивании выделяются несколько строк.

Быстро выделить слово, предложение, абзац или весь текст можно щелчками мыши. Для этого надо поместить курсор в любое место слова, предложения или абзаца соответственно и выполнить действия, описанные ниже в таблице.

Быстрое выделение фрагментов текста

Выделяемый фрагмент	Действие
слово	двойной щелчок мыши
предложение	Ctrl + щелчок мыши
абзац	тройной щелчок мыши
весь документ	тройной щелчок мыши в полосе выделения (на левом поле)

Другой способ выделения фрагментов текста — **использование клавиатуры**. Для этого следует курсор поместить в начало фрагмента, нажать клавишу Shift, удерживать её и, нажимая клавиши со стрелками влево, вправо, вверх или вниз, переместить курсор в конец фрагмента. Для выделения больших фрагментов можно использовать клавиши Page Up или Page Down. Для выделения текста от текущего положения курсора до начала или конца строки следует при нажатой клавише Shift нажать клавишу Home или End соответственно.

Для выделения всего документа используют клавиши Ctrl + A.

Если требуется **выделить несколько отдельных фрагментов**, расположенных не подряд, выделяют первый фрагмент, затем нажимают клавишу Ctrl и, удерживая её, выделяют остальные фрагменты.

Копирование и перемещение фрагментов текста

При *копировании* исходный фрагмент сохраняется, копия фрагмента может быть помещена в другое место этого же документа, в другой документ или в другое приложение. *Перемещение* фрагмента от копирования отличается тем, что фрагмент не сохраняется на прежнем месте.

Для копирования и перемещения фрагментов документа используется *буфер обмена*.

> **Буфер обмена** (англ. clipboard) — область оперативной памяти компьютера, в которой могут сохраняться данные различных форматов для переноса или копирования их между приложениями или частями одного приложения.

Последовательность действий при копировании или перемещении фрагментов:

1) выделить фрагмент документа;
2) выделенный фрагмент скопировать в буфер обмена с сохранением на прежнем месте или забрать в буфер обмена без сохранения на прежнем месте;
3) курсор поместить на требуемое место в документе и вставить фрагмент из буфера.

	Забрать в буфер (при перемещении)	Скопировать в буфер (при копировании)	Вставить из буфера
Кнопки панели инструментов			
Команда меню «Правка»	Вырезать	Копировать	Вставить
Горячие клавиши	Ctrl + X или Ctrl + Delete	Ctrl + C или Ctrl + Insert	Ctrl + V или Shift + Insert

Более быстрый способ копирования или перемещения фрагментов в пределах одного документа и желательно в пределах одного экрана:

1) выделить требуемый фрагмент;
2) для перемещения — перетащить мышью фрагмент на новое место;
3) для копирования — нажать клавишу Ctrl и, удерживая её, и перетащить мышью фрагмент на новое место.

При таком способе буфер обмена не используется, поэтому невозможно повторно вставить тот же фрагмент в другое место документа.

Копирование формата

Для форматирования фрагмента документа, как правило, требуется выполнить не одно, а несколько действий, например: выбрать начертание шрифта, гарнитуру, размер, расстояние между строчками абзаца. Если в документе уже есть фрагмент с соответствующими параметрами форматирования, можно скопировать формат. Для этого нужно:

1) поместить курсор в любое место фрагмента — источника формата;
2) щёлкнуть на кнопке с кисточкой на панели инструментов  ;
3) протянуть мышь над фрагментом текста, который надо отформатировать так же.

Если необходимо выполнить копирование одного и того же формата на несколько фрагментов текста, надо дважды щёлкнуть на кнопке с кисточкой (она станет утопленной). Таким образом можно копировать формат многократно. Для отмены режима копирования формата щёлкните на утопленной кнопке копирования формата.

> Для **повторного выполнения последней команды** достаточно выполнить подготовительные действия, если они необходимы, например выделить фрагмент документа, и нажать клавишу F4.

Шрифт

Неотъемлемым атрибутом любого текста является **гарнитура (семейство)** шрифта. Гарнитура определяет стилистическое единство рисунка шрифта для его различных размеров и начертаний. Разработкой гарнитур занимаются дизайнеры и художники.

Текстовые процессоры предоставляют широкий выбор гарнитур шрифтов. Чаще других используются шрифты Times New Roman, Arial, Courier New, Comic Sans MS, Verdana.

Примеры начертания шрифтов различных гарнитур и размеров:

Шрифт			
Одной гарнитуры одного размера		**Разных гарнитур одного размера**	
Пример шрифта	Пример шрифта	Пример шрифта	Пример шрифта
Одной гарнитуры разного размера		**Разных гарнитур разного размера**	
Пример шрифта	Пример шрифта	Пример шрифта	Пример шрифта
Одной гарнитуры разного начертания		**Разных гарнитур разного начертания**	
Пример шрифта	Пример шрифта	Пример шрифта	*Пример шрифта*

Среди многообразия гарнитур выделяют гарнитуры **пропорциональные** и `непропорциональные (моноширинные)`. Пропорциональные шрифты состоят из букв различной ширины и считаются более удобными для чтения. Именно поэтому для печатных материалов чаще всего применяются пропорциональные шрифты, например, Times New Roman, который впервые был использован известной газетой «The Times». Моноширинные шрифты, такие, как `Courier New`, с буквами одинаковой ширины достались текстовым процессорам в наследство от пишущих машинок. Непропорциональные шрифты применяются, например, для печати исходных кодов программ в текстовых документах.

Пример пропорционального и непропорционального шрифтов:

Непропорциональный шрифт	Пропорциональный шрифт

`/* Моноширинный шрифт позволяет сохранить структуру программы и сделать код более читаемым */`
`for (i=1; i<100; i++)`
`{`
`  a[i]+=2;`
`}`
	Пропорциональные шрифты лучше смотрятся в полиграфических изданиях, например в газетах и журналах

Пропорциональные и непропорциональные гарнитуры подразделяются на **рубленые (без засечек)** и **с засечками.** Шрифты без засечек, например Arial, Tahoma, Trebuchet MS, широко применяются в презентациях и на веб-страницах. Шрифты с засечками, например Times New Roman, Book Antiqua, `Courier New`, в основном встречаются в книгах и документах.

Шрифт, определённый гарнитурой, может иметь разный размер (кегль). Размеры компьютерных шрифтов и расстояние между строками измеряют в **типографических пунктах** (1 пт – 0,353 мм).

Кроме возможности выделения текста при помощи размера и различных гарнитур, существует возможность изменять **начертание** шрифта. Наиболее популярными являются: *курсив*, **полужирное** и подчёркнутое начертание. Различные начертания могут применяться в комбинациях, например, ***полужирный курсив*** или **полужирный подчёркнутый**.

Не стоит злоупотреблять *различными начертаниями* ***шрифтов***, **иначе текст будет крайне** сложно **понять.**

Практические советы по работе со шрифтами

Управление шрифтами в текстовом процессоре MS Word 2003 осуществляется в диалоговом окне «Шрифт», открыть которое можно командой «Шрифт» пункта главного меню «Формат».

В этом окне имеется две вкладки. Вкладка «Шрифт» предоставляет набор основных настроек внешнего вида шрифта (гарнитуры, начертания, размера, цвета) и дополнительных настроек, таких, как ~~зачеркнутый~~, $^{\text{надстрочный}}$, $_{\text{подстрочный}}$ и др.

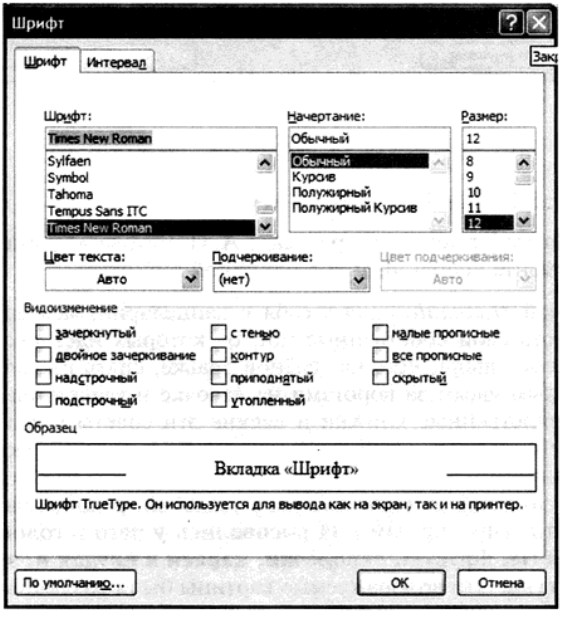


Рис. 20.

Вкладка «Интервал» служит для управления смещением символов в строке по вертикали и изменением расстояния между символами в слове (по горизонтали). Изменение расстояния между символами в сторону увеличения называется р а з р е ж е н и е м, в сторону уменьшения — уплотнением.

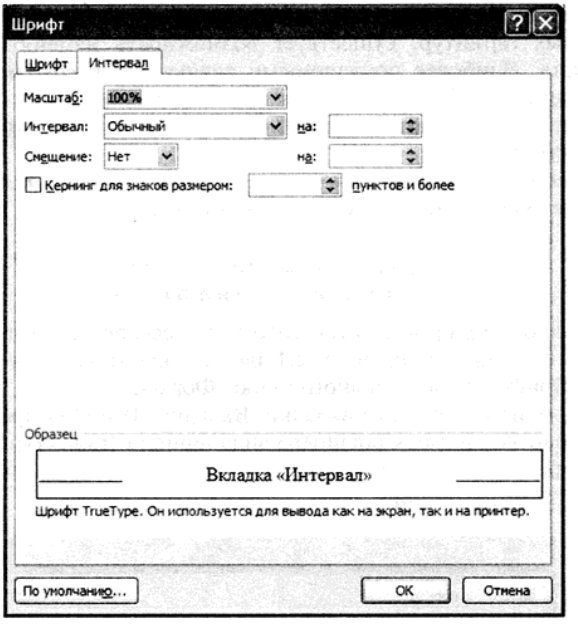


Рис. 21.

Пример 8.1. Задание с кратким ответом

Дан фрагмент текста из рассказа А. П. Чехова «Крыжовник». Для всего фрагмента использован шрифт одной гарнитуры.

Брат мой *Николай*, сидя у себя в канцелярии, мечтал о том, как он будет есть свои собственные щи, от которых идёт такой вкусный запах по всему двору, есть на зелёной травке, спать на солнышке, сидеть по целым часам за воротами на лавочке и глядеть на поле и лес. Сельскохозяйственные книжки и всякие эти советы в календарях составляли его радость, любимую духовную пищу; он любил читать и газеты, но читал в них одни только объявления о том, что продаются столько-то десятин пашни и луга с усадьбой, рекой, садом, мельницей, с проточными прудами. **И рисовались у него в голове дорожки в саду, цветы, фрукты, скворечни, караси в прудах и, знаете, всякая эта штука**. Эти воображаемые картины были различны, смотря по объявлениям, которые попадались ему, но почему-то в каждой из них непременно был крыжовник. Ни одной усадьбы, ни одного поэтического угла он не мог себе представить без того, чтобы там не было крыжовника.

Какие из перечисленных ниже свойств шрифта изменяются в указанном фрагменте текста? В ответе перечислите номера этих свойств в порядке возрастания, например: 123.

1. Начертание шрифта (прямое, курсивное).
2. Насыщенность шрифта (светлый, полужирный, жирный).
3. Размер шрифта.
4. Интервал между символами в словах.

Решение. Весь текст набран шрифтом одного размера. Слово «*Николай*» выделено курсивом, т. е. отличается по начертанию. Предложение «**И рисовались у него в голове дорожки в саду, цветы, фрукты, скворечни, караси в прудах и, знаете, всякая эта штука**» выделено жирным шрифтом, т. е. отличается насыщенностью.

Ответ: 12.

Абзац

В текстовом процессоре **абзац** — это текст, который завершается нажатием клавиши **Enter**. Для оформления разных фрагментов текста может потребоваться разное позиционирование абзацев на странице.

Выравнивание абзацев в тексте

Расположение строк абзаца относительно края документа называется **выравниванием**.

Выделяют четыре типа выравнивания: по левому краю, по правому краю, по центру и по ширине. На панели инструментов «Форматирование» за выравнивание абзацев отвечают иконки, показанные на рис. 22.

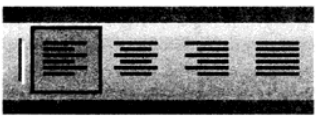


Рис. 22. Кнопки форматирования выравнивания абзаца.

Визуально определить вид выравнивания поможет мысленно проведенная по краям текста линия.

Выравнивание по левому краю:
Блистательный мне был обещан день,
И без плаща я свой покинул дом.
Но облаков меня догнала тень,
Настигла буря с градом и дождём.

По правому краю:
Пускай потом, пробившись из-за туч,
Коснулся нежно моего чела,
Избитого дождём, твой кроткий луч,—
Ты исцелить мне раны не могла.

По центру:

Меня не радует твоя печаль,
Раскаянье твоё не веселит.
Сочувствие обидчика едва ль
Залечит язвы жгучие обид.

По ширине:

Но слез твоих, жемчужных слёз ручьи,
Как ливень, смыли все грехи твои!

Границы абзацев и отступы

Для каждого абзаца можно настроить левую и правую границы и отступ. Они отмеряются от полей документа. На рис. 23 показаны контуры абзацев: слева с отступом и справа с выступом.

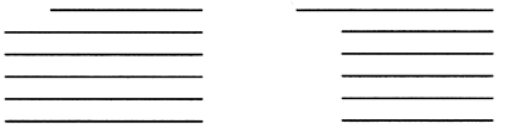


Рис. 23.

Для настройки границ и отступов можно использовать горизонтальную линейку, находящуюся под панелью инструментов над текстом в окне редактирования. Единицы измерения на линейке можно настроить, по умолчанию используются сантиметры.

Рис. 24.

На линейке более тёмным цветом показаны поля страницы, верхний бегунок отмечает положение красной строки (отступ или выступ), нижние бегунки показывают границы текущего абзаца (того, на котором установлен курсор). Поля и бегунки можно перемещать мышью. Линейка на рис. 24 соответствует левому контуру абзаца с отступом на рис. 23.

Междустрочный интервал

Расстояние от нижней кромки одной строки абзаца до верхней кромки следующей строки называется **междустрочный интервалом**. Пользователь может настраивать также **интервал** между абзацами.

Управление абзацами в текстовом процессоре осуществляется при помощи функций, доступных в пункте меню «Формат» — «Абзац». Вкладка «Отступы и интервалы» позволяет устанавливать левую и правую границы и размер отступа абзацев в текстовом документе.

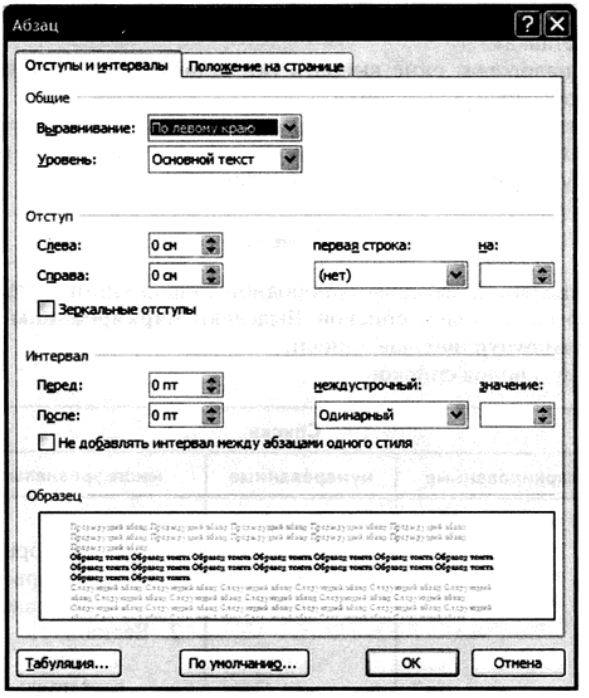


Рис. 25.

Стили

Стиль в текстовом процессоре — это комбинации параметров форматирования шрифта и абзаца. В текстовом процессоре Word имеется богатый набор встроенных стилей, в то же время есть возможность создавать пользовательские стили.

Каждый стиль имеет наименование, например: Обычный, Заголовок 1, Название, Строгий, Цитата и т. д. Именно использование стилей позволяет автоматически создавать оглавление документа, нумеровать рисунки и таблицы.

Рассмотрим создание **оглавления**. Как правило, в текстовом процессоре имеется несколько стилей заголовков, они пронумерованы, начиная с единицы. «Заголовок 1» — это заголовок верхнего уровня, он используется, например, для названия главы. Названия подзаголовков главы могут иметь стиль «Заголовок 2». Для автоматического создания оглавления следует:

1) Задать некоторым абзацам текста стили Заголовок 1, Заголовок 2 и т. д.
2) Установить курсор в том месте документа, где должно разместиться оглавление.

3) Выполнить команду «Ссылки» -> «Оглавления и указатели» меню «Вставка».
4) В диалоговом окне выбрать тип оглавления, установить другие необходимые настройки.

Появится оглавление с номерами страниц. Если навести курсор мыши на номер станицы в оглавлении, нажать клавишу Ctrl и щелкнуть мышью, курсор переместится к заголовку в документе.

Списки

Перечисления, массивы однородной информации в текстах могут быть оформлены в виде списков. Выделяют **маркированные, нумерованные** и **многоуровневые** списки.

Примеры видов списков:

Списки		
маркированные	**нумерованные**	**многоуровневые**
• Зима • Весна • Лето • Осень	1. Зима 2. Весна 3. Лето 4. Осень	1. Зима a. Декабрь b. Январь c. Февраль 2. Весна a. Март b. Апрель c. Май

Для маркированных списков можно настроить вид маркеров. Для нумерованных устанавливается вид и способ нумерации.

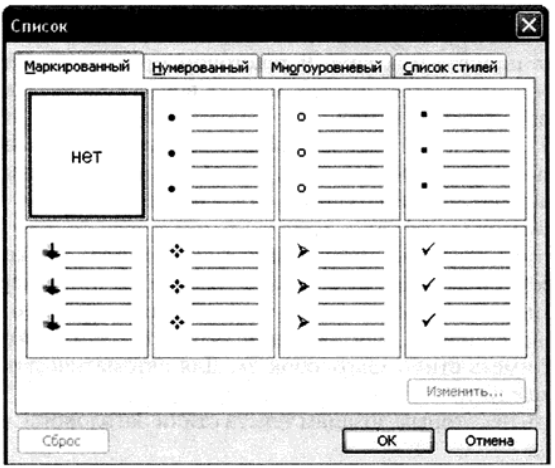


Рис. 26. Окно форматирования списков.

Вставка списка в текст осуществляется при помощи команды «Список» меню «Формат». В открывшемся диалоговом окне можно, например, выбрать вкладку «Маркированный» — для управления внешним видом маркеров списка; вкладку «Нумерованный» — для управления внешним видом нумерованных списков. Вкладка «Многоуровневый» позволяет создать иерархический список.

Таблицы

Некоторую информацию, например справочную, удобно представлять в виде таблиц. Границы таблиц можно сделать видимыми, невидимыми и частично видимыми.

Для вставки таблицы в текст необходимо установить курсор в то место страницы, куда планируется поместить будущую таблицу, и выбрать пункт меню «Таблица» — «Вставить» — «Таблица». Диалоговое окно «Вставка таблицы» позволяет задать количество столбцов и строк.

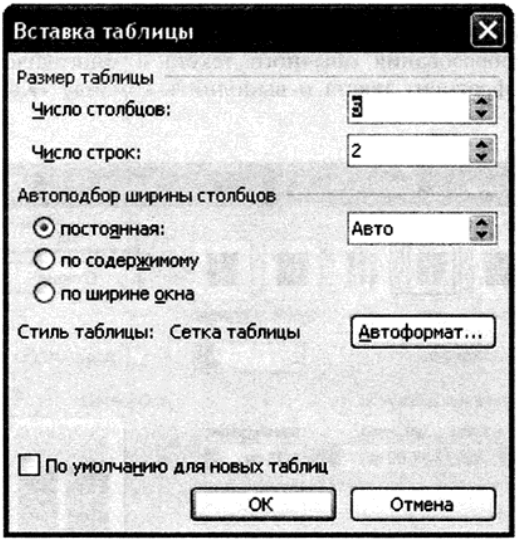


Рис. 27.

Изменить число столбцов или строк созданной таблицы можно при помощи функции «Вставить» и «Удалить» из меню «Таблица».

Объединить несколько ячеек таблицы можно командой «Объединить» в меню «Таблица». Разбить ячейку на несколько более мелких следует командой «Разбить ячейки». В ячейках таблиц может находиться текст, рисунок, другая таблица. Параметры форматирования задаются как для всей таблицы, так и для отдельных строк, столбцов или ячеек.

Многоколонный текст

В зависимости от расположения текста на странице мы можем безошибочно с первого взгляда отличить газету от книги. На страницах газет текст располагается в несколько колонок, в книгах — в одну колонку. На рис. 28 представлены примеры расположения текста в одну и в две колонки.

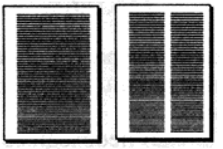


Рис. 28.

Колонки отличаются от таблиц: при использовании многоколонного расположения текст «перетекает» в соседнюю колонку, а в таблицах нет автоматического «перетекания» текста из ячейки в ячейку.

Для преобразования обычного текста в многоколонный следует выделить фрагмент текста и выполнить команду «Колонки» меню «Формат».

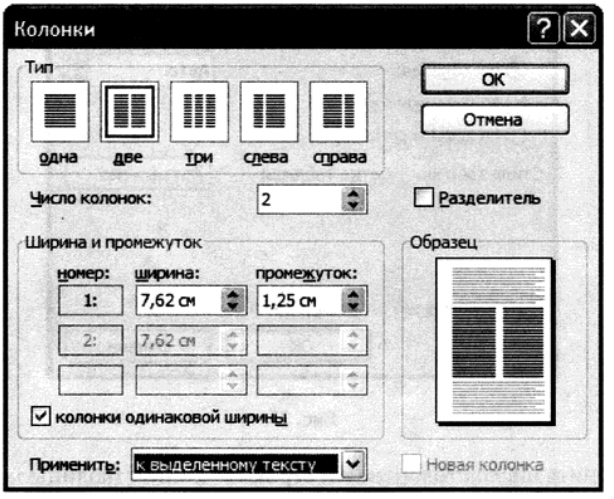


Рис. 29. Диалоговое окно «Колонки».

В диалоговом окне «Колонки» можно выбрать количество колонок, установить расстояние между ними, указать, к какой части документа применить команду (к выделенному фрагменту или к выделенным разделам) и т. д.

Страницы

Печатные материалы: книги, журналы, документы — состоят из страниц текста. Текстовые процессоры разбивают текст на страницы автоматически. Такое деление не всегда удобно с точки зрения представления информации или форматирования документа, поэтому пользователь может принудительно делить текст на страницы нажатием Ctrl + Enter.

К параметрам форматирования страниц относятся ориентации страницы, размеры полей страницы и др. Установка параметров производится в диалоговом окне, которое вызывается командой «Параметры страницы» меню «Файл». Настройки можно применить ко всему документу, выделенному фрагменту, текущему разделу и т. д.

