

ИНФОРМАТИКА

Учебное пособие для 8 класса
общеобразовательных учреждений
с русским языком обучения

*Допущено
Министерством образования
Республики Беларусь*

Минск
«Народная асвета»
2010

Правообладатель Народная асвета

УДК 004(075.3=161.1)

ББК 32.81я721

И74

Авторы:

Е. Л. Миняйлова (главы 2, 3, 4, 5), Д. А. Вербовиков (главы 2, 3, 4, 5),
Н. Р. Коледа (главы 2, 3, 4, 5), Н. В. Якунина (глава 1)

Рецензенты:

кафедра математического обеспечения АСУ
Белорусского государственного университета
(доцент кафедры, кандидат физ.-мат. наук, доцент С. И. Кашикевич);
учитель информатики высшей категории
СШ № 209 г. Минска Л. И. Мосткова

ISBN 978-985-03-1342-3

© Оформление. УП «Народная асвета», 2010

Правообладатель Народная асвета

ОТ АВТОРОВ

Уважаемый читатель!

Развитие современного общества связано с широкой информатизацией всех его сфер. Это требует от людей разных специальностей высокого уровня владения современными информационными технологиями. Данная книга поможет Вам научиться применять информационные компьютерные технологии, освоить основы программирования.

Учебное пособие состоит из 5 глав.

В первой главе «Основы алгоритмизации и программирования» рассматриваются основные алгоритмические структуры и операторы языка программирования Pascal. Большое количество рассматриваемых задач из различных предметных областей помогут Вам в достаточной мере освоить азы программирования.

Во второй главе «Технология обработки текстовых документов» Вы продолжите изучение методов обработки текстовых документов.

Третья глава «Работа с векторной графикой» познакомит Вас с графическим редактором CorelDRAW.

В четвертой главе «Вредоносное программное обеспечение и защита информации» обсуждаются важные для каждого пользователя компьютеров вопросы информационной безопасности.

Материал главы 5 «Работа с электронной почтой» описывает принципы работы с электронной почтой. В главе 5 рассматриваются также правила поведения, общения в Сети, оформления электронных писем, меры безопасности при переписке.

Материал, отмеченный знаком , предназначен для любознательных учеников и не требует обязательного изучения.

Восклицательным знаком ! отмечены места в тексте, на которые надо обратить особое внимание.

Вопросы после параграфов предназначены для закрепления изученного материала и помечены знаком ?.

В конце параграфов предлагаются упражнения для практической работы.

Сведения, приводимые в пособии, помогут сделать Вашу работу на компьютере более интересной и эффективной.

Желаем успехов!

Глава 1

ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

§ 1. Алгоритмическая конструкция ВЕТВЛЕНИЕ

1.1. Принятие решений в программе

Как известно, операторы ввода и вывода, оператор присваивания позволяют составлять *линейные* программы, в которых все команды выполняются последовательно, одна за другой. Но в жизни часто приходится принимать решения в зависимости от некоторых условий. Например, если сделаны уроки, можно отдохнуть; если холодно, необходимо теплее одеться; если в театральной кассе имеются билеты, можно пойти на спектакль, иначе можно просто погулять в парке. Принимая решения, человек анализирует ситуацию. В программировании также используются команды, которые позволяют компьютеру принимать решения в зависимости от некоторого условия. Одна из таких команд реализуется с помощью условного оператора *if* языка программирования Pascal. В общем виде этот оператор выглядит так:

```
if <условие> then <оператор _ 1>
else <оператор _ 2>;
```

Оператор *if* анализирует некоторое условие. Если условие верно (истинно), то выполняется оператор *_ 1*, иначе (если условие неверно (ложно)) выполняется оператор *_ 2*. Оператор *_ 1* и оператор *_ 2* называются **ветвями программы**.

Рассмотрим работу оператора *if* на примере.

Пример 1. Составить программу, которая запрашивает ответ пользователя на вопрос «Сколько будет $3 \times 5?$ », сравнивает введенное значение с числом 15 и

выводит соответствующее сообщение: «Верно» либо «Неверно».

Программа может быть такой:

```
program proverka;
var a: integer;
begin
    writeln('Сколько будет 3x5?');
    readln(a);
    {Ответ вводится с клавиатуры и записывается в переменную a}
    if a=15 then writeln('Верно')
        else writeln ('Неверно');
    {В операторе if сравнивается значение переменной a с числом 15. Если a = 15, то выводится сообщение 'Верно', в противном случае - 'Неверно'}
end.
```

Результаты выполнения программы при вводе чисел 15 и 25:

Сколько будет 3x5?
15
Верно

Сколько будет 3x5?
25
Неверно

1.2. Простые условия

Условие — это выражение, стоящее в операторе **if..then..else** после слова **if**. В зависимости от результатов проверки этого условия (его истинности либо ложности) выполняется та либо иная ветвь программы. В блок-схемах оператор **if** обозначают ромбом, называемым **блоком проверки условия**. Алгоритмическая конструкция ветвления представлена на рисунке 1.1.

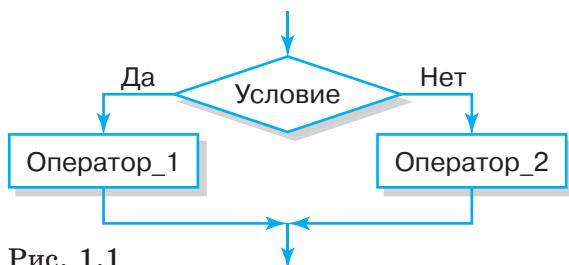


Рис. 1.1

На рисунке 1.2 показана блок-схема алгоритма, реализованного в примере 1.

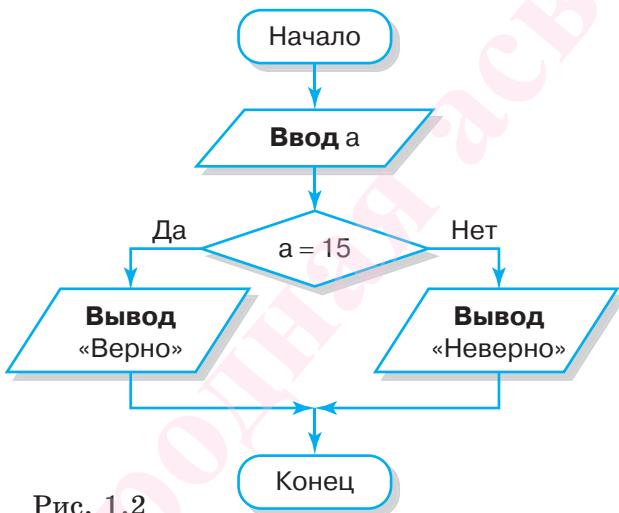


Рис. 1.2

Простое условие — это сравнение двух выражений по величине. Операции сравнения в языке Pascal записываются с помощью следующих знаков: $=$ (равно), $<$ (меньше), \leq (меньше либо равно), $>$ (больше), \geq (больше либо равно), \neq (не равно).

Примеры записи простых условий:

$$\begin{array}{lll} a \neq b, & a > b, & a > 0, \\ a \leq 0, & b > 6, & A < 0, \\ a + 3 * c \geq 20, & a + c = 100, & 5 * c \geq 80. \end{array}$$

В качестве оператора `_1` и оператора `_2` могут использоваться любые команды.

! Важно помнить: нельзя ставить знак «;» перед словом `else`.

1.3. Составные условия

При решении задач иногда возникает необходимость проверять выполнение двух (например, $0 < a < 5$) и более простых условий. Такие условия называют **составными**. Составные условия строятся из простых с помощью следующих логических операций:

- **and** — логическое «и»;
- **or** — логическое «или»;
- **not** — логическое отрицание.

Примеры записи составных условий:

`(a>0) and (a<5), (x>10) or (y<100).`

! Простые условия обязательно заключаются в круглые скобки, так как логические операции имеют приоритет перед операциями сравнения.

Результаты выполнения логических операций приведены в следующей таблице:

Условие A	Операция	Условие B	Результат
Истинно	and	Истинно	Истинно
Ложно	and	Истинно	Ложно
Истинно	and	Ложно	Ложно
Ложно	and	Ложно	Ложно
Истинно	or	Истинно	Истинно
Истинно	or	Ложно	Истинно
Ложно	or	Истинно	Истинно
Ложно	or	Ложно	Ложно
—	not	Истинно	Ложно
—	not	Ложно	Истинно

Так, результаты использования логических операций в следующих составных условиях будут такими:

($2>5$) and ($2>3$) — ложно;
($1<5$) and ($1>0$) — истинно;
($6>9$) and ($7>5$) — ложно;

($3>2$) or ($3<1$) — истинно;
($3>2$) or ($2>0$) — истинно;
($1>2$) or ($1<0$) — ложно;

not ($5>6$) — истинно;
not ($6>5$) — ложно.

Рассмотрим на примерах использование составных условий.

Пример 2. Составить программу, которая определяет, является ли введенное целое число двузначным, и выводит на экран соответствующее сообщение: «Число двузначное» либо «Число не двузначное».

Программа может быть такой:

```
program c2;
var a: integer;
begin
  write('Введите целое число: ');
  readln(a);
  if (a>=10) and (a<=99)
    then writeln('Число двузначное')
  else writeln('Число не двузначное');
  {Если (a>=10) и (a<=99), то выводится
  сообщение 'Число двузначное', иначе -
  'Число не двузначное'}
end.
```

Результаты выполнения программы при вводе чисел 45 и 125:

Ведите целое число: 45
Число двузначное

Ведите целое число: 125
Число не двузначное

1.4. Полная и сокращенная формы условного оператора

Запись `if..then..else` называется **полной формой** условного оператора. Она позволяет выполнять программу по одной из двух ветвей. Такую алгоритмическую конструкцию называют **ветвлением** (по аналогии с развиликой у дороги). В языке Pascal имеется также **сокращенная форма** условного оператора. Она имеет следующий вид:

If <условие> **then** <оператор>;



Рис. 1.3

Оператор выполняется, только если условие верно. Блок-схема сокращенной формы условного оператора показана на рисунке 1.3.

Рассмотрим примеры использования сокращенной формы условного оператора.

Пример 3. Составить программу, которая выводит на экран результат целочисленного деления заданного целого числа на 2, если оно четное.

Алгоритм решения этой задачи в виде блок-схемы представлен на рисунке 1.4.

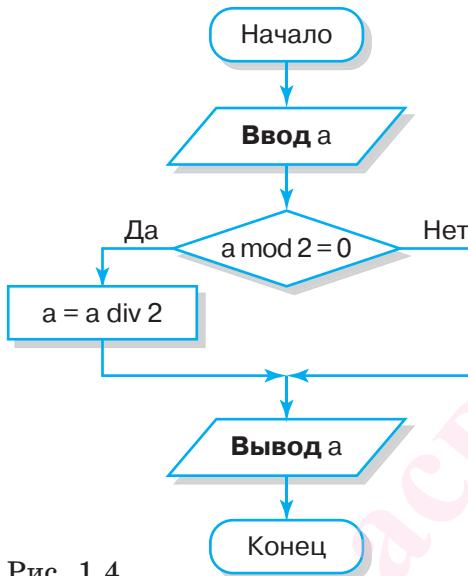


Рис. 1.4

Программа, реализующая данный алгоритм, может быть такой:

```

program c2_2;
var a: integer;
begin
  write('Введите целое число: ');
  readln(a);
  if (a mod 2=0) then a:=a div 2;
  write(a);
end.
  
```

Результаты выполнения программы при вводе чисел 16 и 13:

Введите целое число: 16
8

Введите целое число: 13
13

Пример 4. Составить программу, которая запрашивает два целых числа, и если каждое из них отлично от нуля, то выводит на экран их среднее арифметическое.

Программа может быть такой:

```
program sr_arifm;
var a, b: integer;
begin
  write('Введите два целых числа: ');
  readln(a, b);
  if (a<>0) and (b<>0) then writeln((a+b)/2);
end.
```

Результаты выполнения программы для чисел 5 и 7, а также для чисел 0 и 10:

Введите два целых числа: 5 7
6

Введите два целых числа: 0 10

1.5. Составной оператор

Если в операторе **if** при выполнении либо невыполнении условия необходимо исполнить несколько действий, их объединяют в группу. Операторы такой группы записывают между зарезервированными словами **begin** и **end**. Например:

```
if x>0 then begin x:=x*x; write(x) end;
```

Группа операторов (**x:=x*x;** **write(x)**), записанная между словами **begin** и **end**, называется **составным оператором**, а зарезервированные слова **begin** и **end** — открыющей и закрывающей **операторны-**

ми скобками. В общем виде составной оператор может выглядеть так:

```
begin
    Оператор _ 1;
    Оператор _ 2;
    ...
end;
```

Рассмотрим примеры использования составного оператора.

Пример 5. Составить программу, которая запрашивает целое число, не равное нулю, сообщает, положительное оно или отрицательное, и при этом положительное число уменьшает в 2 раза, а отрицательное возводит в квадрат.

Алгоритм решения задачи в виде блок-схемы представлен на рисунке 1.5.

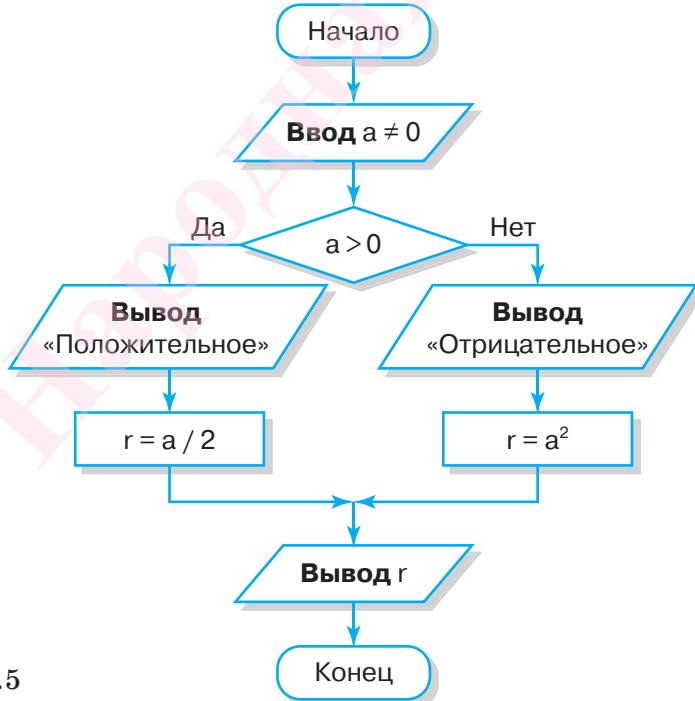


Рис. 1.5

Программа, реализующая данный алгоритм, может быть такой:

```
program chislo;
var a: integer; r: real;
begin
    write('Введите целое число, не равное
        нулю: '); readln(a);
    if a>0 then
        begin
            r:=a/2; {Число уменьшается в 2 раза}
            writeln('Положительное. r = ', r);
        end
        else
        begin
            r:=sqr(a); {Число возводится в квадрат}
            writeln('Отрицательное. r = ', r);
        end;
    end.
```

Результаты выполнения программы для чисел –5 и 8:

Введите целое число, не равное нулю: –5
Отрицательное. r = 25

Введите целое число, не равное нулю: 8
Положительное. r = 4

! При записи составных операторов необходимо соблюдать правило вложенности операторных скобок: каждое слово **begin** «закрывается» словом **end**.

Пример 6. Составить программу, которая выводит на экран число десятков (первую цифру) и число единиц (вторую цифру) в заданном числе, если оно

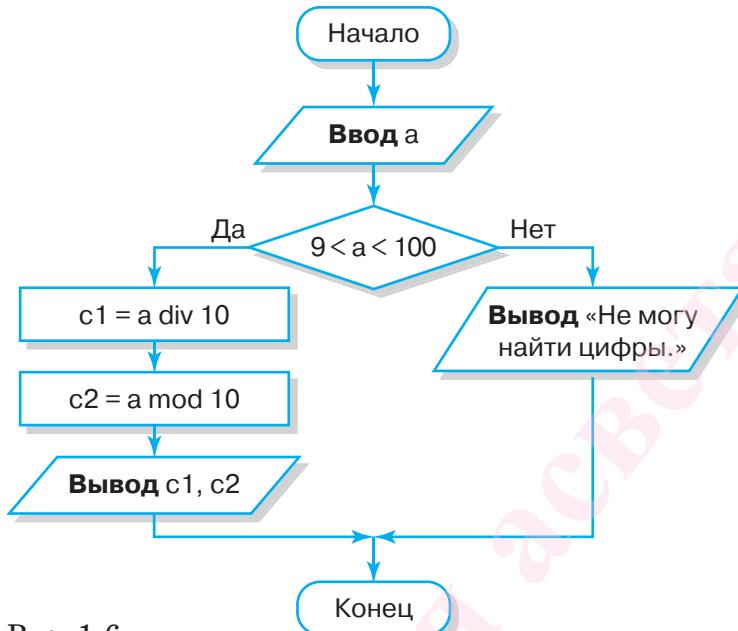


Рис. 1.6

двузначное. В противном случае — сообщение «Не могу найти цифры».

Алгоритм решения этой задачи в виде блок-схемы представлен на рисунке 1.6.

Программа, реализующая данный алгоритм, может быть такой:

```

program cifry;
var a, c1, c2: integer;
begin
  write('Введите целое число: ');
  readln(a);
  if (a>9) and (a<100) then
    begin
      c1:=a div 10;           {Число десятков}
      c2:=a mod 10;           {Число единиц}
      writeln(c1,' - число десятков.');
    end
end.
  
```

```

write(c2,' – число единиц.');
end
else
    write('Не могу найти цифры.');
end.

```

Результаты выполнения программы для чисел 27 и 345:

Ведите целое число: 27
 2 – число десятков.
 7 – число единиц.

Ведите целое число: 345
 Не могу найти цифры.

- ? 1.** В каких случаях в программе используется условный оператор `if`?
- 2.** Как записывается в общем виде полная форма условного оператора `if`?
- 3.** Как записывается в общем виде сокращенная форма условного оператора `if`?
- 4.** Какие операции сравнения используют для записи простых условий?
- 5.** Что такое составной оператор?

Упражнения

- 1.** Составьте программу, которая задает вопрос: «В каком году был основан город Минск (например, в 1067 г., в 993 г. либо в 1147 г.)?» В случае верного ответа программа должна сообщить «Верно», в противном случае — «Вы ошиблись».
- 2.** Составьте программу, которая определяет, является ли введенное целое число нечетным, и выводит на экран соответствующее сообщение: «Да» либо «Нет».

3. Составьте программу, которая возводит в квадрат введенное целое число, если оно отрицательное, и выводит на экран результат.

4. Составьте программу, которая выводит на экран пример на умножение двух целых чисел, запрашивает ответ пользователя, проверяет его и выводит соответствующее сообщение: «Правильно!» либо «Вы ошиблись», а также правильный результат.

5. Составьте программу, которая проверяет, является ли человек заданного возраста школьником, если возраст школьника — от 6 до 17 лет, и выводит на экран соответствующее сообщение: «Является» либо «Не является».

6. Составьте программу, которая проверяет, равны ли 3 введенных целых числа, и выводит на экран соответствующее сообщение: «Равны» либо «Не равны».

§ 2. Составление и реализация алгоритмов с ветвлением

Напомним, что ветвление — алгоритмическая конструкция, которая позволяет выполнять программу по одной из двух ветвей. Алгоритмы, содержащие конструкцию ветвления, называют **алгоритмами с ветвлением**. Составляя алгоритмы с ветвлением, можно решать задачи, требующие анализа создавшихся условий и выбора различных вариантов действия. Например, с помощью алгоритма с ветвлением можно найти большее из двух либо из трех и более заданных чисел.

Пример 1. Составить программу, которая находит большее из двух заданных различных целых чисел и выводит его на экран.

Программа может быть такой:

```
program bid;
var a, b: integer;
begin
    writeln('Введите два разных числа:');
    write('a = '); readln(a);
    write('b = '); readln(b);
    if a>b then writeln('Большее число = ', a)
        else writeln('Большее число = ', b);
end.
```

Результаты выполнения программы для чисел 15 и 12, а также для чисел 11 и 18:

```
Введите два разных числа:
a = 15
b = 12
Большее число = 15
```

```
Введите два разных числа:
a = 11
b = 18
Большее число = 18
```

В первом случае программа выполнила оператор, принадлежащий ветви `then`, так как условие `a>b` истинно. Во втором случае программа выполнила оператор, принадлежащий ветви `else`, так как условие `a>b` ложно.

