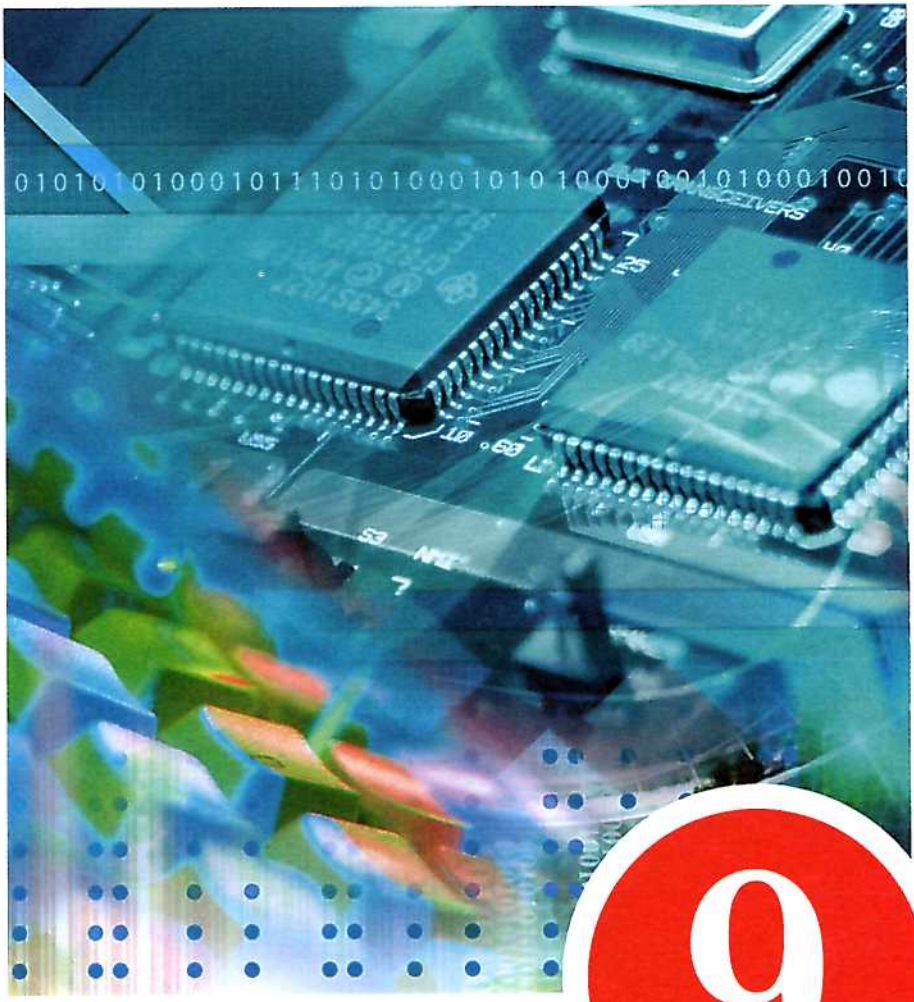
Ю. А. Быкадоров

ИНФОРМАТИКА и ИКТ



9

ДРОФА

ВЕРТИКАЛЬ

Ю. А. Быкадоров

ИНФОРМАТИКА и ИКТ

Учебник для общеобразовательных учреждений

Рекомендовано
Министерством
образования и науки
Российской Федерации

ВЕРТИКАЛЬ

9

Москва
ДРОФА
2013

УДК 373.167.1:002
ББК 32.81я72
Б95

Быкадоров, Ю. А.

Б95 Информатика и ИКТ. 9 кл. : учеб. для общеобразоват. учреждений / Ю. А. Быкадоров. — М. : Дрофа, 2013. — 336 с. : ил.

ISBN 978-5-358-09901-2

Учебник является частью УМК по курсу «Информатика и ИКТ» для 8—9 классов. В учебнике излагаются современные представления об основных понятиях предмета «Информатика и ИКТ» и о возможностях персонального компьютера. Материал учебника основан на применении Microsoft Windows 7 и Microsoft Office 2010 и дополнен изучением языка программирования Pascal. На прилагаемом CD-ROM размещены рабочие материалы для выполнения упражнений, презентации к урокам и фрагменты материалов занятий на базе Microsoft Windows XP и Microsoft Office 2003.

Учебник соответствует Федеральному государственному образовательному стандарту основного общего образования, одобрен РАН и РАО, имеет гриф «Рекомендовано» и включен в Федеральный перечень учебников в составе завершенной предметной линии.

УДК 373.167.1:002
ББК 32.81я72

Учебное издание

Быкадоров Юрий Александрович

ИНФОРМАТИКА И ИКТ. 9 класс

Учебник для общеобразовательных учреждений

Зав. редакцией *О. В. Муравина*. Редакторы *Т. С. Зельдман, Д. Ю. Усенков*
Художественный редактор *А. В. Пряхин*. Художественное оформление *А. В. Копалин*. Технический редактор *И. В. Грибкова*
Компьютерная верстка *Н. В. Полякова*. Корректор *Г. И. Мосякина*

В соответствии с Федеральным законом от 29.12.2010 г. № 436-ФЗ знак информационной продукции на данное издание не ставится

Сертификат соответствия
№ РОСС RU. АЕ51. Н 16238.

Подписано к печати 18.10.12. Формат 60 × 90 $^{1}/_{16}$. Бумага офсетная. Гарнитура «Ньютон». Печать офсетная. Усл. печ. л. 21,0. Тираж 2000 экз. Заказ М-1509.

ООО «Дрофа». 127018, Москва, Сущевский вал, 49.

Предложения и замечания по содержанию и оформлению книги просим направлять в редакцию общего образования издательства «Дрофа»: 127018, Москва, а/я 79. Тел.: (495) 795-05-41. E-mail: chief@drofa.ru

По вопросам приобретения продукции издательства «Дрофа» обращаться по адресу: 127018, Москва, Сущевский вал, 49. Тел.: (495) 795-05-50, 795-05-51. Факс: (495) 795-05-52.

Сайт ООО «Дрофа»: www.drofa.ru

Электронная почта: sales@drofa.ru

Тел.: 8-800-200-05-50 (звонок по России бесплатный)

Отпечатано в типографии филиала ОАО «ТАТМЕДИА» «ПИК «Идел-Пресс». 420066, г. Казань, ул. Декабристов, 2.

ISBN 978-5-358-09901-2

ГЛАВА 1

ВВЕДЕНИЕ В ПРОГРАММИРОВАНИЕ

§ 1. Алгоритмы и исполнители

Алгоритмы. Понятие алгоритма является одним из основных в информатике. Слово «алгоритм» произошло от имени древнего учёного, который жил в 783—850 гг. в г. Хорезме. В своей книге «Об индийском счёте» он изложил правила записи натуральных чисел при помощи арабских цифр и правила арифметических действий над ними (вычисления «столбиком»). В XII в. книгу перевели на латынь, и она стала известна в Европе. Имя учёного по-русски пишется как Хива Абу Абдула Мухаммед бен Муса аль-Маджус аль Хорезми (из Хорезма), что на латинском записали как algorithmi (алгоритм).

Длительное время алгоритмами пользовались только математики, понимая под алгоритмом любое описание процесса решения задачи. Алгоритмы решения математических задач предназначены для исполнения человеком. В алгоритме друг за другом следуют элементарные предписания, в которых описано, что и как делать. Человек исполняет эти предписания и получает результат.

Алгоритм — это точное и понятное исполнителю предписание выполнить конечную последовательность действий для достижения поставленной цели.

В обыденной жизни вместо понятия «алгоритм» используются понятия «план», «порядок действий», «сценарий» (фильма или спектакля). Алгоритмы являются составными частями многих инструкций и правил. Например, инструкция к современному телевизору содержит алгоритмы его настройки с помощью элементарных команд, инструкция по сборке мебели — алгоритм её сборки. Правила дорожного движения содержат алгоритмы проезда перекрёстков автомобилями, перехода дорог пешеходами и др.

Алгоритмизация — это процесс создания алгоритма.

Исполнители алгоритмов. Алгоритм представляет собой набор элементарных предписаний, которые обычно называют командами. Воспринимать и исполнять элементарные предписания (команды) может не только человек. Воспринимают команды и автоматические устройства. Причём некоторые из них способны воспринимать целые наборы команд. Первым из таких устройств является конечно же компьютер. Но и в докомпьютерной истории человечества существовало много различных механических игрушек и устройств, выполнявших сложные наборы действий.

В XVIII—XIX вв. особой известностью пользовались механические подобия людей, названные андроидами в честь французских часовщиков Пьера-Жака и Анри Дро.

Текстильщики всего мира чтят француза Жозефа-Мари Жаккарда, который в 1804—1808 гг. создал ткацкую машину для выработки крупноузорчатых тканей (жаккардовый станок). Набор команд станку задавался с помощью бумажных или картонных перфокарт.

На аналогичных принципах работали в прошлом механические музыкальные устройства: шарманки, шкатулки, ящики, табакерки, механические пианино. В шарманках набор команд записывался в виде выступающих штифтов на специальном барабане или диске. В шкатулках набор команд имел вид точечных углублений на металлическом диске. В Музее радио имени А. С. Попова в г. Екатеринбурге есть прекрасные образцы таких музыкальных устройств.

Исполнителем алгоритмов мы будем называть человека или автоматическое устройство, которое способно воспринимать и исполнять алгоритмы.

Система команд исполнителя — это перечень элементарных предписаний (команд), которые исполнитель может исполнить.

Любой исполнитель имеет ограниченную систему команд. Даже среди людей вряд ли можно найти исполнителя, который умеет делать всё. Один может исполнить только четыре арифметических действия. Другой — найти интеграл от элементарной функции. Если в качестве исполнителей рассматривать школьников, то восьмиклассник может исполнить больше предписаний, чем первоклассник.

Свойства алгоритмов. Любой алгоритм обладает набором отличительных свойств.

- *Целенаправленность* означает, что любой алгоритм направлен на достижение определённой цели. Чаще всего целью алгоритма является получение результата при решении какой-нибудь задачи (математической, производственной, бытовой и др.).
- *Дискретность* проявляется в том, что алгоритм состоит из элементарных предписаний (команд).
- *Понятность* состоит в том, что элементарные предписания (команды) алгоритма должны быть точно сформулированы и однозначно понятны исполнителю, а исполнитель должен быть в состоянии их выполнить.
- *Однозначность* проявляется в том, что после исполнения очередного элементарного предписания (команды) исполнителю точно определено, что делать дальше.
- *Массовость* состоит в том, что алгоритм можно использовать для решения той же задачи при других допустимых исходных данных.

Формы записи алгоритмов. Для алгоритмов используют несколько форм записи:

- словесную;
- графическую;
- на языке программирования.

Если алгоритм предназначен для человека, то в качестве предписаний можно использовать привычные для человека предложения и фразы. Такая форма записи алгоритма называется *словесной*.

Правила записи алгоритмов в словесной форме просты:

- предписания записываются одно за другим и нумеруются;
- в записи алгоритма могут использоваться служебные слова **Начало** и **Конец**.

Упражнение 1

Составим алгоритм кипячения воды в чайнике на электроплите. Дадим алгоритму название (имя) и запишем его.

Алгоритм «Кипячение воды»
Начало

1. Налить в чайник воды.
2. Поставить чайник на электроплиту.
3. Включить электроплиту.
4. Подождать, пока вода закипит.
5. Выключить электроплиту.

Конец

Предписания приведённого алгоритма строгими и однозначными не являются. Например, предписания «Поставить чайник на электроплиту», «Подождать, пока вода закипит» не описывают строго, какие действия нужно совершить, и рассчитаны на опытного человека. Для менее опытного человека такие предписания нужно детализировать, т. е. разбить на более простые, которые этому человеку понятны и могут быть им исполнены.

Алгоритм решения задачи — это только последовательность предписаний (команд) исполнителю. Чтобы достичь цели, исполнитель должен ***исполнить алгоритм***.

Исполнить алгоритм «Кипячение воды» можно только в том случае, когда имеются чайник, электроплита, вода, напряжение в сети и если исполнитель правильно понимает команды. Алгоритмы в словесной форме исполняются, начиная с первого предписания. Если в очередном предписании не указано, к какому предписанию следует перейти, то переходят к следующему предписанию.

Линейные алгоритмы — это алгоритмы, в которых предписания исполняются в той последовательности, в которой они записаны.

Алгоритм «Кипячение воды» исполняется последовательно в порядке записи предписаний. Таким образом, этот алгоритм является линейным.

Вопросы и задания

1. Расскажите о предыстории понятия «алгоритм».

2. Перечислите автоматические устройства, которые можно рассматривать как исполнители алгоритмов.

3. С помощью информационных ресурсов Интернета создайте коллективные рефераты на темы:

1) «История технической игрушки»;

2) «История механической звукозаписи».

4. Какие понятия равносильны понятию алгоритма?

5. В чём заключаются свойства понятности и однозначности алгоритмов?

6. Объясните суть свойства массовости алгоритма.

7. Приведите примеры алгоритмов, которые девятиклассник исполнить сможет, а второклассник — нет.

8. Перечислите формы записи алгоритмов.

9. В каком порядке исполняются предписания линейного алгоритма?

10. Составьте алгоритм действий дежурного в классе перед началом урока.

11. Чем удобна словесная форма записи алгоритмов?

12. Запишите при помощи текстового редактора Word ваш собственный алгоритм действий от утреннего подъёма до выхода из дома в школу. Как исполнялся этот алгоритм на прошлой неделе?

§ 2. Линейные алгоритмы в словесной форме

Примеры алгоритмов. Словесная форма записи характерна для рецептов приготовления блюд, которые по сути являются алгоритмами.

Упражнение 2

Запишем алгоритм (рецепт) приготовления рассыпчатой гречневой каши.

Алгоритм «Приготовление рассыпчатой гречневой каши»

1 Стакан гречневой крупы прожарить на сковороде до тех пор, пока она не подрумянится.

2. Налить в кастрюлю с плотной крышкой ровно два стакана воды, добавить соль и поставить на огонь.
3. Подождать, пока вода закипит.
4. Всыпать в кипящую воду калёную гречневую крупу и накрыть крышкой. Крышку не снимать до полного приготовления каши.
5. Кашу варить 15 мин сначала на сильном, потом на среднем и в конце — на слабом огне.
6. Готовую кашу заправить мелко нарезанным, поджаренным на масле до золотистого цвета репчатым луком и сухими грибами, предварительно обработанными.

Современного человека окружает множество электронных и электрических приборов, среди которых радиоприёмники, телевизоры, телефоны, магнитофоны, видеокамеры, плееры, электронные часы, микроволновые печи, автоматические стиральные машины и т. п. Команды в такие приборы вводятся с помощью управляющих приспособлений (кнопочных, сенсорных, поворотных или движковых). Инструкции для электронных приборов являются сборниками алгоритмов в словесной форме.

■ Упражнение 3

Запишем алгоритм набора номера с трубки из инструкции к беспроводному телефону.

Алгоритм «Набор номера с трубки»

1. Нажать кнопку активизации трубки.
2. Набрать нужный номер.
3. После окончания разговора снова нажать кнопку активизации трубки.

Алгоритмы управления компьютером и программными средствами. Мы уже говорили, что понятия алгоритма и порядка действий являются синонимами. В 8 классе мы неоднократно использовали самые разнообразные порядки действий. Напомним только два из них, давая алгоритмам названия.

Упражнение 4

Запишем порядок действий при выключении компьютера.

Алгоритм «Выключение компьютера»

1. Завершить работу всех прикладных программ.
2. Завершить работу операционной системы Windows.
3. Выключить электропитание компьютера.

Упражнение 5

Запишем порядок действий при запуске на исполнение программы «Блокнот» с помощью кнопки **Пуск**.

Алгоритм «Запуск программы “Блокнот”»

1. Переместить курсор по экрану на кнопку **Пуск** и щёлкнуть левой клавишей мыши.
2. Переместить курсор по меню на пункт **Все программы** и щёлкнуть левой клавишей мыши.
3. Переместить курсор по темной полосе в дополнительное меню, установить на папку «Стандартные» и щёлкнуть левой клавишей мыши.
4. Переместить курсор по тёмной полосе в новое дополнительное меню и там щёлкнуть левой клавишей мыши по пункту **Блокнот**. (На рабочем столе открывается окно программы «Блокнот». Одновременно на панели задач появляется кнопка с названием программы.)

Алгоритмы управления программными средствами при помощи меню мы записывали в виде команд меню. При запуске программы «Блокнот» команда меню имеет вид

Пуск|Все программы|Стандартные|Блокнот

Команда меню — это краткая запись линейного алгоритма управления программным средством.

Для описания информационно-коммуникационных технологий, которые мы изучаем, используют алгоритмы обработки, передачи и хранения информации. Именно с помощью алгоритмов (порядков действий) в 8 классе мы осваивали технологии запуска программных средств, создания компьютерных графических объ-

ектов, обработки текстовой информации, поиска информации в информационных ресурсах Интернета и т. п.

Упражнение 6

Запишем алгоритм создания бумажного делового письма с помощью текстового редактора Word.

Алгоритм «Создание делового письма»

1. Включить компьютер.
2. Загрузить текстовый редактор Word.
3. Загрузить шаблон делового письма.
4. Заполнить шаблон своими данными.
5. Сохранить письмо в файле на диске.
6. Распечатать письмо на принтере.
7. Завершить работу всех программ, операционной системы и выключить компьютер.

Алгоритмы и алгоритмизация составляют основу всего курса «Информатика и ИКТ».

Решение логических задач. Есть целый раздел задач, которые называются *логическими* и результатом решения которых является алгоритм в словесной форме, приводящий к решению задачи.

Упражнение 7

Запишем алгоритм, который приводит к решению задачи: «Человеку, находящемуся на берегу реки, нужно переправить на противоположный берег волка, козу и капусту. В лодку может поместиться человек либо с волком, либо с козой, либо с капустой. На берегу нельзя оставить волка с козой или козу с капустой. Как следует поступить?»

Алгоритм «Перевозчик»

1. Перевезти козу.
2. Возвратиться самому.
3. Перевезти волка.
4. Возвратиться вместе с козой.

5. Перевезти капусту.
6. Возвратиться самому.
7. Перевезти козу.

■ Упражнение 8

Запишем алгоритм решения задачи: «Есть бочка воды и два пустых ведра объёмом 5 и 7 л. Как налить в 5-литровое ведро 4 л воды?»

Алгоритм «Переливания»

1. Из бочки наполнить 7-литровое ведро.
2. Наполнить из 7-литрового ведра 5-литровое ведро. (В 7-литровом ведре останется 2 л воды.)
3. Вылить в бочку воду из 5-литрового ведра.
4. Перелить воду из 7-литрового ведра в 5-литровое ведро. (В 5-литровом ведре имелось 2 л воды.)
5. Из бочки наполнить 7-литровое ведро.
6. Долить 5-литровое ведро из 7-литрового. (В 7-литровом ведре останется 4 л воды.)
7. Вылить в бочку воду из 5-литрового ведра.
8. Перелить воду из 7-литрового ведра в 5-литровое ведро.

□ Вопросы и задания

1. Запишите в словесной форме алгоритм приготовления вашего любимого блюда.

2. Из инструкции к любому электронному прибору выпишите алгоритм в словесной форме.

3. Запишите алгоритм построения на бумаге чертежа прямоугольного треугольника по заданным длинам двух катетов с помощью карандаша и угольника.

4. Определите, для решения какой задачи предназначен следующий алгоритм.

Начало

1. Постройте отрезок AB.
2. Установите раствор циркуля равным длине отрезка AB.
3. Поставьте ножку циркуля в точку A.

4 Проведите окружность с центром в точке A.

5 Поставьте ножку циркуля в точку B.

6 Проведите окружность с центром в точке B.

7 Проведите прямую через точки пересечения окружностей.

Конец

5. Вспомните 10 алгоритмов управления программными средствами (порядков действий), записанных в учебнике для 8 класса.

6. Составьте алгоритм создания бумажной поздравительной открытки с помощью графического редактора Paint.

7. Есть бочка воды и два пустых ведра объёмом 3 и 7 л. Составьте алгоритм, в результате выполнения которого в 3-литровом ведре окажется 2 л воды.

8*[1]. Двое солдат подошли к реке, по которой в лодке катаются двое мальчиков. Составьте алгоритм переправы солдат на противоположный берег, если лодка вмещает только одного солдата или двух мальчиков, а солдата вместе с мальчиком уже не вмещает.

9*. Как поступить, если в предыдущей задаче солдат будет не двое, а целый взвод?

§ 3. Ветвления. Повторения. Блок-схемы

В линейных алгоритмах предписания исполняются в том порядке, в котором они записаны. Однако не все алгоритмы являются линейными. Существуют алгоритмы, в которых одни предписания могут менять порядок исполнения других. В этом случае в предписании указано, что следует перейти в другое место алгоритма.

■ Упражнение 9

Запишем алгоритм междугородного IP-соединения по карте предварительной оплаты.

Алгоритм «Междугородный звонок»

Начало

1 Набрать телефонный номер службы IP-телефонии.

2 Проверить условие «Телефон поддерживает тональный набор».

[1] Значок «*» после номера вопроса (задания) или подзаголовка в тексте учебника обозначает задания или материалы повышенной сложности.

Если условие выполняется, перейти к предписанию 3.
Если условие не выполняется, перейти к предписанию 8.

3. Перейти в тональный набор.
4. Прослушать сообщение.
5. Набрать PIN-код карты предоплаты.
6. Прослушать сообщение.
7. Набрать междугородный код и телефонный номер. Завершить набор символом #.
   Перейти к предписанию 10.
8. Дождаться ответа оператора.
9. Сообщить оператору PIN-код карты предоплаты и телефонный номер.
10. Дождаться соединения.

Конец

Примеры алгоритмов из школьного курса математики — алгоритмы нахождения наибольшего общего делителя, наименьшего общего кратного, разложения числа на простые множители и др.

■ Упражнение 10

Запишем алгоритм нахождения наибольшего общего делителя (НОД) двух натуральных чисел.

Алгоритм «Нахождение НОД двух натуральных чисел»
Начало

1. Записать два заданных натуральных числа.
2. Проверить условие «Числа равны».
   Если условие выполняется, перейти к предписанию 5.
   Если условие не выполняется, перейти к предписанию 3.
3. Вычесть из большего числа меньшее.
4. Записать два числа — вычитаемое и разность.
   Перейти к предписанию 2.
5. Записать любое из полученных чисел как НОД исходных.

Конец

В упражнениях 9, 10 использованы ***предписания с проверкой условия***.

В зависимости от результата проверки условия исполнение алгоритма может пойти двумя разными путями. Если условие выполняется — условие ***верно***, ***истинно*** (*true*), исполнение алгоритма идёт по одному пути, если условие не выполняется — условие ***неверно***, ***ложно*** (*false*) — по другому.

Условия в предписаниях с проверкой имеют вид некоторого утверждения, например условие «Телефон поддерживает тональный набор». Проверка выясняет, выполняется утверждение или нет.

В математике условия имеют вид утверждений, в которых некоторые величины сравниваются, например «Числа равны».

Алгоритмы с ветвлениями (ветвящиеся алгоритмы) — это алгоритмы, в которых при разных условиях исполняются разные наборы команд (ветви алгоритма).

Примером ветвящихся алгоритмов является алгоритм «Междугородный звонок» (см. упр. 9). В этом алгоритме при выполнении условия «Телефон имеет тональный набор» исполняются предписания 1—7, 10. Если условие не выполняется, то исполняются предписания 1, 2, 8, 9, 10. А вот другой пример алгоритма с ветвлением (из курса математики).

■ Упражнение 11

Запишем алгоритм решения квадратного уравнения.

Квадратное уравнение $ax^2 + bx + c = 0$ считается заданным, если заданы его коэффициенты a, b, c, причём $a \neq 0$.

Алгоритм «Решение квадратного уравнения»
Начало

1. Записать числа a, b, c.
2. Вычислить дискриминант $D = b^2 - 4ac$.
3. Проверить условие $D \geqslant 0$.
   Если условие выполняется, перейти к предписанию 4.
   Если условие не выполняется, перейти к предписанию 5.
4. Вычислить в качестве ответа значения выражений

$$x_1 = \frac{-b + \sqrt{D}}{2a} \quad \text{и} \quad x_2 = \frac{-b - \sqrt{D}}{2a}.$$

Перейти к предписанию 6.

5 Ответом является фраза «Уравнение решений не имеет».

6 Записать ответ.

Конец

Алгоритмы с повторениями (циклические алгоритмы) — это алгоритмы, в которых некоторые наборы команд при исполнении алгоритма повторяются (или могут повторяться) несколько раз.

Примером циклического алгоритма является алгоритм «Нахождение НОД двух натуральных чисел» (см. упр. 10). В этом алгоритме несколько раз могут выполняться предписания 2, 3, 4. Вот ещё один циклический алгоритм.

■ Упражнение 12

Запишем алгоритм проезда регулируемого перекрёстка на автомашине.

Алгоритм «Проезд регулируемого перекрёстка»

Начало

1 Подъехать к перекрёстку.

2 Проверить условие «Сигнал светофора зелёный».
Если условие выполняется, перейти к предписанию 4.
Если условие не выполняется, перейти к предписанию 3.

3 Постоять у светофора. Перейти к предписанию 2.

4 Проехать перекрёсток.

Конец

В зависимости от того, в какой последовательности исполняются предписания (команды) алгоритма, выделяют следующие типы алгоритмов:

- линейные алгоритмы;
- алгоритмы с ветвлениями;
- алгоритмы с повторениями;
- алгоритмы с ветвлениями и повторениями.

На практике сложные алгоритмы являются алгоритмами с ветвлениями и повторениями.

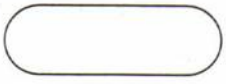
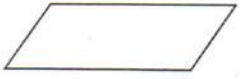
Блок-схемы. Словесная форма достаточно удобна для записи небольших алгоритмов. Если же алгоритм содержит десятки предписаний, то в словесной форме записи довольно трудно проследить всевозможные переходы от одного предписания к другому. Для записи сложных алгоритмов используют графическую форму записи алгоритма, которую называют также ***блок-схемой алгоритма***. Блок-схемы предназначены для прочтения человеком.

При составлении блок-схем выполняют ряд п р а в и л.

- Блок-схемы алгоритмов состоят из графических фигур — блоков, которые содержат словесные описания действий и соединены стрелками.
- Блоки могут нумероваться. Стрелки показывают последовательность исполнения предписаний и могут стыковаться (соединяться) и пересекаться.

Изображения основных блоков, используемых при записи блок-схем алгоритмов, приведены в таблице 1.

Таблица 1

Наименование	Изображение
Блок начала или блок завершения (алгоритма)	
Блок ввода-вывода (данных)	
Блок процесса (действия)	
Блок проверки условия	

Рассмотрим примеры блок-схем алгоритмов.

В линейных алгоритмах блок проверки не используется.

Рис. 1

■ Упражнение 13

Создадим блок-схему линейного алгоритма «Кипячение воды» (см. упр. 1). Каждому предписанию алгоритма соответствует свой блок. Порядок исполнения предписаний задаётся стрелками. Блок-схема приведена на рисунке 1.

В алгоритмах с ветвлениями используется блок проверки. Блок проверки соответствует предписанию проверки условия (для алгоритма в словесной форме). На блок-схеме в блок проверки входит одна стрелка, а выходят две. Выходящие стрелки подписываются кратким ответом «Да» или «Нет» на вопрос «Условие выполняется?».

■ Упражнение 14

Создадим блок-схему алгоритма с ветвлением «Междугородный звонок» (см. упр. 9). Блок-схема алгоритма представлена на рисунке 2. Предписанию проверки условия соответствует блок проверки, из которого стрелки идут в соответствующие блоки.

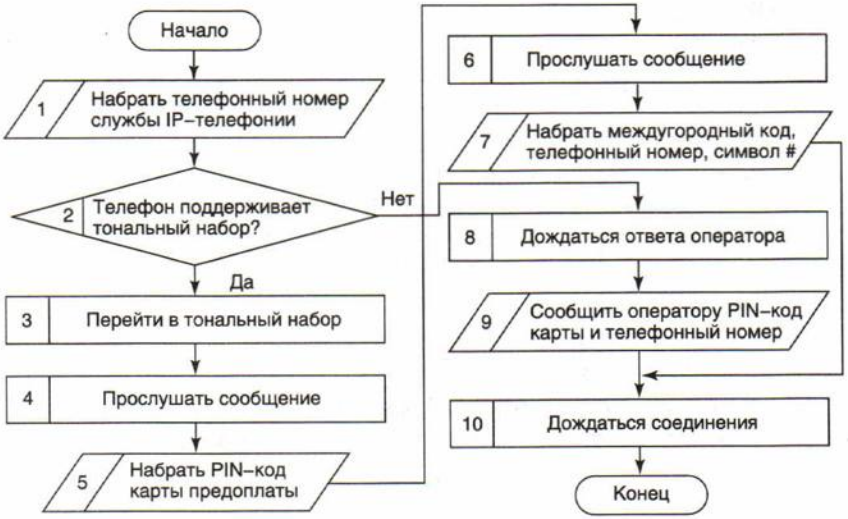

Начало
1 Набрать телефонный номер службы IP–телефонии
2 Телефон поддерживает тональный набор?
Нет
Да
3 Перейти в тональный набор
4 Прослушать сообщение
5 Набрать PIN–код карты предоплаты
6 Прослушать сообщение
7 Набрать междугородный код, телефонный номер, символ #
8 Дождаться ответа оператора
9 Сообщить оператору PIN–код карты и телефонный номер
10 Дождаться соединения
Конец

Рис. 2

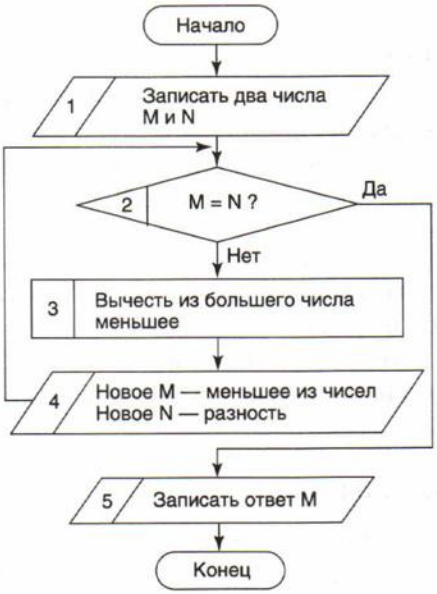

Начало
1 Записать два числа M и N
2 M = N ?
Да
Нет
3 Вычесть из большего числа меньшее
4 Новое M — меньшее из чисел
Новое N — разность
5 Записать ответ M
Конец

Рис. 3

■ Упражнение 15

Создадим блок-схему алгоритма с повторением «Нахождение НОД двух натуральных чисел» (см. упр. 10). Блок-схема алгоритма представлена на рисунке 3.

Алгоритмические конструкции — это конструкции формы записи алгоритма, с помощью которых организуются ветвления, повторения и другие стандартные способы обработки данных.

Различают алгоритмические конструкции:

- *следования* (рис. 4);
- *ветвления* (рис. 5);
- *повторения* (рис. 6).

Конструкцию на рисунке 5 обычно называют *конструкцией полного ветвления*. Если в одной из ветвей конструкции нет блоков процесса, то её называют *конструкцией неполного ветвления* (рис. 7).

☐ Вопросы и задания

1. Перечислите типы алгоритмов.

2. Запишите наборы предписаний, которые исполняются в упражнениях 11 и 12 в зависимости от выполнения условия в алгоритме.

3. Запишите в виде алгоритма с повторениями алгоритм постройки девятиэтажного жилого дома.

4*. Имеются три пары монет: золотые, серебряные и бронзовые. В каждой паре одна монета подлинная, а другая — фальшивая. Все подлинные монеты имеют одинаковый вес. Все фальшивые монеты также имеют одинаковый вес, но каждая легче подлинной. Имеются весы с двумя чашками без гирь. С их помощью можно определить, одинаковый ли вес находится на чашках. Составьте алгоритм, позволяющий за два взвешивания определить подлинные монеты.

5*. Имеются шесть монет, среди которых две фальшивые. Вес фальшивой монеты меньше веса подлинной. Составьте алгоритм, позволяющий за три взвешивания на чашечных весах без гирь определить фальшивые монеты.

6*. Имеются 1000 монет, из них одна фальшивая, которая легче остальных. Составьте алгоритм нахождения фальшивой монеты за семь взвешиваний на чашечных весах без гирь.

7. Какие основные блоки используются при составлении блок-схем?

8. Какие блоки используются в блок-схемах линейных алгоритмов и алгоритмов с повторениями?

9. Чем алгоритмическая конструкция неполного ветвления отличается от алгоритмической конструкции полного ветвления?

10. Какие алгоритмические конструкции использованы в упражнениях этого параграфа?

11. Составьте на бумаге блок-схему алгоритма «Набор номера с трубки» (см. упр. 3).

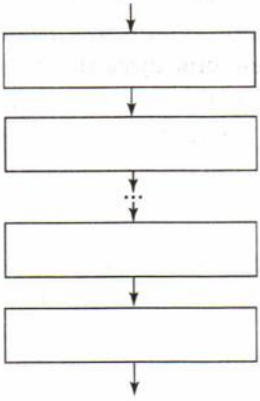


Рис. 4

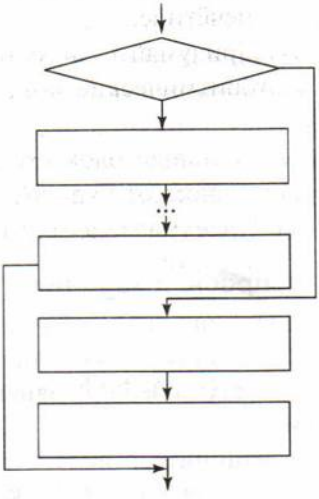


Рис. 5

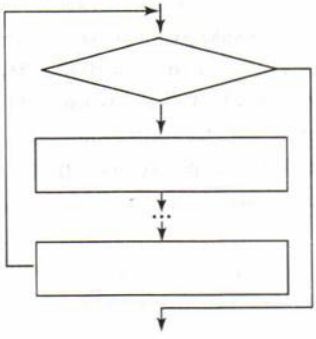


Рис. 6

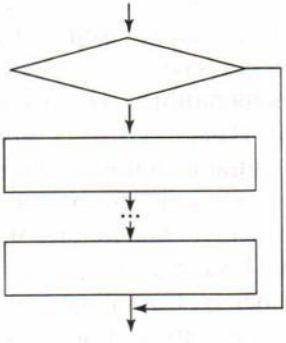


Рис. 7

12. Составьте на бумаге блок-схему алгоритма «Проезд регулируемого перекрёстка» (см. упр. 12). Какие алгоритмические конструкции использует эта блок-схема?

13. Даны два натуральных числа. Составьте блок-схему алгоритма вычисления:

1) суммы этих чисел, если оба числа чётные;

2) частного этих чисел, если оба числа нечётные;

3) произведения этих чисел в случае, если одно из чисел чётное, другое нечётное.

14. Придумайте алгоритмы, блок-схемы которых используют алгоритмические конструкции ветвления (полного и неполного).

15. Составьте блок-схему алгоритма вычисления суммы натуральных чисел от 15 до 47.

16. Придумайте алгоритм с повторением и составьте на бумаге его блок-схему.

§ 4. Языки программирования

Напомним, что язык программирования — это формальный язык для записи компьютерных программ.

Программа — это алгоритм, записанный на языке программирования.

Программирование — процесс создания программы.

Главным исполнителем в компьютере является конечно же процессор.

Но следует отметить, что процессор непосредственно исполняет только программы, написанные на языке двоичных кодов. Этот язык программирования называют также языком машинных кодов.

Программы, написанные на других языках программирования, процессор не воспринимает. Такие программы могут быть переведены на язык машинных кодов с помощью программ, которые называют трансляторами.

Транслятор — это программа, которая преобразует программу, написанную на одном (входном) языке, в программу, представленную на другом (выходном) языке.

Трансляторы бывают двух видов:

- ***компилятор*** — это транслятор, который преобразует программу, написанную на языке программирования высокого уровня, в программу, которую можно исполнить на компьютере;
- ***интерпретатор*** — это транслятор, который способен параллельно переводить и исполнять программу, написанную на языке программирования высокого уровня.

Трансляторы являются составной частью любой системы программирования. Системы программирования включают и некоторые вспомогательные программы (редактор текста программы и др.), которые обеспечивают дополнительные удобства для программиста при работе с транслятором.

Любая система программирования, как правило, имеет два основных режима работы:

- режим ввода текста программы;
- режим исполнения программы на языке программирования.

Учебные системы программирования позволяют работать с исполнителями, которые способны совершать наглядные действия на экране компьютера.

В школьной информатике широко известны такие исполнители, как «Черепашка», «Чертёжник», «Робот».

Классификация языков программирования. Перечень наиболее известных языков программирования мы уже приводили. Сейчас настало время познакомиться с их классификацией, которая возникла в ходе развития компьютерной индустрии.

Языки программирования по принципу удобства использования человеком делятся на:

- ***языки машинных кодов***;
- ***языки программирования низкого уровня*** (Ассемблер);
- ***языки программирования высокого уровня***.

К языкам высокого уровня относятся Fortran («фортран»), Cobol («кобол»), Basic («бэйсик»), Algol («алгол»), Pascal («паскаль»), Modula («модула»), C («си»), C++ («си плюс плюс»), C# («си шарп»), Logo («лого»), Lisp («лисп»), Visual Basic («вижэл бэйсик»), Java («джава») и др.

Из множества языков программирования высокого уровня выделим языки сценариев (языки скриптов, скрипты) VBScript («ви-би-скрипт»), PowerScript («пауэ-скрипт»), JavaScript («джава-скрипт»), PowerShell («пауэ-шэл»), которые обладают неплохими возможностями по обработке информации, а JavaScript ещё и широко используется при создании веб-сайтов.

Эти языки позволяют реагировать на некоторые события в компьютере — щелчок по кнопке мыши, попадание курсора на картинку, завершение ввода данных с клавиатуры и т. п. На языках сценариев пишут программы (сценарии) обработки данных в случаях наступления в компьютере таких событий.

В курсе «Информатика и ИКТ» мы будем изучать возможности языка сценариев JavaScript.

☐ Вопросы и задания

1. Приведите примеры исполнителей алгоритмов.

2. Почему процессор компьютера не может исполнять программы, написанные на языке программирования Basic или Pascal?

3. Что такое компилятор?

4. Что такое интерпретатор?

5. Чем компилятор отличается от интерпретатора?

6. Перечислите основные режимы работы системы программирования.

7. На какие классы делятся языки программирования по принципу удобства их использования?

8. Какие черты отличают языки сценариев?

9. Используя информационные ресурсы Интернета, составьте хронологическую последовательность создания языков программирования с фотографиями их основных разработчиков.

§ 5. Введение в программирование на языке JavaScript

Язык сценариев JavaScript был разработан компанией Netscape. Сначала язык именовался LiveScript. Под новым названием он был представлен 4 декабря 1995 г. совместно компаниями Netscape и Sun. Компания Microsoft вскоре создала свою версию

языка сценариев под названием JScript. Мы будем изучать именно эту версию. Не следует путать язык JavaScript с языком Java. Это разные языки программирования.

JavaScript можно отнести к объектно-ориентированным языкам, хотя возможностями полноценных объектно-ориентированных языков C++ и Java он не обладает.

Программа (сценарий) на языке JavaScript легко встраивается в текст веб-страницы на языке разметки гипертекста HTML. Многие веб-страницы, размещённые в Интернете, содержат сценарии на JavaScript. Такие сценарии добавляют в интерактивные веб-страницы самые неожиданные динамичные эффекты.

Интерпретаторы языка JavaScript. Работа с языками программирования требует наличия на компьютере соответствующей системы программирования или транслятора. Системы программирования и трансляторы популярного в своё время учебного языка программирования Basic в первые персональные компьютеры просто встраивались. Для работы с языком Pascal систему программирования нужно было приобретать.

Язык сценариев JavaScript исполняется на компьютере с помощью интерпретаторов, встроенных в программное обеспечение (в операционную систему Windows и в браузер Internet Explorer). В ОС Windows интерпретатор носит название «Сервер сценариев Windows».

С языком сценариев JavaScript можно работать в текстовом редакторе «Блокнот». Причём удобнее использовать его расширенный вариант, который носит имя «Notepad++» и свободно распространяется в Интернете.

Первая программа на языке JavaScript. Программа представляет собой текст формата TXT. Файлу с программой на языке JavaScript присваивают расширение .js (тип JS).

Программу на языке JavaScript называют также **JS-программой** или **сценарием.** Программу (на любом языке) часто называют **программным кодом**, а команды программы — **инструкциями**.

■ Упражнение 16

Создадим файл с простейшей программой на языке JavaScript и исполним его с помощью сервера сценариев Windows.

В текстовом редакторе «Блокнот» вводим команду вывода для сервера сценариев:

```
WScript.echo("Привет от сервера сценариев Windows!");
```

Сохраняем файл под именем ex16.js в рабочей папке. Для этого вводим команду меню **Файл|Сохранить как** . Открывается диалоговое окно. В окне открываем список поля «Тип файла» и выбираем пункт **Все файлы**. Вводим имя файла с расширением .js.

С помощью программы «Проводник» находим имя нашего JS-файла и двойным щелчком по имени запускаем его на исполнение. Всё остальное сервер сценариев сделает сам.

Рис. 8

Появляется сообщение в диалоговом окне (рис. 8).

Использование интерпретатора браузера. В наибольшей степени возможности языка JavaScript раскрываются в браузере Internet Explorer.

■ Упражнение 17

Создадим файл с простейшей программой на языке JavaScript и исполним его с помощью браузера.

Запускаем текстовый редактор «Notepad++» и вводим текст (можно использовать файл-заготовку pre1.txt с CD-диска):

```
<HTML>
<!-- saved from url=(0014)about:internet -->
<Script>
document.writeln("Привет от интерпретатора браузера");
</Script>
</HTML>
```

Вводим команду меню **Синтаксис|H|HTML**, которая позволяет выделить цветом типовые элементы программы.

Формально мы создаём программу на языке разметки гипертекстов HTML (HTML-программу), поэтому весь текст обрамляется тегами <HTML> и </HTML>. Теги выделяются голубым цветом.

Вторая строка программы включает режим автоматического исполнения JS-сценариев в браузере.

В HTML-программу вставлен текст JS-программы (одна команда), поэтому текст JS-программы обрамляется тегами <Script> и </Script>.

Сохраняем файл под именем ex17.htm.

Запускаем файл ex17.htm на исполнение командой меню **Запуск|Launch in IE**. Расширение .htm браузер подхватит и отобразит текст приветствия в своём окне. В адресной строке браузера отображается полное имя файла.

Замечание. Если вместо приветствия отображаются непонятные символы, браузер нужно настроить. Командой меню **Вид|Кодировка** вызывается меню, в котором удаляют флажок в пункте **Автовыбор** (если он там есть) и выбирают пункт **Кириллица (Windows)**.

Подключение внешних файлов JavaScript. Программу на языке JavaScript не обязательно вписывать в HTML-программу. Её можно сохранить в файле типа JS и подключить к HTML-программе.

■ Упражнение 18

Подключим к HTML-программе внешний файл prim.js из папки Prim, которую необходимо скопировать в рабочую папку с CD-диска. 

В новой вкладке «Notepad++» запишем текст HTML-программы (можно также использовать файл-заготовку pre1.txt):

```
<HTML>
<!-- saved from url=(0014)about:internet -->
<Script src="Prim\prim.js"> </Script>
</HTML>
```

Снова вводим команду меню **Синтаксис|H|HTML**. Сохраним программу в рабочей папке в файле с именем ex18.htm.

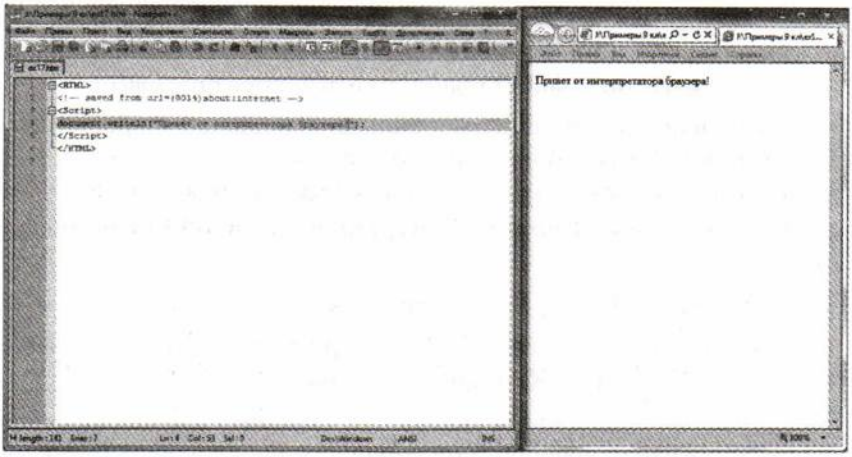


Рис. 9

Теперь файл ex18.htm нужно командой меню **Запуск|Launch in IE** запустить в браузере. После запуска подключённый JS-файл демонстрирует некоторые возможности языка сценариев JavaScript.

Простейшая система программирования. В результате выполнения упражнения 17 мы освоили простейшую систему программирования JavaScript, которую составили текстовый редактор «Notepad++» и браузер. Эта система позволяет разрабатывать программы на языке JavaScript и запускать их на исполнение. Для большего удобства окна «Notepad++» и браузера следует размещать на экране без перекрытий (рис. 9).

Начальный алгоритм работы с простейшей системой программирования JavaScript следующий.

1. Запустить редактор «Notepad++» и при необходимости открыть новую вкладку командой меню **Файл|Новый** или загрузить файл-заготовку командой **Файл|Открыть**.
2. Ввести команду меню **Синтаксис|H|HTML**.
3. Создать текст программы.
4. Сохранить текст программы в файле, для чего:
   - ввести команду меню **Файл|Сохранить как** (открывается диалоговое окно);
   - ввести имя файла с расширением .htm и щёлкнуть по кнопке **Сохранить**.

5. Запустить HTML-программу в браузере, для чего ввести в «Notepad++» команду меню **Запуск|Launch in IE** или сочетание клавиш **Ctrl+Alt+Shift+I** (файл открывается в браузере).

После исполнения начального алгоритма работы с простейшей системой программирования на первых порах придётся также развести по экрану окна текстового редактора и браузера.

Алгоритм доработки программы включает следующие предписания.

1. В окне «Notepad++» изменить текст программы.
2. Сохранить изменения в тексте программы, для чего ввести команду меню **Файл|Сохранить** или сочетание **Ctrl+S**.
3. В браузере щёлкнуть по кнопке **Обновить** правее адресной строки (изменённая программа запускается на исполнение).

■ Упражнение 19

Сократим текст приветствия (см. упр. 17) до слова «Привет!» и запустим программу на исполнение, используя алгоритм доработки программы.

Если возникает надобность сохранить файл под другим именем, то опять используют команду меню **Файл|Сохранить как**.

Для создания и запуска программы на языке JavaScript с помощью простейшей системы программирования мы будем использовать следующую HTML-конструкцию, которая хранится в файле pre2.txt на CD-диске  :

```
<HTML>
<!-- saved from url=(0014)about:internet -->
<title> </title>
<Script>

</Script>
</HTML>
```

Между тегами <title> </title> мы будем записывать *название JS-программы*.

Между тегами <Script> </Script> мы будем записывать *текст JS-программы*.

Вопросы и задания

1. Как программы на языке сценариев JavaScript встраиваются в программы на языке разметки гипертекста HTML?

2. Где размещены интерпретаторы языка JavaScript в современном компьютере?

3. Что такое сервер сценариев Windows?

4. Какое расширение присваивается файлам с программами на языке JavaScript?

5. Как в текстовом редакторе «Блокнот» сохранить файл с расширением, отличающимся от .txt?

6. Как запускается на исполнение файл типа JS?

7. Какие теги обрамляют программу на языке JavaScript в файле с расширением .htm?

8. Запишите в программе (см. упр. 17) в разных строках две-три команды вывода с разными текстами. Запустите программу на исполнение. Каким оказался результат?

9. Запишите текст HTML-программы для подключения внешнего файла local.js.

10. Каков начальный алгоритм работы с простейшей системой программирования JavaScript?

11. Каков алгоритм доработки программы в простейшей системе программирования JavaScript?

§ 6. Основные понятия языка программирования JavaScript

Типы данных. В математике при вычислениях по формулам данными являются числа. В информатике в качестве данных используются числа, строки (цепочки) символов, тексты (наборы строк) и др.

Числа в языке сценариев JavaScript записываются по обычным правилам, но в дробных числах целая и дробная части разделяются точкой. Например: 2.3; 17.55.

При записи отрицательных чисел используется знак «–» (минус), при записи положительных — «+» (плюс). Например: –347.67; +0.56. Как обычно, знак «плюс» никто не пишет, потому что +0.56 и 0.56 — это одно и то же число.

Числа в языке JavaScript называются данными типа ***number*** или данными **числового типа**.

Строки символов в языке JavaScript записываются в кавычках — двойных «"» или одинарных «'». Например: "Привет!"; 'ответ на вопрос'; "–347.67".

Число –347.67 и строка "–347.67" в программе на языке JavaScript — это не одно и то же. Строка "" не содержит ни одного символа и называется пустой (*пустая строка*).

Строки символов в языке JavaScript называются данными типа ***string*** или данными **строкового типа**.

С другими типами данных мы познакомимся позднее.

Переменные величины. В математике и информатике для записи выражений в виде формул используют переменные величины.

Переменная величина — это условный объект, который имеет имя и значение. Имя используется для записи формульного выражения, а значение заменяет имя при вычислении. Значение переменной величины при исполнении программы может меняться.

По одной формуле можно вычислять результаты для разных значений входящих величин.

В языке JavaScript имя переменной величины имеет вид слова (без пробелов). Кроме букв в именах могут использоваться символ подчёркивания «_» и цифры, но с цифры имя начинаться не может.

Примеры имён переменных величин:

```
a, _AB, Summa, x1, malt4_345, a2b1, a_b.
```

В языке JavaScript в качестве имён величин нельзя использовать служебные слова (слова команд языка). Строчные и прописные символы считаются различными: alpha, Alpha и ALPHA — это разные имена.

Команда присваивания. Значения переменных величин в языке JavaScript задаются командой, которая строится с помощью оператора присваивания.

Оператор присваивания обозначается знаком равенства «=» и читается «присвоить значение».

Принцип построения команды присваивания ясен из примера:

```
x1=3.78;
```

Команда читается так: «величине x1 присвоить значение 3.78». По этой команде величине с именем x1 (слева от знака равенства) будет присвоено значение 3.78 (справа от знака равенства).

Переменным величинам могут присваиваться значения любого типа — числового, строкового и др. Например:

```
tr2="Нет ответа";
```

Правила записи программы на языке JavaScript. В программе на языке JavaScript команды записываются одна за другой и отделяются друг от друга точкой с запятой «;». В одной строке программы могут быть записаны несколько команд:

```
x21=3.8;  a2=-34.8;  alts="Море";
```

В конце строки точку с запятой можно не ставить, но это считается плохим тоном.

Комментарий — это пояснительный текст, который в программе записан, но при исполнении программы игнорируется.

Комментарий в одной строке располагают правее знака, который составляют две косые черты «//». Например:

```
//Пример вычислений
x1=3.8;                          //Начальные данные
x2=-4.8;
```

Если комментарий занимает несколько строк, то перед ним ставится знак, который составляют косая черта и звёздочка «/*». После комментария ставится знак из звёздочки и косой черты «*/». Например:

```
x=1.18;     /*Обозначения для переменных величин дают
              возможность по имени переменной выяснить,
              с каким процессом эта переменная связана*/
y=3.1;
```

☐ Вопросы и задания

1. Как записываются в языке JavaScript дробные числа?

2. Какие символы используются в языке JavaScript для обрамления данных строкового типа?

3. Как образуются имена переменных величин в языке JavaScript?

4. Как задаются в языке JavaScript значения переменных величин?

5. Как в программе записываются комментарии?

6. Запишите в текстовом редакторе «Notepad++» команды присваивания:

1) переменной kl значения –4,35;
2) переменной dec значения 0,072;
3) переменной tok значения «разность»;
4) переменной tell значения «Большая перемена».

§ 7. Арифметические операторы и выражения. Объекты

Арифметические операторы. Вычисления в математике проводятся по формулам, которые в информатике называются ***выражениями***. Выражения строятся при помощи операторов.

Перечень ***арифметических операторов*** языка JavaScript приведён в таблице 2.

Таблица 2

Арифметический оператор	Название операции	Пример записи выражения
+	Сложение	a+b
–	Вычитание	a–b
*	Умножение	a*b
/	Деление	a/b
%	Остаток от деления	a%b

Первые четыре оператора хорошо нам известны по работе с виртуальным «Калькулятором». Операция «остаток от деления»

выдаёт остаток от деления первого числа на второе. Например, результатом исполнения выражения 11%4 будет число 3.

Арифметические выражения. Арифметические выражения так же, как и для «Калькулятора», записываются в одну строку в виде линейной цепочки символов. Приведём примеры записи арифметических выражений:

```
12.5+a1      d/2          3.1*x        a%I          x*x*x
```

При необходимости в выражениях используются круглые скобки:

$\frac{a+b}{c+d}$ ⇒ `(a+b)/(c+d)`

$a+\frac{b}{c}+d$ ⇒ `a+b/c+d`

Пробелы в выражениях (между знаками операторов, именами переменных, числами и скобками) интерпретатор игнорирует. Поэтому одинаково правильно исполняются операторы:

`2.5+a12*(4.7 – 2.8)` и `2.5 + a12 * ( 4.7 – 2.8 )`

Объекты. В арифметических выражениях можно использовать математические функции. В языке JavaScript набор математических функций входит в более широкий набор, который называется объектом.

Объект языка JavaScript — это описание любого объекта окружающего мира в виде набора переменных величин, команд и функций, объединённых одним именем.
Переменные величины, входящие в объект, называются ***свойствами объекта***. Свойства объекта хранят данные об объекте.
Команды и функции, входящие в объект, называются ***методами объекта***. Методы объекта позволяют проводить операции над данными.

Имя объекта языка JavaScript строится так же, как имя переменной величины. Например: `Myarray`, `Scr1`, `Mart`.

Существуют стандартные (встроенные) объекты с именами `Math`, `document`, `window` и др.

Свойства и методы объектов имеют краткие и полные имена. Краткие имена строятся по принципу имён переменных, причём к именам методов добавляются круглые скобки. Например: свойства `length`, `colorDepth`; методы `getDay()`, `toFixed()`.

Полные имена свойств и методов похожи на доменные имена.

Полное имя свойства объекта строится по шаблону:

Имя_объекта.Имя_свойства

Например: Myarray.length или Scr1.colorDepth.

Полное имя метода объекта строится по аналогичному шаблону:

Имя_объекта.Имя_метода( )

Например: Mart.getDay() или Scr1.toFixed().

На первый взгляд понятие объекта довольно искусственное, а имена длинноваты. Но при создании сложных программ использование такого подхода даёт существенные преимущества.

Объект Math (математика) объединяет математические константы, функции и некоторые операции.

Свойства стандартного объекта Math (математика) имеют неизменяемые значения:

Math.PI — значение постоянной π;

Math.E — значение числа *e* и др.

Методами объекта Math (математика) в основном являются обычные математические функции:

Math.sqrt() — корень квадратный от аргумента;

Math.abs() — абсолютная величина аргумента;

Math.pow(a, b) — число *a* в степени *b* и др.

Методами объекта Math (математика) являются также математические операции, построенные в виде функций:

Math.round() — округление аргумента до целого числа;

Math.floor() — округление аргумента до целого в сторону уменьшения;

Math.ceil() — округление аргумента до целого в сторону увеличения и др.

☐ Вопросы и задания

1. Перечислите арифметические операторы языка JavaScript.

2. Как действует операция «остаток от деления»?

3. Как лишние пробелы в записи выражений влияют на результат их исполнения?

4. Как образуются имена объектов языка JavaScript?

5. Что такое свойство объекта?

6. Что такое метод объекта?
7. Как образуются имена свойств объектов?
8. Как образуются имена методов объектов?
9. Запишите на языке JavaScript выражение:

1) $\sqrt{a+b}$; 2) $3{,}7 \cdot f_2 - 5{,}36 \cdot x^3$;

3) $(u + 2v)\dfrac{(1{,}89|u| + 0{,}15 - 0{,}68v)}{1 + \sqrt{u^2 + v^2}}$.

§ 8. Линейные программы вычислений на языке JavaScript

Организация вычислений в программе. Для организации вычислений в программе используются команды присваивания. Справа от оператора присваивания записывают арифметические выражения. Слева от этого оператора записывают имя переменной величины. Значение результата вычислений присваивается этой переменной. Например:

```
p=12.5+a1;
a4=(3.1*x+5.6*y)/7+7.9*(3.9+x);
```

Операции в выражениях для числовых данных исполняются по обычным правилам.

В выражениях можно использовать свойства и методы объекта Math (математика).

Например, математические формулы

$$m_1 = 34{,}7 - \sqrt{x_1^2 + y_1} \quad \text{и} \quad m_2 = |x - 3{,}5| \cdot \left(\frac{\pi x}{3}\right)^3$$

преобразуются в команды присваивания:

```
m1=34.7-Math.sqrt(x1*x1+y1);
m2=Math.abs(x-3.5)*Math.pow((Math.PI*x/3),3);
```

Вывод результатов вычислений из программы. Для вывода результатов вычислений из программы используют одну из двух команд:

- document.writeln();
- alert().

По форме записи видно, что первая команда является методом стандартного объекта document. Следует сказать, что вторая команда — это также метод объекта с именем window. Но объект window единственный, имя которого можно опускать при записи полного имени метода, поэтому мы пишем alert() вместо window.alert().

Первую команду мы уже использовали при выводе приветствия в окно браузера. Аргументом команды записывают имя переменной, значение которой надо вывести. Например:

```
p=12.5+35.6/3.6;
document.writeln(p);
```

Команда alert() выводит результат в диалоговое окно. Её аргумент записывается аналогично предыдущей команде. Например:

```
x = 1.2; y = –4;
a4=(3.1*x+5.6*y)/7+7.9*(3.9+x);
alert(a4);
```

Мы рассмотрели примеры, в которых одни методы языка JavaScript используются в программах как обычные функции (методы объекта Math), а другие — как команды (методы document.writeln() и alert()). Позднее мы познакомимся с методами, которые можно использовать как в качестве функций, так и в качестве команд.

■ Упражнение 20

Составим программу вычислений по формуле

$$\frac{0{,}36 \cdot |a| + 1{,}6 \cdot \sqrt{4 + b^2}}{1{,}7 \cdot (0{,}425 - 0{,}78 \cdot b)}$$

при разных значениях величин a и b.

Записываем в «Notepad++» HTML-конструкцию, представленную в конце параграфа 5.

В конструкцию между тегами <title> </title> записываем название программы «Вычисление по формуле». Произвольно задаём начальные значения a (= 1) и b (= 1).

Числитель и знаменатель формулы, а также её результат запишем в новые переменные. Имена переменным даются произвольно, но со смыслом.

Записываем команды присваивания и вывода:

```
a=1;
b=1;
up=0.36*Math.abs(a)+1.6*Math.sqrt(4+b*b);
dn=1.7*(0.425-0.78*b);
rez=up/dn;
alert(rez);
```

Сохраняем файл с именем ex20.htm, запускаем его в браузере.

Если появилось диалоговое окно с ответом, то текст программы введён правильно. Если окна нет, то, скорее всего, в текст закрались ошибки.

Отладка программы. При составлении программы часто допускаются ошибки. Практика показывает, что это неизбежно.

Отладка программы — это процесс выявления и устранения ошибок в программе.

Ошибки в программах делятся на синтаксические и логические.

Синтаксические ошибки — это ошибки в написании команд программы.
Логические ошибки — это ошибки алгоритма.

Синтаксические ошибки часто допускают при расстановке всевозможных скобок. Иногда программисты пропускают буквы в именах объектов, в выражениях используют переменные, для которых не заданы значения и т. п.

Синтаксические ошибки в программах на языке JavaScript интерпретатор браузера выявляет автоматически.

Если программа имеет синтаксическую ошибку, в углу окна браузера IE8 (рис. 10) появляется значок предупреждения (окно браузера иногда нужно развернуть). В браузере IE9 предупреждения не предусмотрены. Двойной щелчок по значку предупреждения в IE8 или по строке состояния в IE9 выводит диалоговое окно, где важно показать подробности (рис. 11).

Чтобы окно всегда появлялось при наличии ошибок, следует в нём пометить флажок в месте, отмеченном стрелкой на рисунке 11.

В диалоговом окне интерпретатор указывает номер строки и место расположения неверного символа, даёт описание ошибки.

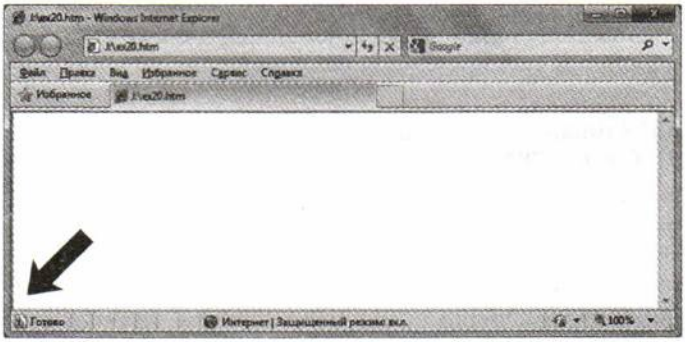


Рис. 10

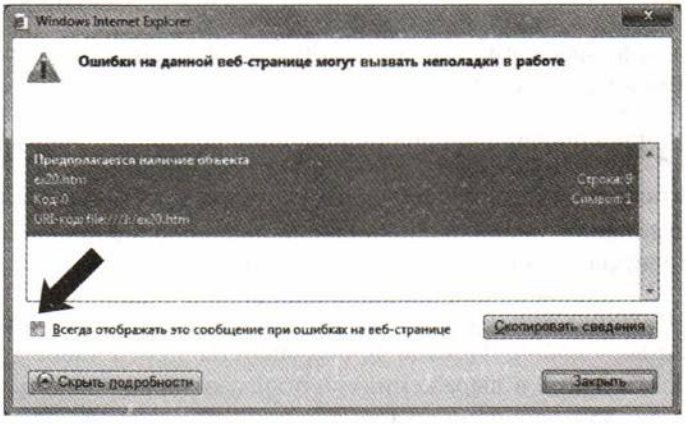


Рис. 11

Программист устраняет ошибку, сохраняет программу и запускает её снова, пользуясь алгоритмом доработки программы, представленным в § 5. Так можно добиться исправления всех синтаксических ошибок в программе.

■ Упражнение 20 (продолжение)

Если синтаксические ошибки в программе были, устраняем их. Если ошибок не было, для проверки работы интерпретатора внесём ошибку. Например, удалим одну букву в команде alert(), сохраним и запустим файл. Ошибка сразу будет выявлена.

В упражнении 20 мы составили простой линейный алгоритм, поэтому логических ошибок в нём, конечно, нет. Но если алгоритм имеет много ветвлений, то логические ошибки — не редкость.

Поиск логических ошибок в программах проводится в основном тестированием.

Тест в программировании — это совокупность исходных данных и соответствующих им результатов.

Обычно готовится набор тестов, способных охватить максимум возможных ситуаций (ветвей алгоритма). Технология тестирования сводится к испытанию программы на всём наборе тестов.

Не секрет, что чем сложнее алгоритм, тем труднее найти в нём логическую ошибку. Один из самых авторитетных специалистов-теоретиков программирования голландский учёный Эдсгер Дейкстра (1930—2002) считал, что тестирование способно показать только наличие ошибок, но не их отсутствие, поскольку при новых входных данных могут обнаружиться новые ошибки.

Некоторые логические ошибки в программах устраняются через месяцы эксплуатации. Сообщения о таких ошибках нередко появляются в средствах массовой информации.

Вопросы и задания

1. Как при помощи команды document.writeln() вывести в окне браузера результат вычислений?

2. В чём состоит особенность вывода данных командой alert()?

3. Какие виды ошибок встречаются в программах?

4. Каким образом выявляются синтаксические ошибки в программах на языке JavaScript?

5. Каким методом устраняются синтаксические ошибки в программах?

6. Каким методом выявляются логические ошибки в программах?

7. Что такое тест в программировании?

8. Создайте на языке JavaScript программы вычислений по следующим формулам:

1) $\dfrac{\sqrt{2{,}5c^2 + 0{,}34ab}}{a^2}(1{,}7b - 5)$;

2) $\dfrac{|4x^2 - 5x + 3y|}{\sqrt{2 + 7x^2}}$;

3) $\dfrac{5{,}3 : 2{,}4 - u}{0{,}702 : 1{,}3 + v^2} + 3{,}73$;

4) $\left|\dfrac{a^2 - b^2}{c}\right| + \left|\dfrac{c^2 - a^2}{b}\right|$.

ГЛАВА 2

ИСПОЛНИТЕЛЬ «ФЛОМАСТЕР»

§ 9. Общие сведения

Описание исполнителя «Фломастер». Исполнитель «Фломастер» предназначен для создания на экране компьютера рисунков, чертежей и графиков.

Создавать компьютерные графические объекты позволяет графический редактор Paint. В окне графического редактора рисунки строятся вручную: на виртуальный холст пользователь последовательно наносит следы от курсора мыши.

Исполнитель «Фломастер» строит чертежи и рисунки другим способом.

Исполнитель «Фломастер» — это невидимый виртуальный объект, который может оставлять цветные следы на экране компьютера в соответствии с программой.

Чтобы создать рисунок, нужно составить программу действий «Фломастера» и запустить её на исполнение. «Фломастер», исполняя программу, оставит на экране цветные следы, которые и составят рисунок (рис. 12). По такому же принципу построен уже давно известный исполнитель «Чертёжник».

Для рисования «Фломастер» использует виртуальный холст. На холсте задана обычная прямоугольная система координат, начало которой находится в левом нижнем углу. Весь холст поделён на клетки-квадраты со стороной в одну единицу длины.

Размеры холста — 10×10 клеток.

Холст появляется в окне браузера после запуска на исполнение программы для «Фломастера».

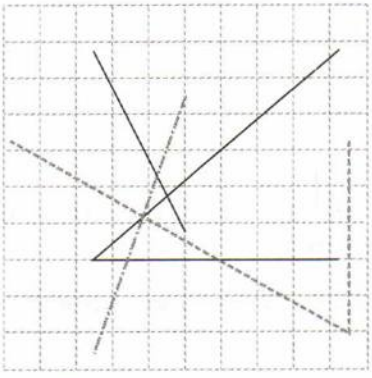


Рис. 12

Объект Flo. Исполнитель «Фломастер» в системе программирования JavaScript задаётся как объект с именем Flo (фломастер).

Объект объединяет несколько методов и свойств.

Метод line() — построение прямой линии.

Flo.line(x1,y1,x2,y2) — команда «Фломастеру» нарисовать на холсте прямую линию от точки с координатами (x1,y1) до точки с координатами (x2,y2).

Координаты задаются как целыми, так и дробными числами. Координаты могут задаваться и за пределами холста, но за пределами холста «Фломастер» следов не оставляет. Цвет линии — чёрный.

Свойства width (толщина следа) и style (стиль следа).

Flo.width хранит код толщины следа «Фломастера». Свойству присваивается значение толщины линии в пикселах (целое число от 1 до 3). По умолчанию задана толщина 1.

Flo.style хранит код стиля следа «Фломастера» (целое число от 1 до 3): 1 — непрерывная линия; 2 — штриховая линия; 3 — пунктирная линия.

По умолчанию задан стиль непрерывной линии (код 1).

Значения свойств «Фломастера» при необходимости задаются командами присваивания (перед командой рисования), например:

```
Flo.width=3;          //Толщина равна 3
Flo.style=2;          //Штриховая
Flo.line(1,1,2,2);    //Нарисовать штриховую линию
                      //толщиной 3 пиксела
```

С другими свойствами и методами объекта Flo мы познакомимся позднее.

Подготовка компьютера к работе с «Фломастером». На прилагаемом CD-диске в папке JS собраны файлы, которые обеспечивают работу исполнителя «Фломастер» и облегчают работу с языком JavaScript.

Внимание! 1. Следует скопировать папку JS с прилагаемого CD-диска в корневую папку диска C: (при помощи «Проводника»).

2. Следует создать на рабочем столе ярлык для файла Help_JS.htm, скопированного на диск C: в составе папки JS. Этот файл содержит документ «Справка JavaScript», в котором описаны команды языка JavaScript и исполнителя «Фломастер».

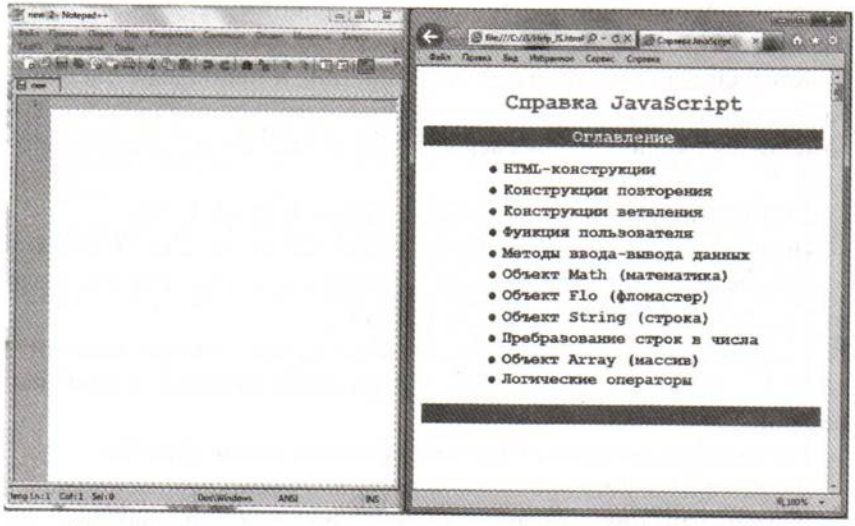


Рис. 13

Готовясь к составлению программ для исполнителя «Фломастер», следует запустить сначала «Notepad++», а затем при помощи ярлыка — файл Help_JS.htm (файл откроется в браузере).

Располагать окна текстового редактора и браузера рекомендуется без перекрытия (рис. 13).

HTML-конструкция для «Фломастера». Чтобы использовать объект Flo в программе на языке JavaScript, нужно к HTML-документу подключить программу flom.js с описанием объекта. В первой главе подобное подключение мы уже проделывали. Программа flom.js находится в папке C:\JS, поэтому при работе с «Фломастером» мы будем использовать HTML-конструкцию вида:

```
<HTML>
<!-- saved from url=(0014)about:internet -->
<Script src="C:\JS\flom.js"></Script>
<title>          </title>
<Script>


</Script>
</HTML>
```

Теги <Script src="C:\JS\flom.js"></Script> подключают файл flom.js. Между тегами <title> </title> мы будем по-прежнему записывать название программы.

Использование файла Help_JS.htm. Чтобы ускорить процесс создания программ на языке JavaScript, HTML-конструкцию для «Фломастера», а также другие конструкции и команды при необходимости мы не будем набирать с клавиатуры, а будем копировать из файла Help_JS.htm.

Файл Help_JS.htm используется для ввода конструкций, свойств и методов в программу в следующем п о р я д к е.

1. Найти нужную конструкцию в окне браузера с файлом Help_JS.htm.
2. Скопировать конструкцию в буфер обмена, для чего щёлкнуть по кнопке **Копировать** около конструкции.
3. Перейти в окно программы «Notepad++» и щёлкнуть левой клавишей в месте вставки конструкции.