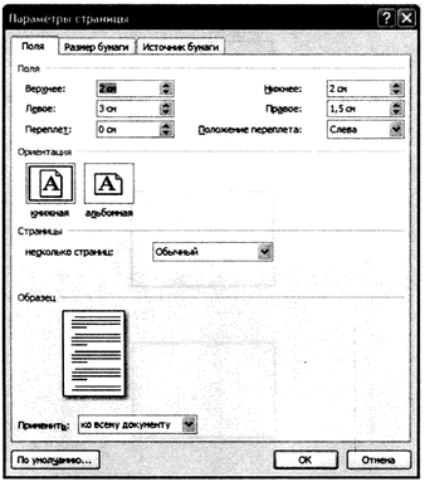


Рис. 30. Диалоговое окно «Параметры страницы».

Поля

Поля документа определяют область, в которой может присутствовать текст на странице. Отсчитываются они от края страницы. В разных документах тексты оформляются полями разных размеров. В докладах и рефератах левое поле обычно бывает шире правого, а развороты книг оформляются зеркальными полями. На рис. 31 показаны зеркальные поля.

Рис. 31.

Ориентация страницы

Страница может иметь одну из двух возможных ориентаций — книжную или альбомную. В книжной ориентации высота страницы больше её ширины, в альбомной — наоборот. Альбомная ориентация используется, например, когда требуется разместить в тексте широкую таблицу.

При работе с документами иногда появляется необходимость использовать страницы разной ориентации (альбомной и книжной) в одном документе.

Если выбрать альбомную ориентацию текущей страницы в документе, все страницы которого имеют книжную ориентацию (рис. 32а), и применить её ко всему документу, все страницы документа изменят вид (рис. 32б). Для того чтобы только отдельные страницы изменили ориентацию, как показано на рис. 32в, применяются **разделы** документа.

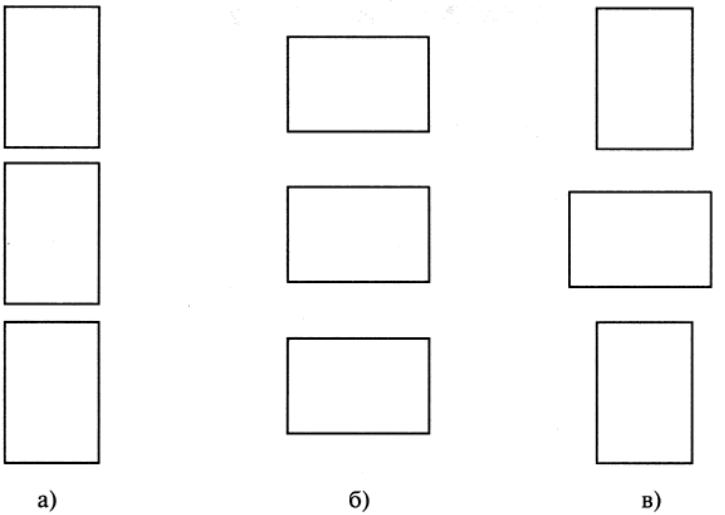


Рис. 32. Ориентация страниц документа.

Колонтитулы

Колонтитул — данные, размещаемые автоматически вверху или внизу всех или нескольких страниц документа. В колонтитулах книг обычно размещают номера страниц, названия разделов, глав, параграфов и т. д. В колонтитул можно поместить небольшой рисунок, дату и время создания документа. Колонтитулы помогают ориентироваться в документе.

Для создания колонтитулов используется команда «Колонтитулы» меню «Вид». Происходит автоматический переход в режим «Разметка страницы», создаются верхний и нижний колонтитулы, основной текст документа становится бледным. Одновременно появляется панель инструментов «Колонтитулы», на которую вынесены основные функции, которые могут понадобиться при работе с колонтитулами — вставка

номера страницы, времени и даты создания документа, полного пути к файлу документа и т. д.

После размещения необходимой информации в верхнем и/или нижнем колонтитуле панель надо закрыть соответствующей кнопкой на панели или двойным щелчком мыши на основном тексте.

Разделы

Раздел — это часть документа, имеющая заданные параметры форматирования страницы. Использование разделов позволяет не только изменять ориентацию отдельных страниц, но и менять колонтитулы в разных фрагментах документа, начинать нумерацию страниц с определённого номера в разделе, использовать разное количество колонок текста и т. д. Раздел может начинаться с новой страницы или продолжаться на текущей.

Для установки начала нового раздела документа используется команда «Разрыв» меню «Вставка», в появившемся окне необходимо выбрать в группе «Разрыв раздела» пункт «Следующая страница» или «Текущая страница».

Начало нового раздела со следующей страницы позволит изменить ориентацию отдельных страниц документа, изменить колонтитулы для раздела и т. д.

Начало нового раздела на текущей странице документа позволит оформить текст с разным количеством колонок в пределах одной страницы.

Сноски

Сноски используются в документе для пояснений, комментариев и ссылок на другие документы. Для пояснений и комментариев лучше использовать **обычные сноски**, которые по умолчанию помещаются внизу страницы, а для ссылок на источники — **концевые сноски**, которые по умолчанию помещаются в конце документа.

Сноска состоит из двух связанных частей: знака сноски и текста сноски. Знак сноски вставляется автоматически, текст сноски следует набрать на клавиатуре.

В MS Word сноски нумеруются автоматически, используется либо сквозная нумерация по всему документу, либо отдельно для каждого раздела.

Для вставки сноски в документ:

1. В режиме разметки укажите место для вставки знака сноски * (например, в этом абзаце вставлена обычная сноска, поясняющая, что такое режим разметки).
2. В меню «Вставка» выберите команду «Ссылка» — > «Сноска...». Появится диалоговое окно (см. рис. 33), с помощью которого можно настроить формат сноски.

---

* Режим отображения документа или других объектов в том виде, какой они будут иметь на бумаге. Например, заголовки, сноски, колонки и надписи занимают свои действительные места.

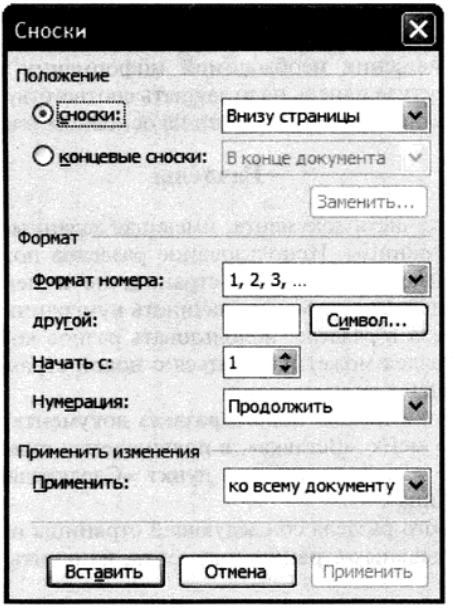


Рис. 33. Диалоговое окно «Сноски».

После нажатия на кнопку «Вставить» в документ будет вставлен знак сноски, а курсор окажется справа от номера сноски внизу страницы (обычная сноска), в конце документа или раздела (концевая сноска).

1. Введите текст сноски.
2. Дважды щёлкните номер сноски для возврата к знаку сноски в документе.

При перемещении, копировании или удалении автоматически нумеруемых сносок оставшиеся знаки сносок автоматически перенумеровываются.

Для удаления сноски следует удалить знак сноски в окне документа, а не текст сноски в области сносок: в документе выделить знак обычной или концевой сноски, которую следует удалить, и нажать клавишу Delete.

Проверка правописания и грамматики

Текстовые процессоры имеют встроенные средства проверки орфографии (правописания) и грамматики. Автоматическая проверка текста может осуществляться либо непосредственно при его наборе, либо после того, как текст набран. Включение и отключение автоматической проверки осуществляется командой «Параметры» меню «Сервис» на вкладке «Правописание».

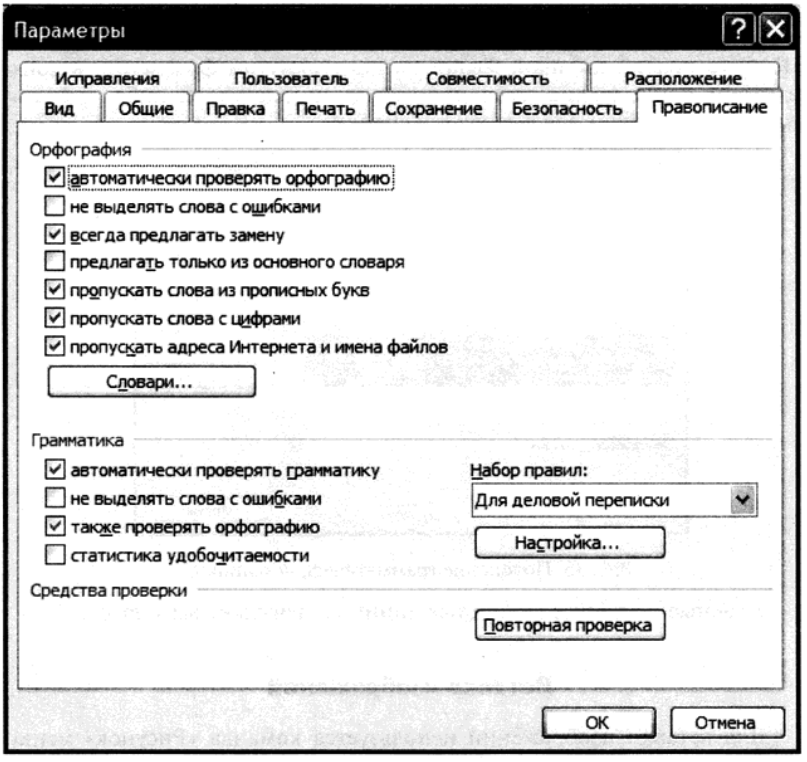


Рис. 34. Вкладка «Правописание» диалогового окна «Параметры».

На рис. 34 показано, что автоматическая проверка как орфографии, так и грамматики включена.

Для проверки орфографии используются словари. Если слово написано с ошибкой или его нет в словаре, оно подчёркивается красной волнистой линией. Щёлкнув правой кнопкой на таком слове, можно

- получить список близких по написанию слов из словаря, одно из которых можно выбрать и заменить ошибочно написанное слово;
- пропустить ошибку;
- добавить неизвестное слово в словарь.

Например, вашей фамилии может не быть в словаре, она будет считаться ошибкой и подчёркиваться. После добавления её в словарь подчёркивания исчезнут.

При проверке грамматики кроме пунктуационных ошибок выявляются также ошибки стиля, например неправильное использование заглавных и строчных букв, повторение одного и того же слова несколько раз подряд, отсутствие пробела между словами, отсутствие второй кавычки и т. п. Все указанные ошибки выявляются на основе сравнения текста документа с хранящимися в памяти правилами.

Если процессор выявляет грамматическую ошибку или ошибку стиля, фрагмент текста подчёркивается зелёной волнистой линией. Щёлкнув правой кнопкой мыши на подчёркнутом фрагменте, в контекстном меню можно либо пропустить предложение, либо выбрать пункт «грамматика» для вывода пояснений.

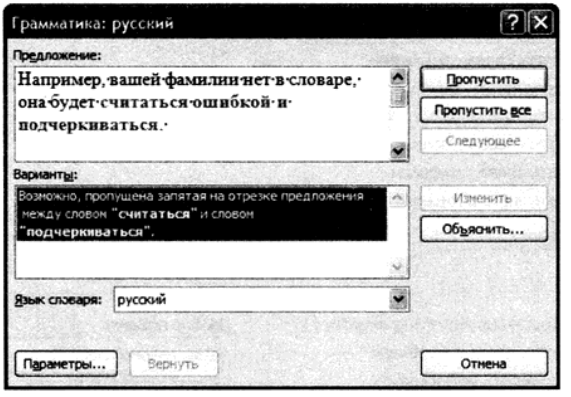


Рис. 35. Пояснение грамматической ошибки.

Красные и зелёные волнистые линии на печать не выводятся.

Вставка изображений

Для вставки изображений используется команда «Рисунок» меню «Вставка».

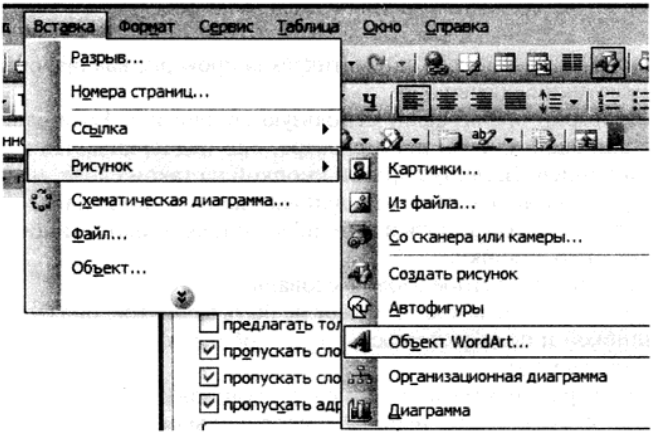


Рис. 36. Команда вставки изображений.

Самостоятельно попытайтесь вставить разные рисунки в документ, изменить их размер, обрезать рисунок, повернуть его и т. д.

Невидимые символы

Символ «Разрыв раздела», так же, как «Разрыв страницы», «Конец абзаца», пробел и некоторые другие называются **скрытыми символами форматирования**, или **непечатаемыми знаками**. Это значит, что они невидимы, если не включён режим отображения невидимых символов кнопкой на панели инструментов  .

Если не понимать значения невидимых символов, они «мешают» работать с документом. Умение работать с невидимыми символами позволяет использовать богатые возможности текстового процессора по оформлению текста. Наиболее полезные символы приведены в таблице.

Невидимые символы

Изображение символа	Название символа
○○○○.....	неразрывные пробелы и обычные пробелы
¶	конец абзаца
↵	принудительный разрыв строки
Разрыв страницы	принудительный разрыв страницы
Разрыв раздела (на текущей странице) Разрыв раздела (со следующей страницы)	разрывы разделов

Правила оформления текстовых документов

Общие правила

Слова в строках должны быть разделены одним пробелом. Интервалы между словами могут увеличиваться автоматически при выравнивании по ширине. В случаях, когда необходимо оставить интервалы между словами равными величине одного пробела, применяют неразрывный пробел: Ctrl + Shift + Пробел.

Клавиша Enter используется только для начала нового абзаца. Для принудительного разрыва строки в текущем абзаце применяется комбинация клавиш Shift + Enter.

Знаки препинания

Все **одинарные знаки** препинания, кроме тире, следуют за текстом без пробелов, но отделяются одним пробелом от текста после них.

Тире в текстовых документах представлено символом «длинное тире» и отделяется от предшествующего и последующего текста

пробелами. Чтобы набрать знак «длинное тире», необходимо, удерживая клавишу Ctrl, набрать знак «минус» на дополнительной цифровой клавиатуре.

Кавычки и **скобки** всегда прилегают к тексту, который окружают. Пробелы ставятся перед открывающими и после закрывающих кавычек и скобок.

Дефис, или «короткое тире», в качестве знака препинания не используется и применяется только в качестве разделителя в сложных словах.

В качестве примера рассмотрим фрагмент правильно отформатированного текста из произведения И. С. Тургенева «Ася». Обратите внимание на оформление знаков препинания:

«Вы находите моё поведение неприличным,— казалось, говорило её лицо,— всё равно: я знаю, вы мной любуетесь».

Формулы

В формулах знаки арифметических операций обрамляются пробелами. Знаки процента % и градуса ° от предыдущего текста пробелом не отделяются. Показатели степени также не отделяются пробелом.

Громоздкие и важные формулы принято выносить в отдельную строку (отдельный абзац), например:

$a \cdot x^2 + b \cdot x + c = 0$

100%

T = 100°

Неразрывный пробел

Неразрывный пробел используется для отделения фамилии от инициалов, при этом инициалы друг от друга пробелом не отделяются. Также неразрывный пробел используется перед единицами измерения и после знаков **№** и **§**, например:

```
§ 1. Вступительное слово
Задача № 56
Автор: И.Я. Иванов
```

Когда не ставится точка

Точки не используются в конце:

- колонтитулов;
- заголовков;
- строк таблиц;
- подписей под рисунками, схемами и диаграммами;
- обозначений системы мер;
- условных сокращённых обозначений.

Как оформлять заголовки

Для форматирования заголовков следует использовать Стили, предлагаемые Word. Для стилей «Заголовок 1», «Заголовок 2» и т. д. определены начертания шрифтов, размещение в строке (по центру, по левому

краю и т. п.) и другие параметры. Для заголовков высоких уровней задаётся параметр форматирования абзаца «Не отрывать от следующего» для того, чтобы не получить заголовок в конце одной страницы, а основной текст на следующей странице. Как правило, абзац, содержащий заголовок, отделяется от предыдущего и последующего текста дополнительными интервалами.

Пользователь имеет возможность изменить встроенные стили или создать свой набор стилей. Но следует учитывать, что параметры форматирования встроенных стилей «Заголовок X» хорошо продуманы разработчиками. Преимущества использования стилей — возможность автоматически создать оглавление, а также работать в режиме структуры документа *, что значительно облегчает работу с большими документами.

Пример 8.2. Задание с выбором одного ответа

В каком из перечисленных ниже предложений правильно расставлены пробелы между словами и знаками препинания?

1) бука —наука, а ребятам — бука.
2) Азбука — наука , а ребятам — бука.
3) Азбука—наука, а ребятам—бука.
4) Азбука — наука, а ребятам — бука.

Решение. Рассмотрим каждую строку из предложенных:

1) после тире отсутствует пробел;
2) запятая отделена от предшествующего текста пробелом;
3) знаки тире не отделены от текста пробелами;
4) соблюдены все правила набора текста.

Ответ: 4.

Пример 8.3. Задание с выбором одного ответа

В каком из перечисленных ниже предложений правильно расставлены пробелы между словами и знаками препинания?

1) Лёгкое горе болтливо , тяжёлое — безмолвно.
2) Лёгкое горе болтливо, тяжёлое — безмолвно.
3) Лёгкое горе болтливо, тяжёлое— безмолвно.
4) Лёгкое горе болтливо, тяжёлое —безмолвно.

Решение.

1) запятая отделена от предшествующего текста пробелом;
2) соблюдены все правила набора текста;
3) и 4) тире не отделено от текста пробелом.

Ответ: 2.

---

* Режимы работы с документами в пособии не рассматриваются, изучите их самостоятельно.

Пример 8.4. Задание с кратким ответом

Даны два фрагмента текста из произведения И. А. Бунина «Антоновские яблоки». В обоих фрагментах используется шрифт одной гарнитуры.

А мальчишки в белых замашных рубашках и коротеньких порточках, с белыми раскрытыми головами, всё подходят. Идут по двое, по трое, мелко перебирая босыми ножками, и косятся на лохматую овчарку, привязанную к яблоне. Покупает, конечно, один, ибо и покупкито всего на копейку или на яйцо, но покупателей много, торговля идёт бойко, и чахоточный мещанин в длинном сюртуке и рыжих сапогах — весел. Вместе с братом, картавым, шустрым полуидиотом, который живёт у него «из милости», он торгует с шуточками, прибаутками и даже иногда «тронет» на тульской гармонике. И до вечера в саду толпится народ, слышится около шалаша смех и говор, а иногда и топот пляски...

К ночи в погоду становится очень холодно и росисто. Надышавшись на гумне ржаным ароматом новой соломы и мякины, бодро идёшь домой к ужину мимо садового вала. Голоса на деревне или скрип ворот раздаются по студёной заре необыкновенно ясно. Темнеет. И вот ещё запах: в саду — костёр, и крепко тянет душистым дымом вишнёвых сучьев. В темноте, в глубине сада — сказочная картина: точно в уголке ада, пылает около шалаша багровое пламя, окружённое мраком, и чьи-то чёрные, точно вырезанные из чёрного дерева силуэты двигаются вокруг костра, меж тем как гигантские тени от них ходят по яблоням. То по всему дереву ляжет чёрная рука в несколько аршин, то чётко нарисуются две ноги — два чёрных столба. И вдруг всё это скользнёт с яблони — и тень упадёт по всей аллее, от шалаша до самой калитки...

Какими из ниже перечисленных свойств шрифта и абзацев различаются левый и правый фрагменты? В ответе перечислите номера различающихся свойств в порядке возрастания, например: 123.

1) начертание шрифта (прямое, курсивное)
2) насыщенность шрифта (светлый, полужирный, жирный)
3) размер шрифт
4) выравнивание строк (по левому краю, по правому краю, по центру, по ширине)

Решение. Текст слева набран курсивом, т. е. отличается начертанием.
Текст слева набран шрифтом большего размера, чем текст справа.
Текст слева выровнен по ширине, текст справа — по правому краю.

Ответ: 134.

Задания для самостоятельного решения

Задания с выбором одного ответа

Пример 8.5. В каком из перечисленных ниже предложений правильно расставлены пробелы между словами и знаками препинания?

1) Работай—сыт будешь;учись—умён будешь.
2) Работай — сыт будешь;учись — умён будешь.
3) Работай — сыт будешь; учись — умён будешь.
4) Работай — сыт будешь ; учись — умён будешь.

Пример 8.6. В каком из перечисленных ниже предложений правильно расставлены пробелы между словами и знаками препинания?

1) Разум — золота краше, а правда — солнца свет.
2) Разум — золота краше,а правда — солнца свет.
3) Разум — золота краше , а правда — солнца свет.
4) Разум —золота краше, а правда —солнца свет.

Пример 8.7. В каком из перечисленных ниже предложений неправильно расставлены пробелы между словами и знаками препинания?

1) Ремесло не коромысло: плеч не отдавит, а век пропитает.
2) Речь вести—не лапти плести.
3) Где родился, там и сгодился.
4) Говорить — смешно, утаить — грешно.

Задания с кратким ответом

Пример 8.8. Даны два фрагмента текста из произведения А. П. Чехова «Дом с мезонином». В обоих фрагментах используется шрифт одной гарнитуры.

А у белых каменных ворот, которые вели со двора в поле, у старинных крепких ворот со львами, стояли две девушки. Одна из них, постарше, тонкая, бледная, очень красивая, с целой копной каштановых волос на голове, с маленьким упрямым ртом, имела строгое выражение и на меня едва обратила внимание; другая же, больше — тоже тонкая и бледная, с большим ртом и с большими глазами, с удивлением посмотрела на меня, когда я проходил мимо, сказала что-то по-английски и сконфузилась, и мне показалось, что и эти

Вскоре после этого, как-то в полдень, когда я и Белокуров гуляли около дома, неожиданно, шурша по траве, въехала во двор рессорная коляска, в которой сидела одна из тех девушек. Это была старшая. Она приехала с подписным листом просить на погорельцев. Не глядя на нас, она очень серьёзно и обтоятельно рассказала нам, сколько сгорело домов в селе Сиянове, сколько мужчин, женщин и детей осталось без крова и что намерен предпринять на первых порах погорельческий комитет,

два милых лица мне давно уже знакомы. И я вернулся домой с таким чувством, как будто видел хороший сон.

членом которого она теперь была. Давши нам подписаться, она спрятала лист и тотчас же стала прощаться.

Какими из ниже перечисленных свойств шрифта и абзацев различаются левый и правый фрагменты? В ответе перечислите номера различающихся свойств в порядке возрастания, например: 123.