Пример 2. Составить программу, которая определяет по введенному номеру месяца пору года и выводит на экран ее название.

Программа может быть такой:

```
program seasons;
var nm: integer;
```

```
begin
    write('Введите номер месяца: ');
    readln(nm);
    if (nm>=3) and (nm<=5) then writeln('Весна');
    if (nm>=6) and (nm<=8) then writeln('Лето');
    if (nm>=9) and (nm<=11) then writeln('Осень');
    if (nm=12) or (nm<=2) then writeln('Зима');
end.
```

Результат выполнения программы при вводе номера месяца 7:

```
Введите номер месяца: 7
Лето
```

В зависимости от введенного номера месяца только для одного из четырех операторов **if** значение условия будет истинным.

Вспомним графические возможности языка программирования Pascal.

Пример 3. Составить программу, которая проверяет, можно ли поместить круг радиуса r в квадрат со стороной a . Результат подтвердить рисунком, нарисовав в графическом окне круг данного радиуса в квадрате со стороной данной длины.

Программа может быть такой:

```
program krug_v_kvadr;
uses graphabc;
var a, a2, r: integer;
begin
    write('Введите длину стороны квадрата: ');
    readln(a);
    write('Введите радиус круга: ');
    readln(r);
```

```

if a>=2*r then
    writeln('Круг можно поместить в квадрат')
    else
        writeln('Круг нельзя поместить в квад-
        рат');
    SetWindowSize(640, 480);
    SetBrushStyle(bsClear);
    SetPenWidth(5);
    SetPenColor(clRed);
    Circle(320, 240, r);
    SetPenColor(clGreen); a2:=a div 2;
    Rectangle(320-a2, 240-a2, 320+a2, 240+a2);
end.

```

Результат выполнения программы при $a = 300$ и $r = 100$ показан на рисунке 1.7.

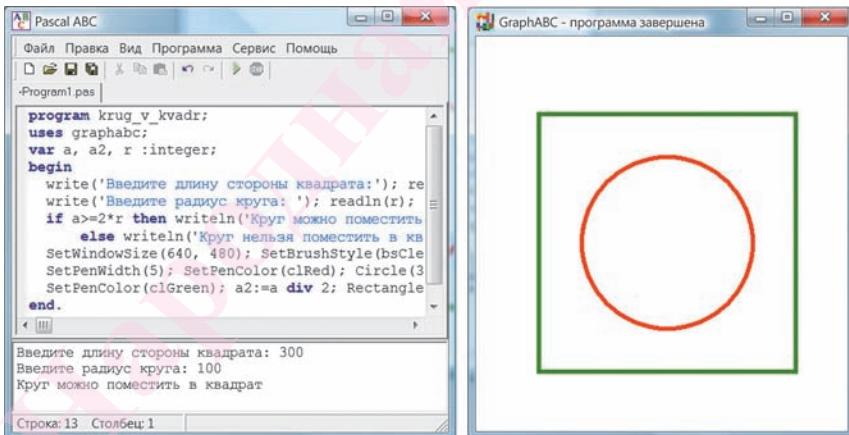


Рис. 1.7

Рассмотрим задачу, которая демонстрирует вложенность одного оператора **if** в другой оператор **if**.

Пример 4. Составить программу, которая по заданным коэффициентам a и b исследует корни уравнения $ax + b = 0$ и выводит на экран значение вычис-

ленного корня или соответствующее сообщение: «Корень — любое число» либо «Корней нет».

Известно, что решение данной задачи зависит от значений a и b . Если $a \neq 0$, то $x = -b/a$. Если же $a = 0$, то необходимо проверить, равен ли коэффициент b нулю. При $b = 0$ корнем является любое число, в противном случае корней нет.

Программа может быть такой:

```
program lin_ur;
var a, b: integer; x: real;
begin
    write('a = '); readln(a);
    write('b = '); readln(b);
    if a<>0 then writeln('x = ', -b/a)
    else if b=0 then
        writeln('Корень - любое число.')
    else
        writeln('Корней нет.');
end.
```

Результаты выполнения программы при разных значениях a и b :

a = 0
b = 0
Корень — любое число.

a = 5
b = 20
x = -4

a = 0
b = 4
Корней нет.

Из рассмотренных примеров видно, что в качестве оператора _ 1 и оператора _ 2 может быть простой либо составной оператор, в том числе и оператор if. Обратите внимание, что в программе примера 4 слово else в записи

else writeln('Корней нет.')

относится к последнему оператору if.

Рассмотрим еще один пример, демонстрирующий вложенность оператора if.

Пример 5. Составить программу, которая запрашивает значения роста (в сантиметрах) трех учеников и выводит на экран рост самого высокого.

Программа может быть такой:

```
program max_3;
var a, b, c: integer;
begin
  writeln('Введите различные числа — рост
3 учеников: ');
  readln(a, b, c);
  if (a>b) and (a>c)
    then writeln('Рост самого высокого
ученика = ',a)
  else
    if b>c
      then writeln('Рост самого высокого
ученика = ',b)
    else writeln('Рост самого высокого
ученика = ',c);
end.
```

Результат выполнения программы при $a = 145$, $b = 160$, $c = 154$:

Введите различные числа — рост 3 учеников:

145 160 154

Рост самого высокого ученика = 160

Следующую задачу можно решить с помощью вложенного оператора if. Но есть и более простой способ решения — с помощью составного условия.

Пример 6. Составить программу, которая определяет, является ли указанный год високосным, и

выводит на экран соответствующее сообщение: «Високосный» либо «Невисокосный».

Год считается високосным, если его номер делится без остатка на 4, кроме случаев, когда он делится на 100. Если номер года делится на 400, то он все равно високосный. Например, високосными являются 48 г., 1968 г., 1600 г., 2000 г., а невисокосными — 159 г., 1800 г., 1900 г.

Программа может быть такой:

```
program vis _ god;
var a: integer;
begin
  write('Введите год: ');
  readln(a);
  if (a mod 400=0) or ((a mod 100<>0)
and (a mod 4=0))
    then writeln('Високосный')
    else writeln('Невисокосный');
end.
```

Результаты выполнения программы для 1900 г. и 2008 г.:

Введите год: 1900
Невисокосный

Введите год: 2008
Високосный

! Точка с запятой слева и справа от **then** и от **else** не ставится.

Чтобы текст программы был более понятен, вложенные операторы оформляют со сдвигом вправо. При каждом следующем вложении операторы сдвигают на несколько позиций вправо.

При наборе текста программы рекомендуется сразу после слова **begin** писать слово **end**, затем встав-

лять между ними операторы. Это позволяет избегать появления непарных `begin` или `end`. Аналогично поступают при наборе апострофов и скобок.

Упражнения

1. Составьте программу, которая проверяет, делится ли на 3 введенное целое число, и выводит на экран соответствующее сообщение: «Делится» либо «Не делится».
2. Составьте программу, которая сообщает, можно ли разделить поровну между двумя друзьями n конфет, выводя на экран соответствующее сообщение: «Да» либо «Нет». Нарисуйте блок-схему алгоритма решения этой задачи.
3. Составьте программу, которая запрашивает три целых числа i , если ни одно из чисел не равно нулю, выводит на экран их среднее арифметическое. Нарисуйте блок-схему алгоритма решения этой задачи.
4. Составьте программу, которая определяет, принадлежит ли введенное целое число x интервалу $(10, 20)$, и выводит на экран соответствующее сообщение: «Принадлежит» либо «Не принадлежит».
5. В компьютер поступают результаты соревнований по плаванию (количество минут и секунд) для трех спортсменов. Составьте программу, которая выводит на экран лучший результат.
6. Составьте программу, которая выводит на экран число сотен в заданном числе, если оно трехзначное, а в противном случае сообщает: «Число не трехзначное».
7. Составьте программу, которая по заданному a исследует корни уравнения $ax = 5$ и выводит на экран значение вычисленного корня либо сообщение «Корней нет».

8. Составьте программу, которая проверяет, можно ли в данном круге радиуса r вырезать круг радиуса a . Если можно, то рисует в графическом окне круг радиуса a в круге радиуса r , иначе — выводит на экран сообщение «Нельзя».

9. Составьте программу, которая определяет, является ли треугольник с заданными длинами сторон равносторонним, равнобедренным или разносторонним, и выводит на экран соответствующее сообщение: «Равносторонний», «Равнобедренный» либо «Разносторонний». Предполагается, что треугольник с заданными сторонами существует.

10. Для продажи книг в магазине используется компьютер. Составьте программу, которая по стоимости книги и сумме денег, внесенной покупателем, выводит на экран сумму причитающейся сдачи, если денег внесено больше. Если сдачи не требуется, программа выводит на экран «Спасибо!».

§ 3. Алгоритмическая конструкция ПОВТОРЕНИЕ

3.1. Цикл (повторение)

Программы, рассмотренные в предыдущих параграфах, обладают одним общим свойством: при их выполнении действие или группа действий совершается один раз либо вообще не совершается. В жизни часто встречаются случаи, в которых много раз подряд нужно выполнять один и тот же набор действий: перелистывать страницы, пока не закончится книга, шагать по ступенькам, пока не поднимешься на нужный этаж, и т. д. В программах также иногда необходимо повторять определенные действия. Для

этого используется новая форма организации действий — цикл (повторение).

Цикл является одной из базовых алгоритмических конструкций и представляет собой последовательность действий, которая выполняется неоднократно до тех пор, пока выполняется некоторое условие. Саму последовательность повторяющихся действий называют **телом цикла**.

Циклы позволяют записывать действия в компактной форме. Например, для вычисления суммы десяти первых натуральных чисел можно выбрать простое решение:

$$\text{Sum} := 1+2+3+4+5+6+7+8+9+10;$$

А что делать, если число слагаемых достаточно велико, например 1000 или 10 000? Представьте себе программу с записью оператора присваивания, который занимает несколько десятков страниц! Создание такой программы нерационально.

Рассмотрим по шагам алгоритм вычисления суммы пяти первых натуральных чисел. В переменной Sum будем накапливать сумму чисел, в переменной i будем записывать очередное натуральное число (слагаемое).

0-й шаг: $i=0$

 $\text{Sum}_0 = 0$

1-й шаг: $i=1 (i=i+1=0+1)$

 $\text{Sum}_1 = \text{Sum}_0 + i = 0+1=1$

2-й шаг: $i=2 (i=i+1=1+1)$

 $\text{Sum}_2 = \text{Sum}_1 + i = 1+2=3$

3-й шаг: $i=3 (i=i+1=2+1)$

 $\text{Sum}_3 = \text{Sum}_2 + i = 3+3=6$

4-й шаг: $i=4 (i=i+1=3+1)$

 $\text{Sum}_4 = \text{Sum}_3 + i = 6+4=10$

5-й шаг: $i=5 (i=i+1=4+1)$

 $\text{Sum}_5 = \text{Sum}_4 + i = 10+5=15$

Нетрудно заметить, что при вычислении суммы в определенном порядке повторяются всего две операции:

1. Прибавить слагаемое к ранее полученной сумме.
2. Увеличить на 1 значение слагаемого.

Следовательно, задачу можно решить с помощью алгоритма:

1. Присвоить переменной Sum значение, равное нулю ($Sum := 0$).
2. Присвоить переменной i (слагаемому) значение, равное нулю ($i := 0$).
3. Повторять шаги:
 - 3.1. Увеличить i на 1 ($i := i + 1$).
 - 3.2. Прибавить к сумме значение слагаемого i ($Sum := Sum + i$).

Повторив шаги 3.1 и 3.2 нужное количество раз, получим требуемую сумму.

Это пример использования алгоритмической конструкции цикл (повторение).

В языке программирования Pascal имеются разновидности цикла, например:

- цикл с параметром (for..to, for..downto);
- цикл с условием (while).

Каждая из этих разновидностей цикла имеет свои особенности.

3.2. Оператор цикла с параметром

Если число повторений цикла известно заранее, то удобно использовать алгоритмическую конструкцию цикл с параметром. В языке Pascal повторение некоторой последовательности действий известное число раз выполняет оператор for. Рассмотрим работу этого оператора на примере.

Пример 1. Составить программу, которая 5 раз выводит на экран фразу «Наша Родина — Беларусь!».

Программа может быть такой:

```
program priz10;
var i: integer;
begin
  for i:=1 to 5 do
    {Параметр цикла i (счетчик) меняется
     от 1 до 5 с шагом 1}
    write('Наша Родина – Беларусь!')
    {Эта строка – тело цикла – выполнится
     5 раз}
end.
```

Результат выполнения программы:

```
Наша Родина – Беларусь! Наша Родина – Бе-
ларусь! Наша Родина – Беларусь! Наша Ро-
дина – Беларусь! Наша Родина – Беларусь!
```

В программе priz10 переменная *i* является *параметром цикла*. Параметр цикла является счетчиком выполнения команд *тела цикла* (команды `write('Наша Родина – Беларусь!')`). В *заголовке оператора* (`for i:=1 to 5 do`) указаны начальное (1) и конечное (5) значения параметра цикла *i*. При первом выполнении тела цикла значение параметра цикла *i* = 1, при втором — *i* = 2, ..., при последнем — *i* = 5. Каждый раз перед выполнением тела цикла текущее значение параметра цикла *i* сравнивается с конечным значением. После каждого выполнения тела цикла параметр *i* увеличивается на 1. Как только *i* превышает конечное значение, выполнение цикла прекращается и осуществляется переход на следующий после цикла *for* оператор программы.

Оформим ранее рассмотренный алгоритм вычисления суммы натуральных чисел в виде программы.

Пример 2. Составить программу, которая вычисляет сумму десяти первых натуральных чисел и выводит на экран результат.

Программа может быть такой:

```
program summa;
var i, sum: integer;
begin
  sum:=0;           {Начальное значение суммы}
  for i:=1 to 10 do
    {Параметр цикла i меняется от 1 до 10
    с шагом 1}
    sum:=sum+i;
    {Эта команда – тело цикла – выполнится 10 раз, каждый раз прибавляя к
    значению sum очередное значение i}
  writeln('Sum = ',sum); {Вывод результата}
end.
```

Результат выполнения программы:

```
Sum = 55
```

В общем виде оператор **for** может быть представлен в двух формах.

- **for** <параметр цикла>:=<начальное значение> **to** <конечное значение> **do** <оператор>;
или кратко:

```
for i:=N1 to N2 do <оператор>; (N1<N2).
```

Здесь значение параметра цикла последовательно увеличивается на 1.

- **for** <параметр цикла>:=<начальное значение> **downto** <конечное значение> **do** <оператор>;
или кратко:

```
for i:=N1 downto N2 do <оператор>; (N1>N2).
```



Рис. 1.8

Здесь значение параметра цикла последовательно уменьшается на 1.

В этих записях **for..do** — заголовок цикла, <оператор> — тело цикла, *i* — параметр цикла. Тело цикла может быть про-

стым либо составным оператором. Параметр цикла, его начальное и конечное значения должны принадлежать к одному типу данных (чаще всего это целочисленный тип *integer*, но могут быть и другие типы, кроме вещественного).

Блок-схема цикла с параметром *for* представлена на рисунке 1.8.

Пример 3. Составить программу, которая выводит на экран 10 первых натуральных чисел в обратном порядке.

Программа может быть такой:

```

program n10_1;
var i: integer;
begin
  for i:=10 downto 1 do write(i,' ');
  {Параметр i меняется от 10 до 1 с шагом -1}
end.
  
```

Результат выполнения программы:

10	9	8	7	6	5	4	3	2	1
----	---	---	---	---	---	---	---	---	---

! Нельзя изменять параметр цикла *for* в теле цикла с помощью команды присваивания. Это может привести к «зацикливанию» программы (бесконечному повторению тела цикла) либо к неверным результатам выполнения программы.

- ! Если в заголовке оператора `for..to..do` начальное значение параметра цикла больше конечного значения, то тело цикла не выполнится ни разу.
Если в заголовке оператора `for..downto..do` начальное значение параметра цикла меньше конечного значения, то тело цикла не выполнится ни разу.

Рассмотрим еще несколько примеров использования оператора `for`.

Пример 4. Составить программу, которая выводит на экран 5 первых нечетных натуральных чисел.

Программа может быть такой:

```
program n5;
var i, x: integer;
begin
  x:=1;
  {Первое нечетное натуральное число равно 1}
  for i:=1 to 5 do
    {Параметр i меняется от 1 до 5 с шагом 1}
    begin write(x,' ');
    {Вывод очередного нечетного числа}
    x:=x+2;
    {Следующее нечетное число на 2 больше предыдущего}
  end;
end.
```

Результат выполнения программы:

1	3	5	7	9
---	---	---	---	---

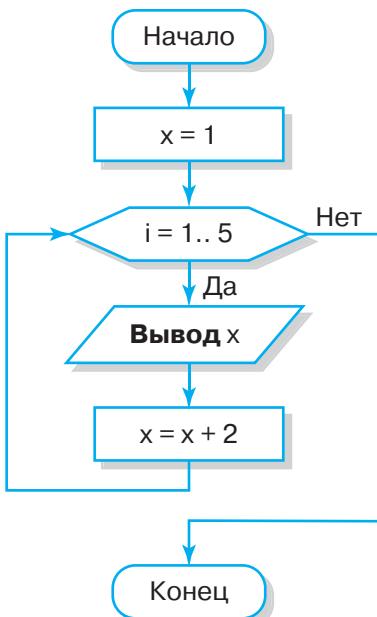


Рис. 1.9

Программа может быть такой:

```

program line_n;
uses graphabc;
var k, n: integer;
begin
  write('n = '); readln(n);
  ClearWindow(clSilver); SetPenWidth(5);
  SetPenColor(clBlue);
  for k:=1 to n do Line(random(25),
    random(25), random(600), random(600));
end.
  
```

В программе используется функция `random(n)`, генерирующая случайные числа от 0 до $n-1$. В данном примере генерируются случайные значения от 0 до 24 (`random(25)`) для x_1 , y_1 и случайные значения от 0 до 599 (`random(600)`) для x_2 , y_2 .

В примере 4 тело цикла — составной оператор, который образуют команды `write(x, ' ')` и `x:=x+2`.

Блок-схема алгоритма программы n5 представлена на рисунке 1.9.

Оператор `for` позволяет легко нарисовать множество графических фигур.

Пример 5. Составить программу, которая выводит на экран n отрезков голубого цвета, расположение которых определяется случайным образом.

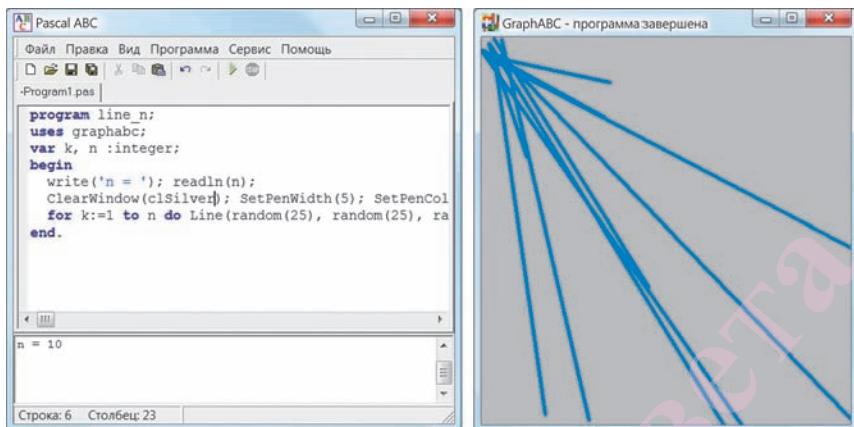


Рис. 1.10

Результат выполнения программы при $n = 10$ представлен на рисунке 1.10.

Пример 6. Составить программу, которая выводит на экран натуральные числа от 5 до 9.