4 Вставить конструкцию в текст программы, для чего:

• щёлкнуть правой клавишей мыши (появляется контекстное меню);

• в контекстном меню щёлкнуть левой клавишей мыши по пункту **Вставить.**

☐ Вопросы и задания

1. Опишите способ, которым исполнитель «Фломастер» строит чертежи и рисунки.

2. В каком порядке записываются аргументы метода Flo.line()?

3. Сколько свойств объекта Flo вы знаете?

4. Какую толщину можно выбрать для следа «Фломастера»?

5. Какие стили изображения линий можно выбрать для следа «Фломастера»?

6. В чём отличие HTML-конструкции для «Фломастера» от обычной HTML-конструкции для программ на языке сценариев JavaScript?

§ 10. Линейные алгоритмы

Изображение прямоугольника.

■ Упражнение 21

Построим программу для изображения прямоугольника размерами 2 на 3 клетки.

Будем использовать следующий порядок создания программы.

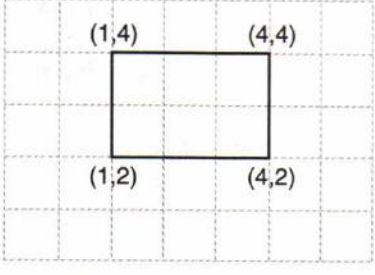


Рис. 14

1 Строим фигуру на клетчатой бумаге и определяем координаты концов прямых отрезков (рис. 14).

2 Вводим текст программы, для чего:

• копируем в «Notepad++» HTML-конструкцию «Фломастера» из файла Help_JS. htm, открытого в браузере;

• даём программе имя «Прямоугольник» (записываем имя между тегами <title> </title>);

• снова используя копирование, записываем команды (в данном случае их четыре):

```
Flo.line(1,2,1,4);
Flo.line(1,2,4,2);
Flo.line(1,4,4,4);
Flo.line(4,2,4,4);
```

3. Сохраняем программу в файле ex21.htm.

4. С помощью команды меню запускаем программу на исполнение (рис. 15).

Работать с исполнителем «Фломастер» удобнее, если расположить окно браузера с холстом на месте окна браузера с файлом Help_JS.htm. На экран эти окна можно выводить попеременно, используя соответствующие им кнопки на панели задач операционной системы.

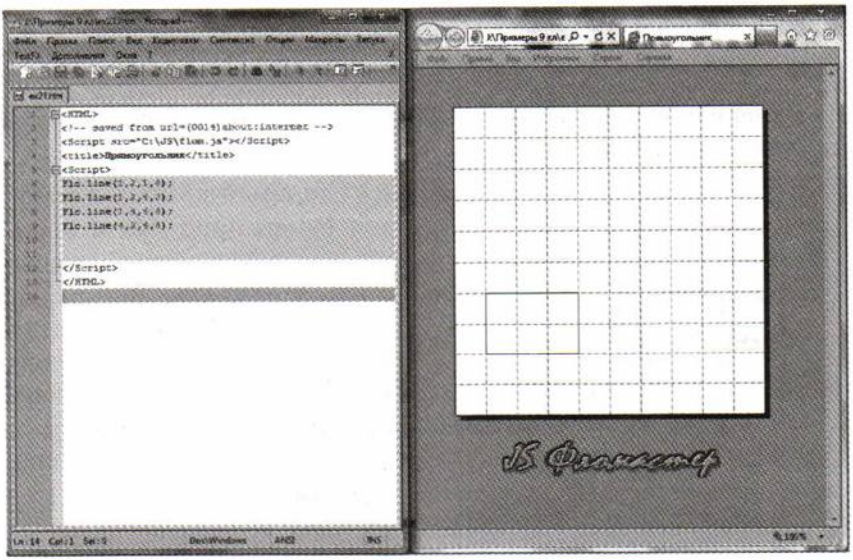


Рис. 15

Замечание. Щелчок по холсту «Фломастера» в окне браузера скрывает клетчатую разметку холста, а одиночные щелчки по надписи «JS Фломастер» меняют оформление холста.

5 Проводим синтаксическую отладку программы (поиск и устранение синтаксических ошибок).

После запуска программы для «Фломастера» может появиться диалоговое окно с предупреждением об ошибках. Если диалоговое окно не появилось, значит, программа синтаксических ошибок не содержит. Если синтаксические ошибки есть, их надо устранить.

6 Проводим отладку логики программы (поиск и устранение логических ошибок).

Логические ошибки в программе для «Фломастера» приводят к неправильному изображению фигур на холсте. На этом этапе устанавливают причину неправильного изображения (ошибки алгоритма) и устраняют её.

В итоге «Фломастер» должен оставить на холсте чёрные следы толщиной 1 пиксел в виде прямоугольника.

Изображение букв.

Упражнение 22

Построим программу для изображения букв «Н» и «А».

Разберём основные этапы алгоритма создания программы для изображения указанных букв.

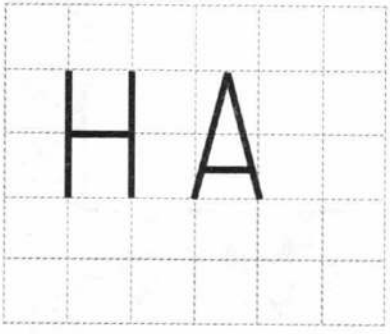


Рис. 16

1 Строим буквы на клетчатой бумаге (рис. 16) и определяем координаты концов прямых отрезков. Возникают сложности с координатами концов перекладины буквы «А». Следует записать примерные (на глаз) координаты и доработать алгоритм на этапе отладки.

2 Даём программе название «Буквы». (Команды запишите самостоятельно.)

3 Программу сохраняем в файле ex22.htm.

4 Отладка логики программы — это выбор координат для перекладины буквы «А». В процессе отладки нужно изменить координаты отрезка-перекладины, запустить программу и проверить точность формы буквы. И так — несколько раз.

Изображение объёмных геометрических фигур.

■ **Упражнение 23**

Построим программу изображения на плоскости куба. Невидимые рёбра изобразим штриховыми линиями (рис. 17).

Действуем в соответствии с выбранным алгоритмом.

1 Задача построения изображения куба фактически свелась к задаче изображения плоской фигуры, только некоторые линии надо построить как штриховые.
Толщину линий выберем равной 2 пиксела.

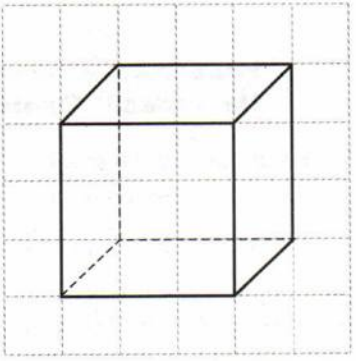


Рис. 17

2 Даём программе имя «Куб» и записываем команды, начиная со штриховых линий (они расположены на заднем плане). Сначала задаем толщину линии (2) и её стиль (штриховая — код 2):

```
Flo.width=2;
Flo.style=2;
```

Записываем команды рисования штриховых линий:

```
Flo.line(2,2,1,1);
Flo.line(2,2,2,5);
Flo.line(2,2,5,2);
```

Далее меняем стиль следа на непрерывный (код 1) и записываем остальные команды:

```
Flo.style=1;
Flo.line(1,4,1,1);
Flo.line(1,4,4,4);
Flo.line(1,4,2,5);
Flo.line(4,1,1,1);
Flo.line(4,1,4,4);
Flo.line(4,1,5,2);
Flo.Iine(5,5,5,2);
Flo.line(5,5,2,5);
Flo.line(5,5,4,4);
```

3 Программу сохраняем в файле ex23.htm.

Отладку программы проведите самостоятельно.

Изображение контура ёлочки. Цвет следа «Фломастера».

■ Упражнение 24

Построим программу для изображения контура ёлочки (рис. 18) с треугольником зелёного цвета и коричневым стволом.

Цвет следа «Фломастера» задаётся с помощью дополнительного *аргумента метода* line().

Flo.line(x1,y1,x2,y2,col) — это команда «Фломастеру» нарисовать от точки (x1,y1) до точки (x2,y2) прямую линию цвета с кодом col.

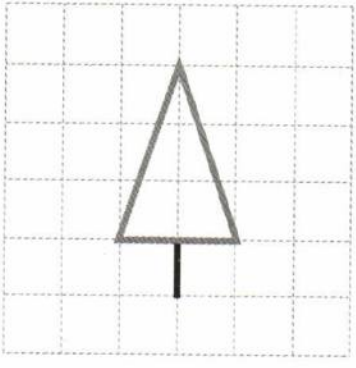


Рис. 18

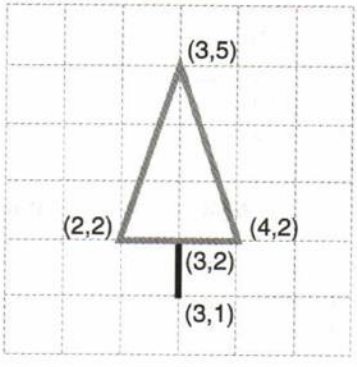


Рис. 19

Коды цветов линии следующие:

0 — чёрный;	4 — зелёный;
1 — фиолетовый;	5 — жёлтый;
2 — синий;	6 — красный;
3 — голубой;	7 — коричневый.

1. Определяем координаты концов прямых отрезков (рис. 19).

2. Даём программе название «Ёлочка». Выбираем толщину следа равной 3 пикселам. Треугольник строим «Фломастером» зелёного цвета (код 4), а ствол — коричневого (код 7).

Записываем команды:

```
Flo.width=3;
Flo.line(1,2,2,5,4);          //Линия зелёного цвета
Flo.line(1,2,3,2,4);          //Линия зелёного цвета
Flo.line(2,5,3,2,4);          //Линия зелёного цвета
Flo.line(2,1,2,2,7);          //Линия коричневого цвета
```

3. Программу сохраняем в файле ex24.htm.

Отладку программы проведите самостоятельно.

☐ Вопросы и задания

1. Перечислите этапы построения линейной программы для исполнителя «Фломастер».

2. Как можно задать цвет следа «Фломастера»?

3. Сколько кодов цвета различает исполнитель «Фломастер»?

4. Постройте программу цветного изображения:

1) букв «К» и «Ж»;
2) своих инициалов;
3) своего имени;
4) слова «алгоритм».

5. Постройте программу цветного изображения стилизованных букв (образец — файл Задание 5.ppt на CD-диске).

6. Постройте программу изображения на плоскости:

1) пирамиды (образец — файл Задание 6.1.ppt на CD-диске);
2) призмы (образец — файл Задание 6.2.ppt на CD-диске).

7. Постройте программу изображения ещё одного контура ёлочки (образец — файл Задание 7.ppt на CD-диске).

8. Постройте программу изображения розы ветров, представленной на образце в файле Задание 8.ppt на CD-диске.

9*. Постройте программы изображения правильного треугольника и правильного шестиугольника (образцы — файл Задание 9.ppt на CD-диске ).

10*. Постройте программу изображения правильной пятиконечной звезды (образец — файл Задание 10.ppt на CD-диске ).

§ 11. Понятие о технологии программирования

Технология программирования — это совокупность процессов, возникающих при разработке и сопровождении программ.

Составной частью технологии программирования является алгоритм создания программы. В предыдущих параграфах мы использовали *алгоритм создания программы* для исполнителя «Фломастер», который имел 6 этапов (предписаний). Такие этапы имеют свои названия, что позволяет записать алгоритм создания программы в следующем виде.

1. Формализация задачи.
2. Ввод текста программы.
3. Сохранение программы в файле.
4. Запуск программы на исполнение.
5. Синтаксическая отладка.
6. Отладка логики алгоритма.

Для простых задач, которые сводятся к линейным алгоритмам, первый этап был очень простым. Мы рисовали фигуры и определяли на глаз координаты узловых точек.

Этап формализации задачи. Для сложных задач этап формализации задачи может включать несколько внутренних этапов.

На *этапе формализации* задачи необходимо:

- выявить, какие объекты являются исходными данными и какие значения они могут принимать;
- выявить, каким должен быть ожидаемый результат;
- выбрать метод решения задачи;
- построить набросок алгоритма (мысленно или в словесной форме);
- провести детализацию алгоритма, используя словесную или графическую форму.

Детализация алгоритма.

Детализация алгоритма — это разбиение общих предписаний алгоритма на более конкретные.

Фактически мы использовали метод детализации алгоритма, расписав внутренние этапы для этапа формализации задачи.

Детализацию алгоритма заканчивают, когда любое предписание алгоритма в словесной или графической форме можно реализовать несколькими командами на языке программирования.

Например, набросок алгоритма математических расчётов по формуле имеет следующий вид.

1. Ввод исходных данных.
2. Проведение вычислений.
3. Вывод результатов на экран.

Каждое предписание алгоритма можно реализовать несколькими командами на языке JavaScript.

Таким образом, для простых задач алгоритмы в словесной форме сами по себе детализации не требуют. Они сразу реализуются командами языка программирования.

Вопросы и задания

1. Перечислите этапы алгоритма создания программы.
2. Какие задачи решаются на этапе формализации?
3. Приведите примеры детализации алгоритмов.

§ 12. Программы с повторениями. Цикл «пока»

Построение вертикальной штриховки. Формализация задачи.

Упражнение 25

Построим программу изображения на холсте «Фломастера» вертикальной штриховки (рис. 20).

Проведём формализацию задачи. В программе следует один за другим построить несколько вертикальных отрезков. Согласно ри-

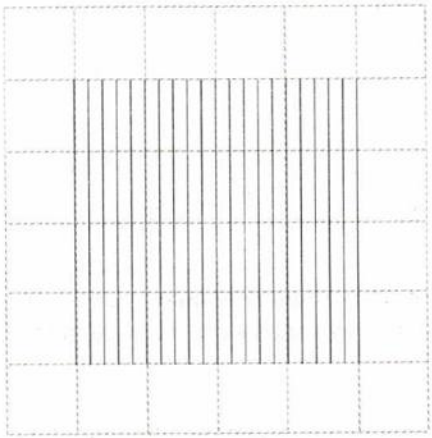


Рис. 20

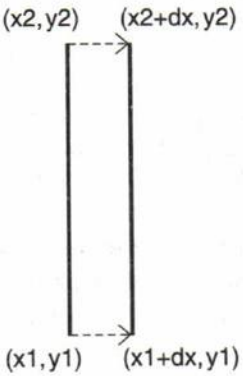


Рис. 21

сунку 20 координаты y концов отрезков не меняются, а координаты x изменяются от $x = 1$ до $x = 5$.

Рассмотрим первые два отрезка слева. Обозначим координаты концов первого отрезка в общем виде (x1,y1) и (x2,y2), а сдвиг координат вправо по x — через dx (рис. 21). Тогда координаты второго отрезка будут иметь вид (x1+dx,y1) и (x2+dx,y2).

Точно так же соотносятся между собой координаты каждой пары соседних отрезков штриховки. Следовательно, программу надо организовать по следующей схеме.

1. Ввод начальных данных.
2. Построение одной линии штриховки между точками (x1,y1) и (x2,y2).
3. Увеличение значения переменных x1 и x2 на значение переменной dx и возврат к предписанию 2 до тех пор, пока не будет построена вся штриховка.

Переходим ко второму этапу алгоритма создания программы — вводу её текста.

Записываем в «Notepad++» HTML-конструкцию программы и даём программе название «Штриховка 1». Задаём начальные значения переменных величин:

```
x1=1;  y1=2;
x2=1;  y2=6;
dx=0.2;
```

Повторение и ветвление в программах на языке JavaScript организуются с помощью специальных алгоритмических конструкций.

Алгоритмические конструкции в языке программирования — это сложные команды, которые могут включать в свой состав целые наборы команд.

Алгоритмическая конструкция повторения «пока». Повторение в программах организуется с помощью алгоритмических конструкций повторения (циклов).

Алгоритмическая конструкция повторения «пока» (цикл «пока») на языке JavaScript имеет вид:

```
while ( ) {

}
```

В круглых скобках после служебного слова while («до тех пор, пока») записывают условие. В фигурные скобки заключаются команды.

На языке JavaScript наборы команд, которые заключены в фигурные скобки, называются ***блоками***.

Блок цикла исполняется, когда выполняется условие конструкции while. После каждого исполнения блока цикла условие проверяется снова. Как только после исполнения блока условие становится невыполнимым, блок цикла не исполняется, а исполняется команда, следующая за блоком цикла.

Запись условий сравнения на языке JavaScript. На языке JavaScript условия обычно строятся с помощью **операторов сравнения**. Перечень обозначений операторов сравнения приведён в таблице 3.

Таблица 3

Оператор сравнения	Название	Пример записи условия
==	Равно	a==b
!=	Не равно	a!=b
>	Больше	a>b
>=	Больше или равно (не меньше)	a>=b
<	Меньше	a<b
<=	Меньше или равно (не больше)	a<=b

Замечание. Оператор сравнения «Равно» записывается *с помощью двух символов равенства* «==». Один символ равенства «=» — это оператор присваивания. Неправильная запись условия приведёт к неправильным результатам и может привести к «зависанию» компьютера во время исполнения программы.

■ Упражнение 25 (продолжение)

Записываем конструкцию повторения «пока» в программу после команд ввода начальных данных.

Команды блока конструкции должны повторяться до тех пор, пока выполняется условие x1⩽5. Значит, в команде цикла следует записать условие x1<=5.

В блок сразу записываем команду построения одной линии штриховки:

```
while (x1<=5) {
   Flo.line(x1,y1,x2,y2);

}
```

Заметим, что команды в блоке конструкции записываются со сдвигом вправо. Это позволяет визуально контролировать состав команд в блоке. Сдвиг первой команды блока в «Notepad++» можно выполнить с помощью клавиши **Tab** клавиатуры.

Теперь надо записать команды увеличения значений переменных x1 и x2 на значение переменной dx.

Увеличение значений переменных величин в цикле. Увеличение значения переменной величины x на значение переменной величины h в цикле производится *командой*

```
x=x+h;
```

Именно в этой команде проявляется различие между знаком равенства в математике и оператором присваивания в программировании.

Выясним, к каким результатам приводит эта команда при её повторении.

Зададим начальные значения. Пусть x=2 и h=0.5.

Исполняясь в первый раз, команда x=x+h присвоит переменной x значение 2.5 (2 + 0.5).

Исполняясь во второй раз, команда x=x+h присвоит переменной x значение 3 (2.5 + 0.5) и т. д.

Таким образом, значение переменной x при каждом исполнении команды увеличивается на значение h.

Замечание. На языке JavaScript команду x=x+h можно записать в сокращённом виде x+=h с помощью *дополнительного оператора присваивания* «+=». Аналогично записываются дополнительные операторы присваивания для других арифметических операторов.

■ Упражнение 25 (окончание)

В рассматриваемом упражнении значения переменных x1 и x2 в цикле нужно увеличивать на dx, поэтому в блок цикла записываем команды x1=x1+dx и x2=x2+dx. Получаем программу:

```
x1=1;  y1=2;
x2=1;  y2=6;
dx=0.2;
while (x1<=5) {
  Flo.line(x1,y1,x2,y2);
  x1=x1+dx;
  x2=x2+dx;
}
```

Теперь программу можно сохранить в файле ex25.htm, запустить её и проводить отладку.

Чтобы отображалась последняя линия штриховки, условие цикла «пока» можно записать в виде x1<=5.1. Заметим, что, если условие цикла записать в виде x1<=1, построится только одна линия. А если записать условие в виде x1<1, то блок цикла не исполнится ни одного раза и построений не будет.

Вопросы и задания

1. Постройте блок-схему решения задачи из упражнения 25.

2. Приведите примеры операторов сравнения.

3. Запишите на языке JavaScript следующие условия, записанные в математической форме:

1) $y_1 \geqslant y_2$; 2) $x = 3{,}5$; 3) $x_4 < y_2$; 4) $t \neq 4$.

4. Изменив начальные данные в программе «Штриховка 1», постройте изображение наклонной (под 45°) штриховки, представленное на рисунке 22.

5. Создайте программу для изображения горизонтальной штриховки на холсте «Фломастера».

§ 13. Программы с повторениями. Цикл «для»

Метод постоянных приращений при построении штриховых рисунков. Программа, построенная в упражнении 25 предыдущего параграфа, позволяет с помощью небольших переделок строить штриховые рисунки самого разного вида. В основе

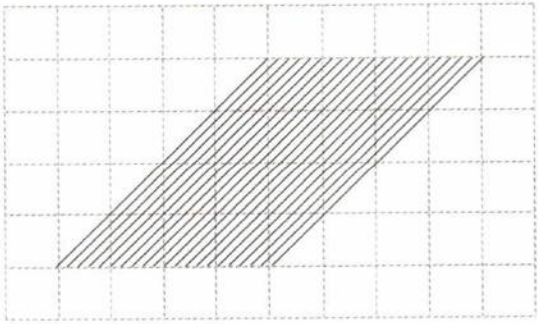


Рис. 22

этой программы лежит достаточно общий *метод постоянных приращений*, в соответствии с которым каждая из координат в цикле «пока» может получать постоянное приращение, причём это приращение может равняться нулю и даже быть отрицательным (рис. 23). При этом и поведение линий штриховых рисунков будет самым разнообразным.

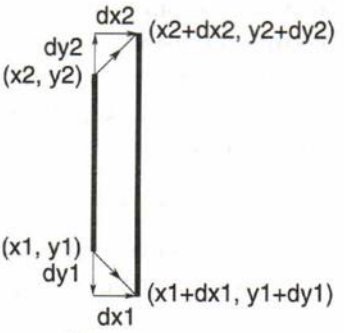


Рис. 23

■ Упражнение 26

Создадим программу построения вертикальной штриховки (см. упр. 25), в которой полностью реализован метод постоянных приращений.

Даём программе имя «Штриховка 2». Командами присваивания задаём начальные значения координатам концов отрезка и приращениям координат. Начальные значения берём из упражнения 25. Начальные значения для тех переменных, которые в нём не были заданы, возьмём нулевыми:

```
x1=1;  y1=2;
x2=1;  y2=6;
dx1=0.2;  dy1=0;
dx2=0.2;  dy2=0;
```

Будем строить 21 линию штриховки. Для организации повторений используем алгоритмическую конструкцию повторения «для».

Алгоритмическая конструкция повторения «для» (цикл «для») в языке JavaScript имеет вид:

```
for ( ; ; ) {

}
```

Цикл «для» используется для повторения блока команд конкретное количество раз.

Указание о том, сколько раз повторяется блок цикла, записывается после служебного слова for («для») в круглых скобках. При

этом обязательно вводится ***переменная цикла***, и для неё записываются три выражения.

Например, если в упражнении 26 нам нужно исполнить блок команд 21 раз, то для переменной цикла i первая строка цикла будет иметь вид:

```
for (i=1; i<=21; i++) {
```

Запись в круглых скобках означает:

1) введена переменная цикла i и для неё задано начальное значение i=1;

2) цикл будет повторяться до тех пор, пока i<=21, т. е. 21 раз;

3) значение i увеличивается на единицу каждый раз, когда исполняется блок конструкции.

Команда i++ является сокращённой записью команды присваивания i=i+1.

Команда i=i+1 называется ***командой организации счётчика.***

Счётчиком является переменная i. Именно в ней организуется подсчёт количества повторений.

Замечание. В цикле «для» можно также использовать команду организации счётчика вида i=i−1 или i−−. В этом случае в круглых скобках нужно записать выражения (i=21; i>=1; i−−), а счётчик будет давать количество оставшихся повторений. При построении штриховки результат построений будет тем же.

■ Упражнение 26 (продолжение)

Записываем в программу конструкцию цикла «для» и выражения в круглых скобках в первой строке конструкции.

В блок цикла со сдвигом вправо записываем команду рисования линии и четыре команды изменения координат:

```
x1=1;  y1=2;
x2=1;  y2=6;
dx1=0.2;  dy1=0;
dx2=0.2;  dy2=0;
for (i=1;  i<=21;  i++) {
   Flo.line(x1,y1,x2,y2);
   x1=x1+dx1;  x2=x2+dx2;
   y1=y1+dy1;  y2=y2+dy2;
}
```

Записываем программу в файл ex26.htm. В результате отладки программа «Штриховка 2» должна строить вертикальную штриховку.

Построение штриховых рисунков. Разноцветные штриховки.

■ Упражнение 27

Построим изображение штриховки треугольника (рис. 24).

В программе «Штриховка 2» эта задача решается записью нового начального значения dy2=–0.2.

■ Упражнение 28

Построим изображение разноцветной штриховки треугольника (рис. 25).

В программе «Штриховка 2» такая форма штриховки задаётся записью новых значений приращений dx1=0; dy2=–0.2.

Чтобы штриховка была разноцветной, команду построения линии запишем в виде:

```
Flo.line(x1,y1,x2,y2,i);
```

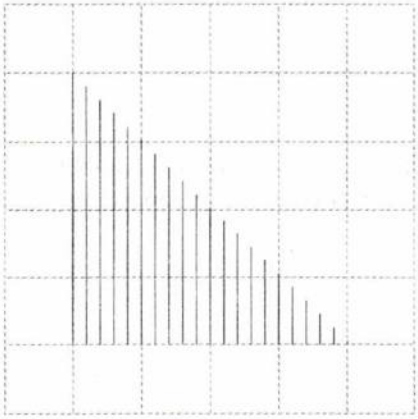


Рис. 24

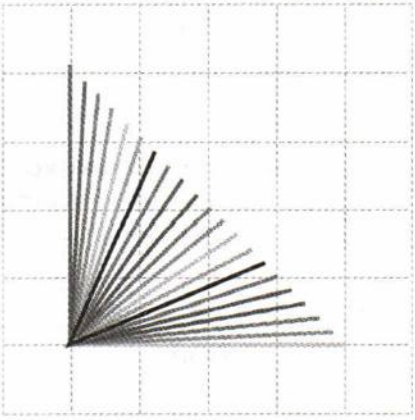


Рис. 25

Код цвета линии тогда будет меняться в цикле каждый раз, когда меняется i. Возникает вопрос о значениях i больше 7. Но тут следует знать о следующем свойстве.

Если код цвета задан целым числом больше 7, команда будет использовать код, который является остатком от деления кода на 8. Например, команда

```
Flo.line(x1,y1,x2,y2,19);
```

построит линию с кодом цвета 3 (19%8).

Для увеличения яркости рисунка в начале программы установим толщину линии равной 2 пикселам.

■ Упражнение 29

Построим изображение ещё одной цветной штриховки треугольника (рис. 26).

В этом примере исходный отрезок вырожден в точку (x1=x2, y1=y2). Координата y1 (нижнего конца отрезка) имеет отрицательное приращение (dy1=–0.2), а координата y2 (верхнего конца) — положительное (dy2=0.2). Начальные данные зададим в виде:

```
x1=1;  y1=5;
x2=1;  y2=5;
dx1=0.2;  dy1=–0.2;
dx2=0.2;  dy2=0.2;
```

Сдвиг треугольника. Метод постоянных приращений можно использовать для построения программ с изображением не только линий, но и многоугольников.

■ Упражнение 30

Построим программу построения рисунка вложенных треугольников (рис. 27).

Построим программу «Штриховка 3», дополняя программу «Штриховка 2».

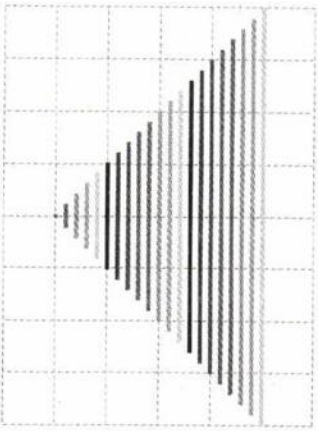


Рис. 26

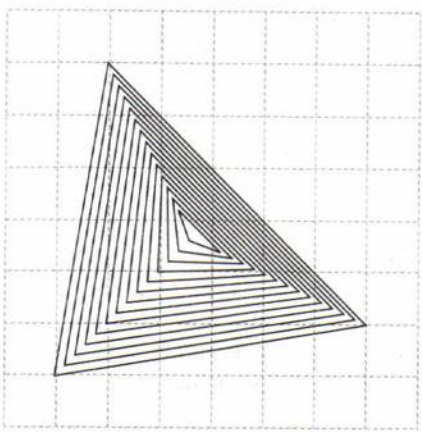


Рис. 27

Вводим дополнительные исходные данные: x3, y3 и dy3, dy3.

Координаты определяем по рисунку (третья точка вверху), а приращения задаём следующие:

```
dx1=0.2;  dy1=0.2;
dx2=-0.2;  dy2=0.1;
dx3=0.1;  dy3=-0.2;
```

Блок цикла дополняем двумя командами рисования сторон треугольника и двумя командами изменения координат:

```
x3=x3+dx3;  y3=y3+dy3;
```

Количество повторений в цикле следует уменьшить до 12—15. Сделаем штриховку разноцветной.

☐ Вопросы и задания

1. Опишите суть метода постоянных приращений при построении штриховых рисунков.

2. Опишите особенности изображения, которое получается в результате исполнения программы «Штриховка 2» с начальными данными:

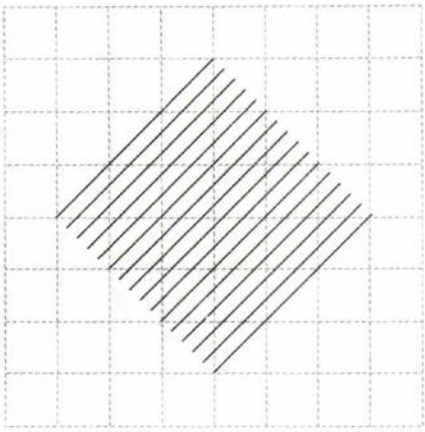


Рис. 28

1) x1=5; y1=1; x2=5; y2=7; dx1=–0.2; dy1=0.2; dx2=0.2; dy2=–0.2;

2) x1=1; y1=2; x2=1; y2=6; dx1=0.2; dy1=0; dx2=0; dy2=–0.2.

3. Как «Фломастер» строит разноцветные штриховки?

4. Постройте изображение разноцветной штриховки повёрнутого квадрата (рис. 28).

5. Постройте изображение штриховки четырёхугольника, показанного на рисунке 29.

6. Постройте штриховой рисунок ёлочки (рис. 30).

7. Постройте изображение вложенных квадратов. Внутренний квадрат является единичным, а остальные получаются из него увеличением и поворотом на небольшой угол по часовой стрелке (чем больше квадрат, тем больше угол поворота).

8. Постройте изображение вложенных ромбов (рис. 31).

9*. Создайте программу для построения радиальной штриховки (рис. 32).

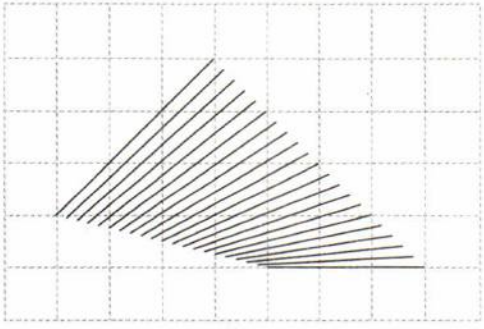


Рис. 29

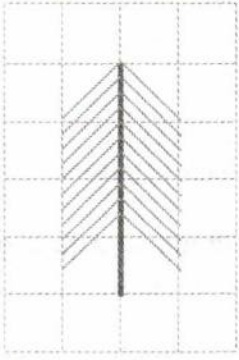


Рис. 30

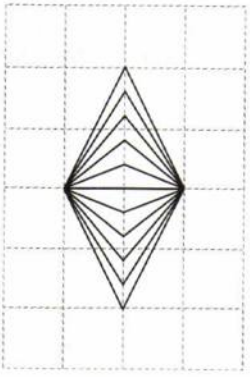


Рис. 31

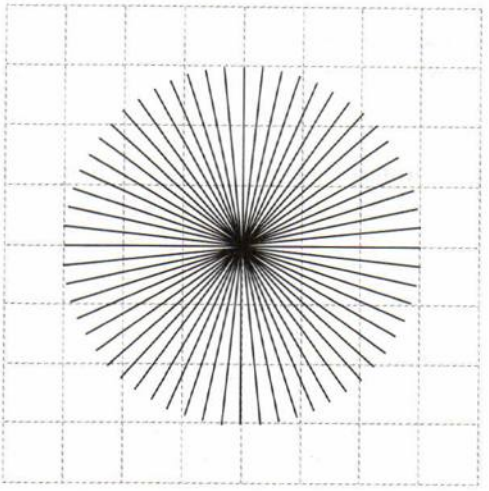


Рис. 32

§ 14. Программы с ветвлениями

Алгоритмические конструкции ветвления. Алгоритмические конструкции ветвления называют конструкциями «если».

Алгоритмическая конструкция ветвления (конструкция «если») в языке JavaScript имеет вид:

```
if ( ) {

}
else {

}
```

В круглых скобках после служебного слова if («если») записывают условие.

Конструкция содержит два блока команд. При этом:

- если условие выполняется, то исполняется только блок команд после служебного слова if;

• если условие не выполняется, то исполняется только блок команд после служебного слова else (иначе).

Алгоритмическая конструкция ветвления может быть неполной.

Алгоритмическая конструкция неполного ветвления (неполная конструкция «если») имеет вид:

```
if ( ) {

}
```

В случае выполнения условия исполняется блок команд. В случае, когда условие не выполняется, команды блока не исполняются.

Цветная штриховка с чередованием двух цветов.

■ **Упражнение 31**

Создадим программу построения штриховки, представленной на рисунке 33. Штриховка образована чередованием линий двух цветов: красного и синего.

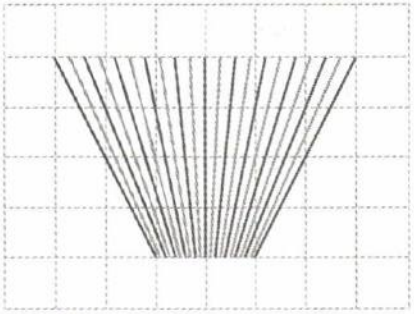


Рис. 33

Используем программу «Штриховка 2», которая позволяет строить вертикальную штриховку.

Новую программу назовём «Штриховка 4».

Командами присваивания введём начальные данные:

```
x1=3;  y1=2;
x2=1;  y2=6;
dx1=0.1;  dy1=0;
dx2=0.3;  dy2=0;
```

Команду рисования запишем в виде:

```
Flo.line(x1,y1,x2,y2,col);
```

Переменная col перед исполнением в цикле команды рисования должна поочерёдно получать значения 3 (код синего цвета) и 6 (код красного цвета), т. е. нечётные линии должны быть синего цвета, а чётные — красного.

Номер линии определяется значением переменной цикла i. Чётность и нечётность i можно оценить остатком от деления на 2. Если остаток равен 1, то номер i нечётный, если 0 — чётный.

Остаток от деления i на 2, в свою очередь, легко вычисляется с помощью арифметического выражения i%2. Поэтому условие проверки нечётности имеет вид i%2==1.

В цикл перед командой рисования записываем конструкцию ветвления:

```
if (i%2==1) {
    col=3;                          //Синий цвет
}
else {
    col=6;                          //Красный цвет
}
```

Команды в блоках конструкции записываем со сдвигом. Сохраняем программу в файле с именем ex31.htm.

Построение изображения мерной шкалы.

■ **Упражнение 32**

Построим программу изображения мерной шкалы (рис. 34).

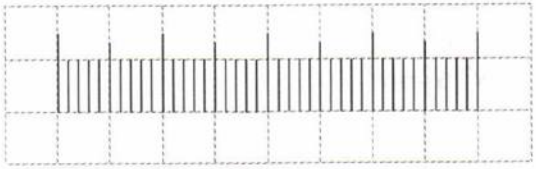


Рис. 34

Используем программу «Штриховка 2». Новую программу назовём «Мерная шкала».

Задаём в программе начальные данные для основной штриховки:

```
x1=1;  y1=1;
x2=1;  y2=2;
dx1=0.2;  dy1=0;
dx2=0.2;  dy2=0;
```

Чтобы получить 41 линию штриховки, записываем в цикле «для» условие i<=41.

Теперь надо удлинить вверх на 0.5 основные штрихи с номерами 1, 11, 21, ... и на 0.2 — основные штрихи с номерами 6, 16, 26, ...

Будем дорисовывать штрихи вверх командами:

```
Flo.line(x2,y2,x2,y2+0.5);        //Удлинение на 0.5
Flo.line(x2,y2,x2,y2+0.2);        //Удлинение на 0.2
```

В каждой последовательности номеров штрихов разница между номерами составляет 10 единиц, поэтому для построения условий будем использовать арифметический оператор нахождения остатка от деления на 10.

Для номеров штрихов 1, 11, 21, ... в цикле выполняется условие i%10==1, для номеров 6, 16, 26, ... — условие i%10==6. Соответственно используем две конструкции неполного ветвления.

Конструкция цикла «для» с включением двух конструкций неполного ветвления имеет вид:

```
for (i=1; i<=41; i++) {
    Flo.line(x1,y1,x2,y2);                //Основной штрих
    if (i%10==1) {
       Flo.line(x2,y2,x2,y2+0.5);         //Удлинение на 0.5
    }
    if (i%10==6) {
       Flo.line(x2,y2,x2,y2+0.2);         //Удлинение на 0.2
    }
    x1=x1+dx1;  x2=x2+dx2;
    y1=y1+dy1;  y2=y2+dy2;
}
```

Заметим, что здесь в конструкцию цикла «для» вложены две конструкции ветвления «если», поэтому в блоках конструкций

«если» команды при записи сдвинуты вправо дважды. Сохраняем программу в файле ex32.htm.

Построение штриховки квадрата.

■ Упражнение 33

Построим программу изображения вертикальной штриховки квадрата, повёрнутого на 45° (рис. 35).

Используем программу «Штриховка 2». Новую программу назовём «Штриховка 5».

Количество повторений (21) в цикле «для» не изменяем, а окончательно подберём его значение в процессе отладки.

Левую половину штриховки квадрата мы уже изображали в упражнении 29. Поэтому записываем в программу исходные данные из него:

```
x1=1;  y1=5;
x2=1;  y2=5;
dx1=0.2;  dy1=-0.2;
dx2=0.2;  dy2=0.2;
```

Для построения второй половины штриховки в цикле «для» начиная с какого-то номера (пусть с 17) должны поменяться знаки приращений dy1 и dy2.

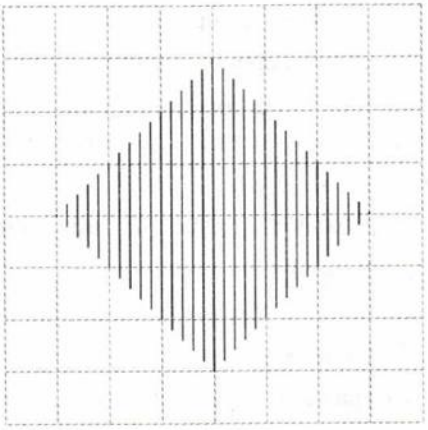


Рис. 35

Используем конструкцию неполного ветвления и запишем её в цикле после команды рисования линии, задавая dy1=0.2 и dy2=–0.2 при условии i>17:

```
for (i=1; i<=21; i++) {
   Flo.line(x1,y1,x2,y2);
   if (i>17) {
      dy1=0.2; dy2=–0.2;
   }
   x1=x1+dx1; x2=x2+dx2;
   y1=y1+dy1; y2=y2+dy2;
}
```

Сохраним программу в файле ex33.htm.

В процессе отладки логики подбираются два числа — в условии цикла «для» и в условии конструкции «если». Добавление разноцветности в штриховку проблем вызвать не должно.

☐ Вопросы и задания

1. Как в зависимости от выполнения условия конструкции «если» исполняются её блоки?

2. Как в зависимости от выполнения условия неполной конструкции «если» исполняется её блок?

3. Постройте изображение цепочки заштрихованных квадратов (образец — файл Задание 3.ppt на CD-диске ).

4. Постройте разноцветную вертикальную штриховку параллелограмма (образец — файл Задание 4.ppt на CD-диске ).

5. Постройте изображение горизонтальной штриховки шестиугольника с чередованием линий трёх цветов (образец — файл Задание 5.ppt на CD-диске ).

6. Постройте штриховки квадратов (образцы — файл Задание 6.ppt на CD-диске ).

7. Постройте штриховки сложных областей (образцы — файл Задание 7.ppt на CD-диске ).

8. Постройте изображения разноцветных штриховых дорожек (образцы — файл Задание 8.ppt на CD-диске ).

§ 15. Вспомогательные программы (подпрограммы)

При составлении алгоритмов одним из основных методов детализации алгоритма является ***метод разбиения задачи на подзадачи***, используя который в задаче выделяют внутренние частные задачи (подзадачи) и увязывают между собой их исходные данные и результаты.

Для решения всей задачи составляется программа, которая называется ***головной***.

Для решения каждой подзадачи составляют свою программу на языке программирования, которая называется ***вспомогательной программой (подпрограммой).***

Головная программа при исполнении использует результаты работы подпрограмм.

В разных языках программирования существуют разные конструкции для записи подпрограмм. В некоторых языках для них используют ту же конструкцию, что и для головной программы. В таком случае головная и вспомогательные программы записываются в разных файлах или одна за другой в одном файле.

В языке JavaScript для записи вспомогательных программ используют ***конструкцию «функция пользователя»***, которую записывают прямо в тексте головной программы. Таким образом, на языке JavaScript небольшие вспомогательные программы являются частью головной и для решения задачи составляется одна программа.

Конструкция «функция пользователя» на языке JavaScript имеет следующий вид:

```
function Имя_функции() {

}
```

Имя функции выбирается так же, как имя переменной. В частности, нельзя использовать служебные слова языка JavaScript.

Блок функции в данном случае включает команды языка JavaScript и исполнителя «Фломастер».

Если для промежуточных вычислений в блоке функции нужно ввести переменные величины, то их имена объявляются в начале блока после служебного слова var.

Например:

```
function Shift() {
   var x1,x2,y1,y2;
   x1=1;  x2=3;
   y1=x2+2;
   y2=x1-2;
   Flo(x1,y1,x2,y2);
}
```

Имена объявленных в блоке функции переменных можно произвольно использовать для других целей за пределами блока функции. За пределами данной функции они не существуют.

Команды в блоке функции пользователя записывают с уже привычным сдвигом вправо.

Для исполнения функции пользователя в программе на языке JavaScript необходимо записать *команду вызова функции*.

Команда вызова функции пользователя включает имя функции с круглыми скобками.

Например, вызов функции пользователя Shift() в программе проводится командой

```
Shift();
```

Функция пользователя в программе на языке JavaScript должна быть записана раньше команды вызова этой функции. Именно поэтому на языке JavaScript программисты обычно все свои функции записывают в самом начале программы.

Заметим, что рассмотренная функция пользователя не является функцией в привычном понимании (подобно математической функции), а представляет собой небольшую программу. Команда вызова функции пользователя просто «заменяет собой» команды, которые вошли в блок функции.

Вопросы и задания

1. Что такое вспомогательная программа?
2. Что такое головная программа?
3. Почему в языке JavaScript нет понятия головной программы?
4. В каком месте программы на языке JavaScript записывают описания функций пользователя?

§ 16. Использование подпрограмм при построении изображений

Построение изображения леса. Задача и подзадача.

Упражнение 34

Построим программу для изображения леса из ёлочек, образец которых представлен на рисунке 18.

Задача построения леса имеет естественную подзадачу: построение одной ёлочки. Если мы решили изобразить на холсте «Фломастера» семь ёлочек, то на холсте нужно выбрать семь точек и решить в каждой точке одну и ту же подзадачу (построить ёлочку).

Программу назовём «Лес». Впишем в текст программы конструкцию функции пользователя и назовём функцию elka. В блок функции запишем (скопируем через буфер обмена) текст программы «Ёлочка» из файла ex24.htm (см. упр. 24), открытого в другой вкладке «Notepad++»:

```
function elka() {
   Flo.width=3;
   Flo.line(1,2,2,5,4);
   Flo.line(1,2,3,2,4);
   Flo.line(2,5,3,2,4);
   Flo.line(2,1,2,2,7);
}
```

Далее в тексте программы записываем команду

```
elka();
```

Программу сохраним в файле с именем ex34.htm и запустим на исполнение. После отладки ёлка рисуется на холсте в совершенно определённом месте.

Метод begin(). Перенос начала координат. Если использовать только известные нам методы исполнителя «Фломастер», то для рисования других ёлок надо записать ещё шесть аналогичных функций пользователя с другими именами и другими координатами. Понятно, что никакого выигрыша от такого использования функции пользователя программист не получит. Выигрыш обеспечит метод begin() объекта Flo.

Flo.begin(xx,yy) — команда «Фломастеру» переместить начало виртуальной системы координат на холсте в точку холста с координатами (xx,yy).

Для координат (xx,yy) этого метода начало отсчёта во всех случаях расположено в нижнем левом углу холста. По этой команде в точку (xx,yy) холста перемещаются виртуальные оси координат. Координаты (xx,yy) могут задавать точку и за пределами холста.

■ Упражнение 34 (продолжение)

Команды построения одной ёлки у нас уже записаны. Чтобы нарисовать ещё шесть ёлок, будем перемещать начало координат в разные точки холста и в этих точках вызывать функцию пользователя elka().

Окончание программы получит вид:

```
Flo.begin(2,–1); elka();
Flo.begin(–1,4); elka();
Flo.begin(1,5); elka();
Flo.begin(5,0); elka();
Flo.begin(4,5); elka();
Flo.begin(7,3); elka();
```

Сохраняем программу и запускаем её. После синтаксической отладки выясняется, что на холсте отображаются оси координат. Добавим в программу последнюю команду, которая убирает оси:

```
Flo.begin(0,0);
```

Напомним, что щелчок мышью по холсту убирает клетчатую разметку. Сохраняем программу в файле с именем ex34.htm.

Метод begin(). Поворот вокруг начала координат. Метод begin() объекта Flo также даёт возможность поворачивать изображение на любой угол.

Flo.begin(xx,yy,ang) — команда «Фломастеру» переместить начало виртуальной системы координат на холсте в точку (xx,yy) холста и повернуть оси координат на угол ang в градусах вокруг нового начала координат.

В результате в новой системе координат любой рисунок строится повёрнутым на угол ang.

Например, в программе «Лес» последние три команды запишем в виде:

```
Flo.begin(7,3,90); elka();          //Поворот на 90°
Flo.begin(0,0);
```

Получим изображение, на котором последняя ёлка лежит (повёрнута на 90°).

Построение изображения снежинки.

■ Упражнение 35

Построим программу для изображения снежинки синего цвета (рис. 36).

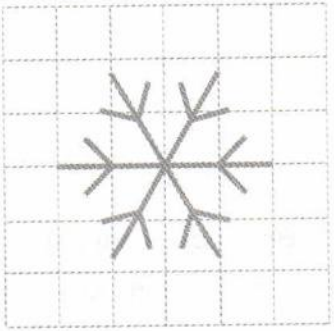


Рис. 36

Рисунок можно построить, рассчитав координаты концов всех линий. Однако мы воспользуемся возможностями метода Flo.begin(), построив только один луч снежинки (например, правый) в подпрограмме (функции пользователя) luch(). Программу назовём «Снежинка».

Для рисунков с поворотами удобнее все построения начинать из начала координат. Будем и мы луч снежинки строить из начала координат вправо:

```
function luch() {
   Flo.line(0,0,2,0,2);
   Flo.line(1,0,1.5,0.5,2);
   Flo.line(1,0,1.5,–0.5,2);
}
```

В программу «Снежинка» дописываем команды:

```
Flo.width=3;          //Толщина линий = 3
Flo.begin(5,5);       //Начало координат в точке (5,5) холста
luch();
```

Сохраняем программу в файле ex35.htm. Отладкой добиваемся правильности отображения луча снежинки.

Затем записываем команды отображения ещё пяти лучей:

```
Flo.begin(5,5,60);          //Поворот на 60°
luch();
Flo.begin(5,5,120);         //Поворот на 120°
luch();
Flo.begin(5,5,180);         //Поворот на 180°
luch();
Flo.begin(5,5,240);         //Поворот на 240°
luch();
Flo.begin(5,5,300);         //Поворот на 300°
luch();
Flo.begin(0,0);             //Убираем с холста оси координат
```

Отладкой добиваемся правильности изображения снежинки в центре холста.

☐ Вопросы и задания

1. Какая система отсчёта работает на холсте «Фломастера» при исполнении команды Flo.begin(xx,yy)?

2. Каких построений следует ожидать после записи в программе для «Фломастера» команды Flo.begin(4,4,180)?

3. Постройте изображение леса из всех видов ёлочек, построенных ранее.

4. Постройте изображение многоэтажного дома с окнами.

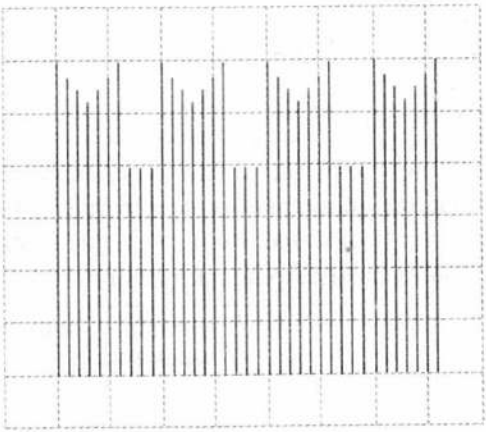


Рис. 37

5. Постройте штриховку в виде фрагмента кремлёвской стены (рис. 37).

6. Постройте изображение кирпичной кладки.

7. Постройте изображение снежинки, отличающейся по форме от снежинки из упражнения 35.

§ 17. Передача параметров в подпрограмму

В программе построения снежинки (см. упр. 35) само собой напрашивается внесение команды Flo.begin(5,5,ang) в подпрограмму luch().

Команды Flo.begin(5,5,ang) с различными значениями углов ang и команды вызова функции пользователя luch() у нас так парами в программе и записаны. Трудность состоит только в том, что значение угла поворота ang меняется и его надо передавать каждый раз из программы в подпрограмму (функцию пользователя).

Если надо передать значение параметра в подпрограмму (функцию пользователя), то для такого параметра выбирают имя переменной, записывают это имя в конструкцию (в круглых скобках после имени функции) и называют переменную *формальным параметром*.

Например:

```
function nomil(n) {          // n — формальный параметр

}
```

Далее при записи команд в блоке функции считается, что фактическое значение формального параметра n уже задано.

В подпрограмму (функцию пользователя) можно передавать значения нескольких параметров, например:

```
function mol(n,m,str) {          // n, m, str — формальные
                                 // параметры

}
```

При записи команд в блоке этой функции считается, что фактические значения формальных параметров n, m, str уже заданы.

При вызове функции пользователя в программе вместо формальных параметров записывают фактические параметры. *Фактические параметры* — это конкретные числа, строки или имена переменных из программы, которые к этому моменту получили конкретные значения. Например:

```
k=2.6;
str23="вывод";
mol(3.9, k, str23);
```

■ Упражнение 36

Модифицируем программу «Снежинка» (см. упр. 35), используя возможность передавать параметр в подпрограмму.

Загружаем в текстовый редактор «Notepad++» файл ex35.htm, меняем имя программы на «Снежинка 2» и сохраняем её в файле ex36.htm. Подпрограмму построения луча по-прежнему будем называть luch(). Добавляем в скобках параметр ang, а в блок функции дописываем команду Flo.begin(5,5 ang):

```
function luch(ang) {
   Flo.begin(5,5,ang);
   Flo.line(0,0,2,0,2);
   Flo.line(1,0,1.5,0.5,2);
   Flo.line(1,0,1.5,–0.5,2);
}
```

Для построения снежинки в программе достаточно команд:

```
Flo.width=3;
luch(0);
luch(60);
luch(120);
luch(180);
luch(240);
luch(300);
Flo.begin(0,0);          // Удаление с холста осей координат
```

Заметим, что количество команд программы можно сократить, если вместо шести команд построения лучей использовать одну конструкцию цикла «для»:

```
for (i=0; i<=5; i++) {
   luch(i*60);
}
```

Построение звёздного узора.

■ Упражнение 37

Построим узор, составленный из разноцветных звёздочек разного размера, по образцу, представленному на рисунке 27.

Трудность построения звезды заключается в необходимости вычисления или подбора координат концов отрезков.

Назовём программу «Звёздный узор». Сначала создадим функцию пользователя star(), с помощью которой «Фломастер» будет рисовать красную звезду с центром в точке (0,0):

```
function star() {
   x1=1.902;  y1=0.618;
   x2=1.176;  y2=-1.618;
   Flo.line(0,2,-x2,y2,6);
   Flo.line(x1,y1,-x2,y2,6);
   Flo.line(x1,y1,-x1,y1,6);
   Flo.line(x2,y2,-x1,y1,6);
   Flo.line(x2,y2,0,2,6);
}
```

Теперь сместим начало координат и вызовем функцию star():

```
Flo.begin(3,3);
star();
Flo.begin(0,0);                    // Удаление с холста осей координат
```

Программу сохраним в файле ex37.htm и запустим на исполнение. После отладки «Фломастер» нарисует красную звезду.

Изменим функцию star(), чтобы задавать цвет как значение параметра с именем col:

```
function star(col) {
   x1=1.902;  y1=0.618;
   x2=1.176;  y2=-1.618;
   Flo.line(0,2,-x2,y2,col);
   Flo.line(x1,y1,-x2,y2,col);
   Flo.line(x1,y1,-x1,y1,col);
   Flo.line(x2,y2,-x1,y1,col);
   Flo.line(x2,y2,0,2,col);
}
```

Увеличим толщину линий, изменим команду вызова функции:

```
Flo.width=2;
Flo.begin(3,3);
star(6);                           //Красная звезда
```

Добавим ещё несколько звёзд другого цвета и в других местах:

```
Flo.begin(7,4);
star(2);                           //Голубая звезда
Flo.begin(4,7);
star(4);                           //Зелёная звезда
Flo.begin(0,0);                    //Удаление с холста осей координат
```

Сохраним изменённый файл и запустим его на исполнение.

Изменение масштаба изображения. Чтобы нарисовать звезду другого размера, используют свойство scale объекта Flo.

Flo.scale — масштаб изображения. Свойству присваивается значение в виде любого положительного числа. По умолчанию значение свойства равно 1.

Если значение Flo.scale больше единицы, то оно показывает, во сколько раз «Фломастер» уменьшает изображение. Например, если Flo.scale=3, то изображение уменьшается в 3 раза.

Если значение Flo.scale меньше единицы, то «Фломастер» увеличивает изображение.

Продолжая выполнять упражнение 37, запишем команду выбора масштаба в блок функции star() и будем передавать значение масштаба как значение параметра sc. Программа получит вид:

```
function star(col,sc) {
   Flo.scale=sc;
   x1=1.902;  y1=0.618;
   x2=1.176;  y2=-1.618;
   Flo.line(0,2,-x2,y2,col);
   Flo.line(x1,y1,-x2,y2,col);
   Flo.line(x1,y1,-x1,y1,col);
   Flo.line(x2,y2,-x1,y1,col);
   Flo.line(x2,y2,0,2,col);
}
Flo.width=2;
Flo.begin(3,3);
star(6,3);                    //Красная звезда уменьшена в 3 раза
Flo.begin(7,6);
star(3,4);                    //Голубая звезда уменьшена в 4 раза
Flo.begin(6,3);
star(4,2);                    //Зелёная звезда уменьшена в 2 раза
Flo.begin(0,0);               //Удаление с холста осей координат
```

В команде вызова функции пользователя значения параметров записываются в том порядке, в котором они заданы в конструкции.

Сохраняем файл ex37.htm с программой и запускаем его на исполнение. Теперь можно добавить на рисунок достаточное количество звёзд. Для них можно использовать и повороты.

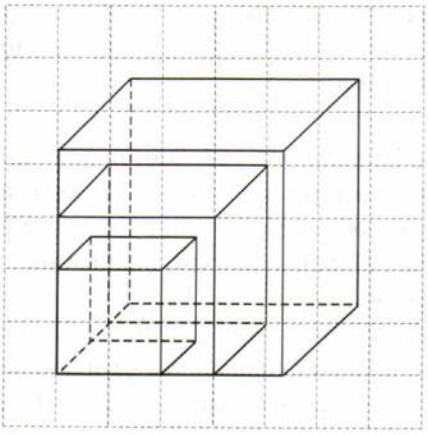


Рис. 38

Вопросы и задания

1. Как организовать передачу параметра из программы в подпрограмму?

2. Если в функцию пользователя передаётся несколько параметров, то в каком порядке значения этих параметров записываются в команде вызова функции?

3. Как изменится изображение, если свойству Flo.scale присвоить: значение 4; значение 0,5?

4. В программе «Звёздный узор» модифицируйте функцию пользователя star() так, чтобы команда Flo.begin() была записана в блок функции, а значения координат передавались в эту функцию как параметры.

5. Постройте изображение ковра из шестиугольников и звёзд разного цвета и размера.

6. Постройте изображение «летящей» звезды, организуя в цикле сдвиг изображения звезды вдоль некоторой прямой (по аналогии со штриховкой) одновременно с уменьшением размеров звезды.

7. По аналогии с заданием 6 постройте изображение «летящей» ёлки.

8*. Используя программу «Снежинка 2», постройте рисунок снегопада из разноцветных снежинок разных размеров.

9*. Постройте изображение вложенных кубов (рис. 38).

ГЛАВА 3

ПРОГРАММИРОВАНИЕ НА ЯЗЫКАХ JAVASCRIPT И PASCAL

§ 18. Вычисление сумм и произведений

Вычисление суммы положительных слагаемых.

■ Упражнение 38

Запишем программу вычисления суммы вида

$$S = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \ldots + \frac{1}{n}$$

для любого натурального числа n и вычислим значение S для $n = 15$.

Мы уже отмечали, что программы вычислений имеют чётко выраженные части:

1) задание исходных (начальных) данных;

2) вычисления;

3) вывод результатов.

Даём программе имя «Сумма 1» и будем использовать обычную HTML-конструкцию.

Переменную nn отводим для исходных данных и в начале программы задаём её значение:

```
nn=15;
```

Для суммы отводим переменную S и даём ей начальное значение 0:

```
S=0;
```

Вычислять сумму будем в цикле «для». Записываем конструкцию цикла:

```
for (n=1; n<=nn; n++) {
}
```

Условие цикла записано так, что цикл повторится ровно nn раз.

В блоке цикла запишем команду накопления суммы дробей вида 1/n:

```
S=S+1/n;
```

Заметим, что команду накопления мы уже использовали при построении штриховых рисунков.

В конце программы организуем вывод результата в окно браузера:

```
document.writeln(S);
```

В итоге программа получит вид:

```
nn=15;                          //Исходные данные
S=0;
for (n=1; n<=nn; n++) {         //Вычисления
   S=S+1/n;                     //Накопление суммы
}
document.writeln(S);            //Вывод результата
```

Сохраняем программу в файле ex38.htm.

После синтаксической отладки получаем результат:

```
3.3182289932289936
```

Далее вычисления можно проводить для разных исходных данных (значений nn).

Вычисление суммы с чередованием знаков слагаемых.

■ Упражнение 39

Запишем программу вычисления суммы вида

$$S = 1 - \frac{1}{2^2} + \frac{1}{3^2} - \frac{1}{4^2} + \ldots + (-1)^{n-1} \cdot \frac{1}{n^2}$$

для любого натурального числа n.

Знаки членов суммы чередуются. Вычислим значение S для $n = 20$.

Назовём программу «Сумма 2» и построим её, изменяя программу «Сумма 1». Изменения требуются только в названии и команде накопления суммы. Запишем эту команду в виде:

```
S=S+zn/(n*n);
```

где zn — переменная величина, которая равна +1 для нечётного n и −1 для чётного n.

Значение переменной величины zn будем задавать с помощью конструкции полного ветвления. Условие конструкции построим при помощи оператора нахождения остатка от деления (%). Условие должно отделять случай чётного n, когда n%2 даёт 0, от случая нечётного n, когда n%2 даёт 1. Аналогичные условия мы уже строили ранее.

Конструкцию ветвления записываем в блоке цикла перед командой накопления суммы:

```
nn=20;                          //Исходные данные
S=0;
for (n=1; n<=nn; n++) {         //Вычисления
   if (n%2==1) {                //Определение знака слагаемого
      zn=1;
   }
   else {
      zn=-1;
   }
   S=S+zn/(n*n);                //Накопление суммы
}
```

Вывод результата организуем с помощью команд:

```
document.writeln(nn);                    //Вывод исходных данных
document.writeln("<BR>");                //Перевод на новую строку
document.writeln(S);                     //Вывод результата
```

Сохраняем программу в файле ex39.htm.

После синтаксической отладки получаем результат:

```
20
0.8212793783297532
```

Вычисление факториала.

■ Упражнение 40

Построим программу вычисления факториала $n!$ и вычислим его при $n = 10$, т. е. 10!

Программу «Факториал» будем строить на базе программы «Сумма 1».

По определению факториала $n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$. Поэтому в цикле надо накапливать произведение. Будем его накапливать в переменной P. Перед циклом вместо команды S=0 задаём начальное значение произведения в виде:

```
P=1;
```

Команду накопления суммы заменим на команду накопления произведения в виде:

```
P=P*n;
```

Вывод организуем с помощью команд:

```
document.writeln(nn);                    //Вывод исходных данных
document.writeln("<BR>");                //Перевод на новую строку
document.writeln(P);                     //Вывод результата
```

Сохраняем программу в файле ex40.htm. Для nn=10 результат должен быть равен 3 628 800.

При больших значениях nn результат может быть выведен в другой форме. Например, для nn = 24 результат будет иметь вид

6.204484017332394e+23.

Последнее число — это строчная запись числа

$6.204484017332394 \times 10^{23}$.

Вопросы и задания

1. Какой вид имеет команда накопления суммы в цикле?

2. Какой вид имеет команда накопления произведения в цикле?

3. Составьте программу вычисления суммы для любого натурального числа n и вычислите её для $n = 12$:

1) $S = \frac{1}{2} + \frac{2}{3} + \frac{3}{4} + \ldots + \frac{n}{n+1}$; тестовая сумма 4.4071428571428575 для $n = 6$;

2) $S = \frac{1}{1 \cdot 2} + \frac{1}{2^2 \cdot 3} + \frac{1}{3^2 \cdot 4} + \ldots + \frac{1}{n^2(n+1)}$; тестовая сумма 0.6342460317460318 для $n = 6$.

Указание. Тестовые суммы используются для отладки.

4. Составьте программу вычисления суммы с чередованием знаков для любого натурального числа n и вычислите её для $n = 15$:

1) $S = \frac{2}{1} - \frac{3}{2} + \frac{4}{3} - \frac{5}{4} + \ldots + (-1)^{n-1} \cdot \frac{n+1}{n}$; тестовая сумма 0.6345238095238093 для $n = 8$;

2) $S = 1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \frac{1}{5} + \ldots + (-1)^{n-1} \cdot \frac{1}{n}$; тестовая сумма 0.7595238095238095 для $n = 7$.

5. Составьте программу вычисления суммы членов с номерами 3—7 из задания 3(1).

6. Составьте программу вычисления суммы членов с номерами 5—15 из задания 4(1).

7*. Составьте программу вычисления произведения вида $P = \left(1 + \frac{1}{n}\right)^n$ для любого натурального числа n и вычислите его для $n = 1000$. Тестовое произведение 2.5937424601000023 для $n = 10$.

8*. Для больших n произведение из задания 7 является приближённым значением числа e, которое, как и число π в математике, является известной постоянной. Организуйте в программе вывод

дополнительной строки с точным значением числа *e* (свойство Math.E объекта Math). При каком значении *n* совпадут первые пять цифр приближённого и точного значений?

§ 19. Обработка натуральных чисел

Последняя цифра натурального числа.

■ **Упражнение 41**

Задано многозначное натуральное число. Составим программу для определения последней цифры этого числа.

Будем строить программу с названием «Последняя цифра». Для нахождения последней цифры натурального числа воспользуемся арифметическим оператором нахождения остатка от деления (%). Последняя цифра натурального числа — это просто остаток от деления этого числа на 10.

В начале программы задаём исходное число, а далее следует команда нахождения остатка от деления и команда вывода исходного числа и результата:

```
nn=473;
n=nn%10;
document.writeln(nn);                    //Вывод исходных данных
document.writeln("<BR>");                //Перевод на новую строку
document.writeln(n);                     //Вывод результата
```

Сохраняем программу в файле ex41.htm и исполняем её.

Вычисление суммы цифр натурального числа.

■ **Упражнение 42**

Задано многозначное натуральное число. Составим программу вычисления суммы цифр этого числа.

Назовём программу «Сумма цифр». Вычисления построим следующим образом.

1. Считаем, что сумма цифр числа равна 0.
2. Определяем последнюю цифру числа и добавляем её к сумме цифр.

3. Делим число на 10, округляем его вниз до ближайшего целого.
4. Возвращаемся к предписанию 2. И так делаем до тех пор, пока число после округления не станет равным нулю.

Последнюю цифру числа мы будем вычислять с помощью программы из предыдущего упражнения. Для этого на её основе создадим функцию пользователя Last(). В эту функцию будем передавать значение параметра m (натуральное число):

```
function Last(m) {
   var n=m%10;
   return n;
}
```

Возвращение результата из функции пользователя в программу. Работая с исполнителем «Фломастер», мы использовали функции пользователя в качестве подпрограмм, при этом нам достаточно было передавать параметры в функцию. При построении вычислительных программ нужно не только передавать в функцию пользователя параметры, но и получать результат обратно в программу.

Именно для этого служит последняя команда в блоке функции пользователя Last(). Команда return («возвратить») присваивает значение результата имени функции. О функции пользователя в этом случае говорят, что функция возвращает значение результата. Заметим, что функция пользователя, построенная с помощью команды return, становится функцией в обычном математическом смысле этого слова.

В программе «Сумма цифр» выражение Last(m) будет рассматриваться как обычная функция, у которой значением является последняя цифра числа m. Например, значение Last(485) — это 5.

В блоке функции переменная n введена со служебным словом var. Слово var указывает, что имя переменной n работает только в пределах данного блока функции.

За пределами блока можно в любых целях использовать точно такое же обозначение для другой переменной величины, и они никак не будут связаны.

Продолжаем записывать программу «Сумма цифр». Задаём исходные данные:

```
nn=726;                          //Исходные данные
S=0;
```

Значение переменной величины nn оставим для вывода результата, а работать будем с переменной n (мы можем делать это за пределами функции Last()):

```
n=nn;                            //Вычисления
```

Далее требуется конструкция повторения «пока»:

```
while (n>0) {
   S=S+Last(n);                  //Увеличение суммы цифр
   n=Math.floor(n/10);           //Округление вниз до целого
}
document.writeln(nn);            //Вывод исходных данных
document.writeln("<BR>");        //Перевод на новую строку
document.writeln(S);             //Вывод результата
```

Сохраняем программу в файле ex42.htm и исполняем её.

Вычисление НОД двух чисел.

■ **Упражнение 43**

Составим программу вычисления наибольшего общего делителя (НОД) двух натуральных чисел.

Для этой задачи алгоритм в словесной форме был приведён в § 4 главы 1 (см. упр. 10), а блок-схема — в § 5 той же главы (см. упр. 15).

Назовём программу «НОД». Для записи программы нужно использовать конструкцию ветвления «если»:

```
nn=100;                          //Исходные данные
mm=45;
n=nn;  m=mm;                     //Вычисления
while (n!=m) {
   if (n>m) {
      n=n-m;
   }
```

```
    else {
        m=m-n;
    }
}
document.writeln(nn);                  //Вывод исходных данных
document.writeln("<BR>");              //Перевод на новую строку
document.writeln(mm);                  //Вывод исходных данных
document.writeln("<BR>");              //Перевод на новую строку
document.writeln(n);                   //Вывод результата
```

Сохраняем программу в файле ex43.htm и исполняем её.

Взаимно простые числа.

■ Упражнение 44

Задано натуральное число. Составим программу определения чисел, которые меньше заданного и взаимно простые с ним, а также подсчёта количества таких чисел.

Напомним, что два натуральных числа называются взаимно простыми, если их НОД равен 1. Будем действовать следующим образом.

1. Запишем программу «НОД» из упражнения 43 в качестве функции пользователя NOD().
2. Зададим натуральное число как значение переменной nn и выведем его на экран.
3. В переменной k организуем счётчик взаимно простых чисел. Зададим k=0.
4. В цикле «для» рассмотрим все натуральные числа от 2 до nn-1 как значения переменной i (в блок цикла входят предписания 5 и 6).
5. Для каждого числа i составим пару с числом nn и найдём НОД этой пары.
6. Если НОД = 1, то число i взаимно простое с nn. Такое число выведем на экран и увеличим на единицу счётчик: k=k+1.
7. Выведем на экран значение счётчика k.

Назовём программу «Взаимно простые». Запишем эту программу:

```
function NOD(nn,mm) {                    //Функция
   var n=nn;  m=mm;
   while (n!=m) {
      if (n>m) {
         n=n-m;
      }
      else {
         m=m-n;
      }
   }
   return n;
}
nn=15;                                   //Исходные данные
document.writeln(nn);                    //Вывод исходных данных
document.writeln("<BR>");                //Перевод на новую строку
document.writeln("<BR>");
k=0;                                     //Вычисления
for (i=2; i<nn; i++) {
   p=NOD(i,nn);
   if (p==1) {
      k=k+1;                             //Увеличение счётчика на 1
      document.writeln(i);               //Вывод числа
      document.writeln("<BR>");
   }
}
document.writeln("<BR>");
document.writeln(k);                     //Завершение вывода
```

Сохраняем программу в файле ex44.htm и исполняем её. Программа работает правильно, если для nn=15 отображаются числа: 2, 4, 7, 8, 11, 13 и 14 общим количеством 7 штук.

Вопросы и задания

1. Как в функциях пользователя на языке JavaScript организуется возврат результата в программу?

2. Составьте программу вычисления наименьшего общего кратного (НОК) двух натуральных чисел n и m $\left(\text{НОК}(n;m)=\frac{n\cdot m}{\text{НОД}(n;m)}\right)$.

3. Составьте программу, которая позволит определить, делится ли заданное натуральное число на 9.

4. Составьте программу подсчёта количества всех трёхзначных натуральных чисел, которые при зачёркивании средней цифры уменьшаются в 9 раз.

5*. Сумма цифр натурального двузначного числа равна 11. Если к этому числу прибавить 27, то получится число, записанное теми же цифрами, но в обратном порядке. Составьте программу, позволяющую определить исходное число.

§ 20. Строковые константы и строковые переменные

В языке JavaScript среди данных строкового типа различают строковые константы и строковые переменные.

Строковая константа — это набор символов (строка), который обрамлён двойными кавычками «"» или одинарными кавычками «'».

Для обрамления строковых констант обычно используют двойные кавычки. Например: "Исходные данные", "Результат".

Одинарные кавычки используют, когда сами константы содержат двойные кавычки, например: 'Выставка "Экспо-2006"'.

Строковая переменная величина (переменная величина строкового типа) — это переменная величина языка JavaScript, которая принимает значения строкового типа.

Значения строковым переменным присваиваются с помощью команды присваивания. Например:

```
str="Информатика";
num="Число";
```

Длина строки. В языке JavaScript любые строковые данные рассматриваются как копии объекта String (строка) со своими свойствами и методами.

Длиной строки называется число, равное количеству её символов.

Для вычисления длины строковых данных используется свойство length. Например, для строковой переменной str1 её

длина автоматически определяется как значение переменной str1.length.