1) начертание шрифта (прямое, курсивное)
2) насыщенность шрифта (светлый, полужирный, жирный)
3) размер шрифта
4) выравнивание строк (по левому краю, по правому краю, по центру, по ширине)

Пример 8.9. Даны два фрагмента текста из произведения А. П. Чехова «Дом с мезонином». В обоих фрагментах используется шрифт одной гарнитуры.

В горнице было тепло, сухо и опрятно: новый золотистый образ в левом углу, под ним покрытый чистой суровой скатертью стол, за столом чисто вымытые лавки; кухонная печь, занимавшая дальний правый угол, ново белела мелом; ближе стояло нечто вроде тахты, покрытой пегими попонами, упиравшейся отвалом в бок печи; из-за печной заслонки сладко пахло щами — разварившейся капустой, говядиной и лавровым листом.

Приезжий сбросил на лавку шинель и оказался ещё стройнее в одном мундире и в сапогах, потом снял перчатки и картуз и с усталым видом провёл бледной худой рукой по голове — седые волосы его с начёсами на висках к углам глаз слегка курчавились, красивое удлинённое лицо с тёмными глазами хранило кое-где мелкие следы оспы. В горнице никого не было, и он неприязненно крикнул, приотворив дверь в сенцы:

— Эй, кто там!

Какими из ниже перечисленных свойств шрифта и абзацев различаются левый и правый фрагменты? В ответе перечислите номера различающихся свойств в порядке возрастания, например: 123.

1) начертание шрифта (прямое, курсивное)
2) насыщенность шрифта (светлый, полужирный, жирный)
3) размер шрифта
4) выравнивание строк (по левому краю, по правому краю, по центру, по ширине)

Задания с развёрнутым ответом

Пример 8.10. Создайте в текстовом редакторе документ и напишите в нём следующий текст, оформив его по образцу:

шрифт: Times New Roman, 14 пт;
выравнивание основного текста: по ширине;

отступ первой строки: 1,5 см;
междустрочный интервал: полуторный;
Отдельные слова выделить полужирным шрифтом и курсивом.

Допустимо, чтобы ширина полученного текста отличалась от ширины текста в примере, поскольку ширина текста зависит от размера страницы и полей. Разбиение текста на строки должно соответствовать стандартной ширине абзаца.

Текст сохраните в файле: my_text1.doc.

Квадратным уравнением называется уравнение вида

$$ax^2 + bx + c = 0,$$

где a, b, c — заданные числа, $a \neq 0$, x — неизвестное.

Коэффициенты a, b, c квадратного уравнения обычно называют так: *a — правым* или *старшим коэффициентом*, *b — вторым коэффициентом*, *c — свободным членом*.

$a < 0$	Уравнение не имеет корней
$a > 0$	Уравнение имеет два корня

Пример 8.11. Создайте в текстовом редакторе документ и напишите в нём следующий текст, оформив его по образцу:

шрифт: Arial, 14 пт;
выравнивание основного текста: по ширине;
отступ первой строки: 0 см;
междустрочный интервал: полуторный.
Отдельные слова выделить полужирным шрифтом и курсивом.
Текст списка выровнен по левому краю.

Допустимо, чтобы ширина полученного текста отличалась от ширины текста в примере, поскольку ширина текста зависит от размера страницы и полей. Разбиение текста на строки должно соответствовать стандартной ширине абзаца.

Текст сохраните в файле: my_text3.doc.

Противная задача Григория Остера

Хор, состоящий из
- 280 мальчиков и
- 105 девочек

исполняет задушевную песню. К счастью, лишь четвёртая часть мальчиков и третья часть девочек орёт во всё горло, остальные только открывают рот. Найди разность между мальчиками и девочками, орущими во всё горло.

Р е ш е н и е:

$$X = 280 : 4 - 105 : 3 = 70 - 35 = 35.$$

О т в е т: 35.

ГЛАВА 9

БАЗЫ ДАННЫХ

Первые компьютеры создавались с целью автоматизации вычислений сложных расчётных задач. Для решения каждой задачи — расчёта траектории полёта ракеты, проектирования нового оборудования и т. п. — разрабатывались свои алгоритмы и программы. Быстродействие первых компьютеров было невысоким, они были дорогими, их эксплуатация также требовала больших затрат.

С изменением элементной базы, с появлением интегральных схем, стоимость компьютеров и их эксплуатации постепенно снижалась, компьютеры стали доступны более широкому кругу пользователей. Появилось второе основное направление использования компьютеров — хранение и обработка больших объёмов информации. Стали проводиться исследования по организации хранения данных, которая обеспечивала бы достаточную скорость доступа к данным, поиска необходимой информации, возможность пополнения данных и их изменения и т. д.

В 60-х годах XX века начались работы по созданию и развитию специализированных систем хранения и обработки данных, которые впоследствии стали называть системами управления базами данных (СУБД), а сами хранилища информации получили название «базы данных» (БД).

С базами данных в разных видах — книгах, картотеках — вы сталкивались уже не раз. Это телефонные справочники, словари, каталоги книг в библиотеках и т. д. Информация в них структурирована и упорядочена. Например, в телефонном справочнике записаны упорядоченные по фамилиям владельцев номера телефонов. Информация о книгах в библиотеках хранится на карточках, в которых записаны авторы, наименование книги, её выходные данные, библиотечный шифр. Такие карточки объединяют в библиотечные каталоги, которые существуют в двух вариантах — упорядоченные по фамилиям авторов (алфавитный каталог) и по темам (тематический каталог).

Если же потребуется по телефонному справочнику в виде книги найти владельца телефона по номеру, придётся просмотреть очень много страниц справочника.

Использование компьютеров для хранения баз данных позволяет хранить большие объёмы структурированной информации, легко и быстро сортировать данные в любом порядке, осуществлять быстрый поиск, предоставлять информацию по запросам пользователей и т. д.

> База данных (БД) — это совокупность систематизированных данных определённой предметной области, хранящихся во внешней памяти компьютера.

База данных является **информационной моделью** предметной области.

Предметная область — часть реального мира, сведения о которой необходимо хранить, обрабатывать и использовать для решения задач управления, автоматизации деятельности и т. п.

Предметная область состоит из *объектов* и *связей* между ними.

Объекты предметной области — это предметы, понятия, явления и т. д. Например, объекты предметной области «Библиотека» — это книга, читатель, хранилище (залы, стеллажи, полки), сведения о движении книг и т. д. Каждый объект имеет множество **свойств**. Не все свойства объектов важны для данной предметной области, поэтому из совокупности свойств объектов необходимо выделить наиболее существенные. Например, свойства объекта «книга» — автор, наименование, шифр, год издания важны для библиотеки, а сорт использованных при её печати бумаги или типографской краски не важны. Существенные свойства объекта «читатель» — фамилия, имя, отчество, адрес, номер читательского билета, но не цвет глаз и не размер обуви.

Каждый объект предметной области состоит из *экземпляров*. **Экземпляры объектов** «книга» и «читатель» — это конкретная книга (например, «Информатика. 9 класс») и конкретный читатель (например, Колосов Сергей Петрович).

При разработке информационной модели следует не только выделить объекты предметной области, выбрать наиболее существенные свойства объектов, определить их взаимосвязи, но и выбрать *модель данных*.

Модель данных — это совокупность структур данных и связей между ними *.

В настоящее время промышленным стандартом является *табличная (реляционная) модель данных*.

Реляционная база данных

Реляционная модель данных (англ. relation — отношение) была предложена И. Ф. Коддом в 1970 году. Она основана на теории множеств и математической логике, что обеспечивает её математическую строгость и обуславливает широкое распространение этой модели на практике.

В реляционной БД данные хранятся в виде совокупности взаимосвязанных двумерных таблиц.

Каждая таблица реляционной БД описывает один объект предметной области и состоит из строк и столбцов.

Столбцы таблицы соответствуют свойствам объекта и называются **полями**. Каждое поле имеет имя — заголовок столбца таблицы. В таблице не должно быть двух одинаковых имён полей. Каждое поле имеет определённый тип данных, т. е. все ячейки одного столбца таблицы должны содержать данные одного типа.

* Более полное определение модели данных включает операции обработки данных, но в данном пособии они не рассматриваются.

Напомним, что тип данных определяет:

1) представление данных в памяти (в том числе сколько байтов памяти данные будут занимать);

2) возможные значения данных;

3) допустимые действия над данными указанного типа.

В базах данных используются следующие типы данных: числовой (целое, действительное и т. д.), дата, текстовый, логический и т. д.*

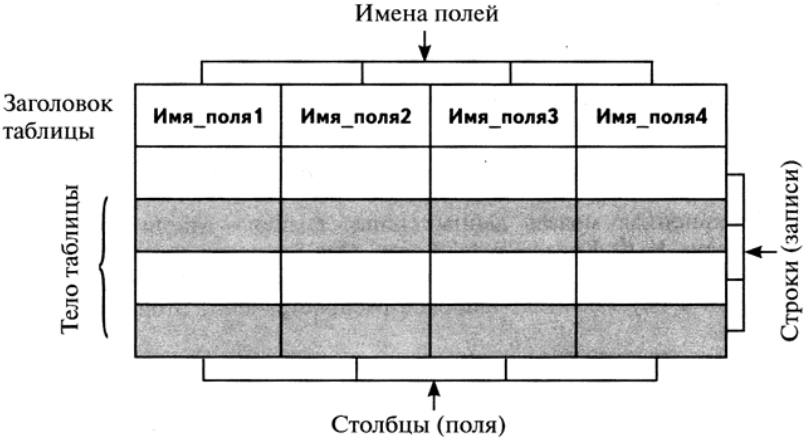
Имена полей образуют **заголовок таблицы**. Он определяет структуру таблицы и не меняется во время работы с базой данных. Если в таблице изменить, добавить или удалить поля, то получим уже другую таблицу (пусть даже с прежним именем).

Строки таблицы называют записями. Каждая строка таблицы описывает один экземпляр объекта. В таблице не должно быть двух одинаковых записей.

Запись — это совокупность данных разного типа, относящихся к одному экземпляру объекта.

Тело таблицы — набор записей об экземплярах объекта. Тело таблицы изменяется во время работы с базой данных — записи об экземплярах объекта могут изменяться, добавляться и удаляться.

Таблица реляционной модели данных



Отметим, что реляционная база данных может состоять как из нескольких, так и из одной таблицы. В данном пособии рассматриваются однотабличные реляционные базы.

Приведём пример базы данных «Телефонный справочник». В справочнике должно храниться: номер телефона, фамилия, имя, отчество владельца телефона, адрес, состоящий из наименования улицы, номера

* В различных СУБД наименования типов могут различаться.

дома (будем считать, что номер дома обозначается одним числом) и номера квартиры. Если все эти данные поместить в одну таблицу, получим примерно следующий список полей:

Имя поля	Тип данных
Телефон	текстовый
Фамилия	текстовый
Имя	текстовый
Отчество	текстовый
Улица	текстовый
Дом	числовой целый
Квартира	числовой целый

Заполненная таблица может иметь вид:

Телефон	Фамилия	Имя	Отчество	Улица	Дом	Квартира
225-64-90	Анастази	Ирина	Михайловна	1-я Первомайская	6	8
159-58-36	Байбурин	Иван	Борисович	Космонавтов	5	56
566-89-74	Власов	Олег	Петрович	Серебристая	58	244
198-34-54	Воеводин	Пётр	Вадимович	Богатырская	8	78

Системы управления базами данных

Для того чтобы использовать базы данных: добавлять, изменять, удалять, сортировать данные и находить нужные сведения, необходимы специальные программы — системы управления базами данных. Существует множество разновидностей СУБД: от простых, с которыми может работать один или несколько пользователей, до очень сложных, на основе которых строятся системы автоматизации крупных организаций, когда сотни людей одновременно работают с огромными объёмами самых разных данных.

СУБД — это специализированное программное обеспечение для создания и использования баз данных: редактирования, просмотра и поиска информации в них.

СУБД различаются по своим возможностям, но, как правило, обеспечивают выполнение следующих функций:

- создание новых БД (создание таблиц и установка связей между ними);
- заполнение БД;
- редактирование БД;
- сортировка данных;
- обработка запросов пользователей;
- поиск информации в БД;
- печать данных в форме отчётов и т. д

Примеры реляционных СУБД: Microsoft Access, MS SQL Server, MySQL и др.

Основные операции с базами данных

С базами данных работают три категории людей: проектировщики, администраторы и пользователи.

При проектировании БД необходимо определить, из каких таблиц будет состоять БД, какие поля будут в каждой таблицы и как таблицы будут связаны между собой.

Администраторы баз данных поддерживают их в работоспособном состоянии.

Пользователи обращаются к БД в поисках нужной информации или с целью получения новой информации путём обработки данных, хранящихся в БД.

Запросы на выборку данных

Обращение пользователя к БД называется **запросом**. Для составления запросов используются специальные языки запросов. Чаще всего используются запросы на выборку данных, в качестве критерия отбора используются *логические выражения*. Логические выражения могут быть *простыми* и *составными*.

В простых логических выражениях накладываются требования на значение поля. Например, если в приведённой выше БД «Телефонный справочник» требуется найти владельцев телефонов с фамилией Волков, условие отбора будет иметь вид: «Фамилия = "Волков"».

Для построения запросов будем использовать базу данных «Туристические поездки». Структура таблицы базы данных:

Имя поля	Тип данных
страна	текстовый
город	текстовый
продолжительность	числовой целый
стоимость	числовой целый
дата_заезда	дата

Заполненная таблица:

Страна	Город	Продолжительность	Стоимость	Дата_заезда
Россия	Москва	5	40 000	14.07.2010
Франция	Париж	7	65 000	14.07.2010
Россия	Сочи	18	70 000	12.06.2010
Великобритания	Лондон	8	70 000	15.07.2010
Италия	Турин	5	50 000	02.08.2010
Италия	Рим	7	65 000	02.09.2010

Для отбора туров продолжительностью от семи дней и больше условие отбора имеет вид: «Продолжительность >= 7».

Составные логические выражения конструируются из простых при помощи логических операций И, ИЛИ, НЕ. Например, в приведённой выше БД «Туристические поездки» необходимо найти туры стоимостью от 50 000 до 68 000. Условие отбора имеет вид: «Стоимость >= 50 000 И Стоимость <= 68 000». Этому условию отвечают три записи, в результате выполнения запроса получим таблицу:

Страна	Город	Продолжительность	Стоимость	Дата_заезда
Франция	Париж	7	65 000	14.07.2010
Италия	Турин	5	50 000	02.08.2010
Италия	Рим	7	65 000	02.09.2010

Пример 9.1. Задание с одним верным ответом

Дана база данных, содержащая сведения об оценках учеников:

№ записи	Фамилия	Имя	Математика	Физика	Информатика
1	Ипатов	Егор	4	4	5
2	Киреев	Александр	5	5	5
3	Илюхин	Сергей	5	4	4
4	Ивашкина	Татьяна	5	5	4

Запросу, содержащему условие отбора

(Физика = 5 или Информатика = 5) и Математика = 5

отвечают записи:

1) 1, 3, 4
2) 4
3) 3, 4
4) 2, 4

Решение. В первую очередь в логическом выражении (условии отбора) выполняется логическая операция ИЛИ, так как она находится в круглых скобках. Условию «Физика = 5 или Информатика = 5» отвечают записи 1, 2 и 4. Условию «Математика = 5» отвечают записи 2, 3, 4. Для того чтобы составное логическое выражение было истинным, необходима истинность обеих его частей. Покажем решение в табличном виде (поля Фамилия и Имя не показаны):

№ записи	Математика	Физика	Информатика	Физика = 5	Информатика = 5	Физика = 5 или Информатика = 5	Математика = 5	(Физика = 5 или Информатика = 5) и Математика = 5
1	4	4	5		да	да		
2	5	5	5	да	да	да	да	да
3	5	4	4					
4	5	5	4	да		да	да	да

Условию отбора отвечают две записи: 2 и 4.

Ответ: 4.

Сортировка данных

Одной из основных операций при работе с базами данных является **сортировка**. При выполнении этой операции требуется указать поля, по которым проводится сортировка, и порядок сортировки (по возрастанию или убыванию значений). Следует учитывать тип данных полей, по которым проводится сортировка.

Текстовые значения (строки) сортируются в лексикографическом порядке в соответствии с кодовой таблицей. Это значит, что две строки сравниваются посимвольно слева направо до первого несовпадающего символа.

Строка, в которой первый несовпадающий символ имеет больший номер в таблице кодировки символов, считается больше. Например, строка «Иванников» меньше строки «Иванов», так как первый несовпадающий символ «н» расположен в кодовой таблице раньше символа «о», имеет меньший номер в кодовой таблице. Строка «СТУДЕНТ» меньше строки «студент», так как заглавные символы в кодовой таблице имеют меньшие номера, чем строчные.

Если строки имеют различную длину, но в общей части символы совпадают, более короткая строка меньше, чем более длинная, например, строка «Иванова» больше строки «Иванов».

Строки равны, если они полностью совпадают по длине и содержат одну и ту же последовательность символов.

Напомним порядок расположения символов в кодовой таблице:

1) символы цифр;
2) заглавные латинские буквы в алфавитном порядке,
3) малые латинские буквы в алфавитном порядке,
4) заглавные русские буквы в алфавитном порядке,
5) малые русские буквы в алфавитном порядке.

Приведём пример сортировки строк. Заданы строки символов латинского алфавита

"ABCD", "Abcd", "abcd", "ABC", "ABc".

После сортировки по возрастанию они расположатся в следующем порядке:

"ABC", "ABCD", "ABc", "Abcd","abcd".

После сортировки по убыванию:

"abcd", "Abcd", "ABc", "ABCD", "ABC".

Ещё один важный пример иллюстрирует, что необходимо учитывать тип данных при выполнении сортировки. Допустим, в БД «Классные руководители» есть поле Класс и поле Классрук.

Класс	Классрук
2	Перова М. В.
8	Савельева Н. К.
11	Дедушкин П. В.
9	Егоров А. А.
1	Воинов А. Д.
10	Егорова К. Д.

Очевидно, поле Классрук имеет тип текстовый. А вот тип поля Класс может быть как числовым, так и текстовым. Рассмотрим сортировку БД в двух случаях:

1) поле Класс имеет тип числовое целое;
2) поле Класс имеет тип текстовый.

После проведения сортировки по возрастанию по полю Класс получим

тип поля Класс — числовое целое	
Класс	**Классрук**
1	Воинов А. Д.
2	Перова М. В.
8	Савельева Н. К.
9	Егоров А. А.
10	Егорова К. Д.
11	Дедушкин П. В.

тип поля Класс — текстовый	
Класс	**Классрук**
1	Воинов А. Д.
10	Егорова К. Д.
11	Дедушкин П. В.
2	Перова М. В.
8	Савельева Н. К.
9	Егоров А. А.

Если первая таблица не вызывает вопросов, то вторая требует пояснений. В кодовой таблице символы цифр расположены в следующем порядке '0', '1', '2',...,'8','9'. Для текстовых значений сравниваются не числа, а последовательности символов. Символ '0' расположен в кодовой таблице раньше символа '1', а символ '1' раньше символа '2', поэтому строка "10" меньше строки "11", а строка "11" меньше строки "2".

Записи (строки таблицы) сортируются целиком.

Отсортируем базу данных «Туристические поездки» по возрастанию стоимости. Получим:

Страна	Город	Продолжительность	Стоимость	Дата_заезда
Россия	Москва	5	40 000	14.07.2010
Италия	Турин	5	50 000	02.08.2010
Франция	Париж	7	65 000	14.07.2010
Италия	Рим	7	65 000	02.09.2010
Россия	Сочи	18	70 000	12.06.2010
Великобритания	Лондон	8	70 000	15.07.2010

Пример 9.2. Задание с выбором одного ответа

Представлена база данных «Телефонный справочник».

Фамилия ИО	Телефон
Иванов И. И.	234-56-98
Иванова А. П.	235-60-07
Кедров А. К.	435-88-78
Иванов И. К.	568-98-00
Иванников П. П.	384-15-15

После проведения сортировки по полю Фамилия ИО в порядке возрастания запись, содержащая сведения о телефоне 384-15-15, переместится:

1) на 1 строку вверх
2) на 2 строки вверх
3) на 3 строки вверх
4) на 4 строки вверх

Решение. Поле Фамилия ИО имеет текстовый тип. После сортировки получим:

Фамилия ИО	Телефон
Иванников П. П.	384-15-15
Иванов И. И.	234-56-98
Иванов И. К.	568-98-00
Иванова А. П.	235-60-07
Кедров А. К.	435-88-78

Запись, содержащая номер телефона 384-15-15, переместилась на 4 строки вверх.

Ответ: 4.

Пример 9.3. Задание с выбором одного ответа

Представлена база данных «Волшебные Страны».

Страна	Население	Площадь
Нарния	148	46,9
Оз	155	95,3
Швамбрания	132	53,5
Лукоморье	199	47,7
Зазеркалье	211	76,2

После проведения сортировки сведения о Нарнии переместились на 1 строку вниз. Сортировка проводилась:

1) в порядке возрастания по полю Страна;
2) в порядке убывания по полю Площадь;
3) в порядке возрастания по полю Население;
4) в порядке убывания по полю Страна.

Решение. Для решения этой задачи придётся провести сортировку для всех вариантов ответов. Поле Страна имеет текстовый тип, сортировка проводится в лексикографическом порядке. При сортировке по возрастанию по полю Страна получим:

Страна	Население	Площадь
Зазеркалье	211	76,2
Лукоморье	199	47,7
Нарния	148	46,9
Оз	155	95,3
Швамбрания	132	53,5

Сведения о Нарнии переместились на 2 строки вниз, ответ 1 не подходит. Если сортировать данные по полю Страна в порядке убывания, Нарния тоже будет находиться в 3-й строке, ответ 4 не подходит.

Проверим ответ 2, отсортируем данные в порядке убывания по полю Площадь.

Страна	Население	Площадь
Оз	155	95,3
Зазеркалье	211	76,2
Швамбрания	132	53,5
Лукоморье	199	47,7
Нарния	148	46,9

Сведения о Нарнии переместились на 4 строки вниз, ответ не подходит. Остаётся проверить ответ 3, отсортировать таблицу в порядке возрастания по полю Население.

Страна	Население	Площадь
Швамбрания	132	53,5
Нарния	148	46,9
Оз	155	95,3
Лукоморье	199	47,7
Зазеркалье	211	76,2

Сведения о Нарнии переместились на 1 строку вниз.

Ответ: 3.