Программа может быть такой:

```
program n5_9;
var i: integer;
begin for i:=5 to 9 do write(i,' ');
      {Параметр i меняется от 5 до 9}
end.
```

Результат выполнения программы:

5	6	7	8	9
---	---	---	---	---

Данный пример показывает, что начальное значение параметра цикла не обязательно равно единице. Взяв разность между начальным и конечным значениями параметра цикла по модулю и прибавив к ней 1, можно вычислить количество повторений тела цикла. В примере 6 тело цикла выполнится 5 раз ($(9 - 5) + 1 = 5$).

3.3. Оператор цикла с предусловием

Оператор цикла с параметром `for` удобен, когда число повторений действий заранее известно. Однако часто приходится решать задачи, в которых число повторений тела цикла заранее неизвестно и определяется только в ходе выполнения цикла. В этом случае применяют цикл с условием. В языке Pascal имеется две разновидности цикла с условием:

- *цикл с предварительным условием* — условие цикла проверяется перед выполнением тела цикла;
- *цикл с последующим условием* — условие цикла проверяется после выполнения тела цикла.

Остановимся на цикле с предварительным условием, или с предусловием.

Цикл с предусловием — это цикл, который повторяется до тех пор, пока условие истинно. Блок-схема цикла с предусловием представлена на рисунке 1.11.

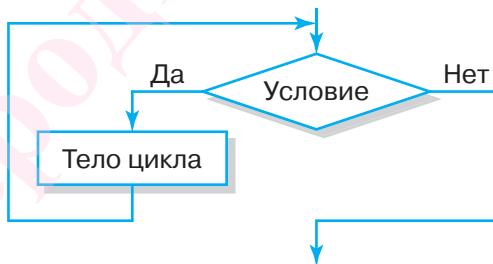


Рис. 1.11

Для реализации цикла с предусловием в языке Pascal используется оператор `while`, который имеет следующий вид:

`while <условие> do <оператор>;`

Если тело цикла состоит из нескольких операторов, необходимы операторные скобки `begin..end`:

```

while <условие> do ←———— Заголовок цикла
begin
  <оператор _ 1>;
  <оператор _ 2>; ←———— Тело цикла
  ...
end;

```

В операторе while проверка условия происходит перед каждым выполнением тела цикла. Если условие истинно, выполняется тело цикла и снова проверяется условие. Если условие становится ложным, происходит выход из цикла.

Рассмотрим работу оператора while на знакомом нам примере вычисления суммы n первых натуральных чисел.

Пример 7. Составить программу, которая вычисляет сумму n первых натуральных чисел и выводит на экран результат.

Программа может быть такой:

```

program summa _ n _ 2;
var i, n, sum: integer;
begin
  write('n = '); readln(n);
  {n – количество первых натуральных чисел}
  sum:=0; {Начальное значение суммы}
  i:=0; {Переменной i присвоено начальное
         значение 0, цикл выполняется, пока i<n}
  while i<n do begin i:=i+1;
  {Переменная i принимает значения от 1 до n}
  sum:=sum+i;
  {Увеличение суммы на очередное число}
  end;
  writeln('Sum = ',sum);
end.

```

Результат выполнения программы при $n = 10$:

```
n = 10
Sum = 55
```

Цикл в этой программе работает следующим образом:

- вначале $i = 0$, $n = 10$, $sum = 0$; условие $i < 10$ истинно, и тело цикла выполняется:
 - i увеличивается на 1 ($i = i + 1 = 0 + 1 = 1$);
 - значение суммы увеличивается на 1 ($sum = sum + i = 0 + 1 = 1$);
- условие $i < 10$ вновь истинно, поэтому тело цикла выполняется в очередной раз:
 - i увеличивается на 1 ($i = i + 1 = 1 + 1 = 2$);
 - значение суммы увеличивается на 2 ($sum = sum + i = 1 + 2 = 3$);
- условие $i < 10$ истинно, поэтому тело цикла выполняется в очередной раз и т. д.

После 10 повторений данной последовательности действий ($i = i + 1$ и $sum = sum + i$) получим $i = 10$, $sum = 1 + 2 + \dots + 10$. Условие $i < 10$ ложно, поэтому выполнение цикла завершится.

! Если условие сразу оказывается ложным, оператор `while` не выполняется ни разу!

В теле цикла должны быть операторы, которые могут изменить значение условия, сделав его ложным. Иначе цикл будет выполнять бесконечное число раз.

Зациклившуюся программу можно остановить командой **Программа → Завершить**.

В операторах `for`, `while` точка с запятой не ставится перед словом `do` и после него.

Рассмотрим еще несколько примеров использования оператора `while`.

Пример 8. Составить программу, которая определяет, сколько слагаемых должно быть в сумме последовательных четных чисел ($2 + 4 + 6 + 8 + \dots$), чтобы эта сумма оказалась больше некоторого заданного натурального числа n ($n \geq 2$), и выводит на экран результат — количество слагаемых.

Программа может быть такой:

```
program kol_slag;
var i, sum, n, x: integer;
begin
  write('n = '); readln(n); x:=2;
  {Первое слагаемое равно 2}
  sum:=2; i:=1;
  {Начальное значение счетчика слагаемых
  равно 1}
  while sum<=n do
    {Пока sum<=n, выполняется тело цикла}
    begin x:=x+2;
    {Очередное слагаемое на 2 больше предыдущего}
    i:=i+1;
    sum:=sum+x;
    {Увеличение счетчика слагаемых на 1. Прибавление к сумме очередного слагаемого}
  end;
  writeln(i,' слагаемых');
end.
```

Результат выполнения программы при $n = 30$:

```
n = 30
6 слагаемых
```

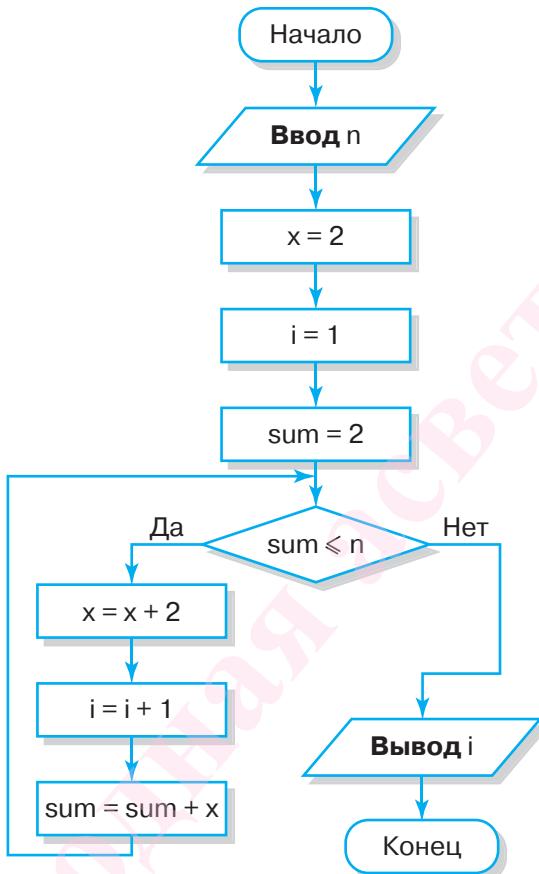


Рис. 1.12

Блок-схема алгоритма программы kol_slag представлена на рисунке 1.12.

Цикл while удобно использовать для рисования повторяющихся графических фигур. При составлении следующих программ будем использовать размер графического окна 640×480 .

Пример 9. Составить программу, которая рисует ряд зеленых кругов радиуса 20 по горизонтали вдоль верхнего края графического окна.

Программа может быть такой:

```

program krugi;
uses graphabc;
var x, y, r: integer;
begin
  ClearWindow(clSilver);
  SetBrushColor(clGreen);
  x:=22; y:=22; r:=20; {Начальные значения x, y}
  while x<600 do
    begin Circle(x, y, r);
    {Пока x<600, рисование очередного круга}
    x:=x+40;
    {Смещение координаты x для следующего
    круга}
  end;
end.

```

Результат выполнения программы показан на рисунке 1.13.

Пример 10. Составить программу, которая имитирует движение зеленого шарика радиуса 20 по го-

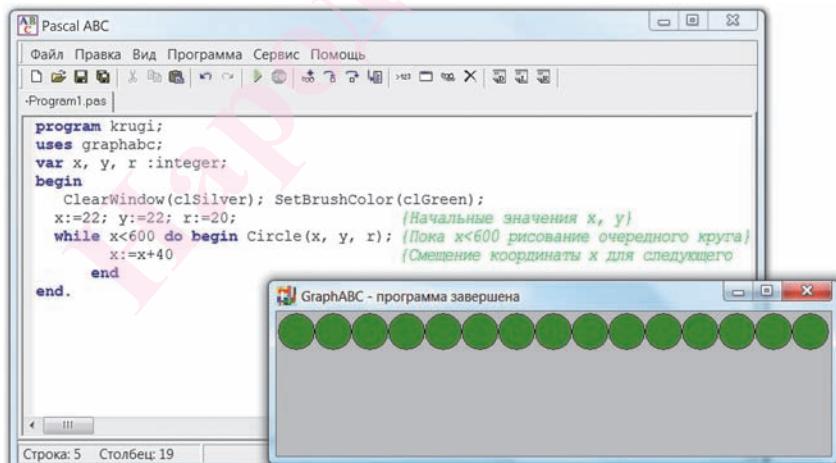


Рис. 1.13

ризонтали слева направо вдоль верхнего края графического окна.

Для получения иллюзии движения объекта надо выполнить следующие действия:

- 1) нарисовать объект и организовать временную паузу;
- 2) нарисовать объект на прежнем месте цветом фона (стереть объект);
- 3) повторять действия 1) и 2), изменения координаты объекта.

```
program dvigenie;
uses graphabc;
var x, y, r: integer;
begin
    x:=22; y:=22; {Начальные координаты объекта}
    r:=20;
    while x<600 do begin
        {Пока x<600, повторение действий}
        SetBrushColor(clGreen); Circle(x, y, r);
        Sleep(10); {Пауза в миллисекундах}
        SetBrushColor(clWhite);
        SetPenColor(clWhite);
        Circle(x, y, r); {Стирание объекта}
        x:=x+4; {Изменение координат объекта}
    end;
end.
```

Результат выполнения программы dvigenie получите самостоятельно.

- ? 1. Какие формы оператора **for** Вы знаете?
- ? 2. Можно ли изменять параметр цикла в теле цикла **for**?
- ? 3. Какой тип данных может иметь параметр цикла?
- ? 4. В каких случаях при написании программ лучше использовать оператор цикла с предусловием?

5. В каких случаях цикл с предусловием может быть бесконечным?
6. При каких условиях цикл с предусловием не выполнится ни разу?

Упражнения

1. Сколько раз выполнится тело цикла
`for n:=3 to 6 do write('*');`
2. Составьте программу, которая 6 раз выведет на экран фразу «Я люблю свою Родину!» в строку через пробел.
3. Составьте блок-схему и напишите программу вывода на экран:
 - а) 5 первых четных натуральных чисел;
 - б) n первых натуральных чисел.
4. Составьте программу, которая вычисляет сумму натуральных чисел от 10 до 15 и выводит на экран результат.
5. Составьте программу, которая определяет, сколько слагаемых должно быть в сумме $1 + 2 + 3 + 4 + \dots$, чтобы сумма оказалась:
 - а) больше 20;
 - б) больше заданного числа n .Результат — количество слагаемых — выведите на экран. Составьте блок-схемы.
6. Составьте программу, которая рисует красные круги радиуса 30 по вертикали вдоль левого края графического окна.
7. Составьте программу, которая имитирует движение голубого шарика радиуса 20 по горизонтали слева направо вдоль нижнего края графического окна.

§ 4. Составление и реализация алгоритмов с повторением

Вы уже знаете, что циклы имеют свои разновидности и особенности. Составляя алгоритмы с повторением для решения задач, следует выбирать наиболее подходящий оператор цикла.

Пример 1. Составить программу, которая вычисляет сумму вводимых с клавиатуры целых чисел до ввода отрицательного числа и выводит на экран значение вычисленной суммы.

Так как неизвестно, сколько чисел будет введено, логично будет использовать оператор `while`, в заголовке которого условием выполнения тела цикла будет положительность либо равенство нулю введенного числа.

Алгоритм решения задачи в виде блок-схемы представлен на рисунке 1.14.

Программа, реализующая представленный алгоритм, может быть такой:

```

program sum _ do _ otr;
var sum, a: integer;
begin sum:=0;
    write('Введите первое число: ');
    readln(a);
    {Ввод первого числа}
    while a>=0 do
        {Проверка числа на положительность либо
        равенство нулю. Если число а отрицательное,
        происходит выход из цикла}
        begin
            sum:=sum+a;
            {Прибавление числа а к сумме}

```

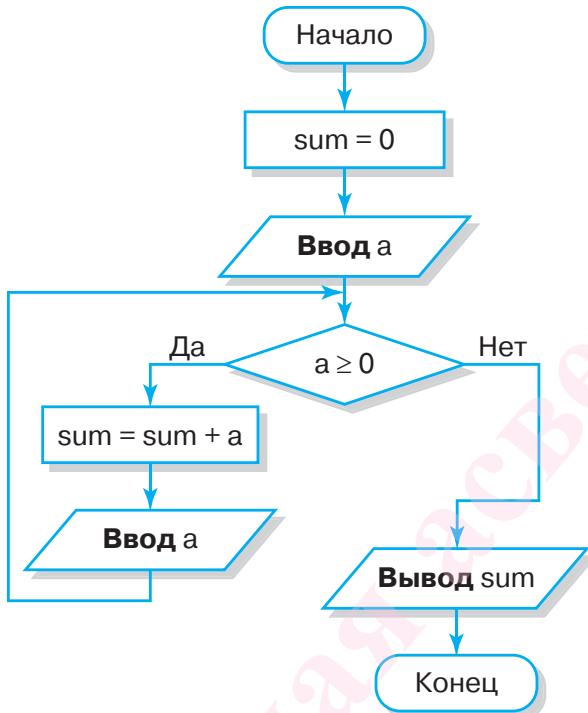


Рис. 1.14

```

write('Введите следующее число: ');
readln(a);
{Ввод очередного числа}
end;
writeln('Sum = ',sum);
end.
  
```

Результат выполнения программы при введенных числах 2, 5, -4:

```

Введите первое число: 2
Введите следующее число: 5
Введите следующее число: -4
Sum = 7
  
```

Если первое же введенное число — отрицательное, то тело цикла не выполнится ни разу:

```
Введите первое число: -5
Sum = 0
```

Пример 2. Составить программу, которая вычисляет произведение n первых натуральных чисел и выводит на экран результат.

Количество повторений действий известно, поэтому разумно использовать оператор `for`. Алгоритм вычисления произведения n чисел похож на алгоритм вычисления суммы n чисел, только операцию сложения в теле цикла необходимо заменить операцией умножения. Начальное значение переменной, в которой будет накапливаться произведение, должно быть равным 1.

Программа может быть такой:

```
program proizv_n;
var i, n, P: integer;
begin
  write('n = '); readln(n);
  {n - количество первых натуральных чисел}
  P:=1;           {Начальное значение P равно 1}
  for i:=2 to n do P:=P*i; {В переменной P накапливается произведение}
  writeln('P = ',P);
end.
```

Результат выполнения программы при $n = 5$:

```
n = 5
P = 120
```

Порядок выполнения действий в теле цикла важен. Если поменять местами операторы, програм-

ма будет работать по-другому. Это может приводить к **логическим ошибкам**. Такие ошибки не мешают выполнению программы, но могут приводить к неверным результатам. Понять, правильно ли работает программа, можно при выполнении вручную нескольких шагов цикла с отслеживанием значений переменных. Такой метод отладки программы называется **трассировкой**. Трассировка программы `proizv_n` для $n = 3$ показана в следующей таблице:

Оператор	i	Результат проверки условия продолжения цикла	Вычисление	P
<code>P:=1; n:=3;</code>	1	Истинно	—	1
<code>For i:=2 to 3 do P:=P*i</code>	2	Истинно	$1 \cdot 2 = 2$	2
<code>For i:=2 to 3 do P:=P*i</code>	3	Истинно	$2 \cdot 3 = 6$	6
<code>For i:=2 to 3 do P:=P*i</code>	4	Ложно, выход из цикла	—	—

Трассировка позволяет вникнуть в логику работы программы и на каждом шагу проверить, правильны ли были рассуждения при ее составлении.

Известно, что операция возведения числа в натуральную степень основана на произведении заданного количества множителей, равных основанию степени:

$$2^5 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \quad \text{или} \quad a^4 = a \cdot a \cdot a \cdot a.$$

Следовательно, на основе алгоритма программы `proizv_n` можно составить алгоритм вычисления

натуральной степени целого числа a^n . Для этого в теле цикла $P:=P*i$ переменную i необходимо заменить на основание степени a .

Пример 3. Составить программу, которая выводит на экран значение выражения a^n , где a — целое число, n — натуральное.

Программа может быть такой:

```
program stepen;
var a, n, i, P: integer;
begin
    write('a = '); readln(a); {Ввод основания}
    write('n = '); readln(n);
    {Ввод показателя степени}
    P:=1; {Начальное значение Р равно 1}
    for i:=1 to n do P:=P*a;
    {В переменной Р накапливается значение
    степени}
    writeln('P = ',P); {Вывод результата}
end.
```

Результаты выполнения программы для пары чисел 2 и 5, а также для пары чисел 5 и 0:

a = 2

n = 5

P = 32

a = 5

n = 0

P = 1

Используя алгоритм вычисления степени числа, нетрудно решить следующую задачу.

Пример 4. Составить программу, которая запрашивает целое число $a < 5$, находит наименьшее натуральное число n , при котором $a^n \geq 50$, и результат — наименьшее число n — выводит на экран.

Степень числа неизвестна, поэтому используем цикл while.

Программа может быть такой:

```
program poisk_n;
var a, n, St: integer;
begin
  write('Основание a = '); readln(a);
  n:=0; St:=1; n – показатель степени, в переменной St накапливается степень числа a}
  while St<50 do
    begin
      n:=n+1; St:=St*a;
      {Увеличение степени n, пока St < 50}
    end;
  writeln('Наименьшее n = ',n);
  {Результат – последнее значение n}
end.
```

Результат выполнения программы при $a = 2$:

```
Основание a = 2
Наименьшее n = 6
```

Применим возможности операторов цикла при работе с графикой.

Пример 5. Составить программу, которая рисует 30 отрезков двух разных цветов. Координата начала каждого отрезка соответствует координате центра графического окна. Координата конца отрезка определяется случайным образом.

Программа может быть такой:

```
program linii_v_centre;
uses graphabc;
var k: integer;
begin
  SetWindowSize(1280, 1024);
```

```
ClearWindow(clYellow);
SetPenWidth(5);
for k:=1 to 30 do
begin if k mod 2=0
    then SetPenColor(clFuchsia)
    else SetPenColor(clGreen);
    Line(640, 512, random(1000),
        random(900));
end;
end.
```

Результат выполнения программы показан на рисунке 1.15.

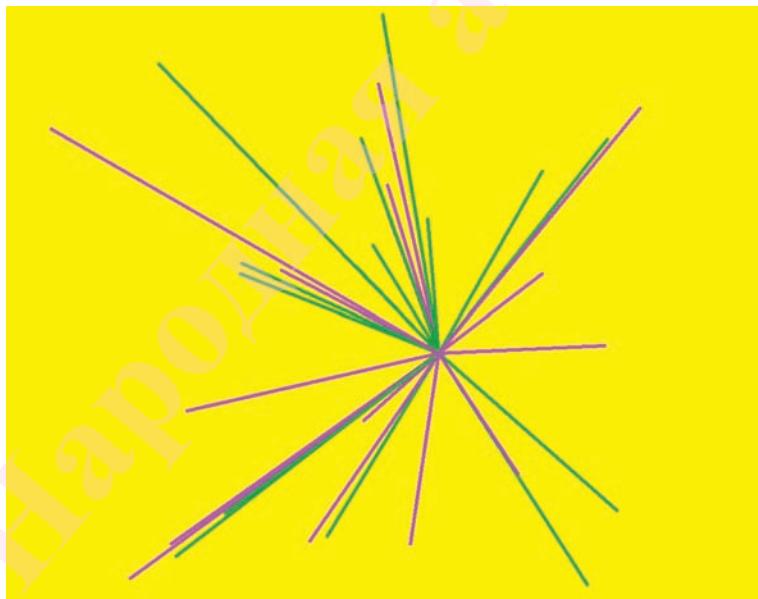


Рис. 1.15

Пример 6. Составить программу, которая рисует 200 горизонтальных разноцветных отрезков одинаковой длины.