Интересно строится имя объекта для строковой константы. Например, для константы "Информатика" объект носит такое же имя — "Информатика". Тогда длина строки "Информатика" определяется как значение переменной "Информатика".length.

Пустая строка не содержит ни одного символа, её длина равна 0.

Строковая константа " ", состоящая из одного пробела, имеет длину 1.

Оператор склейки и его свойства.

Арифметический оператор сложения "+" при работе со строковыми данными называют **оператором склейки (слияния, конкатенации)**. При помощи оператора склейки из нескольких строк можно составить (склеить) одну.

Например, в результате исполнения команды

```
m1="Привет" + "всем!";
```

переменная m1 получает значение "Приветвсем!".

Чтобы слова не сливались, нужно в одной из констант добавить пробел:

```
m1="Привет " + "всем!";
```

Теперь значением переменной величины m1 будет являться строка "Привет всем!".

Склеивать можно несколько данных — и строковые константы, и строковые переменные.

■ Упражнение 45

Составим программу склейки предложения из строк «Команда», «склейки», «строк», «строится как команда сложения чисел».

Назовём программу «Склейка». В HTML-конструкцию вписываем следующие команды:

```
g1="склейки ";
g2="строится как команда сложения чисел";
fin="Команда " + g1 + "строк " + g2;
```

Далее к этим строкам дописываем команду вывода:

```
document.writeln(fin);
```

Сохраняем программу в файле ex45.htm и исполняем её.

Напомним, что следующие команды присваивают значения разных типов:

```
dot="256";                    //Значение строкового типа
ntr=256;                      //Значение числового типа
```

В программе на языке JavaScript данные одного типа можно преобразовывать в данные другого типа.

Числовой тип данных в *текстовый тип* преобразуется автоматически при склейке строк и чисел.

Например:

```
n=15;                         //Значение n — число 15
nstr=n + " учеников";         //Значение nstr — строка "15 учеников"
```

Вывод результатов вычислений с пояснениями. Программы вычислений этой главы в качестве результата выдавали в окне браузера только числа.

Вывод можно организовать так, чтобы вместе с результатами выводились текстовые пояснения.

■ Упражнение 46

В программе «Сумма 2» (см. упр. 39, файл ex39.htm) вывод результата организуем в виде:

```
Число слагаемых = 20
Результат = 0.8212793783297532
```

Для этого нужно дополнить команду вывода исходных данных:

```
document.writeln("Число слагаемых = " + nn);
```

Здесь строковая константа "Число слагаемых = " склеена с переменной величиной nn.

Текстовое дополнение можно организовать и для команды вывода результата:

```
document.writeln("Результат = " + S);
```

Здесь строковая константа "Результат = " склеена с переменной величиной S.

Сохраним программу «Сумма 2» в файле ex46.htm и исполним её. Результаты исполнения на экране должны получить требуемый вид.

Ввод строковых данных методом prompt(). Составляя программы на языке JavaScript, мы задавали исходные данные прямо в программе при помощи команд присваивания. Это очень удобно для проведения отладки. А вот для работы с отлаженной программой обычно используют метод prompt() стандартного объекта window.

Метод prompt() позволяет вводить в программу строковые данные в процессе исполнения программы и используется как обычная функция, которая возвращает строку, введённую пользователем.

■ Упражнение 47

Составим программу вывода в окне браузера приветствия вида «Здравствуй, Виталий!» с возможностью ввода имени при исполнении программы.

Назовём программу «Приветствие». Запишем команды:

```
name=prompt("Введите свое имя","");
document.writeln("Здравствуй, "+name+"!");
```

Сохраняем программу в файле ex47.htm.

При исполнении программа, дойдя до команды присваивания с методом prompt(), останавливается, и появляется диалоговое окно «Запрос пользователю» (рис. 39). В окно с клавиатуры вводят данные (в нашем случае имя) и щёлкают по кнопке **OK**. Переменная величина name получает введённое значение строкового типа.

Рис. 39

При записи метода prompt() первый аргумент может быть любым пояснительным текстом. Второй аргумент лучше оставлять в виде пустой строки "".

Преобразование строк в числа. В программе на языке JavaScript данные строкового типа преобразуют в данные числового типа с помощью функций Number(), parseInt() и parseFloat(), которые являются методами объекта window.

Функция Number() преобразует строку в число (целое или дробное). Если в строке записано не только число или нет числа, то функция возвращает особое значение NaN (не число).

Функция parseInt() преобразует строку, которая начинается числом, в целое число, при этом она выделяет из строки первые цифры, если они есть.

Например:

```
set="384 человека";        //Значение set — строка "384 человека"
m=parseInt(set);           //Значение m — число 384
```

Если строка не начинается числом, то функция возвращает значение NaN (не число).

Функция parseFloat() преобразует строку в дробное число и работает по правилам, аналогичным функции parseInt().

Для ввода в программу числа можно использовать сложную функцию, включающую функцию Number() и метод prompt(). Например, по команде

```
mm=Number(prompt("Введите число",""));
```

значением переменной величины mm будет введённое число (данные числового типа).

■ Упражнение 48

В программе «Сумма цифр» (см. упр. 42, файл ex42.htm) сменим способ ввода данных.

Для этого команду

```
nn=726;
```

программы заменим на команду

```
nn=Number(prompt("Введите натуральное число",""));
```

Программе даём имя «Сумма цифр 2» и сохраняем её в файле ex48.htm.

Запускаем программу несколько раз, чтобы ввести разные исходные данные.

☐ Вопросы и задания

1. Какая переменная величина называется переменной величиной строкового типа?

2. Что называют строковой константой?

3. Как в программе, записанной на языке JavaScript, вычисляют длину строковых данных?

4. В чём заключается операция склейки строк в программе, записанной на языке JavaScript?

5. Как в программе на языке JavaScript данные числового типа преобразовать в строковые?

6. Добавьте пояснения при выводе результатов вычислений в программах из § 18, 19.

7. Какие функции в программе, записанной на языке JavaScript, преобразуют строковые данные в числа?

8. Для чего в программе, записанной на языке JavaScript, служит первый аргумент метода prompt()?

9. По каким правилам функция parseInt() преобразует строку в число?

10. Как можно использовать функцию parseFloat() для ввода в программу чисел с помощью метода prompt()?

11. Поменяйте способ ввода исходных данных в программе «Сумма 2» (файл ex39.htm).

12. Поменяйте способ ввода исходных данных в программе «НОД» (файл ex43.htm).

§ 21. Обработка строк

Выделение подстроки. В языке JavaScript для выделения фрагмента строки (нескольких подряд идущих символов строки) используется метод substr() объекта String (строка).

Метод имеет два аргумента и в программе используется как функция. Первый аргумент метода указывает номер позиции первого символа фрагмента (символы нумеруются, начиная с 0). Второй аргумент указывает, сколько символов фрагмент содержит.

Например, по команде

```
st1="Информация".substr(2,5);
```

значением величины st1 становится строка "форма".

■ Упражнение 49

С помощью метода substr() из букв слова «Информация» образуем слово «анимация» и выведем на экран оба этих слова и их длины.

Программу назовём «Изменение слова».

```
st="Информация";
st1=st.substr(1,1);              //"н"
st2=st.substr(5,5);              //"мация"
st3=st.substr(6,1);              //"а"
st4=st.substr(8,1);              //"и"
rez=st3 + st1 + st4 + st2;
n=st.length;
```

```
m=rez.length;
document.writeln('В слове " '+ st + ' "  ' + n + " букв");
document.writeln("<BR>");
document.writeln('В слове " '+ rez +' "  ' + m + " букв");
```

Сохраняем программу в файле ex49.htm и запускаем её на исполнение.

■ Упражнение 50

Задано значение строковой переменной. Сформируем другую строку, в которой первый и последний символы исходной строки поменялись местами.

Назовём программу «Обмен». Пусть исходная строка — str0, а строка результата — str1. Сначала с помощью метода substr() мы вычленим из исходной строки первый символ, последний символ и оставшуюся часть, а затем «склеим» новую строку.

```
str0="топор";                                  //Исходная строка
st1=str0.substr(0,1);                          //Первый символ
st2=str0.substr(str0.length-1,1);              //Последний символ
st3=str0.substr(1,str0.length-2);              //Середина
str1=st2 + st3 + st1;                          //Результат
document.writeln("Исходная строка = " + str0); //Вывод
document.writeln("<BR>");
document.writeln("Результат = " + str1);
```

Сохраняем программу в файле ex50.htm и запускаем её на исполнение.

■ Упражнение 51

Задано значение строковой переменной. Сформируем другую строку, в которой символы записаны в обратном порядке.

Если, например, исходным значением строки является «краб», то в результате нужно получить значение «барк». Решение может быть таким. Определяем последний символ строки. Записываем его первым в новую пустую строку. Затем определяем предпослед-

ний символ исходной строки и дописываем его вторым в новую строку и т. д.

Назовём программу «Обратный порядок». Пусть исходная строка — str0, а строка результата — str1. При этом изначально строка результата пустая.

Формировать значение строки str1 мы будем в цикле с количеством повторений str0.length. Вспомним только, что нумерация символов строки str0 начинается с 0 и заканчивается номером str0.length–1, а мы будем вести отсчёт от str0.length–1 до 0 в обратном порядке.

```
str0="краб";                          //Исходные данные
str1="";                              //Начальное значение результата
for (i=str0.length-1;i>=0;i--) {
   str1=str1 + str0.substr(i,1);      //Накопление результата
}
document.writeln("Исходная строка = " + str0); //Вывод
document.writeln("<BR>");
document.writeln("Результат = " + str1);
```

Программу сохраняем в файле ex51.htm и запускаем её на исполнение.

Сменим способ ввода в программе. Вместо команды

```
str0="краб";                          //Исходные данные
```

запишем

```
str0=prompt("Введите строку символов","");
```

Программу сохраним в том же файле.

Сравнение строковых данных. Строковые данные можно сравнивать. Например, о двух строковых переменных или константах можно сказать, что они равны или не равны. При записи условий сравнения используются операторы сравнения «==» (равно) и «!=» (не равно). Например:

```
str=="",  a!="синий".
```

Упражнение 52

Составим программу, позволяющую определить, сколько раз буква «а» встречается в заданном значении строковой переменной.

Назовём программу «Число вхождений буквы».

Для решения задачи нужно создать счётчик, рассмотреть каждый символ значения заданной переменной и сравнить его с буквой «а». Если рассматриваемый символ равен букве «а», то значение счётчика нужно увеличить на 1.

Просмотр всех символов переменной организуем с помощью конструкции повторения «для», где количество повторений — это длина заданной строки. Её мы определим с помощью свойства length. Запишем программу:

```
str="абракадабра";                    //Исходные данные
k=0;                                  //Начальное значение счётчика
for (i=0; i<str.length; i++) {
    p=str.substr(i,1);                //Выбор очередного символа
    if (p=="a") {                     //Неполное ветвление
        k=k+1;
    }
}
document.writeln("Число вхождений буквы а = " + k);          //Вывод
```

Программу сохраним в файле ex52.htm. После отладки результат отобразится в окне браузера.

Изменим способ ввода исходных данных. Сохраним программу в том же файле и проверим её работу на других исходных данных. Если в исходной строке буква «а» отсутствует, то при выводе число вхождений должно равняться 0.

Упражнение 53

Задано значение строковой переменной. Составим программу, позволяющую удалить все пробелы.

Порядок решения задачи выберем следующий. Для результата отведём переменную, начальное значение которой — пустая строка. В цикле будем рассматривать по одному символы исходной строки и, если это не пробел, дописывать их в переменную результата.

Назовём программу «Уплотнение». Пусть исходная строка — str0, а строка результата — str1. Как и в упражнении 51, изначально строка результата пустая.

Формировать значение строки str1 будем в цикле с числом повторений str0.length (от i=0 до i=str0.length-1). Команда формирования новой строки str1 очень похожа на команду накопления суммы, но в случае строк эта команда накапливает строку результата.

```
str0="т  е х но л о гия";          //Исходные данные
str1="";                           //Начальное значение результата
for (i=0; i<str0.length; i++) {
   st=str0.substr(i,1);
   if (st!=" ") {
      str1=str1 + st;              //Накопление результата
   }
}
document.writeln("Исходная строка = " + str0); //Вывод
document.writeln("<BR>");
document.writeln("Результат = " + str1);
```

Программу сохраним в файле ex53.htm. После отладки можно поменять способ ввода.

■ Упражнение 54

В качестве строки рассматривается предложение, которое начинается и заканчивается символами, отличающимися от пробела. Знак «тире» в предложении отсутствует. Составим программу подсчёта количества слов в предложении.

Заметим, что в предложении между словами может быть пробел или знак препинания и пробел. Количество слов в таком предложении равно количеству пробелов плюс один. Таким образом, чтобы подсчитать количество слов, достаточно подсчитать число пробелов, организовав счётчик.

Назовём программу «Количество слов».

```
str0="Слышен, слышен конский топот!";   //Исходные данные
k=1;                                    //Начальное значение результата
```

```
for (i=0; i<str0.length; i++) {
    r=str0.substr(i,1);
    if (r==" ") {
        k = k + 1;                          //Накопление результата
    }
}
document.writeln(str0);                     //Вывод
document.writeln("<BR>");
document.writeln("Число слов = " + k);
```

Программу сохраним в файле ex54.htm. После отладки можно поменять способ ввода исходных данных.

☐ Вопросы и задания

1. Определите все возможные значения переменной k, если k=ms.substr(i,5–i) и ms="курок".

2. Составьте программу, в которую методом prompt() вводится слово и которая выводит на экран исходное слово и слово, составленное из половинок исходного, записанных в обратном порядке.

3. Составьте программу, которая выводит на экран столбец из всевозможных фрагментов по три символа подряд, взятых из слова «детализация».

4. Составьте программу, которая выводит на экран столбец из двух заданных слов, причём первым отображается слово с меньшим числом символов.

5. Какие операторы позволяют образовывать условия сравнения строковых данных?

6. Какой командой в цикле организуется накопление строкового результата?

7. Задана строка. Составьте программу для:

1) подсчёта числа встречающихся сочетаний «со»;

2) замены окончаний слов «ова» на «ава»;

3) замены слова «Марино» на «Сан-Марино»;

4) замены первой буквы слова «футбол» на прописную.

8. Задана строка. Составьте программу, позволяющую определить, является ли её значение «перевёртышем». *Перевёртышами* (*палиндромами*) называют слова, которые читаются одинаково слева направо и справа налево. Например: «топот», «шалаш».

9*. Задана строка. Составьте программу, определяющую, встречаются ли в ней:

1) два одинаковых символа;

2) два идущих подряд одинаковых символа.

10*. Задано предложение. Выведите все числа, если они есть в предложении, а затем выведите их сумму.

§ 22. Логические значения, выражения, операции

Элементарные логические выражения. При составлении программ с повторениями и ветвлениями мы использовали условия сравнения для данных числового и строкового типа. Например:

```
st!=" "    p==1    p=="a"    n>m
```

Условие сравнения может принимать только два значения:

true (истинно) — условие выполняется; условие верно, справедливо;

false (ложно) — условие не выполняется; условие неверно, несправедливо.

Логическое значение — одно из двух значений: true или false.

Логическое выражение — это выражение, которое может принимать только логические значения.

Условия сравнения являются логическими выражениями. Из-за простоты их обычно называют ***элементарными логическими выражениями.***

■ Упражнение 55

Составим программу для решения квадратного уравнения $ax^2 + bx + c = 0$ в случае $a \neq 0$.

Алгоритм в словесной форме для решения этой задачи рассмотрен в § 4 главы 1 (см. упр. 11). Этот алгоритм включает проверку условия сравнения (элементарного логического выражения).

Назовём программу «Решение квадратного уравнения». Зададим некоторые значения коэффициентов.

```
a=2;                                    //Исходные данные
b=4;
c=1;
D=b*b-4*a*c;                            //Вычисление дискриминанта
if (D>=0) {
   x1=(-b+Math.sqrt(D))/2/a;
   x2=(-b-Math.sqrt(D))/2/a;
   rez="Корни " + x1 + " и " + x2;      //Вариант ответа
}
else {
   rez="Нет корней";                    //Вариант ответа
}
document.writeln(rez);                  //Вывод ответа
```

Сохраним программу в файле ex55.htm. После синтаксической отладки следует проверить правильность работы программы на разных наборах исходных данных.

Логические переменные.

Переменная величина логического типа, логическая переменная величина — переменная величина, которая принимает логические значения.

Переменная логического типа тоже является логическим выражением.

На языке JavaScript любая переменная величина становится переменной логического типа, если ей присвоить логическое значение.

■ Упражнение 56

Составим программу присвоения логических значений переменным величинам lz1 и lz2 и вывода этих значений на экран.

Назовём программу «Логика 1».

```
p="к";                                  //Исходные данные
lz1=(p=="a");
lz2=true;
document.writeln(lz1);                  //Вывод
document.writeln("<BR>");
document.writeln(lz2);
```

Программу сохраним в файле ex56.htm и исполним её.

Логические операции.

Логические операции — это операции «НЕ», «И», «ИЛИ», которые применяют к логическим выражениям.

Логическая операция «НЕ» (логическое отрицание) меняет значение логического выражения на противоположное.

На языке JavaScript логическая операция «НЕ» описывается оператором ! (логический оператор «НЕ»).

Оператор записывается прямо перед именем переменной или перед выражением.

Пример: !lz1.

■ Упражнение 57

Проверим действие логического оператора «НЕ», используя его при выводе значений величин lz1 и lz2 из упражнения 56.

В программе «Логика 1» в командах вывода поменяем записи lz1 на !lz1 и lz2 на !lz2. Новую программу назовём «Логика 2» и сохраним её в файле ex57.htm. При её исполнении ранее выводимые на экран значения должны поменяться на противоположные.

Логическая операция «И» связывает два логических выражения в сложное логическое выражение, которое верно только тогда, когда верны оба входящих в него выражения.

На языке JavaScript логическая операция «И» описывается оператором && (логический оператор «И»).

■ Упражнение 58

Выведем на экран таблицу значений логического выражения lz1&&lz2 в зависимости от значений входящих в него логических переменных.

Назовём программу «Логика 3». Имеется 4 возможных варианта сочетаний разных значений переменных величин lz1 и lz2.

В цикле значение переменной lz1 будем менять каждый раз командой lz1=!lz2, а значение переменной lz2 поменяем после первых двух повторений цикла с помощью конструкции «если».

```
lz1=true;                                    //Исходные данные
lz2=true;
document.writeln("lz1 " + "lz2 " + "&&");    //Заголовок таблицы
document.writeln("<BR>");
for (i=1; i<=4; i++) {
   if (i>2) {
      lz2=false;
   }
   document.writeln(lz1);                    //Вывод строк таблицы
   document.writeln(lz2);
   document.writeln(lz1&&lz2);
   document.writeln("<BR>");
   lz1=!lz1;
}
```

Сохраним программу в файле ex58.htm и исполним её.

Логическая операция «ИЛИ» связывает два логических выражения в сложное логическое выражение, которое верно в случаях, когда верно хотя бы одно из входящих в него выражений.

На языке JavaScript логическая операция «ИЛИ» описывается оператором || (логический оператор «ИЛИ»).

■ Упражнение 59

Выведем на экран таблицу значений логического выражения lz1||lz2 в зависимости от значений входящих в него логических переменных.

В программе «Логика 3» надо поменять в двух командах логические операторы && («И») на || («ИЛИ»). Программу назовём «Логика 4» и сохраним в файле ex59.htm.

■ Упражнение 60

Составим программу, позволяющую определить, сколько раз буква «а» (как строчная, так и прописная) встречается в заданном значении строковой переменной.

В § 21 мы составили программу «Число вхождений буквы» (см. упр. 52, файл ex52.htm). Но программа отслеживает только строчную букву «а». Чтобы программа делала подсчёт и строчных, и прописных букв, нужно в конструкции ветвления элементарное

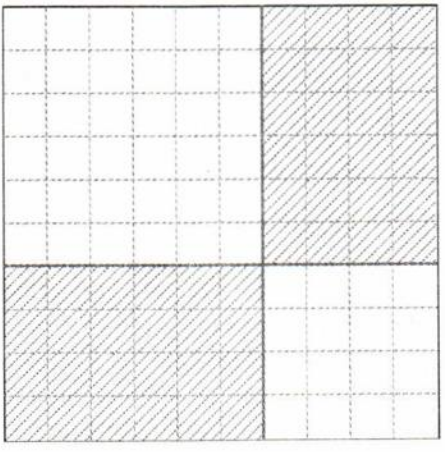


Рис. 40

логическое выражение p=="a" заменить на сложное выражение с логическим оператором «ИЛИ» (обозначение ||) и круглыми скобками (p=="a")||(p=="A").

Доработанную программу назовём «Число вхождений буквы 2» и сохраним в файле ex60.htm.

■ Упражнение 61

На холсте «Фломастера» построены две прямые линии $x = 6$ и $y = 4$ (рис. 40). Случайным образом называются два числа от 0 до 10 — координаты некоторой точки на холсте. Составим программу, которая определяет, принадлежит ли точка одной из заштрихованных областей (не включая их границы).

Назовём программу «Принадлежность». Начнём с записи логического выражения, которое будет верно только в случае принадлежности точки с координатами (x, y) одной из заштрихованных областей.

Верхняя заштрихованная область описывается двумя условиями: $x > 6$, $y > 4$, которые должны выполняться одновременно. Это приводит к логическому выражению

(x>6)&&(y>4)

для верхней области.

Аналогично для нижней заштрихованной области имеем условия: $x < 6$ и $y < 4$, что приводит к сложному выражению

(x<6)&&(y<4).

Но это ещё не всё.

Принадлежность к одной из областей — это выполнение одного условия или другого (логическая операция «ИЛИ»). Значит, в итоге получается сложное логическое выражение

```
((x>6)&&(y>4))||((x<6)&&(y<4))
```

Чтобы получить случайные координаты исходной точки, воспользуемся специальными возможностями языка JavaScript. Этот язык позволяет использовать в программах числа, которые называются *псевдослучайными*, но мало чем отличаются от случайных. Для их получения применяется метод random() объекта Math (математика).

Метод Math.random(), как функция, возвращает случайное число между 0 и 1. Для наших целей нужно случайное число из диапазона между числами 0 и 10. Такое случайное число даёт арифметическое выражение 10*Math.random().

Составляем программу «Принадлежность»:

```
x=10*Math.random();
y=10*Math.random();
if (((x>6)&&(y>4))||((x<6)&&(y<4))) {
   str="";
}
else {
   str="не";
}
document.writeln("Точка ("+x+", "+y+")"
                        +str+" принадлежит заштрихованной области ");
```

Сохраним программу в файле ex61.htm. Каждый новый запуск в браузере даёт координаты новой точки.

Таким образом, в программе «Принадлежность» мы использовали ещё один (третий) *способ ввода исходных данных* в программу — ввод случайного числа *методом* Math.random().

☐ Вопросы и задания

1. Какие операторы языка JavaScript позволяют образовывать элементарные логические выражения?

2. Измените способ ввода коэффициентов уравнения в программе «Решение квадратного уравнения» на ввод методом prompt().

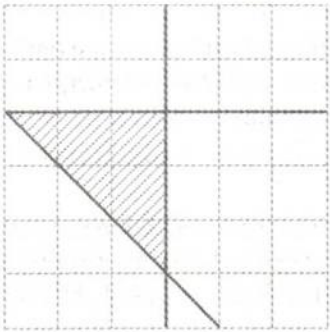


Рис. 41

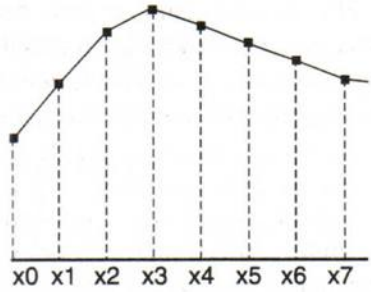


Рис. 42

3. Решите при помощи программы «Решение квадратного уравнения» следующие уравнения:

1) $x^2 - 5x + 6 = 0$; 3) $x^2 - 6x + 9 = 0$;

2) $4x^2 - 2x + 15 = 0$; 4) $3x^2 - 4x + 8 = 0$.

4*. Составьте программу вычисления корней квадратного уравнения с возможным значением $a = 0$.

5*. Составьте программу вычисления корней уравнения, коэффициенты которого задаются с помощью метода `Math.random()`.

6. Какие операторы языка JavaScript реализуют логические операции?

7. Назовите и охарактеризуйте три способа ввода числовых данных в программу на языке JavaScript.

8. Задана строка. Составьте программу подсчёта числа встречающихся в ней сочетаний «ко» и «Ко».

9*. На холсте «Фломастера» построены три прямые линии: $x = 3$, $y = 4$ и $y = 4 - x$ (рис. 41). Случайным образом называются два числа от 0 до 10, которые являются координатами точки на холсте. Составьте программу, которая определяет, принадлежит ли точка заштрихованному треугольнику.

§ 23. Построение графиков функций

Метод построения. С графиками простейших функций (линейной, квадратичной) вы знакомы из курса алгебры. Попробуем строить графики функций программными средствами.

Для построения графика функции $y = f(x)$ воспользуемся следующим методом. Пусть график данной функции надо построить на отрезке $[x_0, x_E]$ значений переменной x. На этом отрезке мы рассматриваем значения $x_0, x_1, x_2, \ldots$, которые отстоят друг от друга на одно и то же число dx (шаг), и вычисляем соответствующие значения функции: $f(x_0), f(x_1), f(x_2), \ldots$.

На координатной плоскости отмечаем точки с координатами $(x_0, f(x_0))$, $(x_1, f(x_1))$, $(x_2, f(x_2))$, ... Затем соединяем прямыми отрезками соседние по индексам точки. Получается ломаная линия (рис. 42).

Если значение величины dx достаточно мало, то и ломаная линия имеет вид гладкой линии. Её можно рассматривать как график функции $y = f(x)$ на отрезке $[x_0, x_E]$. Примерно так график функции можно строить на миллиметровой бумаге.

Для построения графика функции на экране компьютера воспользуемся возможностями исполнителя «Фломастер».

■ Упражнение 62

Создадим программу построения графика функции $y = x^2 + x - 4$ на отрезке $[-3, 2]$ исполнителем «Фломастер».

Назовём программу «График 1». Исходными данными для построения графика являются значения переменных x0, xE и dx. Будем считать dx = 0.1 и записываем в HTML-конструкцию для «Фломастера» команды:

```
x0= -3;                    //Начальные данные
xE= 2;
dx= 0.1;                   //Шаг
```

Метод Flo.begin() позволяет переносить начало координат в любую точку холста «Фломастера». Командой Flo.begin(5,5) перенесём начало системы координат в середину холста:

```
Flo.begin(5,5);
```

Строить график будем в цикле «для» с переменной x, начальным значением x0, конечным значением xE+dx (для учёта возможных

погрешностей) и приращением dx. В блоке цикла вычисляем значение функции и присваиваем его переменной y:

```
for (x=x0; x<xE+dx; x=x+dx) {
    y=x*x+x-4;                          //Вычисление значения y

}
```

Для построения «Фломастером» ломаной нужны координаты концов её элементарных отрезков. В цикле по формулам вычисляются координаты (x,y). Будем считать их координатами конца очередного элементарного отрезка. Координаты начала элементарного отрезка обозначим (xs,ys). Строить отрезки будем командой

```
Flo.line(xs,ys,x,y);
```

После построения отрезка в цикле координаты (x,y) будем запоминать как координаты (xs,ys) начала следующего отрезка командами

```
xs=x; ys=y;                 //Сохранение вычисленных координат
```

В предложенной схеме есть только одно исключение. При первом исполнении цикла координаты (x,y) вычисляются как координаты начала первого элементарного отрезка.

В этом исключительном случае построение элементарного отрезка мы делать не будем, для чего воспользуемся конструкцией ветвления:

```
if (x!=x0) {
    Flo.line(xs,ys,x,y);
}
```

В итоге получаем следующую программу:

```
x0=-3;                              //Начальные данные
xE=2;
dx=0.1;                             //Шаг
```

```
Flo.begin(5,5);
for (x=x0; x<xE+dx; x=x+dx) {
    y=x*x+x-4;                    //Вычисление значения y
    if (x!=x0) {
        Flo.line(xs,ys,x,y);      //Построение отрезка
    }
    xs=x; ys=y;                   //Сохранение вычисленных координат
}
```

Сохраняем программу в файле ex62.htm и исполняем её.

Рассмотренный метод подходит для построения графиков любых, даже очень сложных функций. Главное, нужно суметь записать выражение $f(x)$ средствами языка JavaScript и выбрать такой отрезок $[x_0, x_E]$, на котором значения выражения для $f(x)$ можно вычислить. К примеру, график функции $y = \sqrt{x}$ на отрезке [–2, 2] построить не получится, так как для отрицательных значений x корень не вычисляется.

Графики тригонометрических функций*. Из геометрии вы знаете, что такое синус и косинус острого угла в прямоугольном треугольнике. В старших классах вы подробно познакомитесь с функциями $y = \sin x$ и $y = \cos x$, которые называются *тригонометрическими функциями* и могут вычисляться для любых значений аргумента x вне зависимости от вида треугольника.

Отсутствие знакомства с тригонометрическими функциями не помешает нам построить графики этих функций, поскольку стандартный объект Math (математика) языка JavaScript имеет методы-функции:

Math.sin() — синус аргумента;
Math.cos() — косинус аргумента.

Вот только в геометрии углы вами измерялись в *градусах*, а в информатике их обычно измеряют в *радианах*. Связь между величиной α угла в радианах и величиной β того же угла в градусах даёт простая формула:

$$\alpha \text{ (радианы)} = 0{,}017453293 \cdot \beta \text{ (градусы)}.$$

На «Калькуляторе» вида «Инженерный» справа под индикатором есть кнопки выбора единицы изменения углов: градусы, ради-

аны, грады (ещё одна единица измерения углов). На виртуальной клавиатуре «Калькулятора» есть и кнопки для вычисления тригонометрических функций.

■ Упражнение 63

Создадим программу построения графика тригонометрической функции $y = \sin x$ на отрезке [–4, 4]. Такой график называется синусоидой.

Назовём программу «Синусоида» и будем строить её, изменяя программу «График 1».

Изменения потребуются самые минимальные. Меняем название программы и значения начальных данных (x0, xE). Затем меняем выражение для вычисления значения y:

```
y=Math.sin(x);                    //Вычисление значения y
```

Сохраняем программу в файле ex63.htm и исполняем её.

После построения графика становится понятным, почему в физике тригонометрические функции используются для описания колебательных процессов.

Графики параметрических функций*. Функции вида $y = f(x)$ описывают зависимость переменной y от переменной x в явной форме.

Однако существуют и другие виды функций, например параметрические функции, в которых такая зависимость явно не описывается.

Параметрические функции задаются двумя формулами

$$x = f_1(t), \quad y = f_2(t),$$

где значения третьей переменной t принадлежат некоторому отрезку $[t_0, t_E]$. Переменная величина t называется *параметром*.

Каждому значению t соответствуют значения x и y; при этом считают, что y зависит от x. Соответственно строится и график функции. Тонкость построения в том, что точками t_0, t_1, t_2, ... разбивается отрезок $[t_0, t_E]$, потом вычисляются значения $x_0 = f_1(t_0)$, $x_1 = f_1(t_1)$, ... и $y_0 = f_2(t_0)$, $y_1 = f_2(t_1)$, ... А затем полученные точки графика (x_0, y_0), (x_1, y_1), ... строятся так же, как это делается для обычной функции вида $y = f(x)$.

Упражнение 64

Создадим программу построения графика параметрической функции $x = t^3$, $y = t^2$ на отрезке [−1,6, 1,6] изменения параметра t. Такой график называется параболой Нейля.

Построим программу «График 2» на основе программы «График 1». Прежде всего меняем обозначения и значения исходных данных, а также обозначение переменной цикла. Затем добавляем команду вычисления значения x.

```
t0=-1.6;
tE=1.6;
dt=0.1;
Flo.begin(5,5);
for (t=t0; t<tE+dt; t=t+dt) {
    x=t*t*t;                    //Вычисление значения x
    y=t*t;                      // Вычисление значения y
    if (t!=t0) {
        Flo.line(xs,ys,x,y);    //Построение отрезка
    }
    xs=x; ys=y;                 //Сохранение координат
}
```

Сохраняем программу в файле ex64.htm и исполняем её.

Для некоторых параметрических функций построенные графики становятся более гладкими, если уменьшить величину шага до $dt = 0.01$.

Упражнение 65

Создадим программу построения графика параметрической функции $x = 2(1 + \cos t)\cos t$, $y = 2(1 + \cos t)\sin t$ на отрезке [−3,2, 3,2]. Такой график называется кардиоидой.

Используем программу «График 2». Новую программу назовём «Кардиоида».

Изменим начальные значения для t_0, t_E. Поменяем также выражения для вычисления x и y:

```
x=2*(1 + Math.cos(t))*Math.cos(t);
y=2*(1 + Math.cos(t))*Math.sin(t);
```

Сохраняем программу в файле ex65.htm и исполняем её. Если уменьшить шаг до dt=0.01, то график становится заметно более гладким.

❑ Вопросы и задания

1. В чём состоит метод построения графика функции с помощью компьютера?

2. Если на построенном графике функции заметны углы ломаной линии, то что следует сделать для увеличения гладкости графика?

3. Чем параметрические функции отличаются от обычных?

4. Постройте на экране компьютера график функции:

1) $y = x^2 - 4$ на отрезке $[-3, 3]$;

2) $y = x^2 - 4x + 1$ на отрезке $[-1, 5]$;

3) $y = \frac{3}{x}$ на отрезке $[1, 5]$;

4) $y = \sqrt{x}$ на отрезке $[0, 10]$;

5) $y = \frac{1}{1 + x^2}$ на отрезке $[-4, 4]$ (локон Аньези).

5*. Постройте на экране компьютера графики параметрических функций:

1) $x = 3\cos t$, $y = 2\sin t$ на отрезке $[-3{,}2, 3{,}2]$ (эллипс);

2) $x = 4\cos t$, $y = 3\sin 3t$ на отрезке $[-3{,}2, 3{,}2]$ (фигура Лиссажу).

6*. Постройте на экране компьютера графики параметрических функций:

1) $x = 4\cos^3 t$, $y = 4\sin^3 t$ на отрезке $[-3{,}2, 3{,}2]$ (астроида);

2) $x = 4|\sin 2t|\cos t$, $y = 4|\sin 2t|\sin t$ на отрезке $[0, 6{,}3]$ (четырёхлепестковая роза).

7*. Постройте и проанализируйте набор графиков параметрической функции $x = 4\cos at$, $y = 4\sin bt$ на отрезке $[-3{,}2, 3{,}2]$ значений параметра t, построенных при различных целых значениях коэффициентов a и b из отрезка $[1, 7]$. Сколько всего различных графиков должно получиться?

8*. Постройте график параметрической функции $x = 4\sin at \times \cos bt$, $y = 4\sin at \cdot \sin ct$ на отрезке $[-3{,}2, 3{,}2]$ значений параметра t при различных целых значениях коэффициентов a, b и c из отрезка $[1, 5]$. Попробуйте установить зависимость формы графика от значений коэффициентов.

§ 24. Линейные массивы

Массив — это упорядоченный набор переменных величин, объединённых общим именем.

Элемент массива — это переменная величина, которая входит в массив.

Массив — линейный (одномерный), если все его элементы пронумерованы.

Имя массива в языке JavaScript строится по правилам имён для переменных. Например: mas, b2, u.

Имя элемента линейного массива складывается из имени массива и индекса (порядкового номера в массиве). В языке JavaScript индекс элемента записывают в квадратных скобках после имени, а нумерация элементов массива начинается с 0. Например, элементы массива mas имеют имена mas[0], mas[1] и т. д.

Массивы бывают не только линейными, но и *многомерными*. В многомерных массивах порядок элементов задаётся другими способами.

Создание линейных массивов. В языке JavaScript имеется один стандартный объект Array (массив), который является линейным массивом. Массив Array является массивом-образцом.

Другие массивы в программах создаются как копии объекта Array. Например, массив с именем mas и числом элементов 20 создаёт следующая команда:

```
mas = new Array(20);
```

Элементы массива можно использовать в программах как обычные переменные величины.

■ Упражнение 66

Составим программу, которая по заданным катетам треугольника вычисляет его гипотенузу.

Назовём программу «Прямоугольный треугольник». Вернёмся к обычной HTML-конструкции программы на языке JavaScript.

Введём в программе массив из трёх элементов и будем использовать элементы этого массива для записи длин катетов и гипотенузы.

```
tr=new Array(3);
tr[0]=3;                                    //Первый катет
tr[1]=4;                                    //Второй катет
tr[2]=Math.sqrt(tr[0]*tr[0]+tr[1]*tr[1]);   //Гипотенуза
document.writeln("Гипотенуза = " + tr[2]);  //Вывод
```

Сохраняем программу в файле ex66.htm и исполняем её.

Задать массив и начальные значения его элементов в JavaScript можно и другим способом.

Например, массив mas с четырьмя элементами и их начальными значениями создаёт следующая команда:

```
mas=[2, 3.4, 5, –2.9];
```

Начальные значения элементов массива mas в этой команде разделены запятыми.

Линейные числовые массивы. Массивы в программах используют для обработки больших объёмов данных с помощью алгоритмических конструкций повторения.

■ Упражнение 67

Задана последовательность из 11 чисел:

5, 8, –6, 3, 1, –7, 4, –5, –3, 2, –4.

Составим программу вычисления суммы n первых чисел последовательности для любого n от 2 до 10.

Назовём программу «Сумма массива». Введём в этой программе соответствующий массив.

Суммирование будем вести в цикле «для» с помощью команды накопления суммы, которая имеет легко угадываемый вид.

Напоминаем, что индексы (номера) элементов массива начинаются с 0.

Для контроля выведем на экран элементы исходного массива. Вывод элементов массива в одну строку (через запятую) обеспечи-

вает метод document.writeln() с именем массива в качестве аргумента.

```
dat=[5, 8, –6, 3, 1, –7, 4, –5, –3, 2, –4];
nn=5;                                  //Число чисел в сумме
S=0;                                   //Начальное значение суммы
for (i=0; i<nn; i++) {
   S=S + dat[i];                       //Команда накопления суммы
}
document.writeln(dat);                 //Вывод массива
document.writeln("<BR>");              //Отделяем первый вывод от второго
document.writeln("Сумма = " + S); //Вывод результата
```

Сохраняем программу в файле ex67.htm и исполняем её.

При необходимости в цикле организуют проверку значений элементов массива, подсчёт элементов с определёнными свойствами.

■ Упражнение 68

Составим программу подсчёта количества положительных чисел среди первых n чисел последовательности для любого n от 2 до 10 с исходными данными из упражнения 67.

Назовём программу «Количество положительных». Построим её на базе программы «Сумма».

Количество проверяемых чисел задаём как значение переменной nn (где $nn \leqslant 11$).

Для подсчёта положительных чисел организуем счётчик в переменной k с начальным значением k=0.

В цикле запишем конструкцию «если» с проверкой условия положительности. В случае выполнения условия значение счётчика увеличиваем на 1.

```
dat=[5, 8, –6, 3, 1, –7, 4, –5, –3, 2, –4];
nn=5;                    //Количество проверяемых чисел
k=0;                     //Начальное значение счётчика
for (i=0; i<nn; i++) {
   if (dat[i]>0) {       //Проверка положительности
      k=k+1;             //Команда увеличения счётчика
   }
}
```

```
document.writeln(dat);                    //Вывод
document.writeln("<BR>");
document.writeln("Положительных чисел " + k);
```

Сохраняем программу в файле ex68.htm и исполняем её.

■ Упражнение 69

Составим программу поиска наибольшего числа в последовательности и определения его номера с исходными данными из упражнения 67.

Назовём программу «Наибольшее число». Построим её на базе программы «Количество положительных». Для хранения номера наибольшего числа оставляем переменную k с начальным значением k=0. Для значения наибольшего числа вводим переменную max с начальным значением max=dat[0]. Тогда в цикле можно брать i=1 как начальное значение переменной цикла.

```
dat=[5, 8, –6, 3, 1, –7, 4, –5, –3, 2, –4];
nn=10;                    //Кол-во проверяемых чисел
k=0;                      //Начальное значение номера
max=dat[0];               //Начальное значение наибольшего числа
for (i=1; i<nn; i++) {
   if (dat[i]>max) {
      k=i;                //Запись номера числа
      max=dat[i];         //Новое наибольшее число
   }
}
document.writeln(dat);                    //Вывод
document.writeln("<BR>");
document.writeln("Наибольшее число " + max ↵[1]
+ " с номером " + k);
```

Сохраняем программу в файле ex69.htm и исполняем её; она выдаёт номер наибольшего числа как номер индекса в массиве. Можно изменить программу так, чтобы номер числа был порядковым (нумерация начиналась с 1).

[1] Знак переноса. Перенос строки команды допускается только при записи на бумажном носителе. При введении программы для исполнения строка команды должна быть непрерывной.

Линейные массивы строк. Язык JavaScript позволяет работать с массивами строк. Но нужно учитывать, что строковые данные имеют свои способы обработки.

■ Упражнение 70

Заданы 7 слов: кувшинка, мак, лилия, одуванчик, роза, василёк, фиалка. Составим программу поиска слова с наибольшим числом букв.

Программу «Длинное слово» построим на базе программы «Наибольшее число» (см. упр. 69). В условии сравнения в конструкции ветвления надо использовать свойство length для строковых элементов массива.

```
dat=["кувшинка", "мак", "лилия", "одуванчик", ↵
"роза", "василёк", "фиалка"];
nn=7;                          //Число проверяемых слов
max=0;                         //Начальное значение длины слова
rez="";                        //Начальное значение результата
for (i=0; i<nn; i++) {
   if (dat[i].length>max) {
      rez=dat[i];              //Запись длинного слова
      max=dat[i].length;       //Новая наибольшая длина
   }
}
document.writeln(dat);         //Вывод
document.writeln("<BR>");
document.writeln("Самое длинное слово " ↵
+ rez+ " имеет " + max + " букв");
```

Сохраняем программу в файле ex70.htm и исполняем её.

■ Упражнение 71

Составим программу поиска слова, которое начинается с буквы «р», и определения номера этого слова в заданной последовательности. Используем исходные данные упражнения 70.

Программу «Слово на букву» будем строить на базе предыдущей программы.

```
dat=["кувшинка", "мак", "лилия", "одуванчик", ↵
"роза", "василёк", "фиалка"];
nn=7;                          //Число проверяемых слов
k=0;                           //Начальное значение номера слова
rez="";                        //Начальное значение результата
for (i=0; i<nn; i++) {
   if (dat[i].substr(0,1)=="p") {
      rez=dat[i];              //Запись слова
      k=i;                     //Запись номера слова
   }
}
document.writeln(dat);         //Вывод
document.writeln("<BR>");
document.writeln('Слово "' + rez + ' " имеет номер ' + k);
```

Сохраняем программу в файле ex71.htm и исполняем её.

Напоминаем, что нумерация слов в массиве начинается с 0.

Изменение порядка следования элементов в массиве. Объект Array (массив) мы рассматривали как образец линейного массива. В то же время объект Array и его копии имеют несколько свойств и методов, которые недоступны в других языках программирования.

Число элементов массива хранит *свойство* length (длина массива). Как обычно, в программах имя свойства начинается с имени массива, например: mas.length, dat.length.

Изменять порядок следования элементов в линейных массивах позволяют *методы* sort() (сортировка) и reverse() (обратный порядок), которые в программах используются как команды.

Начнём с обработки массива строк.

■ Упражнение 72

Создадим программу, которая меняет порядок следования названий дней недели на обратный и выстраивает названия в алфавитном порядке. Выведем на экран названия дней одним столбцом в исходном порядке, в обратном и в алфавитном.

Назовём программу «Дни недели». Вывод в столбик будем организовывать в цикле, используя функцию пользователя OutV().

```
dat=["понедельник", "вторник", "среда", "четверг", ↵
"пятница", "суббота", "воскресенье"];
function_OutV(mas) {
   for (i=0; i<mas.length; i++) {
      document.writeln(mas[i]);
      document.writeln("<BR>");
   }
   document.writeln("<BR>");
}
OutV(dat);                    //Вывод в исходном порядке
dat.reverse();                //Изменяем порядок на обратный
OutV(dat);                    //Вывод
dat.sort();                   //Изменяем порядок на алфавитный
OutV(dat);                    //Вывод
```

Сохраняем программу в файле ex72.htm и исполняем её.

Заметим, что методы sort() и reverse() меняют исходный массив (меняют индексы элементов). Метод sort() хорошо работает с массивами строк, так как упорядочивает элементы в соответствии с возрастанием кодов символов в кодировочной таблице.

Для работы с числами в скобках после имени метода записывают имя функции сравнения, которая имеет два параметра x и y. При определении функции считается, что x — это элемент массива, который идёт раньше элемента y, а действие функции определяется её отрицательными значениями. Например, следующая функция пользователя задаёт в массиве порядок возрастания числовых значений:

```
function comp(x,y) {
   return x-y;                    // x - y < 0, т. е. x < y
}
```

Выражение $y - x$ в блоке функции будет задавать порядок убывания значений массива ($y - x < 0$, т. е. $x > y$).

■ Упражнение 73

Создадим программу, которая выстраивает элементы заданного числового массива в порядке возрастания значений. Выведем на экран исходный массив и результат в две строки.

Назовём программу «Возрастание».

```
function comp(x,y) {
   return x-y;
}
dat=[5, 8, -6, 3, 1, -7, 4, -5, -3, 2, -4];
document.writeln(dat);          //Вывод в исходном порядке
document.writeln( "<BR>");
dat.sort(comp);                 //Изменяем порядок
document.writeln(dat);          //Вывод
```

Сохраняем программу в файле ex73.htm и исполняем её.

Вопросы и задания

1. По каким правилам образуются имена массивов?

2. По каким правилам образуются имена элементов линейных массивов?

3. Как создать в программе новый массив?

4. Какие имена имеют элементы массива, который определён командой a=new Array(10)?

5. Какими способами можно задать начальные значения элементам массива?

6*. В программе «Количество положительных» (файл ex68.htm) увеличьте количество исходных чисел до 50. Значения чисел должны быть целыми, из диапазона от −10 до 10.

Указание. Для задания значений элементов массива используйте в отдельном цикле команду

```
dat[i]=Math.round(20*Math.random())-10;
```

7. Задан числовой массив. Составьте программы для:

1) присвоения значения 0 всем элементам массива;

2) присвоения значений элементов исходного массива элементам другого массива;

3) изменения знаков всех чисел массива на противоположные;

4) изменения знаков отрицательных чисел массива на противоположные;

5) вычисления суммы положительных чисел;

6) поиска наименьшего из чисел и его индекса;

7) вычисления разности между наибольшим и наименьшим числами в массиве;

8)* поиска номера первого отрицательного элемента в массиве;

9)* поиска в массиве наибольшего числа, а если их несколько, то подсчёта их количества.

8. Задан строковый массив. Составьте программы для:

1) подсчёта количества слов, которые оканчиваются на «ка»;

2) поиска в массиве слова наименьшей длины;

3) поиска в массиве слов определённой длины;

4)* подсчёта числа вхождений во все слова буквы «а»;

5)* поиска в массиве слов, которые содержат букву «в»;

6)* поиска в массиве слов-перевёртышей;

7)* поиска слова, в котором чаще всего встречается буква «к».

9. Задан строковый массив. Составьте программу, с помощью которой можно:

1) расположить элементы массива в порядке, обратном алфавитному;

2) упорядочить элементы массива по убыванию их длин;

3)* расположить элементы массива в алфавитном порядке последних букв.

§ 25. Динамические массивы. Стеки. Списки

Динамические массивы. Массивы в языке JavaScript, в отличие от многих других языков программирования, являются динамическими.

Динамический массив — это массив, количество элементов которого может меняться при исполнении программы.

В частности, в языке JavaScript можно создать массив, который не содержит ни одного элемента. Объявление «пустого» массива делается командой

```
dop = new Array();
```

В этой команде после имени объекта Array количество элементов нового массива dop просто не указано.

■ Упражнение 74

Создадим два числовых массива, один из которых пустой, и выведем на экран их длины.

Назовём программу «Длина массива».

```
dop = new Array();
dat=[5, 8, -6, 3, 1, -7, 4, -5, -3, 2, -4];
document.writeln(dop.length);
document.writeln("<BR>");
document.writeln(dat.length);
```

Сохраняем программу в файле ex74.htm и исполняем её. Программа выдаёт в качестве результата числа 0 и 11.

Стеки.

Стек — это динамический массив, для которого возможны только две операции:

- удалить из массива последний элемент;
- добавить новый элемент в конец массива.

В языке JavaScript указанные операции обеспечивают *методы* pop() и push() объекта Array (массив). Заметим, что данные методы поддерживает интерпретатор браузера Internet Explorer версии 5.5 и старше. Приведём описание этих методов:

pop() — удалить последний элемент массива и возвратить его значение;

push(mm) — добавить значение переменной mm в конец массива и возвратить новую длину массива.

Методы pop() и push() являются примерами методов, которые можно использовать как в качестве обычных функций (они возвращают значения), так и в качестве команд (если нет потребности в возвращаемых значениях).

■ Упражнение 75

В конец пустого стека добавим три элемента и удалим один.

Назовём программу «Стек». Методы pop() и push() используем как команды.

```
dop = new Array ();
dop.push(2);  dop.push(-5);  dop.push(14);
```

```
document.writeln(dop);                    //Вывод
document.writeln("<BR>");
dop.pop();
document.writeln(dop);                    //Вывод
```

Сохраняем программу в файле ex75.htm и исполняем её.

Если возвращаемые значения методов pop() и push() нужны в программе, то эти методы используют как функции в командах присваивания. Например, допишем в нашу программу строки:

```
mm=dop.pop();
document.writeln("<BR>");
document.writeln(mm);                     //Вывод
```

Рассмотренные методы дают возможность работать со стеками на языке JavaScript, что нередко помогает при решении задач.

■ Упражнение 76

Задан числовой стек. Создадим два новых стека, в которых собраны соответственно положительные и неположительные числа из исходного стека.

Назовём программу «Положительные и неположительные».

```
dat=[5, 8, −6, 3, 1, −7, 4, 0, −3, 2, −4];
poz=new Array();
neg=new Array();
document.writeln(dat);                    //Вывод исходного стека
while (dat.length>0) {
   mm=dat.pop();
   if (dat[i]>0) {
      poz.push(mm);
   }
   else {
      neg.push(mm);
   }
}
document.writeln("<BR>");
document.writeln(poz);                    //Вывод положительных
document.writeln("<BR>");
document.writeln(neg);                    //Вывод неположительных
```

Сохраняем программу в файле ex76.htm и исполняем её.

Списки.

Линейный список (список) — это динамический массив, для которого возможны только две операции:
- удалить любой элемент массива;
- добавить новый элемент в любое место массива.

В языке JavaScript указанные операции обеспечивают *методы* splice() и push(). Данные методы поддерживает интерпретатор браузера Internet Explorer версии 5.5 и старше. Метод push() был уже рассмотрен нами ранее. Приведём описание метода splice(), который обычно используют как команду:

splice(i,1) — удалить элемент массива с индексом i и возвратить значение удалённого элемента;

splice(i,0,mm) — добавить значение переменной mm перед элементом с индексом i.

■ Упражнение 77

В заданном числовом списке удалим пятый элемент, затем удалим первый элемент и добавим новый элемент –32 в изменённый список перед восьмым элементом.

Назовём программу «Линейный список». Напомним, что индексы элементов массива начинаются с 0.

```
dat=[5, 8, –6, 3, 1, –7, 4, 0, –3, 2, –4];
document.writeln(dat);              //Вывод исходного списка
document.writeln("<BR>");
dat.splice(4,1);                    //4, начиная с 0, это 5-й элемент
dat.splice(0,1);                    //0, это 1-й элемент
document.writeln(dat);              //Вывод после удаления
document.writeln("<BR>");
dat.splice(7,0,–32);                //7, начиная с 0, это 8-й элемент
document.writeln(dat);              //Вывод результата
```

Сохраняем программу в файле ex77.htm и исполняем её.

Метод splice() можно также использовать как функцию.

■ Упражнение 78

Задан набор чисел. Переставим в заданном наборе значения первого и последнего элементов, пользуясь только операциями работы со списками.

Программу назовём «Перестановка».

```
dat=[5, 8, –6, 3, 1, –7, 4, 0, –3, 2, –4];
document.writeln(dat);              // Вывод исходного списка
document.writeln("<BR>");
mm=dat.splice(0,1);                 // Удаление и сохранение первого
nn=dat.splice(dat.length–1,1);      // Удаление и
                                    //  сохранение последнего
dat.splice(0,0,nn);                 // Вставка нового первого
dat.push(mm);                       // Вставка нового последнего
document.writeln(dat);              // Вывод после изменения
```

Сохраняем программу в файле ex78.htm и исполняем её.

☐ Вопросы и задания

1. Что такое динамический массив?

2. Сколько элементов имеет массив, который определён командой a = new Array()?

3. Что такое стек?

4. Что такое линейный список?

5. Задан числовой массив (стек) с целыми числами. Составьте программу, с помощью которой можно: 1) создать из чисел исходного массива два новых массива (стека), один с положительными числами, другой с отрицательными, и вывести на экран все три массива; 2) создать из чисел исходного массива два новых массива (стека), один с чётными числами, другой с нечётными, и вывести на экран все три массива; 3) создать из чисел массива новый массив (стек), в котором положительные и отрицательные числа из первого массива сначала чередуются, а затем идут те, которых было больше, и вывести оба массива на экран; 4)* создать из чисел массива новый массив (стек), в который вошли только положительные чётные числа, и вывести оба массива на экран.

6. Заданы два числовых массива (стека) одинаковой длины с целыми числами. Составьте программу, с помощью которой можно: 1) все числа из первого массива переписать во второй массив и вывести на экран три массива (два исходных и результат); 2) создать третий массив (стек), каждый элемент которого содержит произведение соответствующих элементов исходных массивов, и вывести на экран все три массива; 3) все числа из первого стека переписывать во второй стек до тех пор, пока значение последнего элемента первого

стека не окажется чётным, и вывести на экран четыре стека (два исходных и два изменённых).

7. Задан числовой массив (список) с целыми числами. Составьте программу, с помощью которой можно: 1) продублировать в массиве первый и последний элементы (вставить перед теми, что уже есть, точно такие же) и вывести на экран два массива (исходный и результат); 2) создать массив (список), в котором записаны все элементы исходного списка с нечётными индексами, и вывести на экран оба массива; 3) создать массив (список), в котором записаны все элементы исходного списка с чётными значениями, и вывести на экран оба массива; 4) создать массив (список), в котором записаны и продублированы все элементы исходного списка с чётными индексами, и вывести на экран оба массива; 5) создать массив (список), в котором записаны и продублированы все элементы исходного списка с нечётными значениями, и вывести на экран оба массива; 6)* создать массив (список), в котором сначала идут элементы исходного массива с чётными значениями, а потом с нечётными, и вывести на экран оба массива; 7)* создать массив (список), в котором сначала идут элементы исходного массива с нечётными индексами, а потом с чётными, и вывести на экран оба массива; 8)* если указаны два индекса i и j (i<j), создать новый массив (список), в котором сначала идут элементы исходного массива с индексами от i до j, затем элементы с индексами менее i (если такие есть) и элементы с индексами более j (если такие есть), а затем вывести на экран оба массива.

§ 26. Знакомство с языком программирования Pascal

Исторический очерк. Язык программирования Pascal был создан в 1968 г. под руководством профессора Никлауса Вирта в Швейцарском федеральном технологическом институте ETH (Eidgenoessische Technische Hochschule). В 1970 г. под его руководством был написан и первый транслятор для этого языка — ETH Pascal.

Язык назван в честь великого французского математика, физика и философа Блеза Паскаля, который в 1642 г. изобрёл счётную машину для арифметических операций. Язык Pascal считают наследником языка программирования Algol, популярного в 1960-е гг.

По утверждению самого Н. Вирта, язык Pascal был разработан в качестве языка для обучения. Пик его популярности пришёлся на 1980-е гг. Но до сих пор Pascal используют как для обучения программированию, так и для разработки прикладных программ.

Самую большую роль в массовом распространении языка Pascal сыграла компания Borland International, которая в ноябре 1983 г. выпустила в свет знаменитую интегрированную среду разработки (систему программирования) Turbo Pascal. Версия языка получила такое же название. Система Turbo Pascal, начиная с версии 7.0, была переименована в Borland Pascal. Так же назвали и версию языка.

В 1995 г. компанией Borland была создана интегрированная среда разработки Delphi. А вот версия языка для этой системы программирования стала называться Object Pascal или Delphi Object Pascal. Но начиная с версии Delphi 7.0 язык стали называть просто Delphi.

Все перечисленные системы программирования для языка Pascal были коммерческими. Но со временем были разработаны и свободно распространяемые системы программирования, среди которых выделим Free Pascal, PascalABC, PascalABC.NET.

Для знакомства с языком Pascal мы будем использовать систему программирования PascalABC.NET, которая разработана в России на факультете математики, механики и компьютерных наук Южного федерального университета. Версия языка для этой системы также носит название PascalABC.NET и имеет высокую совместимость с версиями Borland Pascal и Delphi. Система PascalABC.NET может быть свободно загружена с сайта разработчиков.

Описание окна системы и подготовка к работе. После запуска система PascalABC.NET открывается в стандартном окне на рабочем столе ОС Windows (рис. 43).

В окне системы выделяют *рабочее окно* и *служебные зоны*. В рабочем окне (в середине по высоте окна Windows) вводят и редактируют текст программ. Служебные зоны содержат строку меню и панель инструментов (в верхней части окна), нижнюю панель и строку состояния (в нижней части окна). Нижнюю панель можно убирать с экрана (или возвращать на экран) сочетанием клавиш **F5** либо щелчком указателя мыши по крайней правой кнопке на панели инструментов.

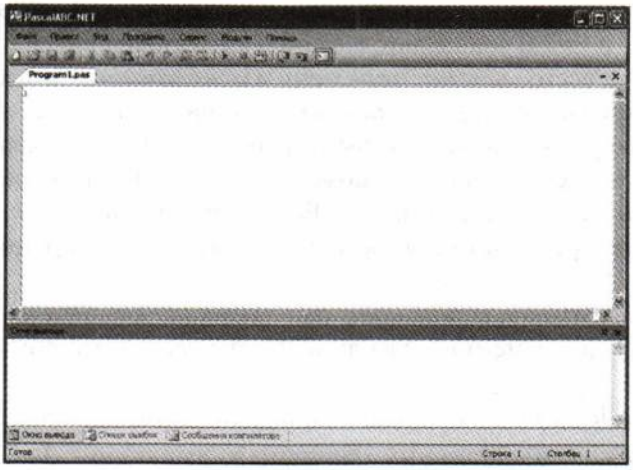


Рис. 43

Нижняя панель имеет три вкладки «Окно вывода», «Список ошибок» и «Сообщения компилятора», которые активизируются щелчком указателя мыши.

Вкладка «Окно вывода» предназначена для отображения числовых и текстовых результатов исполнения программы. Содержание других вкладок определяется их названиями.

Внимание! Готовясь к составлению программ на языке Pascal, следует создать на рабочем столе ярлык для файла Help_PABC.htm. Этот файл будет скопирован на диск C: в составе папки JS с прилагаемого CD-диска. Файл содержит документ «Справка PascalABC.NET», в котором описаны конструкции и команды языка PascalABC.NET.

Сходство и различия языков JavaScript и Pascal. Язык JavaScript, с которым мы уже познакомились, является типичным наследником языка программирования C («си»), который был разработан в начале 1970-х гг.

Языки Pascal и C в 1980-е гг. часто противопоставляли друг другу. Но даже один из авторов языка C, Деннис Ритчи, как-то сказал: «Я утверждаю, что Pascal очень близок языку C. Одни, быть может, этому удивятся, другие — нет... Даже интересно, насколько они близки друг другу. Эти языки больше расходятся в деталях, но в основе своей одинаковы...»

Попытаемся на конкретных примерах проследить, в чём языки JavaScript и Pascal совпадают, а в чём расходятся.

Прежде всего, *имена переменным величинам* в рассматриваемых языках задаются ***одинаково***: в виде слова без пробелов (допускаются латинские буквы, цифры и символ подчёркивания «_»), но с цифры имя начинаться не может. Правда, небольшое ***отличие*** всё же есть: в языке Pascal *строчные и прописные буквы не различаются*: beta, Beta и BETA — это одно и то же имя.

В обоих языках ***одинаково*** одну команду от другой отделяют знаком «;», а комментарии в одной строке располагают правее знака «//».

Совпадают в обоих языках *правила записи чисел и арифметических выражений*. Например, выражение $34{,}7 - x_1^2 + \frac{5y_1}{x_2 - 3{,}4}$ записывается как `34.7 - x1*x1 + 5*y1/(x2 - 3.4)`.

■ Упражнение 79

С помощью браузера Internet Explorer запустим файлы `Help_JS.htm` и `Help_PABC.htm` и разместим их окна на левой и правой половинах экрана без перекрытий.

Выведем на экран одновременно и просмотрим объекты, приведённые в таблице 4.

Таблица 4

JavaScript	Pascal
HTML-конструкции	Конструкции программы
Конструкции повторения	Конструкции повторения
Конструкции ветвления	Конструкции ветвления
Функции пользователя	Процедуры и функции

После беглого сравнения конструкций становится ясно, что внешне язык JavaScript ***отличает*** *использование фигурных скобок* «{» и «}» *для выделения блоков команд* (это наследство языка C). А вот Pascal для тех же целей унаследовал от языка Algol зарезервированные (служебные) слова **begin** и **end**, которые принято называть *операторными скобками*. Но суть использования операторных скобок от этого не изменилась.

Не всегда в рассматриваемых языках совпадает терминология. На языке JavaScript программа состоит из *команд*. В языке Pascal команды программы обычно называют *операторами*.

Чем, конечно, ***различаются*** рассматриваемые языки, так это *именами стандартных* (*встроенных* или *системных*) *функций*, *методов* и *констант* (табл. 5).

Таблица 5

Описание	JavaScript	Pascal
Значение константы π	Math.PI	PI
Значение константы *e*	Math.E	E
Корень квадратный	Math.sqrt()	Sqrt()
Абсолютная величина числа	Math.abs()	Abs()
Число *a* в степени *b*	Math.pow(a,b)	Power(a,b)
Округление до целого числа	Math.round()	Round()
Округление до целого числа в сторону уменьшения	Math.floor()	Floor()
Округление до целого числа в сторону увеличения	Math.ceil()	Ceil()
Вывод текстовых и числовых данных	document. writeln()	writeln()

Существенно ***по-разному*** строится в рассматриваемых языках *команда* (*оператор*) *присваивания*. В языке JavaScript используется знак равенства «=», а в языке Pascal — сложный знак «:=» (двоеточие, равно). Но привыкнуть к этим различиям несложно.

■ Упражнение 80

Сравним следующие два фрагмента программ и выясним, на каком языке (JavaScript или Pascal) написан каждый из них:

```
a = 2.3;    //фрагмент 1
b = -1.6;
z = a + 8.2*b;
```

```
a := 2.3;    //фрагмент 2
b := -1.6;
z : = a + 8.2*b;
```

Только по *знаку присваивания* можно различить, что фрагмент 1 написан на языке JavaScript, а фрагмент 2 — на языке Pascal. Других различий эти фрагменты не имеют.

Структура программы на языке Pascal. Имеется одна существенная ***отличительная особенность*** языка Pascal: *в программах на языке Pascal обязательно надо объявлять типы используемых данных.* В языке JavaScript можно было об этом не беспокоиться: присвоил значение переменной величине — и этого достаточно.

В языке Pascal структура программы предусматривает специальный раздел для объявления (описания) типов данных. Рассмотрим конструкцию простейшей программы на языке Pascal, которая включена в документ «Справка PascalABC.NET»:

```
program имя_программы;
var

begin

end.
```

Первая строка называется *заголовком программы*. Имя программы записывается по тем же правилам, что и имена переменных. В сущности, эта строка не является обязательной.

Далее идёт тот самый раздел объявлений (описаний), который начинается с зарезервированного слова **var**. В этом разделе, как правило, для всех используемых переменных величин должны быть описаны типы значений.

В языке JavaScript мы изучили данные трёх типов: числовые (тип number), строковые (тип string) и логические.

В языке Pascal числовые данные имеют два основных типа: целый (тип integer) и вещественный (тип real). Целый тип используется для переменных с целыми числовыми значениями, а вещественный — для переменных со значениями в виде дробных десятичных чисел. Как и в языке JavaScript, зарезервированное слово string используется для обозначения данных строкового типа. Значения переменных и констант этого типа задаются как наборы символов, обрамлённых одинарными кавычками, например 'катет'. Логические данные обозначаются как данные типа boolean. С другими типами данных языка Pascal (а их достаточно много) можно познакомиться в справке к системе программирования PascalABC.NET.

В разделе объявлений типы переменных величин обычно описываются по шаблону:

список имен : тип;

Например:

```
var
  x, y, z : real;
```

Переменные разных типов описываются в разных строках. Например:

```
var
  x1, y1 : real;
  str : string;
  om1 : boolean;
```

Блок программы, ограниченный операторными скобками **begin**...**end**, мы будем называть *разделом операторов*. Здесь записываются операторы (команды) программы.

Собственно, этим основные отличия в программах на языках JavaScript и Pascal и ограничиваются. Теперь любую программу вычислений, записанную на языке JavaScript, вы легко сможете переписать на язык Pascal.

■ Упражнение 81

Запишем на языке Pascal программу вычислений по формуле

$$\frac{0{,}36 \cdot |a| + 1{,}6 \cdot \sqrt{4 + b^2}}{1{,}7 \cdot (0{,}425 - 0{,}78 \cdot b)},$$

приведённой в упражнении 20 на с. 36 учебника.

В упражнении 20 на языке JavaScript была составлена следующая программа:

```
a=1;
b=1;
up=0.36*Math.abs(a)+1.6*Math.sqrt(4+b*b);
dn=1.7*(0.425-0.78*b);
rez=up/dn;
alert(rez);
```

Открываем на левой половине экрана окно системы программирования PascalABC.NET, а на правой половине с помощью браузера Internet Explorer — документ «Справка PascalABC.NET».

Пользуясь документом «Справка PascalABC.NET», записываем в рабочей зоне системы PascalABC.NET простейшую конструкцию программы. Назовём программу calc81.

Далее описываем используемые переменные в разделе объявлений, считая их дробными числами:

```
program calc81;
var
  a, b, up, dn, rez : real;
begin
end.
```

Осталось переписать команды программы JavaScript на язык Pascal, заменяя знак присваивания, имена математических функций и команду вывода. Не забываем про сдвиг вправо. Получаем следующий текст программы:

```
program calc81;
var
  a, b, up, dn, rez : real;
begin
  a := 1;
  b := 1;
  up := 0.36*Abs(a)+1.6*Sqrt (4+b*b);
  dn := 1.7*(0.425-0.78*b);
  rez := up/dn;
  writeln(rez)
end.
```

Заметим, что в языке Pascal перед операторной скобкой **end** точку с запятой можно не ставить. Сохраняем программу в рабочей папке под именем program81.pas и запускаем её. Если ошибок нет, в окне вывода отображается результат: **-6.52478658434078.**

☐ Вопросы и задания

1. Расскажите об истории создания и развития языка Pascal.

2. Какая из вкладок нижней панели в окне системы PascalABC.NET предназначена для вывода числовых и текстовых данных?

3. Чем различаются правила записи имён переменных величин в языках JavaScript и Pascal?

4. Какие правила записи программ на языках JavaScript и Pascal полностью совпадают?

5. В чём состоит различие в записи команды (оператора) присваивания в языках JavaScript и Pascal?

6. Какой раздел программ на языке Pascal отсутствует в программах на языке JavaScript и почему?

7. Как выглядит шаблон для описания типов переменных величин в программе на языке Pascal?

8. Как описать переменные разных типов в программе на языке Pascal?

9. Создайте на языке Pascal программы вычислений по следующим формулам:

1) $\dfrac{\sqrt{2a+3{,}4b}\cdot(7{,}5c^2-0{,}6a)}{2{,}1b+1{,}5}$;

2) $\dfrac{x^2-3xy+4y^2}{4x^2-3y^2}$;

3) $\dfrac{2{,}3u-v}{4{,}3u+v^2}-5{,}3\,uv$;

4) $\dfrac{(x_1-x_2)(y_1-y_2)}{(x_1-x_2)^2+(y_1-y_2)^2}$.

§ 27. Обработка чисел и строк на языке Pascal

Вычисление сумм. При составлении программ вычислений на языке Pascal мы будем по-прежнему выделять в программах три части: 1) задание исходных данных; 2) вычисления; 3) вывод результатов.

■ Упражнение 82

Запишем на языке Pascal программу для вычисления суммы вида

$$S=\frac{1}{1\cdot 2}+\frac{1}{2\cdot 3}+\frac{1}{3\cdot 4}+\ldots+\frac{1}{n(n+1)}$$

для любого натурального n.

В рабочее окно системы PascalABC.NET вводим конструкцию простейшей программы и даём этой программе имя calc82.

Вспоминаем программы вычислений, составленные в § 18 на языке JavaScript.

Переменную величину nn отводим для исходных данных, в разделе объявлений задаём ей целый тип, а в разделе операторов — её значение (пусть оно равно 15):

```
program calc82;
var
   nn : integer;
begin
   nn := 15;  //Начальные данные

end.
```

Для суммы отводим переменную s типа **real** и задаём её исходное значение 0. Вычислять сумму будем в цикле «для» (**for**). Принцип записи этой конструкции на языке Pascal ясен из сопоставления с аналогичной конструкцией на языке JavaScript:

```
for i := 1 to 10 do  //Pascal
begin


end.
```

```
for (i=1; i<=10; i++) { // JavaScript


}
```

Очевидно, что в языке Pascal *начальное и конечное значения переменной цикла* задаются несколько иным способом. А вот *увеличение значений переменной цикла на единицу* происходит автоматически. Надо также не забыть задать целый тип значений переменной цикла в разделе объявлений программы. Записываем операторы:

```
program calc82;
var
   nn, i : integer;
   s : real;
begin
   nn := 15;
   s := 0;
   for i := 1 to nn do  // Вычисления
   begin

   end;
end.
```

Теперь в теле цикла запишем оператор накопления суммы:

```
s := s + 1/(i*(i + 1));
```

Внимание! Если в конструкции повторения или ветвления в операторные скобки **begin**...**end** заключена только одна команда, то эти операторные скобки можно удалить (не записывать).

Ниже тела цикла запишем две команды (два оператора) вывода — для количества слагаемых и для результата:

```
writeln(nn); // Вывод результатов
writeln(s);
```

Сохраняем программу в рабочей папке под именем program82.pas и запускаем её. Если ошибок нет, в окне вывода отображаются в двух строках числа 15 и 0.9375. Каждый оператор writeln() автоматически переводит вывод на новую строку.

На языке Pascal один оператор вывода позволяет выводить в одну строку несколько значений. В этом случае переменные или значения записываются через запятую. Например, в программе calc82 можно дополнить вывод чисел текстовыми пояснениями:

```
writeln('Количество слагаемых = ', nn);
writeln('Результат = ', s);
```

Процедуры и функции. В языке JavaScript для вспомогательных программ (подпрограмм) мы использовали два варианта конструкций функции пользователя. В языке Pascal также существует два варианта описания подпрограмм. Первый вариант такой конструкции называется ***процедурой*** (описание начинается со служебного слова **procedure**), а второй вариант называется ***функцией*** (описание начинается со служебного слова **function**, как и в языке JavaScript). С этими конструкциями можно познакомиться в документе «Справка PascalABC.NET».