Задания для самостоятельного решения

Задания с выбором одного ответа

Пример 9.4. Во фрагменте базы данных

№	Автор	Серия	Наименование	Год издания	Количество страниц
1	Уолш Р.	Для начинающих	Windows7	2009	128
2	Султанов И.	Для чайников	Энциклопедия Delphi	2005	300
3	Кирсанов Д.	Для чайников	Word 7.0	2008	236
4	Визе М.	Компьютер для носорога	Access	2007	255

запросу, содержащему условие отбора

Серия – «Для чайников» **или** **Год_издания** >2007,

отвечают записи

1) только 3
2) только 1,2,3
3) только 1, 3
4) только 1

Пример 9.5. Для фрагмента базы данных «Детали»

Номер строки	Номер детали	Деталь	Вес	Материал
1	25	Гайка	13	Сталь
2	27	Гайка	18	Латунь
3	41	Шайба	17	Сталь
4	52	Болт	20	Чугун
5	53	Шайба	20	Алюминий

условию отбора

(Деталь <> «Гайка» ИЛИ Деталь= «Шайба») И Вес < 20

отвечают записи с номерами строк

1) только 3
2) только 1, 2
3) только 3, 5
4) 1, 2, 3, 5

Пример 9.6. Представлена база данных «Телефонный справочник»:

Фамилия ИО	Телефон
Иванов И. И.	234-56-98
Иванова А. П.	235-60-07
Кедров А. К.	435-88-78
Иванов И. К.	568-98-00
Иванников П. П.	384-15-15

После проведения сортировки по полю Фамилия ИО в порядке убывания запись, содержащая сведения о телефоне 435-88-78, переместится

1) на 1 строку вверх
2) на 2 строки вверх
3) на 3 строки вверх
4) на 4 строки вверх

Пример 9.7. Представлена база данных «Классы школы»:

Класс	Кол_учеников	Староста
9а	27	Колесник
10а	26	Андреев
8б	30	Чебаев
11а	18	Раков
10б	24	Крупинский

После сортировки в порядке возрастания по полю Класс сведения о 10б классе переместятся:

1) на 4 строки вверх
2) на 1 строку вверх
3) на 3 строки вверх
4) на 2 строки вверх

Пример 9.8. Представлена база данных «Волшебные Страны»:

Страна	Население	Площадь
Нарния	48	46,9
Оз	55	95,3
Швамбрания	32	53,5
Лукоморье	99	47,7
Зазеркалье	51	76,2

После проведения сортировки сведения о Зазеркалье переместились на 2 строки вверх. Сортировка проводилась

1) в порядке возрастания по полю Страна
2) в порядке убывания по полю Площадь
3) в порядке возрастания по полю Площадь
4) в порядке возрастания по полю Население

Пример 9.9. Представлена база данных «Продажи автомобилей»:

Модель	Цена	Продано
BMW	30	5
MERSEDES500	27	8
VAZ21099	10	12
Ford	22	2
UAZ	6	3

После проведения сортировки сведения об автомобиле MERSEDES 500 переместились на 1 строку вниз. Сортировка проводилась

1) в порядке возрастания по полю Цена
2) в порядке убывания по полю Продано
3) в порядке возрастания по полю Модель
4) в порядке убывания по полю Цена

Задания с кратким ответом

Пример 9.10. Дан фрагмент базы данных, содержащей сведения о чемпионате:

Команда	Страна	Всего_игр	Выиграно	Проиграно
Безумные медведи	Швеция	6	1	5
Пронырливые волки	Германия	7	5	2
Угрюмые слоны	Норвегия	5	1	3
Голодные лисы	Франция	5	2	2
Грозные буйволы	Испания	7	4	3

Условию отбора

Всего_игр>5 И **Выиграно**>=2

в заданном фрагменте соответствует _____ записей. В ответе запишите одно число.

Пример 9.11. Дан фрагмент базы данных, содержащей сведения об успеваемости учащихся по информатике:

Фамилия	Класс	Год_рождения	Оценка
Фёдоров	9	1995	3
Сидоркин	11	1994	5
Масленников	10	1995	4
Пастбин	9	1995	5
Шарифуллина	10	1995	4

Поле Класс имеет числовой тип.
Условию отбора

Оценка>4 И **Год_рождения**=1995 И **Класс**<10

в заданном фрагменте соответствует _____ записей. В ответе запишите одно число.

Пример 9.12. Дан фрагмент базы данных, содержащей сведения об успеваемости учащихся по информатике:

Фамилия	Класс	Год_рождения	Оценка
Фёдоров	9	1995	3
Сидоркин	11	1994	5
Масленников	10	1995	4
Пастбин	9	1995	5
Шарифуллина	10	1995	4

Поле Класс имеет текстовый тип.
Условию отбора

Оценка>=4 И **Класс**>10

в заданном фрагменте соответствует _____ записей. В ответе запишите одно число.

Пример 9.13. Дан фрагмент базы данных «Детали»:

Номер строки	Номер детали	Деталь	Вес	Материал
1	25	Гайка	12	Сталь
2	27	Шуруп	18	Латунь
3	41	Шайба	17	Сталь
4	52	Болт	25	Чугун
5	53	Шайба	20	Чугун

Условию отбора

Материал=«Чугун» ИЛИ **Материал**=«Сталь» И **Вес**<20

в заданном фрагменте соответствует _____ записей. В ответе запишите одно число.

Пример 9.14. Дан фрагмент базы данных «Детали»:

Номер строки	Номер детали	Деталь	Вес	Материал
1	25	Гайка	12	Сталь
2	27	Шуруп	18	Латунь
3	41	Шайба	17	Сталь
4	52	Болт	20	Чугун
5	53	Шайба	20	Алюминий

Условию отбора

(**Материал**=«Чугун» ИЛИ **Материал**=«Сталь») И **Вес**>=15

в заданном фрагменте соответствует _____ записей. В ответе запишите одно число.

ГЛАВА 10

ЭЛЕКТРОННЫЕ ТАБЛИЦЫ

Электронные таблицы (ЭТ) — это прикладные программы, предназначенные для математических, финансовых, статистических расчётов, построения диаграмм, ведения простых баз данных. Данные в электронных таблицах хранятся и обрабатываются в прямоугольных таблицах, состоящих из строк и столбцов, на пересечении которых находятся ячейки.

Без преувеличения можно сказать, что появление электронных таблиц в начале 1980-х годов совершило революцию в информационных технологиях, значительно расширив круг пользователей компьютеров. Уже первые программы работы с ЭТ (MultiPlan, SuperCalc и др.) давали возможность проводить большое количество несложных вычислений без программирования, представлять и анализировать полученные результаты с помощью диаграмм и графиков, легко и быстро получать ответ на вопрос «что будет, если...».

Современные прикладные программы для работы с данными в таблицах предоставляют значительно больше возможностей по сравнению с предшественниками, их называют **табличными процессорами (ТП)** или по-прежнему электронными таблицами. Наиболее известными табличными процессорами являются Microsoft Excel, OpenOffice.org Calc.

Основными **задачами** при работе с табличными процессорами являются:

- разработка, создание и редактирование таблиц;
- оформление и печать таблиц;
- построение диаграмм и графиков.

Современные табличные процессоры позволяют работать с электронными таблицами как с **базами данных**: выполнять сортировку списков, выборку данных по запросам, создавать итоговые и сводные таблицы и т. д. Кроме того, в ТП существуют встроенные языки программирования, позволяющие разрабатывать довольно сложные программы для работы с данными.

Для эффективного использования ТП надо знать их основные возможности и технологии разработки. В данном пособии мы будем изучать технологии работы с ТП на примере Microsoft Excel 2003.

Основные понятия и обозначения

При запуске Excel мы увидим примерно то, что показано на рис. 37 (но без выделения строк, столбцов и пр.). Файл, с которым мы начинаем работать, по умолчанию называется `Книга1` и имеет расширение `.xls`. Книга состоит из листов, каждый лист — это прямоугольная таблица.

Ячейка, на которой стоит курсор, называется *активной ячейкой*, она выделяется чёрной рамкой. Данные, находящиеся в активной ячейке, отображаются в строке формул. Строка формул состоит из *поля ввода* и *поля имени*, между ними находится *кнопка вызова Мастера функций*.

При работе с электронными таблицами используются следующие обозначения (см. табл. на с. 194):

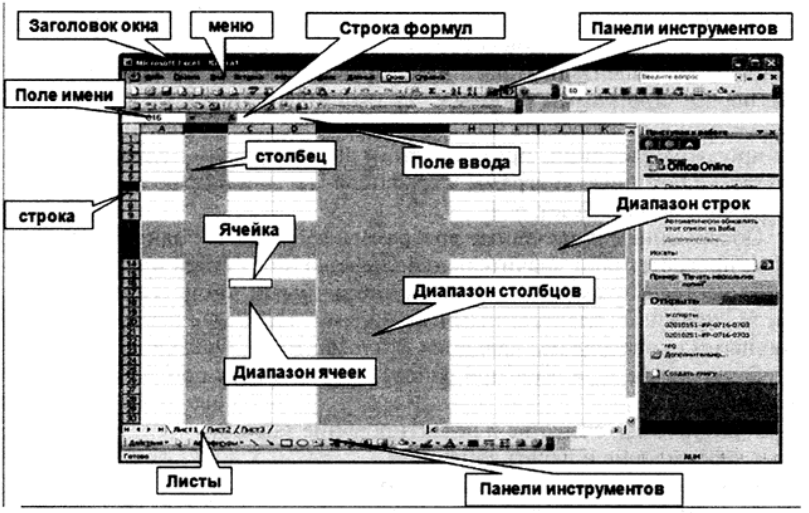


Рис. 37. Окно MS Excel 2003.

Элемент таблицы	Обозначение	Примеры
строка	натуральные числа	1, 2, 3, ..., 256, ...
столбец	буквы латинского алфавита, затем пары символов	A, B, ..., Z, AA, AB, ..., AZ, BA, ..., IV
ячейка (клетка)	имя столбца и номер строки	A1, SZ86
диапазоны строк, столбцов	граничные позиции (номера строк или имена столбцов) в любом порядке, разделённые точкой или двоеточием	5.10 или 10:5 — диапазон строк; C:A или A.C — диапазон столбцов
прямоугольный диапазон (блок ячеек)	диагональные клетки прямоугольного диапазона в любом порядке, разделённые точкой или двоеточием	C1:D5 или F15.A4

Обозначения элементов таблицы определяют **адреса** элементов.

Прямоугольный диапазон содержит несколько ячеек, например: в блок **D7:F3** электронной таблицы входят клетки пяти строк и трёх столбцов, следовательно, в блоке $5 \cdot 3 - 15$ ячеек.

Возможности форматирования для внешнего оформления таблиц аналогичны MS Word:

- форматирования шрифта;
- выравнивание содержимого ячейки;
- заливка и оформление границ ячеек;
- изменение ширины столбцов и высоты ячеек и т. д.

Содержимое ячеек электронной таблицы

Ячейки электронных таблиц могут содержать числа в различных форматах, текст, логические значения и формулы. При вводе данных в ячейку табличный процессор переходит в режим ввода.

Числа

В ЭТ могут храниться целые и вещественные числа, дата и время. Все они имеют свой внутренний формат хранения и формат вывода в ячейку (на экран или на принтер), которые не следует путать. Например, одно и то же вещественное число 12,3 может быть представлено в разных форматах.

Форматы и представление числовых данных

Формат ячейки	Вывод
общий	12,3
процентный	1230%
денежный	12,30 р.
числовой с тремя знаками после запятой	12,300
экспоненциальный (1,23 * 10 ^{1})	1,23E + 01
дробный	12 1/3

Если вместо числа, содержащегося в ячейке, выводится #######, значит, недостаточна ширина столбца (в строке формул число отображается правильно). Ширину столбца можно увеличить мышью, установив курсор на правой границе заголовка столбца (см. рис. 38 на с. 196). Двойной щелчок мыши автоматически увеличивает размер столбца до необходимой ширины.

Дата хранится как целое число — количество дней, прошедших с 01.01.1900 (точка отсчёта может отличаться в других ЭТ). Для внешнего представления дат используется 12 форматов, например: одна и та же дата 27 мая 2010 г. (день проведения ЕГЭ по информатике) может быть представлена в виде:

27.05.2010	27.05.10	27 мая 2010 г.	Май 2010	2010, 27 мая

Так как дата хранится как число, с ней можно выполнять арифметические действия, например вычитание. Так мы можем узнать, сколько дней прошло между двумя датами (см. рис. 39 на с. 196).

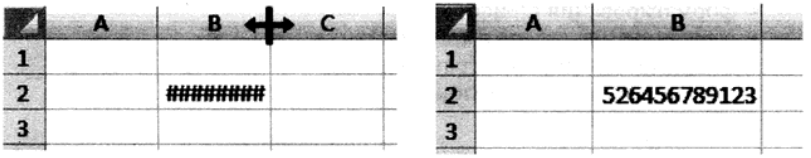


Рис. 38. Изменение ширины столбца.

Рис. 39.

Текст

Текст — это последовательность любых символов. Отображается в ячейке так же, как в строке формул, по умолчанию выравнивается по левому краю ячейки. Обычно электронная таблица автоматически считает текстом всё, что не является числом или формулой. Текст, как правило, используется для оформления таблиц.

Тексты в ЭТ имеют тип «строка». Напомним, что для каждого типа данных определены допустимые операции. Строки нельзя складывать, вычитать, умножать и делить. Над строками в ЭТ, как и в языках программирования, определены операции сцепления и сравнения. В табличных процессорах существует также набор встроенных функций для работы со строками, в Excel они относятся к категории «Текстовые».

Формулы

Формулы — это выражения, по которым проводятся вычисления в таблице. Формула вводится в одну строку, начинается с одного из знаков: равно (=), плюс (+) или минус (–) — и состоит из:

знаков операций (*, /, +, –, ^ *, >, <, >=, <=, =,) и круглых скобок;

операндов (элементов, над которыми выполняются действия).

---

* ^ — операция возведения в степень, например: 2^5 — это два в пятой степени, результат будет равен 32. Такая операция встречается в некоторых языках программирования и в электронных таблицах.

Операции выполняются в соответствии с их приоритетами.

Операндами в формулах могут быть:

константы (числовые, текстовые и т. д.);

встроенные функции;

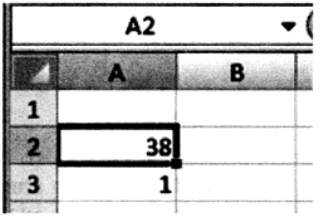
ссылки на ячейки, строки, столбцы, их диапазоны.

Константы — это числа или текстовые значения, введённые непосредственно в формулу. Встроенные функции рассмотрены ниже.

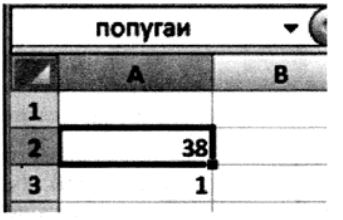
Ссылками в формулах могут быть как адреса ячеек, строк, столбцов и диапазонов, так и имена, заданные пользователем.

Пользователь может задать имя любому элементу таблицы. Для этого используется поле адреса/имени (в левой части строки формул), в которое пользователь вводит с клавиатуры имя активной ячейки (см. рис. 40б, в). Далее эти имена можно использовать в формулах наряду с адресами ячеек (см. рис. 40г). Результат вычислений в ячейках B2 и B3 одинаковый, так как формулы различаются лишь способом указания ссылки.

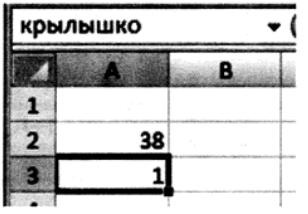
На рис. 40г показана таблица в режиме вывода формул. Для перехода в этот режим выполните команду «Параметры» меню «Сервис», в диалоговом окне на закладке «Вид» установите флажок «формулы». Для возврата в обычный режим флажок надо снять.



а)



б)



в)

	A	B
1		
2	38	=A2+A3
3	1	=попугаи+крылышко

г)

	A	B
1		
2	38	39
3	1	39

д)

Рис. 40. Задание имен ячейкам.

Ссылки являются аналогами переменных в языках программирования.

Если ссылка указывает на ячейку, содержащую константу, вычисления проводятся со значением константы.

Если ссылка указывает на ячейку, содержащую формулу, в расчётах используется результат вычислений по формуле.

Если ссылка указывает на пустую ячейку, эта ячейка не учитывается при расчётах.

Для того чтобы добавить в формулу ссылку на некоторую ячейку необходимо либо вручную набрать её адрес (имя), либо кликнуть по требуемой ячейке мышью.

В таблице, содержащей числа и формулу (см. рис. 41), активной является ячейка C3 — она содержит формулу, которая показана в строке формул. В ячейке C3 выводится результат вычисления по формуле. В поле адреса показан адрес активной ячейки.

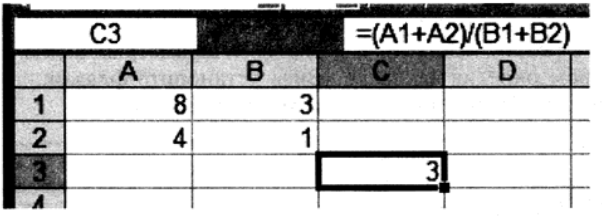


Рис. 41. Таблица, содержащая числа и формулу.

Значение в ячейке C3 зависит от значений в ячейках A1, A2, B1, B2. Ячейку C3 называют **зависимой**. Ячейки A1, A2, B1, B2 называют **влияющими**. Если на ячейке, содержащей формулу, дважды кликнуть мышкой (в данном случае это ячейка C3), то ЭТ переходит в режим ввода: в ячейке отображается формула, а влияющие на неё ячейки выделяются цветными рамками. В поле адреса показываются встроенные функции (см. рис. 42).

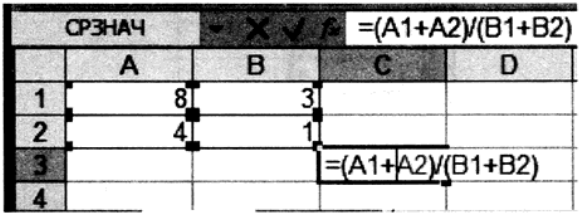


Рис. 42. Пример таблицы в режиме ввода.

При изменении значения в любой влияющей ячейке A1, A2, B1 или B2 результат вычислений в зависимой ячейке C3 тут же изменится.

Виды ссылок

Ссылки на ячейки электронной таблицы могут быть трёх видов:
относительная ссылка — `A1`;
абсолютная ссылка — `$A$1`;
смешанные ссылки — `$A1` или `A$1`.

Относительная часть ссылки специально не помечается. Если при вводе формул указывать ячейку-операнд с помощью мыши, ссылка будет относительной по умолчанию. **Абсолютная часть ссылки** помечается символом $ перед именем строки и/или столбца. Это своего рода «замочек», который не разрешает изменять помеченную им часть ссылки.

Если при вводе формулы нажать клавишу F4, изменится вид ссылки, на которой находится курсор. Вид меняется циклически: `A1->$A$1->A$1->$A1->A1`.

Рассмотрим поведение относительных и абсолютных ссылок при копировании и перемещении ячеек.

Копирование ячеек

Технологии копирования и перемещения ячеек практически не отличаются от копирования и перемещения фрагментов текста.

Для копирования ячеек их необходимо выделить: одну ячейку — щелчком мыши на ячейке, диапазон ячеек — протягиванием мыши с нажатой левой кнопкой либо с помощью клавиатуры клавишами со стрелками при нажатой клавише Shft. Второй шаг — выполнить команду «копировать», третий шаг — выделить ячейку или диапазон ячеек, в которые выполняется копирование. Наконец, последний шаг — выполнить команду «вставить». Команды «копировать» и «вставить» находятся в меню «Правка» и в контекстном меню, которое вызывается щелчком правой кнопки мыши. Эти команды можно выполнить также с помощью кнопок панели инструментов.

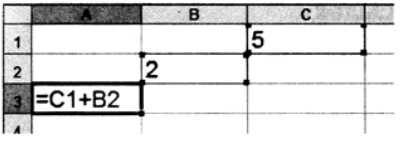
Если ячейки расположены в пределах одного экрана, ячейку можно просто перетащить мышью, удерживая нажатой клавишу Ctrl, курсор при этом должен иметь вид ✢.

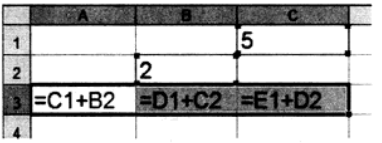
Относительная часть ссылки при копировании ячеек, содержащих формулы, меняется, при этом сохраняется **относительное** взаимное расположение влияющих и зависимых ячеек.

Абсолютная часть ссылки не меняется при копировании ячеек, содержащих формулы.

Рассмотрим примеры. Будем копировать ячейку `A3` в диапазон ячеек `B3:C3` при использовании относительных, абсолютных и смешанных ссылок. Слева показаны таблицы до копирования, справа — после копирования.

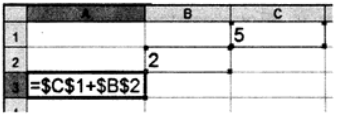
Пример копирования формул с относительными ссылками:

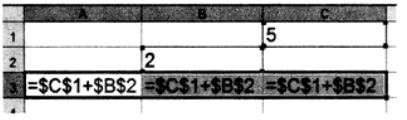




Ссылки C1 и B2 в исходной формуле относительные (влияющая ячейка C1 находится на две строки выше и два столбца левее зависимой ячейки A3, влияющая ячейка B2 — на одну строку левее и один столбец выше зависимой ячейки A3). При копировании формулы ссылки изменились, при этом относительное положение влияющих ячеек на содержащие формулы ячейки B3 и C3 сохранилось.

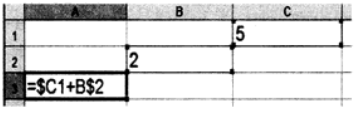
Пример копирования формул с абсолютными ссылками:

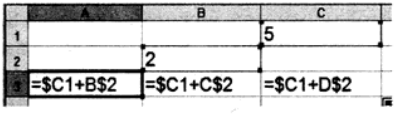




Ссылки C1 и B2 абсолютные, при копировании не изменились.

Пример копирования формул со смешанными ссылками:

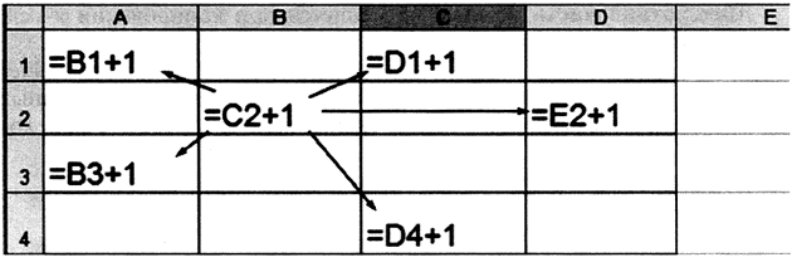




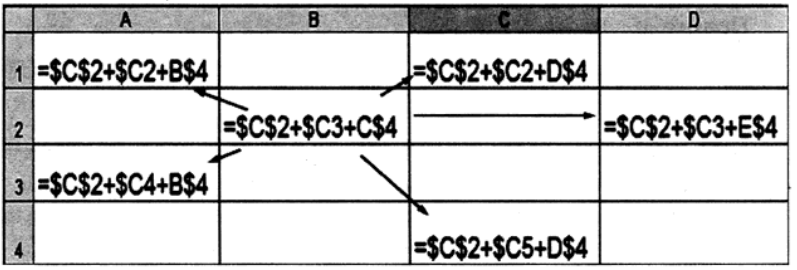
Ссылки $C1 и B$2 смешанные. При копировании относительные части ссылок изменились с сохранением зависимостей, абсолютные части ссылок не изменились.

Приведём ещё два примера копирования формул. Результаты копирования ячейки B2 в ячейки указаны стрелками.

Пример копирования формул с относительными ссылками:



Пример копирования формул с абсолютными и смешанными ссылками:



Попытайтесь самостоятельно пояснить полученные результаты.