Программа может быть такой:

```
program cvet _ gamma;
uses graphabc;
var k: integer;
begin
  SetWindowSize(1280, 1024);
  ClearWindow(clSilver);
  SetPenWidth(5);
  for k:=1 to 200 do
    begin
      SetPenColor (rgb(k*10, k*30, k*50));
      {Функция rgb(r,g,b) задает цвет, состав-
ленный из красного (r), зеленого (g) и
синего (b) компонентов цветовой гаммы в
диапазоне от 0 до 255}
      Line(200, k*5, 600, k*5);
    end;
end.
```

Результат выполнения программы показан на рисунке 1.16.

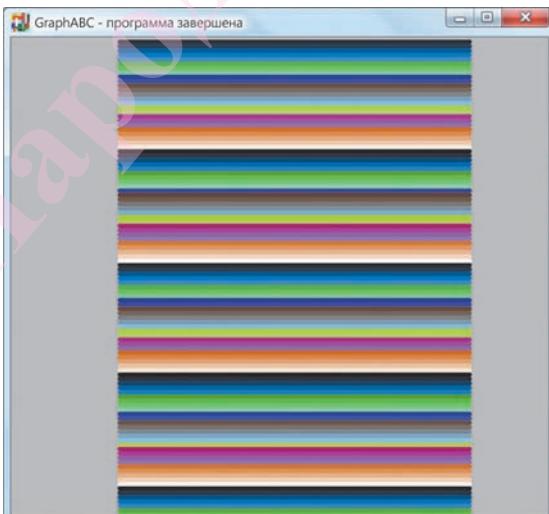


Рис. 1.16

Пример 7. Составить программу, которая рисует картинку «Звездное небо»: 256 разноцветных кругов («звезд»), радиус которых определяется случайным образом в диапазоне от 2 до 10, на черном фоне.

Программа может быть такой:

```
program nebo;
uses graphabc;
var x, y, k: integer;
begin
  SetWindowSize(1280, 1024);
  ClearWindow(clBlack);
  for k:=1 to 256 do
    begin
      SetBrushColor(rgb(k*100, k*25, k*50));
      {Цвет кисти разных оттенков}
      x:=random(1100); y:=random(1000);
      Circle(x, y, random(9)+2);
    end;
end.
```

Результат выполнения программы показан на рисунке 1.17.

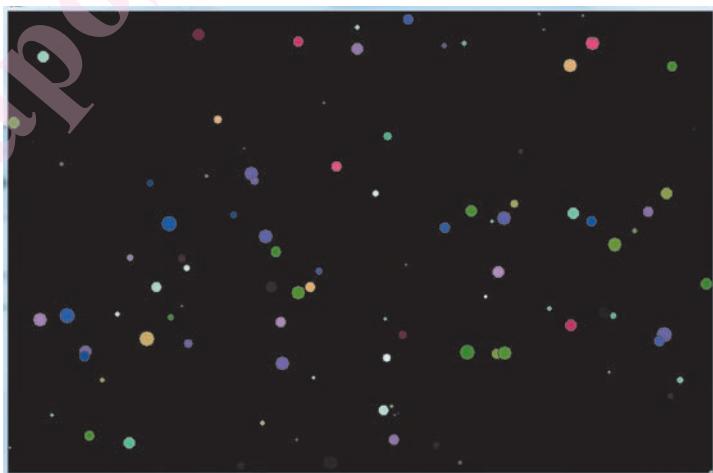


Рис. 1.17

```

Pascal ABC
Файл Правка Вид Программа Сервис Помощь
-Program1.pas
program tabulir;
var x, y :real;
begin
  writeln('-----'); {Оформление вывода результата}
  writeln(' x      y');
  writeln('-----'); {Начальное значение x}
  x:=-2; {Проверка, не выходит ли x за границу отрезка}
  while x<=2 do
    begin
      y:=x+1; {Вычисление и вывод очередного значения y}
      writeln(x:4:1, y:5:1); {Нахождение очередной точки x}
      x:=x+0.5
    end
  end.

```

x	y
-2.0	-1.0
-1.5	-0.5
-1.0	0.0
-0.5	0.5
0.0	1.0
0.5	1.5
1.0	2.0
1.5	2.5
2.0	3.0

Строка: 4 Столбец: 33

Рис. 1.18

С помощью оператора `while` удобно вычислять координаты точек графиков функций.

Пример 8. Составить программу, которая выводит на экран таблицу значений функции $y = x + 1$, вычисленных в каждой точке x на отрезке $[-2, 2]$ с шагом 0,5 (рис. 1.18).

Процесс вычисления значений функции в каждой точке на заданном отрезке с заданным шагом называется *табулированием функции*.

Программа может быть такой:

```

program tabulir;
var x, y: real;
begin

```

```

{Оформление вывода результата}
writeln('-----'); writeln(' x      y');
writeln('-----');
x:=-2;   {Начальное значение x}
while x<=2 do {Проверка, не выходит ли x за
границу отрезка}
begin y:=x+1; writeln(x:4:1, y:5:1);
{Вычисление и вывод очередного значе-
ния y}
x:=x+0.5; {Очередная точка x}
end;
end.

```

 **Пример 9.** Составить программу, которая находит наибольший общий делитель (НОД) двух данных целых чисел и выводит на экран результат.

Найдем НОД двух чисел с помощью следующего алгоритма. Будем из большего числа вычитать меньшее до тех пор, пока они не станут равными. Полученное число и есть результат. Такой алгоритм требует использования оператора while.

Программа, реализующая данный алгоритм, может быть такой:

```

program nod _ 2;
var a, b: integer;
begin
write('a = '); readln(a);
write('b = '); readln(b);
while a<>b do {Проверка равенства чисел}
  if a>b then a:=a-b else b:=b-a;
{Всегда из большего вычитаем меньшее}
writeln('НОД = ', a) {После выхода из цикла
a=b - искомый результат}
end.

```

Результат выполнения программы при $a = 12$ и $b = 20$:

```
a = 12  
b = 20  
НОД = 4
```

 **Пример 10.** Составить программу, которая находит все делители заданного натурального числа, большего 2, и выводит их на экран.

Чтобы найти все делители натурального числа (кроме единицы и самого числа), необходимо разделить число на все числа от 2 до половины данного числа. Этот алгоритм можно реализовать в виде цикла с параметром.

Программа может быть такой:

```
program deliteli;  
var a, d: integer;  
begin  
    write('Введите число: '); readln(a); write('1 ');\n    for d:=2 to (a div 2) do\n        if a mod d = 0 then write(d,' ');\n        {Если остаток от деления a на d равен  
        нулю, то вывести на экран делитель d}\n        write(a);\nend.
```

Результат выполнения программы при $a = 12$:

```
Введите число: 12  
1 2 3 4 6 12
```

Рассмотрим задачу о банковских вкладах.

Известно, что сбережения в банке приносят доход — проценты от вклада, которые начисляются по

истечении некоторого времени (год, месяц). При этом годовые проценты могут начисляться двумя способами: с капитализацией или без нее.

Если сбережения не снимать со счета длительное время, то при капитализации начисленные проценты увеличивают сумму вклада, т. е. в дальнейшем проценты начисляются на всю накопленную сумму. В таких случаях при расчетах используют формулу сложных процентов.

Без капитализации начисление процентов осуществляется на сумму исходного вклада без учета ранее начисленных процентов. Тогда используют формулу простых процентов.

 **Пример 11.** Составить программу, которая определяет, через сколько лет сумма в 200 000 р., положенная в банк под 10 % годовых с капитализацией, увеличится в 2 раза.

Программа может быть такой:

```
program vklad;
const sum0=200000; proc:=10;
      {Начальная сумма вклада и процент}
var sum: real; k: integer;
begin
  sum:=sum0; k:=0;   {k - счетчик лет}
  while sum<2*sum0 do {Проверка «пока
    вклад не увеличился в два раза»}
    begin
      sum:=sum+sum*proc/100; k:=k+1;
      {Ежегодное увеличение суммы и счетчика лет}
    end;
    writeln('Через ',k,' лет');
end.
```

Результат выполнения программы:

Через 8 лет

Проведите эксперимент, изменяя в программе значения констант первоначальной суммы вклада и ежегодного процента.

Рассмотрим пример игровой программы.



Пример 12. Составить программу, с помощью которой компьютер «задумает» число от 0 до 10, предложит Вам его угадать и определит, за сколько попыток Вы это сделали.

```
program igra;
var x, a, n: integer;
begin
x:=random(11);
{Случайное число от 0 до 10 помещается
в переменную x}
writeln('Задумано целое число от 0 до
10. Угадайте его!');
write('Введите предполагаемое число: ');
read(a);
n:=1; {Сделана первая попытка}
while a<>x do begin
  n:=n+1;
  writeln('Не угадали!');
  {Следующая попытка}
  if a<x then writeln('Ваше число мень-
  ше задуманного!')
    else writeln('Ваше число боль-
    ше задуманного!');
  writeln('Попытайтесь еще раз!');
  readln(a);
end;
```

```
writeln('Поздравляю! Число угадано за ',
n,' попыток.');
end.
```

Поиграйте с компьютером, реализуя эту программу.

Упражнения

1. Составьте программу, которая вычисляет произведение натуральных чисел от 5 до 10 и выводит на экран результат.

2. Составьте программу, которая выводит на экран квадраты n первых натуральных чисел.

3. Составьте программу, которая выводит на экран значение выражения:

$$\text{а) } 5^n; \quad \text{б) } a^{10}; \quad \text{в) } (a + 1)^n.$$

4. Составьте программу, которая рисует картинку «Звездное небо»: n маленьких разноцветных кругов («звезд»), радиус которых определяется случайным образом в диапазоне от 4 до 14, на темно-синем фоне.

5. Составьте программу, которая рисует 10 окружностей зеленого цвета разных радиусов с одним центром.

6. Составьте программу, которая определяет, являются ли два данных натуральных числа взаимно простыми, и выводит на экран соответствующее сообщение: «Да» либо «Нет». (Натуральные числа называются взаимно простыми, если их наибольший общий делитель равен 1.)

7. Составьте программу, которая определяет, является ли данное натуральное число простым, и выводит на экран соответствующее сообщение: «Да» либо «Нет». (Простое число делится только на 1 и на само себя.)

 8. Составьте программу, которая определяет, через сколько лет сумма в S р., положенная в банк под P % годовых с капитализацией, увеличится в 3 раза.

 9. Составьте программу вывода на экран n первых чисел Фибоначчи. (Числа Фибоначчи — это члены числовой последовательности 1, 1, 2, 3, 5, 8, 13, ..., которые вычисляются по следующему правилу: $a_1 = 1$, $a_2 = 1$, $a_3 = a_2 + a_1$, ..., $a_n = a_{n-1} + a_{n-2}$.)

 10. Составьте программу «Угадай!». Программа «задумывает» случайное число в диапазоне от 0 до 99. После каждой попытки играющего угадать число программа выводит на экран одно из сообщений: «Больше», «Меньше» или «Угадано». На отгадывание числа дается 10 попыток. Программа завершается при выполнении одного из условий: дан правильный ответ либо все попытки исчерпаны.

Глава 2

Технология обработки текстовых документов

Электронным текстовым документом называется текст, представленный в электронно-цифровом виде, содержащий элементы оформления или включающий различные объекты. Объектами могут быть таблицы, рисунки, диаграммы и т. д.

Напомним, что **текстовыми редакторами** называются компьютерные программы для работы с текстом. Текстовые редакторы делят на:

- **простейшие текстовые редакторы**, предназначенные для создания, редактирования и вывода на печать текстов (например, **Блокнот**);
- **текстовые процессоры**, позволяющие создавать электронные текстовые документы (например, **WordPad**, **Word**).

В 6-м классе мы изучали приемы создания, редактирования и форматирования текстовых документов. При этом подробно разбирали возможности, доступные текстовому процессору **WordPad**. Однако в **WordPad** предусмотрен минимальный набор инструментов форматирования. Их недостаточно, например, при разработке документа, содержащего таблицы, рисунки, формулы. В этих случаях требуется применение более широкого набора возможностей текстового процессора.

В Беларуси получил широкое распространение текстовый процессор **Microsoft Word**. Рассмотрим технологию обработки текстовых документов на примере **Microsoft Office Word 2003**.

В программе существует четыре способа доступа к командам: главное меню, кнопки на панелях инструментов, контекстное меню, клавиатурные сокращения.

Главное меню, как правило, открывает наиболее полный доступ к параметрам каждой команды. Поэтому чаще всего будем использовать его.

При запуске приложения Word по умолчанию на экране отображаются две панели: Стандартная и Форматирование. Командой Вид → Панели инструментов... можно включать другие панели инструментов.

§ 5. Поиск и замена в тексте, проверка правописания

5.1. Функции поиска и замены

Поиск нужного фрагмента, особенно в многостраничном документе, может занять много времени. Процедуру поиска можно ускорить благодаря возможностям текстового процессора.

Если в большом документе требуется отобрать сведения, например, о жирафе, то в меню выбирают команду Правка → Найти.... В поле Найти диалогового окна Найти и заменить (рис. 2.1) вводят образец для поиска, например слово «жираф». При щелчке по кнопке Найти далее начинается поиск.

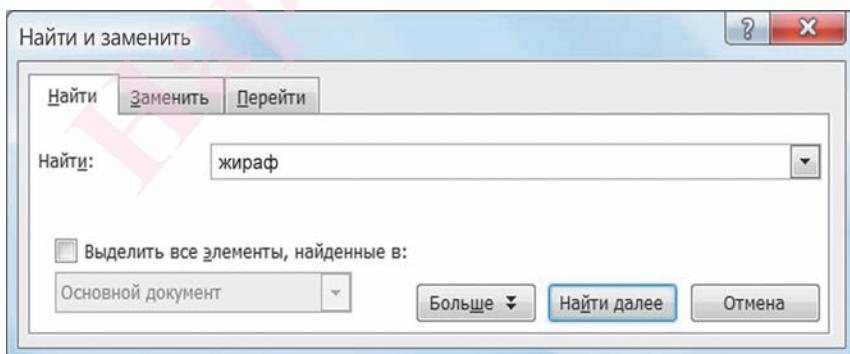


Рис. 2.1

Найдя совпадение с образцом, программа выделяет найденный фрагмент. Окно **Найти и заменить** остается открытым. Можно продолжить поиск, повторно нажав кнопку **Найти далее**.

Если перед вызовом команды выделить часть текста, поиск будет вестись в выделенном фрагменте.

Пример 1. Найти в тексте слова «верблюд» и «верблюжонок».

Поставим текстовый курсор в начало текста и выберем команду **Правка → Найти...**. В поле **Найти** наберем образец «верблю» — общую часть искомых слов. Нажмем кнопку **Найти далее**. Будет выделено первое найденное слово. Для поиска остальных слов будем повторно нажимать кнопку **Найти далее**.

Кроме поиска, можно выполнять замену фрагментов текста. Если требуется найти и исправить все ошибочные слова, например «сабака», то выбирают команду **Правка → Заменить...** и открывают вкладку **Заменить** окна **Найти и заменить** (рис. 2.2). В поле **Найти** указывают образец для поиска слова в любых падежах «сабак». В поле **Заменить на** — образец для замены «собак». При щелчке по кнопке

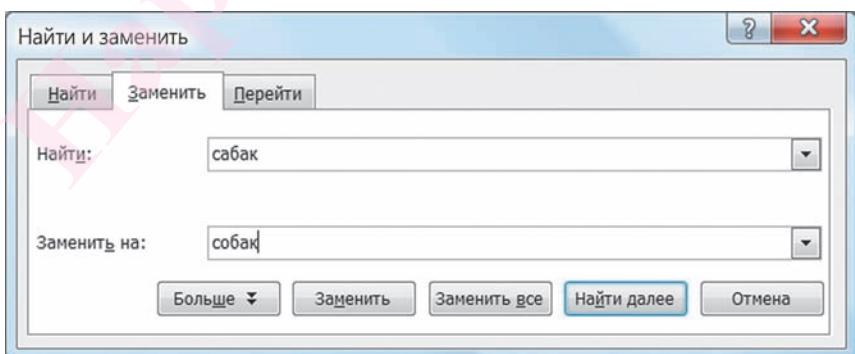


Рис. 2.2

Заменить все выполняется замена указанных образцов во всем тексте.

Пример 2. В тексте «Минск, Брест, Витебск, Гомель, Гродно, Могилев» удалить лишние гласные.

Поставим текстовый курсор в начало текста и выберем команду **Правка → Заменить...**. В поле **Найти** введем образец «ии», а в поле **Заменить на** — образец «и». Нажмем кнопку **Заменить все**. Все сочетания «ии» заменятся на букву «и». Аналогично заменим сочетания «ее» и «oo».

5.2. Проверка правописания

Текстовый процессор Word позволяет автоматически проверять правописание. Проверить можно орфографию, пунктуацию и стиль документа. Напомним, что орфография — это правила написания слов, пунктуация — правила расстановки знаков препинания, а стиль — манера построения фраз и предложений. При проверке орфографии программа сравнивает слова с образцами во встроенном словаре. Если слова нет в словаре, то в тексте оно подчеркивается красной волнистой линией. Пунктуационные ошибки и ошибки стиля Word отмечает зеленой волнистой линией.

Предлагаемые программой варианты исправления найденной ошибки можно увидеть в контекстном меню. Для этого надо щелкнуть правой кнопкой мыши по подчеркнутому слову или фрагменту текста. Программа не всегда может предложить варианты исправления слов с ошибками. Для ошибок пунктуации и стиля часто дается только описание ошибки.

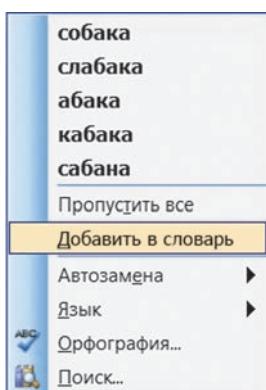


Рис. 2.3

Пример 3. Исправить ошибки в словах «сабака», «жырав».

Щелкнем правой кнопкой мыши по первому слову, подчеркнутому красной волнистой линией. В открывшемся контекстном меню с вариантами исправления ошибки (рис. 2.3) щелчком мыши выберем правильный вариант «собака». Щелчок правой кнопкой мыши по второму слову откроет контекстное меню, в котором нет нужного слова «жираф». Исправим его сами.

Выполнить проверку правописания в любой момент можно, выбрав команду Сервис → Правописание. Она открывает диалоговое окно Правописание (рис. 2.4), в котором можно исправить все обнаруженные программой ошибки.