В программе на языке Pascal описания пользовательских процедур и функций располагают в разделе объявлений, обычно после описания типов данных. Имена процедурам и функциям задаются по правилам для имён переменных величин. Заметим, что на языке Pascal типы формальных параметров процедуры или функции должны быть описаны в строке заголовка. Например:

```
procedure first(x, y : real; str : string; n : integer);
```

Для функции в строке заголовка описывается и тип результата исполнения функции. Например:

```
function back(i, j : integer) : real;
```

В каждой процедуре или функции, как и в программе, есть раздел объявлений для описания типов входящих вспомогательных переменных. Например:

```
procedure first(x, y : real; str : string; n : integer);
var d, t : real;
begin

end;
```

Осталось обратить внимание на последнюю особенность в записи функции. Чтобы присвоить имени функции значение результата вычислений, используют служебную переменную **Result**, которой и присваивают итоговое значение.

```
function back(n, m : integer) : real;
var t : real;
begin
  t := Sqrt(1/(n+m));
  Result := t*n/m
end;
```

Обработка натуральных чисел. При составлении программ вычислений на языке Pascal мы будем пользоваться уже разработанными программами на языке JavaScript. Различия проявляются только в особенностях записи этих программ.

■ Упражнение 83

Запишем на языке Pascal программу вычисления суммы цифр заданного многозначного натурального числа. Аналогичную задачу на языке JavaScript мы решали в упражнении 42.

Вводим в окно системы PascalABC.NET конструкцию простейшей программы на языке Pascal. Назовём программу calc83.

В разделе объявлений программы описываем переменные nn и n целого типа (для исходного числа и для рабочей переменной).

Последнюю цифру вычисляем в подпрограмме-функции last(). Для её описания вводим в разделе объявлений программы конструкцию функции. В функцию last() мы будем передавать параметр m целого типа. Результат будет типа **integer**. Это связано с тем, что последняя цифра целого числа будет целым числом, и для её нахождения мы воспользуемся арифметической операцией mod нахождения остатка от деления числа на 10 (аналог оператора % в языке JavaScript). В Pascal результат операции mod тоже будет иметь тип **integer**.

```
program calc83;
var nn, n : integer;
function last(m : integer) : integer;
begin
  Result := m mod 10;
end;

begin

end.
```

Теперь записываем в разделе операторов программы оператор, задающий начальное значение числа, и оператор присвоения значения рабочей переменной:

```
nn := 473;
n := nn;
```

Сумму цифр будем накапливать в переменной s типа integer. Для этого описываем её тип в разделе объявлений и задаём нулевое значение в разделе операторов.

Воспользуемся конструкцией повторения «пока» (**while**) для накопления суммы цифр. В языке Pascal условие окончания цикла после служебного слова **while** записывается без скобок.

```
while n>0 do
begin
  s := s + last(n);
  n := Floor(n/10);
end;
```

После цикла в разделе операторов записываем команды вывода. Получаем следующую программу:

```
program calc83;
var nn, n : integer;
    s : integer;
function last(m : integer) : integer;
begin
  Result := m mod 10;
end;
begin
  nn := 473;                    // Исходные данные
  n := nn;
  s := 0;
  while n>0 do                  // Вычисления
  begin
    s := s + last(n);
    n := Floor(n/10);
  end;
  writeln('Число = ', nn);      // Вывод результатов
  writeln('Сумма = ', s)
end.
```

Сохраняем программу в рабочей папке под именем program83.pas и запускаем её.

Замечание. В условии сравнения цикла «пока» (**while**) на языке Pascal можно использовать знаки сравнения «<» (меньше), «<=» (меньше или равно), «>» (больше), «>=» (больше или равно), которые использовались и в языке JavaScript. А вот для знаков «равно» и «не равно» используются другие знаки (табл. 6).

Таблица 6

Название знака	JavaScript	Pascal
Равно	==	=
Не равно	!=	<>

Другие обозначения имеет язык Pascal и для логических операций (табл. 7).

Таблица 7

Название логической операции	JavaScript	Pascal
И	&&	and
ИЛИ	||	or
НЕ	!	not

Обработка строк. Значения переменных строкового типа на языке Pascal задаются уже известным нам способом — с помощью оператора присваивания. Знак «+» используется для «склейки» значений. Например, переменная величина str0 строкового типа с помощью оператора присваивания

```
str0 := 'Миру' + ' мир';
```

получит значение «Миру мир».

Длина строки вычисляется с помощью функции Length() как величина целого типа. Например, в результате исполнения команды (оператора)

```
mm := Length ('Информация');
```

величина целого типа mm получит значение 10.

Символы в строке на языке Pascal нумеруются, начиная с 1 (в отличие от языка JavaScript). А выделить один символ можно по имени строковой переменной с индексом, равным порядковому номеру символа. Например, команды

```
str := 'Информация';
writeln(str[5]);
```

выведут на экран символ «р» (пятый по счёту).

Выделить подстроку (фрагмент строки) позволяет функция Copy(), которая имеет три аргумента, причём второй и третий аргументы должны иметь целый тип. Например, по команде

```
str1 := Copy('Информация', 3, 5);
```

величина str1 строкового типа получит значение «форма» (подстрока, которая начинается с третьего символа и содержит 5 символов).

■ Упражнение 84

Задано значение строковой переменной. Построим строку, в которой символы записаны в обратном порядке. Аналогичная задача на языке JavaScript решалась в упражнении 51.

Вводим в окно системы PascalABC.NET конструкцию простейшей программы на языке Pascal. Назовём программу calc84.

В разделе объявлений программы описываем переменные str0 и str1 типа **string** (для исходной строки и для результата).

В разделе операторов задаём значение исходной строки и значение «пустая строка» для результата.

```
program calc84;
var
  str0, str1 : string;
begin
  str0 := 'краб';
  str1 := '';
end.
```

По аналогии с программой на языке JavaScript в цикле «для» (**for**) будем получать по одному значения символов исходной строки, начиная с конца, и добавлять их (склеивая) к строке результата. Добавим также в раздел объявлений переменную цикла i целого типа.

```
for i := Length(str0) downto 1 do
  str1 := str1 + str0[i];
```

Заметим, что в цикле «для» (**for**) вместо служебного слова **to** записано служебное слово **downto**. Служебное слово **downto** в программах на языке Pascal означает, что значение переменной цикла уменьшается каждый раз на единицу.

Программу завершает вывод исходной строки и результата:

```
writeln('Исходная строка = ', str0);
writeln('Результат = ', str1);
```

Сохраняем программу в рабочей папке под именем program84.pas и запускаем её.

Ввод исходных данных с клавиатуры. Вводить в программу значения строкового типа можно и во время исполнения программы. Для этого служит оператор readln(). В качестве аргумента этого оператора указывают имя переменной строкового типа.

Например, в программе program84.pas вместо оператора присвоения значения переменой str0 можно записать оператор:

```
readln(str0);
```

При исполнении программы в нижней части окна системы открывается строка для ввода данных. Ввод завершается нажатием клавиши **Enter** клавиатуры или щелчком по кнопке **Ввести** в строке для ввода. Введённое значение присвоится переменной str0.

Для ввода чисел в программу можно использовать тот же приём.

Для преобразования строк в числа и наоборот используют *функции преобразования типов*:

StrToInt(s: **string**): **integer** (преобразует строковое представление целого числа к числовому значению);

StrToFloat(s: **string**): **real** (преобразует строковое представление дробного числа к числовому значению);

IntToStr(a: **integer**): **string** (преобразует целое число к строковому представлению);

FloatToStr(a: **real**): **string** (преобразует дробное число к строковому представлению).

☐ Вопросы и задания

1. Поясните особенности записи конструкции повторения «для» (**for**) в языке Pascal.

2. Поясните особенности записи оператора вывода writeln() в языке Pascal.

3. Чем различаются правила записи функций пользователя в языке JavaScript и процедур и функций в языке Pascal?

4. Какая переменная служит для присвоения значения функции в языке Pascal?

5. Какова система индексации отдельных символов в строковых данных на языке Pascal?

6. Какой оператор позволяет вводить данные в программу в процессе её исполнения?

7. Составьте программу вычисления суммы квадратов натуральных чисел от 1 до n.

8. Составьте программу, которая выводит на экран первые n нечётных натуральных чисел.

9. Составьте программу вычисления НОД двух натуральных чисел.

10. Дана строка и два символа. Составьте программу, которая определяет, какой символ встречается в строке чаще другого.

11. Дана последовательность цифр в форме строки. Составьте программу, которая определяет количество нечётных цифр в строке.

§ 28. Линейные массивы и работа с графикой на языке Pascal

Линейные массивы. Понятие линейного массива является единым для всех языков программирования. Различаются только способы создания массивов в программах.

В языке Pascal для создания массивов существует тип данных **array** (массив). Чтобы задать массив в виде переменной величины, следует описать её в разделе объявлений программы. Например:

```
mas : array [1 .. 20] of real;
mm1 : array [0 .. 9] of string;
```

В квадратных скобках после имени типа **array** задаётся диапазон индексов (числа разделены двумя точками), а затем тип значений элементов. При описании типа элементам массива можно сразу задать начальные значения. Например:

```
mas : array [1 .. 10] of integer := (1, 2, 3, 4, 5, 6, ,7, 8, 9, 10);
m : array [1 .. 4] of string := ('зима', 'весна', 'лето', 'осень');
```

К каждому элементу массива обращаются по имени массива с индексом, например: mas[2], m[4]. Значения отдельным элементам массива можно задавать и в программе с помощью оператора присваивания.

■ Упражнение 85

Дан массив из 5 слов: зебра, слон, лошадь, вол, як. Составим программу поиска самого короткого слова.

Вводим в окно системы PascalABC.NET конструкцию простейшей программы на языке Pascal. Назовём программу calc85.

Алгоритм строим по образцу программы на языке JavaScript в упражнении 70. В разделе объявлений программы описываем переменную mas типа **array** (для исходных данных), причём элементам mas сразу присваиваем значения. Для вспомогательных целей объявляем переменные mm, i целого типа и rez строкового типа.

В разделе операторов задаём длину первого слова как значение переменной min, а само это слово записываем как значение переменной rez. Далее в цикле просматриваем элементы массива и вычисляем длины слов. Если новое слово имеет меньше букв, записываем его в rez, а его длину — в min.

```
program calc85;
var
   mas : array [1 .. 5] of string := ('зебра', 'слон', 'лошадь', 'вол', 'як');
   min, i : integer;
   rez : string;
begin
   min:= Length(mas[1]);
   rez := mas[1];
   for i := 2 to 5 do
   begin
      if Length(mas[i])<min then
      begin
            rez := mas[i];
            min := Length(mas[i])
      end;
   end;
   writeln('Самое короткое слово '', rez,'' содержит ', min,' букв');
end.
```

Сохраняем программу в рабочей папке под именем program85.pas и запускаем её.

Заметим, что оператор вывода содержит в строковых константах запись трёх одинарных кавычек подряд. Дело в том, что для вывода одного штриха в окно вывода системы нужно записать два штриха подряд. Третий штрих относится к обрамлению всей строковой константы.

Работа с графикой. Для создания программ на языке Pascal часто используются внешние файлы, также написанные на языке Pascal, которые называются ***модулями***. Подключение модулей к программе производится специальной записью в начале раздела объявлений программы.

В частности, для работы с графикой в программах используется модуль GraphABC. Подключается этот модуль записью вида

```
uses GraphABC;
```

Такая запись по-английски означает «используется модуль GraphABC». Заметим, что одной строкой **uses** можно подключить сразу несколько модулей. Для этого имена модулей записываются справа от служебного слова **uses** через запятую.

После подключения модуля GraphABC результаты (данные и графические изображения) выводятся из программы в отдельном графическом окне системы с заголовком «GraphABC.NET».

По умолчанию считается, что в графическом окне задана прямоугольная система координат и каждому пикселу в окне поставлена в соответствие пара целочисленных координат.

Для построения отрезка прямой линии в графическом окне используют оператор

```
Line(x1, y1, x2, y2 : integer);
```

■ Упражнение 86

Построим линию, соединяющую точки с координатами (0, 0) и (200, 200).

Вводим в окно системы PascalABC.NET конструкцию программы для работы с графикой. Назовём программу calc86.

Вводим в разделе операторов команду построения линии:

```
program graph86;
uses GraphABC;
begin
  Line(0,0,200,200);
end.
```

Сохраняем программу в рабочей папке под именем program86.pas и запускаем её. Если программа не содержит ошибок, то на графическом экране выводится прямая линия. Но расположена она несколько иначе, чем ожидалось. Она выходит из левого верхнего угла экрана. Дело в том, что начало координат в графическом окне расположено именно в левом верхнем углу окна. Ось OX при этом направлена вправо, а ось OY — сверху вниз. Это обязательно надо учитывать при построениях.

Графические примитивы. При работе в графическом окне системы PascalABC.NET, кроме прямых линий, можно использовать и другие графические примитивы. Это могут быть:

`Rectangle(x1, y1, x2, y2);` — прямоугольник;

Circle (z, y, r); — окружность;

Arc(x, y, r, ang1, ang2); — дуга окружности между лучами под углом ang1 и ang2 в градусах;

Pie(x, y, r, ang1, ang2); — сектор круга, ограниченного дугой окружности, и др.

Цвет линий устанавливается оператором вида SetPenColor(col); где col — символическое имя цвета.

Некоторые из *символических имён цветов* приведены в таблице 8.

Таблица 8

Цвет	Имя	Цвет	Имя
чёрный	clBlack	голубой	clSkyBlue
красный	clRed	фиолетовый	clPurple
зелёный	clGreen	коричневый	clBrown
жёлтый	clYellow	серебряный	clSilver
синий	clBlue	белый	clWhite

Новый цвет закрашивания области одного цвета устанавливает оператор вида FloodFill(x, y, col); где (x, y) — координаты некоторой точки из рассматриваемой области; col — символическое имя нового цвета.

Толщину линии устанавливает оператор вида SetPenWidth(w); где аргумент w задаёт количество пикселов.

Стиль пера устанавливается оператором вида SetPenStyle(p); где аргумент p может принимать значения:

psSolid — сплошная линия;

psDash — штриховая линия;

psDot — пунктирная линия и др.

■ Упражнение 87

Построим изображение квадрата 200 на 200 из красных линий шириной 3 пиксела и вписанной в него окружности синего цвета с линией толщиной 2 пиксела. Угловые криволинейные треугольники зальём жёлтым цветом.

Вводим в окно системы PascalABC.NET конструкцию программы для работы с графикой. Назовём программу graph87.

```
program graph87;
uses GraphABC;
begin
  SetPenWidth(3);
  SetPenColor(clRed);
  Rectangle(100, 100, 300, 300);
  SetPenWidth(2);
  SetPenColor(clBlue);
  Circle (200, 200, 100);
  FloodFill(105, 105, clYellow);
  FloodFill(105, 295, clYellow);
  FloodFill(295, 105, clYellow);
  FloodFill(295, 295, clYellow);
end.
```

Сохраняем программу в рабочей папке под именем program87.pas и запускаем её.

Построение графиков функций. Графические возможности системы PascalABC.NET позволяют строить графики функций. Трудности могут возникнуть только из-за особенностей системы координат в графическом окне, но они вполне преодолимы.

■ Упражнение 88

Перенесём начало координат графического экрана в точку с координатами (300, 250), направим вверх ось ординат и построим график функции $y = x^2 - x - 2$ на отрезке $[-2, 2]$ с масштабом 1 : 50 (в одной единице длины — 50 пикселов экрана).

Если взять за новое начало координат точку (x0, y0) графического экрана, то формулы пересчёта координат (x, y) обычной системы координат в координаты (xg, yg) графического экрана имеют вид:

```
xg := x0 + x;
yg := y0 - y;
```

Если же учитывать масштаб как значение переменной ms, то формулы примут вид:

```
xg := x0 + ms*x;
yg := y0 - ms*y;
```

Вводим в окно системы PascalABC.NET конструкцию программы для работы с графикой. Назовём программу graph88.

В разделе объявлений описываем как переменные типа **real** координаты x, y, значения xb, xe концов заданного отрезка и переменную h для шага построений по x. Описываем целые типы для переменных x0, y0, xg, yg и ms из формул, а также для переменных xs, ys, которые должны хранить координаты xg, yg предыдущего шага. Строим оси координат и задаём начальные значения.

Программу строим по аналогии с программой упражнения 62.

```
program graph88;
uses GraphABC;
var x, y, xb, xe, h : real;
   x0, y0, ms : integer;
   xg, yg, xs, ys : integer;
begin
                                        // Оси координат
   x0 := 300; y0 := 250;
   Line(x0-200,y0,x0+200,y0);
   Line(x0,y0-200,x0,y0+200);
                                        // Начальные значения
   xb := -2; xe := 2;
   ms := 50;
   h := 0.05;
   x := xb;
                                        // Построение графика
   while x<=xe do
   begin
      y := x*x-x-2;
      xg := x0+Round(ms*x);
      yg := y0-Round(ms*y);
      if x<>xb then
         Line(xs,ys,xg,yg);
      xs := xg;
      ys := yg;
      x := x+h
   end
end.
```

Сохраняем программу в рабочей папке под именем program88.pas и запускаем её.

Вопросы и задания

1. Опишите способ задания линейного массива в программе на языке Pascal.

2. Какими способами задают значения элементов массива?

3. Дан массив из 10 чисел. Составьте программу, которая выводит два самых больших по модулю элемента этого массива.

4. Дан массив названий месяцев. Составьте программу, которая выводит на экран названия, включающие буквы «а» и «д».

5. Каким способом подключаются к программе на языке Pascal внешние файлы, написанные на языке Pascal?

6. Какой модуль используется в программах на языке Pascal для работы с графикой?

7. Опишите систему координат графического окна системы PascalABC.NET.

8. Опишите графические примитивы языка Pascal.

9. Почему в программе program88.pas для вычисления значений переменных xg, yg в арифметическом выражении используется функция Round()?

10. Составьте программу для изображения трёх пересекающихся окружностей с закраской областей в разные цвета.

11. Составьте программу штриховки квадрата под углом 45°.

12. Составьте программу построения штриховой линией графика функции $y = x^2 - 3$ на отрезке $[-3, 3]$.

13*. Составьте программу построения в одном окне двумя цветами графиков функций $y = 2x^3$ и $y = \frac{1}{x^2 + 1}$ на отрезке $[-3, 4]$.

ГЛАВА 4

МОДЕЛИРОВАНИЕ И ПРОЕКТИРОВАНИЕ

§ 29. Модели и моделирование

Модель (от лат. modulus — образец) как понятие возникло в античные времена и связано с практической деятельностью человека.

Модели самолётов, судов, автомобилей знакомы нам с детства. При желании можно купить специальные наборы деталей для создания этих маленьких копий реальных машин. Любители спортивного моделизма посвящают всё своё свободное время созданию таких моделей.

В обыденной жизни мы привыкли, что модель — это имитация, повторение в уменьшенном масштабе какого-либо реального объекта (его называют моделируемым объектом). Глобус, например, — это модель земного шара, а плюшевый медведь — модель живого медведя.

Модель не всегда создаётся искусственно. Иногда в качестве модели используют другие реальные объекты. В медицине, например, новые методы сложных операций отрабатывают на кроликах, собаках, обезьянах. Организм этих животных для медицинских целей служит моделью человеческого организма. Именно так отрабатывались операции по пересадке сердца и других жизненно важных органов человека.

В современном понимании модель не всегда внешне похожа на моделируемый объект. Например, лётчики, показывая друг другу особенности фигур высшего пилотажа, часто используют ладонь руки. Ладонь в этом случае становится моделью самолёта. А радиоэлектронная модель сердца вообще представляет собой электрическую схему (радиодетали, соединённые проводниками).

Моделируемый объект по отношению к модели называют *объектом-оригиналом*.

Модель объекта — это другой материальный или информационный объект, который отражает особенности объекта-оригинала, существенные с точки зрения поставленной задачи.

Примеры моделей в форме информационных объектов — это чертежи, рисунки, фотографии объектов-оригиналов и т. п.

Модели процессов на практике также используются весьма широко. Деловые игры являются моделями реальных процессов коллективного управления, процесс испытаний автомобиля является моделью процесса его эксплуатации, некоторые виды тренировок спортсменов являются моделями процесса соревнований и т. п. Схемы на бумаге, кино- и видеозаписи процессов являются, с одной стороны, информационными объектами, а с другой — моделями этих процессов.

Модель процесса — это другой процесс или информационный объект, который отражает особенности процесса-оригинала, существенные с точки зрения поставленной задачи.

Следует отметить, что слово «модель» в русском языке может иметь и другой смысл, например фотомодель (человек, позирующий фотографу), модель автомобиля (тип, марка) и др.

Моделирование — это процесс создания и исследования моделей.

Жизнь постоянно ставит перед человечеством самые разнообразные задачи: как создать новые машины и механизмы, как построить более удобные жилища и производственные сооружения, как уберечь дикую природу от уничтожения, как избавиться от голода и болезней, как бороться с глобальным потеплением и многие, многие другие. Моделирование позволяет решать эти задачи быстрее и с меньшими затратами средств.

Варианты решения возникающих задач всё чаще проверяются на моделях, потому что испытания сложных и дорогостоящих объектов требуют больших затрат и не всегда возможны в принципе.

Например, чтобы изучить сейсмические свойства здания, никто не станет устраивать искусственное землетрясение. Гораздо проще и дешевле создать модель здания и испытать её на вибрационном стенде.

Другой пример — космические исследования. Полёты космических кораблей и станций всегда сопровождаются постройкой макетов космических объектов в натуральную величину, которые работают в бассейнах с водой в периоды, когда космические объекты находятся в полёте. Это делается для оказания помощи космонавтам в критических ситуациях. Когда такая ситуация возникает, водолазы находят наилучший путь устранения аварии на макете, и информация об этом передаётся на орбиту. Моделирование — это практически единственный путь получения такой ценной информации.

Модели дают возможность получать наилучшие результаты при решении сложных задач. Роль моделирования растёт вместе с ростом сложности задач в науке и технике.

Заметим, что в некоторых отраслях человеческой деятельности моделирование понимается только как процесс создания моделей. К таким отраслям относятся спортивное моделирование, деятельность в области искусства (театр и кино) и т. п.

Рассмотрим пример использования моделирования в кинематографии. В легендарном фильме «Белое солнце пустыни» один из драматичнейших эпизодов — взрыв баркаса — снят с помощью модели. Настоящий баркас конечно же не взрывали, а в кинофильм вошли кадры взрыва небольшой модели. Это так называемые комбинированные съёмки. Модели такого рода часто использовались при съёмках крушений поездов, больших разрушений. В последнее время для этих целей используют компьютерные модели.

В быту мы часто используем моделирование, даже не задумываясь над этим. Самый простой пример — фотографирование. Фотография, несомненно, является моделью объекта-оригинала, но мы никогда не употребляем этот термин.

Адекватность модели — это степень соответствия модели целям моделирования.

Модель создаётся для того, чтобы отразить информацию о свойствах моделируемого объекта или процесса. Большинство рассмотренных нами моделей хранят информацию о внешней форме моделируемого объекта (спортивные модели, модели самолётов, плюшевый медведь, модель космического корабля, модели для комбинированных съёмок). Глобус несёт информацию о внешней форме и особенностях земной поверхности, сейсмическая модель здания — информацию о внешней форме и прочности здания.

Модель обычно проще моделируемого объекта и отражает только те его свойства, которые требуют изучения или использования. Но именно требуемые свойства оригинала должны максимально отражаться в модели (т. е. модель должна быть адекватной целям моделирования), чтобы исследователь (пользователь) мог воспользоваться результатами моделирования.

В зависимости от целей моделирования для одного и того же объекта могут быть построены совершенно непохожие модели. Одна модель может отражать одни свойства моделируемого объекта, другая — совершенно иные. Мы, например, обращались к двум разным моделям самолёта. В одном случае это была небольшая копия самолёта, в другом — ладонь лётчика.

История моделирования имеет немало интересных страниц. Одна из древнейших ветвей моделирования связана с технологией обработки металлов литьём. По данным археологии, в III тыс. до н. э. уже использовались предметы и украшения, которые были изготовлены из бронзы и драгоценных металлов путём отливки в формы. Формы, в свою очередь, изготавливались по моделям-образцам (отсюда и пошло название modulus — образец). Эта технология не претерпела никаких изменений до сих пор. На заводах с литейным производством и сейчас есть модельные цеха, где изготавливаются деревянные модели для создания форм.

А вот пример эпохи итальянского Возрождения. Архитектор Филиппо Брунеллески (1377—1446) при сооружении купола Флорентийского собора изготовил сначала малую модель купола. Специальное жюри утвердило его вид, и была изготовлена более крупная рабочая модель, которая использовалась как образец в процессе постройки.

В середине XIX в. в связи с развитием металлургии и созданием паровых машин начался переход от деревянных парусных судов к стальным пароходам. Для тихоходных парусников обводы корпуса

не имели большого значения. А вот у скоростных пароходов от формы корпуса в значительной степени зависел расход топлива и дальность плавания.

Решить задачу выбора формы корпуса прямым опытным путём невозможно. Слишком дорогое удовольствие создавать суда с разными формами корпуса только для испытаний. Задача была решена с помощью малых моделей судов в специальных бассейнах.

Моделирование получило признание не сразу. Поучительным примером служит история броненосца «Кэптен», который был построен в Англии в 1870 г. Во время строительства судна учёный-кораблестроитель В. Рид по своей инициативе создал и исследовал его модель. Моделирование показало, что в море при малейшем волнении броненосец должен опрокинуться.

Результаты исследования были доложены в Адмиралтейство (морское министерство Великобритании). Но заявление учёного с его «игрушечной моделью» никто всерьёз не воспринял. При выходе в море броненосец перевернулся, и 523 моряка погибли.

☐ Вопросы и задания

1. Что такое модель?

2. Приведите примеры известных вам моделей.

3. Что такое моделирование?

4. Почему люди используют моделирование в своей деятельности?

5. Что такое адекватность модели?

6. Какие свойства самолёта отражает лётчик, используя ладонь руки как модель? Является ли ладонь руки для лётчика адекватной моделью?

7. Какое название для моделей внешней обстановки используется при подготовке театральных постановок?

8. Что такое сценарий фильма с точки зрения информатики?

§ 30. Виды моделей

Каждая модель имеет свои особенности, причём важнейшими считаются:

- форма реализации;
- внешние размеры;

- зависимость от времени;
- связь с конкретной отраслью знаний.

Вид модели определяется её особенностями. Если несколько различных моделей имеют одну и ту же особенность, то их относят к одному виду.

Виды моделей в зависимости от формы реализации. Реализация модели может быть самой разной: в форме материального объекта, в форме мысленного образа, в форме документа, в форме изображения на экране компьютера и т. д.

В зависимости от формы реализации различают два основных вида моделей: материальные и информационные.

Материальная модель — это модель в форме материального объекта или комплекса материальных объектов.

Материальные модели наиболее известны. Иногда их также называют *предметными*. Например, глобус — это материальная модель земного шара. То же самое можно сказать о радиоэлектронной модели сердца. Материальные модели создаются искусственно или подбираются из имеющихся материальных объектов.

Информационная модель — модель в форме информационного объекта.

Информационные модели распространены очень широко. Например, любое изображение материального объекта или процесса на бумаге (рисунок, чертёж, схема) является информационной моделью. Или возьмём учебник истории. С точки зрения информатики он содержит информационные модели (описания) различных исторических событий. Теоретические исследования учёных посвящены изучению реальных и воображаемых миров, представленных в виде записи текстами, формулами и схемами, т. е. посвящены изучению информационных моделей.

Информационные модели, в свою очередь, также можно классифицировать. В зависимости от формы реализации выделяют следующие виды информационных моделей: мысленные, вербальные, документальные, аппаратно-зависимые.

Мысленная модель — это информационная модель в мысленной форме. Человек способен представить в своём воображе-

нии самые разные предметы, ситуации, явления. Такие представления человека и есть мысленные модели.

Каждый из нас, например, хотя бы раз попадал в ситуацию, когда приходилось передвигаться по комнате в полной темноте. Если комната вам хорошо знакома, то вы легко находите выход. Это означает, что в вашем сознании сложилась мысленная модель комнаты.

Хороший автомеханик всегда знает, как устранить неисправность в автомобиле, потому что в его сознании существует мысленная модель исправного автомобиля. Неисправность при сравнении с моделью в его памяти становится заметной. Специалист отличается от неспециалиста именно тем, что в его сознании есть достаточное количество нужных мысленных моделей.

Мысленные модели воображаемых объектов дают простор творчеству инженеров и конструкторов, художников и дизайнеров.

Вербальная модель — это информационная модель в устной словесной форме. «Вербальный» (от лат. verbalize) означает «словесный, устный». Примерами вербальных моделей являются устные объяснения учителем нового материала на уроках, развёрнутые устные ответы учеников, научные положения в устных докладах и выступлениях. Вербальные модели — это чаще всего устный пересказ мысленных моделей.

Документальная модель — это информационная модель, записанная на бумаге, картоне или другом плоском носителе информации. Примерами документальных моделей являются текстовые описания событий, фотографии, схемы. Любая газета, например, заполнена описаниями событий прошедших дней, т. е. моделями этих событий.

Со многими объектами окружающего мира мы знакомы только по их документальным моделям: фотографиям, описаниям. Мало кто из нас, например, воочию видел Эйфелеву башню, однако все хорошо представляют, как она выглядит. В этом нам помогли её документальные модели. Документальными моделями являются чертежи машин и механизмов, географические карты, картины художников и т. п.

Аппаратно-зависимая модель — это информационная модель, которая записана на твёрдых носителях информации и для восприятия которой требуется специальная аппаратура.

К аппаратно-зависимым относятся информационные модели, которые записаны на фото-, кино- и магнитных плёнках, компью-

терных носителях. Пользуясь определениями из учебника для 8 класса, можно сказать, что информация в аппаратно-зависимых моделях представлена в технической, или компьютерной, форме. Для работы с такими моделями требуется специальная аппаратура: магнитофоны, проигрыватели, проекторы, фильмоскопы, видеомагнитофоны, CD-плееры и, наконец, компьютеры.

Аппаратно-зависимые информационные модели совершенно естественно делятся на компьютерные и некомпьютерные.

Некомпьютерные модели — информационные модели в форме слайдов, кинофрагментов, аудио- и видеозаписей на магнитной плёнке. *Компьютерные модели* — модели в форме электронных документов и графических объектов, цифровых аудио- и видеозаписей. Компьютерными моделями являются, в частности, виртуальные объекты на экране компьютера.

Особый класс компьютерных моделей связан с компьютерными программами, которые называются *моделирующими*. Эти информационные модели создаются на экране компьютера во время исполнения моделирующих программ. Примерами таких программ являются компьютерные игры, создающие на экране компьютера невообразимое многообразие виртуальных миров. Учебные исполнители, включая исполнитель «Фломастер», — это также компьютерные модели, которые связаны с работой моделирующих программ.

Следует иметь в виду, что для одного и того же объекта могут быть построены модели, которые отражают одни и те же свойства объекта-оригинала, но имеют разные формы реализации. Например, модель участка земной поверхности можно представить мысленно (мысленная модель), в форме рельефного макета (материальная модель), в форме карты (документальная модель) или в форме графического объекта на экране компьютера (компьютерная модель). Схема деления моделей на виды в зависимости от формы реализации приведена на рисунке 44.

В зависимости от внешних размеров модели делятся на масштабные и немасштабные.

Модель называется ***масштабной***, если внешние размеры модели получены пропорциональным увеличением или уменьшением размеров моделируемого объекта. Масштабную предметную модель обычно называют *макетом*.

Рис. 44

Глобус Земли — масштабная модель. Сюда же относятся чучела животных, муляжи. Документальная и компьютерная модели также могут быть масштабными, например документальная географическая карта, электронный чертёж.

Если внешние размеры модели не отражают пропорционально внешних размеров моделируемого объекта, то такую модель называют ***немасштабной***.

Немасштабной моделью, например, является любое описание моделируемого объекта в виде текста и формул.

Модель называется ***статической***, если она не учитывает зависимость свойств моделируемого объекта от времени. Модель называется ***динамической***, если она моделирует процесс.

Например, движущиеся модели судов являются динамическими, а модели судов в музеях — статическими. Плюшевый медведь — статическая модель медведя, а заводная игрушка — динамическая.

Если динамическая модель имитирует свойства некоторого процесса, то такая модель называется *имитационной*.

Виды моделей по отраслям знаний. Каждая наука при описании фактов и явлений пользуется своим языком: для математики — это язык цифр и математических формул, для физики — это язык физических закономерностей. Свои подходы и свой язык имеют химия, биология, генетика, социология и др.

Использование научных знаний в процессе моделирования повышает адекватность моделей. Поэтому при создании моделей обычно используют язык, законы, методы той или иной науки. В зависимости от этого модели разделяют по отраслям знаний: *математические*, *статистические*, *биологические*, *социологические* и т. д.

Чаще других используются математические модели, потому что математика даёт возможность учитывать важные числовые свойства объектов (размеры, скорость, углы, дальность, массу, плотность и т. п.).

Отнесение модели к определённому виду. Чтобы отнести модель к определённому виду, нужно рассмотреть все её особенности по отдельности.

Примеры рассмотрения некоторых моделей представлены в таблице 9.

Таблица 9

Модель	Форма реализации	Зависимость от внешних размеров	Зависимость от времени	Отрасль знания
Глобус	Материальная	Масштабная	Статическая	География
Старинная фотография	Документальная	Масштабная	Статическая	История
Исполнитель «Фломастер»	Компьютерная	Немасштабная	Динамическая	Информатика
Текст в окне текстового редактора	Компьютерная	Масштабная	Статическая	Информатика

Для определения вида моделей могут учитываться и другие их особенности: способ решения (аналитические и численные модели), способ записи (алгоритмические и неалгоритмические модели) и т. д.

☐ Вопросы и задания

1. Перечислите основные особенности изученных вами видов моделей.

2. Как определяется вид модели?

3. Какие модели называют материальными? Приведите примеры.

4. Какие модели называют информационными? Приведите примеры.

5. Перечислите виды информационных моделей в зависимости от формы их реализации. Приведите примеры.

6. Приведите примеры моделей одного и того же объекта, различных по форме реализации.

7. Какие модели называют масштабными; динамическими; статическими? Приведите примеры.

8. Может ли модель быть одновременно математической и биологической; математической и химической? Поясните ответы.

9. К какому виду моделей следует отнести плюшевого зайца, заводную машинку, электронную географическую карту?

§ 31. Проекты и проектирование

Понятия «проект» и «проектирование» связаны, прежде всего, с разработкой новых машин, зданий и сооружений. Например, перед сборкой нового сложного технического объекта необходимо изготовить его составные части общим числом, доходящим до нескольких тысяч деталей. Детали должны иметь такие параметры, которые позволяли бы совместить их в единое целое. Без специального процесса разработки документальных моделей отдельных деталей и всего объекта это невозможно.

Дословно «проект» — это замысел, план (от лат. progectus — брошенный вперёд). В технике, архитектуре и других смежных областях это понятие более конкретизировано.

Проект — проектная документация, которая содержит описания, расчёты, чертежи будущих сооружений или технических комплексов.

Проектная документация может включать один или несколько технических документов. Кроме проектной документации в проект могут входить макеты (сооружений, технических комплексов или их отдельных составляющих). Например, архитектурные проекты непременно включают макеты возводимых зданий и сооружений.

Проект в технике и архитектуре фактически является моделью воображаемого объекта. Заметим, что такие модели часто называ-

ют *моделями-прототипами*. Поэтому, говоря языком информатики, проект — это документальная модель-прототип, которая может быть дополнена материальными моделями-прототипами (макетами).

Отметим, что понятие «проект» может иметь и другой смысл. Например, «музыкальный проект», «театральный проект».

Проектирование — процесс создания проекта.

Проектирование — это достаточно длительный процесс, который выполняется последовательно в несколько стадий:

- эскизное проектирование;
- техническое проектирование;
- рабочее проектирование.

На каждой стадии модель-прототип уточняется и конкретизируется. Каждая стадия проектирования заканчивается разработкой проекта с соответствующим названием:

- проектное задание (эскизный проект), в котором выявляются возможность и целесообразность создания проектируемого объекта и устанавливаются основные технические и художественные решения и экономические показатели;
- технический проект, который содержит подробную разработку предложенных в эскизном проекте технических и технологических решений;
- рабочий проект, в котором чертежами определены окончательные формы и размеры всего проектируемого объекта и каждого его элемента, а также уточнены технико-экономические показатели.

Проектирование и модели. С целью выбора наиболее оптимальных технических решений на отдельных стадиях проектирования создаются и исследуются самые разнообразные модели. Форма реализации модели при этом напрямую связана со сложностью решаемых задач проектирования.

По историческим меркам не столь далеко то время, когда при проектировании использовались только мысленные, документальные и материальные модели.

Рассмотрим использование таких видов моделей на простых примерах.

Для решения несложных задач достаточно мысленной модели. Например, направляясь из школы домой, задачу выбора правильного пути каждый решает в уме (с помощью мысленной модели). Карта или схема при этом не нужны — дорога хорошо знакома. Мысленная модель адекватно отражает реальную дорогу.

Для решения более сложных задач нужно строить документальную модель. Большинство таких задач трудно решить в уме, и при решении требуются записи. Именно документальные модели составляют основу любого проектирования.

А вот при решении особо сложных задач проектирования могут потребоваться материальные модели. Примеры тому — решение задачи проектирования корпусов стальных судов в XIX в. и решение задач расчёта аэродинамических нагрузок с помощью продувки в специальных трубах.

Документальные модели называют также образно-знаковыми, имея в виду способ записи на носителе.

Образно-знаковая модель — информационная модель, представленная с помощью визуальных образов и знаков.

Знаки — это буквы алфавита, цифры, знаки препинания и символы формул. Образы — это графические или фотографические изображения.

Образно-знаковыми моделями являются словесные описания, таблицы, рисунки, схемы, чертежи, графики, планы, карты, фотографии как существующих, так и разрабатываемых объектов или процессов.

Образно-знаковые модели делятся на текстовые и графические. Эти понятия связаны с известными нам традиционными формами хранения информации на бумаге.

Текстовая (знаковая) модель — это информационная модель в форме текста.

Текстовые модели (модели-описания) могут быть написаны с использованием различных естественных и формальных языков. Они могут содержать формулы. Текстовые модели могут являться текстами компьютерных моделирующих программ.

Графическая (образная) модель — это информационная модель в форме графического объекта.

Нередко документальные модели (в том числе электронные) имеют вид текстовых моделей с включением графических объектов. Именно так выглядит проектная документация для сложных объектов и сооружений.

Среди множества документальных графических моделей выделим модели, которые будем называть чертёжно-графическими.

Чертёжно-графическая модель — это графический документ, в котором изображение состоит из линий и точек и который выполнен с соблюдением определённых правил.

К чертёжно-графическим моделям относятся эскизы, схемы, планы, чертежи, карты и т. п. Чертёжно-графические модели, как правило, выполняются чёрным цветом на белой бумаге.

Эскиз, технический рисунок — это чертёжно-графическая модель, которая может быть выполнена от руки, но с соблюдением основных пропорций.

Схема — это чертёжно-графическая модель объекта, в которой его составные части, их взаимное расположение и связи показаны в виде условных изображений и обозначений.

План — это чертёжно-графическая модель территории или горизонтального разреза здания (его части), которая содержит контурное или упрощённое обозначение входящих элементов с соблюдением масштаба.

Чертёж — это чертёжно-графическая модель создаваемого объекта, которая выполнена с соблюдением требований государственных стандартов.

Чертежи часто называют языком техники. Создавать машины и механизмы, строить здания и сооружения без чертежей просто немыслимо. Для производственных чертежей соблюдение требований стандартов очень важно, потому что одним и тем же чертежом пользуются разные специалисты и всем без исключения чертёж должен быть понятен.

Карта — это чертёжно-графическая модель поверхности Земли, небесного тела, звёздного неба или их фрагментов, которая содержит условные обозначения и отображает реальные и условные объекты.

Современные карты создаются на основе аэрокосмических снимков. Карты являются представителями немногочисленной группы многоцветных чертежей. Если для карт лунной поверхности и карт звёздного неба многоцветность не нужна, то на картах географических, политических, этнографических, да и многих других цветом выделяются зоны разного характера, в том числе условные. Например, этнографические карты содержат обозначения границ зон расселения этнических групп, но реально таких границ на земной поверхности не существует.

Компьютерные модели в проектировании. С появлением компьютеров в проектировании всё активнее стали использоваться компьютерные модели. В последнее время роль компьютерных моделей в проектировании заметно возросла, и они стали преобладать. Это связано с тем, что заметно возросли возможности компьютерной техники. Современные компьютеры позволяют создавать компьютерные модели практически любой сложности.

Компьютерные модели обладают универсальными возможностями. С одной стороны, они могут заменить документальные модели их электронными вариантами, с другой — заменить материальные модели адекватными компьютерными моделирующими программами. Сегодня, например, даже продувку корпусов самолётов и автомобилей в аэродинамических трубах можно не проводить. Вполне достаточно изучения этого процесса с помощью компьютерных моделей.

Таким образом, компьютерные информационные модели позволяют в процессе проектирования заменить и документальные, и материальные модели.

☐ Вопросы и задания

1. Что такое проект?

2. Что такое модель-прототип?

3. Что такое проектирование и каковы его стадии?

4. С какой целью используются модели в процессе проектирования?

5. Какие модели называют образно-знаковыми? Приведите примеры.

6. Почему образно-знаковые модели не связаны с формами их реализации?

7. Что такое текстовая модель? Приведите примеры.

8. Что такое графическая модель? Приведите примеры.
9. Перечислите виды документальных графических моделей.
10. Что такое чертёжно-графическая модель?
11. Что такое чертёж?
12. Что такое карта?
13. Почему компьютерные модели считаются универсальными?

§ 32. Введение в векторную графику

Вы уже знаете, что компьютерная графика — это технология создания виртуальных графических объектов. Знакомство с основами компьютерной графики у вас состоялось в 8 классе на базе растрового графического редактора Paint.

Напомним, что в растровом графическом редакторе экран подобен листу бумаги, на который художник наносит рисунок красками. Только на экране пикселы, как мельчайшие соты, заполняются не краской, а цветом. Существуют и виды компьютерной графики, основанные на других принципах построения изображения.

Векторная компьютерная графика — технология создания графических объектов из отдельных элементов по типу создания коллажа.

Векторный графический объект собирается на экране компьютера из отдельных элементов, подобно тому как собирается автомобиль из деталей в сборочном цеху. Элементы рисунка могут налегать друг на друга, частично перекрывая друг друга сверху. При этом рисунок всегда можно изменить, т. е. поменять взаимное расположение элементов, изменить или удалить любой элемент.

Название «векторная» такая технология получила из-за особенностей представления элементов графических объектов в памяти компьютера. Для создания векторных графических объектов используются векторные графические редакторы.

Векторный графический редактор, встроенный в текстовый редактор Word 2010, может с успехом использоваться для создания графических объектов и моделей. Будем использовать его и мы.

Для создания графического объекта следует открыть чистую страницу нового электронного документа. Хранить векторные гра-

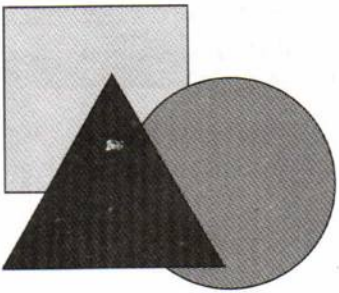


Рис. 45

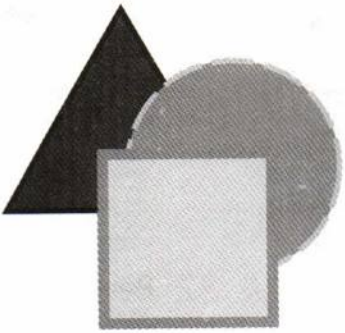


Рис. 46

фические объекты, созданные в редакторе Word, удобно в формате документа Word. При необходимости векторные изображения всегда можно вставить в текстовый документ.

Графические примитивы. Векторные графические редакторы для создания графических объектов используют большое число графических примитивов. Напомним, что графические примитивы — это линии, прямоугольники, эллипсы и другие фигуры, которые можно выводить на экран графического редактора целиком.

Графические примитивы выводятся на экран по одному. Они могут налегать друг на друга, как бумажные фрагменты коллажа (рис. 45). При необходимости всегда можно поменять параметры (атрибуты) любого примитива — его положение, размеры, толщину линий, цвет, угол поворота и т. п. (рис. 46). Любой набор примитивов на экране можно объединить в группу и сформировать тем самым новый примитив. Группу элементов можно, в свою очередь, разгруппировать.

Во встроенном в Word 2010 графическом редакторе графические примитивы носят название *фигуры*.

Меню «Фигуры» и вкладка «Формат». С основным приёмом вставки фигур мы познакомились, изучая текстовый редактор Word 2010. В режиме «Разметка страницы» на вкладке «Вставка» в группе «Иллюстрации» нужно щёлкнуть по кнопке **Фигуры**. Открывается меню «Фигуры» (рис. 47), где щелчком мыши выбирают фигуру и протяжкой мыши выводят её на лист документа.

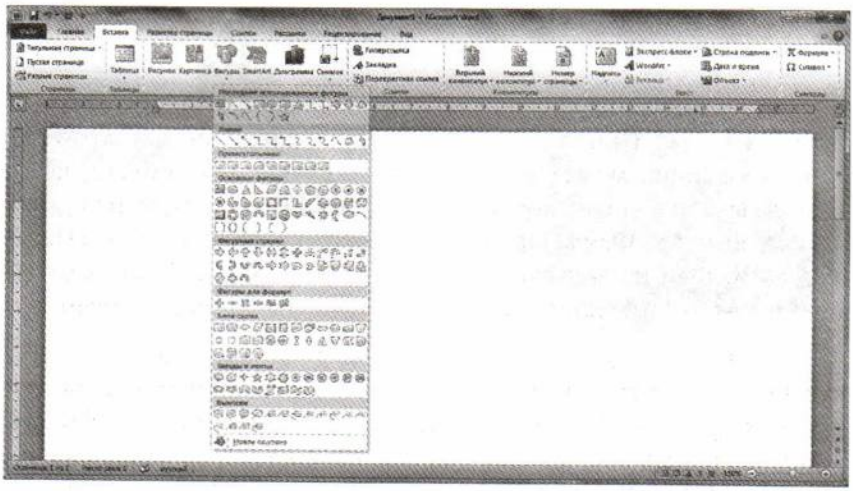


Рис. 47

Меню «Фигуры» содержит 9 разделов. В этих разделах, начиная со второго, размещены группы фигур с определёнными свойствами: линии, прямоугольники и т. п. В первом разделе собраны последние использованные фигуры из других разделов. Названия фигур в меню можно узнать с помощью всплывающих подсказок.

После вывода на лист документа фигура остаётся выделенной, что проявляется в наличии на ней цветных маркеров. Одновременно с появлением фигуры на ленте появляется и новая вкладка «Формат» («Средства рисования»), которая содержит инструменты для изменения атрибутов фигуры.

В группе «Вставка фигур» размещено поле с фигурами, которое позволяет раскрывать меню «Фигуры» на этой вкладке.

Группа «Стили фигур» содержит инструменты для изменения цвета и стиля заливки фигуры, а также цвета, стиля и толщины её контура. Инструмент **Эффекты фигур** позволяет добавить для фигуры тень, отражение, свечение, объём.

В группе «Размер» при необходимости можно задать точный размер фигуры по вертикали и по горизонтали.

Инструменты группы «Упорядочить» позволяют задать точное положение фигуры на листе и способ её обтекания текстом. Инструмент **Повернуть** этой группы (правая нижняя кнопка) позволяет повернуть фигуру вокруг её центра на определённое целое число градусов, отразить её сверху вниз или справа налево.

Особенности вывода графических примитивов. После вывода графический примитив остаётся выделенным и появляется вкладка «Формат».

Замечание. При создании векторных рисунков на экран без предупреждения может выводиться прямоугольный объект, который называется «полотно». Вывод полотна можно отключить, если командой меню **Файл|Параметры** вызвать диалоговое окно «Параметры Word» и на вкладке «Дополнительно» убрать флажок около пункта «Автоматически создавать полотно при вставке автофигур».

Чтобы снять выделение выведенной фигуры, нужно щёлкнуть курсором по свободному месту страницы. Но если необходимо вывести несколько фигур подряд, то выделение первой фигуры снимать не следует. Остальные фигуры удобнее выводить с помощью группы «Вставка фигур» на вкладке «Формат», а снятие выделения автоматически удаляет вкладку «Формат» и открывает вкладку «Главная». Следует также помнить о серьёзном отличии операции вывода примитивов в редакторе Word от аналогичной операции в редакторе Paint.

Внимание! Во встроенном графическом редакторе примитив после его выбора можно вывести на экран только один раз.

Это означает, что если, например, надо вывести четыре линии подряд, то придётся четыре раза щёлкнуть по кнопке **Линия** и совершить вывод примитива.

Клавиши **Ctrl** и **Shift** клавиатуры увеличивают возможности вывода примитивов. Если при выводе примитива удерживать на клавиатуре клавишу **Shift**, то:

- линии и стрелки строятся по фиксированным направлениям, которые отличаются друг от друга на 45°;
- прямоугольник выводится как правильный квадрат;
- овал выводится как правильная окружность.

Если при выводе примитива удерживать на клавиатуре клавишу **Ctrl**, то начальная точка вывода становится серединой выводимого примитива (примитив «вырастает» в разные стороны одновременно). Клавиши **Shift** и **Ctrl** клавиатуры можно удерживать одновременно.

Выделение, перемещение и изменение графических примитивов. Чтобы изменить атрибуты графического примити-

ва, его надо выделить. Эта операция хорошо знакома нам по другим редакторам, но в данном редакторе имеет свои особенности.

Чтобы выделить примитив, надо установить на него указатель мыши. При этом особенно внимательно надо следить за формой указателя. Попадая на примитив, он принимает вид стрелки с перекрестием:  . При налегании одних примитивов на другие добиться этого не всегда просто. Щелчок левой клавишей мыши в этот момент выделяет примитив — появляются его маркеры.

Положение выделенного примитива на экране меняется перетаскиванием курсора мыши. «Зацепиться» за примитив можно только в случае, когда курсор мыши имеет вид стрелки с перекрестием:  . Можно использовать и клавиши-стрелки клавиатуры.

Если при перемещении удерживать клавишу **Shift** клавиатуры, то перемещение из исходного положения возможно только по горизонтали или по вертикали.

Выделение фигуры отменяется щелчком по свободному от фигур месту листа или нажатием клавиши **Esc** клавиатуры.

Изменить выделенный примитив можно протяжкой мыши, уцепившись курсором за один из маркеров. При этом мышиный курсор должен иметь вид двунаправленной стрелки.

Изменить выделенный примитив можно и с помощью кнопки **Изменить фигуру** (правая верхняя пиктограмма на вкладке «Формат» в группе «Вставка фигур»). В меню этой кнопки пункт **Изменить фигуру** позволяет заменить выделенную фигуру на другую. Пункт же **Начать изменение узлов** переводит редактор в новый режим: на контуре фигуры появляются чёрные метки, перемещение которых может изменить контур до неузнаваемости.

Изменение атрибутов графических примитивов. Все фигуры выводятся на экран с контурами заданного по умолчанию цвета и толщины, а также с заливкой определённого цвета. Все эти атрибуты можно поменять. Для этого служат инструменты группы «Стили» на вкладке «Формат».

Поле «Стили» содержит экспресс-стили — образцы сочетаний цвета и толщины контура, а также цвета заливки фигур.

Кнопка **Заливка фигуры** (её правая часть) открывает меню с шаблонами цвета, градиентных заливок и текстур. Пункты этого меню позволяют изменить внутренность фигуры и даже сделать её прозрачной (пункт **Нет заливки**).

Кнопка **Контур фигуры** открывает меню с шаблонами цвета, толщины и стиля (пункт **Штрих**). Пункты этого меню позволяют изменить цвет и толщину контура, его стиль (штриховая линия) или не отображать контур совсем (пункт **Нет контура**).

Пункт **Стрелки** меню кнопки **Контур фигуры** открывает шаблоны стиля стрелок и позволяет превратить любую линию в стрелку или поменять вид стрелок.

Заметим, что для выделенного примитива графический редактор сам определяет перечень атрибутов, которые доступны для изменения.

■ Упражнение 89

Построим прямоугольник и выделим его. Просмотрим меню кнопки **Контур фигуры**. Пункт **Стрелки** этого меню имеет специфическое «туманное» изображение, и воспользоваться им невозможно.

Устанавливать вручную новые атрибуты для каждого примитива не обязательно. Новые атрибуты одной фигуры можно перенести на другую фигуру с помощью виртуальной кнопки **Формат по образцу** (на вкладке «Главная» в группе «Буфер обмена»). С помощью этой кнопки мы уже переносили атрибуты шрифта и стиль абзаца при обработке текстовой информации.

Копирование и удаление графических примитивов. Если в рисунке должны присутствовать несколько примитивов одного типа, то пользоваться каждый раз кнопками панели не обязательно. Достаточно вывести один примитив, а остальные получить копированием (практически аналогично действиям в редакторе Paint).

Перед копированием примитив на экране надо выделить. Затем курсором мыши нужно «зацепиться» за примитив, удерживая клавишу **Ctrl** клавиатуры. Около курсора появляется небольшой знак «плюс». Далее курсор (с копией примитива) перетаскиванием мыши отводят в другую часть экрана и клавишу отпускают.

Если при копировании удерживать одновременно клавиши **Shift** и **Ctrl** клавиатуры, то перемещение копии возможно только по горизонтали или вертикали относительно оригинала. Этот приём удобен при создании схем.

Чтобы удалить примитив с экрана, его выделяют и нажимают клавишу **Del** (**Delete**) клавиатуры.

Вращение, поворот и отражение примитивов. Встроенный графический редактор позволяет произвольно вращать примитивы вокруг их центра. Примитив перед этим нужно выделить. Один из маркеров имеет вид зелёного кружочка; уцепившись за него, примитив вращают протяжкой мыши.

Поворот на 90° и симметричное отражение выделенного примитива также производятся с помощью инструмента **Повернуть** (в виде кнопки с пиктограммой на вкладке «Формат» в группе «Упорядочить»). В меню кнопки **Повернуть** выбирают соответствующий пункт. Выбор пункта **Другие параметры поворота...** открывает диалоговое окно «Разметка», в котором на вкладке «Размеры» можно установить угол поворота фигуры в градусах. Здесь же можно установить точный размер фигуры и масштаб её отображения в процентах.

Вопросы и задания

1. Что такое компьютерная графика?
2. Что такое векторная компьютерная графика?
3. Что такое графические примитивы?
4. Как вызвать на экран вкладку «Формат»?
5. Сколько инструментов содержит группа «Стили» вкладки «Формат»? Каково их назначение?
6. Чем различаются порядки вывода примитивов одного типа во встроенном графическом редакторе и в растровом редакторе Paint?
7. Как влияет на вывод примитива удержание нажатой клавиши **Shift** клавиатуры?
8. Как влияет на вывод примитива удержание нажатой клавиши **Ctrl** клавиатуры?
9. Как выделить примитив на экране?
10. Как переместить примитив по экрану?
11. Как влияет на перемещение примитива удержание нажатой клавиши **Shift** клавиатуры?
12. Как изменить размеры примитива?
13. Как сделать прозрачной внутреннюю область примитива?
14. Как делается невидимым контур примитива?
15. Опишите порядок копирования примитивов.

16. Как влияет на копирование одновременное удержание нажатыми клавиш **Shift** и **Ctrl** клавиатуры?

17. Как удалить примитив с экрана?

18. Каков порядок вращения примитива вокруг центра?

19. Какой инструмент на вкладке «Формат» связан с операцией симметричного отражения примитива?

20. Создайте векторный компьютерный вариант композиции из квадратов и кругов (рис. 48).

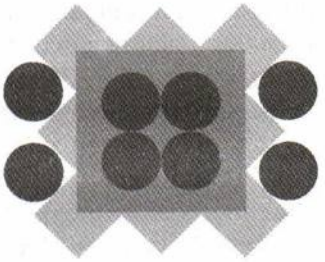


Рис. 48

§ 33. Построение рисунков и схем средствами векторной графики

Графические модели многих типов можно с успехом создавать с помощью растровых графических редакторов. Но векторные графические редакторы хороши тем, что в них может иметься много сложных фигур-заготовок.

В векторном графическом редакторе, встроенном в текстовый редактор Word 2010, фигуры-заготовки собраны в нескольких разделах меню кнопки **Фигуры** (на вкладке «Вставка» в группе «Иллюстрации»). Например, в разделе «Блок-схема» размещены заготовки для построения блок-схем алгоритмов. Раздел «Фигурные стрелки» содержит заготовки для описания любых процессов, в том числе информационных.

В некоторых фигурах после вывода и выделения появляются дополнительные маркеры жёлтого цвета. Изменение относительного положения этих маркеров перетаскиванием указателя мыши приводит к изменению формы фигуры.

Трудности при выводе могут возникнуть только с тремя последними фигурами-линиями в разделе «Линии» (меню кнопки «Фигуры»), так как они имеют разные режимы вывода.

Это фигуры «Рисованная кривая», «Полилиния» и «Кривая», если рассматривать их справа налево.

Режим «Рисованная кривая» является прямым аналогом инструмента «Карандаш» в редакторе Paint.

В режиме «Полилиния» строится ломаная линия из прямых отрезков. При её построении курсором мыши щёлкают в началь-

ной точке и в точках изломов. Этот режим несколько похож на режим построения с использованием инструмента «Многоугольник» в редакторе Paint. Завершают ввод ломаной последним щелчком по её начальной точке (ломаная замыкается) или двойным щелчком в конечной точке (ломаная остаётся разомкнутой).

В режиме «Кривая» строится гладкая линия (без изломов). При построении щёлкают курсором в точках предполагаемого расположения кривой. Через отмечаемые точки проходит кривая линия, которая ведёт себя как стальная упругая проволока, автоматически изгибаясь на поворотах. Завершают ввод щелчком по начальной точке (кривая замыкается) или двойным щелчком в конечной точке (кривая остаётся разомкнутой).

Кроме фигур (линий и заготовок) элементом векторного рисунка может быть фрагмент растрового рисунка (построенного или сканированного) или цифровой фотографии, вставленный через буфер обмена, например из графического редактора Paint.

Группировка и разгруппировка элементов рисунка. Векторный рисунок из примитивов собирается постепенно. Количество примитивов при этом растёт, образуя целую систему. Чтобы избежать возможных случайных сдвигов этой системы, результаты построений целесообразно время от времени фиксировать.

Фиксацию фрагментов векторного рисунка выполняет операция **Группировка**. Фигуры, предназначенные для группировки, выделяют последовательными щелчками левой клавишей мыши с удержанием клавиши **Shift** или **Ctrl** клавиатуры. Далее на вкладке «Формат» в группе «Упорядочить» щёлкают по кнопке **Группировать**. Группа фигур объединяется в одну сложную фигуру.

Далее рисунок продолжают собирать из графических примитивов и на определённом этапе группируют новый фрагмент. Готовый рисунок также целесообразно сделать единой фигурой.

Можно воспользоваться и другим способом проведения операции группировки. После выделения группы фигур по одной из них щёлкают правой клавишей мыши. Появляется контекстное меню, в котором имеется пункт-кнопка **Группировать**. После щелчка по этой кнопке появляется меню с пунктом **Группировать**, по которому опять-таки надо щёлкнуть.

Выделение с группы фигур всегда можно снять щелчком по свободному от этих фигур месту на листе или нажатием клавиши **Esc**

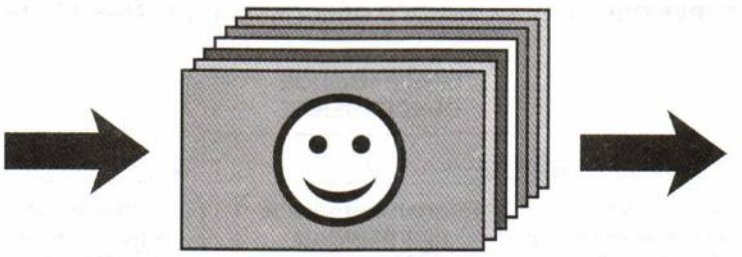


Рис. 49

клавиатуры. Выделение отдельной фигуры в группе снимается повторным щелчком мыши (с нажатой клавишей **Shift** или **Ctrl**).

Если фигура является группой более простых фигур, то её всегда можно разгруппировать. Для этого сложную фигуру выделяют, на вкладке «Формат» в группе «Упорядочить» щёлкают по кнопке **Группировать** и в меню выбирают пункт **Разгруппировать**.

Если после выделения фигуры на рисунке щелчок по кнопке **Группировать** (на вкладке «Формат» в группе «Упорядочить») не открывает меню (эта кнопка имеет тусклое изображение), то разгруппировать эту фигуру нельзя.

Замечание. После команды разгруппировки все входившие в данный фрагмент элементы остаются выделенными. Чтобы избежать нежелательных последствий, выделение следует снять, щёлкнув левой клавишей мыши за пределами рисунка.

■ Упражнение 90

Построим графическое изображение модели информационного процесса, которое приведено на рисунке 49 и основу которого составляет пакет из семи разноцветных прямоугольников.

Открываем новый документ. Выводим примитив дальнего прямоугольника. Заливаем его внутренность красным цветом. Затем выделяем его и копируем со сдвигом шесть раз. Каждый новый прямоугольник при этом перекрывает предыдущий.

Последовательно выделяя каждый прямоугольник, меняем цвет его заливки. Добавляем фигуры — рожицу и стрелку и заливаем их бирюзовым цветом. Копию стрелки перемещаем на другую сторону рисунка. В завершение выделяем все элементы схемы и группируем их.

СТРУКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ КОМПЬЮТЕРА

Рис. 50

Вывод текстовых надписей. Вывести текстовую надпись в любом месте векторного объекта можно с помощью фигуры «Надпись». Эта фигура находится в разделе «Основные фигуры» меню кнопки **Фигуры** на вкладке «Вставка» и подобна фигуре «Прямоугольник» со всеми его атрибутами.

После вывода фигуры «Надпись» на экран в прямоугольнике, окаймлённом штриховкой, появляется текстовый курсор. Нужно установить параметры абзаца и атрибуты шрифта, как это делается при вводе текстовых документов, и ввести текст. Чтобы прямоугольник сделать невидимым, нужно установить атрибут **Нет заливки** для заливки фигуры и атрибут **Нет контура** для контура фигуры. Текст остаётся видимым.

Выводить текст можно и во внутренних областях фигур с помощью пункта **Добавить текст** контекстного меню, которое вызывается щелчком по фигуре правой клавишей мыши.

■ Упражнение 91

Построим графическое изображение схемы на рисунке 50. Эта схема была изучена нами в 8 классе.

Открываем новый документ и выводим фигуру «Надпись» для названия схемы. Устанавливаем параметры абзаца, атрибуты шрифта и вводим текст. Щёлкаем по контуру прямоугольника и устанавливаем атрибуты **Нет заливки** и **Нет контура**.

Выводим фигуру «Скруглённый прямоугольник» как верхний элемент схемы. Устанавливаем бирюзовый цвет заливки. Щелчком правой клавиши мыши вызываем контекстное меню, в котором щелчком левой клавиши мыши выбираем пункт **Добавить текст**. Устанавливаем параметры абзаца, включая межстрочный интервал, и атрибуты шрифта. Вводим текст. При необходимости увеличиваем размеры фигуры.

Копированием получаем фигуру второго ряда. Меняем её размеры и текст надписи. Далее копированием (с удержанием нажатой клавиши **Shift** клавиатуры) получаем ещё две фигуры второго ряда, в которых меняем только текст. Осталось копированием получить фигуру третьего ряда, изменить её размеры и текст, а затем скопировать её и изменить текст, чтобы получить вторую фигуру третьего ряда.

Выводим стрелки, точно размещаем их и подбираем их длину. Для точности стыков выделенные примитивы можно сдвигать по экрану клавишами-стрелками клавиатуры. В завершение выделяем все элементы схемы и группируем их.

Изменение порядка слоёв с фигурами. При выводе на экран фигуры могут налегать друг на друга. Особенность состоит в том, что каждая фигура находится на экране в отдельном виртуальном слое. Каждая новая фигура выводится в новом слое, который располагается выше всех остальных.

Положение слоя с фигурой можно менять. В её контекстном меню есть пункты **На передний план** и **На задний план**. Если задержать курсор на одном из этих пунктов, то открывается дополнительное меню, в котором тоже делают выбор. Например, **Переместить вперёд** означает перемещение вперёд на 1 слой. Кнопка **Область выделения** на вкладке «Формат» в группе «Упорядочить» открывает на экране область со списком выведенных фигур (слоёв), и кнопками эти фигуры перемещают по данному списку.

■ Упражнение 92

Составим компьютерную модель цветка (рис. 51) с красными лепестками, жёлтой сердцевиной и зелёными стеблем и листьями.

Открываем новый документ. Для построения первого лепестка (рис. 52, *а*) воспользуемся фигурой «Кривая». На рисунке 52, *б* показаны точки щелчков курсором при её выводе. Последний щел-

чок надо сделать в начальной точке, и контур замкнётся. Затем устанавливаем красный цвет для контура и для заливки (рис. 52, *в*).

Делаем шесть копий лепестков. Поворачиваем лепестки и собираем из них соцветие (рис. 53). Выделяем все элементы и группируем их.

Выводим окружность, устанавливаем жёлтый цвет заливки и линии. Перетаскиваем образовавшийся круг на место сердцевины цветка (рис. 54). Опять группируем все элементы.

С помощью фигуры «Кривая» выводим стебель и левый листок (рис. 55). При выводе в нижней точке стебля делаем двойной щелчок. Устанавливаем зелёный цвет линии и максимальную толщину. Контур листка замыкаем последним щелчком в начальной точке вывода. При необходимости подправляем его форму в режиме изменения узлов. Для листка используем зелёный цвет заливки и контура.

Рис. 51

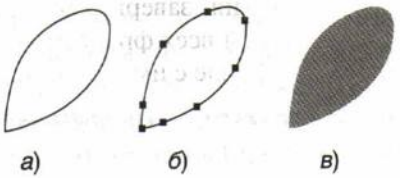


Рис. 52

Рис. 53

Рис. 54

Рис. 55

Рис. 56

Листок копируем и копию симметрично отражаем слева направо. Собираем и группируем стебель с листьями (рис. 56).

Теперь собираем весь цветок. При этом оказывается, что стебель показывается на экране лежащим на соцветии (рис. 57). Это объясняется тем, что последняя выведенная фигура всегда находится в самом верхнем слое. Переводим стебель в самый нижний слой. Для этого выделяем его, раскрываем контекстное меню и в нём выбираем пункт **На задний план**.

Построения завершаем окончательной группировкой всех фрагментов. Сохраняем рисунок в файле с именем ex82.doc.

Рис. 57

Двухмерные графические объекты с тенями. Для добавления тени к двухмерной фигуре используется диалоговое окно «Формат фигуры» (вкладка «Тень»).

По фигуре нужно щёлкнуть правой клавишей мыши и в контекстном меню выбрать пункт **Формат фигуры**. Открывается окно «Формат фигуры». На вкладке «Тень» в меню кнопки **Заготовки** выбирают тип тени или атрибут «Нет тени».

■ Упражнение 93

Добавим тень к фигуре «Солнце» (вкладка «Вставить», меню кнопки **Фигуры**, раздел «Основные фигуры»).

В новом документе выводим фигуру на экран и вызываем окно «Формат фигуры». На вкладке «Тень» в меню кнопки **Заготовки** в разделе «Снаружи» выбираем левый верхний тип тени.

Для некоторых объектов доступны не все возможные типы теней.

Упражнение 94

Создадим компьютерную модель дорожного знака «Пересечение с основной дорогой» с тенью (рис. 58). Знак представляет собой белый треугольник с широкой красной каймой на белом столбике.

В новом документе на вкладке «Вставка» в меню кнопки **Фигуры** в разделе «Основные фигуры» находим фигуру «Равнобедренный треугольник». С удержанием нажатой клавиши **Shift** треугольник получается равносторонний. Вращением устанавливаем треугольник в нужное положение. Для треугольника устанавливаем белый цвет заливки, максимальную толщину (6 пт) и красный цвет контура. Выводим фигуру «Прямоугольник» для столбика и устанавливаем для него белый цвет заливки, чёрный цвет контура и толщину контура 0,5 пт. Составляем знак (рис. 59).

Чтобы треугольник располагался визуально ближе столбика, щелчком правой клавиши мыши по треугольнику вызываем контекстное меню и в нём выбираем пункт **На передний план**.

Выделяем обе фигуры и щёлкаем по одной из них правой клавишей мыши. В контекстном меню выбираем пункт **Группировать**. В новом выпадающем меню снова выбираем пункт **Группировать**.

С помощью контекстного меню открываем диалоговое окно «Формат фигуры». На вкладке «Тень» в меню кнопки **Заготовки** в разделе «Перспектива» выбираем средний шаблон тени. На той же вкладке движком **Прозрачность** можно усилить густоту тени (20%). Сохраняем рисунок в файле ex84.docx.

Двухмерные графические объекты с добавлением объёма. За добавление объёма к фигуре отвечают вкладки «Фор-

Рис. 58

Рис. 59

мат объёмной фигуры» и «Поворот объёмной фигуры» диалогового окна «Формат фигуры».

Для фигуры с помощью контекстного меню вызывается диалоговое окно «Формат фигуры». Обычно объём добавляется перпендикулярно плоскости фигуры, поэтому без поворота объём не виден. Поворот осуществляется на вкладке «Поворот объёмной фигуры» с помощью шаблонов в меню кнопки **Заготовки** или заданием значений углов в разделе «Поворот». Глубина (высота) добавляемого объёма (в пунктах) устанавливается на вкладке «Формат объёмной фигуры» в разделе «Глубина».

■ Упражнение 95

Создадим объёмную компьютерную модель секции забора-штакетника на металлических столбиках (рис. 60).

Для столбика выводим фигуру «Овал» в виде круга диаметром 1 см (серые контур и заливка). Выводим окно «Формат фигуры». На вкладке «Поворот объёмной фигуры» в меню кнопки **Заготовки** выбираем шаблон «Вне оси 1, вверх». На вкладке «Формат объёмной фигуры» устанавливаем глубину 250 пт и угол 30° для освещения (рис. 61). Копированием создаём второй столбик и переносим его параллельно вправо.

Создаём перекладину как горизонтальный вытянутый прямоугольник. Цвет контура и заливки — светло-оранжевый. В окне «Формат фигуры» в меню кнопки **Заготовки** выбираем для поворота шаблон «Наклон вправо вверх». Устанавливаем глубину объёма — 15 пт, а угол освещения 150°. Меняем размеры переклa-

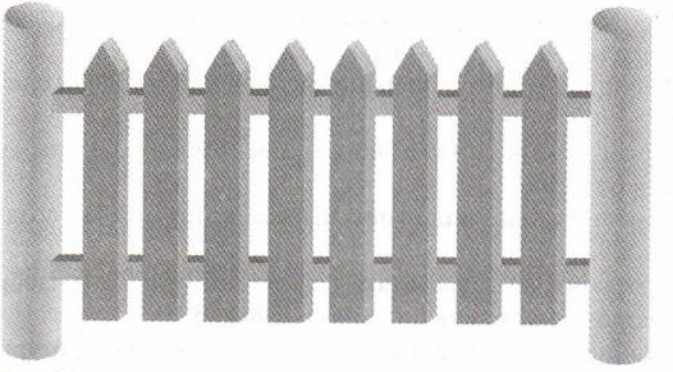


Рис. 60

дины и помещаем её на столбики. Копированием получаем вторую перекладину и так же размещаем её.