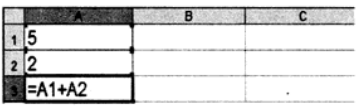
Перемещение ячеек

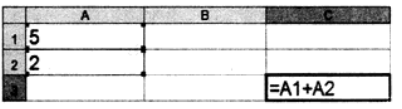
Перемещение ячеек осуществляется так же, как копирование, но вместо команды «копировать» выполняют команду «вырезать». При перетаскивании ячейки мышью не надо удерживать нажатой клавишу Ctrl.

При перемещении ячеек связи между влияющими и зависимыми ячейками сохраняются. Другими словами, после перемещения ячейки результаты вычислений остаются без изменений, меняются только ячейки, в которых эти результаты находятся.

Рассмотрим примеры. В первом и втором примерах ячейка `A3` перемещается в ячейку `C3`. В третьем примере диапазон ячеек `A1:A3` перемещается в диапазон `C1:C3`. Слева показаны исходные таблицы, справа — таблицы после перемещения ячеек.

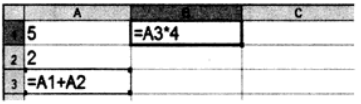
Пример 1:

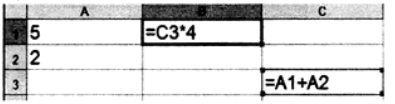




После перемещения ячейки `A3` в ячейку `C3` формула не изменилась.

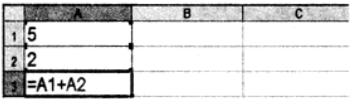
Пример 2:

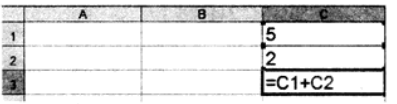




После перемещения ячейки A3 в ячейку C3 формула не изменилась, но изменилась формула в зависимой ячейке B1.

Пример 3:





Перемещается диапазон ячеек A1:A3 в диапазон C1:C3. Формула в C3 изменилась с учётом перемещения влияющих ячеек.

Пример 10.1. Задание с кратким ответом

Дан фрагмент электронной таблицы в режиме отображения формул:

	A	B	C
1	2		
2	= B3 + 1		
3	= A2 + A1	= A1 * 2	= A3 * A2

Результат вычислений в ячейке C3 равен _____ .

Решение. Вычислим значения в ячейках по формулам:

B3 = 2 • 2 – 4;
A2 = 4 + 1 – 5;
A3 = 5 + 2 – 7;
C3 = 7 • 5 – 35.

Ответ: 35.

Рассмотрим более сложные задания.

Пример 10.2*. Задание с выбором одного ответа

Фрагмент электронной таблицы содержит числа и формулы:

	A	B	C
12	7	2	= A12 + B12
13	5,5	4	= A13 * B13
14	6	8	= A14 * B14
15			

После вычислений значение в ячейке C15 равно 12. Ячейка C15 может содержать формулу:

1) =(C12 + C13 + C14)/3
2) =A12 + A13 + B12 + B13
3) =B13 + B14
4) =(A12 + C13)/2

Решение. Выполнив вычисления по формулам в таблице, получим значения в ячейках: C12 – 9, C13 – 22, C14 – 14. Далее проведём вычисления по формулам, предложенным в вариантах ответов:

1) =(C12 + C13 + C14)/3 = (9 + 22 + 14)/3 = 15;
2) =A12 + A13 + B12 + B13 = 7 + 5,5 + 2 + 4 = 18,5;
3) =B13 + B14 = 4 + 8 = 12;
4) =(A12 + C13)/2 = 14,5.

Ответ: 3.

Пример 10.3*. Задание с выбором одного ответа

Дан фрагмент электронной таблицы, содержащий числа и формулы:

	A	B	C
1	10	2	=B1+A1
2	20	15	
3	30	28	

Значение в ячейке C3 после копирования ячейки C1 в ячейки C2:C3 и выполнения вычислений по формулам равно

1) 58 2) 12 3) 35 4) 38

Решение. После копирования в ячейке C3 будет формула =B3+A3. Результат вычислений 28 + 30 = 58.

Ответ: 1.

Пример 10.4*. Задание с выбором одного ответа

Во фрагменте электронной таблицы

	A	B	C
1	1	=A2*4+A3	4
2	2	=$A3+B$1	
3	=A2+A1	2	

содержимое ячейки B2 сначала скопировано в C2, а затем из C2 перемещено в C3. Значение в ячейке C3 равно

1) 4
2) 6
3) 10
4) 7

Решение: После копирования в ячейке C2 появится формула =$A3+C$1, после переноса в C3 формула не изменится. В ячейке A3

результат вычислений 1 + 2 = 3, в ячейке C3 результат вычислений 3 + 4 = 7.

Ответ: 4.

Логические значения

В электронных таблицах используются логические значения ИСТИНА и ЛОЖЬ. Если ввести в ячейки ИСТИНА или ЛОЖЬ с клавиатуры, ЭТ воспримет их не как текст, а как значения логического типа. В памяти эти значения хранятся как 1 и 0 соответственно.

Рассмотрим пример (см. рис. 43). В ячейках A1 и A2 находятся логические значения, в ячейках A3 и A4 — тексты. (Для ввода текста «ЛОЖЬ» необходимо начать ввод с символа ´ (апостроф). На рис. 43а таблица показана в режиме вывода формул, в строке формул — значение активной ячейки A3, которое начинается с апострофа. На рис. 43, показаны результаты вычислений, в строке формул — значение активной ячейки A1.

A3 'ЛОЖЬ

	A	B	C
1	ЛОЖЬ	5	=A1+B1
2	ИСТИНА	5	=A2+B2
3	ЛОЖЬ	5	=A3+B3
4	ИСТИНА	5	=A4+B4

а)

A1 ЛОЖЬ

	A	B	C	D	E
1	ЛОЖЬ	5	5		
2	ИСТИНА	5	6		
3	ЛОЖЬ	5	#ЗНАЧ!		
4	ИСТИНА	5	#ЗНАЧ!		

б)

Рис. 43. Логические значения.

Для текстов операция сложения не определена, поэтому в ячейки C3 и C4 выводится значение ошибки (см. рис. 43б). Так как логические значения ЛОЖЬ и ИСТИНА соответствуют 0 и 1, в ячейках C1 и C2 сложение выполнилось и показаны результаты расчётов (см. рис. 43б).

Логические значения получают в результате выполнения операций сравнения = (равно), < (меньше) , > (больше), <= (меньше или равно), >= (больше или равно), <> (не равно), при вычислении критериев (условий) в некоторых встроенных функциях. На рис. 44 приведены примеры использования операций сравнения в режиме вывода формул (а) и результатов вычислений (б).

В ячейке A5 установлен формат вывода даты, в ячейке B5 то же самое значение выводится в числовом формате. В режиме вывода формул они показаны без учёта форматов, и видно, что A5=B5.

При сопоставлении текстов в ячейках A4 и B4 строки сравниваются в лексикографическом порядке. Это значит, что сравниваются сначала первые символы, если они совпадают — вторые и т. д. Пример лексикографического порядка — расположение слов в орфографических и энциклопедических словарях.

	A	B	C
1	5	6	=A1=B1
2	5	5	=A2=B2
3	музыка	алгебра	=A3=B3
4	Витя	Виталий	=A4>B4
5	40386	40386	=A5<>B5

а)

	A	B	C
1	5	6	ЛОЖЬ
2	5	5	ИСТИНА
3	музыка	алгебра	ЛОЖЬ
4	Витя	Виталий	ИСТИНА
5	27.07.2010	40386	ЛОЖЬ

б)

Рис. 44. Примеры операций сравнения.

Так как для кодирования символов, из которых состоят строки, используются кодовые таблицы, то сравниваются коды (номера) символов в кодовых таблицах. Первые три символа в строках «Витя» и «Виталий» совпадают. Четвёртый символ «я» в строке «Витя» имеет больший номер в кодовой таблице, чем символ «а» в строке «Виталий», следовательно, строка «Витя» больше строки «Виталий».

При грамотной разработке таблиц пользователь вводит не так уж много формул в ячейки. Остальные формулы получают копированием и/или перемещением ячеек. Надо знать, как ведут себя ссылки при копировании и перемещении ячеек (см. темы «Копирование ячеек» и «Перемещение ячеек»).

Диаграммы в электронной таблице

Все табличные процессоры имеют мощные средства построения графиков и диаграмм различных типов. Диаграммы позволяют наглядно представить числовые значения. Исходные данные для диаграмм могут находиться в непрерывных диапазонах или в отдельных ячейках. Элементы списка исходных данных для диаграмм разделяются точкой с запятой. Например, можно построить диаграмму по значениям `A1:A5` или по значениям `A1; A3:A5`.

Диаграммы в Excel динамически связаны с данными, находящимися в таблицах, по которым они построены. Это значит, что при изменении данных в таблицах диаграмма автоматически обновляется.

Построение диаграмм обычно проводится с помощью Мастера диаграмм. Он помогает строить диаграмму шаг за шагом: выбрать тип диаграммы, указать исходные данные, подписи осей и заголовок самой диаграммы и т. д. После создания диаграммы всегда можно изменит её тип, можно добавлять новые ряды данных или изменять текущие, отображая другой диапазон.

Типы диаграмм

Рассмотрим шесть типов диаграмм:
1) круговая;
2) столбчатая (гистограмма);
3) график;
4) с областями;
5) точечная;
6) лепестковая.

Диаграммы будем строить по следующей таблице:

	A	B	C	D	E
1	День недели	входящие	исходящие	вход. мин.	исход. мин.
2	пн	5	4	3,8	8
3	вт	2	8	4,9	3,2
4	ср	5	3	0,7	4,5
5	чт	6	4	3,8	8
6	пт	8	5	4,8	1,25

При построении **круговой диаграммы** (см. рис. 45) используется значение только одной переменной. Весь круг соответствует сумме всех значений, по которым строится диаграмма. Отдельные секторы круга пропорциональны доле одного значения в общей сумме. Построим круговую диаграмму по значениям `A2:B6`. Значения `A2:A6` обозначают категории, значения `B2:B6` — данные для секторов.

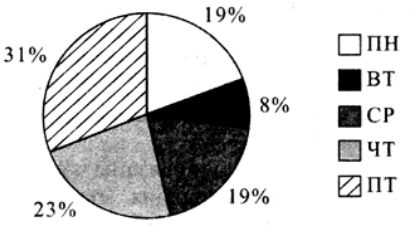


Рис. 45. Построение круговой диаграммы.

Остальные типы диаграмм можно построить по значениям одной или нескольких переменных (см. рис. 46).

На диаграммах типа **гистограмма**, **график**, **с областями** значения переменных откладываются по оси ординат. По оси абсцисс числовые значения из ячеек ЭТ не откладываются. Ось абсцисс разбивается равномерно на части в соответствии с количеством значений одной переменной (остальные переменные, как правило, имеют столько же значений). Ось абсцисс называют **осью категорий**. Единственное, что мы можем сделать — это подписать значения оси абсцисс. Если вы хо-

тите построить график функции, заданной для неравномерно распределённых значений по оси абсцисс, вам это не удастся для таких типов диаграмм. (Название типа диаграммы «график» немного сбивает нас с толку.)

На **лепестковой диаграмме** количество осей, представляющих собой радиусы окружности, равно количеству значений переменной. Вдоль радиусов из центра равномерно откладываются значения от нуля до максимального.

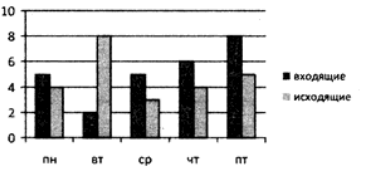
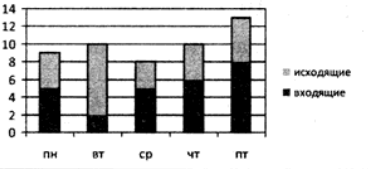
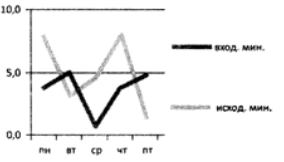
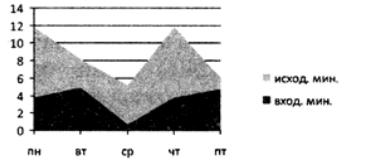
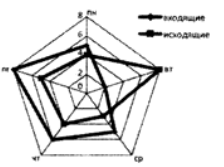
Тип диаграммы	Исходные данные	Вид диаграммы
гистограмма	A2:C6	
гистограмма	A2:C6	
график	A2:C6;D2:E6	
с областями	A2:C6;D2:E6	
лепестковая	A2:C6	

Рис. 46. Построение типов диаграмм.

Если необходимо задавать значения оси абсцисс по данным таблицы, надо использовать тип диаграммы **точечная** (см. рис. 47).

x	sin(x)	cos(x)
–3	–0,14	–0,99
–2	–0,91	–0,42
–1	–0,84	0,54
0	0,00	1,00
1	0,84	0,54
2	0,91	–0,42
3	0,14	–0,99
4	–0,76	–0,65
5	–0,96	0,28

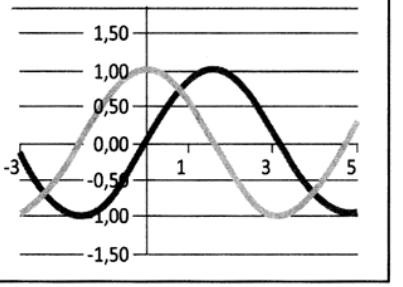
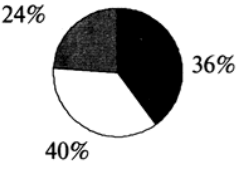


Рис. 47. Построение точечной диаграммы.

Пример 10.5. Задание с выбором одного ответа

Дан фрагмент электронной таблицы:

	AA	AB	AC
7	10	14	10
8	18	20	12
9	12	6	8



Круговая диаграмма построена по значениям диапазона

1) AA7:AC7
2) AA8:AC8
3) AA9:AC9
4) AA7:AA9

Решение. Круговая диаграмма имеет три сектора, значит, построена по трём значениям. Сумма значений соответствует 100%. Из рисунка видно, что среднее значение (36%) составляет 1,5 минимального значения (24%); кроме того, все три значения различны. Диапазон AA7:AC7 не подходит, так как содержит два одинаковых значения. В диапазоне AA8:AC8 выполняется соотношение (18 = 1,5 * 12) и все три значения различны. Остаётся убедиться в том, что остальные варианты ответов не подходят.

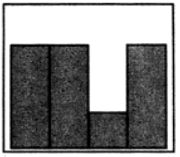
Ответ: 2.

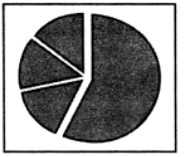
Пример 10.6. Задание с выбором одного ответа

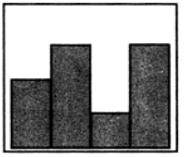
Дан фрагмент электронной таблицы в режиме вывода формул:

	A	B	C	D
1	2		3	
2	=C1-A1/2	=A1 + C1-2	=(C1 + A2)/5	=C2 + A2

По значениям диапазона ячеек A2:D2 была построена диаграмма. Укажите получившуюся диаграмму.


1)


2)


3)


4)

Решение. Вычислим значения в ячейках:

A2 = 3 – 2 : 2 – 3 – 1 = 2;

B2 = 2 + 3 – 2 = 3;

C2 = (3 + 2) : 5 = 1;

D2 = 1 + 2 = 3.

Получили, что два максимальных значения совпадают, два других значения различаются. Этим данным соответствует диаграмма 3.

Ответ: 3.

Пример 10.7. Задание с выбором одного ответа

Дан фрагмент электронной таблицы в режиме вывода формул:

	A	B	C	D
1	7	3	2	
2	=C1*B1-2	=(A2*C1+ +D1)/3	=D1*D2-A1+ +C1/2	=(B1+A2+ +A1)/7

По значениям диапазона ячеек A2:D2 была построена диаграмма

Укажите значение, содержащееся в ячейке D1.

1) 2
2) 5
3) 4
4) 3

Решение. На круговой диаграмме две пары секторов имеют одинаковый размер. Значит, в ячейках A2:D2 пары значений совпадают. Вычислим значения ячеек A1 и D2, так как значения влияющих ячеек известны:

A2 = 2 · 3 − 2 = 6 − 2 = 4;
D2 = (3 + 4 + 7) : 7 = 2.

В ячейках B2 и C2 должны быть числа 2 и 4, так как пары значений совпадают. Решим уравнения для двух вариантов значений.

Первый вариант:
(4 · 2 + D1) : 3 = 4 ⇒ D1 − 4;
D1 · 2 − 7 + 2 : 2 = 2 ⇒ D1 − 4.

Второй вариант:
(4 · 2 + D1) : 3 = 2 ⇒ D1 = −2;
D1 · 2 − 7 + 2 : 2 = 4 ⇒ D1 = 5.

Решение второго варианта нас не устраивает, так как $-2 \neq 5$.

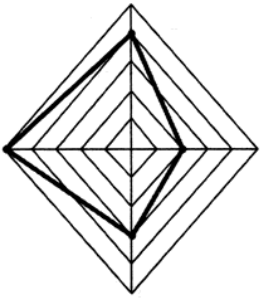
Ответ: 3.

Пример 10.8. Задание с выбором одного ответа

Дан фрагмент электронной таблицы в режиме отображения формул:

	A	B	C	D
1	1		1	2
2	=D1+A1	=D1*A2−5		=C1*3+2

По значениям диапазона ячеек A2:D2 была построена лепестковая диаграмма:



Укажите формулу, которая может содержаться в ячейке B2.

1) =C2−A2+D2
2) =D2*2−C2
3) =A2−C1*2
4) =12−8−C1

Решение. Выполним вычисления, получим

	A	B	C	D
1	2		1	2
2	4	?	3	5

На диаграмме видно, что максимальное значение кратно пяти, что соответствует ячейке D2. В ячейке B2 (горизонтальная ось вправо) должно находиться значение 2.

Рассмотрим ответы:

1) `=C2-A2+D2` = 3 – 4 + 5 = 4;
2) `=D2*2-C2` = 5 · 2 – 3 = 7;
3) `=A2-C1*2` = 4 – 1 · 2 = 2;
4) `=12-8-C1` = 12 – 8 – 1 = 3.

Ответ: 3.

Встроенные функции

Электронные таблицы содержат большой набор встроенных функций. Они помогают выполнять сложные вычисления без программирования. Функции разделены на группы — математические, логические, статистические, финансовые и т. д.

Использование некоторых из них требует специальных знаний, и в школах они не изучаются. Но некоторые несложные функции могут быть использованы при решении заданий итоговой аттестации.

Каждая встроенная функция имеет уникальное имя. При работе с таблицей в Excel используются русские имена функций, например: ЕСЛИ, СУММ. В других табличных процессорах могут использоваться англоязычные имена функций: IIF, SUM.

При обращении к функции в круглых скобках указывается список аргументов, они разделяются точкой с запятой («;»). Функции могут иметь строго определённое количество аргументов, неопределённое количество аргументов, необязательные аргументы или не иметь аргументов. Аргументами могут быть константы, выражения, ссылки на ячейки, диапазоны строк, столбцов, ячеек и др.

Ввод встроенных функций

Для ввода функций в ячейку можно вызвать команду «Функция» меню «Вставка» или использовать кнопку *fx*, расположенную в строке формул. Появится окно Мастера функций (см. рис. 48).

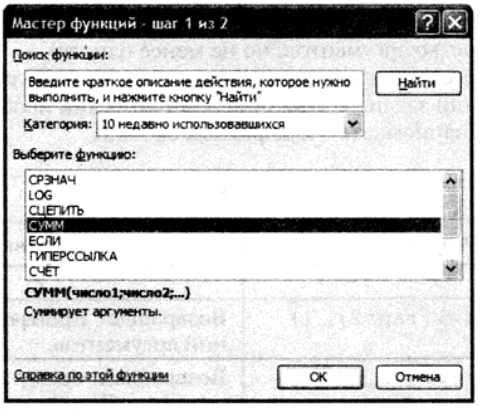


Рис. 48. Первый шаг Мастера функций.

Следует выбрать категорию функций и выделить в списке требуемую. Под списком функций в окне ниже показан её синтаксис и пояснения. После выбора функции (и нажатия кнопки OK) появляется второе окно Мастера функций (см. рис. 49).

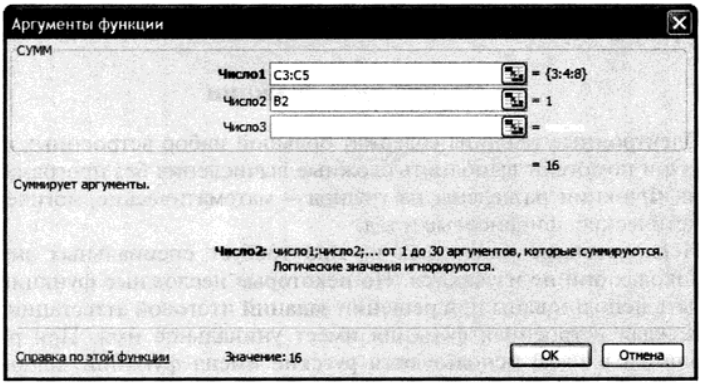


Рис. 49. Второй шаг Мастера функций.

В поля для ввода аргументов можно ввести числа и ссылки. Ссылки вводятся с клавиатуры, но лучше указать их мышкой в таблице.

При вводе функций в ячейку таблицы рекомендуем пользоваться встроенной справкой. Для разных функций действуют разные правила работы с аргументами

Примеры функций:

ПИ() (без аргументов);

КОРЕНЬ (<аргумент>) (всегда один аргумент);

ЕСЛИ (<условие>;<значение_если_истина> [;<значение_если_ложь>]) (может быть два или три аргумента, третий аргумент записан в квадратных скобках, что означает — необязательный аргумент);

СУММ(<аргумент1>[;<аргумент2>;...]) (ограниченное произвольное количество аргументов, но не менее одного).

Ниже рассмотрим функции, которые вы должны уметь использовать при решении заданий. При описании функций необязательные аргументы будем записывать в квадратных скобках.

Математические функции

Функция	Описание
1	2
ПРОИЗВЕД(арг1[;арг2;...])	Возвращает произведение значений аргументов.
СУММ(арг1[;арг2;...])	Возвращает сумму значений аргументов. Логические и текстовые значения игнорируются.

Продолжение таблицы

1	2
СУММЕСЛИ(диапазон1; критерий[;диапазон2])	Суммирует ячейки, удовлетворяющие заданному критерию. диапазон1 — диапазон проверяемых ячеек; критерий — определяет условие, по которому будут выбраны ячейки. Условие можно задать числом, текстом, логическим выражением; диапазон2 — диапазон ячеек, значения которых суммируются. Если третий аргумент не указан, суммируются ячейки из диапазон1, удовлетворяющие **критерию**.

Пример 10.9. Задание с кратким ответом

Представлена таблица в режиме вывода формул:

	A	B	C	D
1	1			
2	1	1		
3	1	1	1	
4	1	1	1	=СУММ(C6;A1:B4)
5		1	1	
6			1	=СУММ(A1:B5;B3:C7;A4:B5)
7		1		

Определите значение в ячейке D6.