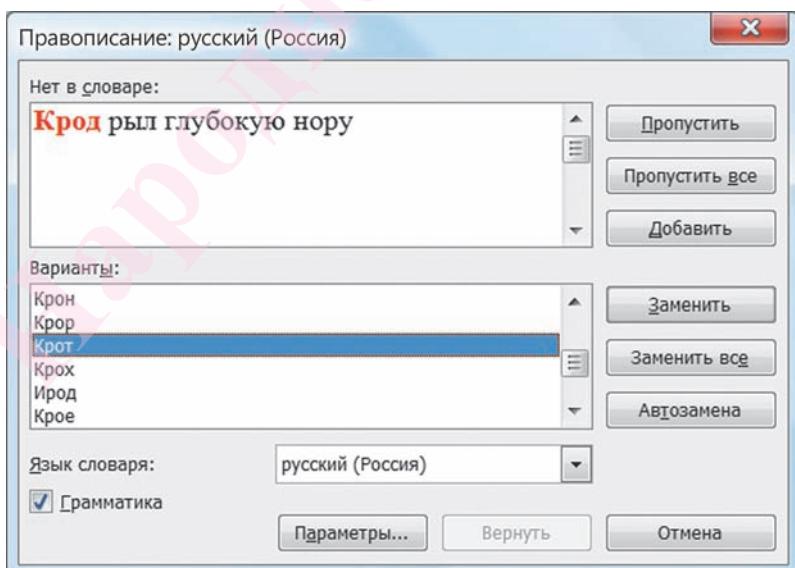


Рис. 2.4

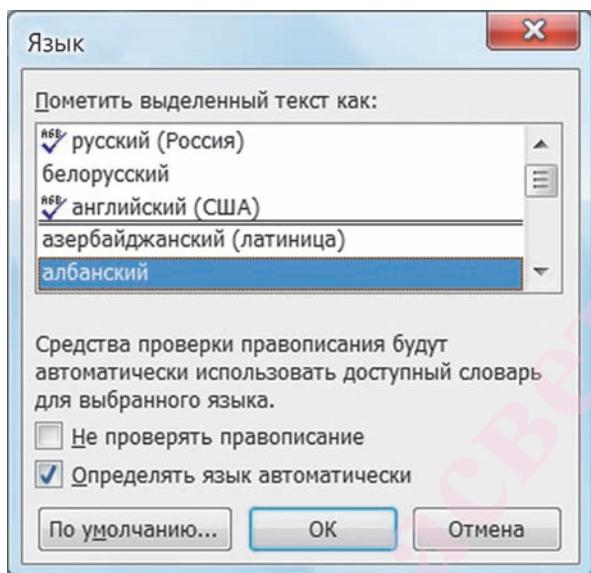


Рис. 2.5

Как мы уже говорили, при проверке орфографии Word использует встроенный словарь. Можно также формировать свой вспомогательный словарь. Он пополняется по команде **Добавить в словарь** контекстного меню (см. рис. 2.3) или нажатием кнопки **Добавить** в окне **Правописание** (см. рис. 2.4). Пользоваться этой возможностью надо аккуратно, чтобы в словарь не попадали слова с ошибками.

На результат проверки правописания влияет используемый язык проверки. Выбрать язык можно командой **Сервис** → **Языки** → **Выбрать язык...**. Знак перед названием языка показывает, возможна ли проверка правописания для этого языка. На рисунке 2.5 такими значками выделены русский и английский языки.

Механизм проверки правописания не может заменить знаний правил и норм языка. Живой язык —

очень сложная, гибкая и богатая система. Предусмотреть в программе проверку всех вариантов правописания не представляется возможным.

! Средства проверки правописания не могут обнаружить все ошибки и предложить варианты их исправления. Более того, средства проверки могут указать и на правильно написанный текст как на текст, содержащий ошибку.

Упражнения

1. Составьте образец, по которому можно найти слово «черешня» в любом падеже единственного и множественного числа.

2. Какой образец нужен для поиска двух вариантов слова: «круг» и «кружок»?

3. Какой образец надо использовать при поиске слова «коробочка» в любом падеже единственного и множественного числа?

4. В тексте

«Любишь·кататься,:люби·и·саночки·возить.¶»

встречаются по два подряд идущих пробела. Составьте образцы поиска и замены, позволяющие удалить лишние пробелы.

5. В тексте

«Белка·,кабан·,лось·,заяц·,рысь·,волк·,зубр·—
·звери·нашего·леса.¶»

пробелы поставлены перед запятыми, а не после. Составьте образцы поиска и замены, позволяющие быстро исправить ошибку во всем тексте.

6. Даны слова с пропущенными буквами. С помощью автоматической проверки правописания определите, для каких слов предлагаются правильные варианты исправления ошибки.

Слово с пропущенной буквой	Правильное слово
млоко	молоко
блако	облако
блоко	яблоко
абука	азбука
бараан	барабан
арта	парта
мика	мишка

§ 6. Создание и форматирование списков

6.1. Маркированный список

С помощью текстового процессора часть текста можно оформить в виде списка. **Список** — это упорядоченный конечный набор элементов.

Список может быть *маркированным*. Начало каждого элемента такого списка помечается особым знаком — маркером.

Пример 1. Построить маркированный список посуды, необходимой в турпоходе.

Посуда для турпохода

- ✓ Кружка.
- ✓ Ложка.
- ✓ Миска.
- ✓ Нож (складной).

Набрав заголовок, перейдем к первой строке создаваемого списка. Выбрав команду меню **Формат → Список...**, откроем окно **Список** (рис. 2.6). На вкладке **Маркированный** выберем подходящий вариант оформления списка.

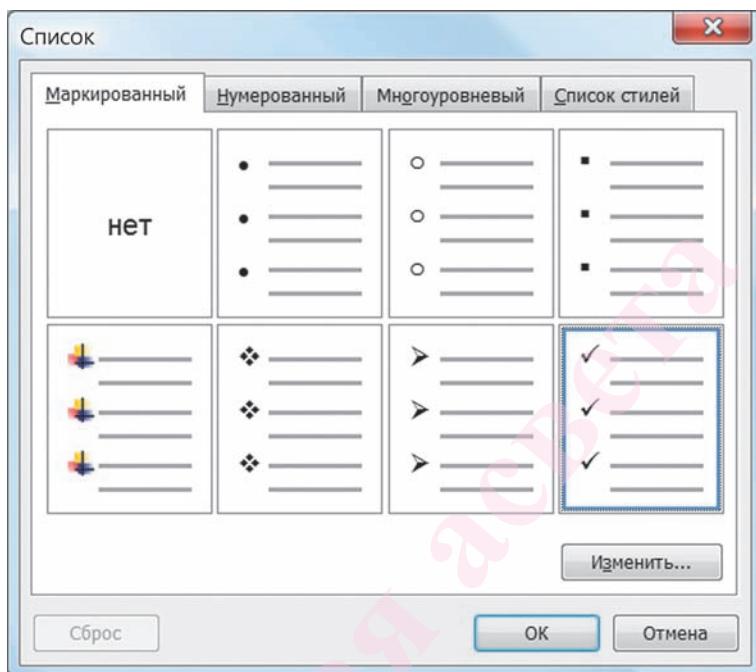


Рис. 2.6

Набрав текст первого элемента, нажатием клавиши Enter перейдем к созданию следующего элемента.

! В программе Word действует правило: каждый элемент списка — это отдельный абзац.

Закончив набор списка, перейдем к следующей строке текста. Теперь необходимо отказаться от продолжения набора списка. Для этого в окне Список выберем вариант оформления Нет (см. рис. 2.6). Список построен.

Удобный способ завершить набор списка — в конце последней строки списка дважды нажать клавишу Enter.

6.2. Нумерованный список

Маркированные списки удобны, если порядок перечисления элементов не имеет значения. В самом деле, нужную для похода посуду можно перечислять в любом порядке. Если порядок перечисления элементов важен, используют другой тип списка — *нумерованный*. В нумерованном списке каждый элемент получает свой индивидуальный номер.

Пример 2. Построить нумерованный список дней недели.

Выбрав команду меню **Формат → Список...**, откроем окно **Список**. На вкладке **Нумерованный** выберем подходящий вариант оформления списка. Наберем «Понедельник» и нажмем клавишу Enter. Аналогично наберем остальные дни недели. Чтобы завершить набор списка, дважды нажмем клавишу Enter. Список построен (рис. 2.7).

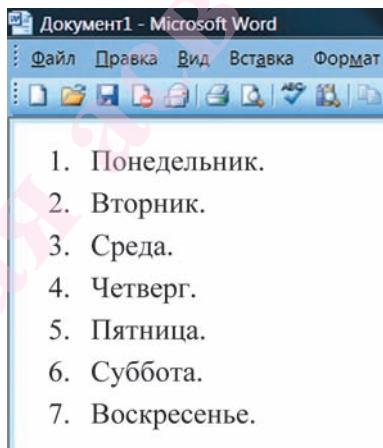


Рис. 2.7

6.3. Вложенные списки

Списки, в которых все или некоторые элементы являются списками, называются *вложенными*.

Пример 3. Построить список крупных населенных пунктов, расположенных на трассе Гомель — Брест и вблизи нее.

Населенные пункты вдоль трассы Гомель — Брест

1. Гомельская область.

 1.1. Гомель.

- 1.2. Речица.
- 1.3. Калинковичи.
- 1.4. Петриков.
- 1.5. Житковичи.
2. Брестская область.
- 2.1. Микашевичи.
- 2.2. Лунинец.
- 2.3. Пинск.
- 2.4. Иваново.
- 2.5. Дрогичин.
- 2.6. Кобрин.
- 2.7. Брест.

Вариант оформления первого элемента списка выберем на вкладке **Многоуровневый** окна **Список** (рис. 2.8).

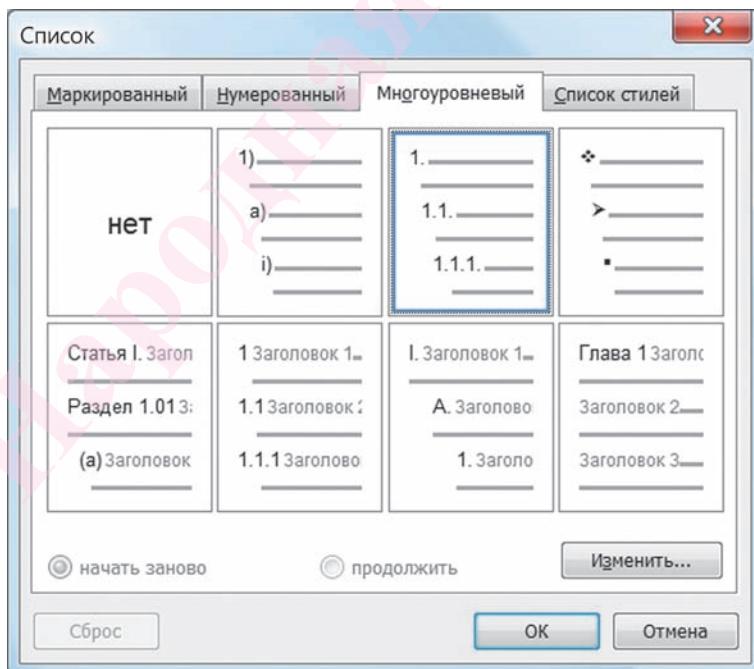


Рис. 2.8

Закончив ввод первого элемента, перейдем к созданию следующего. Этот элемент должен находиться на втором уровне списка. Нажмем клавишу Tab для перехода на уровень ниже. При вводе элемента списка «Брестская область» перейдем на уровень выше, нажав комбинацию клавиш Shift + Tab.

На панели **Форматирование** есть кнопки, которые можно использовать при наборе и редактировании списков (рис. 2.9).



Нумерованный список по умолчанию

Рис. 2.9

- ? 1. Что такое список?
2. Какие виды списков Вам известны?

Упражнения

1. Составьте маркированный список «Семь чудес света». К ним относят древнеегипетские пирамиды, храм Артемиды в Эфесе, мавзолей в Галикарнасе, террасные «висячие сады» Семирамиды, статую Зевса в Олимпии, Колосса родосского, маяк в Александрии.

2. Постройте на белорусском языке двухуровневый список: первый уровень — названия времен года, второй уровень — названия месяцев.

3. Постройте список населенных пунктов на маршруте от вашего дома до ближайшего областного центра или опишите маршрут движения из столицы в любой областной центр. Список подготовьте в двух вариантах: на русском и белорусском языках.

4. Составьте вложенный список «Примеры команд языка Pascal»:

1. Простые.
 - 1.1. Присваивания.

2. Составные.

2.1. Ветвления.

2.1.1. Полная форма.

2.1.2. Сокращенная форма.

2.2. Повторения.

2.2.1. С предусловием.

2.2.2. С параметром.

Элементы списка, не имеющие подчиненных элементов, дополните записью команд.

§ 7. Создание и форматирование таблиц

7.1. Таблица

Не всегда структуру данных удается наглядно представить в виде списка. В таких случаях может помочь использование *таблицы*.

Таблица представляет собой прямоугольник, разделенный на клетки. Клетки таблицы называют *ячейками*. Ячейки, расположенные друг под другом, образуют *столбец*, а ячейки, расположенные друг за другом, образуют *строку* таблицы.

Как правило, первая строка таблицы используется для записи названий столбцов. В первом столбце часто записывают названия строк таблицы.

Имена строк и столбцов задаются не случайным образом. Они описывают содержание строки или столбца таблицы.

Каждая ячейка в таблице расположена на пересечении строки и столбца с определенными именами. Ее содержимое соответствует одновременно признакам, описанным в именах строки и столбца. Так, в таблице «Сведения о странах» на пересечении строки с именем «Беларусь» и столбца с именем «Столица» указано «Минск».

Сведения о странах

Страна	Столица	Площадь, тыс. кв. км	Денежная единица
Беларусь	Минск	208	Белорусский рубль
Латвия	Рига	65	Лат
Литва	Вильнюс	65	Литовский лит
Польша	Варшава	313	Злотый
Россия	Москва	17 075	Рубль
Украина	Киев	604	Грифна

7.2. Создание и редактирование таблиц

Для построения таблиц, их редактирования и форматирования используются команды меню **Таблица**.

Пример 1. Построить таблицу «Сведения о странах».

Наберем заголовок и выберем команду **Таблица** → **Вставить** → **Таблица**. В окне **Вставка таблицы** установим **Число столбцов: 4** и **Число строк: 2** (рис. 2.10). Выберем **Автоподбор ширины столбцов: по содержимому**, чтобы ширина столбцов подбиралась автоматически.

Введем имена столбцов и сведения о Беларуси. Выделив строку с заголовками столбцов, зададим выравнивание по центру и жирное начертание. Аналогично для текста во второй строке установим обычное начертание и выравнивание по левому краю. В ячейке с площадью — выравнивание по правому краю.

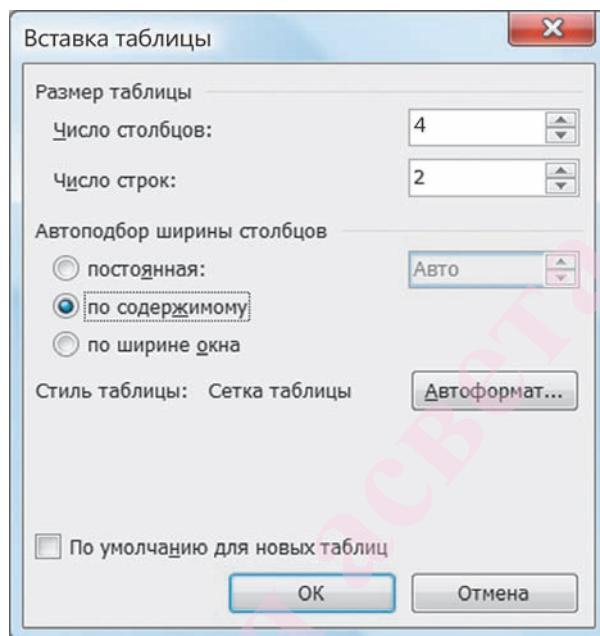


Рис. 2.10

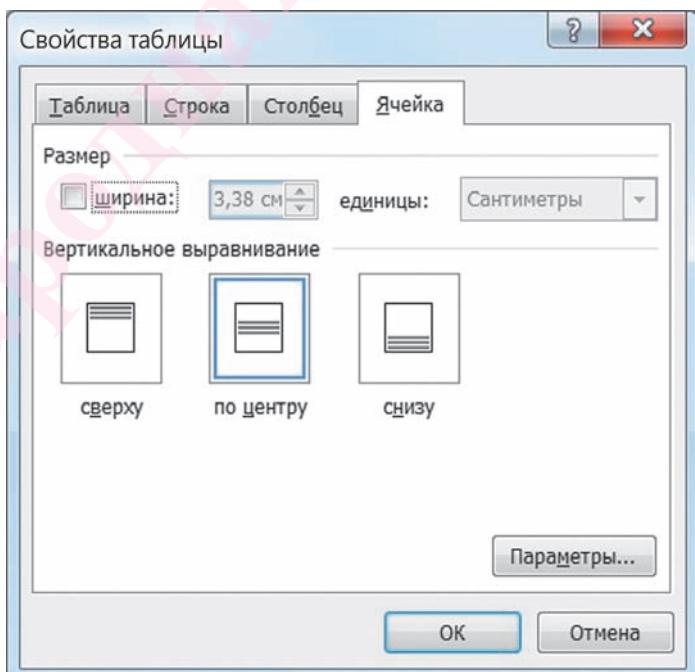


Рис. 2.11

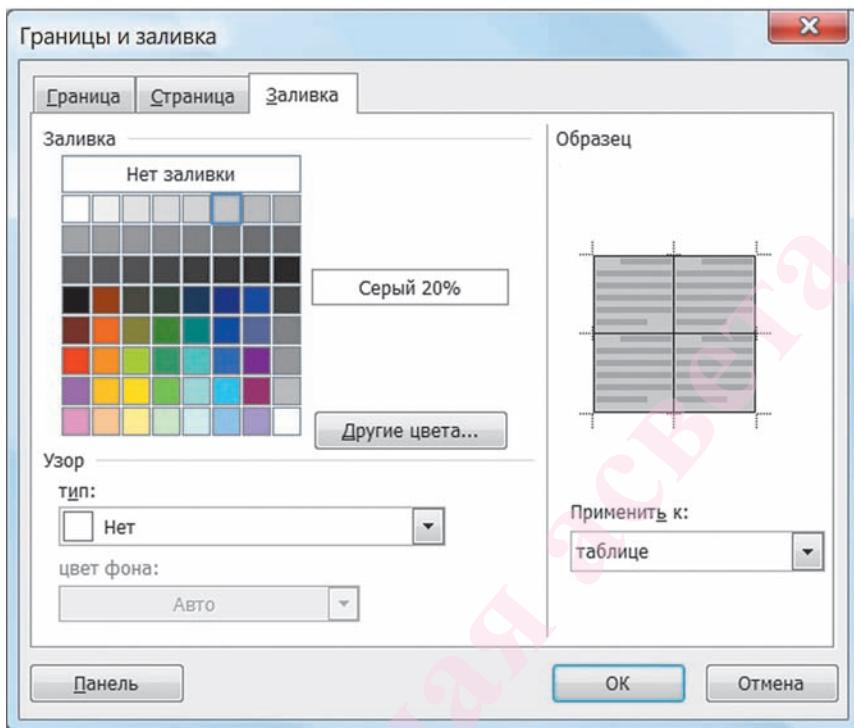


Рис. 2.12

Выделим все ячейки и выполним выравнивание текста по высоте ячеек. Для этого, выбрав команду **Таблица → Свойства таблицы...**, на вкладке **Ячейка** открывшегося окна выберем **Вертикальное выравнивание: по центру** (рис. 2.11). Для изменения цвета строки заголовка выделим ее. Выбрав команду **Формат → Границы и заливка...**, откроем окно **Границы и заливка**. На вкладке **Заливка** выберем светло-серый цвет (рис. 2.12).

Поставим курсор в конец второй строки и нажмем клавишу Tab. Получим новую строку в таблице. Заполним ее. В конце строки еще раз нажмем клавишу Tab. Так постепенно сформируем всю таблицу.

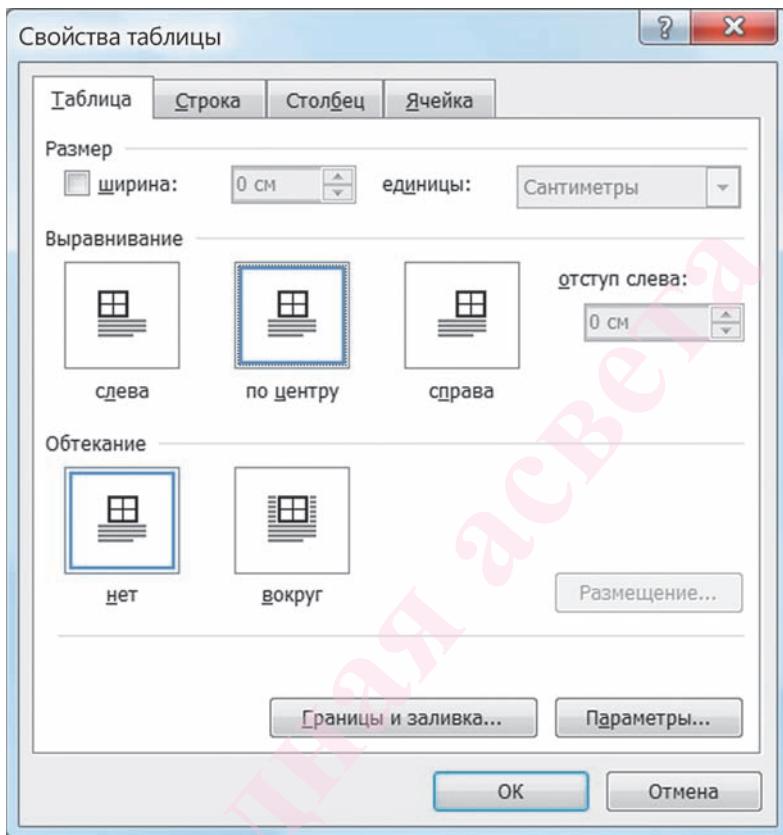


Рис. 2.13

Готовую таблицу выровняем по центру страницы. Для этого на вкладке **Таблица** окна **Свойства таблицы** выберем **Выравнивание: по центру** (рис 2.13). Таблица построена.