Рис. 61

Выводим прямоугольник для штакетника. Сверху к нему пристраиваем фигуру «Треугольник». Группируем элементы доски для штакетника. Атрибуты деревянной перекладины переносим на доску для забора. Для этого выделяем перекладину, затем последовательно щёлкаем по кнопке **Формат по образцу** на вкладке «Главная» и по доске для забора.

Копированием получаем остальные доски и размещаем их на перекладинах. Все элементы рисунка группируем. Рисунок сохраняем в файле ex85.docx.

☐ Вопросы и задания

1. Для чего служат жёлтые маркеры, которые появляются в фигурах после выделения?

2. Как в векторном графическом редакторе, встроенном в текстовый редактор Word 2010, можно выделить на экране несколько фигур; все фигуры?

3. В чём состоит смысл группировки фигур?

4. В каком порядке проводится операция группировки нескольких фигур?

5. Где размещена кнопка **Группировать**?

6. В каком порядке проводится операция разгруппировки сложных фигур?

7. Какие возможности предоставляет встроенный графический редактор для вывода текстов на экран?

8. Как производится операция изменения порядка слоёв с фигурами?

9. Какая вкладка диалогового окна «Формат фигуры» отвечает за работу с тенью?

10. Добавьте тень в рисунок цветка, который был создан в упражнении 92.

11. Создайте во встроенном редакторе Word 2010 изображения других дорожных знаков подобно рисунку 58.

12. Какие вкладки диалогового окна «Формат фигуры» отвечают за добавление объёмов к двумерным примитивам?

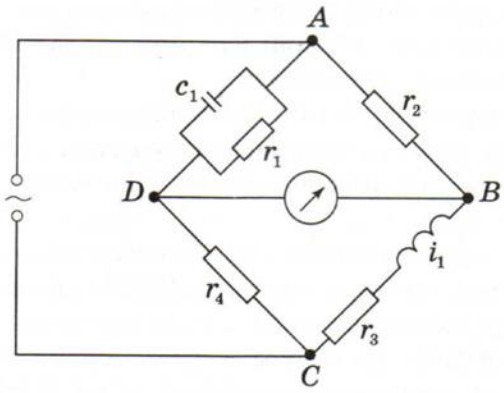


Рис. 62

13. Создайте во встроенном редакторе Word 2010 объёмную композицию доски для игры в шашки с шашками на ней.

14. Попробуйте во встроенном редакторе Word 2010 создать композицию, которая изображена на виртуальной кнопке **Фигуры** на вкладке «Вставка» в группе «Иллюстрации».

15*. Создайте во встроенном редакторе Word 2010 электрическую схему моста для определения индуктивности катушки (рис. 62).

16*. Найдите в библиотеке или в Интернете описание головоломки «Пентамино», создайте во встроенном векторном редакторе Word 2010 набор её элементов и попробуйте решить головоломку, переставляя и вращая элементы прямо на экране компьютера.

17*. Создайте во встроенном графическом редакторе Word 2010 изображение клавиатуры компьютера.

18*. Шрифтовые объекты WordArt на вкладке «Вставка» связаны с кнопкой **WordArt** в группе «Текст». Составьте в форме графического объекта приглашение на свой день рождения, используя объекты WordArt и картинки из компьютерной коллекции.

§ 34. Компьютерная модель размещения

Задачи размещения. На стадии проектирования в машиностроении и строительстве, да и в быту часто возникает задача, когда объёмные объекты нужно разместить на ограниченном пространстве. Например, когда нужно совместить несколько объ-

ёмных элементов в проектируемом механизме или разбить строительное пространство стенами на комнаты. Задачи такого рода называются задачами размещения.

Часто, например, возникает задача размещения мебели в комнате или оборудования в производственном помещении. Рассмотрим подробно задачу расстановки мебели, которая является по сути дизайнерской.

Задача расстановки мебели может решаться непосредственной её переноской. Понятно, что это не самый рациональный способ. В уме можно составить только основную идею перестановки (мысленную модель). Но конкретная расстановка сильно зависит от размеров предметов мебели (войдёт — не войдёт). Для этих же целей можно использовать масштабные плоские макеты комнаты и предметов мебели (материальную модель). Макеты надо вырезать из бумаги и подобрать их наиболее подходящую раскладку.

Но рациональнее всего использовать в задачах размещения компьютерные модели. Недаром все производители индивидуальных наборов мебели имеют мощные компьютерные программы, которые выдают в результате разработки не только план размещения предметов мебели, но и их объёмные компьютерные изображения.

Компьютерная модель размещения. Для создания компьютерной модели размещения воспользуемся возможностями векторного редактора, встроенного в текстовый редактор Word.

■ Упражнение 96

Пусть имеется комната, план которой представлен на рисунке 63. Создадим план расстановки набора мебели, включающего секционную мебель (3 секции и тумба под телевизор с размерами оснований 100×42 см), два дивана (200×90 см и 160×90 см), журнальный столик (70×70 см), круглый стол (диаметр 110 см) и четыре стула при нём.

Запускаем текстовый редактор и открываем новый документ. Работать будем в следующем порядке.

1. Настройка параметров страницы в окне редактора и подбор масштаба отображения.

2. Создание векторной фигуры плана комнаты и подбор размеров этой фигуры.
3. Создание векторных фигур для предметов мебели и подбор их размеров.
4. Подбор походящей расстановки мебели и создание плана.

Этапы 1—3 связаны с построением компьютерной модели, а этап 4 — с её использованием. Итак, будем действовать в соответствии с представленным алгоритмом.

1. На вкладке «Разметка страницы» в группе «Параметры страницы» щелчком по кнопке **Поля** открываем меню и щёлкаем по пункту **Настраиваемые поля...** Открывается диалоговое окно, в котором на вкладке «Поля» устанавливаем альбомную ориентацию и размеры полей (левое — 2,5 см, остальные — 1 см).

На вкладке «Вид» устанавливаем режим просмотра «Разметка страницы» и масштаб «По ширине страницы».

Командой меню **Файл|Параметры...** открываем диалоговое окно «Параметры Word». На вкладке «Дополнительно» в разделе «Экран» в поле «Единицы измерения:» устанавливаем «Сантиметры» и убираем флажок для пункта «Отображать пикселы для средств HTML».

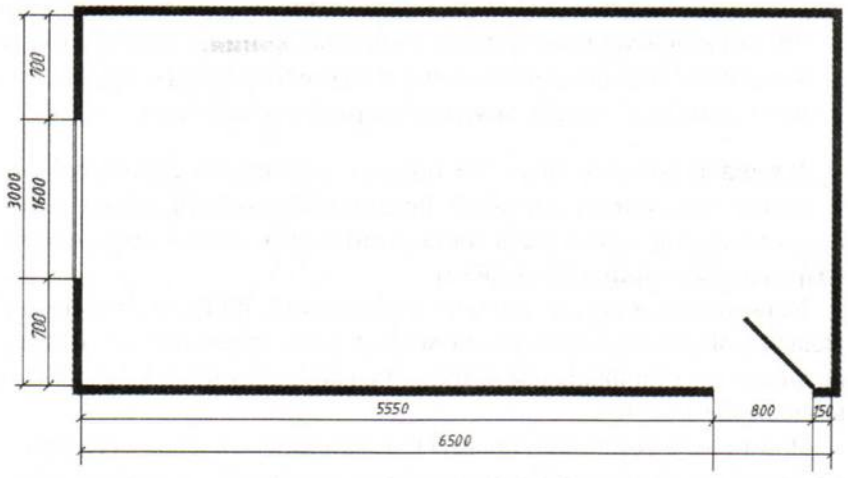


Рис. 63

Комната имеет длину 6,5 м, а виртуальный лист (между границами полей) — чуть более 27 см. Выбираем масштаб таким, чтобы план комнаты занял на листе как можно больше места. Понятно, что удобнее оперировать целыми числами, поэтому можно брать масштаб 4 см : 1 м или 1 : 25. Продолжим настройку.

Для удобства построений выводим на экран вспомогательную сетку. Установим шаг сетки в 2 см или 0,5 м (с учётом масштаба). Для этого на вкладке «Разметка страницы» в группе «Упорядочить» щёлкаем по кнопке **Выровнять**. В открывшемся меню выбираем пункт **Параметры сетки**. Появляется диалоговое окно «Привязка к сетке». В диалоговом окне:

- в разделе «Привязка объектов» флажок не нужен;
- в разделе «Шаг сетки» в полях записываем по 2 см;
- в разделе «Начало сетки» флажок нужен;
- в разделе «Показать сетку» нужны только первые два флажка, а каждое поле должно содержать по единице.

Щелчок по кнопке **ОК** закрывает диалоговое окно «Привязка к сетке», и сетка появляется на экране. Сетку можно сделать невидимой, если на вкладке «Вид» в разделе «Показать» убрать флажок возле пункта **Сетка**.

❷ Выводим фигуру прямоугольника, чтобы её левый верхний угол совпал с верхним левым углом сетки. Делаем внутренность прямоугольника прозрачной, а толщину контура задаём равной 6 пт. Далее, изменяя положение и размеры контура, устанавливаем с помощью сетки внутренние размеры комнаты.

Для изображения дверного проёма используем белый прямоугольник без линий, который перекрывает контур стены. Этот прямоугольник может быть достаточно узким, а его длину установим в соответствии с масштабом.

Установить точный размер выделенной фигуры можно на вкладке «Формат» в группе «Размер». В поле инструмента «Ширина фигуры» устанавливаем ширину прямоугольника в 3,2 см (80 см в масштабе 1 : 25).

Для точной установки проёма (расстояние от правой стены равно 15 см) воспользуемся вспомогательным «мерным» прямоугольником. Выведем прямоугольник на лист и установим его ширину в 0,6 см (15 см в масштабе 1 : 25).

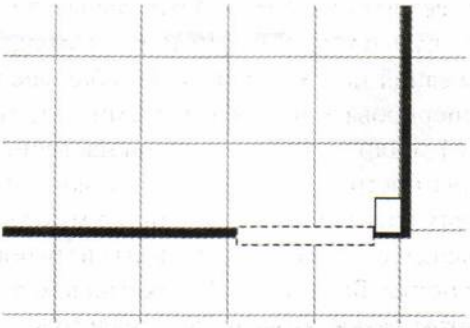


Рис. 64

Далее устанавливаем этот прямоугольник в угол комнаты, левее — дверной проём (рис. 64).

На рисунке фигура дверного проёма специально выделена. Мерный прямоугольник переносим в другое место, поскольку он ещё нам понадобится, чтобы точно установить оконный проём.

По месту на плане (см. рис. 63) устанавливаем дверь в виде отрезка линии толщиной 2,25 пт и окно в виде прямоугольника высотой 0,18 см с белой заливкой.

3 Создаём векторные фигуры для предметов мебели. Можно использовать прямоугольники, овалы и другие фигуры. Несколько предметов мебели есть в компьютерной коллекции картинок, связанной с кнопкой **Картинка** на вкладке «Вставка» в группе «Иллюстрации». Внешние размеры фигур устанавливаются в соответствии с масштабом.

4 Подбираем походящую расстановку мебели. Один из вариантов приведён на рисунке 65.

Полученный план расстановки мебели сохраняем в файле ex86.docx.

Вопросы и задания

1. Опишите ситуации, в которых возникают задачи размещения.

2. Для чего, по вашему мнению, в текстовом редакторе устанавливался сантиметр в качестве единицы измерения?

3. Каким приёмом пользуются, чтобы точно установить на плане изображения дверных и оконных проёмов?

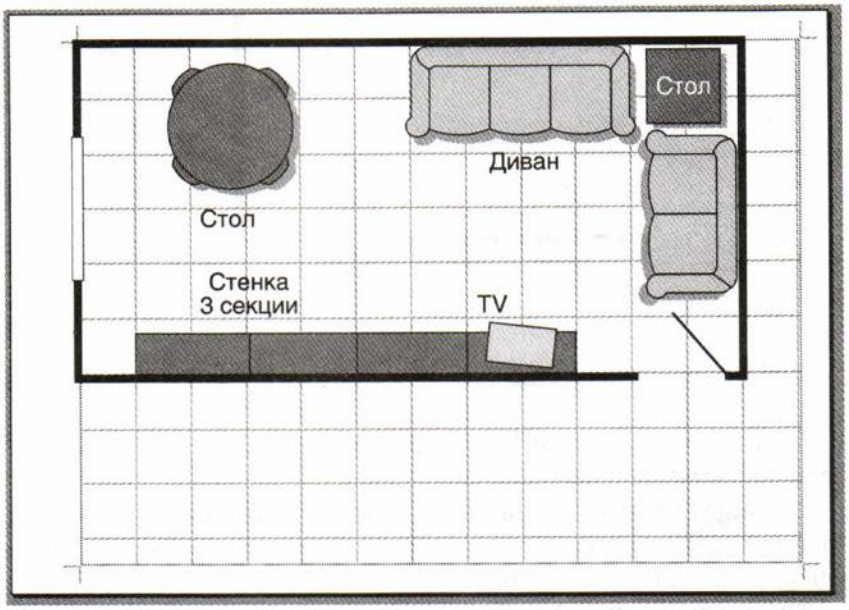


Рис. 65

4. Создайте план расстановки мебели и оборудования в вашем компьютерном классе.

5. Создайте план расстановки мебели в вашей комнате.

6. Создайте план пришкольного участка.

§ 35. Компьютерные методы построения чертежей

Обзор методов. Машиностроительные чертежи определяют конструкцию изделия, содержат информацию, необходимую для его изготовления, монтажа, сборки или ремонта. Не так давно в проектных институтах работала целая армия сотрудников, имевших профессию «чертёжник».

С появлением компьютеров в практику вошли компьютерные реализации чертежей, которые называют электронными чертежами. Теперь построение чертежей — удел инженеров, которые пользуются специальными компьютерными программами. Строить

чертёж с помощью компьютера намного проще, чем вручную чертить линии на бумаге. А при помощи принтера или плоттера электронный чертёж без всяких проблем переносится на бумагу.

Существует достаточно много разновидностей программ для компьютерного черчения, которые различаются принципами построения чертежей и форматами сохраняемой графической информации. Основная масса таких программ использует принцип векторной графики на базе графических примитивов. В профессиональных программах графические примитивы строго позиционируются на виртуальном листе бумаги. Это означает, что их размеры и относительное положение задаются числовыми значениями.

Имеются и программы компьютерного черчения, которые используют другие подходы. В частности, есть программы, которым на входе нужен только компьютерный эскиз изделия, создаваемый в прилагаемом графическом редакторе. При этом на экране вручную задаются опорные точки чертежа (без особой точности в расположении), проводятся линии между опорными точками и задаются размеры. Можно также использовать в качестве фона для создания компьютерного эскиза фотографию изделия. Программа в автоматическом режиме обрабатывает компьютерный эскиз и выдаёт привычное изображение чертежа.

Программы компьютерного черчения чаще всего распространяются на платной основе и требуют значительного времени на освоение. Поэтому при построении электронных чертежей мы воспользуемся уже знакомым нам графическим редактором, встроенным в текстовый редактор Word. С тем, как задавать точные размеры примитивов и их точное относительное положение, мы уже познакомились, создавая компьютерную модель размещения.

Компьютерная реализация требований стандартов. Рассмотрим основные требования стандартов к чертежам, чтобы в дальнейшем учитывать эти требования при построении электронного чертежа.

Чертежи выполняются на листах определённого формата (А0, А1, А2, А3, А4). Мы будем пользоваться форматом А4, чтобы на струйном или лазерном принтере можно было получить документальный вариант чертежа.

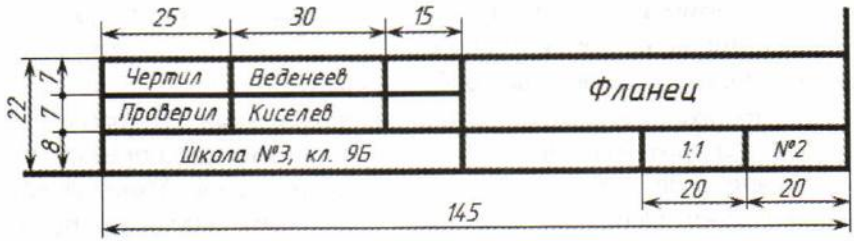


Рис. 66

Стандарты предусматривают наличие на листах рамки с полями (слева — 20 мм, остальные — 5 мм). В правом нижнем углу рамки располагается основная надпись с информацией об изделии, исполнителе и др. Структура, размеры основной надписи для учебных чертежей и пример её заполнения приведены на рисунке 66.

Для надписей в чертежах используется специальный шрифт высотой 2,5; 3,5; 5; 7; 10; 14 мм. Толщина основной линии, включая линии рамки и основной надписи, — от 0,5 до 1,4 мм. Остальные линии должны быть в 2—3 раза тоньше. Линии одного типа должны иметь одинаковую толщину на всём чертеже.

В компьютере с операционной системой Windows принята система измерения толщины линий и высоты шрифта в типографских пунктах (пт). В 1958 г. условились, что английский, или имперский, дюйм (англ. inch) равен 2,54 см. В Англии и в США принято, что 1 пункт = 1/72 дюйма, т. е. 1 пункт = 0,3528 мм. Учитывая пересчёт миллиметров в типографские пункты, для черчения мы будем использовать шрифт высотой (размером) 7, 10, 14, 20, 28 пт. Для основной линии на чертежах выберем толщину 3 пт, а для остальных — 1 пт.

Шрифт для выполнения чертежей действительно имеет существенные отличия от шрифтов, установленных в операционной системе Windows (рис. 67). Здесь на помощь может прийти Интернет. По запросу «чертёжный шрифт» поисковая система выдаст список результатов поиска, в котором обязательно найдутся веб-адреса

Пример использования шрифта для чертежей. 1 2 3 7 8 9 0

Рис. 67

нужных файлов. После скачивания и разархивирования файл со шрифтом следует скопировать в папку WINDOWS\Fonts, и шрифтом можно пользоваться. Надо только запомнить его название. В заготовках, которые размещены на CD-диске, использован чертёжный шрифт GOST 2.304-81.

Постановка задачи и перечень этапов построения. Прежде чем приступить к построению, следует скопировать с прилагаемого CD-диска в рабочую папку файлы form.docx и prim.docx.

■ Упражнение 97

Создадим электронный чертёж корпуса, представленного как эскиз на рисунке 68.

Открываем в текстовом редакторе Word файл prim.docx, который содержит набор чертёжных примитивов. Сразу сохраняем его в файле с именем temp.docx. Этот файл нужен нам временно.

Чертёж будем строить в пять этапов.

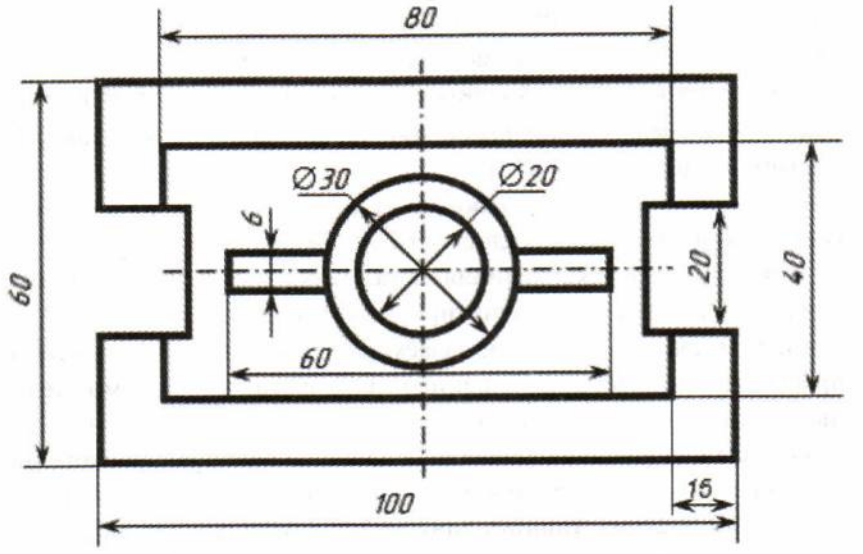


Рис. 68

1. Составление изображения корпуса. Построение осевых линий.
2. Построение выносных линий и размерных стрелок.
3. Создание шрифтовых надписей размеров.
4. Заполнение основной надписи бланка.
5. Размещение чертежа на бланке.

Создание электронного чертежа. Каждый из этапов построения электронного чертежа имеет свои особенности, поэтому опишем их подробно.

1. Чертёж корпуса будет состоять из примитивов прямоугольников и окружностей.

Заготовки квадрата и окружности с фиксированными центрами расположены в верхней части виртуального листа.

Масштаб 1 : 1 позволяет разместить изображение корпуса на бланке. Копию квадрата переводим ниже в рабочую зону и устанавливаем размеры 60×100 (мм) внешнего прямоугольника на чертеже (на вкладке «Формат» в группе «Размер» размеры устанавливаются в см — 6×10). У второй копии квадрата устанавливаем размеры внутреннего прямоугольника 40×80. Совмещаем их центры и группируем. Аналогично накладываем третий прямоугольник размерами 6×60 (рис. 69). Точно таким же образом размещаем копии окружностей.

Заливаем внутренность большой окружности белым цветом. Она должна находиться в слое ниже малой окружности.

Копируя заготовку, удлиняя и перемещая её копии, устанавливаем на чертеже штрихпунктирные осевые линии толщиной 1 пт. Копируем квадрат-заготовку, задаём размеры 20×20 для бокового проёма. Для точности размещения квадрата выводим вспомогательный прямоугольник и задаём размер 15 мм для его ширины. Устанавливаем квадрат правого проёма на место (рис. 70). Аналогично устанавливаем такой же проём слева. Заливаем внутренность проёмов белым цветом. Лишние линии квадратов-проёмов закрываем белыми прямоугольниками. Устанавливаем длину горизонтальной осевой линии. Изображение корпуса построено.

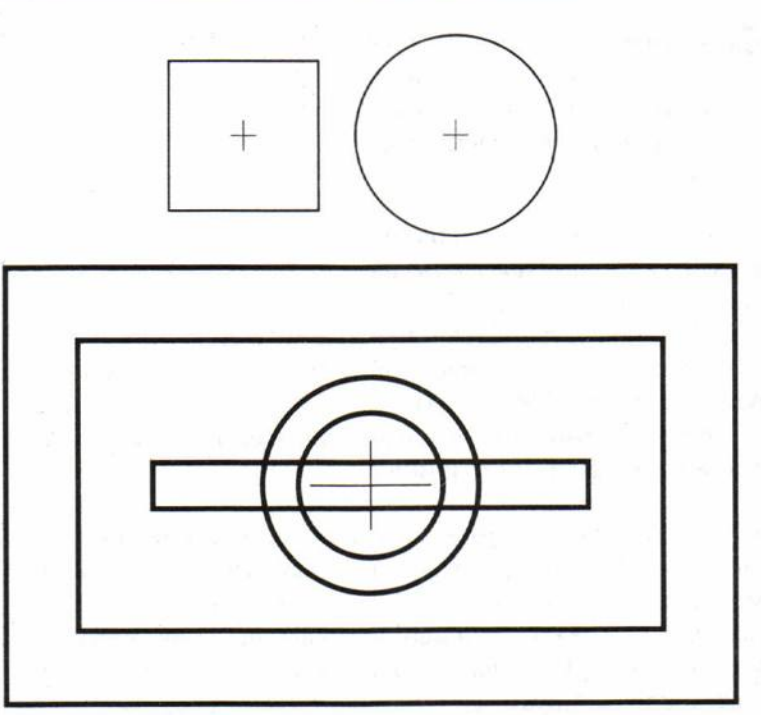


Рис. 69

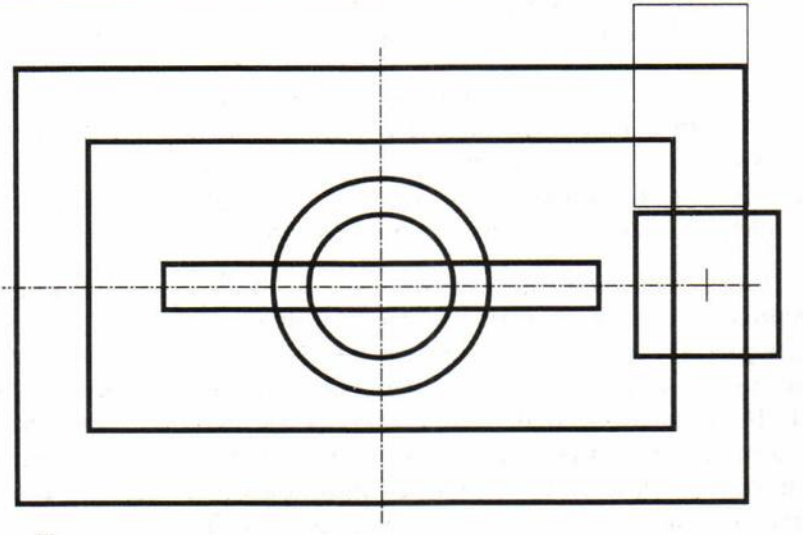


Рис. 70

2 Заготовки наборов из двух выносных линий (толщина 1 пт) и размерной стрелки (той же толщины) представлены на рисунке 71.

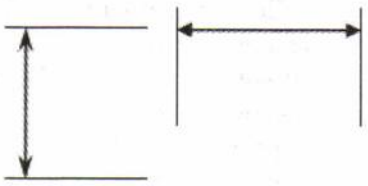


Рис. 71

Заготовки можно копировать, отражать (кнопка **Повернуть** на вкладке «Формат»), менять их размеры, чтобы установить в нужных местах. Для точности установки можно увеличить масштаб изображения. В заключение на чертеже устанавливаются размерные двусторонние стрелки для диаметров окружностей. При необходимости к размерным стрелкам добавляются линии толщиной 1 пт.

3 Построение шрифтовых надписей размеров трудностей не вызывает. Достаточно воспользоваться заготовленными фигурами «Надпись» без заливки и без линий, с курсивом. Если вы используете чертёжный шрифт, отличающийся от GOST 2.304-81, то заготовку «Надпись» нужно выделить и заменить шрифт. Заготовки копируем и в копиях меняем текст. Устанавливаем надписи на чертеже. Все элементы чертежа группируем и получаем электронный чертёж корпуса (см. рис. 68). Чертёж сохраняем (имя временного файла `temp.docx` мы уже задали).

4 Чтобы заполнить основную надпись, открываем в редакторе файл `form.docx` с бланком чертежа. Сохраняем его с именем `ex87.docx`. В бланке использован чертёжный шрифт GOST 2.304-81. Если вы используете другой чертёжный шрифт, то надписи нужно выделить и заменить шрифт.

Заменяя шаблонный текст (его размер 14 пт), вводим фамилию исполнителя в графу правее графы «Чертил», а фамилию учителя в графу правее графы «Проверил». Ниже записываем название школы (можно с сокращением) и через запятую обозначение класса. Запись «1 : 1» означает масштаб. В графе «Материал» вводим название материала изделия (Сталь). Правее графы масштаба расположена графа для номера чертежа. В этой графе шаблонный текст можно удалить. Правее фамилий вносят даты (размер шрифта 10 пт).

Название чертежа «Корпус» запишем шрифтом размера 20 пт.

5 Возвращаемся к документу с изображением корпуса. Копируем его в буфер обмена. Переходим к бланку с заполненной основной надписью. Вставляем изображение из буфера обмена и располагаем его на бланке. Сохраняем результат в файле на диске (имя файла ex87.docx мы уже задали). Теперь чертёж можно распечатывать на принтере.

Вопросы и задания

1. Что такое основная надпись чертежа и где она располагается на чертеже?

2. Назовите форматы бумажных листов, которые используются в черчении.

3. Почему, на ваш взгляд, в текстовом редакторе Word при измерении высоты символов и межстрочных интервалов используется типографская единица измерения — пункт?

4. Почему для построения чертежа толщина основной линии была принята равной 3 пт?

5. Перечислите этапы создания учебного чертежа с помощью текстового редактора Word.

6. В чём при создании чертежа состоит удобство использования вспомогательных фигур с центральными метками?

7. С помощью Интернета найдите обозначения и наименование государственных стандартов, которые регламентируют требования к чертежам.

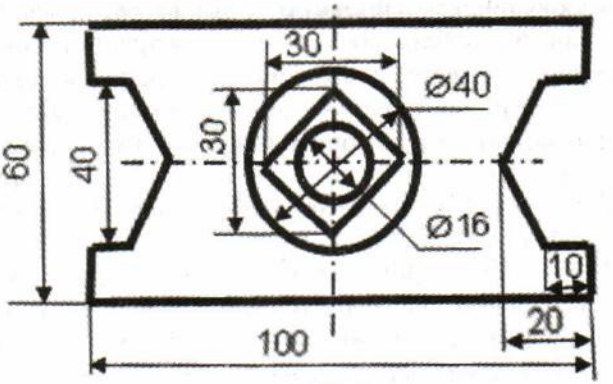


Рис. 72

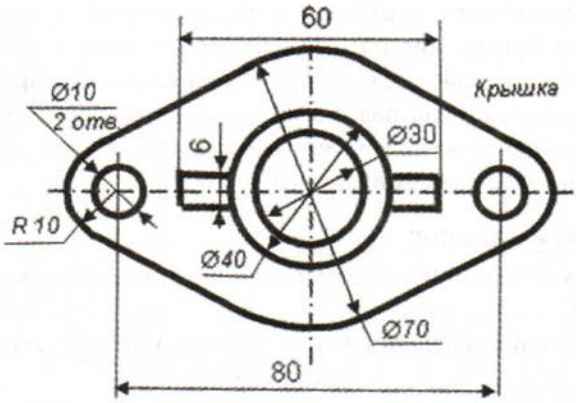


Рис. 73

8. Постройте электронные чертежи по эскизам, показанным на рисунках 72, 73.

9. Одной из разновидностей архитектурно-строительных чертежей являются *фасады* зданий — чертежи внешнего вида одной стороны зданий. Составьте электронный чертёж фасада вашей школы.

§ 36. Введение в трёхмерную графику

Двухмерная и трёхмерная графика. Графические модели (графические объекты) долгое время традиционно создавались на бумаге или другом подходящем плоском носителе. С появлением компьютерной техники графические модели стали строить на экране компьютера с помощью графических редакторов. Электронные графические модели с помощью цветных принтеров в любой момент можно перенести на бумагу практически без потерь качества изображения.

Однако с развитием программного обеспечения появилась возможность построения виртуальных компьютерных объектов другого рода — объёмных, или *трёхмерных*. Именно такими объектами наполнены современные компьютерные игры. Компьютерными трёхмерными объектами стали и персонажи мультфильмов.

В отличие от традиционной «плоской» графики, виртуальные объёмные объекты на экране компьютера можно поворачивать во-

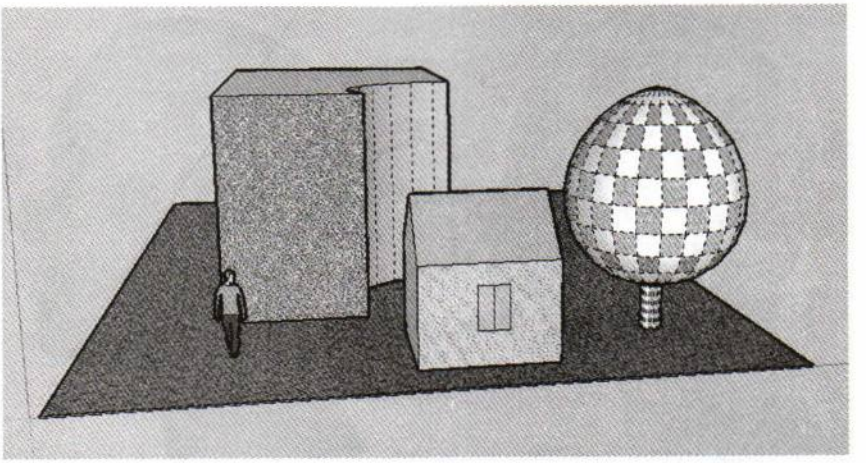


Рис. 74

круг любой оси, рассматривать со всех сторон. Распечатка на принтере изображения такого виртуального объекта не даёт полной информации о его форме (рис. 74).

Появление новых компьютерных объектов и связанной с ними технологии отразилось в создании новой терминологии. Традиционные «плоские» компьютерные графические объекты стали называть двухмерными или 2D-графикой, а объёмные — трёхмерными или 3D-графикой. Латинская буква D является первой буквой английского слова dimension — «измерение».

Двухмерная графика, 2D-графика — плоские компьютерные изображения и технологии их создания.

Трёхмерная графика, 3D-графика — объёмные виртуальные графические объекты и технологии их создания.

Таким образом, изучая растровый графический редактор Paint и векторный графический редактор, встроенный в текстовый редактор Word, мы имели дело с двухмерной графикой.

Методы трёхмерной графики. Для создания трёхмерных графических объектов разработаны достаточно мощные графические редакторы — *3D-редакторы*. Различаются они не только особенностями управления, но и методами моделирования

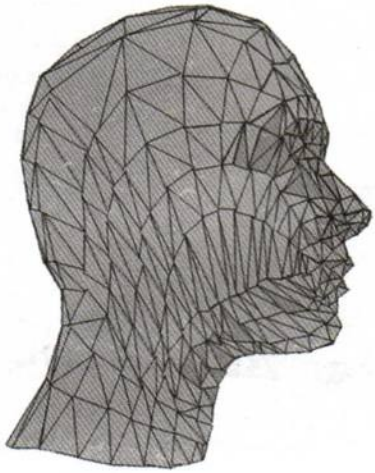


Рис. 75

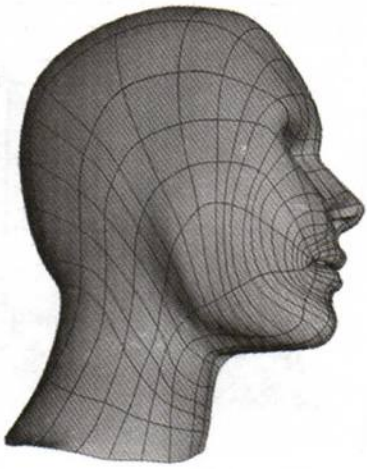


Рис. 76

трёхмерных объектов, среди которых выделим каркасный и твердотельный методы.

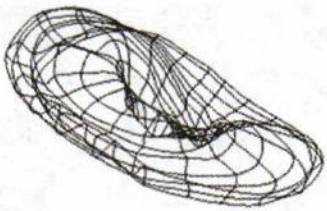
Каркасное моделирование состоит в том, что на экране компьютера поверхность моделируемого объекта изображается в виде каркаса (сетки из линий), в котором каждая ячейка закрывается видимой поверхностью. Если ячейки каркаса являются плоскими многоугольниками, моделирование называют *полигональным* (рис. 75). Если линии сетки являются кривыми, то моделирование называют *поверхностным* (рис. 76).

Обычно для создания каркасной модели в 3D-редакторе строят три плоские проекции воображаемой сетки. Объёмный каркас и поверхности компьютер создаёт автоматически. На рисунке 77 изображены три проекции и объёмная каркасная модель человеческого уха.

Современные 3D-редакторы поддерживают методы быстрого каркасного моделирования и автоматического сглаживания. На одной половине окна в таком 3D-редакторе строят грубую каркасную модель, а на другой половине одновременно возникает поверхностная модель, полученная в результате автоматического сглаживания (рис. 78).

Твердотельное моделирование состоит в том, что на экране можно обрабатывать виртуальную объёмную заготовку: отрезать от неё куски, высверливать в ней отверстия, добавлять недостающий

Каркасная модель человеческого уха



Проекции

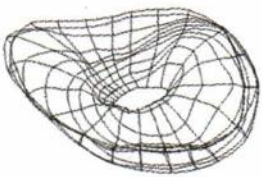
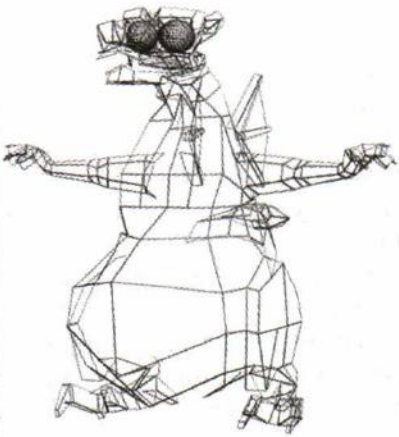


Рис. 77



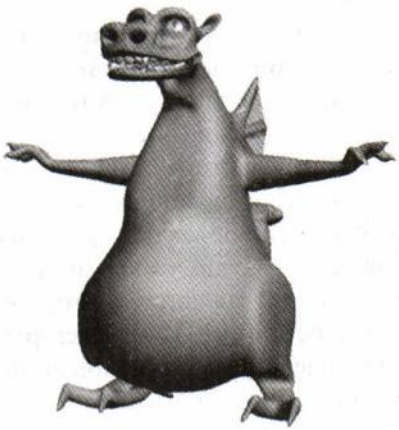


Рис. 78

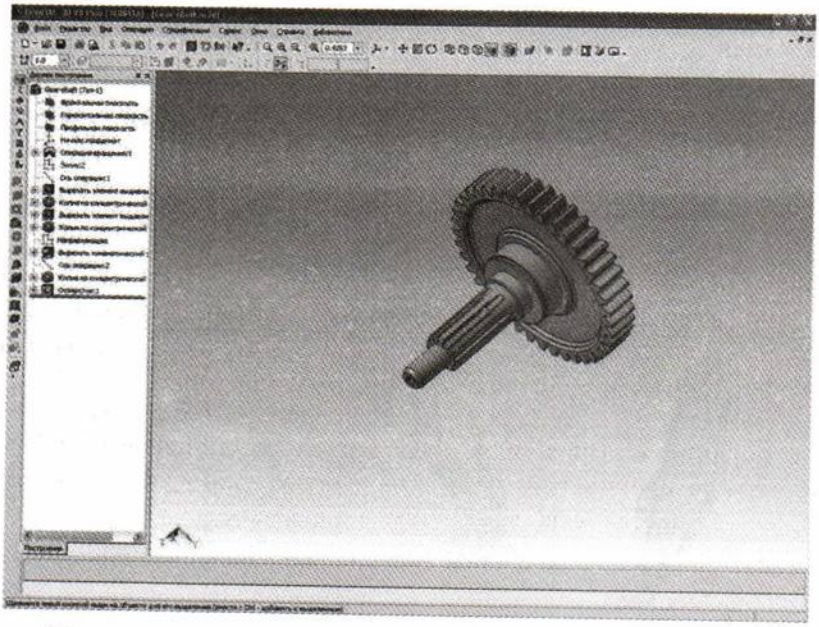


Рис. 79

материал. Работа с твердотельной компьютерной моделью напоминает работу скульптора по дереву, фрезеровщика и токаря одновременно.

Твердотельное моделирование окончательно вытеснило из проектирования макеты, так как позволяет воссоздавать реальные изображения предметов и в автоматическом режиме выдавать их рабочие чертежи (рис. 79).

В 3D-редакторах принята система закрашивания внешних поверхностей с помощью текстур. *Текстуры* — это двухмерные графические объекты, имитирующие внешний вид самых разных покрытий. Существуют многочисленные текстуры обоев, каменной кладки, стальных листов, зелёной травы и т. п.

Современные 3D-редакторы насыщены средствами динамизации моделей, которые можно использовать в компьютерных играх и анимациях. В Интернете можно найти самые разнообразные трёхмерные модели — от динозавров до космических объектов.

Знакомство с 3D-редактором Google SketchUp. Профессиональные 3D-редакторы стоят немалых денег и могут быть

освоены только специалистами. Для учебных целей мы познакомимся с некоторыми возможностями 3D-редактора Google SketchUp, который распространяется через Интернет свободно и имеет несложный интерфейс. Возможности этого редактора позволяют строить достаточно сложные объёмные модели, а в Интернете доступна обширная библиотека готовых моделей, которые созданы пользователями.

Познакомимся с русифицированным редактором SketchUp 7.1. После первого запуска появляется диалоговое окно, в котором нужно выбрать шаблон изображения. Рекомендуется выбрать «Простой шаблон — метры» и щёлкнуть по кнопке **Начать использование SketchUp**. Открывается окно редактора (рис. 80).

В области просмотра редактора изображена трёхмерная система координат с цветными осями: красной, зелёной и синей. Плоскости, проходящие через оси координат, условно разбивают виртуальное пространство на 8 октантов. В одном из верхних октантов для ориентации и восприятия масштаба размещена фигура человека.

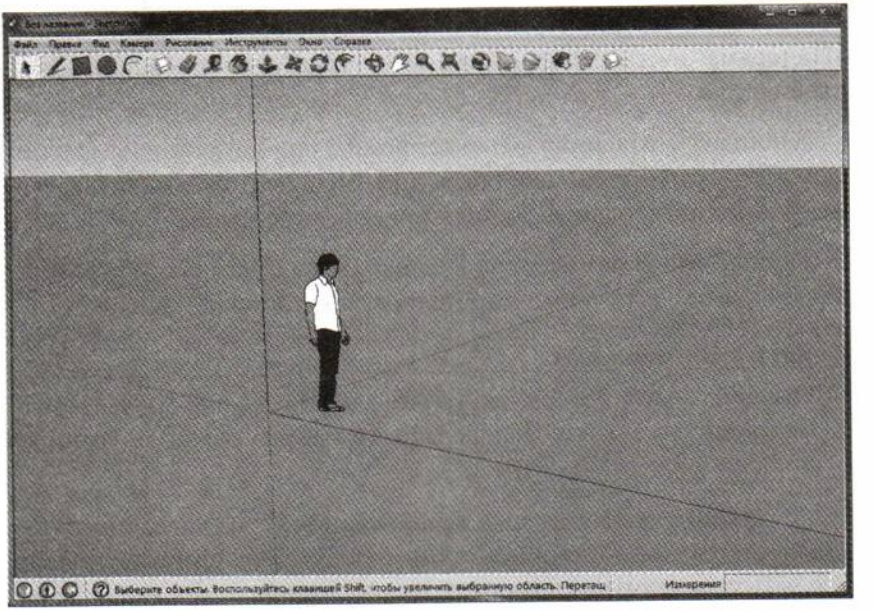


Рис. 80

Управляется редактор с помощью меню и панели инструментов, которая вертикальными отрезками поделена на разделы. В редакторе действует система всплывающих подсказок, которая даёт возможность познакомиться с названиями кнопок.

Перемещение камеры. В данном 3D-редакторе считается, что существует некая виртуальная телекамера, изображение с которой передаётся в окно редактора. Камера может перемещаться в виртуальном пространстве. Соответственно будет меняться и изображение в окне, таким образом обеспечивается возможность построения изображения объёмной модели в разных ракурсах.

Для включения режимов перемещения камеры предназначены кнопки пятого раздела панели.

Кнопка **Орбита** предназначена для включения режима перемещения камеры по произвольной орбите. В этом режиме для перемещения нужно использовать перетаскивание мыши. Этот режим можно включить нажатием и удержанием колёсика мыши.

Кнопка **Панорама** предназначена для включения режима панорамного перемещения камеры. Перемещение связано с перетаскиванием мыши. При включённом режиме «Орбита» режим «Панорама» включается нажатием и удержанием клавиши **Shift** клавиатуры.

Кнопка **Масштаб** включает режим приближения и удаления камеры в направлении оси объектива. Но для этого в любом режиме удобнее пользоваться вращением колёсика мыши.

Кнопка **В размер окна** щелчком мыши мгновенно устанавливает камеру так, что модели полностью заполняют экран.

■ Упражнение 98

Пользуясь кнопками данного раздела панели, переместите изображение на экране. Для выработки навыка нужно потренироваться.

Вывод графических примитивов. С графическими примитивами связаны кнопки второго раздела панели инструментов. Все четыре примитива являются двухмерными.

Инструмент «Линия» рисует отрезки прямых, инструменты «Прямоугольник» и «Окружность» действуют привычным обра-

зом, а последний, четвёртый инструмент «Дуга» этого раздела позволяет выводить дуги кривых. Вывести примитив удобнее всего на горизонтальную плоскость.

■ Упражнение 99

Выведем на экран примитивы прямоугольника и круга.

Начнём с прямоугольника. Прямоугольник перетаскиванием выводится на горизонтальную плоскость. Аналогично выводится круг, только он строится из центра. Построенные плоские фигуры будут основаниями будущих объёмных фигур.

Модификация (преобразование) фигур. За модификацию фигур отвечают инструменты, которые собраны в четвёртом разделе панели инструментов.

Кнопка  **Тяни/Толкай** превращает курсор в инструмент для вытягивания плоских граней. На кнопке такая пиктограмма и изображена.

■ Упражнение 100

Проверим работу кнопки вытягивания.

Выбрав инструмент, указываем концом красной стрелки графического курсора на прямоугольник, который выведен на горизонтальную плоскость (см. упр. 99), и перетаскиваем мышь вверх. За курсором вырастает объёмная фигура параллелепипеда. Аналогично поступаем с кругом. Но и это ещё не всё. Если поместить инструмент у боковой грани параллелепипеда, то она выделяется мелкой сеткой из точек, и её также можно вытянуть.

Кнопка  **Переместить/Копировать** превращает курсор в очень мощный инструмент. Но при этом важно, в каком месте фигуры находится курсор.

■ Упражнение 101

Щёлкнув по кнопке  **Переместить/Копировать**, попытайтесь очень медленно перемещать курсор по изображениям построенных фигур.

На параллелепипеде при некоторых положениях курсора выводятся подсказки, элементы фигуры выделяются, и на них появля-

ется цветная точка. Подсказки обозначают местоположение точки: **На грани** (синяя точка), **На крае** (красная точка), **Конечная точка** (зелёная точка), **Точка середины** (голубая точка).

Щелчок мыши в любом из этих положений включает режим преобразований. Любое движение курсора приводит к изменению фигуры. Чтобы выйти из режима преобразований, надо сделать ещё один щелчок. Заметим, что неудачные преобразования всегда можно отменить сочетанием клавиш **Ctrl + z**.

При перемещении курсора по цилиндру для центра круга добавляется подсказка **Центр** (фиолетовая точка). Интересно, что на боковой поверхности цилиндра подсказки указывают разные точки.

Кнопка **Выбрать** (первая на панели) включает режим выделения. Этот режим позволяет щелчками (перетаскиванием) с удержанием нажатой клавиши **Shift** клавиатуры выделять элементы моделей или модели целиком. Выделение снимается щелчком вне моделей. Выделенное можно объединить с помощью пункта **Создать группу** контекстного меню. Инструментом **Переместить/Копировать** выделенные элементы можно переместить или, при удержании нажатой клавиши **Ctrl** клавиатуры, скопировать. Выделение в этом случае снимается клавишей **Esc**.

Кнопка **Ластик** из третьего раздела панели инструментов превращает курсор в ластик, который удаляет элементы моделей. Как только часть какого-то контура удаляется, исчезает и грань, ограниченная этим контуром.

Заливка граней. Диалоговое окно с материалами (текстурами) для заливки вызывается кнопкой **Заливка** из третьего раздела панели. Диалоговое окно имеет две области просмотра: верхнюю и нижнюю. Между ними в строке управления размещено поле с кнопкой, которая открывает список групп материалов: асфальт и бетон, вода, камень, светопроницаемое, цвета и др.

После выбора имени группы в нижней области появляются образцы материалов. Там и делают выбор. Выбранный материал (текстура) отображается в верхней области просмотра.

Заливку граней и поверхностей объёмных фигур производят, как обычно, щелчком по грани или по поверхности.

В диалоговом окне «Материалы» имеется кнопка **Окраска по образцу**, которая действует как инструмент «Палитра» в

графическом редакторе Paint. После щелчка по этой кнопке курсор принимает вид пипетки (она изображена на пиктограмме). Теперь однократно можно выбрать материал (текстуру) на любой поверхности построенной модели.

Компоненты. Так называют готовые трёхмерные модели. Диалоговое окно «Компоненты» вызывается командой меню **Окно|Компоненты** и устроено аналогично окну «Материалы». Кнопкой слева от поля открывается список групп. Выбирается сначала группа, а затем конкретная модель. Нужно помнить, что большинство компонентов находится на сервере Google и требует для загрузки подключения к Интернету. Модель щелчком мыши устанавливается на место. Чтобы убрать выделение с компонента, нужно щёлкнуть правой клавишей мыши в стороне от модели.

■ Упражнение 102

Постройте модель одноэтажного домика, изображения которого хранятся в файлах на CD-диске .

Построения начинаем с прямоугольника, который будет изображать траву. Далее строим прямоугольник для основания дома и вытягиваем его вверх инструментом **Тяни/Толкай**. Инструментом «Линия» на верхней грани модели дома строим линию, которая будет коньком крыши (рис. 81). Инструментом **Переместить/Копировать**, уцепившись за будущий конёк, подни-

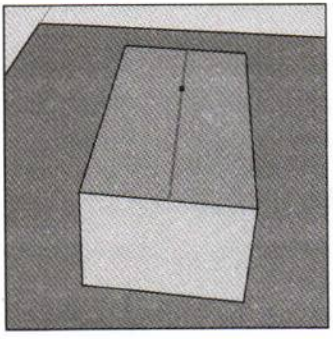


Рис. 81

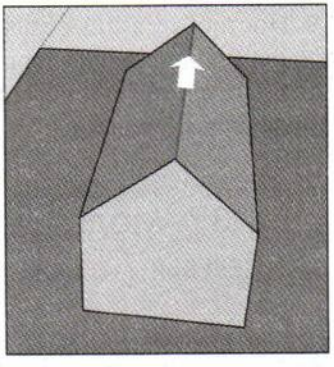


Рис. 82

маем крышу (рис. 82). Инструментом «Линия» отделяем торцевые стены от фронтонов (боковых треугольников). Для трёхмерных построений остались только два прямоугольника для крыльца с вытягиванием вверх.

Теперь на поверхности можно наложить текстуры по своему вкусу. Исходный набор компонентов в редакторе невелик, но двери он содержит. Компоненты-окна загружаются с сервера Google или собираются из прямоугольников, часть которых должна иметь светопроницаемые поверхности.

☐ Вопросы и задания

1. В чём состоит различие между двухмерной и трёхмерной графикой?

2. В чём состоит метод каркасного моделирования трёхмерных объектов?

3. Чем полигональный метод отличается от поверхностного метода моделирования?

4. В чём состоит метод твердотельного моделирования?

5. В чём сущность быстрого каркасного моделирования?

6. В информационных ресурсах Интернета соберите информацию о возможностях наиболее популярных 3D-редакторов.

7. Найдите в Интернете информацию и подготовьте реферат о рельефном моделировании и обратном инжиниринге, о возможностях модификаторов.

8. Найдите в Интернете общедоступные коллекции трёхмерных моделей для редактора Google SketchUp и скачайте их для просмотра.

9. Постройте в 3D-редакторе объёмную модель вашей школы, включая пришкольную территорию.

10*. Постройте в 3D-редакторе объёмную модель интерьера компьютерного класса.

§ 37. Моделирование иерархических систем. Деревья

Иерархическая система — это множество объектов, каждый из которых, кроме одного (главного), подчиняется другому объекту.

Иерархической системой является любое воинское подразделение, возглавляемое командиром, и вооружённые силы страны в целом, возглавляемые главнокомандующим. Иерархическая система в смысле подчинённости — это система управления, основанная на единоначалии, а единственный объект, который в системе никому не подчиняется, — это командир системы.

Если понимать подчинённость в более широком смысле — как отношение детей и родителей, то иерархическую систему составят любой человек и его дети, внуки, правнуки и т. д. Описание такой системы обычно называют родословной.

Подчинённость можно понимать и в смысле отношения части и целого. Тогда в качестве иерархических систем можно представить не только коллективы людей, но и другие множества, например множество всевозможных моделей. В зависимости от формы реализации модели бывают материальные и информационные. Информационные модели включают мысленные, вербальные, документальные и аппаратно-зависимые модели. Другой пример — этот учебник. Он состоит из глав, а главы составлены из параграфов.

Формы информационных моделей. В этом параграфе нас будут интересовать информационные (образно-знаковые) модели иерархических систем. Возьмём в качестве примера родословную князя Изяслава, сына Ярослава Мудрого (XI—XII вв.), составленную Н. М. Карамзиным. Родословная была опубликована в 1844 г. в форме, которая использована на рисунке 83 и которую Н. М. Карамзин называл *родословной таблицей*.

Приведённая запись иерархической системы является её образно-знаковой моделью. В этой модели дочерние записи располагаются на одну строку ниже отцовских и охватываются сверху фигурной скобкой.

В наши дни для представления иерархических систем используют другие формы. Чаще всего иерархические системы представляют в форме схем. Пример такой схемы даёт рисунок 44 с классификацией видов моделей в зависимости от формы их реализации. Иной вид схемы, которая использована для примера родословной на рисунке 83, приведён на рисунке 84.

Изображение иерархической системы в форме разветвлённого дерева так и называют деревом. Заметим, что дерево на рисунке 84

РОД ЯРОСЛАВА МУДРОГО И ИЗЯСЛАВА ЯРОСЛАВИЧА

Изяслав, сын Ярослава Мудрого

Мстислав Святополк–Михаил Евпраксия Ярополк

Ростислав Мстислав Брячислав Изяслав Ярослав Вячеслав
Сбыслава Ярослав Передслава

Юрий Вячеслав

Ярополк Ростислав Иван Святополк Глеб Ярослав Мальфрида Анна

Рис. 83

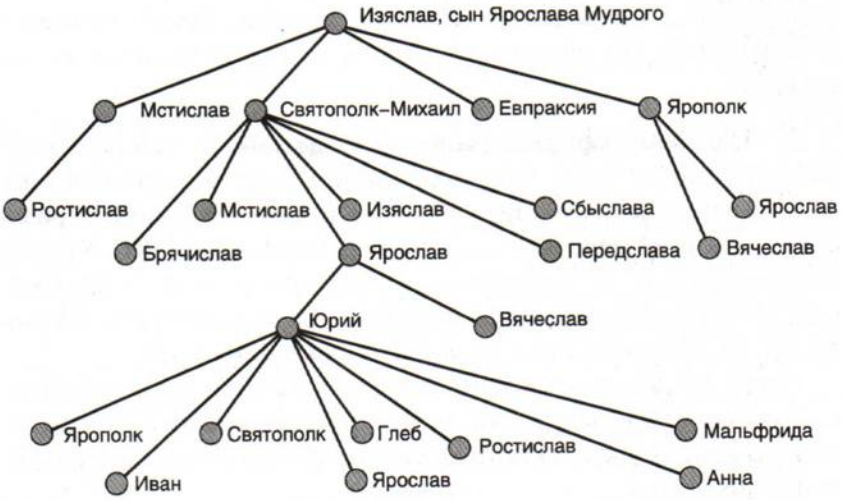


Рис. 84

имеет естественное название: *родовое*, или *генеалогическое*, *дерево*.

Ещё одна форма записи иерархической системы реализована в операционной системе Windows для отображения содержимого дисков. Такую форму изображения называют *деревом папок*.

Терминология. Рассмотренные примеры показывают, что в некоторых задачах информационные объекты (данные) удобно хранить как объекты иерархических систем.

Дерево — это конечное множество информационных контейнеров, которые называются *узлами,* со следующими свойствами:
- некоторые пары узлов связаны отношениями типа «отец—сын»;
- существует один узел, который не имеет «отца» и называется ***корнем***;
- каждый узел, за исключением корня, имеет только одного «отца».

Исторически сложилось, что значительная часть терминологии в области иерархических систем хранения информации связана с родовыми деревьями. Для описания терминологии рассмотрим дерево, представленное на рисунке 85.

Корнем этого дерева является узел A.

Лист — это название узла, не имеющего «сыновей».

На рисунке 85 листьями являются узлы D, F, H, I, J, K.

Братья — это название для узлов, которые являются «сыновьями» одного «отца».

«Братьями» являются узлы B, C; узлы D, E, F; узлы G, H; узлы J, K.

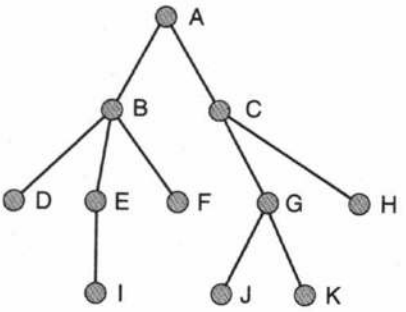


Рис. 85

Степень узла — это количество «сыновей» узла.

Степень узла C равна 2, узла E — 1, листа K — 0.

Степень дерева — это максимальная из степеней его узлов.

В нашем примере максимальная степень дерева равна 3. Такую максимальную степень имеет узел B.

Уровень узла — это количество узлов, включая корень, которые надо пройти на пути от корня до данного узла.

Узел D имеет уровень 2, так как до узла D нужно пройти узлы A и B.

Глубина дерева — это максимальный из уровней его узлов.

Максимальный уровень, равный 3, имеют узлы I, J, K. Следовательно, глубина дерева равна 3.

Выбор компьютерных моделей. Под задачами обработки деревьев мы будем понимать расчёт числовых характеристик узлов и дерева в целом. Вычислить все введённые характеристики для дерева с малым количеством узлов труда не представляет. Достаточно построить графическую схему и подсчитать значения характеристик. Сложности появляются, когда нужно получить характеристики дерева с большим числом узлов, и без компьютера это сделать невозможно. Для решения задач обработки деревьев мы будем составлять программы на языке JavaScript. Расчёты будем проверять, используя дерево на рисунке 85.

До составления программы нужно решить вопрос о том, какая информация о дереве должна храниться в программе. Будем записывать для каждого узла хранимую контейнером информацию (в нашем примере это прописная латинская буква) и ссылку на узел-«отец». Чтобы записывать ссылки, надо присвоить узлам адресные коды (ключи, идентификаторы). Для наших целей узлы достаточно перенумеровать, тогда номер узла будет его адресом (совсем как в Интернете). Заметим, что узлы дерева удобно перенумеровать, используя бумажный эскиз.

Для хранения информации о дереве в программе мы будем использовать два линейных массива: первый — для хранения информации узлов, второй — для адресов узла-«отца». Так как в массивах нумерация элементов начинается с нуля, то нумеровать узлы мы будем, также начиная с нуля, причём адрес 0 присвоим корню. Будем считать, что узел-корень имеет условную ссылку на несуществующего «отца», равную −1.

Ввод исходных данных. Подготовим к работе простейшую систему программирования на языке JavaScript. Открыв файл Help_JS.htm с документом «Справка JavaScript», копированием введём в «Notepad++» HTML-конструкцию для работы с деревьями.

```
<HTML>
<!-- saved from url=(0014)about:internet -->
<Script src="C:\JS\tree.js"></Script>
<title> </title>
<Script>

</Script>
</HTML>
```

Файл tree.js из папки JS позволит по заданным исходным данным строить дерево на экране компьютера.

В начале любой программы для обработки дерева сразу надо ввести исходные данные.

Условимся, что номер узла дерева будет номером соответствующих элементов в массивах.

```
inf = ["A", "B", "C","D", "E", "F", "G", "H", "I", "J", "K"];
ref = [-1, 0, 0, 1, 1, 1, 2, 2, 4, 6, 6];
```

■ Упражнение 103

Подсчитаем количество листьев дерева.

Для решения задачи подсчёта количества листьев нужно вычислить номера узлов, на которые нет ссылок в массиве ref.

Поступим следующим образом.

1. Создадим массив mm с номерами узлов (от 0 до 10).
2. Будем в цикле сравнивать элементы массива ссылок ref с элементами массива номеров mm. Как только элемент массива ref совпадёт с элементом массива mm, элемент из массива mm удаляем. В результате в массиве mm останутся невостребованные номера — это и есть номера узлов-листьев.
3. Подсчитаем длину массива mm и организуем вывод результатов.

Дадим программе имя «Количество листьев» и запишем её.

```
mm = new Array();
for (i=0; i<inf.length; i++) {
   mm[i]=i;
}
for (i=0; i<ref.length; i++) {
   for (j=0; j<mm.length; j++) {
      if (ref[i]==mm[j]) {
         mm.splice(j,1);
         break;                          //Досрочное завершение цикла
      }
   }
}
rez=mm.length;
```

Для вывода на экран графического представления дерева воспользуемся методом OutTree(), который используется как команда. Два аргумента этого метода записываются через запятую. Например, OutTree(inf, ref). Первый аргумент — это имя массива с информацией узлов дерева, второй аргумент — имя массива со ссылками. Если аргументы не образуют дерево, то на экране выводится сообщение.

Вывод результатов счёта организуем с помощью метода alert(). Сделаем это для того, чтобы вывод текста на экран не наложился на графическое представление дерева.

```
OutTree(inf, ref);
alert("Количество листьев = " + rez); //Результат
```

После запуска и отладки сохраняем программу в файле с именем ex93.htm.

Замечание. Обратите внимание на то, что если после вывода дерева на экран установить курсор на знак узла, то на экране появляется всплывающая подсказка с номером этого узла.

■ Упражнение 104

Подсчитаем степень заданного узла.

Дадим программе имя «Степень узла». Степень узла определяется количеством записей его номера в массиве ref. Подсчёты будем вести следующим образом.

Пусть проверяемый узел имеет номер nn. Подсчёт количества его сыновей организуем в переменной rez с помощью счётчика. В цикле организуем сравнение номера nn с номерами в массиве ref. В случае совпадения номеров значение счётчика будем увеличивать командой

```
rez++;
```

Результаты расчётов выведем аналогично предыдущей программе.

```
nn=6;                                    //Номер проверяемого узла
rez=0;                                   //Счётчик
for (i=0; i<ref.length; i++) {
```

```
    if (ref[i]==nn) {
        rez++;
    }
}
OutTree(inf, ref);
alert("Степень узла " + nn + " = " + rez);
```

После запуска и отладки сохраняем программу в файле с именем ex94.htm.

Вопросы и задания

1. Что такое иерархическая система?

2. Какие существуют формы графических моделей иерархических систем?

3. Что такое дерево в теории иерархических систем?

4. Расскажите о числовых характеристиках дерева.

5. Для чего предназначен метод OutTree() и какими должны быть его аргументы?

6. Используя только ввод исходных данных и метод OutTree(), составьте на языке JavaScript программы вывода на экран:

1) родословной, рассмотренной в данном параграфе;

2) родословной одного из ваших предков (деда, прадеда).

7. Для заданного дерева на языке JavaScript создайте программы расчёта:

1) количества «братьев» заданного узла;

2) уровня заданного узла;

3) степени дерева;

4) глубины дерева.

8*. Для заданного дерева на языке JavaScript создайте программы вывода:

1) списка номеров узлов заданного уровня;

2) списка узлов на пути из заданного узла в корень.

9*. На языке JavaScript создайте программы:

1) добавления нового узла-«сына» к любому узлу дерева;

2) удаления из дерева заданного узла и всех его потомков.

Указание. Все такие программы следует строить по схеме: ввод данных исходного дерева; вывод исходного дерева на экран; ввод данных для изменения методом prompt(); расчёты; вывод результатов в виде нового дерева на экран.

10*. Двоичное дерево — это дерево степени 2. В Интернете найдите информацию по терминологии двоичных деревьев и их разновидностей, а также описание задач, которые можно решать с помощью двоичных деревьев.

§ 38. Понятие о графах

Термин «граф» впервые ввёл в 1936 г. венгерский математик Д. Кёниг, но задачи, связанные с использованием графов, исследователи рассматривали и до него. Сегодня усилиями учёных разных стран создана целая теория, которая так и называется — ***теория графов***. Чтобы войти в курс теории графов, нам придётся пройти через большое число новых, непривычных понятий.

Граф — это совокупность множества объектов, которые называют ***вершинами***, и множества пар вершин, которые называют ***рёбрами***.

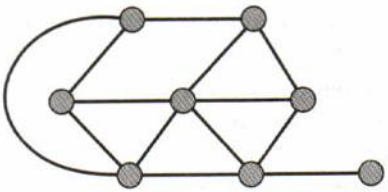


Рис. 86

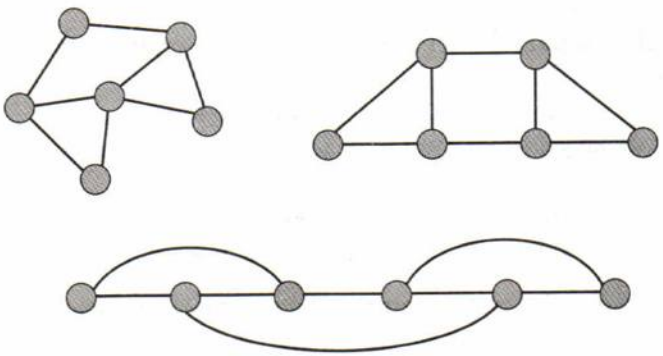


Рис. 87

Теоретическое определение легко проясняется, если граф представить в графическом виде. Каждой вершине на рисунке ставится в соответствие кружок, а каждому ребру — линия, которая соединяет две вершины (рис. 86). Собственно, из-за наглядности такого графического изображения граф и получил своё название.

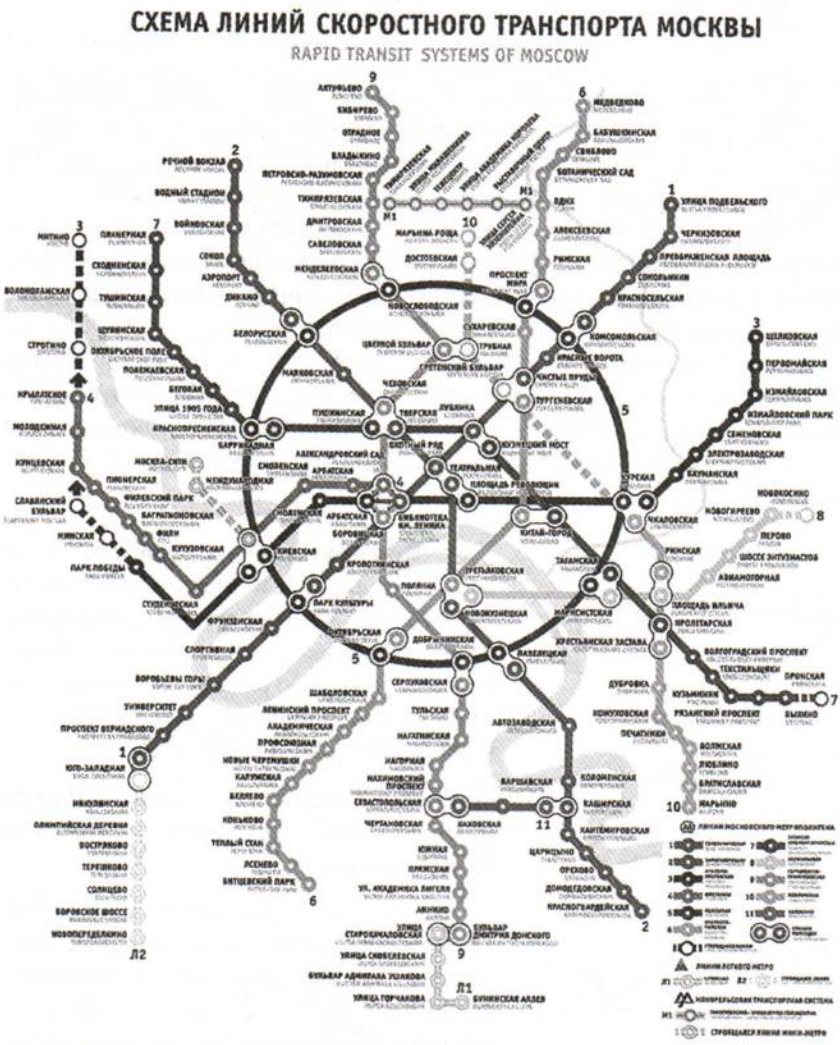


Рис. 88

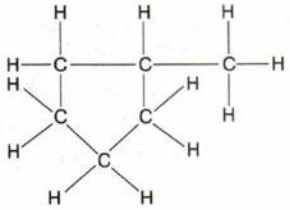


Рис. 89

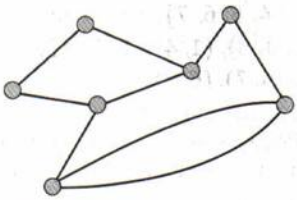


Рис. 90

Заметим, что для одного и того же графа можно построить сколько угодно графических изображений, которые будут различаться только расположением вершин и формой рёбер (рис. 87). Такие графические изображения называют ***изоморфными***.

Графы с успехом используются для моделирования самых различных объектов: схем железных и автомобильных дорог, газопроводов и нефтепроводов и т. д. Наиболее известный пример — это схема линий Московского метрополитена (рис. 88).

Графы применяются в химии (рис. 89), в проектировании электрических и электронных схем (см. рис. 62), во многих других отраслях науки и экономики. Даже методы 3D-моделирования используют объёмную реализацию графа в виде каркасной полигональной модели.

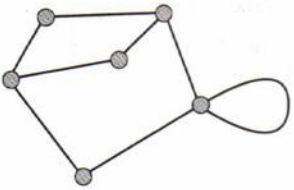


Рис. 91

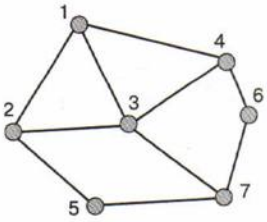


Рис. 92

Разновидности графов. Чаще всего в задачах рассматривают графы, в которых пары вершин соединены одним ребром. Мы также будем рассматривать графы, которые не имеют параллельных рёбер (рис. 90) и рёбер-петель (рис. 91).

Граф называется ***помеченным***, если его вершинам присвоены некоторые метки.

Если граф G — помеченный (рис. 92), то можно в явном виде описать множество GV его вершин и множество GE его рёбер. Например, граф на рисунке 92 помечен с помощью нумерации вершин. Его множество вершин

GV = {1, 2, 3, 4, 5, 6, 7}, множество рёбер GE = {(1, 2), (1, 3), (1, 4), (2, 3), (2, 5), (3, 4), (3, 7), (4, 6), (5, 7), (6, 7)}.

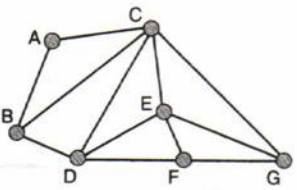


Рис. 93

Порядок графа — это количество его вершин. В примере на рисунке 93 граф имеет порядок 7.

Граф называется ***связным***, если из одной его вершины можно попасть в любую другую, перемещаясь по рёбрам.

Мы до сих пор рассматривали примеры именно связных графов. Пошагово удаляя рёбра из связного графа, можно на определённом шаге нарушить его связность.

Деревом называется связный граф, в котором нельзя, передвигаясь по рёбрам, выйти из произвольной вершины по одному ребру, а вернуться в неё по другому ребру.

На рисунке 94 даны примеры деревьев с пятью рёбрами. Заметим, что количество вершин дерева на единицу больше количества рёбер того же дерева.

Понятие дерева в теории графов шире понятия дерева, введённого в предыдущем параграфе для иерархических систем. Различие в подходах неудивительно, так как понятия введены в разных теориях, а вводили их разные учёные, опираясь только на специфику внешней формы.