Решение. Выполним вычисления в ячейке D6:

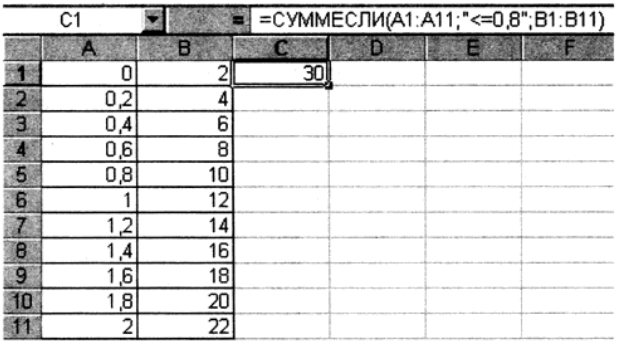
1) сумма значений диапазона A1:B5=8;
2) сумма значений диапазона B3:C7=8;
3) сумма значений диапазона A4:B5=3;
4) общая сумма 8 + 8 + 3 = 19.

Ответ: 19.

Пример 10.10. Задание с кратким ответом

Представлена таблица (см. с. 214).

Определите значение в ячейке C1.



C1	=	=СУММЕСЛИ(A1:A11;"<=0,8";B1:B11)			
	A	B	C	D	E	F
1	0	2	30			
2	0,2	4				
3	0,4	6				
4	0,6	8				
5	0,8	10				
6	1	12				
7	1,2	14				
8	1,4	16				
9	1,6	18				
10	1,8	20				
11	2	22				

Решение. В строке формул показана функция, содержащаяся в ячейке `C1=СУММЕСЛИ(A1:A11;" <= 0,8";B1:B11)`.

Будут суммироваться ячейки из диапазона `B1:B11`, если соответствующие ячейки в диапазоне `A1:A11` содержат значение меньше или равные 0,8. Этому условию соответствуют ячейки диапазона `A1:A5`. Сумма ячеек диапазона `B1:B5` равна 2 + 4 + 6 + 8 + 10 = 30.

Ответ: 30.

Статистические функции

Функция	Описание
1	2
`МАКС(арг1[;арг2;...])`	Возвращает максимальное значение из списка аргументов
`МИН(арг1[;арг2;...])`	Возвращает минимальное значение из списка аргументов
`СРЗНАЧ(арг1[;арг2;...])`	Подсчитывает среднее арифметическое значение аргументов; следует учитывать различия между пустыми ячейками и ячейками, содержащими нулевые значения; пустые ячейки не учитываются, но нулевые значения учитываются.
`СРЗНАЧЕСЛИ(диапазон1; условие[;диапазон2])`	Возвращает среднее арифметическое значение всех ячеек в `диапазоне2`, которое соответствует данному условию. `диапазон1` — диапазон проверяемых ячеек; `условие` — условие в форме числа, выражения, ссылки на ячейку

1	2
	или текста, определяющее те ячейки, которые участвуют в вычислении среднего арифметического значения; например, условие может быть выражено следующим образом: 32, "32", ">32", "Информатика", B4 или ЛОЖЬ; диапазон2 — фактическое множество ячеек для вычисления среднего; если он не указан, вычисляется среднее арифметическое значение ячеек диапазона1, удовлетворяющих **условию**
СЧЕТ(apr1[;apr2;...])	Подсчитывает количество чисел в списке аргументов.
СЧЕТЕСЛИ(диапазон; условие)	Подсчитывает количество непустых ячеек в диапазоне, удовлетворяющих заданному условию. Условие — это число, выражение, ссылка на ячейку или текстовая строка, например: 32, "32", ">32", "Информатика" или B4

Пример 10.11. Задание с кратким ответом

Представлена таблица:

	A	B	C	D
1	1			
2	1	1		
3	1	1	1	
4	2	2	3	
5		5	8	=СРЗНАЧ(A1:C5 A4:C7)
6			1	
7		1		

Определить результат вычислений в ячейке D5.

Решение.: В ячейке D5 вычисляется среднее значение диапазонов, заданных в списке аргументов. Обратите внимание, что диапазоны A1:C5 и A4:C7 разделены не точкой с запятой, а пробелом. Это не ошибка.

Если диапазоны разделены пробелом, в расчётах используется пересечение диапазонов. **Пересечение диапазонов** — это ячейки, которые принадлежат одновременно двум диапазонам.

В этом примере пересечением является диапазон A4:C5.

Вычислим среднее значение диапазона A4:C5:

$$(2 + 2 + 3 + 5 + 8) : 5 = 4.$$

Ответ: 4.

Пример 10.12. Задание с кратким ответом

Представлена таблица:

	A	B	C	D
1	1			
2	1	1		
3	1	1	1	
4	2	aaaaa	3	
5		5	8	=СРЗНАЧ(A1:C5 A4:C7)
6			1	
7		1		

Определить результат вычислений в ячейке D5.

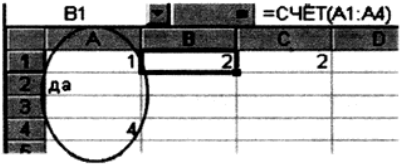
Решение. Пример отличается от предыдущего значением в одной ячейке B4. Она содержит текст «aaaaa» и в расчётах не участвует.

Вычислим среднее значение:

$$(2 + 3 + 5 + 8) : 4 = 4{,}5.$$

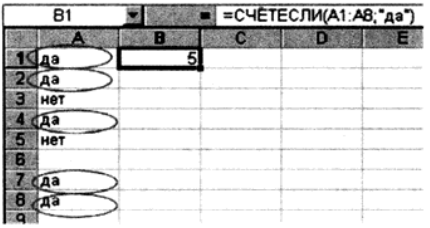
Ответ: 4,5.

Приведём ещё один пример функции, не все аргументы которой являются числом:



В ячейке B1 находится формула =СЧЕТ(A1:A4), показан результат вычислений — число 2, так как в диапазоне A1:A4 только два числа. Ячейка, содержащая текст, и пустая ячейка не учитываются.

Следующий пример иллюстрирует использование функции СЧЕТЕСЛИ.



В ячейке B1 записана формула =СЧЕТЕСЛИ(A1:A8;"да"). Подсчитывается количество ячеек, содержащих текст «да» из диапазона A1:A8, результат равен 5.

Логические функции

В электронных таблицах логические операции реализованы в виде функций. Основные логические функции приведены в таблице ниже.

Функция	Описание
`И(арг1[;арг2;...])`	Возвращает значение ИСТИНА, если значения всех аргументов истинны; если хоть один из аргументов ложный, возвращает значение ЛОЖЬ.
`ИЛИ(арг1[;арг2;...])`	Возвращает значение ИСТИНА, если значение хотя бы одного аргумента истинно; если же все аргументы ложные, возвращает значение ЛОЖЬ.
`ЕСЛИ(арг1;арг2[;арг3])`	`арг1` — логическое выражение; если `арг1` истинно, то возвращается значение аргумента `арг2`; если `арг1` ложно, то возвращается значение аргумента `арг3`.

В функциях И и ИЛИ все аргументы должны иметь логический тип, т. е. аргументами могут быть логические выражения или логические значения.

Рассмотрим пример. В ячейки `B1` на таблицах ниже были записаны формулы, показанные в строке формул. Затем они были скопированы во все ячейки диапазона `B2:B7`. В результате в этих ячейках получены логические значения ИСТИНА или ЛОЖЬ

Функции могут быть **вложенными**, например: `ЕСЛИ(СУММ(A1:A10)>20; ЕСЛИ(G11<0;1;3);5)`.

Максимальный уровень вложений — семь.

Значения ошибок

Если формула содержит ошибку, не позволяющую выполнить вычисления или отобразить результат, Microsoft Excel отобразит в ячейке **значение ошибки**. Каждый тип ошибки вызывается разными причинами.

Значение ошибки	Причины ошибки
#####	недостаточная ширина столбца или дата, время являются отрицательными числами
#ЗНАЧ!	использование недопустимого типа аргумента или операнда
#ДЕЛ/0!	используется ссылка на пустую ячейку или ячейку, содержащую 0 в качестве делителя
#ИМЯ?	Excel не может распознать имя, используемое в формуле
#Н/Д	значение недоступно функции или формуле
#ССЫЛКА!	ссылка на ячейку указана неверно
#ЧИСЛО!	неправильные числовые значения в формуле или функции

Пример 10.13*. Задание с выбором одного ответа

Фрагмент электронной таблицы содержит числа и формулы:

	A	B	C
12	7	2	=A12+B12
13	5,5	4	=A13*B13
14	6	8	=A14+B14
15			

После вычислений значение в ячейке C15 равно 22. Ячейка C15 может содержать формулу:

1) =СРЗНАЧ(C12:C14)
2) =СУММ(A12:B13)
3) =B13+B14
4) =МАКС(A12:C13)

Решение. Выполнив вычисления по формулам, показанным в таблице, получим значения в ячейках: C12=9, C13=22, C14=14. Далее проведём вычисления по формулам, предложенным в вариантах ответов:

1) среднее значение ячеек C12:C14: (9 + 22 + 14) : 3 = 15;

2) сумма ячеек прямоугольного диапазона A12:B13: 7 + 5,5 + 2 + 4 = 18,5;

3) 4 + 8 = 12;

4) максимальное число из чисел прямоугольного диапазона A12:B13 (7; 5,5; 6; 2; 4; 8; 9; 22; 14) равно 22.

Ответ: 4.

Пример 10.14. Задание с выбором одного ответа

Представлен фрагмент электронной таблицы, содержащей числа и формулы:

	J	K	L
22	5	7	9
23	=J22+K22		=K22+L22
24			

После вычисления значение в ячейке L24 равно 8. Ячейка L24 может содержать формулу:

1) =СРЗНАЧ(J22;K23)
2) =СРЗНАЧ(J22:K23)
3) =СРЗНАЧ(L22:K23)
4) =СРЗНАЧ(L23:K23)

Решение. Во всех предложенных вариантах ответов используется функция для вычисления среднего значения. Обратите внимание, что в первом варианте ответа заданы два аргумента, разделённые точкой с запятой. Во всех остальных вариантах мы имеем дело с одним аргументом — диапазоном ячеек.

Выполним вычисления по формулам, показанным в таблице, получим значения в ячейках: J23 5 + 7 = 12, L23 7 + 9 = 16. Затем проведём вычисления по формулам, предложенным в вариантах ответов:

1) среднее значение ячеек J22; K23: 5 : 1 = 5. Ячейка K23 пустая, она не учитывается в расчетах;

2) среднее значение прямоугольного диапазона J22:K23: (5 + 7 + 12) : 3 = 8;

3) среднее значение прямоугольного диапазона L22:K23: (7 + 9 + 16) : 3 = 10,67;

4) среднее значение прямоугольного диапазона L23:K23 равно 16, так как в расчётах учитывается только одна ячейка L23, ячейка K23 пустая и не учитывается в расчётах.

Ответ: 2.

Рассмотрим более сложные примеры на разработку электронной таблицы. Эти задачи относятся к третьей части варианта ГИА, их надо выполнить на компьютере. При решении следует использовать знания о формулах, встроенных функциях, абсолютной и относительной адресации, копировании ячеек. На Государственной итоговой аттестации будет предоставлен файл электронной таблицы, результат работы надо будет сохранить в файле на диске. При выполнении примеров и заданий подобного типа необходимо самостоятельно заполнить исходную таблицу

Пример 10.15*. Задание с развёрнутым ответом

Результаты сдачи выпускных экзаменов по алгебре и русскому языку учащимися 9-х классов некоторого города были занесены в электронную таблицу. На с. 220 приведены первые строки получившейся таблицы:

	A	B	C	D	E
1	Фамилия	Имя	Класс	алгебра	русский язык
2	Кузнецов	Олег	9а	3	5
3	Петров	Николай	9б	5	4
4	Попова	Нина	9б	4	4
5	Иванов	Сергей	9в	5	3

В столбце **A** электронной таблицы записана фамилия учащегося, в столбце **B** — имя учащегося, в столбце **C** — класс, в столбцах **D** и **E** — оценки учащегося по алгебре и русскому языку. Оценки могут принимать значения от 2 до 5. Всего в электронную таблицу были занесены результаты 1000 учащихся.

Выполните задание.

Откройте файл с данной электронной таблицей. На основании данных, содержащихся в этой таблице, ответьте на два вопроса:

1. Какое количество учащихся получило только четвёрки или пятёрки на всех экзаменах? Ответ на этот вопрос (только число) запишите в ячейку **B1002** таблицы.
2. Для группы учащихся, которые получили только четвёрки или пятёрки на всех экзаменах, посчитайте средний балл, полученный ими на экзамене по алгебре. Ответ на этот вопрос (только число) запишите в ячейку `B1003` таблицы.

Полученную таблицу необходимо сохранить под именем, указанным организаторами экзамена.

Решение. Будем решать задачу по шагам.

Шаг 1. Определим, кто из учащихся получил по всем предметам оценки не ниже 3. Будем использовать логическую функцию И, аргументы — ячейки, содержащие оценки. Введём формулу `=И(D2>3; E2>3)` в ячейку `F2`. Результатом будет логическое значение ЛОЖЬ, так как во второй строке оценка по алгебре 3. Если обе оценки удовлетворяют условиям, результат будет ИСТИНА.

Шаг 2. Мы использовали относительные ссылки, поэтому можно скопировать ячейку `F2` в диапазон ячеек `F3:F1000`. Это можно сделать очень быстро, если вы работаете со списком ячеек. Список — это область ЭТ, содержащая данные, окружённая границами ЭТ или пустыми строками и столбцами и имеющая заголовки строк. Соседний со столбцом `F` столбец `E` заполнен значениями, поэтому можно выполнить копирование двойным щелчком мыши, если активной является ячейка `F2`, а курсор мыши подведён к правому нижнему углу ячейки и имеет вид маленького чёрного крестика. Вот как это выглядит в режиме вывода формул:

F2 =И(D2>3;E2>3)

	A	B	C	D	E	F
1	Фамилия	Имя	Класс	алгебра	русский язык	
2	Кузнецов	Олег	9а	3	5	=И(D2>3;E2>3)
3	Петров	Николай	9б	5	4	
4	Попова	Нина	9б	4	4	
5	Иванов	Сергей	9в	5	3	

После копирования в режиме вывода результатов получим следующий результат:

	A	B	C	D	E	F
1	Фамилия	Имя	Класс	алгебра	русский язык	
2	Кузнецов	Олег	9а	3	5	ЛОЖЬ
3	Петров	Николай	9б	5	4	ИСТИНА
4	Попова	Нина	9б	4	4	ИСТИНА
5	Иванов	Сергей	9в	5	3	ЛОЖЬ

Шаг 3. Подсчитаем количество учащихся, получивших только четвёрки или пятёрки на всех экзаменах. Известно, что есть функция `СЧЕТЕСЛИ`, которая подсчитывает количество значений в ячейках, если они удовлетворяют условию. Подсчитаем количество ячеек в диапазоне `F2:F1000`, значение в которых ИСТИНА. В ячейку `B1002` введём формулу

```
СЧЕТЕСЛИ(F2:F1000;ИСТИНА).
```

Маленький секрет — для того чтобы быстро перейти к ячейке `B1002`, поместите курсор в любую ячейку столбца B и нажмите последовательно клавиши End и затем ↓ (стрелка вниз). Так вы можете перемещаться к последним и первым заполненным ячейкам списка как по строке, так и по столбцу (клавиша End и затем соответствующая стрелка).

Шаг 4. Для вычисления среднего значения функция `СРЗНАЧ` здесь не годится, так как требуется выполнение условия — в столбце `F` должна быть ИСТИНА. Воспользуемся функцией `СРЗНАЧЕСЛИ`. В ячейку `B1003` введём

```
=СРЗНАЧЕСЛИ(F2:F1000;ИСТИНА;D2:D1000).
```

Задача решена. Сохраните файл.

Важно! Рекомендуем вам обязательно выполнить проверку разработанной таблицы, протестировать её на небольшом количестве строк. Вы можете сделать это, например, на другом листе книги для пяти-десяти строк исходных данных. Заранее определите правильный результат и сравните его с результатом, полученным в разработанной таблице.

Пример 10.16*. Задание с развёрнутым ответом

Результаты участников олимпиады по алгебре учащихся 9-х классов некоторого города были занесены в электронную таблицу. Приведены первые строки получившейся таблицы:

	A	B	C	D	E	F	G
10	Фамилия	Имя	школа	зад.1	зад.2	зад.3	зад.4
11	Кулагин	Сергей	57	5	6	7	5
12	Сафина	Алина	114	7	7	8	8
13	Балабанов	Николай	114	10	10	10	4
14	Петров	Сергей	2	2	3	5	5

В столбце **A** электронной таблицы записана фамилия учащегося, в столбце **B** — имя учащегося, в столбце **C** — школа, в столбцах **D, E, F, G** — баллы учащегося за каждую задачу. Максимальный балл за каждую задачу — 10. Всего в электронную таблицу были занесены результаты 537 учащихся.

Выполните задание.

Откройте файл с данной электронной таблицей. На основании данных, содержащихся в этой таблице, ответьте на два вопроса:

1. Какое количество учащихся получило за каждую из четырёх задач 5 и более баллов? Ответ на этот вопрос (только число) запишите в ячейку `B7` таблицы.
2. Для группы учащихся, которые получили за каждую задачу не менее 5 баллов, посчитайте, сколько из них набрали не менее 30 баллов в сумме. Ответ на этот вопрос (только число) запишите в ячейку `B8` таблицы.

Полученную таблицу необходимо сохранить под именем, указанным организаторами олимпиады.

Решение. Будем заполнять таблицу немного иначе, чем в примере 10.15. Сначала заполним все необходимые формулы для первого ученика, а затем скопируем ячейки с формулами на весь список. Первая фамилия записана в строку 11, всего строк 537, значит, диапазон заполненных строк 11:547.

Шаг 1. Определим, за сколько задач каждый из учащихся получил не ниже 5 баллов. В ячейку `H11` введем формулу

```
=СЧЕТЕСЛИ(D11:G11;»>4»).
```

Обратите внимание на использование кавычек.

Шаг 2. Подсчитаем сумму баллов каждого участника олимпиады. В ячейку `I11` введем формулу

```
=СУММ(D11:G11).
```

Шаг 3. Для определения, выполняются ли требуемые условия, введём в ячейку `J11` формулу

=И(H11=4;I11>=30).

Шаг 4. Скопируем введённые формулы на весь список. Копируем ячейки H11:J11 в диапазон H12:J547.

После копирования в режиме вывода результатов получим:

	A	B	C	D	E	F	G	H	I	J
10	Фамилия	Имя	школа	зад.1	зад.2	зад.3	зад.4			
11	Кулагин	Сергей	57	5	6	7	5	4	23	ЛОЖЬ
12	Сафина	Алина	114	7	7	8	8	4	30	ИСТИНА
13	Балабанов	Николай	114	10	10	10	4	3	34	ЛОЖЬ
14	Петров	Сергей	2	2	3	5	5	2	15	ЛОЖЬ

Шаг 5. В ячейку B7 введём формулу для подсчёта количества учащихся, получивших за каждую задачу не меньше 5 баллов:

=СЧЕТЕСЛИ(H11:H547;4).

Шаг 6. В ячейку B8 введём формулу для определения количества учащихся, набравших за каждую задачу не меньше 5 баллов, а в сумме не меньше 30 баллов:

=СЧЕТЕСЛИ(J11:J547;ИСТИНА).

Полученная таблица в режиме отображения формул будет иметь вид:

	A	B	C	D	E	F	G	H	I	J
7		=СЧЁТЕСЛИ(H11:H547;4)								
8		=СЧЁТЕСЛИ(J11:J547;ИСТИНА)								
9										
10	Фамилия	Имя	школа	зад.1	зад.2	зад.3	зад.4			
11	Кулагин	Сергей	57	5	6	7	5	=СЧЁТЕСЛИ(D11:G11;">4")	=СУММ(D11:G11)	=И(H11=4;I11>=30)
12	Сафина	Алина	114	7	7	8	8	=СЧЁТЕСЛИ(D12:G12;">4")	=СУММ(D12:G12)	=И(H12=4;I12>=30)
13	Балабанов	Николай	114	10	10	10	4	=СЧЁТЕСЛИ(D13:G13;">4")	=СУММ(D13:G13)	=И(H13=4;I13>=30)
14	Петров	Сергей	2	2	3	5	5	=СЧЁТЕСЛИ(D14:G14;">4")	=СУММ(D14:G14)	=И(H14=4;I14>=30)

Задача решена. Сохраните файл.

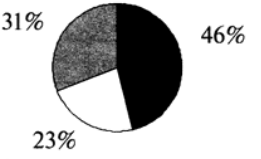
При решении этих двух примеров вы убедились, что при грамотной разработке электронных таблиц надо вводить немного формул. Затем надо использовать копирование ячеек и/или диапазонов ячеек. Используйте встроенные функции для решения задач.

Задания для самостоятельного решения

Задания с выбором одного ответа

Пример 10.17. Дан фрагмент электронной таблицы:

	AA	AB	AC
7	10	14	10
8	18	20	12
9	12	6	8



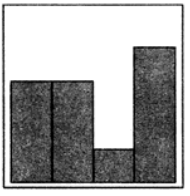
Круговая диаграмма построена по значениям:

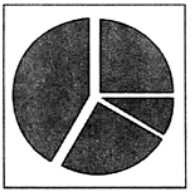
1) AA7:AC7
2) AA8:AC8
3) AA9:AC9
4) AA7:AA9

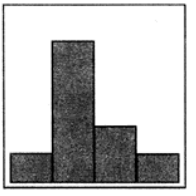
Пример 10.18. Дан фрагмент электронной таблицы в режиме отображения формул:

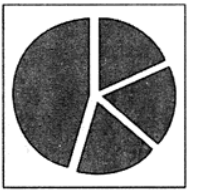
	A	B	C	D
1	4	5		
2	=(B1+A1)/3	=A1-A2	=(B1+B2)/2	=A1+B2

По значениям диапазона ячеек A2:D2 была построена диаграмма. Укажите получившуюся диаграмму.


1)


2)


3)

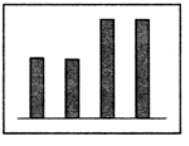

4)

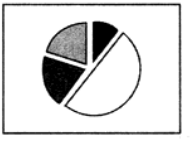
Пример 10.19. Дан фрагмент электронной таблицы в режиме отображения формул:

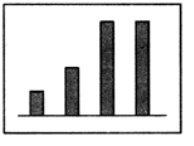
	A	B	C	D
1	2	5	1	6
2	=C1*A1	=A1*2–B1+C1	=(A1+B1+C1)/2	=A2*B1-D1

По значениям диапазона ячеек A2:D2 была построена диаграмма. Укажите получившуюся диаграмму.


1)


2)

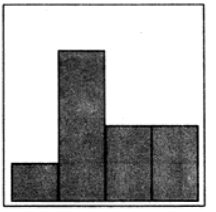

3)


4)

Пример 10.20. Дан фрагмент электронной таблицы в режиме отображения формул:

	A	B	C	D
1	2	5	1	
2	=C1*A1	=СУММ(A1:C1)	=(A1+B1+C1)/2	=A2*B1-D1

По значениям диапазона ячеек A2:D2 была построена диаграмма.



Укажите значение, содержащееся в ячейке D1:

1) 3
2) 5
3) 6
4) 8

Пример 10.21. Дан фрагмент электронной таблицы в режиме отображения формул:

	A	B	C	D
1	1	3	2	0
2	=D1*B1+C1	=СУММ(A1:D1)		=9-B1*2

По значениям диапазона ячеек A2:D2 была построена лепестковая диаграмма.