В некоторых случаях нужна ячейка, которая занимает несколько столбцов или строк таблицы. В этом случае несколько ячеек объединяют в одну. Для этого группу соседних ячеек выделяют и применяют команду **Таблица → Объединить ячейки**.

Иногда требуется обратное — одну ячейку разделить на несколько. Для этого к выбранной ячейке

применяют команду **Таблица → Разбить ячейки...**. Затем в открывшемся окне указывают требуемое количество строк и столбцов.

Пример 2. Построить таблицу:

Канчаткі прыметнікаў у множным ліку

Склон	Аснова											
	на цвёрды зычны			на мяккі зычны			на зацвяр- дзелы зычны			на ə, к, х		
	ж	м	н	ж	м	н	ж	м	н	ж	м	н
Н.	-ыя			-ія			-ыя			-ія		
Р.	-ых			-іх			-ых			-ія		
Д.	-ым			-ім			-ым			-ім		
В.	-ыя, -ых			-ія, -іх			-ым, -ых			-ія, -іх		
Т.	-ымі			-імі			-ымі			-імі		
М.	-ых			-іх			-ых			-іх		

Создадим таблицу из 5 столбцов и 9 строк. В первом столбце выделим ячейки с первой по третью и выполним команду **Таблица → Объединить ячейки**. В полученной ячейке запишем «Склон». Изменим направление текста командой **Формат → Направление текста...**. В первой строке объединим оставшиеся ячейки для слова «Аснова». В третьей строке каждую ячейку столбцов 2—5 разобьем на три. Для этого выделим вторую ячейку, выберем команду **Таблица → Разбить ячейки...**. В открывшемся окне **Разбиение ячеек** укажем **Число столбцов: 3** и **Число строк: 1**, нажмем **OK**. Аналогично разобьем остальные ячейки третьей строки. Далее заполним таблицу.

7.3. Оформление таблицы

В оформлении таблицы, как правило, избегают большого количества разделительных линий, ярких цветов и изобилия цветовых оттенков. Обычно используется выравнивание текста по левому краю. Столбцы цифр выравнивают по правому краю, так они читаются лучше.

Пример 3. Представить в виде таблицы сведения о планетах Солнечной системы.

Планеты Солнечной системы

Назва- ние	Расстояние от Солнца		Период обраще- ния, сут	Масса, масс Земли
	Афелий, млн км, а. е.	Перигелий, млн км, а. е.		
Меркурий	69,7 0,47	45,9 0,31	88	0,06
Венера	109 0,73	107,4 0,72	225	0,82
Земля	152,1 1,02	147,1 0,98	365	1,00
Марс	249,1 1,67	206,7 1,38	687	0,11
Юпитер	815,7 5,46	740,9 4,95	4332	317,83
Сатурн	1507 10,09	1347 9,02	10 759	95,18
Уран	3003,6 20,08	2735 18,38	30 744	14,53
Нептун	4537 30,44	4455,9 29,77	60 190	17,14

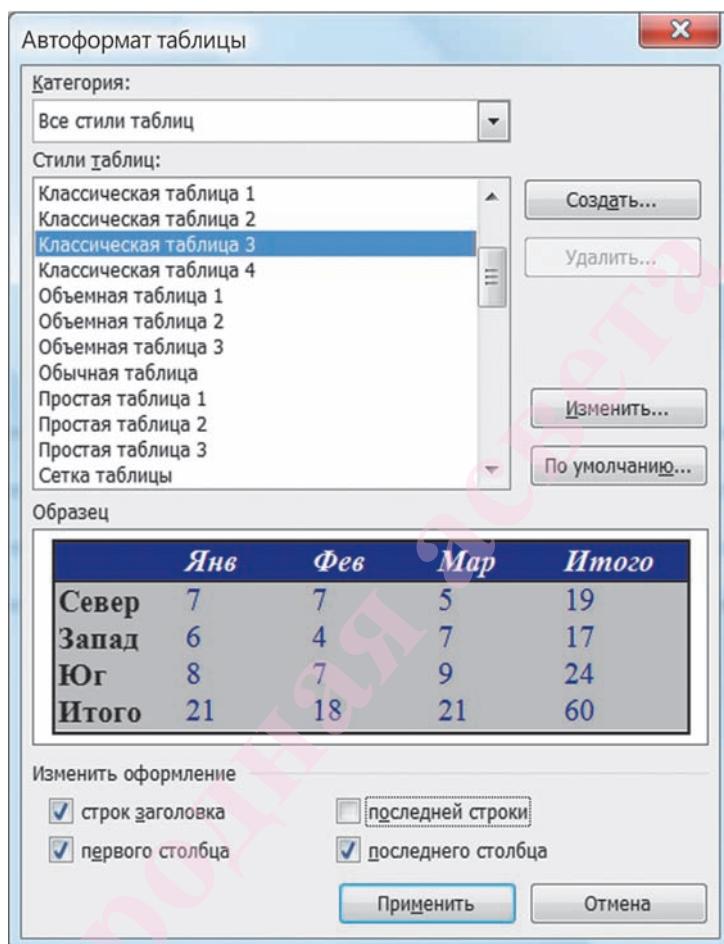


Рис. 2.14

Построив и заполнив таблицу, выберем команду **Таблица → Автоформат таблицы...**. В окне **Автоформат таблицы** (рис. 2.14) выберем **Стиль таблицы: Классическая таблица 3**. В разделе **Изменить оформление** выключим флажок **последней строки**. Таблица будет оформлена в соответствии с образцом. Для того чтобы оформление заголовка применилось ко всем ячейкам заголовка, выделим их и выберем команду **Таблица → Заголовки**.

Поскольку числа в таблице разной длины, выбранный стиль нуждается в небольшой правке. Выделив все ячейки с числами, выберем выравнивание текста по правому краю.

При построении и редактировании таблиц можно пользоваться панелью инструментов **Таблицы и границы**.

Упражнения

1. Создайте текстовый документ, содержащий таблицу «Океаны Земли». Сведения в таблице расположите в алфавитном порядке по названиям океанов.

Океаны Земли

Океан	Площадь, млн кв. км
Атлантический	91,6
Северный Ледовитый	15
Тихий	179
Индийский	76

2. Создайте текстовый документ, содержащий таблицу физических величин.

Величина	Единица измерения			
	Название	Обозначение		Десятичные кратные и дольные единицы
		междуна- родное	русское	
Длина	Метр	<i>m</i>	м	миллиметр сантиметр декиметр километр

Добавьте в таблицу сведения о единицах измерения времени, скорости, массы, плотности, силы, работы.

3. Создайте по образцу текстовый документ, содержащий таблицу некоторых химических элементов.

<i>Элемент</i>	<i>Химический знак элемента</i>	<i>Произношение химического знака в формуле</i>	<i>Относительная атомная масса</i>
<i>Водород</i>	H	аш	1
<i>Азот</i>	N	эн	14
<i>Кремний</i>	Si	силициум	28
<i>Железо</i>	Fe	феррум	56
<i>Ртуть</i>	Hg	гидрагирам	201
<i>Свинец</i>	Pb	плюмбум	207

§ 8. Создание и форматирование колонок

В газетах и журналах часто можно видеть, что текст статьи печатается в виде нескольких колонок. В текстовом процессоре Word документ или некоторые части документа также можно оформить в виде колонок.

Пример 1. Разбить на две колонки текст документа.

Выберем команду **Формат → Колонки... .** В окне **Колонки** укажем **Тип: две** (рис. 2.15). Текст документа будет разбит на две колонки.

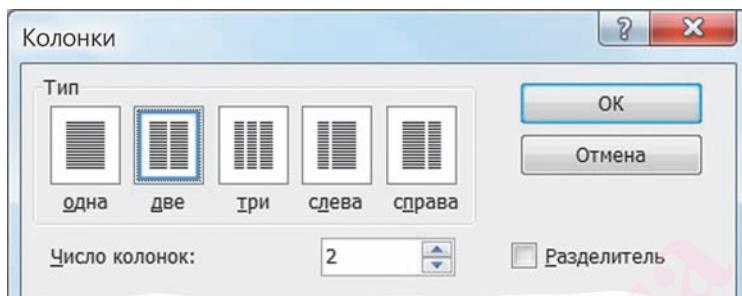


Рис. 2.15

Если для части документа надо изменить параметры форматирования страницы, то эту часть делают отдельным разделом, а затем форматируют созданный раздел. Деление документа на разделы выполняется вставкой разрыва в требуемых местах текста.

Пример 2. Создать документ, в котором часть текста разбита на колонки.

Избранные задачи

Григорий Остер

Задача 20

В бублике 1 дырка, а в кренделе — в два раза больше. На сколько меньше дырок в 7 бубликах, чем в 12 кренделях?

Задача 172

Шерлок Холмс на каждой 12 страницах разоблачает трех преступников. Сколько преступников разоблачит он на 108 страницах?

Задача о задачнике

В задачнике несколько сотен задач. Номер последней задачи неизвестен. Сколько задач в задачнике, если из чисел 157 и 152 одно получено вычитанием из известного номера задачи известного, а другое — вычитанием из неизвестного номера задачи известного?

Наберем текст последовательно: название, имя автора, задачу 20, задачу 172 и задачу о задачнике. Разбиение на колонки нужно применять не ко всему тексту, а к его части, поэтому надо создать разделы в документе. Для этого текстовый курсор установим перед заголовком «Задача 20» и выберем команду **Вставка** →

→ **Разрыв...**. В окне **Разрыв** найдем вариант **на текущей странице** (рис. 2.16). Такой же разрыв вставим перед заголовком «Задача о задачнике». Таким образом документ будет разделен на три раздела.

Затем поставим курсор в любое место второго раздела и командой **Формат** → **Колонки...** оформим созданный раздел в виде двух колонок. Вторая колонка не обязательно будет начинаться с заголовка задачи. Поэтому перед заголовком «Задача 172» вставим вариант разрыва **новую колонку**. Затем командой **Формат** → **Колонки...** изменим ширину колонок так, чтобы в колонках было примерно одинаковое число строк.

- ? 1. Можно ли разделить на колонки весь текст документа?
- 2. Можно ли разделить на колонки только часть документа?
- 3. Можно ли сформировать колонки разной ширины?

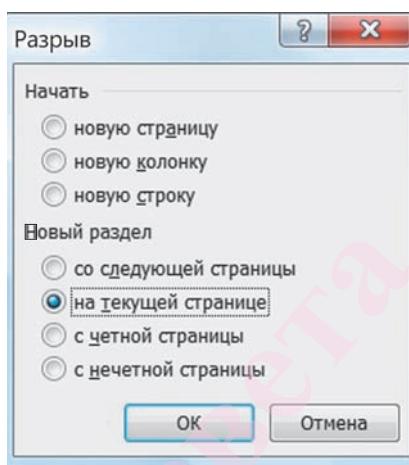


Рис. 2.16

Упражнения

1. Создайте документ, в котором текст разбит на две колонки. В левой колонке поместите список известных Вам кислот, а в правой — щелочей.
2. Создайте текстовый документ, содержащий список учеников вашего класса. Список разделите на две части: ученики, родившиеся до 1-го июля, и ученики, родившиеся после 1-го июля. Каждую часть списка разместите в отдельной колонке.
3. Разработайте англо-белорусский разговорник по образцу.

Англо-белорусский разговорник

Английский вариант

Hello, John.

What is your name?

What do you do?

How old are you?

What is your occupation?

Where do you live?

I live in Minsk.

Белорусский эквивалент

Прывітанне, Джон.

Як Вас завуць?

Чым Вы займаецца?

Колькі Вам гадоў?

Кім Вы працуеце?

Дзе Вы жывяце?

Я жыву ў Мінску.

Добавьте в разговорник 10 фраз на английском языке и их перевод.

Если Вы изучаете другой иностранный язык, разработайте разговорник для него.

§ 9. Вставка декоративного текста

9.1. Создание декоративного текста

Текстовый процессор предлагает массу возможностей оформления текста. Для большинства текстов их достаточно. Иногда требуется художественное

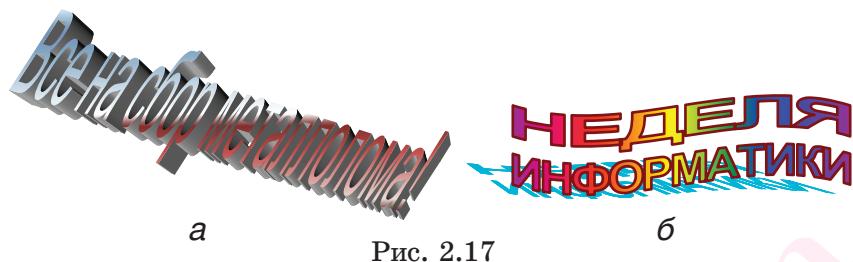


Рис. 2.17

оформление, например в объявлениях (рис. 2.17). В программах Microsoft Office для этого используется объект WordArt.

Пример 1. Создать надпись, как на рисунке 2.17, а.

Выберем команду Вставка → Рисунок → Объект WordArt... . Среди образцов оформления Коллекция WordArt выберем подходящий стиль (рис. 2.18).



Рис. 2.18

В открывшемся окне **Изменение текста WordArt** наберем текст **Все на сбор металлолома!** и нажмем **OK**. Щелчком мыши по созданной надписи выделим ее — появится рамка с маркерами. Перетаскивая угловые или боковые маркеры, подберем подходящую высоту и ширину надписи. Командой **Формат → → Объект WordArt...** откроем соответствующее окно. На вкладке **Размер** укажем **поворот: 25°** и нажмем **OK**. Надпись готова.

9.2. Использование декоративного текста

При выделении созданного объекта открывается панель инструментов WordArt (рис. 2.19).

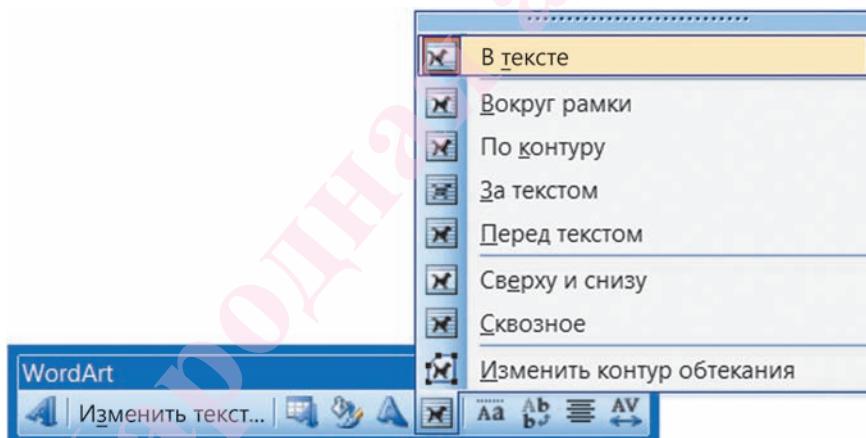
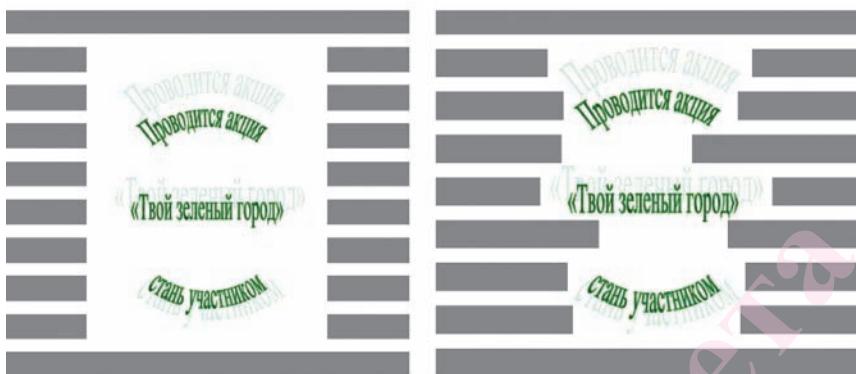


Рис. 2.19

С ее помощью можно изменить положение созданного объекта относительно текста. Например, текст может обтекать добавленный объект **Вокруг рамки** (рис. 2.20, а) или **По контуру** (рис. 2.20, б).

Совместным использованием нескольких объектов WordArt можно добиваться различных выразительных эффектов (рис. 2.21).



а

б

Рис. 2.20

Пример 2. Создать надпись, как на рисунке 2.21, б.

Для вставки объекта WordArt выберем из коллекции левый верхний образец стиля (см. рис. 2.18). Наберем текст «СТОЙ», выберем жирное начертание и нажмем ОК. Выделим созданную надпись. Для изменения формы надписи на панели инструментов WordArt нажмем кнопку . В открывшемся меню выберем вариант . Нажав кнопку панели WordArt, откроем соответствующее окно. На вкладке Цвета и линии выберем красный цвет заливки. На вкладке Размер введем высоту и ширину по 3 см. На вкладке Положение выберем вариант Перед текстом и нажмем ОК.



а

б

Рис. 2.21

Скопируем слово «СТОЙ» и вставим две его копии. Поместим надписи друг под другом. Выделяя по очереди копии надписи, поменяем их цвет на желтый и зеленый. Для изменения текста выделим надпись и выберем на панели WordArt пункт **Изменить текст...**. Слово «ВНИМАНИЕ» наберем в три строки.

С нажатой клавишей Ctrl выделим все три надписи. Правой кнопкой мыши откроем контекстное меню и выберем команду **Группировка** → **Группировать**. Теперь весь текст можно передвигать и изменять как единое целое. Создание надписи завершено.

- ! При работе с текстом не забывайте, что текст предназначен для чтения. Это значит, что используемые приемы оформления не должны мешать восприятию текста.

Упражнения

1. Создайте текстовый документ, содержащий данные о строительстве жилья в Беларуси в 2009 г.



Для наглядности отображения данных используйте декоративный текст.

2. Используя декоративный текст, оформите в текстовом документе название Вашей школы в двух вариантах: на белорусском и русском языках.

3. Создайте текстовый документ с макетом новогодней открытки или открытки ко Дню учителя.

4. С помощью декоративного текста создайте на белорусском языке документ с объявлением об экскурсии в Ветковский музей народного творчества. При подготовке документа используйте следующие сведения:

Веткаўскі музей народнай творчасці
Адрас: Красная Плошча, 5, 247120,
г. Ветка, Гомельская вобл., Беларусь
Музей можна наведаць з 9.00 да 18.00
Выходны дзень — панядзелак
Даведкі па тэлефоне +375 23 302 18 47
www.vetkamuseum.iatp.by

§ 10. Рисунки в документе

10.1. Вставка рисунков в документ

Умелое использование иллюстраций делает документ более живым. Рисунки можно добавлять из коллекции клипов, из файла, с помощью камеры или сканера. Также можно копировать рисунки из различных приложений и вставлять их в документ, используя буфер обмена.

Пример 1. Набрать текст стихотворения и проиллюстрировать его.

Яшчэ на ўзлессі снег ляжыць,
 Але шпакі ўжо штабечуць,
 Звіняць па трубах
 капяжы.
 І кроплі падаюць
 на плечы.
С. Грахоўскі



Наберем текст. Установив текстовый курсор в начало стихотворения, вставим картинку из файла командой **Вставка → Рисунок → Из файла...**. В открывшемся окне **Добавление рисунка** выберем файл с сюжетом весеннего леса.