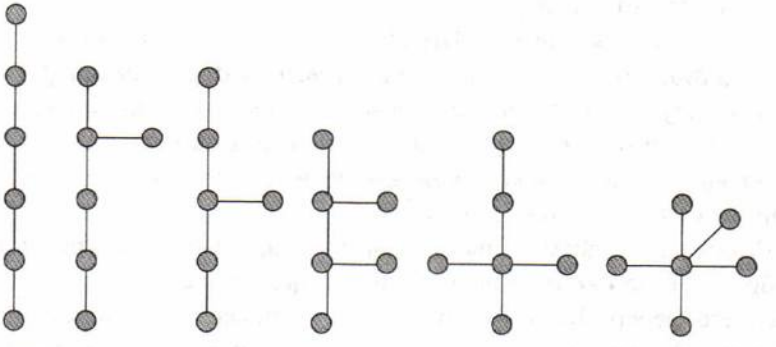


Рис. 94

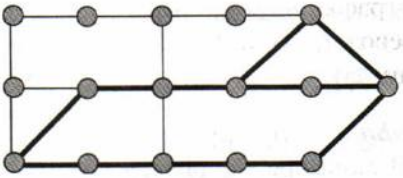


Рис. 95

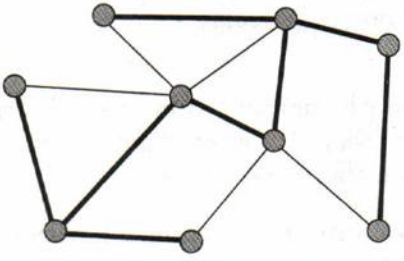


Рис. 96

Чтобы отличить понятие дерева из теории иерархических систем, его часто называют *корневым деревом*.

Задача построения минимального остовного дерева. Для графов существует очень много самых разных задач и алгоритмов их решения. Познакомимся с одной из таких задач.

Граф называется ***подграфом*** исходного графа, если все вершины и рёбра определяемого графа являются вершинами и рёбрами исходного. На рисунке 95 рёбра одного подграфа выделены увеличением толщины рёбер.

Если дерево является подграфом данного графа и включает все его вершины, то оно называется ***остовным деревом графа***.

На рисунке 96 остовное дерево выделено. Остовных деревьев для одного и того же графа может быть несколько.

Граф называется ***взвешенным***, если его вершинам или рёбрам приписаны какие-то числа (веса).

Мы будем рассматривать графы, в которых веса приписаны рёбрам. Тогда ***весом*** взвешенного графа называется сумма весов всех его рёбер. Конечно, вес ребра понимается в обобщённом смысле. Это может быть не только вес, но и длина, затраты, пропускная способность и т. д.

Взвешенные графы служат хорошими моделями дорожных схем, систем перевозок и т. п. В качестве вершин таких графов выступают населённые пункты, в качестве рёбер — дороги. Весом ребра в таком случае является длина дороги. Но весом может служить не только протяжённость дороги, но и затраты на её строительство, которые рассчитываются при проектировании.

Рассмотрим конкретную задачу. Пусть есть сельский район, все населённые пункты которого должны быть телефонизированы. Кабель удобнее укладывать в траншеи вдоль дорог, чтобы иметь возможность ремонтировать его при необходимости. Имеется схема дорог района с указанием их протяжённости (рис. 97). Требуется спроектировать такую схему кабельной сети, чтобы суммарная длина уложенного кабеля была минимальной.

Понятно, что схема кабельной сети будет являться деревом. Это можно доказать. Действительно, если минимальная сеть не является деревом, то существует населённый пункт, из которого можно

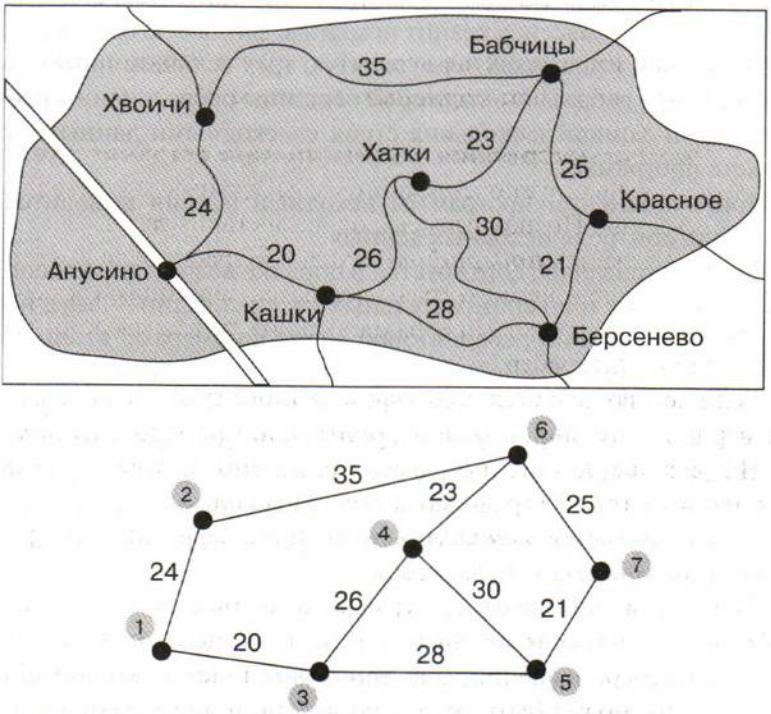


Рис. 97

пройтись вдоль кабеля, сделав условный круг. А это значит, что вдоль одной из дорог этого круга кабель можно не укладывать — и без этого одного звена кабель дойдёт до всех населённых пунктов на круге, т. е. предыдущая сеть не была минимальной.

Остовное дерево графа, которое во взвешенном графе имеет минимальный вес, называется *минимальным остовным деревом*.

Рассмотренная задача телефонизации сельского района в теории графов называется задачей построения минимального остовного дерева.

Алгоритм Прима. Есть несколько алгоритмов решения представленной задачи. При составлении программы на языке JavaScript мы использовали модификацию алгоритма Прима. Программа записана на прилагаемом CD-диске  в файле GraphP.htm.

Вершины графа следует перенумеровать. Граф задаётся списком рёбер. Так как для каждого ребра нужно записать номера двух вершин и его длину, то в программе задаётся массив строк graph. Каждая строка этого массива хранит исходные данные об одном ребре в форме записи в кавычках через запятую трёх величин: номера первой вершины ребра, номера второй вершины ребра и длины ребра.

Образец записи нескольких строк с исходными данными есть в тексте программы.

При исполнении программы исходный массив выводится на экран для контроля исходных данных.

Далее массив graph сортируется в порядке возрастания весов рёбер и создаются три вспомогательных массива. Один — beg с номерами первых вершин рёбер, второй — end с номерами вторых вершин рёбер, третий — массив строк rez для записи результата. Массивы beg и end описывают рабочий вариант исходного графа.

Основной алгоритм состоит в следующем. В массиве graph ищется ребро, которое имеет наименьший вес. Одна из двух вершин этого ребра выбирается начальной. Устанавливается номер второй вершины этого ребра, данные по этому ребру из исходного массива graph удаляются и записываются в массив результата rez.

Далее в рабочем варианте графа две вершины ребра с наименьшим весом условно объединяются в одну начальную, и число вершин в графе уменьшается на единицу. Если при этом появляются рёбра-петли, то они удаляются. Соответствующие строки удаляются и из массива graph.

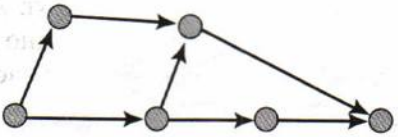


Рис. 98

Теперь в массиве graph вновь ищется ребро, которое выходит из начальной вершины и имеет наименьший вес, и вся процедура, за исключением выбора начальной вершины, повторяется.

Основной алгоритм работает в цикле. Сигналом к окончанию исполнения цикла служит равенство нулю длины массива graph. Это означает, что все строки из массива graph удалены. Тогда алгоритм выводит на экран массив результата rez, вес остовного дерева и прекращает работу.

Ориентированные графы. В заключение отметим, что мы познакомились с графами, которые называются ***неориентированными***, так как направление на рёбрах не задавалось. Если на каждом ребре графа задано направление, то граф называется ***ориентированным*** или ***орграфом***. В орграфах направленные рёбра называются *дугами*, которые графически изображаются стрелками (рис. 98).

Орграфы используются для моделирования транспортных потоков, технологических процессов и т. п.

Вопросы и задания

1. Что такое граф?

2. Приведите примеры использования графов.

3. Что такое петля в графе?

4. Какой граф называется помеченным?

5. Что такое порядок графа?

6. Какой граф называется связным?

7. Что такое дерево в теории графов?

8. Почему дерево в теории иерархических систем стали называть корневым?

9. Что такое подграф заданного графа?

10. Что такое остовное дерево?

11. Какой граф называется взвешенным?

12. Что такое минимальное остовное дерево взвешенного графа?

13. Найдите изоморфные графы на рисунке 99.

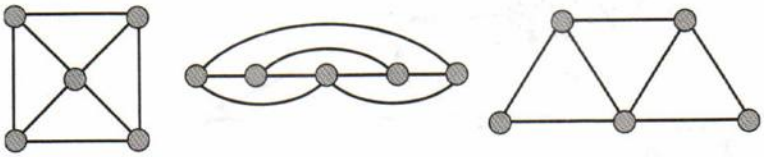


Рис. 99

14. С помощью векторного редактора, встроенного в текстовый редактор Word, изобразите на экране компьютера графы, представленные множествами:

1) GV = {1, 2, 3, 4, 5, 6, 7};
   GE = {(3,4), (7,4), (1,5), (2,3), (6,5), (1,4), (2,7), (6,7), (2,5), (1,3)};
2) GV = {A, B, C, D, E, F};
   GE = {AF, DC, BE, CF, AB, DF, AD, BE, EC}.

15. С помощью векторного редактора, встроенного в текстовый редактор Word, изобразите психологическую схему взаимодействия личности с окружающими людьми в виде графа (рис. 100).

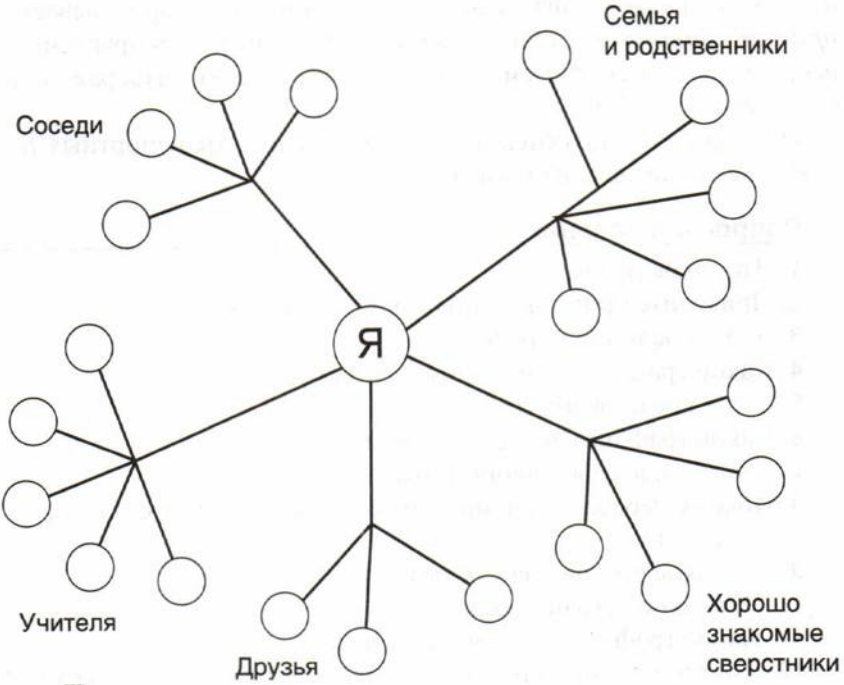


Рис. 100

Количество вершин в каждой группе должно соответствовать вашему личному окружению.

16. С помощью готовой программы алгоритма Прима найдите минимальное остовное дерево для графов, представленных на рисунках 101 и 102.

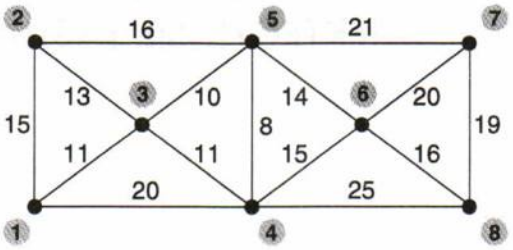


Рис. 101

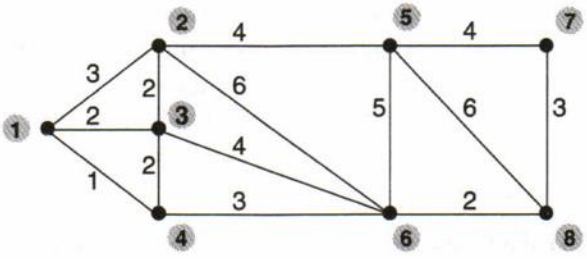


Рис. 102

ГЛАВА 5
ТАБЛИЧНЫЕ МОДЕЛИ И ЭЛЕКТРОННЫЕ ТАБЛИЦЫ

§ 39. Табличные модели и деловая графика

Образно-знаковые информационные модели в предыдущей главе естественным образом были поделены на текстовые (знаковые) и графические (образные) модели. Среди различных документальных графических моделей были выделены чертёжно-графические.

Ещё один класс графических моделей составляют табличные модели.

Табличная модель — это информационная модель в форме прямоугольной таблицы, ячейки которой заполнены числами, текстами или графическими объектами.

Представление в табличной форме характерно для сложных объектов, которые состоят из нескольких объектов-частей. Табличное представление увеличивает наглядность моделей в сравнении с текстовыми описаниями.

В качестве примера таблицей 10 представлена табличная модель Солнечной системы. Представленная модель даёт достаточно полную информацию о параметрах космических объектов Солнечной системы.

Существуют наборы данных, которые представляются только в табличном виде.

Таблица 10

Объект системы	Диаметр, км	Плотность, г/см³	Максимальная температура поверхности, °С	Сутки, земные сутки	Орбитальный радиус, 1 а. е. ≈ 150 млн км	Орбитальный период, лет	Число спутников
Солнце	1 392 000	1,40	5507	25,38	—	—	—
Меркурий	4870	5,43	480	58,60	0,38	0,241	—
Венера	12 102	5,25	480	243	0,72	0,615	—
Земля	12 756	5,52	70	1,00	1,00	1,00	1
Марс	6794	3,94	0	1,03	1,52	1,88	2
Юпитер	143 760	1,31	–160	0,414	5,20	11,86	63
Сатурн	120 420	0,71	0	0,426	9,54	29,46	49
Уран	51 000	1,27	–220	0,718	19,22	84,01	27
Нептун	49 520	1,64	–231	0,671	30,06	164,79	13
Плутон	2324	2,00	–233	6,50	39,50	248,50	3

Самые известные среди них: таблицы умножения (математика), периодическая система элементов Д. И. Менделеева (химия), кодировочные таблицы (информатика), товарно-ценовые предложения или прайс-листы (торговля), таблицы перевода объёмных мер продуктов в граммы (бытовое назначение) и др.

Таблицы составляют основу документов, сопровождающих финансово-хозяйственную деятельность предприятий и организаций. Среди таких документов — всевозможные формы бухгалтерской, статистической и налоговой отчётности, первичные бухгалтерские документы, сметы, калькуляции и т. п. В качестве примера в таблице 11 представлен раздел бухгалтерского баланса.

Таблица 11

Актив	Код показателя	На начало года	На конец года
1. Внеоборотные активы			
Нематериальные активы	110	24 424	23 325
Основные средства	120	52 510 057	51 916 071
Незавершённое строительство	130	4 889 658	2 657 048
Доходные вложения в материальные ценности	135	—	—
Долгосрочные финансовые вложения	140	17 467 177	25 799 389
Отложенные налоговые активы	145	—	—
Прочие внеоборотные активы	150	223 713	410 484
Итого по разделу 1	190	75 115 029	80 806 227

Традиционная форма записи арифметических действий также является табличной. Действия с многозначными числами мы привыкли записывать в столбик.

Например:

$$\begin{array}{r} 234{,}58 \\ +\ 25{,}4568 \\ \hline 260{,}0368 \end{array}$$

Таблица 12

2	3	4,	5	8		
+	2	5,	4	5	6	8
2	6	0,	0	3	6	8

В приведённой записи есть горизонтальные ряды цифр — это сами числа, есть вертикальные столбцы цифр — это соответствующие разряды чисел. Получается, что метод сложения столбиком — это метод сложения с помощью таблицы (табл. 12).

Табличную форму имеют записи операций вычитания, умножения и деления, операций перевода чисел из одной системы счисления в другую.

Среди способов задания функций в математике известен табличный способ. До массового распространения калькуляторов и компьютеров табличный способ задания функций был основным. Таблицы значений функций широко использовались при вычислениях по разнообразным формулам. Четырёхзначные таблицы В. М. Брадиса, пожалуй, самые известные среди таких таблиц. А в пособиях по статистике и сегодня в приложениях приводятся таблицы значений специальных функций.

Практическая деятельность человека, начиная с определённого уровня её развития, неразрывно связана с использованием табличных числовых данных. В связи с бурным развитием экономики в странах Европы эта проблема особенно остро была осознана к концу XVIII в. Банки и ссудные конторы нуждались в таблицах процентов, страховые компании — в таблицах смертности, для мореплавания были нужны астрономические и навигационные таблицы, для проектирования фабричных машин требовались арифметические, тригонометрические и логарифмические таблицы.

Ручная работа по созданию таблиц была очень трудоёмкой. При создании многотомных таблиц ручные вычисления занимали годы и сопровождались невообразимым множеством ошибок. Например, первое издание «Морского календаря» (свода астрономических, навигационных и логарифмических таблиц), вышедшее в 1776 г. в Англии, хотя и готовилось с особой тщательностью, содержало сотни ошибок. Они были результатом неточностей, просчётов и описок.

Деловая графика — это раздел информатики, в котором изучаются методы представления табличных данных в наглядном графическом виде.

С диаграммами, которые назывались схематическими (организационными), мы встречались в 8 классе в задачах обработки текстовой информации с помощью текстового редактора Word. Схематические (организационные) диаграммы по нашей классификации графических моделей являются схемами.

Восприятие большого количества чисел у любого человека вызывает трудности. Поэтому табличные числовые данные для удоб-

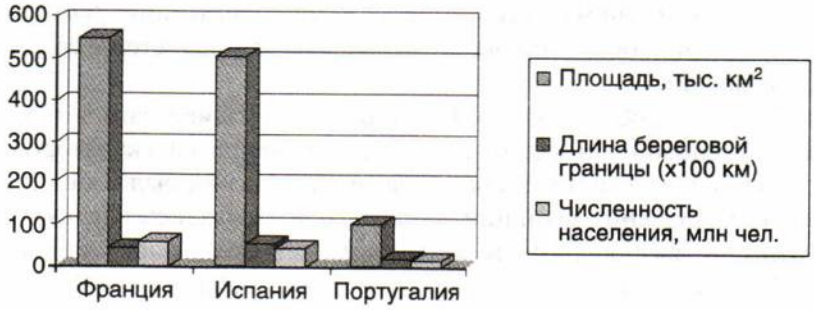


Рис. 103

ства стараются представить в графическом виде с помощью *диаграмм-иллюстраций*.

Например, числовые данные таблицы 13 по трём странам Европы можно представить в виде диаграммы (рис. 103). Очевидно, что сравнивать данные гораздо удобнее в графическом виде.

Таблица 13

Страна	Площадь, км²	Длина береговой границы, км	Численность населения, тыс. чел.
Франция	547 300	3427	57 109
Испания	504 750	4964	39 404
Португалия	92 080	1793	10 562

В диаграммах-иллюстрациях каждому числу из таблицы ставится в соответствие ***маркер данных*** в виде точки или геометрической фигуры.

Для описания числовых данных используют диаграммы-иллюстрации разных типов. К основным типам относятся: столбчатые диаграммы (гистограммы), круговые диаграммы, точечные диаграммы, графики.

Столбчатые диаграммы (*гистограммы*) используют для наглядного сравнения разных показателей. Именно этот вид диаграмм использован на рисунке 103. Числам на нём поставлены в соответствие высоты параллелепипедов.

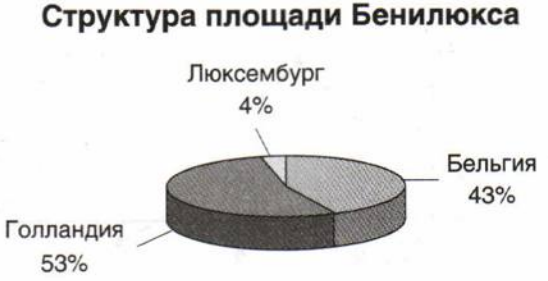


Рис. 104

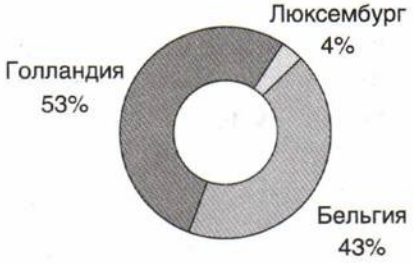


Рис. 105

Круговые диаграммы используют для отображения вклада отдельного показателя в общее значение (рис. 104). Числам здесь поставлены в соответствие выделенные сектора. Частным случаем круговых диаграмм являются *кольцевые диаграммы* (рис. 105).

Точечные диаграммы обычно используются для сравнения значений двух показателей. Они очень удобны для построения графиков функций (рис. 106). Точечной диаграмма называется потому, что один из её возможных вариантов — это действительно только точки на координатной плоскости (рис. 107). С помощью диаграмм типа «график» отображают также развитие той или иной характеристики во времени. Числам на графике поставлены в соответствие точки, которые соединены между собой.

Для работы с табличными моделями и данными в табличной форме разработаны специальные программы — *электронные таблицы*.

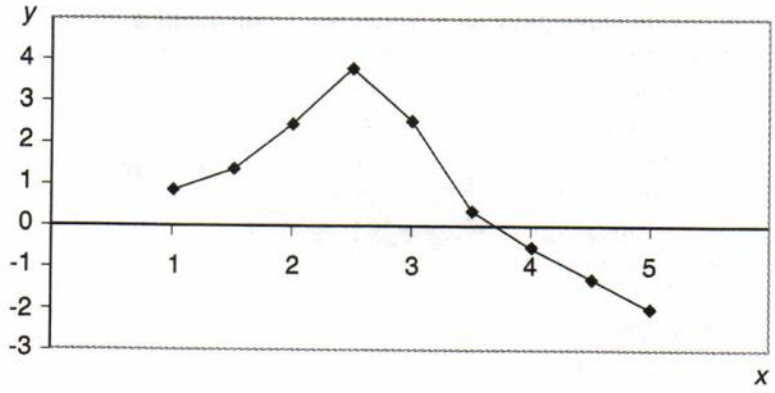


Рис. 106

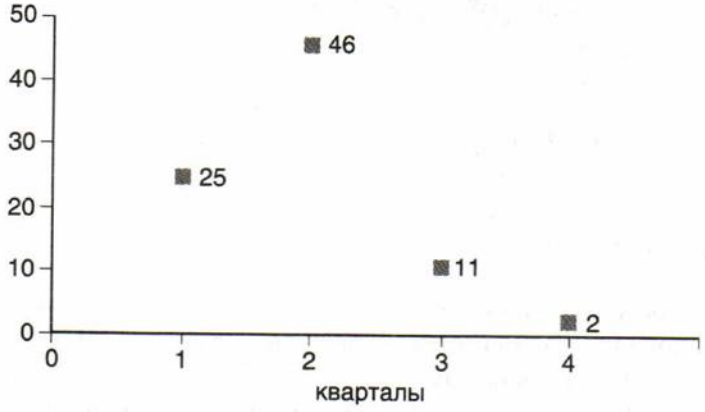


Рис. 107

Электронные таблицы (табличный процессор, редактор электронных таблиц) — компьютерная программа, поддерживающая представление данных в виде таблиц, состоящих из строк и столбцов, на пересечении которых располагаются ячейки таблицы.

Программа представляет часть памяти компьютера в виде прямоугольной таблицы. Таблицу в памяти компьютера называют *электронной таблицей*. Из-за сложившейся терминологии

иногда возникает путаница, так как электронными таблицами называют и таблицы в памяти компьютера, и программы для работы с ними. Именно поэтому мы будем называть программы работы с таблицами *редакторами электронных таблиц*.

Редакторы электронных таблиц позволяют ускорить и облегчить подготовку документов, которые требуют расчётов при их составлении. С помощью электронных таблиц можно решать вычислительные задачи. Редакторы электронных таблиц помогают представить табличные числовые данные в удобном графическом виде с помощью диаграмм-иллюстраций.

Первый редактор электронных таблиц VisiCalc (от англ. visible calculator — видимый калькулятор) для персональных компьютеров появился в 1979 г. С той поры зарубежными фирмами разработано несколько подобных программ: SuperCalc, Multiplan, Lotus 1-2-3, QUATTRO PRO, Excel. Есть разработки подобных программ и на русском языке: «Абак», «Варитаб» и др. Мы будем знакомиться с возможностями редактора электронных таблиц Excel 2010 фирмы Microsoft.

☐ Вопросы и задания

1. Какая модель называется табличной?

2. Перечислите наиболее известные таблицы с данными, которые используются при изучении школьных предметов.

3. Какие документы, построенные на основе таблиц, используются в финансово-хозяйственной деятельности?

4. В чём состоит табличный способ задания функции?

5. Что такое диаграмма-иллюстрация?

6. Назовите типы диаграмм-иллюстраций.

7. Что такое электронная таблица?

8. Что такое редактор электронных таблиц?

9. Когда появился первый редактор электронных таблиц?

10. Пользуясь информационными ресурсами Интернета, создайте реферат на одну из следующих тем:

- «Древнеримские Законы XII таблиц»;
- «Биография В. М. Брадиса — создателя легендарных четырёхзначных таблиц»;
- «История создания компьютерных кодировочных таблиц для кириллицы»;
- «История создания электронных таблиц».

§ 40. Знакомство с редактором электронных таблиц Excel

Запуск и общий вид экрана. Редактор электронных таблиц Excel запускается с помощью ярлыка на рабочем столе операционной системы или из каскадного меню кнопки **Пуск**. В ответ на команду запуска открывается окно программы (рис. 108).

Область просмотра служит для отображения электронной таблицы, которая имеет столбцы и строки. Строки нумеруются в столбце заголовков (левый, серый), а имена столбцов отображаются в строке заголовков (верхняя, серая). Обычно в программе включён режим отображения сетки из разделительных линий.

Вид меню и ленты с вкладками нам уже хорошо знаком. Ниже вкладок размещена строка формул. На вкладке «Вид» в группе «Показать» есть инструменты, которые отвечают за отображение строки формул, сетки и заголовков (столбцов и строк).

Excel имеет несколько режимов просмотра, которые связаны с кнопками на вкладке «Вид» в разделе «Режимы просмотра книги». Чаще всего используются режимы «Обычный», «Разметка страницы» и «Страничный режим». Кнопки этих режимов размещены также в правой части строки состояния перед бегунком изменения масштаба отображения. Изменять масштаб можно и с помощью инструментов группы «Масштаб» на вкладке «Вид». Мы будем

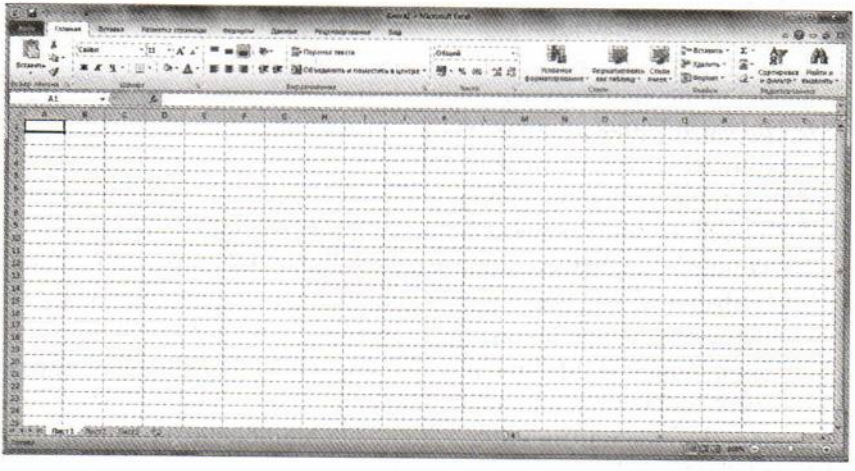


Рис. 108

пользоваться режимом просмотра «Обычный» и масштабом отображения «100%».

Столбцы, строки и ячейки. Познакомимся подробнее с объектами, которые составляют электронную таблицу.

Ячейка (клетка) электронной таблицы — это область электронной таблицы на пересечении столбца и строки.

Содержимое ячейки — это информация, которая записана в ячейке. В любой ячейке электронной таблицы может быть записана информация одного из трёх типов: *числа*, *тексты*, *формулы*. Ячейка электронной таблицы может быть и пустой.

Столбцы, строки и ячейки электронной таблицы имеют имена (заголовки). Столбцы электронной таблицы обозначаются большими буквами латинского алфавита слева направо, а строки нумеруются сверху вниз. Имена столбцов видны в строке заголовков, а имена строк — в столбце заголовков (см. рис. 108). Столбцы и строки так и называют: столбец B, строка 10. Имя ячейки составляется из имени столбца и номера строки. Например, C2, F4.

Excel 2010 электронные таблицы содержат 1 048 576 строк и 15 682 столбца. Так как букв латинского алфавита для имён столбцов не хватает, после столбца с именем Z начинают использовать двойные имена: AA, AB, AC и т. д. до AZ. Потом используют имена BA, BB, BC и т. д. до ZZ. Затем идут тройные имена AAA, AAB и т. д., и так до имени XFD. Поэтому имена ячеек могут быть, например, и такими: DF231, BCX990376.

Табличный курсор — это указатель выделенной ячейки в виде рамки.

В начале сеанса работы табличный курсор расположен на ячейке A1 таблицы (см. рис. 108). Табличный курсор является *клавиатурным* и передвигается по ячейкам электронной таблицы с помощью клавиш-стрелок клавиатуры. Щелчком мыши табличный курсор также можно переставить на любую ячейку.

Электронная таблица обычно больше, чем область просмотра, и в области отображается только часть таблицы. Чтобы отобразить другие ячейки, область передвигают по таблице вправо или вниз. Это делают так же, как двигают область просмотра по тексту в

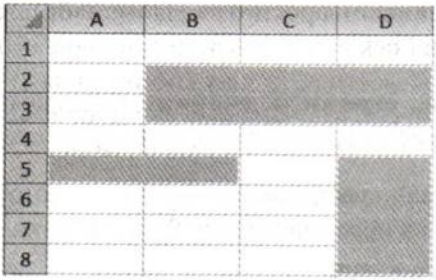


Рис. 109

текстовом редакторе: табличный курсор клавишами-стрелками клавиатуры передвигают до края области и продолжают двигать его дальше. Курсор тянет за собой область просмотра.

Перемещать область просмотра можно и мышью при помощи линеек прокрутки, которые размещены вдоль правого и нижнего края области просмотра.

Блоком электронной таблицы называют совокупность ячеек, которые заполняют некоторый прямоугольник.

Блок называют также *диапазоном*. На рисунке 109 в качестве примеров выделены три различных блока. Блоки обозначают составным именем, которое образуется по следующему правилу:

<имя левой верхней ячейки блока> : <имя нижней правой ячейки блока>

Например, на рисунке 109 блоки получат такие имена:

B2:D3, A5:B5, D5:D8.

Блок электронной таблицы можно выделить. Это делается перетаскиванием мыши с указателем в виде контурного белого креста. Данная операция чем-то похожа на рисование прямоугольника в графическом редакторе Paint. Выделенный блок отличается особым цветом фона и заключается в рамку. Выделение снимается щелчком мыши по любому месту электронной таблицы.

Ввод чисел. Числа в ячейках электронной таблицы записываются по обычным правилам с десятичной запятой, например, 256; –35; 2,45; –0,3. Для ввода чисел на клавиатуре удобно исполь-

зовать группу клавиш цифрового ввода (не забудьте предварительно клавишей **Num Lock** включить режим цифрового ввода).

Чтобы в ячейку ввести число, нужно установить на ячейке текстовый курсор и воспользоваться клавиатурой. В этом учебнике ввод в ячейку A12 числа 1,38 мы будем записывать в следующем виде:

<A12>1,38

■ Упражнение 105

Введём в электронную таблицу следующие числа:

<A2>1,2	<B2>0,7
<A3>2,75	<B3>−2,6
<A4>−3	<B4>17
<A5>157	<B5>−135
<A6>−12	<B6>32

Щелчком мыши устанавливаем табличный курсор на ячейку A2 и начинаем ввод числа с клавиатуры. После ввода первой цифры вводимая информация отображается в ячейке и в строке формул. Если в редакторе строка формул не отображается, то её нужно вывести. Для этого на вкладке «Вид» в группе «Показать» следует пометить флажок возле пункта **Строка формул**.

Завершаем ввод числа 1,2 и нажимаем клавишу **Enter**. Число 1,2 отображается в ячейке A2 электронной таблицы. Сразу заметим, что вместо клавиши **Enter** удобнее нажимать клавишу-стрелку клавиатуры (**Вниз** или **Вправо**), чтобы сразу перевести табличный курсор на следующую ячейку для ввода данных в неё.

Введём в таблицу все заданные числа. Числа в ячейках отображаются с выравниванием вправо (рис. 110).

Ошибки при вводе легко исправить. Последние введённые символы, пока не нажата клавиша **Enter**, удаляются клавишей **Back Space**. Если нажать клавишу **Esc** клавиатуры, то ввод числа вообще прекращается, и число нужно вводить снова. Если же число с ошибкой уже введено, то в ту же ячейку можно ввести новое число. Оно после ввода заменит старое.

Бывает так, что число введено не в ту ячейку и её надо очистить. Тогда на ячейку ставится табличный курсор и нажимается клавиша **Delete (Del)** клавиатуры. Для очистки блока его сначала выделяют, а уже потом нажимают клавишу **Delete (Del)** клавиатуры.

■ Упражнение 106

Очистим блок A6:B6 нашей электронной таблицы и введём числа в обратном порядке:

<A6>32 <B6>–12

Ввод текстов. Текстом в электронных таблицах называют любую строку из символов, которая не является числом. Тексты в электронных таблицах нужны для создания заголовков, надписей, названий граф и т. п.

Тексты вводятся также очень просто. На ячейку устанавливается табличный курсор и с клавиатуры вводится текст. Главное — не забыть включить нужный язык ввода. Ввод завершается нажатием клавиши **Enter** или клавиши-стрелки клавиатуры. Редактирование при вводе делается теми же приёмами, что и при вводе чисел.

■ Упражнение 107

Введём заголовок создаваемой таблицы «Расчётная таблица» в ячейку A1 (см. рис. 110).

Текст в ячейках электронной таблицы обычно отображается с выравниванием влево.

Обратим внимание на то, что содержимое ячейки A1 (17 символов) отображено полностью, хотя ячейка на экране такой ширины

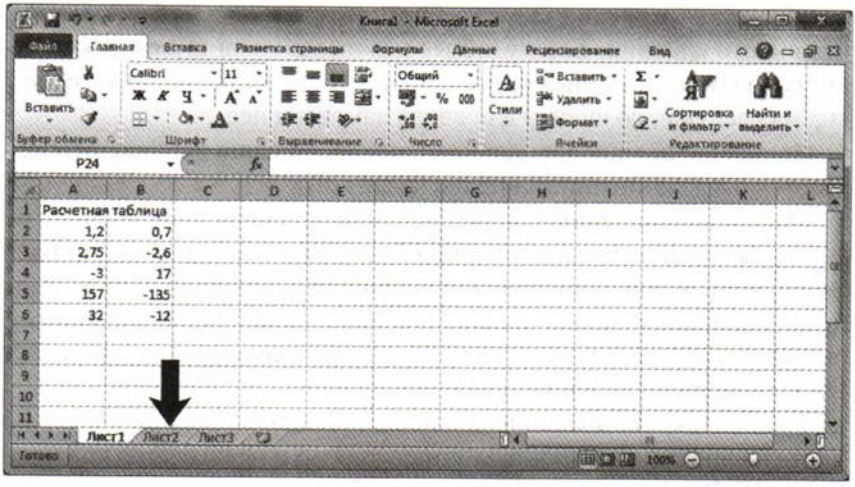


Рис. 110

не имеет. Текст вышел за границу ячейки A1 вправо. Но не следует думать, что излишки текста записаны в ячейку B1.

Чтобы увидеть содержимое ячейки, нужно поместить на неё табличный курсор: содержимое ячейки автоматически отобразится в строке формул. Именно таким образом можно убедиться, что ячейка B1 действительно пуста, а в ячейке A1 находится весь заголовок.

Заметим, что если в ячейку B1 что-то записать, то текст за пределы ячейки A1 уже не выйдет и будет срезан.

■ Упражнение 108

Запишем в ячейку B1 число 4. В результате ввода текст в ячейке A1 обрезается по правому краю ячейки. Очистим ячейку B1 и вернём на экран полный заголовок таблицы.

Простейшие расчёты. Наличие числовых данных в нашей электронной таблице даёт возможность провести некоторые вычисления.

■ Упражнение 109

Найдём сумму чисел первого столбца и среднее арифметическое чисел второго.

Устанавливаем табличный курсор под первым столбцом (в ячейку A7). На вкладке «Главная» в группе «Редактирование» находим кнопку Σ **Автосумма** и щёлкаем по знаку Σ. В ячейке A7 появляется запись =СУММ(A2:A6), и блок A2:A6 автоматически обрамляется тонкой рамкой. Это готовится операция автоматического суммирования. Нажатие клавиши **Enter** завершает суммирование, и результат появляется в ячейке A7. Греческая буква Σ («сигма») в математике используется для обозначения суммы чисел, поэтому на кнопке **Автосумма** она и изображена.

Заметим, что простота операции суммирования даёт возможность использовать редактор Excel для подсчётов даже в бытовых целях. Это надёжнее калькулятора, так как все слагаемые и их сумма находятся на экране и ошибки легко исключить.

Среднее арифметическое — это сумма чисел, делённая на количество этих чисел. Устанавливаем табличный курсор под вторым столбцом (в ячейку B7). Щёлкаем по правой секции кнопки Σ

Автосумма (с треугольником). Открывается новое меню, в котором щёлкаем по пункту **Среднее**. В ячейке B7 появляется запись =СРЗНАЧ(B2:B6), и блок B2:B6 автоматически обрамляется. Нажимаем клавишу **Enter** и завершаем операцию. В ячейке B7 отображается результат.

Пользуясь дополнительным меню кнопки **Автосумма**, можно найти максимальное или минимальное число в столбце, подсчитать их количество (пункт **Число**).

В результате выполнения упражнения 109 в ячейках A7 и B7 записаны формулы. Если просмотреть содержимое ячеек A7 и B7, то можно заметить, что запись формулы начинается со знака равенства «=».

Тут необходимо сказать о главном свойстве электронных таблиц. Если в ячейке таблицы записана формула, то на экране отображается результат вычислений по этой формуле. Вспомним, что при вводе формул в программу «Калькулятор» на индикаторе также отображались только результаты вычислений. Недаром первый редактор электронных таблиц получил название VisiCalc.

Сохранение электронной таблицы. Операция сохранения электронной таблицы аналогична операциям сохранения графического объекта в графическом редакторе или электронного документа в текстовом редакторе. Отличие редактора Excel состоит в том, что он работает с виртуальными книгами, на листах которых расположены таблицы. Мы работали с одной таблицей, а остальные листы с таблицами остались пустыми.

Вывести другие листы книги на экран можно с помощью ярлычков Лист 1, Лист 2, Лист 3 в нижнем левом углу окна (см. рис. 110). Для этого по ярлычку достаточно щёлкнуть левой клавишей мыши. Если по ярлычку щёлкнуть правой клавишей мыши, то выводится меню, с помощью которого лист можно удалить из книги или добавить в книгу новый лист.

Таким образом, Excel сохраняет на диске виртуальные книги из электронных таблиц. Сохраним книгу с созданной электронной таблицей в файле ex99. Расширение .xlsx редактор электронных таблиц Excel 2010 добавит автоматически.

☐ Вопросы и задания

1. Какие объекты отображаются в строке состояния Excel 2010?
2. Что называют ячейкой электронной таблицы?

3. Что такое содержимое ячейки электронной таблицы?

4. Перечислите, какие виды информации могут быть записаны в ячейке электронной таблицы.

5. Как образуется имя ячейки электронной таблицы?

6. Определите, в каком столбце и в какой строке находится ячейка:

1) K34; 4) IA1;
2) BB543; 5) B1234.
3) DA5879;

7. Что такое табличный курсор?

8. Какую роль играет табличный курсор в редакторе электронных таблиц?

9. Что такое блок электронной таблицы?

10. Как строится имя блока электронной таблицы?

11. Запишите имя блока, который образуется пересечением:

1) столбцов C, D, E со строками 5, 6, 7, 8;
2) столбцов F, G со строками 1, 2, 3;
3) столбцов B, C, D, E со строками 9, 10.

12. По каким правилам вводятся числа в ячейки электронной таблицы?

13. Что такое строка формул?

14. Какой вид выравнивания обычно используется при отображении чисел в ячейках электронной таблицы?

15. Как очистить ячейку электронной таблицы?

16. Как вводится текст в ячейку электронной таблицы?

17. Какой вид выравнивания обычно используется при отображении текста в ячейках электронной таблицы?

18. Как можно проверить содержимое ячейки электронной таблицы?

19. Как подсчитать сумму чисел в столбце электронной таблицы?

20. Как найти максимальное число в столбце электронной таблицы?

21. Что такое книга редактора Excel?

22. С помощью Excel найдите сумму следующих чисел: 100; 139,4; 2,45; 13; 86,2; 54,86.

23. С помощью Excel среди чисел из задания 22 найдите максимальное и минимальное.

§ 41. Табличный расчёт успеваемости

Успеваемость класса или группы учащихся оценивается средними арифметическими оценками по отдельным предметам. Редактор электронных таблиц Excel позволяет проводить эффективные табличные расчёты успеваемости.

■ Упражнение 110

В таблице 14 представлены оценки группы учащихся 9 класса по итогам II четверти. Построим таблицу расчёта успеваемости.

Таблица 14

№	Фамилия, имя	Математика	Физика	Химия	Биология	Информатика и ИКТ	Литература	Русский язык	История	География	Иностр. язык	Искусство
1	Авдей А.	5	5	4	5	4	4	5	4	5	5	5
2	Быстров Ю.	4	4	3	4	5	4	3	4	4	3	5
3	Зимина И.	5	4	5	5	4	4	4	4	3	4	5
4	Жогло Н.	4	5	4	5	4	4	5	5	4	4	4
5	Киселева В.	5	5	5	5	5	5	5	5	5	5	5
6	Корчевский О.	4	3	3	4	4	5	4	4	3	3	4
7	Лабунова Л.	4	3	3	4	3	3	4	4	5	3	5
8	Петров А.	3	4	4	3	3	4	5	4	5	3	4
9	Пигулевич С.	5	4	4	5	5	3	4	4	3	4	5
10	Серова Р.	3	3	3	3	3	3	3	4	3	4	4
11	Шербаф А.	4	4	5	4	4	4	4	5	4	4	5
12	Якиманов С.	3	4	4	3	5	4	4	5	4	3	4

Название табличной модели «Расчёт успеваемости» введём в ячейку C1 (рис. 111), а ниже введём данные из таблицы 14.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1			Расчет успеваемости												
2	№	Фамилия, имя	Математика	Физика	Химия	Биология	Информатика и ИКТ	Литература	Русск. язык	История	География	Иностр. язык	Искусство	Сред. ученика	
3	1	Авдей А.	5	5	4	5	4	4	5	4	5	5	5		
4	2	Быстров Ю.	4	4	3	4	5	4	3	4	4	3	5		
5	3	Зимина И.	5	4	5	5	4	4	4	4	3	4	5		
6	4	Жогло Н.	4	5	4	5	4	4	5	5	4	4	4		
7	5	Киселева В.	5	5	5	5	5	5	5	5	5	5	5		
8	6	Корчевский О.	4	3	3	4	4	5	4	4	3	3	4		
9	7	Лабунова Л.	3	4	4	3	3	4	5	4	5	3	4		
10	8	Петров А.	5	4	4	5	5	3	4	4	3	4	5		
11	9	Пигулевич С.	5	4	4	5	5	3	4	4	3	4	5		
12	10	Серова Р.	3	3	3	3	3	3	3	4	3	4	4		
13	11	Шербаф А.	4	4	5	4	4	4	4	5	4	4	5		
14	12	Якиманов С.	3	4	4	3	5	4	4	5	4	3	4		
15		Сред. по предмету													

Рис. 111

Сначала в ячейки строки 2 последовательно вводим заголовки столбцов таблицы. При этом не важно, что введённые горизонтально названия предметов временно перекрывают друг друга. Это поправимо.

Форматирование ячеек. В ячейках B1 и B2 установим выравнивание по центру и посередине. Выделяем эти ячейки и щёлкаем по нужным кнопкам на вкладке «Главная» в группе «Выравнивание».

Чтобы названия предметов не налегали друг на друга, поменяем на 90° их ориентацию (направление надписи). Устанавливаем табличный курсор на ячейке C2 и на вкладке «Главная» щёлкаем по кнопке правее названия группы «Выравнивание».

Рис. 112

Открывается вкладка «Выравнивание» диалогового окна «Формат ячеек». В поле «Ориентация» перетаскиваем мышью красный маркер справа от слова «Надпись» на 90° вверх по окружности (рис. 112) и щёлкаем по кнопке **ОК**. Название предмета «Математика» в электронной таблице поворачивается и размещается вертикально. Высота строки 2 увеличивается.

Стиль форматирования ячейки C2 перенесём на остальные ячейки строки 2 с помощью кнопки **Формат по образцу** из группы «Буфер обмена» на вкладке «Главное».

Изменение высоты строк и ширины столбцов. Высоту строк и ширину столбцов электронной таблицы меняют перетаскиванием мыши.

Чтобы увеличить высоту строки, указатель мыши устанавливают в столбце заголовков на нижнюю границу строки. Указатель мыши должен принять вид ✢. Перетаскивание мыши меняет высоту строки. При необходимости подбираем высоту строки 2 так, чтобы уместились названия предметов.

Аналогично выполняется изменение ширины столбцов. В частности, столбец B нужно увеличить, чтобы уместились фамилии, а остальные столбцы уменьшить, чтобы уместились только номера и оценки.

Для изменения ширины столбца указатель мыши устанавливают в строке заголовков на правую границу столбца. При этом указатель мыши должен принять вид ✢. Перетаскивание указателя меняет положение границы столбца.

Меняем ширину столбцов таблицы расчёта успеваемости.

Переходим к заполнению таблицы. Столбец A с номерами строк пока не заполняем. Вносим фамилии в ячейки столбца B. Вносим оценки.

Заполнение строки или столбца данными. Мы умышленно оставили незаполненным первый столбец таблицы. Редактор Excel позволяет в некоторых случаях автоматизировать процесс ввода данных в электронную таблицу. Познакомимся с такими возможностями.

Введём в ячейку A3 число 1 и установим на ней табличный курсор. Обратим теперь внимание на особенность рамки табличного

курсора. В правом нижнем углу рамки находится маркер в виде квадрата, который называется ***маркером заполнения*** (рис. 113).

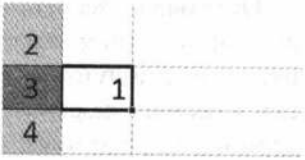


Рис. 113

При попадании на маркер заполнения указатель мыши принимает вид чёрного креста. Именно в этот момент нажимаем левую клавишу мыши и перетаскиваем мышь вниз. Рамка табличного курсора расширяется. После отпускания клавиши мыши во всех выделенных ячейках записывается число 1. Операция называется ***заполнением вниз*** содержимым ячейки A3. Аналогично можно проделать операции заполнения вверх, влево и вправо.

Результат операции в данном случае нас не устраивает, поэтому очистим ячейки столбца, кроме ячейки A3, и познакомимся с другим видом операции заполнения.

В ячейке A3 оставляем единицу, а в ячейке A4 записываем число 2. Обе ячейки выделяем протяжкой мыши (не затрагивая маркер заполнения). А вот теперь, пользуясь маркером заполнения, проводим операцию заполнения вниз содержимым блока A3:A4. Редактор автоматически строит в столбце A нумерацию строк таблицы. Таким способом при необходимости можно построить и нумерацию столбцов.

Ввод формул. Добавим в таблицу для формул расчёта средних арифметических значений столбец N и строку 15. В ячейку N2 вводим заголовок «Сред. ученика», в ячейку B15 — «Сред. по предмету» (см. рис. 111).

Будем вводить формулы расчёта средних арифметических. Сначала устанавливаем табличный курсор на ячейку N3 и используем пункт **Среднее** дополнительного меню кнопки **Автосумма**. Рамка выделяет ячейки с числами слева от ячейки N3. Нажимаем клавишу **Enter** клавиатуры и завершаем операцию.

Чтобы средняя оценка отображалась с двумя цифрами после запятой, ячейку N3 нужно отформатировать.

Устанавливаем на неё табличный курсор и щёлкаем на вкладке «Главная» по кнопке правее названия группы «Число». В диалоговом окне «Формат ячеек» открываем вкладку «Число», выбираем числовой формат «Числовой» и устанавливаем число десятичных знаков «2».

Остальные формулы для вычисления средних в столбце N вводим заполнением вниз содержимым ячейки N3. Все формулы аккуратно перепишутся в ячейки. При этом редактор Excel автоматически изменяет формулы так, что в ячейке N4 вычисляется среднее арифметическое чисел в ячейках блока C4:M4, в ячейке N5 — среднее чисел в ячейках блока C5:M5 и т. д.

Устанавливаем табличный курсор на ячейку C15. Вводим формулу среднего арифметического (меню кнопки **Автосумма** пункт **Среднее**). Рамка выделяет ячейки с числами сверху от ячейки C15. Нажимаем клавишу **Enter** клавиатуры. Ячейку C15 при необходимости форматируем. Формулы вычисления среднего арифметического в другие ячейки строки 15 вводим заполнением вправо содержимым ячейки C15.

В ячейку N15 введём формулу вычисления среднего арифметического всех индивидуальных оценок таблицы. Устанавливаем на ячейку табличный курсор и вводим формулу среднего (меню кнопки **Автосумма** пункт **Среднее**). На появившуюся рамку не обращаем внимания. Рамка — это только подсказка пользователю. Указателем мыши выделяем весь блок таблицы с индивидуальными оценками и нажимаем клавишу **Enter** клавиатуры.

Обрамление таблиц. В типографских документах названия граф таблицы отделяются от содержимого таблицы горизонтальными линиями. Кроме того, графы разделяются вертикальными линиями.

Редактор электронных таблиц Excel обладает оформительскими возможностями типографских машин и позволяет создавать таблицы с любыми горизонтальными и вертикальными линиями. Следует только иметь в виду, что видимая на экране в электронных таблицах сетка является условной и на печать не выводится.

Для создания печатаемой рамки вокруг любого блока ячеек используют диалоговое окно «Формат ячеек». Оно вызывается на вкладке «Главная» щелчком по кнопке со стрелкой правее названия группы «Выравнивание». Открывают вкладку «Граница» и далее следуют подсказкам в этом окне.

Выделим таблицу расчёта успеваемости и сделаем обрамление тонкой линией для всех её ячеек, а толстой линией — для внешней рамки. Вот теперь таблица расчёта успеваемости готова. Сохраняем книгу с таблицей в файле ex100.xlsx на диске.

☐ Вопросы и задания

1. Какие кнопки на вкладке «Главная» вызывают диалоговое окно «Формат ячеек»?

2. Какие операции форматирования ячейки можно проделать, не вызывая диалоговое окно «Формат ячеек»?

3. Какую форму имеет маркер заполнения и где он расположен?

4. Как проводится операция заполнения вниз содержимым одной ячейки?

5. Чем различаются операции заполнения вниз содержимым одной ячейки и заполнения вниз содержимым блока из двух соседних ячеек столбца?

6. Как организовать нумерацию строк с помощью заполнения вниз?

7. Создайте таблицы успеваемости своего класса за две четверти. Сравните средние оценки.

§ 42. Формулы

Познакомимся подробнее с вычислительными возможностями редактора электронных таблиц Excel.

Формулы — это выражения, по которым выполняются вычисления в ячейках электронной таблицы.

Формула начинается со знака равенства «=» и может включать следующие элементы: ссылки, знаки операций, константы и функции. В формулах можно использовать скобки.

Ссылки — это имена ячеек или блоков электронной таблицы.
Константы — это числа или тексты.

Арифметические операции обозначаются привычным образом, как в выражениях языка программирования JavaScript. Например:

```
=3+4/A4        =A2*B2        =(A2-B2)/A5        =5*C3-B3/D1
```

Возведение в степень обозначается символом «^» («крышка»). Например:

```
= A4^2+A5^2                  =(A2-B2)^3
```

Упражнение 111

Создадим электронную таблицу вычисления значений функции $y = x^2 + x - 4$ для разных значений аргумента x.

Для решения вычислительных задач электронные таблицы обычно делятся на три раздела с заголовками «Исходные данные», «Расчётная таблица», «Результаты». В нашем случае расчёты сложными не являются: раздел «Расчётная таблица» выделять не будем.

Открываем новую книгу редактора Excel. Вводим исходные данные:

```
<A1>Исходные данные
<A2>1  <B2>: значение аргумента
```

Замечание. Если при вводе текста допущена ошибка, то её можно исправить. Для этого надо установить на ячейку табличный курсор, в строке формул произвести исправления и нажать клавишу **Enter** клавиатуры. Нажатие клавиши-стрелки в этом случае редактирование не завершает.

Далее пропускаем в таблице строку и вводим:

```
<A4>Результат
```

В ячейку A5 вводим формулу заданной функции, считая, что в ячейке A2 находится значение аргумента. Формула получит вид:

```
<A5>=A2^2+A2-4
```

	A	B	C	D
1	Исходные данные			
2	1	: значение аргумента		
3				
4	Результат			
5	-2	: значение функции		
6				
7				

Рис. 114

Для ввода формулы устанавливаем табличный курсор на ячейку A5 (язык ввода — английский). Вводим с клавиатуры знак равенства «=». Далее имя ячейки A2 можно с клавиатуры не вводить, а сделать по ячейке щелчок левой клавишей мыши прямо в таблице. Далее с клавиатуры вводим знак «^», затем 2, «+» и опять щёлкаем по ячейке A2. Осталось ввести с клавиатуры знак «–» и 4. Если с первого раза ввод не удаётся, прекращаем ввод клавишей **Esc** и делаем новую попытку. Ввод формулы завершаем нажатием клавиши **Enter**. Результат тут же отображается в ячейке A5. Далее вводим:

```
<B5>: значение функции
```

В результате наша таблица получит вид, представленный на рисунке 114.

По разным значениям аргумента, в том числе и дробным, в таблице автоматически вычисляются значения функции.

■ Упражнение 112

В двух классах проведена контрольная работа по биологии. Данные об оценках за работу приведены в таблице 15. Пользуясь электронными таблицами, вычислим процентные доли оценок для каждого класса.

Таблица 15

Оценка	9 «А»	9 «Б»
«5»	7	5
«4»	11	12
«3»	6	8
«2»	2	1

Подробное решение упражнения приведём для 9 «А» класса. Открываем новый лист книги Excel. В электронной таблице выделяем единственный раздел «Расчётная таблица», так как исходные данные и результаты (проценты) удобнее сопоставлять в единой таблице.

```
<A1>Расчётная таблица
<A2>"5"        <B2>7
<A3>"4"        <B3>11
<A4>"3"        <B4>6
<A5>"2"        <B5>2
```

Для расчётов понадобится сумма оценок, которую мы введём с помощью кнопки Σ **Автосумма** в ячейку B6.

В столбце C введём расчётные формулы, а столбец D заполним символом «%».

```
<C2>=B2*100/B6 <D2>%
<C3>=B3*100/B6 <D3>%
<C4>=B4*100/B6 <D4>%
<C5>=B5*100/B6 <D5>%
```

	A	B	C	D
1	Расчётная таблица			
2	"5"	7	26,9	%
3	"4"	11	42,3	%
4	"3"	6	23,1	%
5	"2"	2	7,7	%
6		26		

Рис. 115

	A	B	C	D	E
1	Перевод старинных английских мер длины				
2	3	мили			
3	6	фарлонги			
4	214	ярды			
5	5	футы			
6					

Рис. 116

Для наглядности ячейки блока A2:B5 надо отформатировать кнопкой **Выровнять по центру**, а в ячейках блока C2:C5 задать отображение одного десятичного знака (диалоговое окно «Формат ячеек», закладка «Число», формат «числовой», количество десятичных знаков «1»). Результат показан на рисунке 115. Расчёт процентных долей оценок 9 «Б» класса выполните самостоятельно.

■ Упражнение 113

В разных странах существовали и до сих пор существуют самые разнообразные меры длины, площади, веса. Вспомним старинные английские меры длины и с помощью электронных таблиц переведём в метры 3 легальных мили 6 фарлонгов 214 ярдов 2 фута. Известно, что 1 легальная миля = 8 фарлонгов, 1 фарлонг = 220 ярдов, 1 ярд = 3 фута, 1 фут = 0,3048 м.

Подобные задачи ученики школ дореволюционной России решали, начиная с 1 класса.

Открываем новый лист книги Excel. В ячейку A1 вносим название таблицы «Перевод старинных английских мер длины». Далее в первом столбце вводим исходные данные, а во втором столбце — названия мер:

```
<A2>3          <B2>мили
<A3>6          <B3>фарлонги
<A4>214        <B4>ярды
<A5>2          <B5>футы
```

Выделяем блок B2:B5 и заливаем его ячейки серым цветом с помощью меню кнопки **Цвет заливки** на вкладке «Главная» в группе «Шрифт» (рис. 116). Правее блока B2:B5 будем строить расчётную таблицу.

Сначала вводим переводные коэффициенты:

<C2>8
<C3>220
<C4>3
<C5>1

Следующий блок D2:D5 будет содержать длины английских мер в метрах. Заполняем блок снизу. Сначала длину фута в метрах:

<D5>0,3048

Далее вводим формулу вычисления ярда в метрах:

<D4>=C4*D5

Устанавливаем табличный курсор на ячейку D4 и заполняем вверх содержимым ячейки D4 блок D2:D3. Блок E2:E5 будет содержать части исходной длины, переведённые в метры.

<E2>=A2*D2

Остальные ячейки блока заполняем вниз содержимым ячейки E2. Осталось просуммировать ячейки блока E2:E5. Устанавливаем табличный курсор на ячейку E6 и проводим суммирование с помощью кнопки **Автосумма**.

Для определённости в ячейку F6 вводим текст «метры» и заливаем её серым цветом. Переводная таблица готова. Ответ: 6231,33 м.

В начале параграфа мы говорили, что в формулах Excel могут использоваться функции.

Функции — это заранее определённые формулы, которые включают заданные величины-аргументы.

Функция редактора Excel называется ***блочной***, если её аргументом может служить блок электронной таблицы. Функция суммирования (СУММ) является простейшим примером блочной функции. Блочными являются и другие функции, представленные в меню кнопки **Автосумма**.

В Excel 2010 ввод функций связан с кнопкой **Вставить функцию** в строке формул, которая вызывает диалоговое окно «Мастер функций». На вкладке «Формулы» есть целая группа «Библиотека функций» с кнопками — категориями функций. Вводить имена функций можно и с клавиатуры. Функции в Excel 2010 имеются самые разнообразные, например:

КОРЕНЬ() — вычисление корня квадратного от значения аргумента;

SIN() — вычисление синуса аргумента;

COS() — вычисление косинуса аргумента;

ПИ() — вывод значения числа π;

СЕГОДНЯ() — вывод текущей даты;

ПРОПИСН() — перевод всех букв текстового аргумента в прописные;

СТРОЧН() — перевод всех букв текстового аргумента в строчные;

РИМСКОЕ() — перевод натурального десятичного аргумента в римскую систему счисления.

■ Упражнение 114

С помощью электронных таблиц переведём текст «тридцать три коровы» в текст из прописных букв.

Открываем (добавляем) новый лист книги Excel. Вводим текст «тридцать три коровы» в ячейку A1. На ячейку A2 устанавливаем табличный курсор. Вводим знак равенства «=» и имя функции ПРОПИСН (можно строчными буквами). Далее вводим скобку «(», щёлкаем по ячейке A1 и вводим скобку «)». Ввод формулы завершаем нажатием клавиши **Enter**. Результат тут же отображается в ячейке A2.

■ Упражнение 115

С помощью электронных таблиц переведём в римскую систему счисления десятичное число 2489.

На том же листе Excel вводим аргумент 2489 в ячейку A3. На ячейку B3 устанавливаем табличный курсор. Вводим знак равенства «=» и имя функции РИМСКОЕ (можно строчными буквами). Далее вводим скобку «(», щёлкаем по ячейке A3 и вводим скобку «)». Ввод формулы завершаем нажатием клавиши **Enter**. Результат MMCDLXXXIX тут же отображается в ячейке B3. Для аргумента функции РИМСКОЕ() существует ограничение (< 4000).

Сохраняем книгу Excel с упражнениями в файле ex101-105.xlsx.

☐ Вопросы и задания

1. Что такое формула в редакторе электронных таблиц Excel?

2. Какие элементы могут включать формулы?

3. С помощью редактора электронных таблиц создайте таблицу для вычисления значений функции $y = x^3 + 2x - 5$ для разных значений аргумента x.

4. С помощью редактора электронных таблиц создайте таблицу для вычисления значений функции $z = x^2 - 4xy + 3y^2$ для разных значений x и y.

5. С помощью редактора электронных таблиц создайте таблицу для вычисления длины гипотенузы прямоугольного треугольника по заданным значениям длин катетов.

6. С помощью электронной таблицы упражнения 111 переведите в метры:

1) 6 легальных миль 2 фарлонга 101 ярд 2 фута;

2) 2 легальных мили 7 фарлонгов 1 фут;

3) 4 фарлонга 45 ярдов.

7. Старинные русские меры длины — версты, сажени, аршины. 1 верста = 500 саженей, 1 сажень = 3 аршина, 1 аршин = 71,12 см. С помощью электронной таблицы переведите в метры:

1) 4 версты 52 сажени 2 аршина;

2) 10 верст 435 саженей 1 аршин;

3) 1 версту 15 саженей.

8. Массу (вес) на Руси измеряли в пудах и фунтах. 1 пуд = = 40 фунтов, 1 фунт = 0,40951 кг. С помощью электронной таблицы переведите в килограммы:

1) 3 пуда 20 фунтов;

2) 1 пуд 34 фунта;

3) 27 фунтов;

4) 2 пуда 2 фунта.

9. Укажите отличия в обозначении одних и тех же функций в редакторе Excel и на языке JavaScript.

10. С помощью электронных таблиц определите версию операционной системы, используя формулу =ИНФОРМ("версиясс").

11. С помощью электронных таблиц переведите в двоичную систему счисления десятичное число 214.

12. С помощью электронных таблиц переведите текст «ЭЛЕКТРОННЫЕ ТАБЛИЦЫ» в текст из строчных букв.

13*. С помощью электронных таблиц переведите в римскую систему счисления шестнадцатеричное число A5D.

§ 43. Табличное моделирование

Редактор электронных таблиц Excel позволяет строить в табличной форме самые разнообразные модели. В данном параграфе мы познакомимся с двумя видами таких моделей.

Модель роста и убывания. Рассмотрим следующую задачу. Лесной участок оценивается в 150 тыс. м^3 деловой древесины. Каждый год этот объём увеличивается на 7% за счёт естественного прироста. Спустя 3 года на хозяйственные нужды стали вырубать 20 тыс. м^3 древесины ежегодно. Наступит ли уменьшение объёма деловой древесины на участке до 100 тыс. м^3 и через сколько лет? Каков ответ на первый вопрос, если спустя 4 года процент прироста уменьшился до 5%, а спустя 8 лет объём вырубки увеличился до 25 тыс. м^3 в год?

■ Упражнение 116

С помощью электронных таблиц построим табличную модель для решения вышеизложенной задачи.

Теория вопроса. Сначала введём обозначения. Перенумеруем годы, начиная с нулевого (начального). Расчёты будем вести по состоянию на 1 января каждого года.

Пусть объём деловой древесины в i-м году будет $x(i)$, процент прироста древесины — $p(i)$, а объём вырубки — $v(i)$. Тогда объём $x(i)$ древесины в i-м году — это объём древесины в предыдущем $(i-1)$-м году с увеличением на $p(i-1)$ процентов и за вычетом вырубки предыдущего года $v(i-1)$. Получаем основную формулу модели:

$$x(i) = x(i-1)(1 + p(i-1)/100) - v(i-1).$$

Построение электронной таблицы. Открываем новую книгу Excel. В ячейку B1 вводим название «Модель роста и убывания», а в ячейку A2 — название раздела электронной таблицы «Исходные данные».

Исходными данными для задачи являются начальный объём древесины (150 000), начальный процент прироста (7) и начальный объём вырубки (0). Вводим исходные данные:

```
<A3>150000      <B3>: начальный объём древесины (м3)
<A4>7           <B4>: начальный процент прироста
<A5>0           <B5>: начальный объём вырубки
```

В ячейку A7 вводим заголовок раздела «Расчётная таблица».

Строку 8 отведём для заголовков столбцов электронной таблицы:

```
<A8>Год    <B8>Объем    <C8>Процент    <D8>Вырубка
```

	A	B	C	D	E
1		Модель роста и убывания			
2	Исходные данные				
3	150000	: начальный объем древесины (куб.м)			
4	7	: начальный процент прироста			
5	0	: начальный объем вырубки			
6					
7	Расчётная таблица				
8	Год	Объем	Процент	Вырубка	
9					

Рис. 117

Выделим блок A8:D8 и зальём ячейки блока серым цветом с помощью кнопки **Цвет заливки**. На рисунке 117 приведён вид электронной таблицы до ввода формул.

В строке 9 должны отображаться начальные данные для нулевого года, поэтому формулы здесь будут самые простые:

```
<A9>0        <B9>=A3        <C9>=A4        <D9>=A5
```

Основной для расчётов является строка 10:

```
<A10>1    <B10>=B9*(1+C9/100)-D9        <C10>=C9        <D10>=D9
```

Далее заполним ещё 20 строк расчётной таблицы (включая строку 30). Столбец A заполняем вниз содержимым блока A9:A10. Затем выделяем блок B10:D10 и заполняем вниз содержимым этого блока ячейки ещё 20 строк.

По условию задачи спустя 3 года начались вырубки. В столбец «Вырубка» в строке с номером года 3 вводим число 20 000.

Для ответа на первый вопрос задачи проанализируем расчётную таблицу. Объём деловой древесины на участке уменьшится до 100 тыс. м^3 через 12 лет. Для ответа на второй вопрос внесём изменения в графы «Процент» (для 4-го года) и «Вырубка» (для 8-го года) и проанализируем таблицу.

Сохраняем книгу Excel с моделью в файле ex106.xlsx.

Модель температурных режимов. Пусть имеется квадратная однородная металлическая пластина, которая является деталью некоторого устройства. Во время работы пластины на её краях поддерживается температура 20 °C. В центральной точке пластина нагревается лучом лазера до 200 °C. Каким будет распределение температур во внутренних точках пластины?

Упражнение 117

С помощью электронных таблиц построим модель температурных режимов металлической пластины.

Теория вопроса. Для построения табличной модели условно разобьём пластину на небольшие квадраты общим числом 25×25. Будем считать, что внутри каждого квадрата температура постоянна. Это предположение справедливо лишь приближённо, но чем меньше сторона внутреннего квадрата, тем меньше ошибка. Будем считать, что теплообмен любого внутреннего квадрата с окружающей средой незначителен в сравнении с теплообменом, который происходит между квадратами. С учётом этих предположений из физических законов, описывающих перенос тепла, следует, что температуру любого квадрата можно считать равной среднему арифметическому температур примыкающих квадратов:

$$t_{\text{кв}} = \frac{t_{\text{л}} + t_{\text{в}} + t_{\text{п}} + t_{\text{н}}}{4}.$$

Исключение составляют только квадраты, находящиеся на краю пластины. По условию они имеют постоянную температуру 20 °C.

Составление электронной таблицы. Открываем новую книгу Excel. Название модели вводить не будем, чтобы иметь возможность строить модель, начиная с ячейки A1.

При построении табличной модели в Excel каждому условному квадрату поставим в соответствие ячейку электронной таблицы. Удобно и сами ячейки сделать квадратными. Чтобы перенастроить электронную таблицу, выделим её всю, щёлкнув по ячейке на пересечении строки и столбца заголовков.

На вкладке «Главная» в группе «Ячейки» щёлкаем по кнопке **Формат**, а в появившемся меню — по пункту **Высота строки**. Появляется диалоговое окно, в котором вводим высоту 22.

Не снимая выделение, опять щёлкаем по кнопке **Формат**, а в появившемся меню — по пункту **Ширина столбца**. Появляется диалоговое окно, в котором вводим ширину 4.

Температуру будем показывать в целых числах. Не снимая выделение, кнопкой правее названия группы «Число» вызываем диалоговое окно «Формат ячеек». На вкладке «Число» выбираем формат «Числовой» и в поле «Число десятичных знаков» ставим 0. Снимаем выделение, щёлкнув в любом месте электронной таблицы.

Чтобы вся модель была видна в окне редактора, устанавливаем соответствующий масштаб отображения.

Вводим значение 20 в крайние ячейки модели. Для этого вводим 20 в ячейку A1. Заполняем вниз содержимым ячейки A1 строки до 25 включительно. Выделение не снимаем и заливаем ячейки блока серым цветом. Далее устанавливаем табличный курсор на ячейку A1 и заполняем вправо все ячейки до столбца Y включительно (это 25 столбцов). Аналогично заполняем блоки Y1:Y25 и A25:X25, не забывая про серую заливку.

Осталось ввести формулы в квадраты-ячейки. Но легко предвидеть, что во внутренних соседних ячейках будут стоять формулы, которые ссылаются друг на друга. Редактор Excel без специальной настройки воспринимает такие ссылки как ошибки. Поэтому командой меню **Файл|Параметры** вызываем окно «Параметры Excel». На вкладке «Формулы» в разделе «Параметры вычислений» помечаем флажок «Включить итеративные вычисления», а в поле «Предельное число итераций» вводим число 500. Щёлкаем по кнопке **ОК**.

Теперь вводим формулу

```
<B2>=(C2+B3+A2+B1)/4
```

Ставим табличный курсор на ячейку B2 и заполняем вправо содержимым ячейки B2 блок B2:B24. Выделение не убираем и заполняем вниз блоком B2:B24 всю внутренность модели. Обратите внимание на то, как в результате визуально проявляются автоматические вычисления.

Заливаем серым цветом центральную ячейку M13 и вводим в неё число 200. Модель через какое-то время выдаёт искомое распределение температур. Сохраняем книгу Excel с моделью в файле ex107.xlxs на диске.

☐ Вопросы и задания

1. Изменив построенную модель роста и убывания, решите следующую задачу: «На пищевом комбинате в установку по производству дрожжей заложена 1 т дрожжевой массы. При поддержании оптимальной температуры за сутки масса дрожжей возрастает на 150%. 1,5 т массы ежедневно пускают в производство. В результате неполадки температура в установке повысилась, и прирост составил 160%. Через сколько суток масса дрожжей в установке может достигнуть 3,5 т?»