Укажите формулу, которая может содержаться в ячейке C2.

1) =A2*2+A1
2) =B1/3+5*A1
3) =СУММ(A1:B2)
4) =B2+A1

Пример 10.22. Представлен фрагмент электронной таблицы, содержащей числа и формулы:

	J	K	L
22	5	7	9
23	=J22+K22		=K22+L22
24			

После вычисления значение в ячейке L24 равно 7. Ячейка L24 может содержать формулу:

1) =СРЗНАЧ(J22;K23)
2) =СРЗНАЧ(J22:K23)
3) =СРЗНАЧ(L22:K23)
4) =СРЗНАЧ(J22:L22)

Задания с кратким ответом

Пример 10.23. Дан фрагмент электронной таблицы в режиме отображения формул:

	A	B	C
11	5	2	3
12	=A11*2+B11	=B11+A11	=B12+A12

Результат вычислений в ячейке C12 равен _____ .

Пример 10.24. Дан фрагмент электронной таблицы в режиме отображения формул:

	A	B	C
11	3		5
12	=A11*2+B11	=B11+A11+C11	=B12+A12

Результат вычислений в ячейке C12 равен 18. В ячейке B11 находится значение _____ .

Пример 10.25. Дан фрагмент электронной таблицы в режиме отображения формул:

	K	L	M
1	3	=K1*5	=K1+K2/2
2	4		=M1+L1

Результат вычислений в ячейке M2 равен _____ .

Пример 10.26. Дан фрагмент электронной таблицы в режиме отображения формул:

	K	L	M
1	3	=K1*5	=K1+K2/2
2	?		=M1+L1

Результат вычислений в ячейке M2 равен 19. В ячейке K2 находится значение _____ .

Пример 10.27. Дан фрагмент электронной таблицы в режиме отображения формул:

	A	B	C
14	2	1	=СУММ(A14:B14)
15	1	2	
16	2	2	

Ячейку C14 скопировали в ячейку C16. Результат вычислений в ячейке C16 равен _____ .

Пример 10.28. Представлен фрагмент электронной таблицы в режиме отображения формул:

	A	B
1	4	=ЕСЛИ(И(A1<8; A1>3);»да»;»нет»)
2	3	
3	8	
4	10	
5	4	
6	1	
7	2	
8		=СЧЁТЕСЛИ(B1:B7; «=да»)

Значение в ячейке B8 после копирования формулы из B1 в B2:B7 будет равно _____ .

Задания с развёрнутым ответом

Пример 10.29. Результаты сдачи Государственной итоговой аттестации по информатике, алгебре и русскому языку учащимися 9-х классов некоторого города были занесены в электронную таблицу. Приведены первые строки получившейся таблицы:

	A	B	C	D	E	F
1		Фамилия	Имя	Информа-тика	Алгебра	Русский язык
2				40	38	25
3	1	Рощина	Татьяна	22	40	30
4	2	Кузьмина	Елена	44	50	70
5	3	Гдлян	Анаида	66	60	50
6	4	Коршунов	Сергей	88	45	78

В столбце **B** электронной таблицы записана фамилия учащегося, в столбце **C** — имя учащегося, в столбцах **D, E** и **F** — баллы учащегося по информатике, алгебре и русскому языку. Баллы могут принимать значения от 0 до 100. Всего в электронную таблицу были занесены результаты 457 учащихся.

В ячейках D2, E2, F2 записаны минимальные баллы положительных оценок по каждому предмету. Например, если учащийся набрал меньше 40 баллов по информатике, он получает школьную оценку 2.

Выполните задание.

Откройте файл с данной электронной таблицей. На основании данных, содержащихся в этой таблице, ответьте на два вопроса:

1. Какое количество учащихся получило положительные результаты на всех экзаменах? Ответ на этот вопрос (только число) запишите в ячейку `B1` таблицы.
2. Для группы учащихся, которые получили только положительные результаты на всех экзаменах, посчитайте количество учащихся, набравших в сумме больше 180 баллов. Ответ на этот вопрос (только число) запишите в ячейку `C1` таблицы.

Полученную таблицу необходимо сохранить.

Пример 10.30. Для исходных данных предыдущей задачи (см. пример 10.29) выполните задание

1. Какое количество учащихся получило положительные результаты на всех экзаменах? Ответ на этот вопрос (только число) запишите в ячейку `B1` таблицы.
2. Для группы учащихся, которые получили только положительные результаты на всех экзаменах, посчитайте количество учащихся, набравших по алгебре больше баллов, чем по русскому языку. Ответ на этот вопрос (только число) запишите в ячейку `C2` таблицы.

Полученную таблицу необходимо сохранить.

Пример 10.31. Результаты сдачи Государственной итоговой аттестации по информатике учащимися 9-х классов некоторого города были занесены в электронную таблицу. Приведены первые строки получившейся таблицы:

	A	B	C	D
1	оценка	мин. балл	макс. балл	
2	2	0	25	
3	3	26	48	
4	4	49	67	
5	5	68	100	
6				
7			Информатика	
8	Рощина	Татьяна	35	
9	Кузьмина	Елена	48	
10	Гдлян	Анаида	66	
11	Коршунов	Сергей	88	

В столбце **A** электронной таблицы записана фамилия учащегося, в столбце **B** — имя учащегося, в столбце **C** — баллы учащегося по ин-

форматике. Баллы могут принимать значения от 0 до 100. Всего в электронную таблицу были занесены результаты 234 учащихся.

В ячейках A1:C5 записаны баллы, соответствующие школьным оценкам 2, 3, 4 и 5. Например, если учащийся набрал 48 баллов по информатике, он получает школьную оценку 3, а если набрал 49 баллов, получает оценку 4.

Выполните задание.

Откройте файл с данной электронной таблицей. На основании данных, содержащихся в этой таблице, ответьте на два вопроса:

1. Сколько учащихся получили оценки 2, 3, 4 и 5 по информатике? Ответ на этот вопрос (только числа) запишите в ячейки D2, D3, D4 и D5 таблицы.
2. Для группы учащихся, которые получили только положительный результат, посчитайте средний балл по информатике. Ответ на этот вопрос (только число) запишите в ячейку D6 таблицы.

Полученную таблицу необходимо сохранить.

ГЛАВА 11

ИНФОРМАЦИОННЫЕ КОМПЬЮТЕРНЫЕ СЕТИ

Основные понятия

При соединении двух и более компьютеров между собой образуется **компьютерная сеть**. Для создания компьютерной сети требуется специальная аппаратура — **сетевое оборудование** — и специальные программы — **сетевое программное обеспечение**.

Узел — это устройство, соединённое с другими устройствами компьютерной сети. Узлами могут быть компьютеры, специальные сетевые устройства и т. п.

Для того чтобы устройства в сети могли «общаться», необходимо установить правила, по которым будет передаваться, получаться, кодироваться и декодироваться информация. Своды таких правил называют **протоколами передачи данных**.

Скорость передачи данных по каналу связи измеряется количеством единиц информации, передаваемых за единицу времени. Единицы измерения — бит в секунду, килобит в секунду и т. п.

Объем информации, передаваемой по сети за определённый промежуток времени, называют **трафиком**.

Для того чтобы информация, передаваемая по сети, достигала нужных адресатов, каждый сетевой узел получает уникальный **адрес — IP-адрес**.

IP-адрес представляет собой 32-битное двоичное число, представленное четырьмя группами по 8 бит (**октет**), разделённых пробелами, например:

11000000 10101000 01100100 01000101.

Благодаря этому можно получить более 4 млрд различных адресов.

Для удобства восприятия и сокращения записи принято также использовать десятично-точечную форму представления IP-адресов. Каждый октет заменяют соответствующим ему десятичным числом из диапазона [0; 255], разделяя числа точками. В десятично-точечном представлении IP-адрес, приведённый выше, запишется так:

192.168.100.69.

Компьютеры, объединённые в сеть, могут находиться как в одной комнате или здании, так и в разных городах и даже на разных континентах. В зависимости от территориальной распространённости сети делятся на **локальные, региональные** и **глобальные**. В *локальные сети* входят узлы, расположенные друг от друга не более чем на несколько километров, например локальная сеть школы, вуза, компьютерного клуба и т. д. *Региональные сети* объединяют компьютеры в пределах одного региона (города, области, страны). Выделяют корпоративные сети, где важно защитить информацию от несанкционированного доступа, например сеть банка или министерства обороны. Локальные, региональные, корпоративные сети объединяются в *глобальную сеть* Интернет.

Локальные сети

Локальную сеть также называют **локальной вычислительной сетью (ЛВС)** или **Local Area Network (LAN)**. Локальные сети организуются там, где пользователям нескольких компьютеров требуется иметь общие данные, решать совместные задачи, пользоваться некоторыми устройствами, например принтерами.

Каждый компьютер или другой узел локальной сети должен иметь сетевую плату, основной функцией которой является передача и приём информации из сети. В проводных ЛВС соединение компьютеров между собой производится с помощью кабелей. В беспроводных ЛВС ис-

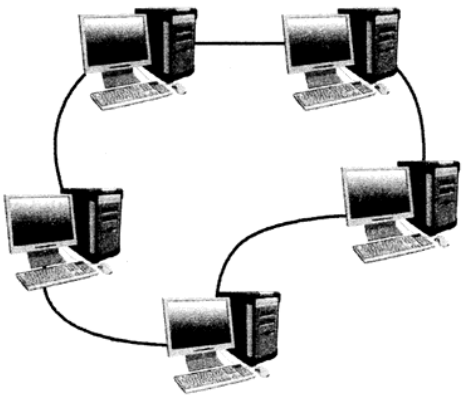


Рис. 50.

пользуется технология Wi-Fi. Центральное сетевое устройство — точка доступа — подключается к сети, а на каждом компьютере должна быть установлена беспроводная сетевая плата.

В зависимости от того, каким образом соединены компьютеры в локальную сеть, можно выделить типы схем соединений узлов в сети. Геометрическая схема соединений компьютеров называется **топологией локальной сети**.

В сети с **кольцевой топологией** (см. рис. 50) компьютеры соединяются каналами связи в кольцо: выход одного компьютера соединяется с входом другого. Сообщение в сети такого вида передается от компьютера к компьютеру по кольцу до тех пор, пока не достигнет адресата. Кольцевая топология сети является эффективной для отправки запросов на все машины сети. Отметим, что повреждение канала связи или одного из компьютеров приводит к неработоспособности всей сети.

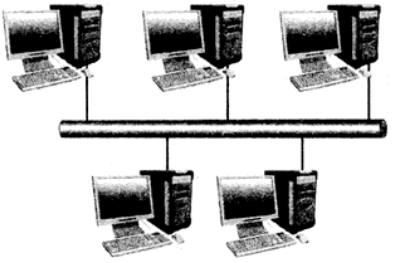


Рис. 51.

Шинная топология локальной сети (см. рис. 51) основана на подключении компьютеров к каналу связи (**шине**) при помощи специальных разъёмов. Сообщение, отправленное от компьютера другому компьютеру, получают все компьютеры сети, но только машина-адресат может его прочитать. Отказ одного из узлов при такой конфигурации сети не повлияет на её работоспособность. Длина шины ограничивает число возможных узлов. Повреждение шины выведет сеть из строя.

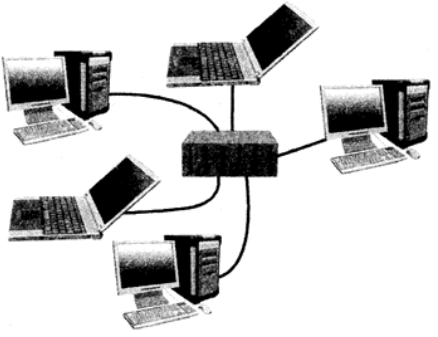


Рис. 52.

Если каждый узел сети подключён к специальному устройству — **концентратору (hub)**, обеспечивающему параллельное подключение узлов, топология локальной сети называется **«звезда»** (см. рис. 52). За пересылкой сообщений следит концентратор, он определяет адресата и направляет ему запрос. При такой организации сети легко может быть добавлен новый узел, сеть остаётся работоспособной при неисправности отдельных компьютеров или каналов связи. На работоспособность сети влияет только отказ концентратора.

На практике часто используются комбинации приведённых топологий и их модификации.

По способу организации взаимодействия компьютеров локальные сети делят на одноранговые и с выделенным сервером (иерархические сети).

В **одноранговой** сети все компьютеры равноправны. Любой пользователь сети может получить доступ к данным, хранящимся на любом компьютере. Основной недостаток таких сетей — слабая защищённость информации.

В иерархической сети, или **сети с выделенным сервером**, выделяются один или несколько серверов — компьютеров, управляющих обменом данных и распределением ресурсов сети. Сервер — это постоянное хранилище разделяемых ресурсов. Любой компьютер, имеющий доступ к услугам сервера, называют **клиентом** сети или **рабочей станцией**. Сам сервер может быть клиентом сервера более высокого уровня иерархии. Серверы обычно представляют собой высокопроизводительные компьютеры, возможно, с несколькими параллельно работающими процессорами, винчестерами большой ёмкости и высокоскоростной сетевой картой.

Локальные сети обычно дистанционно ограничены линиями кабеля или мощностью ретранслятора. Например, районная или городская локальная сеть может успешно функционировать. Однако объединить в локальную сеть компьютеры из разных городов проблематично.

Существует возможность объединять локальные сети, удалённые друг от друга. В этом случае говорят о **распределённых сетях** или **Wide Area Network (WAN)**.

Интернет

Интернет называют сетью сетей. Началом Интернета принято считать 1969 год, когда министерство обороны США приступило к разработке распределённой сети ARPAnet, которая должна была связать между собой множество дистанционно удалённых друг от друга компьютеров и оставаться в рабочем состоянии даже при частичном повреждении узлов сети.

Принципы и правила, заложенные в ARPAnet, оказались достаточно удачными для общего использования, что позволило многим организациям начать создание или усовершенствование собственных сетей с использованием этих принципов. Со временем эти сети стали объединяться между собой в мировую сеть, которая и получила название **Интернет**.

Слово Интернет происходит от английского Internet, сокращённого выражения interconnected networks (связанные сети). В широком смысле Интернет — это глобальное информационное пространство, хранящее огромное количество информации на миллионах компьютерах, которые обмениваются данными.

Пользователи Интернета подключаются к сети с помощью поставщиков Интернет-услуг — Интернет-провайдеров. Провайдеры имеют в собственности или арендуют высокоскоростные каналы связи (оптоволоконные, спутниковые, радиоканалы), поэтому могут предоставлять доступ к сети одновременно тысячам пользователей.

Пользователи подключаются к серверам провайдеров по коммутируемым телефонным каналам, выделенным каналам связи (оптоволоконным линиям), используется также беспроводной доступ. Провайдер может оказывать и другие услуги: выделить дисковое пространство для хранения и обеспечения работы сайтов (хостинг), поддерживать работу почтовых ящиков или виртуального почтового сервера и т. д.

Для подключения к Интернету по коммутируемым телефонным каналам к компьютеру пользователя должен быть подключен *модем*. **Модем** — это электронное устройство, преобразующее цифровой сигнал компьютера в аналоговый сигнал телефонных линий. Такие преобразования называются **МОдуляцией** и **ДЕМодуляцией** сигнала, отсюда и название устройства. Современные модемы используют **ADSL**-технологии (**A**symmetric **D**igital **S**ubscriber **L**ine — асимметричная цифровая абонентская линия), позволяющие использовать обычную телефонную линию одновременно для телефона и для скоростного Интернета.

На основе принципов организации Интернета были разработаны специальные **службы** (**интернет-службы**), без которых сложно представить современную жизнь.

Служба WWW

Одной из наиболее популярных служб современного Интернета является **World Wide Web (WWW)**, или сокращенно **Web (Веб)**.

С WWW связан протокол передачи гипертекста **HTTP** (**H**yper**t**ext **T**ransfer **P**rotocol). Служба WWW позволяет одним пользователям размещать на веб-серверах Интернета **сайты**, а другим — просматривать **веб-страницы** сайтов при помощи браузеров. **Сайт** — это совокупность нескольких веб-страниц, объединённых общей тематикой и дизайном, размещенных на сервере в Интернете. Веб-страницы представляют собой файлы, хранящие гипертекстовые документы.

Гипертекст — специальным образом размеченный текст, при просмотре которого можно осуществлять переходы между частями текстового документа, а также между разными документами. Элементы гипертекста — **гиперссылки** — позволяют осуществлять переходы между веб-страницами, хранящимися на одном компьютере, а также между веб-страницами, находящимися на любом компьютере, подключённом к Интернету. Традиционно гиперссылки в тексте выделяют синим

цветом и подчёркиванием. При наведении курсора мыши на гиперссылку указатель на экране меняет вид. При активации гиперссылки, например щелчком мыши, происходит переход к другому документу (месту в документе), адресуемому ссылкой. Документы могут находиться на разных узлах сети. Современные средства дизайна и настройки пользовательских компьютеров могут влиять на представление гиперссылок.

Гипертекст, понятный службе WWW, создаётся при помощи специального языка гипертекстовой разметки **HTML** (**H**yper**t**ext **M**arkup **L**anguage) и сохраняется в файлах с расширением `.html` или `.htm`. Создать гипертекстовую страницу можно при помощи текстового редактора или специальных программ, например Microsoft FrontPage, Macromedia Dreamweaver. Интерфейсы этих программ очень похожи на интерфейсы текстовых процессоров, они позволяют создавать веб-страницы пользователям, не знающим языка гипертекстовой разметки. Содержимое страницы автоматически снабжается необходимой разметкой в зависимости от внешнего вида, определённого пользователем.

Пример. Простая html-страница

Создайте новый текстовый файл и сохраните его с расширением .html. Поместите в файл следующий текст:

```
<html>
<body>
<b>Моя страничка</b>
</body>
</html>
```

В угловых скобках <> указываются имена команд разметки текста — **тегов**. Теги бывают **парными**, т. е. состоящими из открывающего и закрывающего тегов, например `<html>` и `</html>`, и **непарными**, т. е. состоящими из одного тега, например `<br>`. Парные, или **контейнерные**, теги наделяют набором свойств блок текста, заключённый в них. Например, текст, заключённый между тегами `<b></b>`, будет отображаться жирным шрифтом. Непарные теги, как правило, влияют только на следующую за ними строку. Например, тег `<br>` принудительно разрывает строку текста.

Более полную информацию о гипертекстовой разметке вы можете найти в специальных учебных пособиях и справочниках.

Программы просмотра веб-страниц

Поскольку страницы сайтов (веб-страницы) представляют собой специальным образом размеченные тексты, то для их корректного отображения необходимо специальное программное обеспечение.

	Microsoft Internet Explorer

	Google Chrome
	Mozilla FireFox
	Opera
	Apple Safari

Программы, при помощи которых пользователи Интернета могут подсоединяться к службе WWW и просматривать веб-страницы, называются **веб-браузерами** или просто **браузерами**.

Наиболее популярными браузерами являются **Microsoft Internet Explorer, Mozilla FireFox, Opera, Google Chrome, Apple Safari**.

Для того чтобы открыть веб-страницу в окне браузера, в специальную строку (командная строка) вводится адрес этой страницы.

Адресация в Интернете

Браузер открывает страницы сайтов, обращаясь к ним по уникальным IP-адресам. Пользователям Интернета сложно запомнить числовой IP-адрес. Для упрощения взаимодействия пользователей с Интернет-службами используется **система доменных имён, или** `DNS` (**D**omain **N**ame **S**ystem), позволяющая обращаться к узлам сети не по числовым адресам, а по символическим **доменным именам**.

Доменные имена ставятся в соответствие IP-адресам. Одному IP-адресу может соответствовать сразу несколько доменных имён и наоборот. Пользовательские запросы к доменам обрабатываются специальными узлами в сети — **серверами доменных имён**.

Примеры доменных имён: `www.mydomain.myschooldomain.ru`, `www.ege.edu.ru`. Каждое слово в доменном имени называется **доменом.** При записи имени домены отделяется точкой.

Домены делятся по уровням (`ru` — домен первого уровня, `edu` — домен второго уровня и т. д.). Домены первого уровня бывают двух

типов — географические и административные. Каждой стране мира выделен свой географический домен, обозначаемый двухбуквенным кодом, в котором организации и граждане имеют право зарегистрировать домен второго уровня. Административные домены обозначаются тремя и более буквами и могут быть зарегистрированы во многих странах.

Некоторые имена доменов первого уровня

Административные		Географические	
edu	образовательные проекты и вузы США	ru	Россия
org	некоммерческие организации	uk	Великобритания
com	коммерческие организации	it	Италия

Система именования DNS представляет собой иерархическую древовидную структуру, которую называют **пространством имён DNS**, где есть один корень, у которого может быть любое число поддоменов. **Поддомен** — домен, являющийся частью домена более высокого уровня. У отдельных поддоменов в свою очередь могут быть дочерние поддомены. Например, в имени сайта Министерства образования Российской Федерации `www.mon.gov.ru` поддомен первого уровня `.ru` имеет собственные поддомен `gov.ru`, у которого есть поддомен `mon.gov.ru`.

Корневой домен (root domain) — домен самого верхнего уровня в системе доменных имён. Его иногда называют доменом нулевого уровня, он обозначается пустым (т. е. не содержащим никаких символов) именем. В конце доменного имени может присутствовать точка, которая отделяет пустое имя, соответствующее корневому домену. Если эта точка есть (например, `www.example.com.`), то доменное имя считается полным (абсолютным). Если точки в конце имени нет (`www.example` или `www.example.com`), то имя считается относительным.

При обращении к сайту по доменному имени, например `www.mon.gov.ru`, первым делом отправляется запрос к корневому домену. Этот запрос содержит команду вернуть IP-адрес компьютера, на котором расположена информация о домене `ru`. После получения ответа по запрошенному IP-адресу происходит аналогичное обращение с запросом определить адрес домена `gov` внутри домена `ru` внутри корневого домена. В конце концов у предпоследнего компьютера запрашивается IP-адрес поддомена www в домене `mon.gov.ru`.

Перед доменным именем указывают имя протокола, который используется для общения с узлом. Имя протокола отделяется от доменного имени последовательностью символов `://`. Например, запись

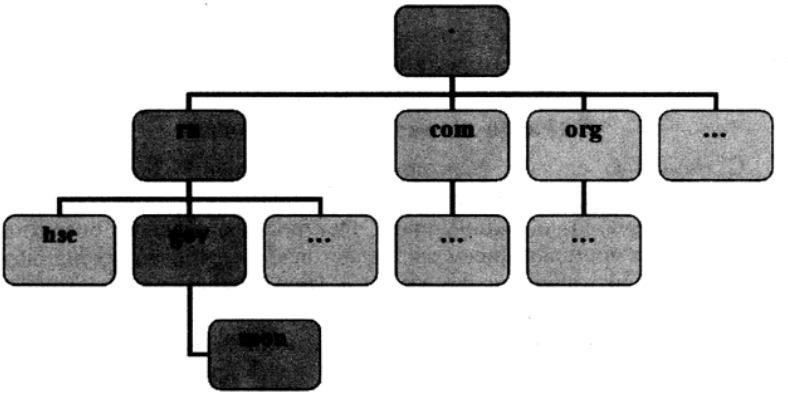


Рис. 53. Иерархическая структура доменных имён.