Выделив рисунок, изменим его размер. Затем командой **Формат → Рисунок...** откроем окно **Формат рисунка**. На вкладке **Положение** выберем обтекание **вокруг рамки** и выравнивание по правому краю (рис. 2.22). Уточним положение рисунка, перетаскивая его мышью.

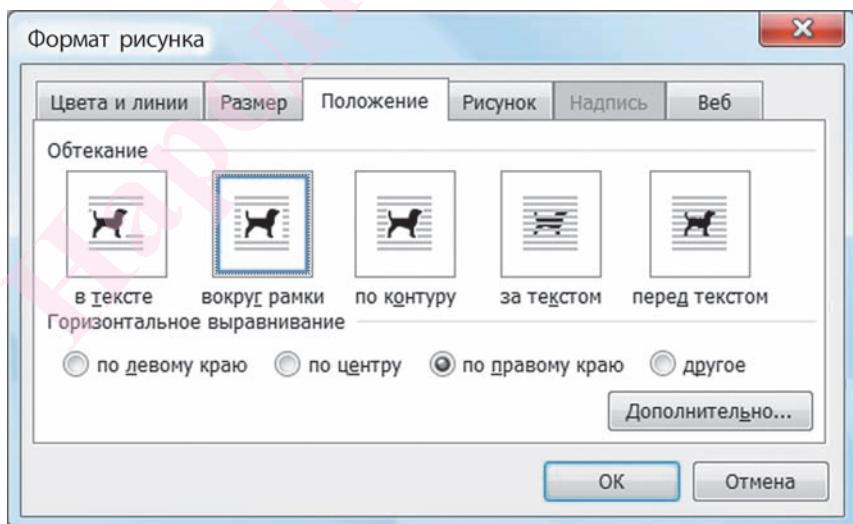


Рис. 2.22

10.2. Использование коллекции клипов

Командой **Вставка** → **Рисунок** → **Картинки...** на панели задач открывается меню **Коллекция клипов**. Мы уже знакомились с коллекцией клипов при подготовке презентаций в программе PowerPoint. Выбор и вставка картинок в текстовый документ выполняются аналогично. Набор доступных картинок зависит от установленных коллекций.

Пример 2. Подготовить эскиз плаката-приглашения на костюмированный бал (рис. 2.23).



Рис. 2.23

Создадим два объекта WordArt: «Бал-маскарад» и «каждый будет рад». Выберем для каждого вариант обтекания **Вокруг рамки** и расположим их друг под другом. Выполнив команду **Вставка** → **Рисунок** → **Картинки...**, из коллекции клипов выберем подходящий по сюжету рисунок. Выделим его и выберем обтекание **Вокруг рамки** (рис. 2.24). С помощью

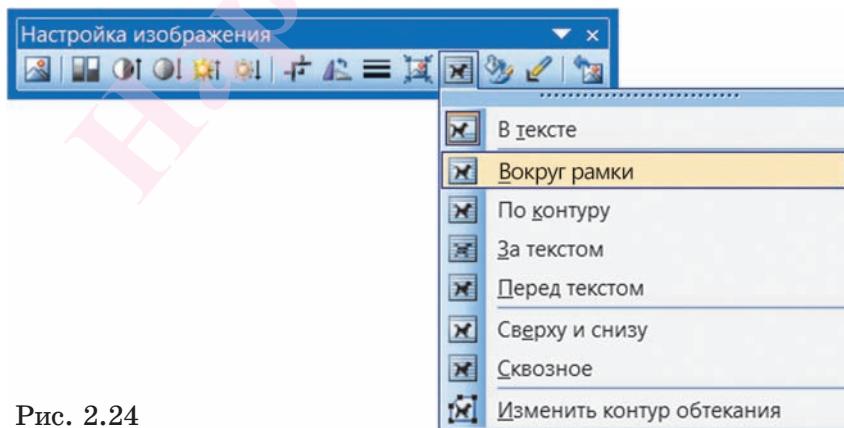


Рис. 2.24

мыши поместим рисунок слева от надписей. Перетаскивая кружочки-маркеры, изменим размер и угол поворота картинки.

Скопируем и вставим картинку. Поместим копию справа от надписей и изменим угол поворота. Рисунок готов.

Если композиция состоит из нескольких элементов, и они накладываются друг на друга, нужная часть изображения может оказаться скрытой. В таких случаях меняют порядок расположения элементов относительно друг друга. Для этого щелчком правой кнопкой мыши по частично скрытой картинке открывают контекстное меню и выбирают команду **Порядок → На передний план**.

В композиции из нескольких объектов удобно использовать группировку, чтобы получить точное взаимное расположение рисунков, художественного текста и основного текста. Для этого с нажатой клавишей Ctrl выделяют группируемые объекты. В контекстном меню выбирают команду **Группировка → Группировать** (рис. 2.25). Таким способом можно собирать композиции из большого числа элементов.

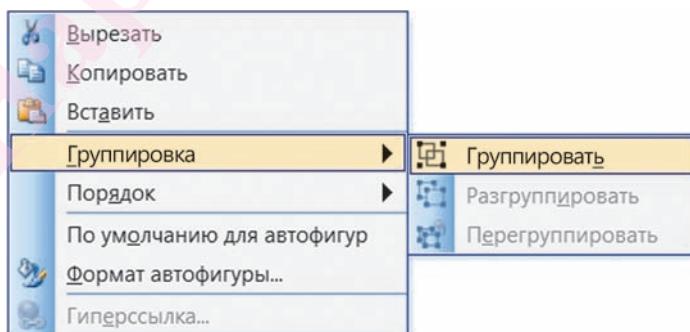


Рис. 2.25

Пример 3. Оформить стихотворение по образцу.



Непрыметна праляцела лета.



Апусцеў за вёскаю палетак.



Вечер распранае ў садзе дрэвы.



Не гучаць птушыныя напевы.



I. Муравейка



Наберем текст стихотворения. Используя объект WordArt, создадим заголовок «Восенню» подходящего размера. Выберем для надписи вариант обтекания Вокруг рамки. Командой Вставка → Рисунок → → Картинки... из коллекции клипов выберем осенний пейзаж. Выделим его и установим обтекание Вокруг рамки. Мышью переместим пейзаж к заголовку и с помощью контекстного меню поместим его на задний план. Сгруппируем пейзаж и заголовок. Выделим группу и командой Формат → Объект... откроем соответствующее окно. На вкладке Положение выберем обтекание по контуру. Перемещая группу, добьемся нужного взаимного расположения заголовка и стихотворения.

Аналогично сгруппируем несколько экземпляров осенних листиков из коллекции. Им также назначим обтекание По контуру. Передвигая рисунки, добьемся требуемого взаимного расположения текста стихотворения и других объектов.

Упражнения

1. Создайте текстовый документ, содержащий эскиз плаката-поздравления победителей предметных олимпиад.



2. Создайте текстовый документ, содержащий таблицу со сведениями о странах.

Страна	Флаг	Столица	Площадь, тыс. кв. км	Населе- ние, тыс. чел.
Европа				
Беларусь		Минск	208	9900
Велико- британия		Лондон	244	60 100

Продолжение

Страна	Флаг	Столица	Площадь, тыс. кв. км	Населе- ние, тыс. чел.
Германия		Берлин	357	82 700
Франция		Париж	552	60 700
Азия				
Индия		Дели	3288	1 132 000
Китай		Пекин	9561	1 318 000
Япония		Токио	372	127 800

§ 11. Вставка формул**11.1. Простые формулы**

В текстах технического и научного характера часто используются формулы. Рассмотрим возможности текстового процессора Word для вставки формул.

Пример 1. Записать в виде формулы свойство прямоугольных треугольников $a^2 + b^2 = c^2$.

Наберем текст « $a^2 + b^2 = c^2$ ». Затем для символов a , b , c выберем курсивное начертание. Выделим двойки в формуле, командой **Формат → Шрифт** назначим видоизменение **Надстрочный**. Формула готова.

Пример 2. Записать уравнение реакции разложения озона.

Наберем текст « $2O_3 = 3O_2$ ». Выделим индексы в формулах озона и кислорода и командой **Формат →**

→ Шрифт назначим видоизменение Подстрочный. Получим формулу в виде: $2\bar{O}_3 = 3\bar{O}_2$.

Пример 3. Записать формулу длины окружности радиуса R : $L = 2\pi R$.

Запись формулы начнем с набора текста « $L = 2R$ ». Затем для символов L и R выберем курсивное начертание. Установим курсор после цифры 2. Для вставки символа π используем команду Вставка → Символ..., которая открывает окно Символ (рис. 2.26). В поле Шрифт выберем значение (обычный текст). Для быстрого поиска символа π в поле Набор выберем тему греческий основной. Щелкнув мышью по символу « π » в таблице и нажав кнопку Вставить, получим формулу в виде: $L = 2\pi R$.

Пример 4. Записать формулу окисления углерода.

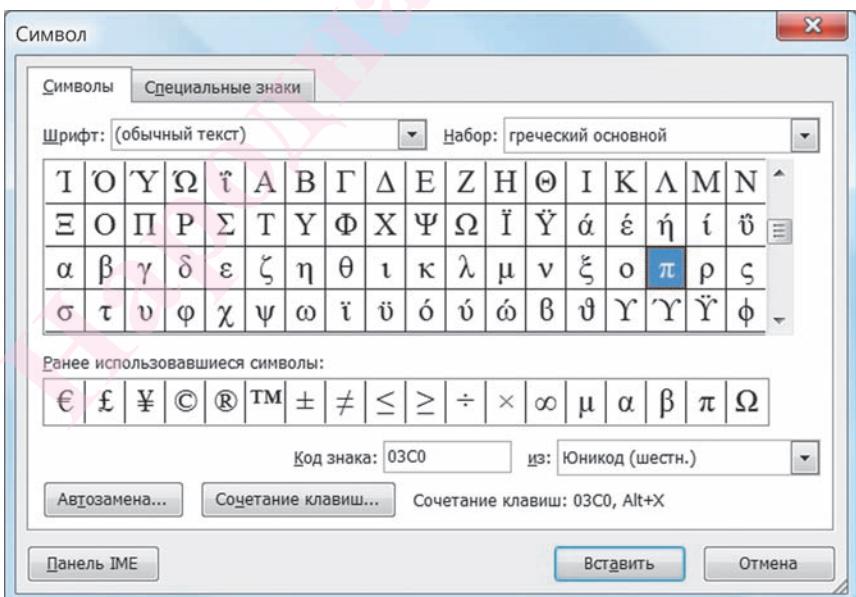


Рис. 2.26

Наберем текст «C + O₂ = CO₂». В формулах кислорода и углекислого газа индексам назначим видоизменение **Подстрочный**. Так как реакция идет с образованием газа, добавим в конце формулы стрелку «↑». Для этого в поле **Шрифт** окна **Символ** установим значение **(обычный текст)**, а в поле **Набор** — значение **стрелки**. Двойным щелчком по стрелке нужного вида в таблице символов вставим ее в текст. В итоге получим: C + O₂ = CO₂↑.

11.2. Объект Microsoft Equation

Для ввода формул служит объект Microsoft Equation. Построение формулы начинается с команды **Вставка** → **Объект...**, открывающей окно **Вставка объекта**. На вкладке **Создание** выбирается тип объекта Microsoft Equation (рис. 2.27).

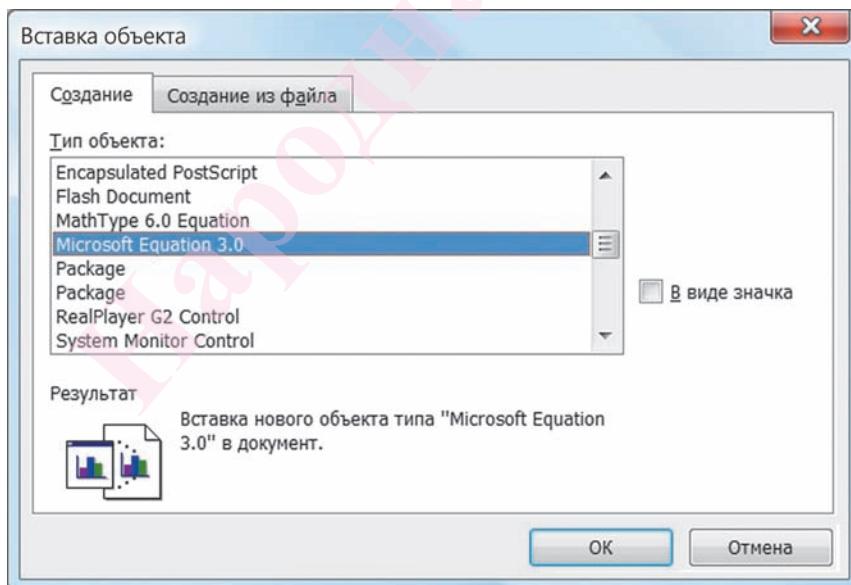


Рис. 2.27



Рис. 2.28

При этом в тексте документа открывается окно для ввода формулы и появляется панель инструментов **Формула** (рис. 2.28). При наборе формул цифры, буквы, знаки «+», «-» и т. д. вводятся с клавиатуры. Панель **Формула** служит для ввода специфических знаков, например \Leftrightarrow , \Rightarrow , \in , \pm , \cup , и выбора шаблонов элементов формулы.

Пример 5. Записать в виде формулы «Одна вторая плюс одна третья равно пять шестых».

Командой **Вставка → Объект...** вызовем объект Microsoft Equation. Ввод дроби начнем с выбора шаблона дроби в группе **Шаблоны дробей и радикалов** (рис. 2.29, а). В окне ввода формулы добавится шаблон дроби (рис. 2.29, б). В числителе дроби напишем «1», нажмем стрелку вниз, курсор переместится в знаменатель. Наберем в знаменателе «2» (рис. 2.29, в). Затем нажмем стрелку вправо, с клавиатуры наберем знак «+» и выберем еще один шаблон дроби. Аналогично продолжим набор формулы (рис. 2.29, г, д).

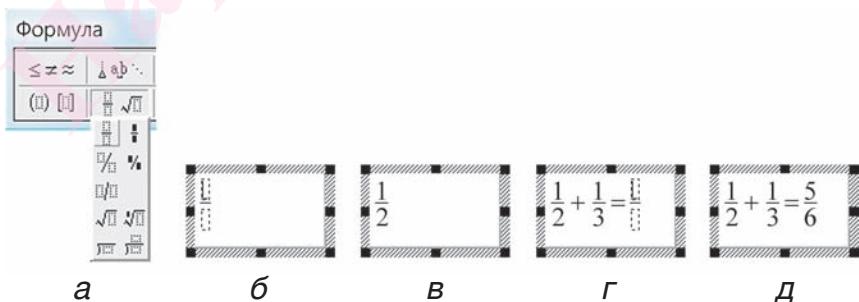


Рис. 2.29

Щелчком левой кнопкой мыши вне окна формулы завершим ее ввод. В итоге получим: $\frac{1}{2} + \frac{1}{3} = \frac{5}{6}$.

Набранные ранее формулы можно редактировать. Для этого двойным щелчком мыши на изображении формулы вызывается редактор формул Microsoft Equation.

Пример 6. Записать правило вычисления площади квадрата со стороной a_1 и объема куба с ребром a_1 .

Наберем текст «Площадь квадрата со стороной a_1 составляет». Для символа « a » выберем курсивное начертание, для « 1 » — видоизменение **Подстрочный**. В продолжении строки вставим объект Microsoft Equation. В формуле наберем « $S = a$ » (рис. 2.30, *а*). В группе **Шаблоны верхних и нижних индексов** выберем шаблон с верхним и нижним индексами . В окошке для верхнего индекса наберем « 2 » (рис. 2.30, *б*), а для нижнего — « 1 ». Получим запись $S = a_1^2$ (рис. 2.30, *в*).

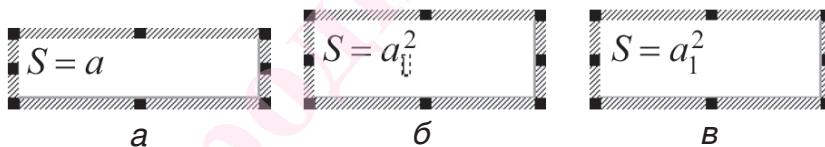


Рис. 2.30

Щелкнем мышью вне окна формулы. Наберем текст «Объем куба с ребром». В первом предложении скопируем « a_1 составляет $S = a_1^2$ ». Двойным щелчком мыши по скопированной формуле перейдем к ее редактированию. В формуле заменим « S » на « V » и показатель степени « 2 » на « 3 ». В итоге получим:

Площадь квадрата со стороной a_1 составляет $S = a_1^2$.

Объем куба с ребром a_1 составляет $V = a_1^3$.

Пример 7. Представить условие задачи о поиске точки пересечения двух прямых $y = 4x - \frac{3}{5}$ и $y = -2\frac{1}{3}x + 6$ в виде системы уравнений

$$\begin{cases} 4x - y = \frac{3}{5}, \\ 2\frac{1}{3}x + y = 6. \end{cases}$$

Набор формулы начнем со вставки объекта Microsoft Equation. Далее выберем шаблон системы  в группе Шаблоны скобок (рис. 2.31, а, б).

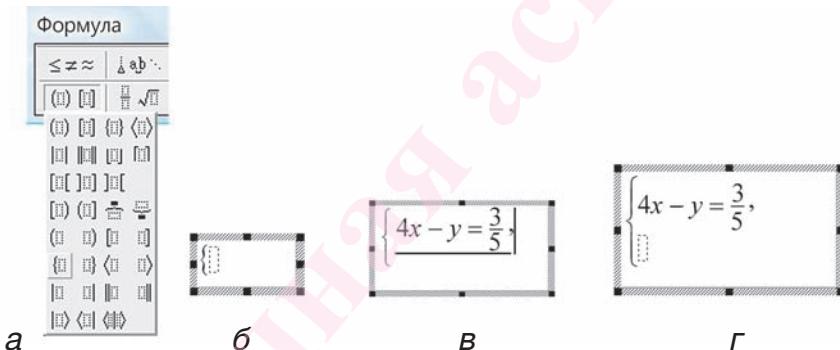


Рис. 2.31

Наберем формулу $4x - y = \frac{3}{5}$. Для набора $\frac{3}{5}$ вставим шаблон дроби. Затем, нажав стрелку вправо, переместим курсор в конец строки, поставим запятую (рис. 2.31, в) и нажмем клавишу Enter для создания второй строки системы (рис. 2.31, г). Во второй строке также потребуется шаблон дроби для ввода $\frac{1}{3}$.

- ?
- 1. Можно ли добавлять в текст символы, отсутствующие на клавиатуре? Приведите примеры таких символов.
- 2. Всегда ли для записи формулы необходим объект Microsoft Equation? Приведите примеры.

Упражнения

1. Наберите текст.

Среднее арифметическое n чисел можно найти по формуле

$$a_{\text{ср}} = \frac{a_1 + a_2 + \dots + a_n}{n}.$$

Пример. Найти среднее арифметическое чисел 5, 6, 7.

Решение. $\frac{5+6+7}{3} = 6.$

2. Составьте таблицу.

Свойства степеней	
$(a \cdot b \cdot c)^2 = a^2 \cdot b^2 \cdot c^2$	$\left(\frac{a}{b}\right)^2 = \frac{a^2}{b^2}$
Свойства арифметических квадратных корней	
$\sqrt{a \cdot b \cdot c} = \sqrt{a} \cdot \sqrt{b} \cdot \sqrt{c}$	$\sqrt{\frac{a}{b}} = \frac{\sqrt{a}}{\sqrt{b}}$

3. Наберите текст. Определите, где требуется использовать вставку формул. Перечислите использованные шаблоны элементов формулы.

а) Корни квадратного уравнения $ax^2 + bx + c = 0$ можно находить по формуле $x_{1,2} = \frac{-b \pm \sqrt{D}}{2a}$, где $D = b^2 - 4ac$. Уравнение имеет два корня, если дискриминант D больше нуля ($D > 0$).

б) Квадратное уравнение $ax^2 + bx + c = 0$ можно привести к виду $x^2 + px + q = 0$. В этом случае корни квадратного уравнения находят по формуле $x_{1,2} = -\frac{p}{2} \pm \sqrt{\left(\frac{p}{2}\right)^2 - q}$. Уравнение имеет два корня, если выполняется условие $\left(\frac{p}{2}\right)^2 - q > 0$.