2. В модели температурных режимов задайте в центральной точке ту же температуру, что и на краю. Как это отразится на температурном режиме?

3. В модели температурных режимов заполните вниз содержимым ячейки M12 ячейку M13. Найдите температуру в центре пластины, если на верхнем краю температура 20 °C, на нижнем — 44 °C, а на левом и правом краях пластины температура возрастает сверху вниз на 1 °C на каждом квадрате.

4. В модели температурных режимов заполните вниз содержимым ячейки M12 ячейку M13. Найдите температуру в центре пластины, если в левом верхнем углу температура 20 °C, вдоль горизонтальных краёв она возрастает слева направо на 1 °C на каждом квадрате, а на вертикальных краях аналогично сверху вниз.

§ 44. Моделирование с использованием деловой графики

Редактор электронных таблиц Excel обладает широкими возможностями в области создания деловой графики. Проиллюстрируем эти возможности на примере.

■ Упражнение 118

Процентное содержание газов в сухом воздухе при температуре 0 °C и атмосферном давлении 760 мм рт. ст. по объёму составляет: азот — 78,09%; кислород — 20,95%; аргон (и другие инертные газы) — 0,93%; двуокись углерода — 0,03%. Создадим диаграмму состава воздуха.

Открываем новую книгу Excel и вносим в электронную таблицу Excel данные по схеме, приведённой на рисунке 118.

Выделяем блок ячеек A1:C5 с данными.

За вставку диаграмм в редакторе отвечает группа инструментов «Диаграммы» на панели «Вставка». Можно вызвать диалоговое окно «Вставка диаграммы» кнопкой со стрелкой правее названия группы или воспользоваться кнопками-категориями диаграмм.

	A	B	C
1		Состав воздуха	
2	Азот	78,09	%
3	Кислород	20,95	%
4	Аргон	0,93	%
5	Двуокись углерода	0,03	%
6			

Рис. 118

Поскольку в упражнении речь идёт об относительном со-

ставе воздуха, остановимся на круговой диаграмме и щёлкнем по кнопке категории **Круговая**.

Открывается меню, в котором предлагается несколько шаблонов круговых диаграмм. Останавливаемся на первом шаблоне диаграммы «Круговая» и щёлкаем по нему. В области просмотра появляется прямоугольная область «Полотно» с диаграммой.

Одновременно в меню появляются три дополнительных пункта: «Конструктор», «Макет» и «Формат». При этом на ленте сразу открывается вкладка «Конструктор».

В разделе «Макеты диаграмм» подбираем шаблон оформления. Останавливаемся на шаблоне «Макет 6». Отметим, что данный макет содержит пояснительный элемент «Легенда», в котором отображается перечень параметров с указанием цвета маркеров на диаграмме. В разделе «Стили диаграмм» можно подобрать другие цвета.

На диаграмме осталось поменять цвет чисел, расположенных в круге, и положение чисел, расположенных над кругом. Чтобы внести изменения, текст или число надо выделить щелчком. Иногда при этом выделяется несколько элементов, тогда щелчок повторяют. Для изменения атрибутов выделенного элемента пользуются кнопками группы «Шрифт» вкладки «Главная». Выделенный элемент можно перетащить указателем мыши. Диаграмма готова (рис. 119).

Рис. 119

Диаграмму можно выделить щелчком мыши и перенести по листу перетаскиванием мышью в требуемое место.

Сохраняем книгу Excel с построенной диаграммой в файле ex108—109.xlsx.

Построение графика функции. До сих пор для построения графика функции мы использовали возможности языка программирования JavaScript. Редактор электронных таблиц Excel также позволяет строить графики функций, используя возможности деловой графики.

Упражнение 119

Построим график функции $y = x^3 + 2x^2 + x$ на отрезке $[-2, 2]$, используя возможности электронных таблиц.

Для построения графика функции надо сначала построить таблицу значений аргумента и функции, а затем по таблице построить точечную диаграмму, которую мы и называем графиком функции.

Построение таблицы. График функции мы должны построить на отрезке $[-2, 2]$. Чтобы график получился достаточно точным, возьмём величину шага аргумента равной 0,2. Тогда в таблице должно быть 21 значение аргумента: −2,0; −1,8; ...; 1,8; 2,0.

Открываем новый лист книги Excel. Табличную модель назовём «График функции» и это название введём в ячейку B1. В модели обозначим только раздел «Расчётная таблица». Расчётная таблица будет иметь два столбца: для значений аргумента x и для значений функции f(x):

```
<A2>Расчётная таблица
<A3>x <B3>f(x)
```

В ячейку A4 введём начальное значение аргумента (−2,0), в ячейку B4 введём формулу функции с аргументом A4:

```
<B4>=A4^3+2*A4^2+A4
```

Осталось заполнить ещё 20 строк расчётной таблицы. В ячейку A5 записываем второе значение аргумента: −1,8. Ячейки столбца A (до ячейки A24 включительно) заполним вниз содержимым блока A4:A5, а ячейки столбца B заполним вниз содержимым ячейки B4.

Результат заполнения таблицы показан на рисунке 120.

Построение точечной диаграммы. Для построения точечной диаграммы выделяем расчётную таблицу (блок A4:B24). Щёлкаем на вкладке «Вставка» в группе «Диаграммы» по кнопке-категории **Точечная**.

Открывается меню, в котором предлагаются несколько шаблонов точечных диаграмм. Щелчком мыши выбираем шаблон «Точечная с гладкими кривыми» (третий в первой строке). В области просмотра появляется полотно с диаграммой.

На открывшейся вкладке «Конструктор» в группе «Макеты диаграмм» открываем меню и выбираем шаблон «Макет 10».

	A	B
1		График функции
2	Расчётная таблица	
3	x	f(x)
4	-2	-2
5	-1,8	-1,152
6	-1,6	-0,576
7	-1,4	-0,224
8	-1,2	-0,048
9	-1	0
10	-0,8	-0,032
11	-0,6	-0,096
12	-0,4	-0,144
13	-0,2	-0,128
14	0	0
15	0,2	0,288
16	0,4	0,784
17	0,6	1,536
18	0,8	2,592
19	1	4
20	1,2	5,808
21	1,4	8,064
22	1,6	10,816
23	1,8	14,112
24	2	18
25		

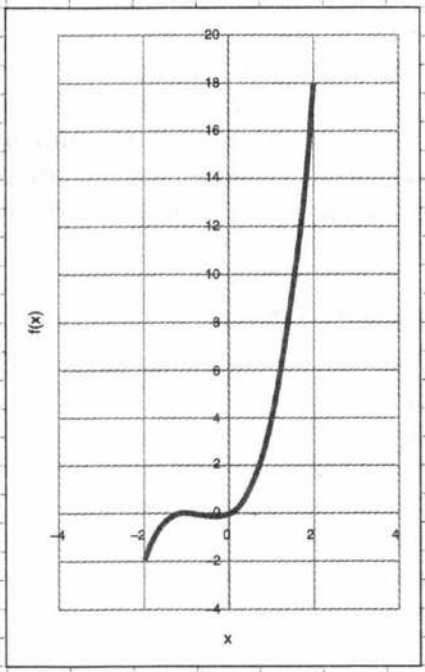


Рис. 120

На диаграмме добавляются линии сетки и к легенде (справа) добавляются надписи «Название оси» снизу и слева.

Легенда на графике никакой роли не играет, поэтому удалим её. Для этого щелчком мыши выделим легенду и нажмём клавишу **Del (Delete)** клавиатуры.

Введём названия осей. Щелчком мыши выделим надпись «Название оси» слева от графика. В строке формул введём текст «f(x)» и нажмём клавишу **Enter** клавиатуры. Введённый текст замещает текст, отображаемый в надписи по умолчанию.

Аналогичным образом вводим текст «x» вместо текста «Название оси» в надпись ниже графика.

Теперь курсором мыши перетащим полотно так, чтобы его левый верхний угол оказался в ячейке D2. Далее перетаскиванием за правый нижний угол полотна сузим график и вытянем его вниз,

чтобы величина единицы по горизонтальной оси максимально приблизилась к величине единицы по вертикальной оси. График практически построен (см. рис. 120).

На дополнительной вкладке «Макет» в группах «Подписи», «Оси» и «Фон» размещены инструменты для добавления и удаления элементов диаграммы. Например, щелчок по кнопке **Название диаграммы** открывает меню, в котором можно выбрать местоположение названия на полотне, чтобы потом ввести его.

Сохраняем книгу с диаграммами.

☐ Вопросы и задания

1. Какие кнопки-категории размещены в группе «Диаграммы» на вкладке «Вставка»?

2. Подсчитайте общее число разновидностей диаграмм в диалоговом окне «Вставка диаграммы».

3. Что такое легенда на диаграмме?

4. «Антигриппокапс» — комплексный препарат, который применяют как средство лечения гриппа и других лихорадочных состояний. Одна капсула препарата (0,32 г) включает: кислоту ацетилсалициловую — 0,15 г, кислоту аскорбиновую — 0,05 г, кальция лактат — 0,1 г, рутин — 0,01 г, димедрол — 0,01 г. Создайте с помощью электронных таблиц диаграмму состава «Антигриппокапса».

5. Постройте с помощью электронных таблиц график функции:

1) $f(x) = x^2(x - 12)$ на отрезке $[4, 7]$;

2) $f(x) = 2x^3 + 3x^2 - 12x + 1$ на отрезке $[0, 2]$;

3) $f(x) = 2 + x - x^3$ на отрезке $[0, 2]$;

4) $f(x) = x(x - 1)^2(x - 2)^3$ на отрезке $[-1, 1]$;

5) $f(x) = \frac{16x}{x^2 + 4}$ на отрезке $[0, 4]$.

§ 45. Моделирование полёта тела, брошенного под углом к горизонту

В качестве примера задачи, которую можно успешно решить с привлечением деловой графики, рассмотрим задачу моделирования полёта тела, брошенного под углом к горизонту.

■ Упражнение 120

Брошен камень с начальной скоростью 30 м/с под углом 60° к горизонту. Сопротивление воздуха учитывать не будем. Надо ответить на следующие вопросы.

1) Как далеко от места бросания упадёт камень?
2) Сколько секунд камень будет находиться в полёте?
3) Какова наибольшая высота взлёта камня?
4) Как скоро от начала полёта будет достигнута наивысшая точка полёта?

Задачу будем решать построением табличной модели.

Теория вопроса. Сначала рассмотрим математическую модель, которая давно используется в физике. В вертикальной плоскости полёта камня зададим декартову систему координат. Считаем, что точка вылета расположена в начале координат (рис. 121).

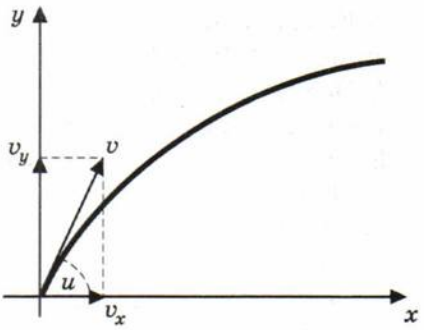


Рис. 121

Начальная скорость v (м/с) разлагается на составляющие v_x и v_y по углу бросания u в градусах. Формулы для этих составляющих имеют вид:

$$v_x = v \cdot \cos\left(\frac{u \cdot 3{,}14}{180}\right); \quad v_y = v \cdot \sin\left(\frac{u \cdot 3{,}14}{180}\right).$$

Положение тела в полёте определяется парой координат $x(t)$, $y(t)$. Зависимость координат от времени t описывается формулами:

$$x(t) = v \cdot \cos\left(\frac{u \cdot 3{,}14}{180}\right) \cdot t, \quad y(t) = v \cdot \sin\left(\frac{u \cdot 3{,}14}{180}\right) \cdot t - \frac{9{,}81 \cdot t^2}{2}.$$

Положение камня в полёте будем рассматривать в отдельные моменты времени. Пусть начальный момент равен 0, а последующие моменты отстоят друг от друга на одну и ту же величину 0,2 с, называемую *шагом времени*.

Создание электронной таблицы. Открываем новую книгу Excel. В ячейку A1 вводим название «Модель полёта тела»,

	A	B	C
1	Модель полёта тела		
2	Исходные данные		
3	30	: начальная скорость (м/с)	
4	60	: угол бросания (градусы)	
5			
6	Расчётная таблица		
7	Время	x(t)	y(t)
8	0	0	0
9	0,2	3,002758	4,998359
10	0,4	6,005516	9,604318
11	0,6	9,008274	13,81788
12	0,8	12,01103	17,63904
13	1	15,01379	21,0678
14	1,2	18,01655	24,10415
15	1,4	21,01931	26,74811
16	1,6	24,02207	28,99967
17	1,8	27,02482	30,85883
18	2	30,02758	32,32559
19	2,2	33,03034	33,39995
20	2,4	36,0331	34,08191
21	2,6	39,03586	34,37147
22	2,8	42,03861	34,25863
23	3	45,04137	33,77339
24	3,2	48,04413	32,88574
25	3,4	51,04689	31,6057
26	3,6	54,04965	29,93326
27	3,8	57,0524	27,86842
28	4	60,05516	25,41118
29	4,2	63,05792	22,56154
30	4,4	66,06068	19,3195

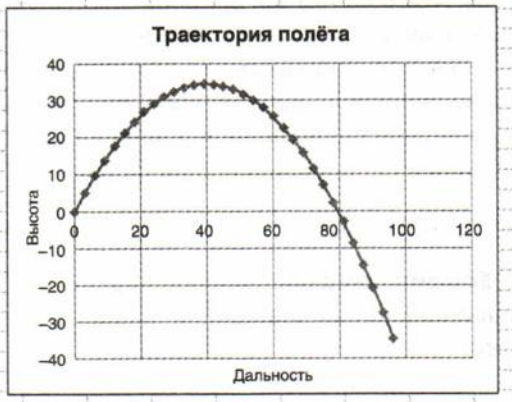


Рис. 122

а в ячейку A2 — название раздела «Исходные данные». Вводим следующие исходные данные.

<A3>30
<B3>: начальная скорость (м/с)
<A4>60
<B4>: угол бросания (градусы)

В ячейку A6 вводим название раздела «Расчётная таблица». В расчётной таблице будем отображать время с начала процесса (графа «Время»), координату $x(t)$ (графа «x(t)») и координату $y(t)$ (графа «y(t)»). Ячейки с названиями граф выравниваем по центру и заливаем серым цветом (рис. 122).

Первая строка расчётной таблицы содержит начальный момент времени и формулы:

<A8>0
<B8>=A8*A3*cos(A4*3,14/180)
<C8>=A8*A3*sin(A4*3,14/180) – 9,81*A8^2/2

Так как эти формулы должны содержать неизменные ссылки на ячейки A3 и A4, в формулах такие ссылки должны иметь специ-

альный вид — вид абсолютных ссылок: A3 и A4 (читается «доллар А доллар 4»).

Ввод абсолютных ссылок автоматизирован. После щелчка по ячейке B8 формула появляется в строке формул, где её можно редактировать. Щелчком мыши устанавливаем текстовый курсор в формуле на ссылку A3 и нажимаем клавишу **F4** клавиатуры. Ссылка A3 меняется на абсолютную A3. Аналогично меняем ссылку A4. Редактирование формулы завершаем нажатием клавиши **Enter** клавиатуры. Аналогичную операцию проделываем с формулой в ячейке C8.

В ячейку A9 вводим шаг времени 0,2. Выделяем блок A8:A9 и заполняем вниз его содержимым следующие 39 строк таблицы (включая строку 47). Выделяем блок B8:C8 и заполняем вниз его содержимым следующие 39 строк таблицы.

Построение диаграммы. Для наглядности решения задачи построим график функции (траекторию полета камня).

Сначала выделяем блок B8:C47 (второй и третий столбцы расчетной таблицы).

Далее на вкладке «Вставка» в разделе «Диаграммы» щёлкаем по кнопке-категории **Точечная**. Раскрывается меню с шаблонами, в котором выбираем шаблон «Точечная с гладкими кривыми и маркерами» (второй шаблон в первой строке). В области просмотра появляется полотно с диаграммой, а в меню — дополнительные пункты.

Наконец, на открывшейся вкладке «Конструктор» в группе «Макеты диаграмм» с помощью нижней кнопки со стрелкой в правой части поля открываем меню и выбираем шаблон оформления «Макет 8». Появляется легенда (снизу) и надпись «Название диаграммы» (сверху).

Удалим легенду, щелчком выделив её и нажав клавишу **Del (Delete)** клавиатуры.

Дадим графику название «Траектория полёта», горизонтальной оси — «Дальность», а вертикальной — «Высота».

Чтобы добавить названия осей, на вкладке «Макет» в группе «Подписи» щёлкаем по кнопке **Названия осей**. Открывается меню, в котором указатель мыши на любом пункте открывает ещё одно меню, где делается выбор способа размещения надписи (для горизонтальной — **Название под осью**, для вертикальной — **Повёр-**

нутое название). Осталось уже известным нам способом ввести названия графика и осей. Диаграмма примет окончательный вид (см. рис. 122).

Сохраняем книгу с моделью в файле ex110.xlsx на диске.

Получение решения. Для ответа на вопросы задачи анализируем расчётную таблицу и диаграмму. В графе «y(t)» находится ячейка, в которой с ростом времени в первый раз появляется отрицательное число (C35). На диаграмме это точка пересечения траектории полёта с осью «Дальность». По данным строк 34 и 35 расчётной таблицы с точностью до величины шага времени определяем момент падения — 5,3 с и дальность — 80 м.

По данным строки 21 расчётной таблицы (наибольшее значение y) с точностью до величины шага времени определяем наибольшую высоту полёта 34,37 м в момент 2,6 с.

Замечание. Когда координата $y(t)$ становится в ячейке C35 отрицательной, модель становится неадекватной — камень оказывается ниже уровня земли. Если щёлкнуть по траектории на диаграмме, то на блоке расчётной таблицы появляются рамки. Чтобы сократить количество отрицательных значений высоты y на диаграмме, объём исходных данных можно уменьшить. Это делают перетаскиванием вверх-вниз нижних угловых маркеров рамок. Диаграмма автоматически меняет вид.

■ Упражнение 121

С помощью модели полёта тела найдём угол бросания, при котором камень с начальной скоростью 40 м/с упадёт в 100 м от места бросания. Найдём также время полёта.

В разделе исходных данных табличной модели устанавливаем начальную скорость 40 м/с. При необходимости максимально расширяем рамки блока исходных данных для диаграммы. По диаграмме подбором находим искомое значение угла. По таблице устанавливаем значение времени полёта. Заметим, что существует два значения угла и два момента времени, которые решают поставленную задачу.

■ Упражнение 122

С помощью модели полёта тела найдём начальную скорость с углом бросания 60°, при которой камень собьёт неподвижную цель на удалении 100 м и высоте 20 м.

Рис. 123

Воспользуемся построенной моделью, но добавим на диаграмму траектории полёта маркер цели.

Построение маркера цели. Щёлкаем по диаграмме левой клавишей мыши. На вкладке «Конструктор» в группе «Данные» щёлкаем по кнопке **Выбрать данные**. Появляется диалоговое окно «Выбор источника данных».

В левом нижнем поле окна есть только одна запись: «Ряд 1». Она соответствует траектории полёта. Щёлкаем по кнопке **Добавить** и в появившемся окне «Изменение ряда» вводим имя нового ряда «Ряд 2» для построения маркера цели. Ниже в поля «Значения X:» и «Значения Y:» вносим соответственно дальность «={100}» и высоту «{20}». На диаграмме появляется цель. Щелчками по кнопкам **ОК** закрываем оба диалоговых окна.

Получение решения. Подбором находим значение начальной скорости, при котором траектория полёта камня на диаграмме пересечёт маркер цели (рис. 123). Исходные данные могут быть и дробными. При необходимости увеличиваем или уменьшаем блок исходных данных для диаграммы.

Вопросы и задания

1. Для дальности полёта камня в 70 м и начальной скорости 35 м/с найдите угол бросания и время полёта.

2. Для дальности полёта камня в 70 м и угла бросания в 60° найдите начальную скорость и время полёта.

3. Найдите угол бросания, при котором камень, имеющий начальную скорость 40 м/с, собьёт неподвижную цель на удалении в 60 м и высоте 30 м.

4. Найдите начальную скорость с углом бросания 70°, при которой камень собьёт неподвижную цель на удалении 50 м и высоте 60 м.

5*. Найдите начальную скорость и угол бросания, при которых камень собьёт подряд две неподвижные цели: первую — на удалении 50 м и высоте 30 м, вторую — на удалении 100 м и высоте 5 м.

ГЛАВА 6

БАЗЫ ДАННЫХ

§ 46. Введение в базы данных

С информационно-поисковыми системами мы познакомились, приступая к изучению информационных ресурсов Интернета. Напомним, что некомпьютерная информационно-поисковая система — это система однотипных информационных объектов, организованная с целью удобства поиска этих объектов. Информационно-поисковые системы делятся на *документальные* (в результате поиска выдаётся документ) и *справочные* (в результате поиска информация предъявляется или сообщается).

Бурное развитие компьютерных технологий преобразило информационно-поисковые системы. Традиционные справочные ИПС на бумажных носителях (словари, энциклопедии, справочники, каталоги и картотеки) получили компьютерные двойники. Компьютерные ИПС, в отличие от традиционных, обеспечивают автоматический поиск информации. В компьютерных справочниках и энциклопедиях средства поиска встроены. Электронные каталоги и картотеки строятся на иных принципах и включают две составляющие: базу данных и систему управления базами данных.

База данных — это компьютерный интегрированный набор данных, который предназначен для хранения и выдачи данных по запросам.

Система управления базами данных (СУБД) — программное средство, которое обеспечивает создание баз данных и работу с ними.

База данных, как правило, содержит данные о каких-то сходных объектах (работниках, клиентах, заказчиках, поставщиках или товарах, книгах, видах работ и т. п.).

Запись базы данных составляют данные, которые характеризуют один объект. Например, если создавать для конкретной школы базу данных «Учащиеся», то информация об одном учащемся составит одну запись. Таким образом, базы данных состоят из записей.

Каждая запись базы данных условно делится на несколько частей, которые называют полями.

Поле базы данных — это часть записи, которая отводится для отдельной характеристики объекта. Например, в базе данных «Учащиеся» можно выделить поля для таких характеристик, как фамилия ученика; его имя; отчество; класс, в котором он учится; домашний адрес; любимый предмет; данные о родителях.

Виды баз данных. Базы данных различаются принципами организации и форматами хранения. В зависимости от принципов организации различают следующие виды баз данных:

- *сетевые* (типа CODASYL);
- *иерархические* (типа IMS);
- *реляционные*;
- *объектно-ориентированные*.

Сетевые и иерархические базы данных — это исторически первые виды баз данных, которые широко использовались в 1970-е гг. Базы этого вида обрабатывались на больших стационарных компьютерах и в значительной степени были связаны с аппаратным обеспечением. Как сетевые, так и иерархические базы данных часто называют навигационными, так как в те годы автоматического поиска информации в них по сути ещё не было. Программист при помощи специальных команд переходил от одной записи к другой и так просматривал базу способом навигации.

Термин «реляционный» происходит от английского relation — «отношение». Основы теории реляционных баз данных заложил в

Фамилия	Имя	Отчество	Класс	Домашний адрес
Антонов	Владимир	Павлович	9а	ул. Лесная, 6
Берзинь	Ольга	Ивановна	9б	ул. Промышленная, 5-34
Васильева	Вероника	Григорьевна	9а	ул. Севастопольская, 21-45
Герасимович	Марина	Олеговна	9б	ул. Лесная, 8
Кузнецов	Василий	Леонидович	9а	ул. Зелёная, 6
Переверзев	Виктор	Александрович	9б	ул. Котовского, 2-53
Семёнов	Пётр	Афанасьевич	9а	ул. Лесная, 23
Жогло	Зоя	Ивановна	9а	ул. Зелёная, 7
Шыманская	Алла	Емельяновна	9б	ул. Свиридова, 4-23

Рис. 124

1970 г. британский учёный Эдгар Кодд, работавший в фирме IBM. Сегодня в массовой практике преобладают именно реляционные базы данных. В 2002 г. журнал Forbs отметил реляционные базы данных как одну из важнейших инноваций за последние 85 лет.

Объектно-ориентированные базы данных стали интенсивно исследоваться в последние годы, что связано с необходимостью создания сложных информационных систем. Принципы построения объектно-ориентированных баз данных подобны принципам, заложенным в объектно-ориентированных языках программирования.

Формат хранения базы данных определяет СУБД, с помощью которой создаётся база. Здесь полная аналогия с форматами хранения текстовых документов, графических объектов и т. п.

Принципы построения реляционных баз данных. В реляционной базе данные хранятся в таблицах, при этом:

- одна запись базы данных занимает одну строку таблицы;
- каждый столбец таблицы имеет имя, которое является именем соответствующего поля записи;
- записи базы данных являются уникальными (не повторяющимися).

На рисунке 124 в виде таблицы представлено начало базы данных «Учащиеся СШ № 3».

Поле реляционной базы данных (столбец таблицы) кроме имени имеет тип и размер.

Тип поля связан с типом (форматом) представления данных в этом поле. К основным типам полей относят:

- *текстовый,* который предназначен для записи текстовой информации;
- *числовой,* который предназначен для записи чисел.

Существуют и другие типы полей.

Фамилия	Имя	Отчество	Класс	Домашний адрес	Любимый предмет
Антонов	Владимир	Павлович	9а	ул. Лесная, 6	1
Берзинь	Ольга	Ивановна	9б	ул. Промышленная, 5-34	6
Васильева	Вероника	Григорьевна	9а	ул. Севастопольская, 21-45	6
Герасимович	Марина	Олеговна	9б	ул. Лесная, 8	4
Кузнецов	Василий	Леонидович	9а	ул. Зелёная, 6	3
Переверзев	Виктор	Александрович	9б	ул. Котовского, 2-53	5
Семёнов	Пётр	Афанасьевич	9а	ул. Лесная, 23	8
Жогло	Зоя	Ивановна	9а	ул. Зелёная, 7	1
Шыманская	Алла	Емельяновна	9б	ул. Свиридова, 4-23	2

Код	Предмет
1	Математика
2	Физика
3	Химия
4	Биология
5	География
6	Информатика и ИКТ
7	Рус. язык
8	Литература

Рис. 125

Размер поля — это максимальное количество символов, которые могут быть записаны в поле.

Структура базы данных — это структура столбцов таблицы базы данных.

Отметим, что реляционные базы данных могут включать несколько таблиц разной структуры. Между таблицами специальным образом можно установить отношения (рис. 125). Именно поэтому базы и называются реляционными. Мы же будем рассматривать реляционные базы данных, построенные на основе одной таблицы.

Основные функции СУБД. СУБД обеспечивают следующие основные функции:

- создание структуры базы данных;
- пополнение, просмотр и редактирование данных в базе;
- формирование запросов к базе данных;
- автоматическую обработку и выдачу данных базы по запросам.

В реляционных СУБД существуют два режима просмотра баз данных: ***режим таблицы*** и ***режим формы***. На рисунке 124 база данных «Учащиеся СШ № 3» представлена именно в режиме таблицы. В режиме формы на экране отображаются данные только

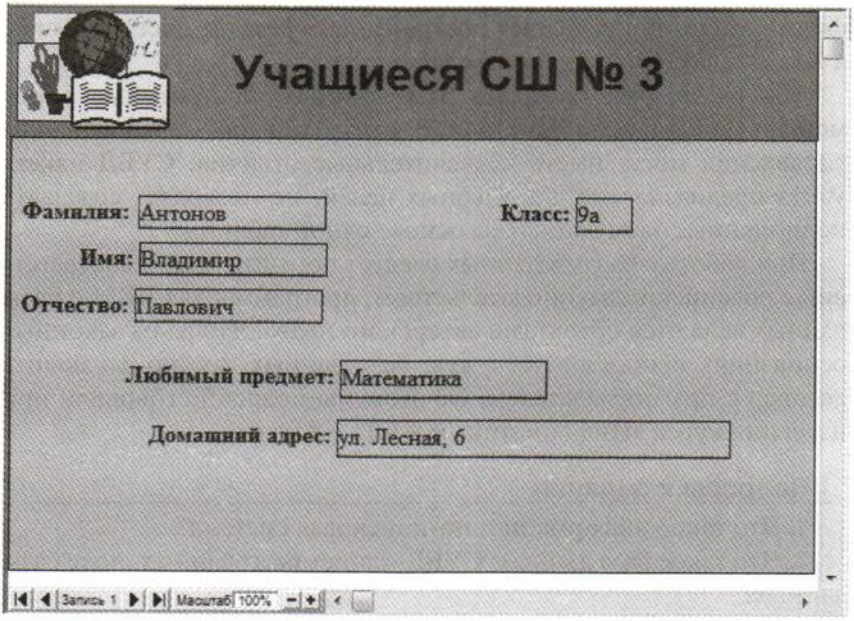


Рис. 126

одной записи базы данных. Для перехода от одной записи к другой предназначена панель навигации с соответствующими кнопками. На рисунке 126 база данных «Учащиеся СШ № 3» представлена в режиме формы. В режиме формы базу данных легче пополнять и редактировать. Панель навигации расположена на форме внизу слева.

Запросы к базе данных. При работе с базами данных чаще всего используются запросы трёх видов:

- запрос на выдачу некоторых данных в форме электронной таблицы;
- запрос на выдачу некоторых данных в форме документа;
- запрос на сортировку (упорядочение) записей базы данных.

Долгое время запросы к базам данных формировались командами специальных языков программирования. Современные СУБД позволяют формировать такие запросы с помощью меню.

На рынке больших профессиональных баз данных сейчас доминируют фирмы-разработчики IBM, Microsoft и Oracle. Для работы с реляционными базами данных на персональных компьютерах

фирма Microsoft предлагает программные средства Access и Works, а также редактор электронных таблиц Excel.

Мы будем рассматривать базы данных, построенные с помощью СУБД пакета Works 9.0. В других версиях СУБД команды управления могут иметь незначительные отличия. СУБД пакета Works предназначена для учебных целей, но позволяет создавать полноценные базы данных на основе одной таблицы.

При работе с базами данных следует помнить, что, в соответствии с российским законодательством, программы для ЭВМ и базы данных являются объектами авторского права. Чтобы на законных основаниях использовать чужую базу данных, нужно заключить договор с правообладателем. Об этом мы подробно говорили при изучении курса «Информатика и ИКТ» в 8 классе.

☐ Вопросы и задания

1. Что такое информационно-поисковая система?

2. Что такое база данных; СУБД; запись базы данных; поле базы данных?

3. Перечислите виды баз данных.

4. Изложите принципы построения реляционных баз данных.

5. Что такое тип поля реляционной базы данных?

6. Что такое размер поля реляционной базы данных?

7. Что называется структурой базы данных?

8. Перечислите основные функции реляционных СУБД.

9. Опишите основные особенности режимов просмотра реляционных баз данных с помощью СУБД.

10. Перечислите виды запросов к базам данных.

11. С помощью информационных ресурсов Интернета расшифруйте аббревиатуру CODASYL.

12. С помощью информационных ресурсов Интернета составьте реферат на одну из тем:

- «История создания СУБД за рубежами России (СССР)»;
- «История создания СУБД в России (СССР)».

§ 47. Знакомство с СУБД пакета Works

Запуск и общий вид экрана. Пакет Works включает в свой состав несколько программных средств. Основными составляющими пакета являются:

• текстовый процессор Works (текстовый редактор);

• электронная таблица Works (редактор электронных таблиц);

• база данных Works (СУБД).

Приведённые выше названия программных средств введены в пакете Works. В скобках приведены названия, которые используются в данном учебнике.

Рис. 127

С помощью ярлыка на рабочем столе или каскадного меню кнопки **Пуск** запускают программу «Средство запуска задач Microsoft Works», а затем с помощью меню — программу «База данных Works».

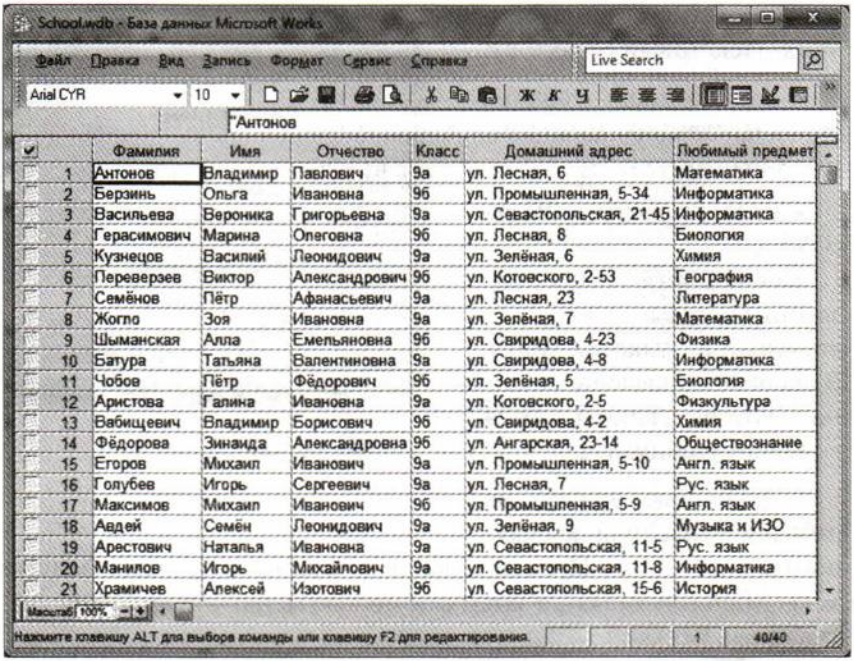
Ярлык «База данных Works» также может располагаться на рабочем столе независимо от пакета Works.

На экране появляется диалоговое окно с запросом (рис. 127). Отмечаем пункт «Открыть базу данных» и щёлкаем по кнопке **OK**.

Появляется стандартное окно открытия файла, в котором на прилагаемом CD-диске находим файл School.wbd и загружаем его.

Открывается окно СУБД (рис. 128). Оно имеет привычную для ОС Windows структуру. Панель инструментов под строкой меню выводится или убирается командой меню **Вид|Панель инструментов|Показать панель**. Оставим панель на экране. Ниже панели инструментов расположена ***строка ввода***, которая очень похожа на строку формул в редакторе электронных таблиц.

Учебная база данных «Учащиеся СШ № 3» представлена в ***режиме «Список»***. Именно так называется режим таблицы в СУБД Works. Имеется в виду, что таблица — это список записей-строк. Учебная база содержит 40 записей с данными учеников двух классов 9 «А» и 9 «Б».



	Фамилия	Имя	Отчество	Класс	Домашний адрес	Любимый предмет
1	Антонов	Владимир	Павлович	9а	ул. Лесная, 6	Математика
2	Берзинь	Ольга	Ивановна	9б	ул. Промышленная, 5-34	Информатика
3	Васильева	Вероника	Григорьевна	9а	ул. Севастопольская, 21-45	Информатика
4	Герасимович	Марина	Олеговна	9б	ул. Лесная, 8	Биология
5	Кузнецов	Василий	Леонидович	9а	ул. Зелёная, 6	Химия
6	Переверзев	Виктор	Александрович	9б	ул. Котовского, 2-53	География
7	Семёнов	Пётр	Афанасьевич	9а	ул. Лесная, 23	Литература
8	Жогло	Зоя	Ивановна	9а	ул. Зелёная, 7	Математика
9	Шыманская	Алла	Емельяновна	9б	ул. Свиридова, 4-23	Физика
10	Батура	Татьяна	Валентиновна	9а	ул. Свиридова, 4-8	Информатика
11	Чобов	Пётр	Фёдорович	9б	ул. Зелёная, 5	Биология
12	Аристова	Галина	Ивановна	9а	ул. Котовского, 2-5	Физкультура
13	Вабищевич	Владимир	Борисович	9б	ул. Свиридова, 4-2	Химия
14	Фёдорова	Зинаида	Александровна	9б	ул. Ангарская, 23-14	Обществознание
15	Егоров	Михаил	Иванович	9а	ул. Промышленная, 5-10	Англ. язык
16	Голубев	Игорь	Сергеевич	9а	ул. Лесная, 7	Рус. язык
17	Максимов	Михаил	Иванович	9б	ул. Промышленная, 5-9	Англ. язык
18	Авдей	Семён	Леонидович	9а	ул. Зелёная, 9	Музыка и ИЗО
19	Арестович	Наталья	Ивановна	9а	ул. Севастопольская, 11-5	Рус. язык
20	Манилов	Игорь	Михайлович	9а	ул. Севастопольская, 11-8	Информатика
21	Храмичев	Алексей	Изотович	9б	ул. Севастопольская, 15-6	История

Рис. 128

Для работы с базой данных её следует сохранить в рабочей папке на жёстком диске, вызвав окно сохранения файла командой меню **Файл|Сохранить как...** .

Режим «Список». В режиме списка база данных представляет собой электронную таблицу, у которой все ячейки каждого столбца (каждого поля) имеют одинаковый формат.

■ Упражнение 123

Познакомимся с параметрами форматирования полей базы данных.

Установим табличный курсор на поле «Фамилия» первой записи и введём команду меню **Формат|Поле...** .

На экран выводится диалоговое окно с закладками, в которых можно изменить имя поля, его формат, вид выравнивания, шрифт, задать рамку или вид тонировки (заливки).

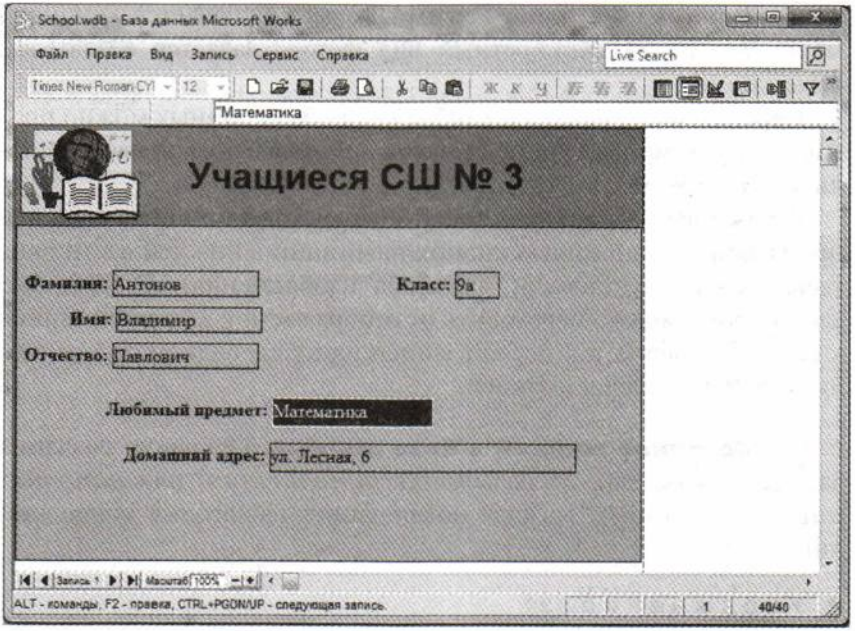


Рис. 129

Заметим, что в СУБД Works существует формат поля «Общий», при котором отображаются как тексты (с выравниванием влево), так и числа (с выравниванием вправо).

В СУБД Works по аналогии с электронными таблицами Excel размер полей (ячеек таблицы) для содержимого практически не ограничен. Но на экране ячейки имеют ограниченную ширину. Чтобы, например, тексты в ячейках отображались на экране полностью, можно увеличить ширину столбцов таблицы. Приёмы при этом используются те же, что и для электронных таблиц в Excel.

Масштаб изображения таблицы устанавливается с помощью соответствующих кнопок в нижней части окна (слева в строке горизонтальной полосы прокрутки, рис. 129). В правой части строки состояния окна указаны число 1 и дробь 40/40. Число 1 указывает на номер активной записи (активную запись можно редактировать). Числитель дроби указывает количество записей, отображаемых в окне программы, знаменатель — общее количество записей в базе.

Режим «Форма». Режим «Форма» включается командой меню **Вид|Форма**. Конкретный вид формы задаётся создателем базы данных.

Переключение режимов отображения базы данных можно производить с помощью кнопок **Список** и **Форма** в правой части панели инструментов.

В режиме «Форма» (см. рис. 126) переход от записи к записи производится с помощью кнопок навигации в нижней части окна (левее кнопок установки масштаба изображения). С помощью этих кнопок можно переходить от отображаемой активной записи к соседней, перейти к первой записи или к свободной строке таблицы после последней записи.

Создание записей в базе данных. Операция создания записей в базе данных называется *пополнением* или *наполнением* базы данных. Каждая новая запись вводится в конец таблицы.

■ Упражнение 124

Введём в учебную базу (см. рис. 128) новую запись следующего содержания:

Феоктистов; Юрий; Сергеевич; 9а; ул. Севастопольская, 9-3; Рус. язык

В приведённой записи поля разделены знаком «точка с запятой».

Операцию пополнения базы данных будем проводить в режиме «Форма».

С помощью кнопок навигации переходим в конец базы (на пустую строку таблицы). Указатель активной записи показывает 41. Вводим содержимое в соответствующие поля. После пополнения сохраняем базу данных.

Редактирование записей в базе данных. Редактирование (изменение) записей в базе данных можно проводить как в режиме «Список», так и в режиме «Форма».

■ Упражнение 125

В записи с номером 9 учебной базы (см. рис. 128) изменим в домашнем адресе номер квартиры на 11.

Переходим в режим «Список». Устанавливаем табличный курсор в строке 9 на поле «Домашний адрес». Редактирование производим в строке ввода. Редактирование завершается нажатием клавиши **Enter** клавиатуры. После редактирования базу данных сохраняем.

Запросы к базе данных Works. СУБД Works поддерживает все известные виды запросов к базам данных. Запросы формируются в специальных диалоговых окнах.

Запросы на выдачу некоторых данных в форме электронной таблицы в СУБД Works называются ***фильтрами***. Фильтры в базе данных хранятся как отдельные объекты со своими именами. Применение фильтра к базе данных в режиме «Список» оставляет на экране только те записи, которые удовлетворяют условиям фильтра.

Запросы на выдачу некоторых данных в форме документа в СУБД Works называются ***отчётами***. Формы отчётов хранятся в базе данных как отдельные объекты со своими именами. Просмотреть отчёты можно в режиме просмотра отчётов СУБД Works.

Запросы на сортировку (упорядочение) записей базы данных в СУБД Works отдельными объектами не являются. Каждый новый такой запрос заново создаётся и применяется. Сортировка меняет порядок записей в базе данных.

☐ Вопросы и задания

1. Перечислите основные составляющие пакета Works.

2. Какая команда меню выводит или убирает с экрана СУБД Works панель инструментов?

3. Какие действия позволяют увидеть параметры форматирования конкретного поля базы данных?

4. Какими действиями можно изменить ширину столбцов базы данных в режиме «Список»?

5. Что означают числа в правой части строки состояния окна СУБД Works?

6. Как можно пополнить базу данных новыми записями?

7. Пополните учебную базу данных «Учащиеся СШ № 3» ещё тремя записями с данными учеников вашего класса.

8. Как редактируют записи базы данных?

9. Что такое фильтр в СУБД Works?

10. Что такое отчёт в СУБД Works?

§ 48. Поиск и сортировка данных в базе

Поиск данных в базе Works поводится с помощью фильтров. Знакомиться с методами построения фильтров в СУБД Works мы будем, выполняя конкретные задания на поиск данных в учебной базе «Учащиеся СШ № 3».

■ Упражнение 126

Выведем в табличной форме все записи базы данных, в которых учащиеся имеют имя Пётр.

Пользуемся режимом «Список». Вводим команду меню **Сервис|Фильтры**. Открывается диалоговое окно «Имя фильтра». СУБД по умолчанию подсказывает имя фильтра «Фильтр 1». Оставляем это имя и щёлкаем по кнопке **ОК**. Окно закрывается. Открывается диалоговое окно «Фильтр». Мы будем работать только с простыми фильтрами.

В разделе «Определение фильтра» надо задать условия, которым удовлетворяют требуемые записи. Для нашего задания содержимое поля «Имя» должно равняться «Пётр».

В столбце «Имя поля» в первой строке с помощью кнопки дополнительного меню выбираем поле «Имя», оставляем сравнение «равно» и в последнем окошке записываем «Пётр» (рис. 130). Щёл-

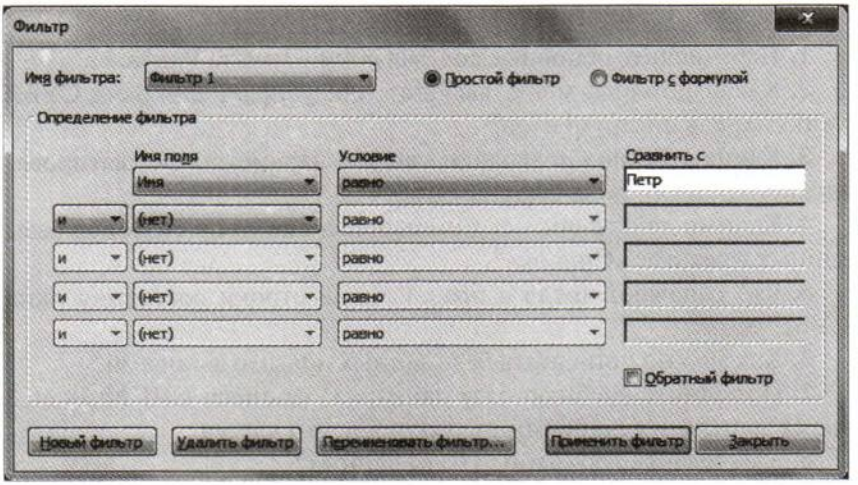


Рис. 130

каем по кнопке **Применить фильтр**. Фильтр сохраняется, и в таблице остаются записи с именем Пётр.

Заметим, что в правой части строки состояния окна СУБД числитель дроби покажет количество отображаемых записей.

Вернуть отображение полного списка записей можно командой меню **Запись|Показать|1 Все записи**. Чтобы снова применить фильтр с именем «Фильтр 1», вводят команду меню **Запись|Применить фильтр|Фильтр 1.**

■ Упражнение 127

Выведем в табличной форме все записи, в которых любимым предметом учащихся является физика.

Вводим команду меню **Сервис|Фильтры...** . Открывается диалоговое окно «Фильтр» с параметрами предыдущего фильтра. Щёлкаем по кнопке **Новый фильтр**. Открывается диалоговое окно для задания имени «Фильтр 2» нового фильтра. Щёлкаем по кнопке **ОК**. Остаётся окно «Фильтр».

В разделе «Определение фильтра» зададим условия. Для нашего задания содержимое поля «Любимый предмет» должно равняться «Физика». Применение фильтра оставляет в таблице требуемые записи.

■ Упражнение 128

Выведем в табличной форме все записи, в которых любимым предметом учащихся 9 «Б» класса является химия.

Формируем фильтр с именем «Фильтр 3». Условий должно быть два. Первое — содержимое поля «Класс» должно равняться «9б». Второе — содержимое поля «Любимый предмет» должно равняться «Химия». Вводим эти данные в две строки диалогового окна «Фильтр». Применяем фильтр. Результат выводится на экран.

■ Упражнение 129

Выведем в табличной форме все записи, в которых ученика зовут Игорь и ему нравится математика.

По аналогии с предыдущим заданием формируем фильтр с именем «Фильтр 4». Должны выполняться два условия: содержимое поля «Имя» должно равняться «Игорь», а содержимое поля «Любимый предмет» должно равняться «Математика». Применяем

фильтр. Появляется диалоговое окно с предупреждением о том, что таких записей нет. Нужно, конечно, проверить правильность условий фильтра, так как любая ошибка, включая пропуск буквы, может привести к такому результату. Однако в данном случае в базе требуемой записи действительно нет. Сохраним фильтр с именем «Фильтр 4» без изменений.

■ Упражнение 130

Выведем в табличной форме все записи об учениках 9 «А» класса, фамилии которых начинаются на букву «В».

Формируем фильтр с именем «Фильтр 5». Первое условие фильтра трудностей не вызывает: содержимое поля «Класс» должно равняться «9а».

Для задания второго условия нужно познакомиться со всеми предлагаемыми видами сравнения. Среди сравнений есть «содержит», «начинается с», «оканчивается». Нас устроит условие: содержимое поля «Фамилия» начинается с «В».

В полях числового типа часто применяются сравнения вида «больше чем», «меньше или равно» и т. п.

Сортировка записей базы данных производится по возрастанию значений поля или по их убыванию. Если для полей числового типа принцип сортировки понятен, то для полей текстового типа необходимы пояснения.

В полях текстового типа сортировка происходит по возрастанию или убыванию цифровых кодов символов в соответствии с применяемой кодировочной таблицей. В кодировочной таблице Windows символы по возрастанию цифровых кодов выстраиваются в следующем порядке: 0 1 2 3 4 5 6 7 8 9 (пробел) ! " # $ % &( ) * , . / : ; ^ _ ' { | } ~ + < > A B C D E F G H J K L M N O P Q R S T U V W X Y Z 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.

Это означает, что сортировка полей текстового типа по возрастанию совпадает с упорядочением в алфавитном порядке, а сортировка по убыванию — с порядком, обратным алфавитному.

■ Упражнение 131

Установим в базе данных «Учащиеся СШ № 3» алфавитный порядок записей по фамилиям учащихся.

Рис. 131

Прежде всего выводим на экран в режиме «Список» все записи (команда меню **Запись|Показать|1 Все записи**). Вводим команду меню **Запись|Сортировка записей...** . Появляется диалоговое окно «Сортировка записей» (рис. 131). В поле «Сортировать по» выбираем имя поля «Фамилия» и оставляем выбор записей по возрастанию. Щелчок по кнопке **OK** преобразует базу данных.

Заметим, что база данных действительно преобразуется, а её записи получают новые порядковые номера. Результаты сортировки можно отменить только один раз в меню пункта **Правка**.

■ Упражнение 132

Установим в базе данных такой порядок записей, при котором сначала в алфавитном порядке идут фамилии учащихся 9 «А» класса, а затем учащихся 9 «Б».

Вводим команду меню **Запись|Сортировка записей...** . Первым вводим условие сортировки содержимого поля «Класс» по возрастанию (сначала отобразятся все записи 9 «А», а потом 9 «Б»). Затем при равных значениях поля «Класс» (а таких значений будет очень много) будем сортировать содержимое поля «Фамилия» также по возрастанию. Щелчок по кнопке **OK** выводит на экран требуемый результат.

☐ Вопросы и задания

1. Какая команда меню приводит к открытию диалогового окна для создания фильтра базы данных?

2. Какая команда меню в режиме «Список» возвращает на экран все записи базы данных после фильтрации?

3. Какой командой можно повторно применить созданный ранее фильтр?

4. В учебной базе «Учащиеся СШ № 3» выведите на экран записи о тех учащихся, которые:

1) проживают на ул. Зелёной;

2) имеют имена, начинающиеся с буквы «В»;

3) учатся в 9 «А» классе и которым нравится физика.

5. По данным учебной базы «Учащиеся СШ № 3» ответьте на вопрос: «Проживают ли на ул. Промышленной учащиеся, которым нравится математика?»

6*. В учебной базе «Учащиеся СШ № 3» выведите на экран данные учащихся 9 «Б» класса, которым нравятся биология, химия или география.

7. Почему для полей текстового типа сортировка по возрастанию совпадает с упорядочением в алфавитном порядке?

8. В учебной базе «Учащиеся СШ № 3» выведите на экран записи:

1) в алфавитном порядке имён учащихся;

2) сгруппированные по месту проживания;

3) сгруппированные по любимым предметам и в алфавитном порядке, причём сначала для учащихся 9 «А» класса, а потом 9 «Б».

§ 49. Отчёт базы данных

Отчёт базы данных Works представляет собой электронный документ, который при необходимости можно распечатать. При создании отчётов могут использоваться фильтры для отбора данных из базы и условия сортировки для упорядочения выводимой информации.

■ Упражнение 133

Создадим и просмотрим отчёт базы данных «Учащиеся СШ № 3», который содержит заголовок «Учащиеся 9 классов СШ № 3» и далее список всех учащихся в алфавитном порядке.

Вводим команду меню **Сервис|Генератор отчётов**. Появляется диалоговое окно с запросом имени нового отчёта, пусть это будет имя «Отчёт 1». После ввода имени и щелчка по кнопке **ОК** появляется диалоговое окно «Генератор отчётов» с множеством вкладок. Переходим к созданию формы отчёта.

1. На вкладке «Заголовок» вводим требуемый заголовок отчёта. Оставляем книжную ориентацию отчёта и шрифт по умолчанию.
2. На вкладке «Поля» отбираем и добавляем в правую зону поля для отображения: сначала «Фамилия», а затем «Имя». Флажки в нижних полях для выбора должны отсутствовать.
3. На вкладке «Сортировка» выбираем поле «Фамилия» и тип сортировки по возрастанию.
4. Вкладку «Группировка» пропускаем.
5. На вкладке «Фильтр» выбираем пункт **(Все записи)**.
6. На вкладке «Итоги» убираем все возможные флажки в полях выбора. Щёлкаем по кнопке **Готово**.

СУБД переходит в ***режим «Отчёт»***, на экране появляется табличная форма (рис. 132) и диалоговое окно с предложением просмотреть отчёт или изменить его. Выбираем просмотр.

СУБД переходит в ***режим предварительного просмотра отчёта***. Убеждаемся в правильности его составления и щелчком по кнопке **Закрыть просмотр** возвращаемся в режим «Отчёт».

В режим «Список» или режим «Форма» можно перейти уже известными нам способами. В режим «Отчёт» переходят командой ме-

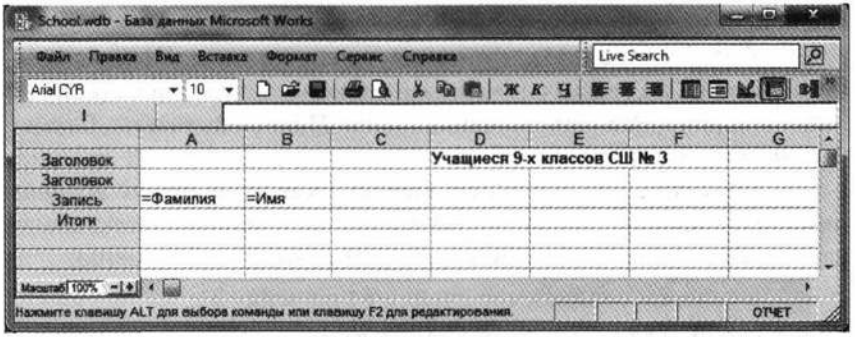


Рис. 132

ню **Вид|Отчёт**. Появляется диалоговое окно выбора отчёта по имени. В том же диалоговом окне можно перейти к просмотру отчёта или отказаться от просмотра.

■ Упражнение 134

Создадим и просмотрим отчёт базы данных «Учащиеся СШ № 3», который состоит из страниц, на каждой из которых содержится заголовок с названием класса и список учащихся в алфавитном порядке.

Вводим команду меню **Сервис|Генератор отчётов...** . Даём отчёту имя «Отчёт 2». Переходим к созданию формы отчёта.

1. На вкладке «Заголовок» удаляем заголовок-подсказку. Оставляем книжную ориентацию отчёта и шрифт по умолчанию.
2. На вкладке «Поля» отбираем и добавляем в правую зону поля для отображения: сначала «Фамилия», затем «Имя». Флажки в нижних полях для выбора должны отсутствовать.
3. На вкладке «Сортировка» выбираем сначала поле «Класс» и сортировку по возрастанию, а затем поле «Фамилия» и сортировку также по возрастанию.
4. На вкладке «Группировка» в разделе «Группировать по: Класс» ставим флажки во всех полях, за исключением поля «Только по первой букве».
5. На вкладке «Фильтр» выбираем пункт **(Все записи)**.
6. На вкладке «Итоги» убираем все возможные флажки в полях выбора. Щёлкаем по кнопке **Готово**.

Просматриваем отчёт. Он действительно включает страницы со списками. Щёлкаем по кнопке **Закрыть просмотр** и возвращаемся в режим «Отчёт» (рис. 133), чтобы внести коррективы. Перетаскиванием курсора по серым служебным ячейкам первых двух строк выделяем строки. Вводим команду меню **Правка|Вырезать**. Строки удаляются.

Теперь перед обозначением каждого класса в отчёте вставим имя поля «Класс». Для этого, пользуясь выделением ячеек, щелчками правой клавиши мыши и выбором в контекстном меню:

1. перенесём содержимое ячейки A1 в ячейку B1;
2. в ячейку A1 введём имя поля «Класс»;
3. для ячейки A1 зададим выравнивание «По центру».

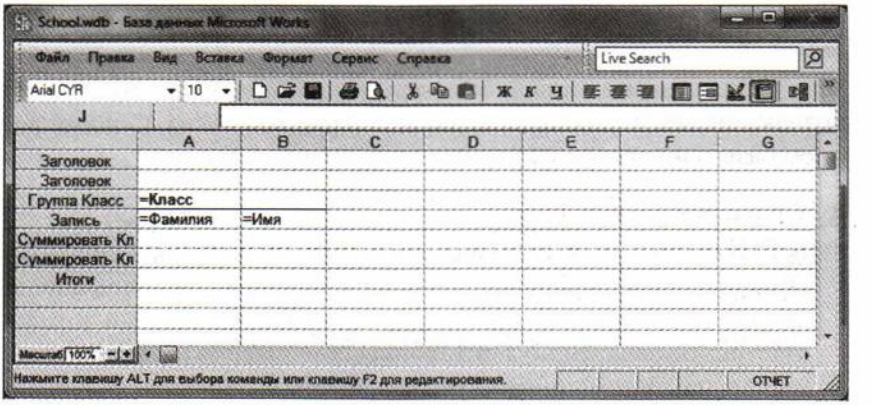


Рис. 133

Чтобы вывести в отчёте правее имён учащихся таблицу (как в классном журнале), выделяем блок на пересечении строки «Запись» и столбцов C-F. Вводим команду меню **Формат|Границы**. В появившемся окне щёлкаем по кнопкам **внешние**, **внутренние** и **ОК**.

Командой меню **Файл|Предварительный просмотр** включаем режим просмотра отчёта и убеждаемся, что отчёт принял вид страниц из классного журнала.

Вопросы и задания

1. Какая команда меню приводит к открытию диалогового окна для создания отчёта базы данных?

2. Сколько вкладок имеет диалоговое окно «Генератор отчётов»?

3. Чем режим «Отчёт» в СУБД Works отличается от режима просмотра отчёта?

4. Какая команда меню позволяет в СУБД Works перейти в режим «Отчёт»?

5. Как в СУБД Works перейти в режим просмотра отчёта?

6. Создайте по данным учебной базы «Учащиеся СШ № 3» отчёт со списками учащихся по группам любителей предметов. Каждая группа должна иметь соответствующее название, а фамилии учащихся должны располагаться в алфавитном порядке и сопровождаться обозначением класса.

§ 50. Создание базы данных

Для создания новой базы данных в СУБД Works достаточно познакомиться с порядком создания структуры базы. С пополнением базы данных записями мы уже знакомы.

■ Упражнение 135

В ГИБДД имеется следующая информация по автомобилям, которые принадлежат частным лицам: 1) модель автомобиля; 2) год выпуска; 3) государственный регистрационный номер; 4) номер кузова; 5) номер технического паспорта; 6) фамилия владельца; 7) домашний адрес владельца.

Создадим базу данных для учёта автомобилей, принадлежащих частным владельцам.

В СУБД Works вводим команду меню **Файл|Создать**. Появляется окно программы «Средство запуска задач Microsoft Works», в котором для создания предлагаются новая база данных или несколько групп шаблонов баз данных. Выбираем для создания новую базу данных. Появляется диалоговое окно «Создать базу данных». Вводим имя первого поля базы, выбираем его тип и щёлкаем по кнопке **Добавить**. На экране появляется первый столбец таблицы. Вводим данные по второму полю и т. д. После ввода имён и типов всех полей щёлкаем по кнопке **Готово**. В результате создаётся таблица для ввода данных.

Так как пополнять базу данных удобнее в режиме «Форма», переходим в этот режим (команда меню **Вид|Форма**). СУБД создаёт форму автоматически и одновременно с созданием таблицы. Но форма может потребовать доработки.

Доработка формы для базы данных проводится в ***режиме «Макет формы»*** (команда меню **Вид|Макет формы** или щелчок по соответствующей кнопке на панели инструментов). В этом режиме названия полей и сами поля предстают как объекты векторного графического редактора. Поэтому их можно перетаскивать по форме, изменять их размеры и шрифт надписей.

С помощью меню пункта **Вставка** можно вывести на экран прямоугольники, надписи (метки) и картинки. С помощью меню пункта **Формат** у выделенных объектов можно менять цвет заливки, цвет и стиль их границ. В общем, пользователю предоставляется полная свобода для дизайнерских решений.

Так как графические объекты могут налегать друг на друга, то для изменения порядка слоёв пользуются выпадающим меню пункта **Формат**. Именно там есть пункты **На передний план** и **На задний план**, хорошо знакомые нам по работе с векторным графическим редактором, встроенным в текстовый редактор Word.

Шаблоны баз данных. В СУБД Works имеется несколько пустых баз данных, которые названы ***шаблонами***. Шаблоны имеют дизайнерски продуманные формы и полностью готовы к пополнению. Среди шаблонов, собранных в группы, есть шаблоны для записей домашней библиотеки, разного рода коллекций, фонотек, списков членов клуба, адресов и т. д. При необходимости шаблоны можно изменить, подстроить под практические потребности пользователя.

Ну а если кто-либо создаст новую базу данных с возможностью широкого применения, то при сохранении базы данных в СУБД Works можно сохранить и шаблон базы.

☐ Вопросы и задания

1. Какие действия переводят СУБД Works в режим создания структуры новой базы данных?

2. Для чего служит режим «Макет формы»?

3. Что такое шаблон базы данных в СУБД Works?

4. По данным таблицы 10 (см. § 39) создайте базу данных «Солнечная система».

5. Используя базу данных «Солнечная система», найдите ответы на следующие вопросы.

1) Сколько спутников имеет планета Сатурн?

2) Какие планеты имеют более пяти спутников?

3) Какие планеты имеют расстояние до Солнца менее 2 а. е. (а. е. — астрономическая единица)?

4) Какие планеты имеют плотность от 2 до 5 г/см^3?

5) Какие планеты имеют диаметр от 10 до 50 км?

6. Составьте отчёт базы данных «Солнечная система» с информацией о планетах, которые имеют среднее расстояние до Солнца более 9 а. е. и более 15 спутников (название планеты, расстояние до Солнца, количество спутников).

7. Какие поля имеет электронный каталог Российской государственной библиотеки в Интернете?

ГЛАВА 7

МУЛЬТИМЕДИЙНЫЕ ТЕХНОЛОГИИ

§ 51. Понятие о мультимедиа и мультимедийных технологиях

Слово «мультимедиа», копия английского multimedia, составлено из слов multi (много) и media (способы, средства). В английском языке слово multimedia является прилагательным и имеет значение «использующий различные средства информации».

В середине 1960-х гг. термин «мультимедиа» был впервые использован в США при описании популярного театрализованного представления под названием The Exploding Plastic Inevitable («Неизбежность взрывающегося пластика»), которое сочетало живую рок-музыку, кино, световые эффекты, танцы и театральное действо. В конце 1970-х гг. этот термин использовался для описания слайд-шоу на базе нескольких проекторов в сопровождении аудиозаписи.

Современный смысл понятия «мультимедиа» сложился в конце 1990-х гг. Понятие «мультимедиа» теперь связывает разные формы представления информации. (Напомним, что с формами представления информации мы подробно знакомились в самом начале 8 класса.)

Мультимедиа, мультимедийный — динамический, использующий визуальную и звуковую формы представления информации в компьютере или телевизоре.

Чтобы уменьшить разнообразие терминологии, далее в учебнике мы будем использовать термин «мультимедийный». Например, из трёх равнозначных понятий: «мультимедийный объект», «мультимедиа-объект» и «объект мультимедиа» мы будем использовать понятие «мультимедийный объект».

Мультимедийный объект — это информационный объект, воспроизведение или проигрывание которого является мультимедийным.

Мультимедийными объектами являются видеофильмы, видеоклипы, мультфильмы и клипы с другими видами анимации, караоке, цифровые аудиозаписи (песни, речь, звуковые сигналы, звуки природы), наборы цифровых графических объектов для слайд-шоу, электронные презентации, компьютерные игры и т. д.

На первый взгляд аудиозаписи не используют при воспроизведении визуальную форму представления информации. Но здесь надо заметить, что современные программы воспроизведения добавляют к звуку динамические зрительные образы, которые непредсказуемо переливаются разными цветами на экране компьютера (рис. 134). С другой стороны, зрительную форму слайд-шоу всегда можно дополнить звуковым музыкальным сопровождением, а электронные презентации (аналог слайд-шоу) позволяют включать звуковые фрагменты в свой состав на равных правах с визуальными образами.

Мультимедийными называют также информационные объекты, которые являются мультимедийными только частично. Напри-

Рис. 134

мер, мультимедийные энциклопедии или мультимедийные обучающие программы включают видео- и аудиофрагменты (клипы), но не состоят из них полностью. В таких случаях говорят о *мультимедийном содержимом* или о *мультимедийном контенте* информационных объектов.

Мультимедийный поток — это поток мультимедийной информации, получаемый по каналам Интернета или телевизионным каналам.

В Интернете мультимедийные потоки предоставляются пользователям радиостанциями и телевизионными станциями интернет-вещания. Телевизионные эфирные каналы наполнены в основном мультимедийными потоками (изображение и звук). В то же время передача телетекста по телевизионному каналу мультимедийным потоком не считается.

Мультимедийные объекты и потоки бывают аналоговыми и цифровыми. *Аналоговые мультимедийные объекты и потоки* связаны с аналоговым (обычным) телевидением. *Цифровые мультимедийные объекты и потоки* связаны с компьютерами, компьютерными сетями, цифровым телевидением, а также с телефонными сотовыми сетями, в которых есть возможность передачи MMS-сообщений (от англ. Multimedia Message Service).

Далее мы сосредоточимся на изучении цифровых компьютерных мультимедийных объектов. Такие объекты хранятся в виде файлов, которые также называются мультимедийными.

Мультимедийные объекты бывают линейными и нелинейными.

Линейный мультимедийный объект автоматически воспроизводится подобно демонстрации кинофильма (от начала до конца) без управления со стороны пользователя.

Речь в первую очередь идёт о цифровых видео- и аудиозаписях, караоке, слайд-шоу. Заметим, что цифровые мультимедийные потоки бывают только линейными.

Для воспроизведения на компьютере линейных мультимедийных объектов и потоков используют специальные программы, которые называются *проигрывателями* или *плеерами*. Плееры имеют на экране хорошо узнаваемый набор управляющих кнопок (см. кнопки на рис. 134).

Нелинейный мультимедийный объект называется *интерактивным* и воспроизводится (проигрывается) только под управлением пользователя.

Самым ярким примером нелинейных мультимедийных объектов являются компьютерные игры, ход которых предугадать невозможно.

К этому же типу относятся адаптивные (приспосабливающиеся) мультимедийные обучающие программы. Электронную презентацию также можно создать в нелинейной форме (с необходимостью выбора порядка показа слайдов).

Мультимедийный компьютер — это компьютер, в котором аппаратные средства и программное обеспечение могут обеспечить показ видеофильма.

Видеофильм при этом в некотором смысле является эталонным образцом насыщенного линейного мультимедийного объекта.

Современный мультимедийный компьютер должен иметь возможность подключения звуковых колонок или стереонаушников и микрофона, иметь в своём составе DVD-дисковод с возможностью записи, обладать достаточным быстродействием и оперативной памятью.

Игровой компьютер — это мультимедийный компьютер с повышенными характеристиками обработки видеоинформации.

Современные компьютерные игры наполнены виртуальными объектами, созданными с использованием 3D-графики. Любое изменение обстановки в ходе 3D-игры требует множества перерасчётов для придания естественного вида и положения виртуальным объектам на экране компьютера.

Источники цифровых мультимедийных объектов и потоков — их иногда называют *медиа* — весьма разнообразны: цифровые фотоаппараты и видеокамеры, телевизионные сигналы, лазерные диски. Но самым обширным источником мультимедийных объектов и потоков являются информационные ресурсы Интернета.

Отображать на экране компьютера мультимедийный поток телевизионного (ТВ) сигнала помогают специальные устройства — *компьютерные ТВ-тюнеры*. ТВ-тюнер (внешний или встроен-

ный) подключается к компьютеру, а к тюнеру подключается обычный кабель телевизионной антенны или кабельной телевизионной сети. ТВ-тюнеры позволяют не только просматривать телевизионные передачи на экране компьютера, но и проводить оцифровку телевизионного сигнала.

Оцифровка аналогового сигнала — это дискретизация аналогового сигнала и его представление в цифровой компьютерной форме.

С этими понятиями мы также знакомились в 8 классе.

О дисках CD и DVD подробно говорить нет необходимости, поскольку они вошли в повседневную реальность. На дисках представлены видеофильмы, мультипликации, музыка, компьютерные игры, мультимедийные энциклопедии, мультимедийные обучающие курсы и разнообразное программное обеспечение, включая программы для работы с мультимедийными объектами.

В Интернете существует большое количество сайтов с мультимедийным контентом (содержимым). С таких сайтов можно скачивать (при выполнении требований правообладателя) файлы с музыкой, видеофильмами, мультфильмами, чтобы потом воспроизвести их на своём компьютере.

Некоторые мультимедийные объекты на сайтах Интернета можно воспроизвести дистанционно. При этом с сайта по сети передаётся мультимедийный поток, который принимается компьютером и воспроизводится. Именно так работают в Интернете радио- и телевизионные станции.

Информационный поток из Интернета сначала проходит *буферизацию* (накопление в буфере) в памяти компьютера. Когда в буфере накопится достаточно информации для воспроизведения, она воспроизводится. При низких скоростях передачи данных по каналам Интернета буферизация порции информации может занимать больше времени, чем её воспроизведение. В этом случае воспроизведение носит неприятный прерывистый характер. Музыка, например, не звучит непрерывно, а иногда прорывается «кусками» сквозь тишину. При высоких скоростях передачи процесс буферизации проходит незаметно для пользователя.

До недавнего времени производством мультимедийных объектов для размещения в Интернете занимались специализированные фирмы.

С развитием цифровой техники в Интернете появляется всё больше мультимедийных объектов, созданных пользователями индивидуально.

Несколько лет назад в Интернете появились сайты, которые называются блогами. Термин «блог» является копией английского термина blog, который получен сокращением web log — «сетевой журнал» (причём от слова web взята последняя буква «b»).

Блог — это сайт, на котором один или несколько авторов ведут дневник (журнал), состоящий из текстовых записей с добавлением графики. Заметим, что компьютерные дневники отличаются от бумажных тем, что записи в них следуют в обратном хронологическом порядке, т. е. последняя по времени запись всегда является первой в порядке доступа.

Блоги предназначены для публичного просмотра. Читатели блога обычно могут оставить свои комментарии, вступить с автором в полемику. Множество всех блогов в Интернете часто называют ***блогосферой***. Многие современные блоги включают мультимедийное содержимое. Поэтому различают ***текстовые блоги***, ***аудиоблоги*** и ***видеоблоги***.

Для организации блога в Интернете необходимо наличие специальной программной системы управления содержимым блога. Автономные блоги со своей программной системой встречаются не часто, потому что с 1999 г. в сети функционируют сайты, которые услугу ведения собственного блога предоставляют пользователям бесплатно.