`http://www.mon.gov.ru` означает, что общение с узлом происходит с использованием протокола HTTP *.

Для того чтобы указать **адрес конкретного файла (веб-страницы)** в Интернете, записывается путь к нему и способ доступа (протокол). В качестве корневого каталога указывается доменный адрес машины, после чего путь к файлу на этой машине. Адрес, записанный в виде:

`<протокол>://<имя_узла>/<путь>/<имя_файла>.<расширение>`

называется **уникальным указателем на ресурс** или **URL** (**U**niform **R**esource **L**ocator). Например, `http://www.somehost.com/index.html` указывает на файл `index.html`, хранящийся на сервере `somehost.com`, доступ к файлу осуществляется по протоколу `http`.

Служба FTP

Для пересылки файлов между компьютерами применяется **интернет-служба FTP**, получившая своё название по протоколу, который описывает правила передачи файлов с компьютера на компьютер. Этот протокол называется **протоколом передачи файлов** или **FTP** (**F**ile **T**ransfer **P**rotocol).

Пользователи могут передавать файлы между компьютерами, используя как службу FTP, так и специальные программы, позволяющие обмениваться файлами через сеть `ftp-клиенты`. Такие программы позволяют просматривать файлы, находящиеся на других сетевых компьютерах (например, файловых серверах в Интернете), копировать файлы на свой компьютер и т. д.

---

* В настоящее время в браузерах можно не указывать имя протокола http:// и префикс службы www, так как современные браузеры самостоятельно подставляют их в адрес запрашиваемого ресурса.

Ftp-клиентами являются программы: AceFTP Pro, CoreFTP и др. Некоторые функции этих программ доступны и современным файловым менеджерам, среди которых, например, Total Commander.

Служба электронной почты

С помощью службы электронной почты любой подключённый к ней пользователь Интернета может отправлять и получать электронные письма. От обычной почты электронную отличает оперативность — время доставки писем составляет всего несколько секунд, возможность вложить в письмо файлы и т. д. Полученные адресатом послания хранятся в электронном почтовом ящике.

Со службой электронной почты связаны **протокол SMTP** (**S**imple **M**ail **T**ransfer **P**rotocol), который регламентирует правила передачи писем от адресата к адресату, и **протокол POP** (**P**ost **O**ffice **P**rotocol), организующий доступ пользователя к почтовому ящику.

Для того чтобы отправить или получить электронное письмо, требуется специальный почтовый адрес, который можно зарегистрировать на одном из бесплатных почтовых сервисов Интернета — **почтовом сервере**. Например, компания Google предоставляет почтовый сервис `Google-mail`, доступный по адресу `http://www.gmail.com`; компания `Mail.ru` — почтовый сервис `Mail.ru`, доступный по адресу `http://www.mail.ru`. Многие организации обзаводятся собственными почтовыми серверами, и адреса на них могут получить только их сотрудники.

Адрес электронной почты имеет следующий вид: `postbox@hostname`. Слева от @ записывается уникальный идентификатор абонента почтовой службы, который следует придумать самостоятельно, если ящик регистрируется на общедоступном почтовом сервере, или получить по правилам, принятым в организации, если адрес предоставляется её сотруднику. Справа от @ — название почтового сервера, предоставляющего услуги. Например, на бесплатном почтовом сервере `Mail.ru` может быть зарегистрирован следующий почтовый ящик: `marina_ivanova@mail.ru`.

Для доступа к электронной почте используются специальные программы — **почтовые клиенты** (мейлеры), например Microsoft Outlook. Многие почтовые клиенты реализованы как веб-приложения. Для сокращения времени на подготовку писем в почтовые программы встроены такие возможности, как автоматическая вставка приветствия, подписи и обратного адреса, использование адресной книги, операции экспорта/импорта. Уменьшение затрат времени при приёме и обработке сообщений достигается за счёт таких средств, как организация дерева папок, поиска сообщений в папках по определённым критериям, задание правил сортировки входящих сообщений, возможность чтения заголовков писем непосредственно на почтовом сервере до фактического их приёма клиентской почтовой программой.

Рекомендации по работе с электронной почтой

Электронная почта является одним из самых распространённых и давно используемых сервисов сети Интернет. Со временем сложились определённые правила сетевого этикета, направленные на уменьшение затрат времени на обработку почты, поддержание уважительных, деловых и дружеских взаимоотношений с помощью электронной почтовой связи.

В электронном сообщении заполняются поля:

- адрес получателя и адрес получателя копии письма;
- тема письма;
- тело письма и подпись;
- вложение (attachment).

Тема письма. Поле «Тема» включает обычно несколько слов, лаконично отражающих его суть. Тема письма видна получателю до открытия письма наряду с датой и отправителем. Рекомендуется всегда заполнять это поле. Отсутствие темы, как и отсутствие сопроводительного текста при пересылке вложения, является признаком плохого тона.

В случае ответного послания почтовая программа автоматически вставляет в это поле «Re» (от англ. reply — ответ) и тему исходного письма. Если в ответном письме меняется тема, измените и поле «Тема».

Тело письма. Пишите кратко. Желательно, чтобы длина письма не превышала страницы (одного экрана). Более длинные сообщения готовят в текстовых редакторах и пересылают в виде вложений. Если рассчитываете получить квалифицированный ответ на своё письмо, включите в него полную информацию по теме.

Пишите аккуратно. Не пишите весь текст заглавными буквами — его тяжело читать. Разбивайте письмо на логические абзацы, избегайте чрезмерно длинных предложений.

Будьте вежливы. Не забывайте о волшебных словах «пожалуйста», «спасибо» и т. п.

Пишите грамотно. Перед тем как послать письмо, прочтите его внимательно ещё раз.

Помните, что не существует надежной почтовой системы. Пишите в электронном письме только то, что написали бы в обычной почтовой открытке.

Подписи. Подпись в конце сообщения обычно состоит из 4—7 строк, содержит более подробные сведения об отправителе и информацию о контактах (номер телефона, ICQ и т. п.). Подпись может включаться в сообщение автоматически почтовым клиентом.

Вложение (attachment). Рекомендуется заархивировать (упаковать) вложение перед отправкой одним из распространённых архиваторов (ZIP, RAR, ARJ). Не следует посылать самораспаковывающийся архив (с расширением `exe`), поскольку он рассматривается как потенциально опасная программа-вирус и не передаётся получателю.

Даже в случае посылки одного только файла-вложения в тело письма в желательно включать несколько сопроводительных приветственных слов.

Ответы. На каждое сообщение желательно дать ответ в течение одного-двух дней. При ответе включайте в письмо отрывки письма, на которое отвечаете, и адресат легче поймёт, о чем идёт речь. Кратко и по существу отвечайте на вопрос. Держитесь как можно ближе к теме. Если хотите сменить тему, лучше послать отдельное письмо с указанием новой темы.

Если Вы отсутствуете и не можете просматривать почту, можно настроить почтовую программу на автоматическую генерацию ответа, например, «Я в отъезде до 25 августа, отвечу на письмо после возвращения».

По электронной почте могут приходить письма, заражённые вирусами. Встречаются случаи заражения макровирусами документов в формате текстового процессора Word. Не следует запускать неизвестную программу или просматривать файл, присланный по почте, без предварительной проверки.

Служба поиска. Поисковые системы

С каждой минутой в Интернете появляется всё больше страниц и сайтов, усложняются связи между ними. Поиск необходимой информации мог бы занять у пользователя всю жизнь, если бы он перемещался от страницы к странице только по уточняющим гиперссылкам. Для облегчения поиска информации в Интернете организованы специальные поисковые системы, позволяющие пользователям находить за считанные секунды необходимую информацию.

Поисковая система — это комплекс программ и мощных компьютеров, автоматически просматривающих ресурсы Интернета и индексирующих их содержание. Поисковые системы могут отличаться по эффективности поиска и другим возможностям. Например, одни поисковые системы находят информацию только в виде веб-страниц, другие могут просматривать и группы новостей, и файловые серверы. Результатом поиска являются гиперссылки на документы, содержащие требуемую информацию.

Наиболее известными из них являются Google, Yahoo!, Aport, Яндекс.

Поисковые системы обычно состоят из трёх компонентов:

- поисковый робот, который перемещается по сети и собирает информацию;
- база данных, которая содержит постоянно обновляемую информацию о местонахождении веб-страниц и файлов на сотнях миллионов серверов Интернета, собираемую роботом;
- поисковый механизм, который используется как интерфейс для взаимодействия с базой данных.

Поисковые роботы — это специальные программы, которые занимаются поиском страниц в сети, извлекают гипертекстовые ссылки на этих страницах и автоматически индексируют информацию, которую они находят для построения базы данных.

Поисковые системы осуществляют поиск информации по *ключевым словам*. **Ключевым словом** (Keyword) документа называется слово

или словосочетание, которое отражает содержание данного документа. Как правило, для успешного запроса надо ввести несколько слов для уточнения поиска. Например, для поиска объявлений о продаже котят в качестве ключевых слов можно использовать два слова: «продажа», «котята». Вы можете сузить запрос, указав в качестве ключевого слова породу кошки.

Службы поиска позволяют связывать ключевые слова логическими операциями И, ИЛИ, искать цитаты, исключать некоторую информацию из поискового запроса и т. д. Все эти возможности определяются языком запросов конкретной поисковой системы. Например, большинство поисковых систем расценивает заключённую в кавычки фразу как цитату. По запросу: «Унылая пора! Очей очарованье!» поисковая система Яндекс найдёт только страницы, содержащие в точности такую строку.

Поисковые службы, как правило, предоставляют возможность расширенного поиска, а также помощь по использованию языка запросов

Пример 11.1. Задание с кратким ответом

Доступ к файлу `htm.txt`, находящемуся на сервере `com.ru`, осуществляется по протоколу `http`. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла в сети Интернет.

А	/
Б	com
В	.txt
Г	://

Д	.ru
Е	htm
Ж	http

Решение. Запишем путь к файлу в соответствии с общим видом:

`название_протокола://имя_сервера/имя_файла.расширение.`

В соответствии с условиями задания название протокола — `http`; имя сервера — `com.ru`; имя файла и расширение — `htm.txt`. Таким образом, получим адрес: `http://com.ru/htm.txt`.

http	://	com	.ru	/	htm	.txt
Ж	Г	Б	Д	А	Е	В

Запишем ответ буквами в соответствии с таблицей, получаем ЖГБДАЕВ.

Ответ: ЖГБДАЕВ.

Пример 11.2. Задание с кратким ответом

В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдёт поисковый сервер по каждому запросу.

Для обозначения логической операции ИЛИ в запросе используется символ |, а для логической операции И — &:

А	Литература & Экзамен
Б	Литература | Экзамен | Билеты
В	Литература | Билеты
Г	Литература & Экзамен & Билеты

Решение. Для решения нам понадобится вспомнить законы логики. Результатом логической операции ИЛИ будет ИСТИНА, если хотя бы одно из перечисленных слов встретится на странице. Результатом же логической операции И будет ИСТИНА только в том случае, если каждое слово обязательно присутствует на странице. Таким образом, максимальное количество страниц будет найдено по запросу, в котором больше слов и они связаны логической операцией ИЛИ. Минимальное количество страниц будет найдено по запросу, в котором больше слов и они соединяются логической операцией И. Следовательно, в порядке возрастания запросы расположатся следующим образом: ГАВБ.

Ответ: ГАВБ.

Задания для самостоятельного решения

Задания с кратким ответом

Пример 11.3. Доступ к файлу `jpg.bmp`, находящемуся на сервере `info.ru`, осуществляется по протоколу `http`. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла в сети Интернет.

А	`/info`
Б	`.bmp`
В	`http`
Г	`.ru`

Д	`/`
Е	`:/`
Ж	`jpg`

Пример 11.4. Доступ к файлу `txt.net`, находящемуся на сервере `edu.com`, осуществляется по протоколу `http`. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла в сети Интернет.

А	`http`
Б	`.com`
В	`/`
Г	`.net`

Д	`txt`
Е	`://`
Ж	`edu`

Пример 11.5. Доступ к файлу doc.doc, находящемуся на сервере edu.info, осуществляется по протоколу ftp. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла в сети Интернет.

А	.doc
Б	://
В	ftp
Г	/

Д	edu.
Е	doc
Ж	info

Пример 11.6. Доступ к файлу ip.htm, находящемуся на сервере tcp.com, осуществляется по протоколу ftp. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла в сети Интернет.

А	com
Б	ftp
В	.htm
Г	ip

Д	://
Е	tcp.
Ж	/

Пример 11.7. Доступ к файлу domain.txt, находящемуся на сервере htm.edu, осуществляется по протоколу http. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла в сети Интернет.

А	domain
Б	.edu
В	://
Г	.txt

Д	htm
Е	http
Ж	/

Пример 11.8. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания количества страниц, которые найдёт поисковый сервер по каждому запросу.

Для обозначения логической операции ИЛИ в запросе используется символ |, а для логической операции И — &.

А	Чемпионат | Футбол | Англия
Б	Англия | Футбол
В	Футбол & Англия & Чемпионат
Г	Футбол & Чемпионат

Пример 11.9. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке возрастания

количества страниц, которые найдёт поисковый сервер по каждому запросу.

Для обозначения логической операции ИЛИ в запросе используется символ |, а для логической операции И — &.

А	Рыба | Тихий океан
Б	Тихий океан & Миграция & Рыба
В	Рыба | Миграция | Тихий океан
Г	Миграция & Рыба

Пример 11.10. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке убывания количества страниц, которые найдёт поисковый сервер по каждому запросу.

Для обозначения логической операции ИЛИ в запросе используется символ |, а для логической операции И — &.

А	Метро | Москва
Б	Проездной & Метро
В	Москва & Проездной & Метро
Г	Москва | Проездной | Метро

Пример 11.11. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке убывания количества страниц, которые найдёт поисковый сервер по каждому запросу.

Для обозначения логической операции ИЛИ в запросе используется символ |, а для логической операции И — &.

А	Информатика | 2010
Б	Олимпиада | Информатика | 2010
В	Олимпиада & Информатика & 2010
Г	Олимпиада & 2010

Пример 11.12. В таблице приведены запросы к поисковому серверу. Расположите обозначения запросов в порядке убывания количества страниц, которые найдёт поисковый сервер по каждому запросу.

Для обозначения логической операции ИЛИ в запросе используется символ |, а для логической операции И — &.

А	Овощи | Помидор | Рассада
Б	Помидор & Рассада & Овощи
В	Помидор | Овощи
Г	Помидор & Рассада

ОТВЕТЫ К ЗАДАНИЯМ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

Глава 2

2.18. 1; **2.19.** 3; **2.20.** 3; **2.21.** 3; **2.22.** 2; **2.23.** 2; **2.24.** 3; **2.25.** 3; **2.26.** 3; **2.27.** 4; **2.28.** 2; **2.29.** 3; **2.30.** 3; **2.31.** 4; **2.32.** 9; **2.33.** 32; **2.34.** 64; **2.35.** 216; **2.36.** 4; **2.37.** 8; **2.38.** 80; **2.39.** 2; **2.40.** 4; **2.41.** 256; **2.42.** 229; **2.43.** 257; **2.44.** 13; **2.45.** 32; **2.46.** 137; **2.47.** 11111111; **2.48.** 11000100; **2.49.** 1001001; **2.50.** 1000000; **2.51.** 10000011.

Глава 3

3.5. 3; **3.6.** 1; **3.7.** 4; **3.8.** 2; **3.9.** 3; **3.10.** 2; **3.11.** 3; **3.12.** 2; **3.13.** 2; **3.14.** 1; **3.15.** 3; **3.16.** 4.

Глава 4

4.5. 2; **4.6.** 2; **4.7.** 1; **4.8.** 2; **4.9.** 3; **4.10.** 3; **4.11.** 4; **4.12.** 3; **4.13.** 1; **4.14.** 3; **4.15.** 1; **4.16.** 3; **4.17.** 1; **4.18.** 2; **4.19.** 15; **4.20.** 23; **4.21.** 32; **4.22.** 3750; **4.23.** 168; **4.24.** 32.

Глава 5

5.5. 3; **5.6.** 1; **5.7.** 1; **5.8.** 3; **5.9.** 4; **5.10.** 2; **5.11.** 1; **5.12.** 3; **5.13.** 1; **5.14.** 3; **5.15.** 4; **5.16.** 3; **5.17.** 2; **5.18.** 2; **5.19.** 4.

Глава 6

6.9. 4; **6.10.** 4; **6.11.** 2; **6.12.** 3; **6.13.** 2; **6.14.** 24; **6.15.** 35; **6.16.** 48; **6.17.** 99; **6.18.** 46; **6.19.** НЛОМ; **6.20.** НИОНИЛМНИ; **6.21.** НПКОРО; **6.22.** АСАТ; **6.23.** 2; **6.24.** 3; **6.25.** 5; **6.26.** 3; **6.27.** 2; **6.28.** 9.

Глава 7

7.6. 4; **7.7.** 3; **7.8.** 1; **7.9.** 2; **7.10.** 3; **7.11.** 21121; **7.12.** 22112; **7.13.** 22111; **7.14.** 2121; **7.15.** 21221; **7.16.** 12121; **7.17.** 231; **7.18.** 13233.

7.19.

```
нц пока не слева свободно
   вверх
кц
вниз
нц пока не слева свободно
   закрасить
   вниз
кц
```

7.20.

```
нц пока не сверху свободно
   влево
кц
вправо
нц пока не сверху свободно
   если не сверху свободно то
   закрасить
   вправо
   все
   если не сверху свободно то
   закрасить
   вправо
   все
   вправо
кц
```

7.21

```
нц пока не слева свободно
   вверх
кц
вниз
нц пока не слева свободно
   закрасить
   вверх
   вверх
кц
```

Глава 8

8.5. 3; **8.6.** 1; **8.7.** 2; **8.8.** 2; **8.9.** 3.

Глава 9

9.4. 2; **9.5.** 1; **9.6.** 4; **9.7.** 3; **9.8.** 4; **9.9.** 3; **9.10.** 2; **9.11.** 1; **9.12.** 2; **9.13.** 4; **9.14.** 2.

Глава 10

10.17. 3; **10.18.** 2; **10.19.** 1; **10.20.** 3; **10.21.** 4; **10.22.** 2; **10.23.** 19; **10.24.** 2; **10.25.** 20; **10.26.** 2; **10.27.** 10; **10.28.** 2.

Глава 11

11.3. ВАГДЖБ; **11.4.** АЕЖБВДГ; **11.5.** ВБДЖГЕА; **11.6.** БДЕАЖГВ; **11.7.** ЕВДБЖАГ; **11.8.** ВГБА; **11.9.** БГАВ; **11.10.** ГАБВ; **11.11.** БАГВ; **11.12.** АВГБ.

Список литературы

Андреева Е., Фалина И. Системы счисления и компьютерная арифметика. 2-е изд. М.: Лаборатория Базовых Знаний, 2000. 248 с.

Гейн А. Г. Информатика и информационные технологии. 8 класс: Учеб. для общеобраз. учрежд. / А. Г. Гейн, А. И. Сенокосов, Н. А. Юнерман. 2-е изд. М.: Просвещение, 2010. 175 с.

Гейн А. Г. Информатика и информационные технологии. 9 класс : Учеб. для общеобраз. учрежд. / А. Г. Гейн, А. И. Сенокосов. 2-е изд. М.: Просвещение, 2009. 336 с.

Гейн А. Г. Информатика и информационные технологии: Задачник-практикум: Учеб. пособие для уч-ся 8—9 кл. общеобраз. учрежд. / А. Г. Гейн, Н. А. Юнерман. М.: Просвещение, 2008. 127 с.

Кузнецов А. А. и др. Информатика: Учеб. для 8 кл. общеобраз. учрежд. / А. А. Кузнецов, С. А. Бешенков, Е. А. Ракитина; Под ред. С. А. Христочевского; Рос. акад. наук, Рос. акад. образования, изд-во «Просвещение». М.: Просвещение, 2008. 175 с.

Кушниренко А. Г. и др. Информатика. 7—9 кл.: Учеб. для общеобраз. учеб. заведений / А. Г. Кушниренко, Г. В. Лебедев, Я. Н. Зайдельман. 3-е изд. М.: Дрофа, 2002. 336 с.

Информатика и ИКТ. Учебник. 8—9 класс / Под ред. проф. Н. В. Макаровой. СПб.: Питер, 2009. 416 с.

Семакин И. Г. Информатика и информационно-коммуникационные технологии. Базовый курс: Учебник для 8 класса / И. Г. Семакин, Л. А. Залогова, С. В. Русаков, Л. В. Шестакова. 3-е изд. М.: БИНОМ. Лаборатория знаний, 2010. 176 с.

Семакин И. Г. Информатика и информационно-коммуникационные технологии. Базовый курс: Учебник для 9 класса / И. Г. Семакин, Л. А. Залогова, С. В. Русаков, Л. В. Шестакова. 3-е изд. М.: БИНОМ. Лаборатория знаний, 2009. 341 с.

Информатика. Задачник-практикум: В 2 т. / Л. А. Залогова, М. А. Плаксин, С. В. Русаков и др.; Под ред. И. Г. Семакина, Е. К. Хеннера. Том 1. 4-е изд. М.: БИНОМ. Лаборатория знаний, 2009. 309 с.

Информатика. Задачник-практикум: В 2 т. / Л. А. Залогова, М. А. Плаксин, С. В. Русаков и др.; Под ред. И. Г. Семакина, Е. К. Хеннера. Том 2. 4-е изд. М.: БИНОМ. Лаборатория знаний, 2009. 294 с.

Угринович Н. Д. Информатика и ИКТ: Учебник для 8 класса. 2-е изд. М.: БИНОМ. Лаборатория знаний, 2009. 178 с.

Угринович Н. Д. Информатика и ИКТ: Учебник для 9 класса / Н. Д. Угринович. 2-е изд. М.: БИНОМ. Лаборатория знаний, 2009. 295 с.

Шауцукова Л. З. Информатика: Учеб. пособие для 10—11 кл. общеобраз. учрежд. 3-е (4-е) изд. М.: Просвещение, 2004. 416 с.

СОДЕРЖАНИЕ

Справочное издание

Авдошин Сергей Михайлович,
Ахметсафина Римма Закиевна,
Максименкова Ольга Вениаминовна и др.

ИНФОРМАТИКА

ГИА

Учебно-справочные материалы
для 9 класса

Редактор *М. А. Рачинская*
Художественный редактор *Л. Г. Епифанов*
Техническое редактирование
и компьютерная вёрстка *Г. А. Филичевой*
Компьютерный набор *Г. В. Черепани*

Налоговая льгота —
Общероссийский классификатор продукции ОК 005-93-953000.
Подписано в печать с оригинал-макета 05.10.2010.
Формат 60×90 $^{1}/_{16}$. Бумага газетная. Гарнитура Newton, рубленая.
Офсетная печать. Усл. печ. 16,0. Уч.-изд. л. 12,29.
Тираж 5000 экз. Заказ 510

Санкт-Петербургский филиал Открытого акционерного общества
«Издательство «Просвещение».
191014, Санкт-Петербург, Литейный пр., 37-39.

Отпечатано с готовых диапозитивов в ГУП «Типография «Наука».
199034, Санкт-Петербург, 9-я линия, 12.