В Интернете широко распространены сайты, которые предоставляют услугу бесплатной публикации видеоматериалов. Пользователи имеют возможность размещать на таких сайтах свои видеоролики, просматривать и комментировать видеофрагменты других пользователей.

Подкасты — это аудиофайлы (в основном в формате MP3), которые размещены в Интернете и предназначены для скачивания или дистанционного воспроизведения. Подкасты содержат записи передач интернет-радиостанций, а также записи, сделанные обычными пользователями.

Подкастинг (от англ. podcasting) — способ публикации подкастов в Интернете. Пионером в деле развития подкастинга выступила фирма Apple, которая планировала распространение цифровых аудиозаписей через Интернет с целью их воспроизведе-

ния на своих портативных аудиоплеерах iPod. Именно отсюда берёт начало первая часть названий «подкастинг», «подкасты». Вторая часть слова «подкастинг» произошла от английского broadcasting — «распространение».

Мультимедийное общение в Интернете. Пока большинство пользователей Интернета были подключены к сети медленными каналами связи, общение между ними было текстовым — с помощью электронной почты и пейджинговых служб. С ростом скорости подключения появилась возможность шире использовать IP-телефонию. Наушники и микрофон позволяют сделать из компьютера телефон, который соединяется с другими абонентами по интернет-каналам. Появились в Интернете и возможности коллективного аудиообщения по типу селекторных совещаний.

Если скорость обмена по сети достигает 30—40 КБ/с, то можно воспользоваться услугами полноценной мультимедийной видеотелефонии. Подключение к компьютеру микрофона, наушников и веб-камеры превращает его в устройство, которое мы давно знаем как видеотелефон.

Мультимедийные технологии — это технологии создания и воспроизведения мультимедийных объектов и потоков.

В этой главе мы познакомимся с некоторыми из мультимедийных технологий.

Вопросы и задания

1. Какие формы представления информации мы изучали в 8 классе?

2. В чём состоит смысл термина «мультимедиа»?

3. Что такое мультимедийный объект?

4. Что такое мультимедийный контент?

5. Дайте определение мультимедийного потока.

6. Чем аналоговые мультимедийные объекты и потоки отличаются от цифровых?

7. Что такое линейный мультимедийный объект?

8. Как называются программные средства для воспроизведения линейных мультимедийных объектов на компьютере?

9. Что такое нелинейный мультимедийный объект?

10. Чем характерен мультимедийный компьютер?

11. Перечислите источники мультимедийных объектов и потоков.

12. Что такое компьютерный ТВ-тюнер?

13. В чём заключается дискретизация аналогового сигнала?

14. Определите понятие «оцифровка аналогового сигнала».

15. Что такое буферизация и для чего она используется?

16. Что такое блог?

17. Что такое подкасты и подкастинг? Что означает в этих терминах составная часть «под»?

18. Дайте определение мультимедийных технологий.

19. С помощью поисковых систем найдите в Интернете сайт с подкастами, скачайте и прослушайте один из подкастов (лучше самый маленький по объёму).

§ 52. Технологии воспроизведения линейных мультимедийных объектов и потоков

Мы уже говорили о том, что для воспроизведения на компьютере линейных мультимедийных объектов и потоков используют программы-плееры (программы-проигрыватели). В частности, в операционной системе Windows среди поставляемых в комплекте программ всегда была программа «Универсальный проигрыватель», которая начиная с Windows XP расширила свои возможности и теперь называется «Проигрыватель Windows Media».

Проигрыватель Windows Media предназначен для воспроизведения цифровых аудио- и видеофайлов и мультимедийных потоков.

Проигрыватель Windows Media версии 12 запускается с помощью ярлыка на рабочем столе или меню кнопки **Пуск**. После запуска на рабочем столе появляется окно программы (рис. 135).

Под строкой заголовка обычно отображается меню, которое можно скрыть или вывести сочетанием клавиш **Ctrl+M** (латинская буква). Левее адресной строки расположены кнопки **Назад** и **Вперёд**. Под адресной строкой расположена ещё одна панель с кнопками.

Область списка можно удалить с помощью меню кнопки **Параметры списка**, которая расположена в правой части продолжения панели задач. Возвращается (и удаляется) область списка щелчком по вкладке «Воспроизведение» (правее адресной строки).

1. Адресная строка
2. Область переходов
3. Область сведений
4. Область списка
5. Область элементов воспроизведения

Рис. 135

■ Упражнение 136

С помощью сочетания клавиш **Ctrl+M** удалим строку меню, а с помощью меню кнопки **Параметры списка** удалим область списка. Затем вернём на место строку меню и область списка.

Проигрыватель имеет три основных режима представления, в которые можно перейти при помощи команд меню (пункт **Вид**) или сочетаний клавиш: *режим библиотеки* (**Ctrl+1**), *режим текущего списка воспроизведения* (**Ctrl+2**) и *режим обложки* (**Ctrl+3**). На рисунке 135 проигрыватель представлен в режиме библиотеки.

■ Упражнение 137

С помощью сочетаний клавиш изменим режим отображения проигрывателя на режим обложки, затем на режим текущего списка воспроизведения, а потом вернёмся в режим библиотеки.

Обложка — это внешний вид проигрывателя в режиме обложки. Выбор новой обложки делается в окне проигрывателя после

ввода команды меню **Вид|Выбор обложки**. Список обложек можно пополнить в Интернете, щёлкнув по кнопке **Другие обложки**. В результате в браузере открывается веб-страница с обложками.

■ Упражнение 138

С помощью команды меню **Вид|Выбор обложки** выведем в окно список имеющихся обложек. Затем щелчком по кнопке управления **Назад** вернёмся в режим библиотеки.

В основных режимах представления окно проигрывателя, как обычно, можно перетаскиванием мыши перемещать по экрану или изменять в размерах.

■ Упражнение 139

Перетаскивая правый нижний угол окна программы, изменим размеры окна программы.

Типы воспроизводимых файлов. Запуск аудио- или видеофайла проводится в диалоговом окне открытия файла, которое вызывается командой меню **Файл|Открыть**.

После вывода диалогового окна можно ознакомиться с типами мультимедийных файлов, которые воспроизводит проигрыватель.

■ Упражнение 140

Вводим команду меню **Файл|Открыть**. В диалоговом окне открытия файла щёлкаем по кнопке правее поля «Имя файла:» (рис. 136). Знакомимся с перечнем типов. В завершение вводим команду отмены открытия файла.

В ОС Windows базовым форматом файлов звукозаписи является формат-контейнер WAV (тип .wav), который обычно содержит данные без потери качества в формате PCM. Другие форматы получаются из PCM сжатием с помощью различных алгоритмов или перекодированием. Именно такими сжатыми форматами являются формат WMA (тип .vma) и популярный формат MP3 (тип .mp3), сжимающий данные формата PCI примерно в 10 раз, но с потерей качества. Кстати, данные в этих форматах также могут быть записаны как файлы формата-контейнера WAV (тип .wav). Сжатие без потерь возможно с помощью форматов FLAC (тип .flac), Monkey's Audio (тип .ape), WMA Lossless (тип .wma) и др., но для воспроизведения файлов таких форматов могут понадобиться кодеки.

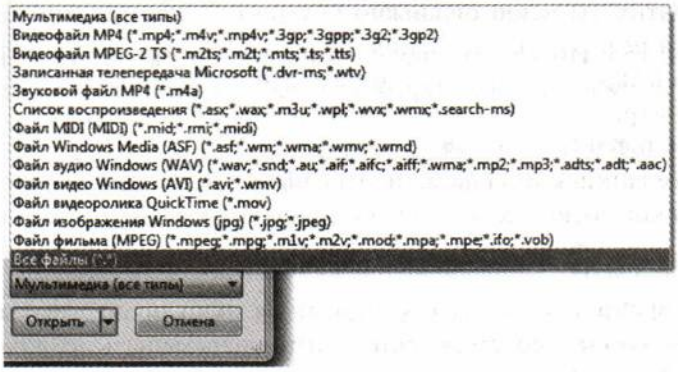


Рис. 136

Кодек — программа кодирования и декодирования аудио- или видеоданных.

Базовыми форматами файлов видеозаписи со звуком в ОС Windows являются формат-контейнер AVI (тип файла .avi) и формат WMV (тип файла .wmv). Для воспроизведения видеофайлов других форматов может понадобиться установка на компьютере видеокодеков.

Воспроизведение записей с музыкального компакт-диска. Запуск и управление воспроизведением проводится с помощью кнопок панели задач и нижней панели окна программы.

■ Упражнение 141

Воспользуемся любым музыкальным компакт-диском для воспроизведения его содержимого. Воспроизведение начинается автоматически, или надо сделать щелчок мышью по пиктограмме компакт-диска в области переходов.

В области сведений и в области списка проигрывателя появляется список имён записей диска с продолжительностью их воспроизведения. Этот список и называется *списком воспроизведения*.

В режиме обложки и в режиме текущего списка воспроизведения в окне проигрывателя автоматически начинается воспроизведение динамических зрительных образов. После щелчка правой клавишей мыши в контекстном меню можно выбрать вид зрительного образа.

Из этих же режимов можно сочетанием клавиш **Alt+Enter** перейти в режим представления «Во весь экран». Выход из режима «Во весь экран» проводится тем же самым сочетанием клавиш.

Проигрыватель исполняет произведения в порядке их следования в списке. Имя исполняемой композиции в списке выделено. При желании всегда можно перейти к воспроизведению любой другой композиции из списка воспроизведения.

■ Упражнение 142

Во время исполнения композиции сделаем двойной щелчок мышью по другому имени в списке воспроизведения. Тут же начинает звучать вновь выбранная композиция.

Можно задать также режим случайного порядка исполнения произведений.

■ Упражнение 143

Включим режим случайного порядка исполнения произведений щелчком по левой крайней кнопке в области элементов воспроизведения.

В режиме текущего списка воспроизведения после щелчка правой клавишей мыши в контекстном меню можно открыть окно настройки дополнительных возможностей проигрывателя.

■ Упражнение 144

В контекстном меню устанавливаем курсор на верхний пункт **Дополнительные возможности**. Открывается ещё одно меню, в котором выбираем любой пункт. Открывается окно настройки дополнительных возможностей. Щелчками по кнопкам со стрелками в левом верхнем углу просмотрим виды настроек.

Воспроизведение аудиофайлов проводится с помощью команды меню **Файл|Открыть**.

■ Упражнение 145

Запустим на исполнение файл формата WAV со звукозаписью с прилагаемого компакт-диска  .

Установим компакт-диск в дисковод и командой **Файл|Открыть** вызовем диалоговое окно открытия файла.

Выберем имя файла и щёлкнем по кнопке **OK**. Начнётся воспроизведение.

Существует и другой способ запуска файла на исполнение в проигрывателе. Запускаем программу «Проводник» и отображаем имя файла в области просмотра. Затем указателем мыши перетаскиваем пиктограмму файла из программы «Проводник» в область списка проигрывателя. Начинается воспроизведение.

Списки воспроизведения. Музыкальные компакт-диски имеют списки воспроизведения, которые можно просмотреть в области списка. Проигрыватель Windows Media даёт возможность пользователю составлять собственные списки воспроизведения из аудиофайлов, имеющихся в компьютере.

При первом запуске проигрыватель автоматически производит поиск файлов в уже известных нам библиотеках Windows «Музыка», «Видео», «Изображения», а также ещё в одной библиотеке «ТВ-записи». В области переходов отображаются эти библиотеки и внешние носители (компакт-диски и флэш-накопители). В области сведений можно вывести список входящих в библиотеку файлов.

■ Упражнение 146

Создадим новый список воспроизведения из записей, которые имеются в библиотеке проигрывателя.

Переводим окно проигрывателя в развёрнутое состояние, открываем область списка, увеличиваем её максимально влево и щёлкаем на её панели по кнопке **Очистить список**. В области переходов щёлкаем по пиктограмме библиотеки «Музыка» и перетаскиваем пиктограммы записей из области сведений в область списка. Затем в области списка вместо надписи «Несохранённый список» вводим имя нового списка и щёлкаем на панели по кнопке **Сохранить список**. Имя нового списка появляется в области переходов.

Воспроизведение видеофайлов производится аналогично воспроизведению аудиофайлов.

■ Упражнение 147

Запустим на исполнение видеофайл формата AVI, который записан на прилагаемом компакт-диске  .

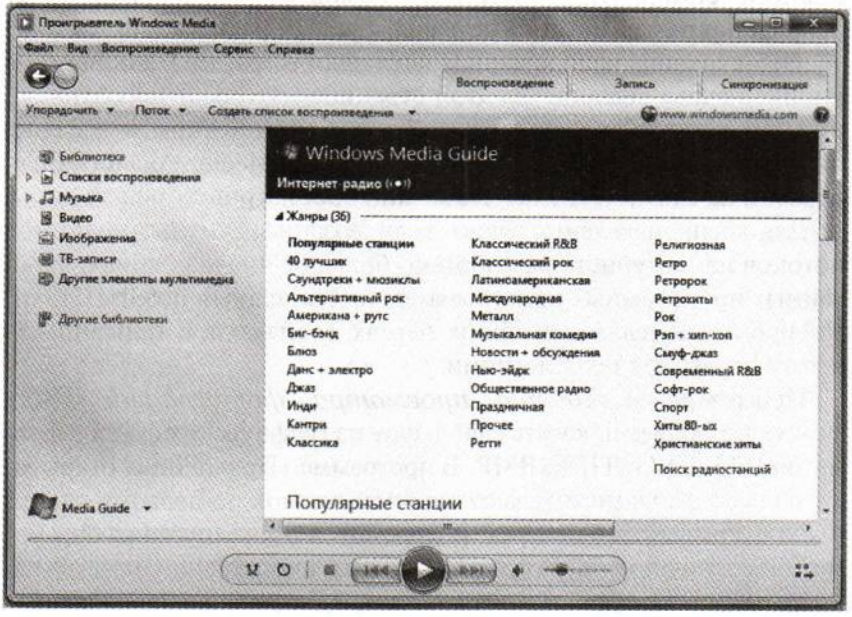


Рис. 137

Воспроизведение мультимедийных потоков из Интернета. Для работы с мультимедийными источниками в Интернете нужно иметь подключение к Всемирной паутине. При этом следует обратить внимание на скорость такого подключения, так как низкая скорость полноценного подключения не создаст.

Воспроизведение мультимедийных потоков в проигрывателе Windows Media связано с кнопкой **Media Guide** в нижней части области переходов. Щелчок по левой части этой кнопки отправляет запрос в Интернет, и через некоторое время в области сведений отображается страница сайта **www.windowsmedia.com**, посвящённая интернет-радио (рис. 137). Ниже списка жанров располагается раздел со списком зарубежных интернет-радиостанций, популярных, по мнению владельцев сайта.

В списке радиостанций около графической эмблемы каждой станции размещены две ссылки: «Слушать» и «Сайт». Щелчок по ссылке «Слушать» даёт команду на подключение проигрывателя к мультимедийному потоку радиовещания.

Браузер Internet Explorer также можно использовать для подключения к интернет-радиостанции. Веб-адрес сайта радио-

станции легко найти в Интернете с помощью поисковой системы. Запрос, например, может иметь вид «русские радиостанции».

Веб-страницу радиостанции загружают в браузер и щелчком по кнопке начала вещания на этой странице начинают прослушивание. Причём для прослушивания браузер может открыть ещё одно окно или вызвать проигрыватель Windows Media. Аналогичным образом на сайтах в Интернете можно просматривать видеоклипы.

Для воспроизведения аудио- и видеозаписей, мультимедийных потоков из Интернета разработано большое число специализированных программных проигрывателей. Технологии работы с такими проигрывателями в общих чертах совпадают с описанными в этом параграфе технологиями.

Программа *«Средство просмотра фотографий Windows»* позволяет показать слайд-шоу из графических файлов форматов JPG, PNG, TIFF, BMP. В программе «Проводник» открывают папку с файлами и делают двойной щелчок по первому из них. Файл открывается в программе просмотра. Для запуска слайд-шоу можно воспользоваться соответствующей кнопкой или нажать клавишу **F11** клавиатуры. Нажатие клавиши **Esc** останавливает шоу.

Показать слайд-шоу из графических файлов позволяет и программа «Windows Media Center», которая отличается необычностью дизайна и также запускается из «Проводника».

☐ Вопросы и задания

1. Для чего предназначен проигрыватель Windows Media?

2. Перечислите основные режимы отображения проигрывателя Windows Media.

3. Какой аудиоформат является базовым для ОС Windows?

4. Какие видеоформаты являются базовыми для ОС Windows?

5. Что такое кодек?

6. Какой объект в окне проигрывателя Windows Media отвечает за отображение и скрытие области списка?

7. В каком из режимов проигрывателя Windows Media можно открыть окно настройки дополнительных возможностей?

8. Из каких режимов представления проигрывателя Windows Media можно перейти в режим представления «Во весь экран»?

9. Чем характерен режим случайного порядка в проигрывателе Windows Media?

10. Опишите порядок составления списка воспроизведения в проигрывателе Windows Media.

11. Какая кнопка в окне проигрывателя Windows Media отвечает за работу с Интернетом?

12. Как использовать браузер Internet Explorer для просмотра видеоклипа с сайта в Интернете?

13. Как организовать демонстрацию слайд-шоу из графических файлов с помощью программы «Средство просмотра фотографий Windows»?

§ 53. Технологии создания графических объектов на базе цифровых фотографий

Обзор технологий создания графических объектов. Цифровые изображения широко используются при создании мультимедийных объектов (слайд-шоу и электронных презентаций), публикуются в Интернете.

Мы уже изучали технологии, которые можно применять для создания графических объектов. Это технологии растровой и векторной графики, 3D-графики, технология сканирования. В последнее время для создания графических объектов всё чаще используются графические планшеты, цифровые фотоаппараты и соответствующие этим устройствам технологии.

Графический планшет — это устройство ввода графической информации в компьютер.

Устройство представляет собой плоскую пластину, по которой, как карандашом по бумаге, можно водить специальным стержнем, который называется ***стилус***.

Используя разные режимы ввода, рисованием «от руки» создают графические образы. Результаты такого ввода фиксируются в компьютере как растровые графические объекты. Для профессиональных художников такой способ ввода графической информации гораздо удобнее, чем способ ввода с помощью мыши. Рисование на бумаге и холсте создаёт определённые навыки, которые трудно использовать, работая мышью в растровом графическом редакторе, так как малые движения мыши вызывают на экране большие движения указателя мыши.

Принципы построения графических планшетов положены в основу устройств, которые называются ***интерактивными доска-***

ми. Функциональными аналогами этих устройств являются аудиторные доски для записи информации в учебных заведениях (меловые, грифельные, фломастерные).

Интерактивная доска является одновременно экраном для проектора и большим графическим планшетом. Операции, проводимые с помощью стилуса, вызывают те же результаты, что и на малом планшете. Вот только результаты эти сразу же проецируются на доску. Создаётся полное впечатление, что стилус оставляет на планшете цветные следы.

Уже выпускаются ноутбуки, в которых прозрачный графический планшет совмещён с экраном компьютера. Такое совмещение позволяет «рисовать» стилусом на экране ноутбука.

Заметим, что тот же принцип положен в основу одного из типов беспроводных компьютерных мышей. Ковриком для такой мыши служит графический планшет, а мышь исполняет роль стилуса. Движения мыши преобразуются в движение указателя на экране.

Цифровые фотоаппараты сегодня являются сложными оптико-электронными приборами, которые не только фиксируют графическое изображение с тем или иным качеством, но и позволяют создавать цифровые видеоклипы.

Цифровые данные записываются в память фотоаппарата в виде файлов и могут переноситься в компьютер, на цветной принтер для печати фотографий, могут просматриваться с помощью встроенного экрана или с помощью телевизора.

В цифровых фотоаппаратах широко используются малогабаритные внешние карты памяти (рис. 138). Для чтения карт памяти в компьютерах используют специальные устройства ввода-вывода — *картридеры*, которые, как и модемы, бывают встроенные и внешние (рис. 139). В современных цифровых видеокамерах носителями записываемой информации являются мини-кассеты с магнитной плёнкой, DVD-диски диаметром 8 см, малогабаритные жёсткие диски.

Кроме того, цифровые фотоаппараты позволяют использовать при фотографировании целый набор эффектов и преобразований. Они могут оставлять на фотоснимке дату и время фотографирования, фиксировать цветное и чёрно-белое изображение, заменять одни цвета другими, проводить съёмку в условиях недостаточного освещения, макросъемку и т. д.

Рис. 138

Рис. 139

И всё же для создания на базе цифровых фотографий графических объектов с целью их включения в мультимедийные объекты и публикации в Интернете требуются дополнительные стилевые и размерные преобразования фотографий.

Программы для обработки цифровых фотографий называются *графическими редакторами*, причём в их названии обычно присутствует слово «фото». Например: Microsoft Photo Editor, Adobe Photoshop и т. п.

Графические редакторы для работы с цифровыми фотографиями во многом похожи, поэтому мы не будем останавливаться на конкретном редакторе, а разберём их общие основные возможности.

Изменение размера цифровых фотографий. К изменению размеров фотографий приходится прибегать, когда надо совместить изображения, сделанные с разным разрешением. Например, пусть одна фотография сделана с разрешением 640×480, а вторая — 1600×1200. Если совместить эти фотографии, то первая будет гораздо меньше второй, так как на экране плотность пикселов постоянна. Специальная операция позволяет изменить (уменьшить) размеры второй фотографии с соблюдением соотношений между размерами её сторон.

Обрезание ненужных частей изображения. Для обрезания ненужных частей изображения используют специальный инструмент выделения типа пунктирного прямоугольника в редакторе Paint. Пунктирный прямоугольник устанавливают на изображении, выделяя то, что должно остаться. Далее пользуются буфером обмена. Выделенный фрагмент копируют в буфер и вставляют в окно редактора как новый графический объект.

Рис. 140

Поворот изображения. Операция поворота изображения знакома нам по работе в графических редакторах. Для цифровых фотографий эта операция практически совпадает с аналогичной операцией в графическом редакторе Paint. Отличие состоит только в том, что в графическом редакторе Paint повернуть объект можно только на угол, кратный 90°, а в редакторах для работы с фотографиями повернуть изображение можно на любой угол, измеряемый целым числом градусов.

Рис. 141

Изменение стиля изображения. Графические редакторы, которые используются для работы с цифровыми фотографиями, именно в вопросах изменения стиля изображения имеют самые большие возможности.

В цифровых фотографиях, как в телевизионном изображении, можно изменить степень яркости, контрастности, цветовой гаммы для всей фотографии и для каждого цвета системы RGB в отдельности. Можно изменить цветность, резкость, добавить размытие. Один из цветов можно сделать прозрачным, что добавляет естественность при взаимном наложении фотографий.

Фильтрами называют операции изменения внешнего вида фотографии.

Применение фильтров может превратить фотографию (рис. 140) в размытый образ, в картину, написанную углём, в мозаику, в гравюру (рис. 141—144).

Рис. 142

Добавление рамок. Специальные фильтры в графических редакторах добавляют к фотографиям изображения рамок (рис. 145). В некоторых графиче-

Рис. 143

Рис. 144

Рис. 145

ских редакторах разнообразие подобных рамок практически необозримо.

Обои и штампы. *Обоями* называют ритмически повторяющийся рисунок, которым заполняют пустые области коллажей. *Штампами* называют простые графические изображения, которые без изменений можно поместить в разных частях фотографии или коллажа.

Возможность использовать фильтр с рамкой, обои и штампы при создании открыток на базе цифровых фотографий стала обычной даже для автономных фотопринтеров, которые позволяют изменять и печатать цветные фотографии без присоединения к компьютеру.

Вопросы и задания

1. Что такое графический планшет?

2. Что такое стилус?

3. Что такое картридер?

4. Что такое интерактивная доска?

5. Перечислите основные операции, которые обеспечивают графические редакторы для работы с цифровыми фотографиями.

6. Что такое фильтр в графическом редакторе для работы с цифровыми фотографиями?

7. Что такое обои в графическом редакторе для работы с цифровыми фотографиями?

8. Что такое штамп в графическом редакторе для работы с цифровыми фотографиями?

§ 54. Технологии компьютерной обработки звука

Оцифровка звука. С азами дискретизации звуковой информации мы познакомились в 8 классе. Напомним, что дискретизация аудиосигналов включает дискретизацию по времени и дискретизацию амплитуды.

Для кодирования звуков обычно используют частоту дискретизации по времени (частоту *сэмплирования*), которая вдвое больше, чем частота кодируемого сигнала. Так как обычный человек воспринимает звуки с частотой от 20 до 20 000 Гц, для качественной дискретизации используют частоту более 40 000 Гц.

Для записи дискретных значений амплитуды сигнала используют код объёмом 16 битов, что соответствует $2^{16} = 65\,536$ дискретным уровням, тогда качество записи с частотой 44 100 Гц называют *качеством CD*, а с частотой 48 000 Гц — *качеством DVD*. С качеством CD в одном звуковом канале каждую секунду записывается $2 \times 44\,100$ байт цифровой информации. При записи стереомузыки используются два независимых канала. Отсюда получается объём цифровой информации более 10 Мб на одну минуту цифровой записи. Это примерная характеристика объёма аудиозаписи в формате WAV (16 битов PCM).

Заметим, что для записи в профессиональных студиях используется *студийное качество* с частотой сэмплирования 88 200 Гц и более и (или) кодами объёмом 24 бита (2^{24} уровней) для амплитуды. С другой стороны, для цифровых диктофонов достаточно одноканальной записи с частотой 8000 Гц и объёмом кода дискретизации амплитуды в 8 битов ($2^8 = 216$ уровней). Такое качество записи называют *качеством речи*. Понятно, что в последнем случае объёмы записываемой цифровой информации существенно уменьшаются.

Сжатие цифровых аудиоданных. Мы уже говорили о том, что для уменьшения объёмов записываемой цифровой информации к данным формата WAV применяют методы сжатия (с потерями и без потерь). При сжатии с потерями часть информации теряется, и качество звука снижается.

Показателем, который характеризует объём оставшейся после сжатия информации, является скорость потока цифровых данных из

сжатого файла. Этот показатель называют *битрейтом* (bitrate — частота битов). Битрейт измеряют в килобитах в секунду.

Очевидно, что чем выше битрейт, тем качественнее звук, но тем больше объём сжатого файла. Для качественного воспроизведения достаточно значения битрейта в 128 Кбит/с. Для записи музыки битрейт ниже 48 Кбит/с не используют. А вот для уже упомянутого диктофона хватает битрейта и в 8 Кбит/с.

Запись звука с помощью компьютера. Для записи звука на компьютере существует много разнообразных программ. Мы воспользуемся программой «Звукозапись», которая поставляется вместе с ОС Windows. Записывать можно только сигналы, которые подаются на вход звуковой платы, куда можно подключить микрофон или аудиопроигрыватель. Запись можно прослушать, повторить, записать в виде файла на диск, отредактировать и снова сохранить на диске.

■ Упражнение 148

Запишем с помощью программы «Звукозапись» звук с микрофона.

Для начала подключим микрофон в гнездо звуковой платы. Запись выполняем в следующем порядке.

1. Запускаем программу «Звукозапись». Запуск этой программы производится с помощью меню кнопки **Пуск**, где программа находится в папке **Стандартные**. Открывается очень маленькое окно программы с двумя кнопками.

2. Настраиваем параметры громкости. Для этого в правой части панели задач щёлкаем по пиктограмме «Динамики». На открывшейся панели с регулятором щёлкаем по кнопке **Микшер**. В открывшемся окне «Микшер громкости» ползунками выставляем максимальные уровни громкости для входящих устройств.

3. Устанавливаем микрофон. В окне программы щёлкаем по кнопке **Начать запись** и произносим несколько слов. Останавливаем запись щелчком по кнопке **Остановить запись**. Открывается окно сохранения файла, в котором вводим имя. Расширение .wma при сохранении добавляется автоматически.

4. В программе «Проводник» двойным щелчком мыши запускаем файл с записью на воспроизведение. Если качество полу-

ченной записи нас не устраивает, то изменяем параметры громкости и повторяем запись.

Программа «Звукозапись» позволяет в одном файле записывать несколько звуковых фрагментов. Для этого после остановки записи первого фрагмента и появления диалогового окна «Сохранить как» нужно в этом окне щёлкнуть по кнопке **Отмена**. Диалоговое окно закрывается, а основная кнопка программы «Звукозапись» преобразуется в кнопку **Возобновить запись**. Щелчком по ней запись в любой момент можно возобновить. Так можно поступать несколько раз. Сохранение файла с записью проводится в соответствии с пунктом 3 порядка действий.

Второй кнопкой программы «Звукозапись» является кнопка **Справка** (с пиктограммой вопроса) функции которой очевидны.

Редактирование аудиозаписей. При редактировании аудиозаписей проводятся следующие основные операции:

- удаление части записи;
- повторение части записи;
- объединение нескольких записей;
- изменение уровня громкости всей записи и её частей;
- изменение формата файла с записью.

Редактирование цифровых аудиозаписей производят с помощью *редакторов аудиофайлов*. Мы познакомимся с возможностями редактора Audacity версии 1.3.12-beta, который поддерживает русскоязычный интерфейс и распространяется свободно.

■ Упражнение 149

С помощью программы «Проводник» скопируем с прилагаемого CD-диска аудиофайл с именем music.wav в рабочую папку. Запустим на исполнение редактора аудиозаписей Audacity, настроим его и загрузим в него файл music.wav.

После запуска редактора Audacity на экране открывается окно программы с излишне большим числом панелей.

Командой меню **Вид|Панели** открываем выпадающее меню с перечнем названий всех панелей редактора. Снимаем пометку флажка у пункта **Панель индикаторов**. Далее аналогичными действиями убираем из окна панель инструментов и панель транскрипции.

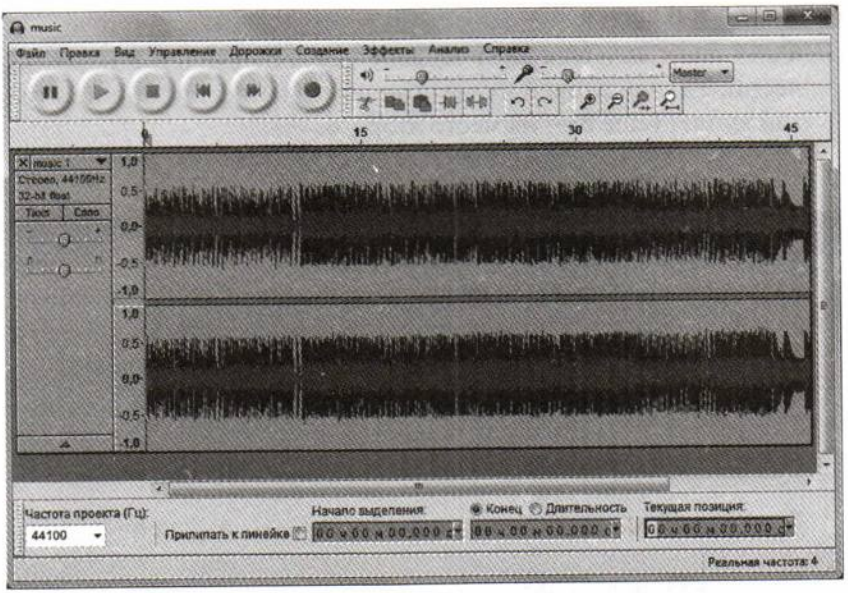


Рис. 146

На панели выделения (в нижней части окна) щелчком по кнопке со стрелкой правее любого из трёх полей открываем меню единиц измерения, в котором выбираем четвёртую сверху позицию. Измерения будут проводиться с точностью до сотых долей секунды.

Командой меню **Файл|Открыть** вызываем диалоговое окно загрузки файла и загружаем файл music.wav. Окно редактора принимает вид, представленный на рисунке 146.

Воспроизведением записи можно управлять с помощью кнопок панели управления, а громкость устанавливать движком на панели микшера (правее пиктограммы громкоговорителя).

Работа в редакторе осуществляется с выделенными фрагментами записи. Фрагмент выделяют перетаскиванием указателя мыши по звуковой дорожке (любой из двух). В нижней части окна на панели выделения в первом поле выводится время от начала записи до начала фрагмента, во втором поле — время от начала записи до конца фрагмента, в третьем — время текущего момента воспроизведения.

Если фрагмент выделен, то запись воспроизводится только в его пределах. Края выделения перемещают перетаскиванием мышью или изменением значений в полях начала и конца фрагмента.

■ Упражнение 150

Файл music.wav содержит музыку четырёх куплетов народной песни. Удалим второй куплет (он помедленнее) и запишем третий куплет (ещё раз) на место второго.

Работаем в следующем порядке.

1. Находим момент начала второго куплета. Для этого запускаем воспроизведение и приостанавливаем его в нужном месте (точность зависит от скорости вашей реакции). Выделяем небольшой фрагмент, включающий момент остановки, и увеличиваем его отображение инструментом «Уместить выделенное». Новое выделение начинаем перед неразрывной группой импульсов. Запоминаем время остановки (10,84 с). Инструментом «Уместить проект» выводим в окне всю запись.

2. Находим момент окончания второго куплета (23,09 с).

3. Выделяем второй куплет. Сначала выделяем его мышью, а затем в полях панели выделения вводим значения моментов начала и конца фрагмента (щелчок по полю и ввод с клавиатуры).

4. Удаляем второй куплет командой меню **Правка|Удалить**.

5. Находим момент окончания нового второго куплета (22,02 с).

6. Выделяем новый второй куплет.

7. Копируем новый второй куплет в буфер обмена командой меню **Правка|Скопировать**.

8. Щёлкаем по звуковой дорожке приблизительно на временно́й отметке 10—11 с. Устанавливаем для начала и конца выделения одно и то же значение 10,84 с.

9. Вставляем куплет из буфера обмена командой меню **Правка|Вставить**.

10. Сохраняем результат в файле music2.wav, вызвав окно сохранения командой меню **Файл|Экспортировать**.

Чтобы объединить две записи, нужно одну из них загрузить, а вторую импортировать (команда меню **Файл|Импортировать|Звуковой файл**). Обе записи отображаются в одном окне одновременно. Одну запись выделяют, копируют и вставляют в конце другой.

Для изменения параметров записи (например, уровня громкости) выделяют всю запись или её фрагмент, а затем используют пункт меню **Эффекты**. В частности, там есть пункты **Плавное нарастание, Плавное затухание** и **Усиление сигнала**.

Чтобы изменить формат файла, файл загружают и экспортируют, выбирая при сохранении другой формат.

Компьютерный синтез звука. Современные компьютеры способны не только проигрывать музыкальные звукозаписи. В звуковые платы компьютеров встроены синтезаторы звуков, которые можно активизировать специальными командами. Набор команд для синтезатора (фактически — программа) записывается в файл с расширением .midi или .mid. Исполнение такой программы приводит к воспроизведению музыки.

Аббревиатура *MIDI* расшифровывается как *Musical Instrument Digital Interface* — цифровой интерфейс музыкальных инструментов. Файлы формата MIDI содержат команды для воспроизведения звучания нескольких инструментов одновременно, т. е. существует несколько каналов воспроизведения. Эту особенность в телефонии называют многоголосием.

Воспроизведение MIDI-файлов компьютер осуществляет с использованием *волновых таблиц*, в которых записаны коды фрагментов звучания реальных инструментов. Такие фрагменты звучания музыкальных инструментов называются *сэмплами*.

К компьютеру можно подключить MIDI-клавиатуру, которая имеет вид клавиатуры электромузыкального инструмента и выдаёт прямые команды синтезатору компьютера. Существуют и программы, которые превращают обычную компьютерную клавиатуру в MIDI-клавиатуру. Исполнять музыку с помощью такой клавиатуры, конечно, можно, но очень затруднительно.

Музыку компьютерных синтезаторов используют для создания караоке-файлов (расширение .kar). Караоке-файлы являются теми же MIDI-файлами, в которых один из каналов содержит не команды синтезатору, а слова песни. Слова отображаются на экране компьютера одновременно и в такт с исполнением музыки.

Вопросы и задания

1. Почему приемлемая частота дискретизации имеет значение 44 100 или 48 000 Гц?

2. Коды какого объёма используют в звукозаписях для записи дискретных значений амплитуды?

3. С какой целью проводят сжатие аудиоданных?

4. Что такое битрейт?

5. В каких единицах измеряется битрейт?

6. Опишите порядок изменения формата аудиофайла с помощью редактора Audacity.

7. Какие операции обычно производятся при редактировании аудиофайлов?

8. Для каких целей используется компьютерный синтез звуков?

9. Найдите в информационных ресурсах Интернета программу для получения MIDI-звуков с помощью клавиатуры компьютера. Попробуйте исполнить что-либо музыкальное.

10. По материалам Интернета составьте сообщение о состоянии и проблемах в области разработки и использования программ распознавания речи.

§ 55. Технологии компьютерной обработки видеоизображений

Композиция и монтаж. Технологии компьютерной обработки видеоизображений имеют глубокие корни в технологиях создания кинофильмов.

Создание фильма (видеофильма или даже видеоклипа) начинается с замысла, который воплощается в *сценарии*. На основе сценария разрабатывается *сценарный план*, в котором описаны все подробности каждой сцены, каждого фрагмента.

После проведения всех подготовительных работ проводятся *съёмки фрагментов* будущего фильма. Порядок съёмки фрагментов может быть произвольным. Иногда съёмки начинают с фрагментов из середины фильма или даже с финальных. И наконец, завершает процесс производства этап монтажа фильма.

Монтаж фильма — это процесс объединения отдельных видеофрагментов в единый фильм.

Монтаж фильма — очень важный этап. Именно на этом этапе режиссёр принимает окончательные композиционные решения, и обретает смысл вся предыдущая работа, разбитая иногда на совершенно несвязанные элементы.

Композиция (от лат. compositio — составление, связывание) — это построение художественного произведения, обусловленное его содержанием, характером и назначением.

Компьютерный монтаж. Носителями современных кинофильмов является как киноплёнка, так и цифровые носители. В быту видеосъёмка на цифровые носители также постепенно становится обыденным явлением. Часто при этом обходятся вообще без цифровой видеокамеры: для цифровой съёмки достаточно возможностей современных цифровых фотоаппаратов, да и многие модели мобильных телефонов имеют встроенные фотоаппараты с возможностью видеосъёмки.

Цифровые фотоаппараты и телефоны с встроенными камерами зачастую имеют ограничения на длительность непрерывной съёмки. Это ограничение обусловлено техническими причинами. Но для создания занимательного видеоклипа такие ограничения препятствием не являются.

Конечно, создание сценария для любительских съёмок — дело необязательное. Ведь любительские съёмки, по сути, являются документальными, они фиксируют окружающие нас явления и события. Предугадать всё или подстроить что-то практически невозможно. Но это означает, что для режиссёров-любителей монтаж единого фильма из отдельных отснятых фрагментов (видеоклипов) приобретает вообще первостепенное значение. Композиция будущего фильма в этих случаях рождается только после просмотра всех фрагментов.

Для монтажа цифровых видеоклипов предназначены специальные компьютерные программы. Профессиональные программы компьютерного видеомонтажа имеют множество функций и достаточно сложны в освоении даже профессионалами. Мы познакомимся с азами компьютерного монтажа видеоклипов на примере стандартной программы видеомонтажа, которая носит название Windows Movie Maker («творец фильмов») версии 2.6 и распространяется свободно.

В программе Windows Movie Maker используются следующие термины.

Клип — фрагмент фильма, видеоклип.

Сборник — папка, которая содержит коллекцию клипов к фильму.

Проект — последовательность клипов в создаваемом фильме.

Видеоэффект — способ изменения визуальных параметров клипа (цвета, резкости и т. п.).

Видеопереход — динамическая форма смены одного клипа другим.

Windows Movie Maker позволяет загрузить исходные видеоматериалы, разбить их на клипы, из клипов составить проект фильма, к любому клипу в проекте добавить видеоэффект (из готового набора), между клипами вставить видеопереход (из готового набора), добавить в проект кадры с названием фильма и титрами. Как результат программа позволяет сохранить проект в файле особого формата MSWMM и сохранить фильм в файле формата Windows Media Video (WMV).

Запуск программы Windows Movie Maker. Программа запускается с помощью меню кнопки **Пуск** или ярлыка на рабочем столе. Открывается окно программы (рис. 147). В окне обычное место занимают строка меню и панель инструментов.

Центральная часть окна разбита по горизонтали на три зоны (см. рис. 147). Левая зона является *панелью сборников*. На ней в виде дерева папок отображены имена существующих сборников, а также сборники видеоэффектов и видеопереходов. Панель может быть убрана или восстановлена командой меню **Вид|Сборники**. На месте панели сборников может отображаться *панель задач* (команда меню **Вид|Панель задач**). На панели задач (второе её название — *панель операций*) отображаются операции, которые можно выполнить с помощью Windows Movie Maker. Средняя зона называется *панелью содержимого*. На этой панели можно отобразить клипы любого сборника, набор видеоэффектов или набор видеопереходов. Правая зона является *монитором*, с помощью которого можно просмотреть любой клип и весь проект фильма целиком. Ниже монитора расположены управляющие кнопки.

Ниже панели сборников, панели содержимого и монитора расположена область проекта, в которой создаётся и редактируется

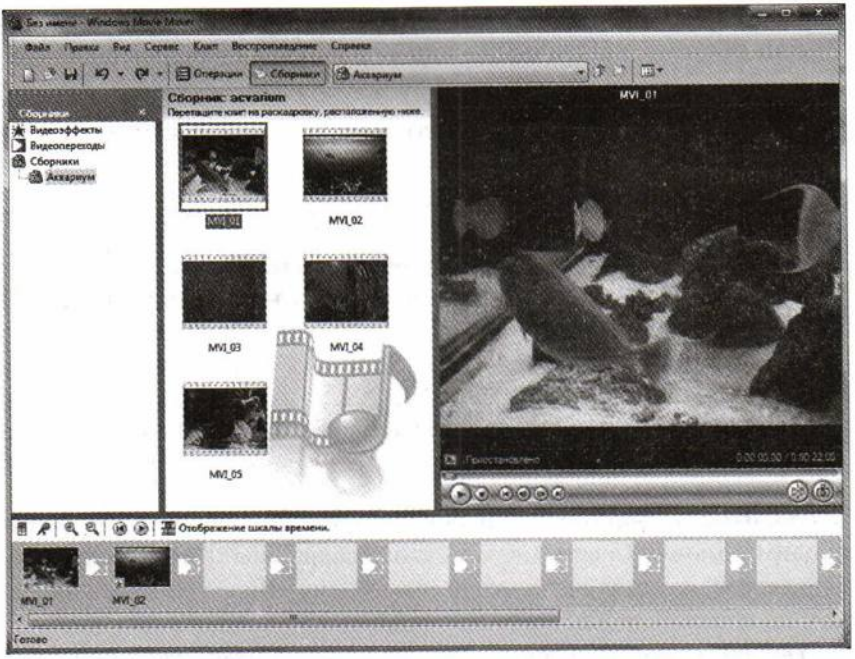


Рис. 147

проект фильма. Область проекта имеет два режима отображения: режим «Раскадровка» и режим «Шкала времени». Переключение между этими режимами производится щелчками по кнопке в верхней части области проекта. В режиме «Раскадровка» отображается последовательность клипов и видеопереходов в фильме.

Настройка программы Windows Movie Maker. Программа видеомонтажа Windows Movie Maker перед использованием требует настройки. Диалоговое окно для настройки вызывается командой меню **Сервис|Параметры**.

1. На вкладке «Общие» в верхнем поле вводят фамилию автора.
2. Там же щёлкают по кнопке **Сброс предупреждений**, чтобы уменьшить количество предупреждающих диалоговых окон.
3. На вкладке «Дополнительные параметры» устанавливают продолжительность изображения (выдержку) и продолжительность перехода (видеоперехода). Продолжительность изображе-

ния используется для преобразования цифровой фотографии или рисунка в статичный клип с заданным временем демонстрации.

4 В разделе «Свойства видео» обычно задают формат видео PAL и отношение сторон 4 : 3.

5 Настройка завершается щелчком по кнопке **OK**.

Загрузка исходных видеоматериалов. Для исходных материалов к новому видеофильму создают новую папку-сборник на панели сборников.

■ Упражнение 151

Создадим папку-сборник с именем «Видео1» и загрузим в неё исходные видеоматериалы.

На панели сборников щёлкаем по папке «Сборники». Затем вводим команду **Сервис|Создать папку сборников**. Папка появляется в дереве, и остаётся только ввести её имя «Видео1». Далее щелчком по папке выделяем её.

Теперь можно загружать ссылки на видеоматериалы. Для этого вводится команда меню **Файл|Импорт в сборники**. Открывается диалоговое окно «Импорт файла».

Файл типа AVI (ссылку) можно импортировать целиком либо разбить его на клипы в автоматическом режиме. Во втором случае в нижней части окна помечается флажок возле надписи «Создание клипов для файлов видео». Затем щёлкают по кнопке **Импорт**. Для каждой загруженной ссылки на файл в разделе «Сборники» автоматически создаётся папка с тем же именем. Теперь переместим все ссылки в папку «Видео1», используя контекстное меню правой клавиши мыши и операции «Вырезать» и «Вставить».

Замечание. Некоторые цифровые фотоаппараты и телефоны записывают видеоклипы в формате MOV. Этот формат разработан фирмой Apple для собственных пользователей и в ОС Windows поддержки не имеет. В Интернете нужно найти программу-декодер формата MOV в формат AVI и перекодировать такие файлы.

Аналогично загружаются ссылки на цифровые фотографии и рисунки. В области сборников они преобразуются в статичные клипы. Можно загрузить только фотографии и создать слайд-шоу.

Редактирование клипов. Щелчок по клипу на панели содержимого открывает его изображение на мониторе, а над монитором появляется имя клипа. На мониторе клип просматривают и при необходимости редактируют (удаляют какие-то его части).

Программа видеомонтажа Windows Movie Maker обеспечивает только одну операцию редактирования клипов: разделение клипа на две части (на два новых клипа). Для разделения клипа движок под монитором устанавливают в точке разделения и щёлкают по кнопке **Разделение клипа на две части по текущему кадру**. Появляются два клипа, из которых один оставляет за собой исходное имя (первый по времени воспроизведения), а другой добавляет к тому же самому имени номер в скобках (второй по времени).

Если нужно оставить какую-то внутреннюю часть клипа, то разделять его придётся дважды. Главное — заметить для себя на оси времени граничные точки требуемой части клипа. При этом поступают следующим образом.

1. Клип разбивают на два клипа в одной граничной точке.
2. Один из получившихся клипов разбивают на два клипа в другой граничной точке.

В результате разделения получатся три клипа, из которых только один пойдёт в работу.

Именно так клипы готовятся к монтажу.

Приёмы монтажа. Для монтажа область проекта переводят в режим «Раскадровка». Один большой прямоугольник — это место для одного клипа. Прямоугольники поменьше предназначены для видеопереходов.

Готовые клипы по одному перетаскивают указателем мыши из панели содержимого в область проекта и устанавливают в кадрах. Перетаскиванием можно менять порядок клипов. Можно выделить клип на раскадровке и воспользоваться буфером обмена (вырезать, копировать, вставить). Можно выделить несколько клипов подряд (щелчками мыши с нажатой клавишей **Shift** клавиатуры) и скопировать их в другую часть ряда клипов.

Чтобы просмотреть на мониторе готовую часть фильма, нужно щёлкнуть в области проекта по первому клипу. С этого момента управляющие кнопки монитора работают уже со всей готовой частью фильма.

Заметим, что Windows Movie Maker в режиме «Шкала времени» позволяет добавлять звуковую дорожку к видеоряду.

Сохранение готового фильма. Проект фильма можно сохранить в файле формата MSWMM (расширение .mswmm) командой меню **Файл|Сохранить проект как**.

Готовый фильм сохраняется в формате Windows Media Video (расширение .wmv). Сохранение начинается с выбора команды меню **Файл|Сохранить файл фильма**. Появляется диалоговое окно мастера сохранения.

С помощью подсказок выбирают или вводят параметры сохранения и переходят от одного окна к другому. После записи фильма щёлкают по кнопке **Готово**.

Понятие о мультипликации. Мультфильмы, кукольные фильмы, компьютерные мультфильмы любимы нами с детства.

Мультипликация (лат. multiplicatio — умножение), **анимация** (англ. animation — одушевление) — вид киноискусства, в котором произведения создаются методом покадровой съёмки отдельных рисунков или сцен.

В графической мультипликации используются графические рисунки, силуэтные теневые изображения, перекладки типа коллажей. В объёмной мультипликации фотографируют на плёнку положения кукол, барельефов, пластилиновых объектов. В компьютерной мультипликации кадры синтезируются специальными программами с помощью расчётных методов. Современными видами мультипликации являются 3D-анимации и Flash-анимации.

Создание простой анимации. Воспользуемся возможностями программы видеомонтажа для создания простой анимации.

■ Упражнение 152

С помощью графического редактора Paint создадим несколько рисунков, которые с помощью программы видеомонтажа Windows Movie Maker используем для создания простого анимационного фильма.

Запускаем графический редактор **Paint** и командой меню **Paint|Свойства** вызываем диалоговое окно. Устанавливаем единицы измерения «Точки» и размеры 320 × 249 пикселов. Рисуем на холсте рисунок переднего вида автомобиля (рис. 148) без пунктирных линий. Сохраняем рисунок под именем M1 с расширением .jpg.

Чтобы последовательность рисунков описывала движение, они должны в небольшой степени отличаться друг от друга. Рисованием мультфильмов занимаются художники, имеющие к этому талант. Для наших же целей удобнее воспользоваться «индустриальными» методами, а конкретнее — методом изменения масштаба изображения.

Выбираем инструмент «Выделение» и строим пунктирный прямоугольник так, как это изображено на рисунке 148 (под нижний обрез изображения). Ставим курсор на угол 1 и немного буксируем его в указанном направлении. Затем то же самое делаем с углом 2 в указанном направлении. Размеры сдвигов должны быть очень малыми. Результат сохраняем под именем M2.jpg.

Опять пользуемся инструментом выделения и делаем то же самое. Результат сохраняем под именем M3.jpg. Аналогично поступаем 8—10 раз. Получаем набор рисунков для анимации.

Запускаем программу Windows Movie Maker. Создаём новую папку-сборник и импортируем в неё все созданные нами рисунки. Это можно сделать за один раз: просто в диалоговом окне для выбора объекта надо выделить сразу все имена рисунков.

Чтобы отдельные статичные кадры в фильме быстро сменяли друг друга, установим минимальные длительности изображения и перехода в окне «Параметры» на вкладке «Дополнительные пара-

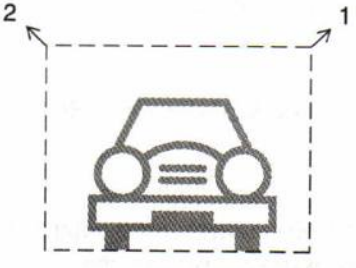


Рис. 148

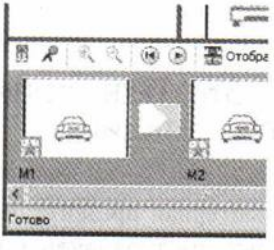


Рис. 149

метры». Напомним, что это окно вызывается командой меню **Сервис|Параметры**.

Затем нужно перетащить рисунки последовательно в область проекта на раскадровку (рис. 149).

В заключение переходим к просмотру видеоклипа мультипликации. Чем больше кадров, тем продолжительнее воспроизведение клипа.

☐ Вопросы и задания

1. Что такое монтаж фильма?

2. Что такое композиция?

3. Опишите внешний вид окна программы Windows Movie Maker.

4. В чём состоит смысл понятия «длительность изображения» при настройке программы Windows Movie Maker?

5. Какой командой меню начинается процесс загрузки исходных видеоматериалов (ссылок)?

6. Какую операцию редактирования клипов обеспечивает программа Windows Movie Maker?

7. Опишите основные приёмы видеомонтажа в программе Windows Movie Maker.

8. Объедините в видеофильмы клипы о морских обитателях, скачанные из Интернета.

9. Что такое мультипликация?

10. Расскажите о видах мультипликации.

11. В информационных ресурсах Интернета найдите информацию по истории мультипликации.

12. Создайте с помощью программы Windows Movie Maker небольшую анимацию по собственному сценарию.

§ 56. Технологии создания компьютерных презентаций

Компьютерная презентация — это набор электронных слайдов, который предназначен для показа на экране компьютера или на большом экране с помощью компьютерного проектора.

В последнее время компьютерные презентации стали очень популярными. Презентации сопровождают доклады и выступления, используются в рекламных целях на выставках и конференциях.

Использование презентации повышает наглядность информации. Компьютерные презентации могут включать элементы мультипликации, звуковые эффекты, записи музыки и речи, поэтому они являются реальными мультимедийными объектами. Строятся компьютерные презентации по правилам обычного слайд-шоу. В ОС Windows для создания мультимедийных презентаций используется программа Microsoft PowerPoint. Мы несколько подробнее остановимся на технологиях работы с Microsoft Office PowerPoint 2010.

Запуск PowerPoint и общий вид экрана. Программа запускается с помощью ярлыка на рабочем столе или с помощью меню кнопки **Пуск**. На рабочем столе открывается окно программы (рис. 150).

Интерфейс программы легко узнаваем по наличию строки меню и ленты с вкладкой — полная аналогия интерфейсу Word 2010 и Excel 2010.

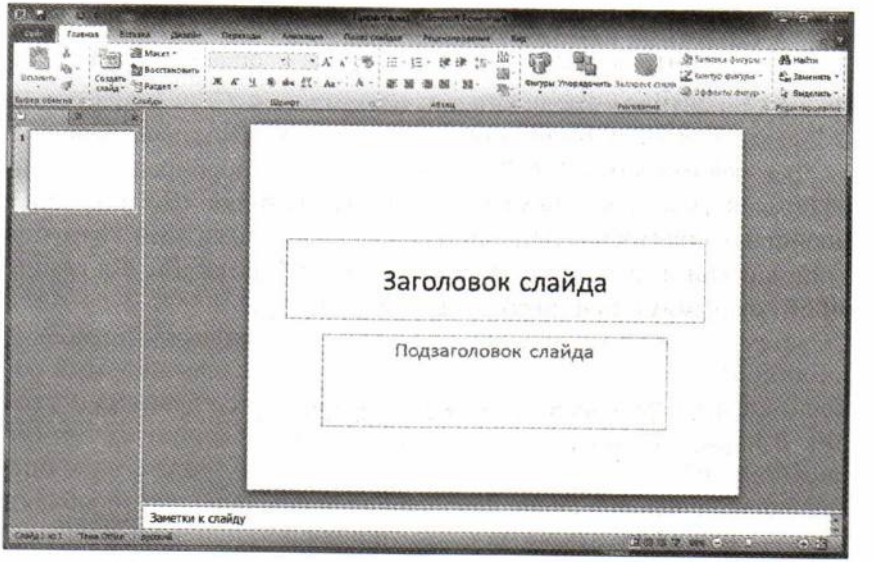


Рис. 150

В области просмотра слева расположена область «Структура и слайды» с вкладками, а справа — рабочая область «Слайд» с заготовкой слайда. Под областью «Слайд» размещена область «Заметки». Границы областей можно перемещать перетаскиванием указателя мыши.

Область «Структура и слайды» можно скрыть кнопкой в правом верхнем углу области. На экран эта область возвращается нажатием клавиши **F6** клавиатуры, причём нажать её надо несколько раз.

В области «Слайд» создают слайды. В области «Структура и слайды» устанавливают и изменяют порядок следования слайдов.

Оформление слайда должно соответствовать его содержимому, но не следует пренебрегать и общими рекомендациями:

- не следует перенасыщать слайд текстом, лучше сделать больше слайдов;
- не рекомендуется использовать много разнообразных шрифтов для надписей;
- не рекомендуется делать слайд слишком разноцветным;
- не следует использовать много анимационных эффектов.

Отступление от приведённых выше рекомендаций приводит к тому, что слушатели отвлекаются от содержания презентации и увлекаются её эффектами.

Создание слайда презентации. Для презентаций определённых видов (викторины, фотоальбомы и др.) в программе PowerPoint 2010 имеются шаблоны оформления. Список шаблонов открывается командой меню **Файл|Создать|Образцы шаблонов**.

Для презентаций других видов сначала подбирают дизайн оформления слайдов (тему и цвета). На вкладке «Дизайн» тему подбирают в группе «Темы». Для выбранной темы в этой же группе кнопкой **Цвета** можно подобрать другой набор цветов, а в группе «Фон» кнопкой **Стили фона** — другой фон.

Затем на вкладке «Главная» в группе «Слайды» с помощью кнопки **Макет** подбирают разметку слайда, если предлагаемый набор пунктирных наполнителей (чаще всего — надписей) не подходит. В наполнители-надписи вводят текст или меняют их тип (на рисунок, диаграмму и т. п.), следуя подсказкам. Можно дополнительно вставить на слайд рисунок, надпись или другой объект, пользуясь инструментами вкладки «Вставка».

Фигуры, знакомые нам по редактору Word, вставляют на вкладке «Главная» в группе «Рисование». В этой же группе кнопка **Упо-**

рядочить позволяет перемещать объекты по слоям (вперёд-назад), группировать, разгруппировывать, поворачивать и отражать их.

После завершения работы над первым слайдом на вкладке «Главная» в группе «Слайды» щёлкают по нижней части кнопки **Создать слайд**. В выпадающем меню щелчком выбирают макет слайда. В области «Слайд» появляется новый слайд. Слева на вкладке «Слайды» также появляется эскиз нового слайда. Дизайн слайдов в презентации лучше не менять. После создания всех слайдов презентации переходят к настройке их демонстрации.

Настройка демонстрации. Программа PowerPoint имеет несколько режимов просмотра презентации.

Режим просмотра «Обычный», в котором мы работали, используется для создания отдельных слайдов.

В этом режиме в области «Структура и слайды» на вкладке «Слайды» можно перетаскиванием менять порядок слайдов или использовать буфер обмена, инструменты которого расположены на вкладке «Главная» в группе «Буфер обмена».

С другими режимами просмотра можно познакомиться, используя на вкладке «Вид» кнопки групп «Режимы просмотра презентации» и «Режимы образцов».

Основным режимом демонстрации является *режим показа слайдов*, который включается на вкладке «Показ слайдов» в группе «Начать показ слайдов» кнопкой **С начала** (либо клавишей **F5** клавиатуры) или **С текущего слайда** (либо сочетанием **Shift+F5**).

Чтобы переход от слайда к слайду производился щелчком мыши или нажатием клавиши **Пробел** клавиатуры, на вкладке «Показ слайдов» в группе «Настройка» нужно снять флажок возле пункта **Использовать время показа слайдов**.

Динамику показу презентации добавляют эффекты при смене слайдов, которые в PowerPoint называют *переходами*. Чтобы добавить переход, слайд выделяют в области «Структура и слайды» и на вкладке «Переходы» в группе «Переход к этому слайду» выбирают вид перехода. Кнопкой **Просмотр** на вкладке «Переходы» эффект перехода можно просмотреть. Желательно щёлкнуть по кнопке **Применить ко всем** в группе «Время показа слайдов», чтобы назначить этот переход всем слайдам презентации.

Настройка анимации. В некоторых случаях в презентацию полезно включить анимацию для объектов на слайде.

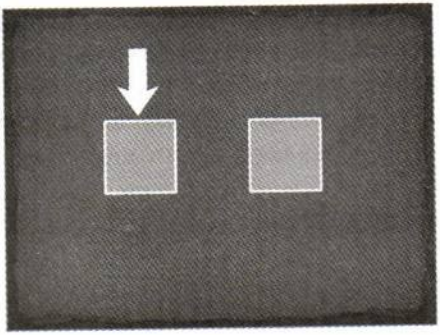


Рис. 151

■ Упражнение 153

Создадим слайд с анимацией, на котором находятся два квадрата. На слайде должна появиться стрелка, указать на один квадрат, затем на другой и вылететь за пределы слайда.

На вкладке «Дизайн» выбираем тему «Бумажная». На вкладке «Главная» в группе «Слайды» в меню кнопки **Макет** выбираем макет «Пустой слайд» и рисуем на нём два квадрата и стрелку (рис. 151). На вкладке «Анимация» в группе «Расширенная анимация» щёлкаем по кнопке **Область анимации**. Область появляется на экране.

Стрелку на слайде выделяем щелчком мыши. На вкладке «Анимация» в группе «Анимация» щелчком по нижней кнопке правее поля с эффектами раскрываем это поле. В разделе «Вход» поля выбираем эффект под названием **Вылет**.

В той же группе «Анимация» щёлкаем по кнопке **Параметры эффектов** и выбираем направление **Сверху**. Результат выбора проверяем щелчком по кнопке **Просмотр** на вкладке слева.

Опять щёлкаем по стрелке, в группе «Расширенная анимация» щёлкаем по кнопке **Добавить анимацию** и при помощи полосы прокрутки смещаемся в нижнюю часть меню. В разделе «Пути перемещения» выбираем пункт **Линии**. Щёлкаем по кнопке **Параметры эффектов** и выбираем направление **Вправо**. Щёлкаем по новой стрелке и удлиняем её вправо (за маркер) до середины второго квадрата. Эффект проверяем щелчком по кнопке **Просмотр**.

Осталось задать эффект выхода стрелки со слайда. Щёлкаем по стрелке, а затем по кнопке **Добавить анимацию** и в её меню в разде-

ле «Выход» выбираем пункт **Вылет за край листа**. Щёлкаем по кнопке **Параметры эффектов** и выбираем направление **Вверх**. Результат проверяем щелчком по кнопке **Просмотр**. В области анимации разместился список использованных эффектов.

В заключение запускаем демонстрацию (**F5**). Используя щелчки мышью, убеждаемся, что задание выполнено.

Создание презентации к докладу. С процессом создания презентации подробнее познакомимся на примере презентации к учебному докладу.

■ Упражнение 154

Создадим презентацию к учебному докладу на тему «Знакомство с программой PowerPoint» по материалам данного параграфа учебника.

Сначала создадим *план* (*сценарий*) *презентации* следующего содержания.

1. *Слайд 1.* Название презентации «Знакомство с программой PowerPoint 2010».
2. *Слайд 2.* Текст первого абзаца параграфа.
3. *Слайд 3.* Текст второго абзаца параграфа.
4. *Слайд 4.* Текст двух первых предложений третьего абзаца параграфа.
5. *Слайд 5.* Текст четвёртого предложения третьего абзаца параграфа.
6. *Слайд 6.* Заголовок «Общий вид окна PowerPoint 2010».
7. *Слайд 7.* На фоне окна программы стрелкой указать в области просмотра на основные рабочие области (сопровождая это текстовыми пояснениями).
8. — 9. *Слайды 8—9.* На фоне окна программы показать последовательно выбор на вкладке «Дизайн» темы оформления и выбор на складке «Главная» макета слайда (сопровождая это текстовыми пояснениями).
10. *Слайд 10.* Слова благодарности за внимание.

Запускаем PowerPoint 2010 и выбираем дизайн для всей презентации (тему оформления «Трек» на вкладке «Дизайн»).

Теперь создадим *слайды презентации*.

1. *Слайд 1.* На вкладке «Главная» в группе «Слайды» кнопкой **Макет** выбираем для слайда макет «Титульный слайд». Рамка подзаголовка нам не потребуется: выделяем её и удаляем. Щёлкаем по наполнителю «Заголовок слайда», вводим название презентации, а в группе «Абзац» кнопкой **Выровнять по центру** устанавливаем соответствующее выравнивание.

2. *Слайд 2.* На вкладке «Главная» щёлкаем по кнопке **Создать слайд**. Для нового слайда выбираем макет «Заголовок и объект». Рамку заголовка удаляем, а в рамку объекта вводим запланированный текст, выделяем и подбираем размер шрифта. При необходимости рамку можно увеличить и переместить.

3. *Слайд 3.* Добавляем новый слайд. В рамку объекта вводим запланированный текст и подбираем размер шрифта.

4. *Слайд 4.* Добавляем новый слайд. В рамку объекта вводим запланированный текст и подбираем размер шрифта.

5. *Слайд 5.* Добавляем новый слайд. В рамку объекта вводим запланированный текст и подбираем размер шрифта.

6. *Слайд 6.* Добавляем новый слайд. Задаём макет «Титульный слайд». Вводим текст заголовка и центрируем его.

7. *Слайд 7.* Добавляем новый слайд. Задаём макет «Пустой слайд» и готовим изображение окна PowerPoint. Для этого командой меню **Файл|Создать** открываем в меню «Файл» вкладку «Создать», в которой по умолчанию выделен шаблон «Новая презентация». Выбор подтверждаем щелчком по кнопке **Создать** в правой части вкладки. Открывается ещё одно окно PowerPoint. Открываем в этом окне область анимации.

Далее возвращаемся в окно PowerPoint с анимацией на вкладку «Вставка» в группу «Изображения». Щелчок по кнопке **Снимок** открывает меню, в котором в разделе «Доступные окна» выбираем подготовленное окно PowerPoint. В результате цифровой снимок окна PowerPoint вставляется на слайд.

С помощью инструмента **Фигуры** в группе «Рисование» на вкладке «Главная» выводим на слайд стрелки и надписи, как показано на рисунке 152. Осталось добавить эффекты анимации. На вкладке «Анимация» в области «Расширенная анимация» щёлкаем по кнопке **Область анимации**. Область анимации открывается.

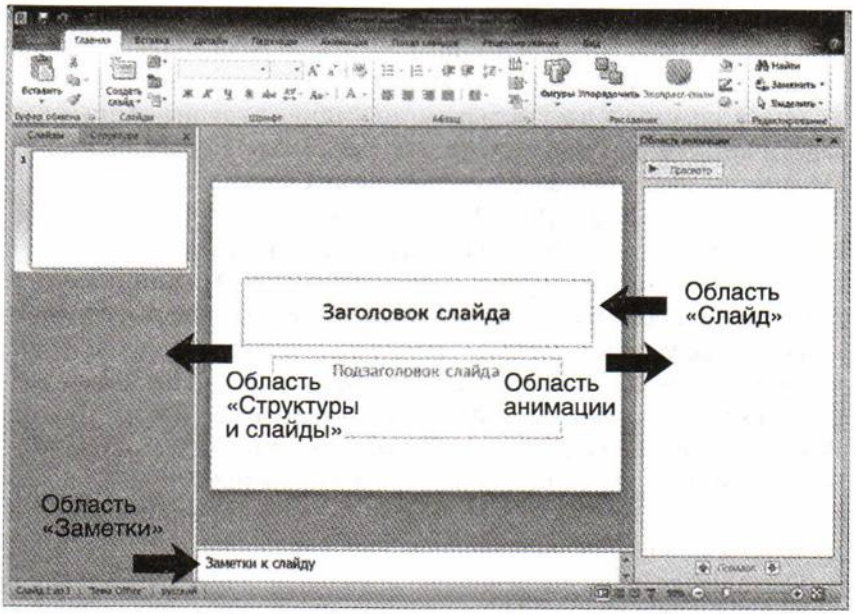


Рис. 152

Выделяем правую верхнюю стрелку и задаём анимацию **Появление** (в разделе «Вход»), а далее **Параметры эффектов|Слева**.

Выделяем надпись к этой стрелке и задаём ту же анимацию. В разделе «Время показа слайдов» в меню кнопки **Начало:** выбираем пункт **После предыдущего**.

Снова выделяем ту же стрелку и кнопкой **Добавить анимацию** в разделе «Выход» задаём анимацию **Выцветание**.

Повторно выделяем надпись к стрелке, добавляем ту же анимацию, что для стрелки, плюс эффект **Начало:|С предыдущим**.

Аналогично поступаем и с остальными парами «стрелка-надпись».

8 *Слайд 8.* Добавляем новый слайд с разметкой «Пустой слайд». В окне PowerPoint, предназначенном для создания цифровых снимков, открываем вкладку «Дизайн». Переходим в окно PowerPoint, в котором создаётся презентация, и делаем снимок.

Выводим на слайд большую стрелку «Вверх», которая указывает на группу «Темы» вкладки «Дизайн».

Щелчком правой клавиши мыши по стрелке открываем контекстное меню, в котором выбираем пункт **Изменить текст**. Внутри стрелки появляется текстовый курсор. Вводим текст «Темы».

Далее для стрелки на вкладке «Анимация» в группе «Анимация» задаём анимацию **Появление** (в разделе «Вход»), а в меню кнопки **Параметры эффектов** задаём направление **Сверху**.

Переходим в окно PowerPoint, предназначенное для создания снимков. Сделать снимок окна с раскрытым меню PowerPoint не позволяет, поэтому воспользуемся программой «Ножницы», которая запускается из меню кнопки **Пуск** (папка «Стандартные»).

После запуска программы «Ножницы» изображение на экране мутнеет и появляется небольшое окно программы. Нажимаем клавишу **Esc** клавиатуры — изображение становится чётким. В окне PowerPoint, предназначенном для снимков, раскрываем поле с набором тем. С клавиатуры вводим сочетание **Ctrl+PrtScr** и на «помутневшем» экране выделяем требуемое поле прямоугольником, как в графическом редакторе.

Окно программы «Ножницы» переходит в режим отображения вырезанного фрагмента. Но мы это окно закрываем, так как фрагмент находится и в буфере обмена.

Переходим в окно PowerPoint, в котором создаётся презентация, и вставляем на слайд фрагмент из буфера обмена. Изменяем размеры фрагмента и его положение на слайде так, чтобы меню выглядело раскрытым в окне PowerPoint. Выделяем фрагмент и задаём для него анимацию **Появление** (в разделе «Вход»), а в меню кнопки **Параметры эффектов** задаём направление **Сверху**.

Правее фрагмента выводим на слайд большую стрелку «Влево», которая указывает на меню. Вводим в неё текст «Набор тем для выбора». Задаём для него анимацию **Появление** (в разделе «Вход»), а в меню кнопки **Параметры эффектов** задаём направление **Слева**.

9 *Слайд 9.* Добавляем новый слайд. Аналогично предыдущему слайду вставляем на нём окна с вкладкой «Главная», стрелку, направленную на группу «Слайды» с текстом «Кнопка «Макет», открытое меню с набором макетов и стрелку с текстом «Набор макетов для выбора».

10 *Слайд 10.* Добавляем новый слайд с макетом «Титульный слайд». В рамку заголовка вводим текст «Спасибо за внимание!» и центрируем его.

Презентация готова.

☐ Вопросы и задания

1. Что такое компьютерная презентация?

2. Каких рекомендаций по оформлению слайдов следует придерживаться?

3. Чем компьютерная анимация в презентации отличается от традиционной формы анимации?

4. Создайте компьютерную презентацию к любому параграфу учебника.

5. Коллективными усилиями, поделив тематику, создайте по материалам Интернета презентацию, посвящённую истории создания персональных компьютеров.

Вместо заключения

Эти строки завершают текст учебника «Информатика и ИКТ», оставляя вас один на один с будущими информационными и техническими новшествами в окружающем нас динамичном и постоянно меняющемся мире. Состоявшееся знакомство с достижениями в области информатики и информационно-коммуникационных технологий ставило своей целью не только научить вас пользоваться существующими теоретическими подходами, информационными и программными средствами, компьютерными и сетевыми комплексами. Главное — в осознании того, что на базе изученного вы сможете в будущем самостоятельно познавать и использовать новые системные принципы, технические и программные средства, которые сегодня, возможно, ещё только разработаны или находятся в стадии замысла. Верьте в собственные возможности, никогда не останавливайтесь в стремлении расширить свои познания — и успех вам гарантирован!

Оглавление

Глава 5. Табличные модели и электронные таблицы

Глава 6. Базы данных

Глава 7. Мультимедийные технологии