4. Наберите текст.

Общее сопротивление двух последовательно включенных резисторов находят по формуле $R = R_1 + R_2$. Общее сопротивление двух параллельно включенных резисторов можно найти из соотношения $\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2}$. Используя эти формулы, можно производить расчет сопротивления различных схем.

Пусть параллельно соединены две цепи. Первая состоит из последовательно включенных резисторов r_1 и r_2 . Вторая — из последовательно соединенных резисторов r_3 и r_4 . Общее сопротивление такой цепи $R = \frac{(r_1 + r_2)(r_3 + r_4)}{r_1 + r_2 + r_3 + r_4}$.

§ 12. Нумерация страниц и подготовка документов к печати

12.1. Формат бумаги и поля

Большинство документов готовится с целью переноса их на бумагу.

Подготовка документа к печати начинается с выбора формата бумаги. Чаще всего для печати используются листы формата А4. Размер такого листа 210×297 мм. Это размер листа обычных альбомов для черчения. Также важна ориентация бумаги. Обычно используют *книжную* ориентацию. При этом лист повернут так, что высота листа больше его ширины. В некоторых случаях используют *альбомную* ориентацию, когда ширина листа больше его длины.

Пример 1. Установить для текущего документа размер бумаги А4 и книжную ориентацию.

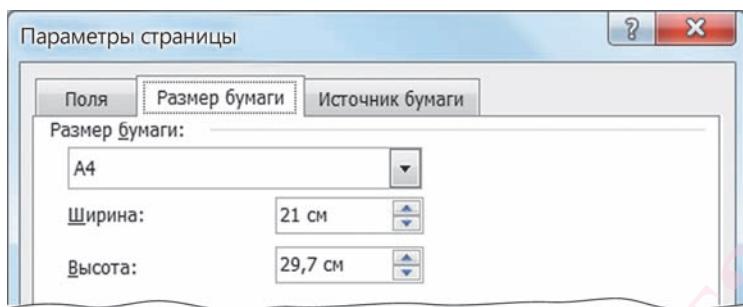


Рис. 2.32

Для этого выберем команду **Файл** → **Параметры страницы...**. На вкладке **Размер бумаги** окна **Параметры страницы** найдем в выпадающем списке размер бумаги A4 (рис. 2.32).

На вкладке **Поля** установим ориентацию бумаги **книжная** (рис. 2.33). Требуемые установки выполнены.

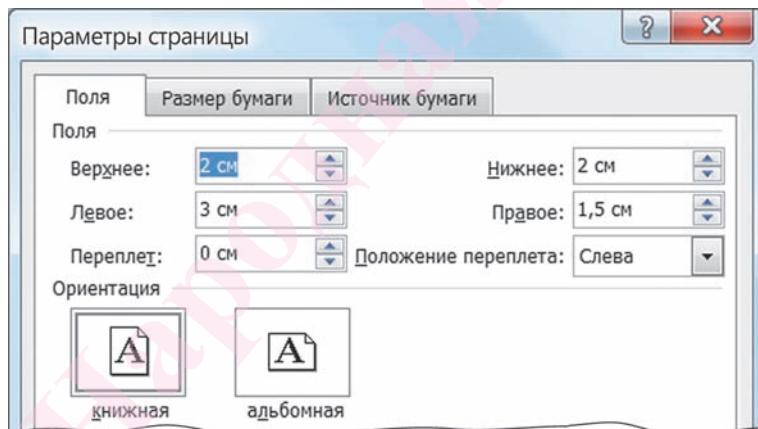


Рис. 2.33

Текст документа почти никогда не печатают от самых краев листа по нескольким причинам. Часто левый или верхний край листов используется для переплета. Свободное пространство по краям позволяет при просмотре документа не закрывать руками

текст. К тому же не все модели принтеров могут печатать от самого края листа.

Свободное пространство по краям листа называют **полями**. Часто используют следующие размеры полей:

- левое поле — 30 мм;
- правое поле — 10 мм;
- верхнее поле — 15 мм;
- нижнее поле — 20 мм.

Устанавливают размеры полей на вкладке **Поля** окна **Параметры страницы** (см. рис. 2.33).

Пример 2. Установить для текущего документа величину верхнего и нижнего полей — по 2 см, левого поля — 3 см, а правого — в 2 раза меньше.

С помощью команды **Файл** → **Параметры страницы...** откроем вкладку **Поля** окна **Параметры страницы**. Наберем значения полей: **Верхнее: 2, Нижнее: 2, Левое: 3, Правое: 1,5**. Требуемая величина полей задана.

12.2. Колонтитулы и нумерация страниц

Повторяющийся вверху и (или) внизу всех или большинства страниц однотипный текст называют **колонтитулами**. Часто встречающиеся варианты колонтитула: название рассказа, главы, раздела, фамилия автора. Самый распространенный пример колонтитула — номера страниц.

Пример 3. Установить для текущего документа нумерацию всех страниц, кроме первого титульного листа, справа вверху.

Выберем команду **Вставка** → **Номера страниц...**. В открывшемся окне **Номера страниц** (рис. 2.34) укажем **Положение: Вверху страницы**, а **Выравни-**

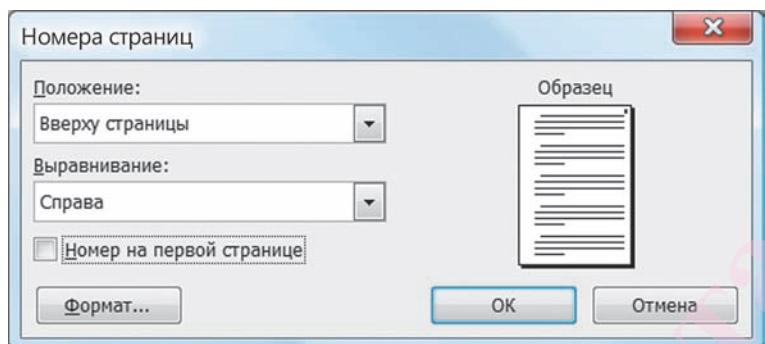


Рис. 2.34

вание: Справа. Для того чтобы на титульной странице номер не ставился, выключим флажок **Номер на первой странице**. Нажмем **ОК**.

12.3. Средство расстановки переносов

Текст, выровненный по ширине, без переносов в словах смотрится некрасиво из-за большой разницы в величине промежутков между словами. На бумаге неравномерная плотность текста заметна больше, чем на экране.

Для автоматической расстановки переносов выполняют команду **Сервис → Язык → Расстановка переносов...**. В открывшемся окне **Расстановка переносов** включают флажок **Автоматическая расстановка переносов** (рис. 2.35).

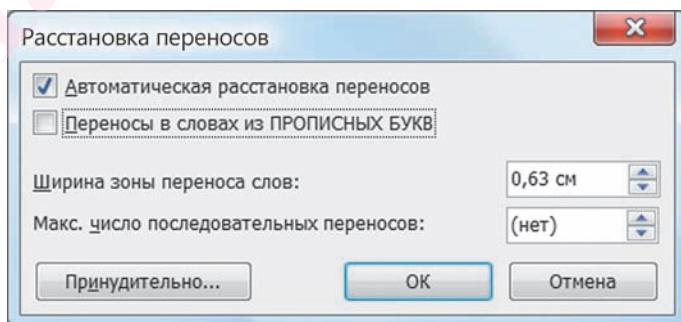


Рис. 2.35

Пример 4. Набрать текст, используя выравнивание по ширине и автоматическую расстановку переносов.

Андрей Рублев достиг непревзойденного мастерства в живописи. В своих изображениях мастер предпочитал особые нежные цвета, именно им впервые использованные в церковной живописи.

Творческая манера Андрея Рублева проявилась в росписи множества храмов, в том числе Успенского собора во Владимире. Всемирную известность приобрела его икона «Троица». Это одно из немногих творений мастера, сохранившихся до нашего времени.

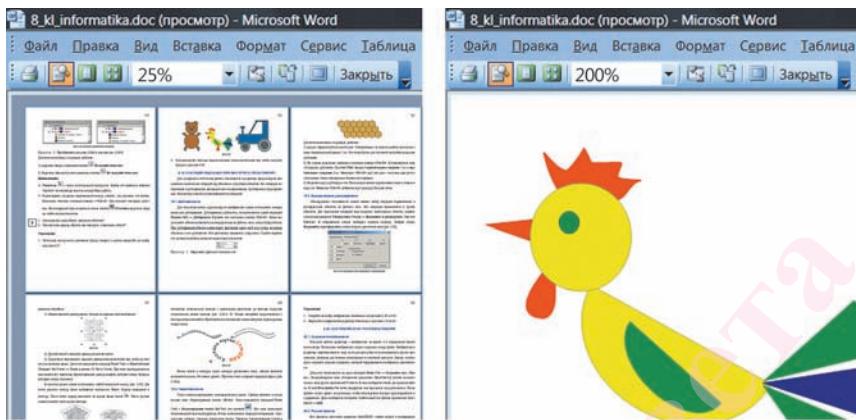
«Троица» Андрея Рублева была написана с нарушением существовавшего тогда иконописного канона (установленных правил). Но уже в XVI в. на церковном соборе она была объявлена образцом.

Наберем текст. Выделим весь текст и зададим выравнивание по ширине, например, командой **Формат → Абзац → Выравнивание по ширине...**. Затем выберем команду **Сервис → Язык → Расстановка переносов...** и в открывшемся окне включим флажок **Автоматическая расстановка переносов** (см. рис. 2.35). Текст оформлен.

Расстановку переносов можно отменить, отключив флажок **Автоматическая расстановка переносов**.

12.4. Предварительный просмотр документа

Перед выводом документа на печать полезно оценить его вид, используя команду **Файл → Предварительный просмотр**. По этой команде документ пока-



а

Рис. 2.36

б

зывается на экране в виде, максимально близком к результату печати. Тем не менее некоторые расхождения возможны, ведь документ не печатается, программа только моделирует результат печати.

Замеченные в режиме просмотра недочеты исправляют, и вновь используют предварительный просмотр. Это экономит бумагу, расходные материалы принтера и время разработчика документа.

Просматривать можно как несколько страниц одновременно, так и фрагмент страницы при большом увеличении (рис. 2.36).

Пример 5. Выполнить предварительный просмотр текущего документа по шесть страниц одновременно и просмотр отдельных фрагментов страницы в масштабе 250 %.

Выполнив команду **Файл → Предварительный просмотр**, перейдем в режим предварительного просмотра. Используя кнопку , выберем показ шести страниц одновременно. В этом режиме можно оценить общий вид страниц документа. Далее в поле

Масштаб наберем 250. Теперь страницы показываются с большим увеличением и можно детально рассмотреть их отдельные фрагменты. Завершим просмотр щелчком мыши по кнопке **Закрыть**.

12.5. Настройка параметров печати

Печать документа выполняется командой **Файл → Печать...**. В окне **Печать** (рис. 2.37) выбираются параметры печати.

К домашнему компьютеру, как правило, подключено единственное печатающее устройство. Компьютер, подключенный к локальной сети, может иметь доступ к нескольким принтерам. Тогда в поле **имя** следует выбрать нужный принтер из выпадающего списка, который содержит все доступные принтеры.

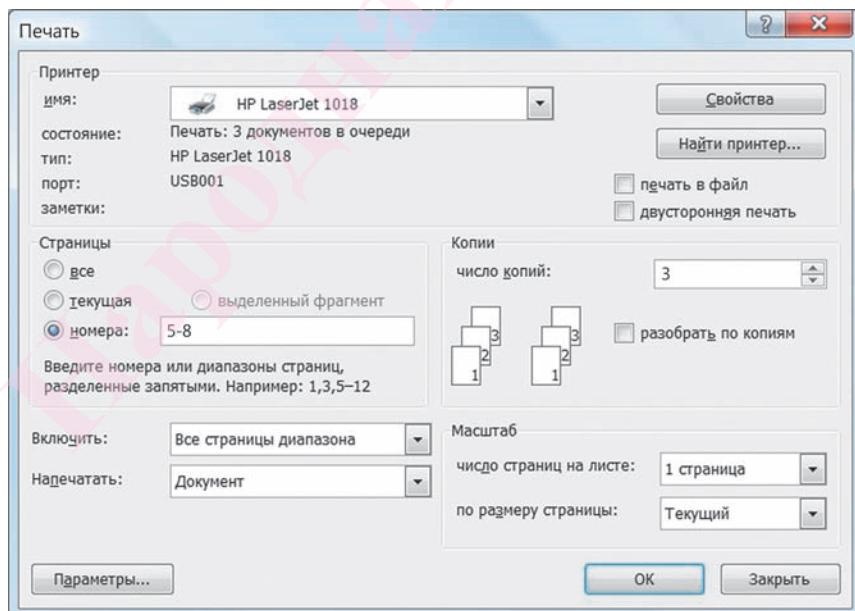


Рис. 2.37

Кнопка **Свойства** открывает доступ к настройкам выбранного принтера.

Можно вывести на печать весь документ или текущую страницу. Возможна выборочная печать страниц. Для этого нужно через запятую перечислить номера страниц или через тире указать диапазоны номеров страниц. Например:

- для печати десятой, пятнадцатой и двадцатой страниц указывают: **10,15,20**;
- для печати диапазона страниц с тридцатой по сотую вводят: **30-100**;
- для печати страниц с первой по десятую, пятнадцатую, двадцатую и диапазона страниц с сороковой по сотую набирают: **1-10,15,20,40-100**.

Пример 6. Напечатать страницы документа с 5-й по 8-ю в трех экземплярах.

Командой **Файл** → **Печать...** откроем окно **Печать**. В поле **номера** зададим диапазон страниц **5-8**, в поле **число копий** — **3**. Затем щелчком мыши по кнопке **OK** отправим документ на печать.

При больших объемах печати стоит выполнять более детальную настройку параметров печати.

Пример 7. Напечатать в черновом варианте титульный лист документа, главу 1 (с. 5—30) и главу 3 (с. 78—106). Печатать по две страницы на листе и без иллюстраций.

Откроем окно **Печать**. В поле **номера** наберем список печатаемых страниц: **1,5-30,78-106**. В поле **число страниц на листе**: выберем **2 страницы**.

Нажав кнопку **Параметры...**, получим доступ к дополнительным настройкам печати (рис. 2.38). Включим флагок **черновой**. Черновая печать выполняется несколько быстрее обычной. Флажки **графи-**

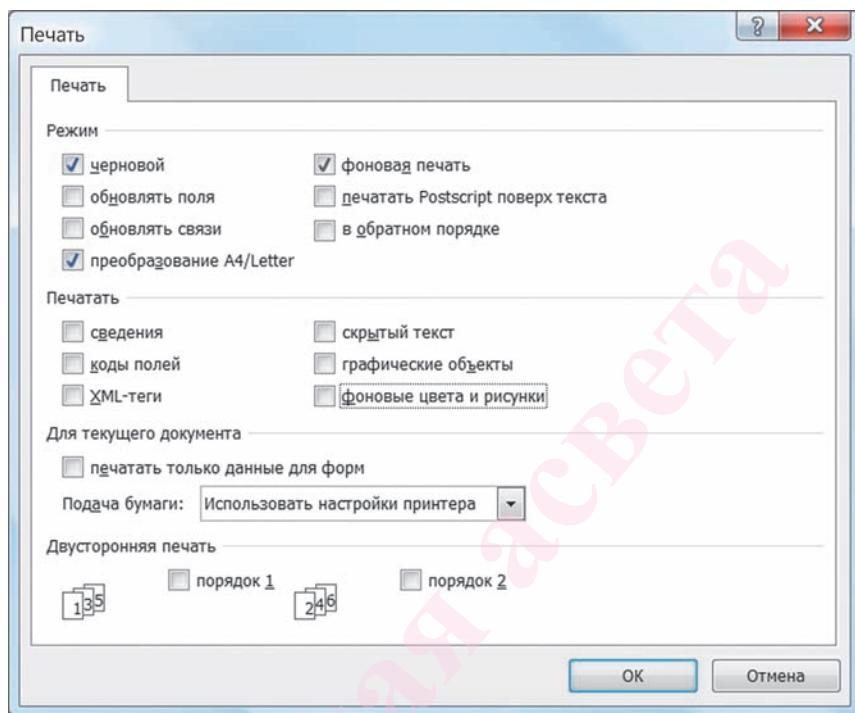


Рис. 2.38

ческие объекты и фоновые цвета и рисунки отключим для экономии расходных материалов. Выполнив настройки, щелчком по кнопке ОК закроем окно.

Отправим документ на печать щелчком мыши по кнопке ОК.

Пример 8. Напечатать документ полностью в 5 экземплярах с разбором по копиям.

Откроем окно Печать. В разделе Страницы выберем вариант все. Включим флашок разобрать по копиям, чтобы сначала напечатались все страницы первого экземпляра, затем второго и так далее. Зададим число копий: 5, чтобы распечаталось 5 экземпляров документа.

Закроем окно щелчком мыши по кнопке ОК. Настройки для печати выполнены, отправим документ на печать.

Прежде чем отправлять какой-либо документ на печать, необходимо убедиться в готовности к работе принтера. Принтер должен быть включен. Также в принтере должна быть бумага подходящего формата в достаточном количестве.

- ?
- 1. С помощью какой команды можно отправить документ на печать?
- 2. Как задается размер бумаги?
- 3. Что такое поля документа и как задают их размер?
- 4. Как задают нумерацию страниц?

Упражнения

1. Откройте указанный учителем документ. Установите для документа книжную ориентацию бумаги и размер полей: левое — 35 мм, правое — 20 мм, верхнее и нижнее — по 15 мм. Задайте нумерацию страниц (кроме первой). Выполните автоматическую расстановку переносов и проверку правописания. Предъявите учителю подготовленный документ в режиме предварительного просмотра.

2. Откройте указанный учителем документ. Сделайте установки для печати первых 10 страниц документа в трех экземплярах с разбором по копиям.

3. Откройте указанный учителем документ. Сделайте установки для печати документа в черновом варианте по 4 страницы на листе без иллюстраций.

4. В документе требуется объединить три небольших рассказа. Тексты рассказов хранятся на диске в трех отдельных файлах. Рассказы надо поместить на лист альбомной ориентации, каждый в отдельной колонке. В текст каждого рассказа необходимо вста-

вить маленький рисунок, относящийся к его сюжету. Рисунки должны иметь одинаковый размер. Размеры полей документа: 25 мм — левое и правое и 15 мм — сверху и снизу. Сделайте установки для печати полученного документа с наилучшим качеством в 5 экземплярах.

§ 13. Применение текстового процессора в разработке документов из различных областей

13.1. Подготовка к работе с новым документом

Цель работы в текстовом редакторе — создание документа. В процессе разработки документов разного назначения или из разных предметных областей есть много общих черт. Так, подготовка документа зачастую состоит из следующих этапов:

- создание и сохранение в файл нового документа;
- установка параметров страницы;
- настройка удобного для работы режима отображения документа в окне редактора;
- набор и форматирование текста документа.

Рассмотрим на примерах этапы построения многостраничного документа, насыщенного различными элементами.

Пример 1. Сделать заготовку для набора документа, содержащего две части: таблицу «Формулы расчета объемов» и текст с формулами «Примеры расчетов».

С помощью команды **Файл → Создать...** выберем пункт **Новый документ**, т. е. формирование документа начнем с «чистого листа». Затем файл документа сохраним, выбрав ему имя и место на диске.

Используя команду **Файл** → **Параметры страницы...**, выберем формат бумаги А4 и книжную ориентацию. Здесь же установим размеры полей документа: левое — 3,5 см, правое — 1,5 см, верхнее и нижнее — по 2,5 см. Далее командой **Сервис** → **Язык** → → **Расстановка переносов...** включим автоматическую расстановку переносов в документе.

Выберем режим отображения документа **Вид** → → **Разметка страниц**. Командой **Сервис** → **Параметры...** откроем окно **Параметры**. Включим показ границ текста, используя флажок **границы текста** на вкладке **Вид** этого окна. Также включим отображение знаков форматирования **все** (рис. 2.39). Удобные для работы установки выполнены.

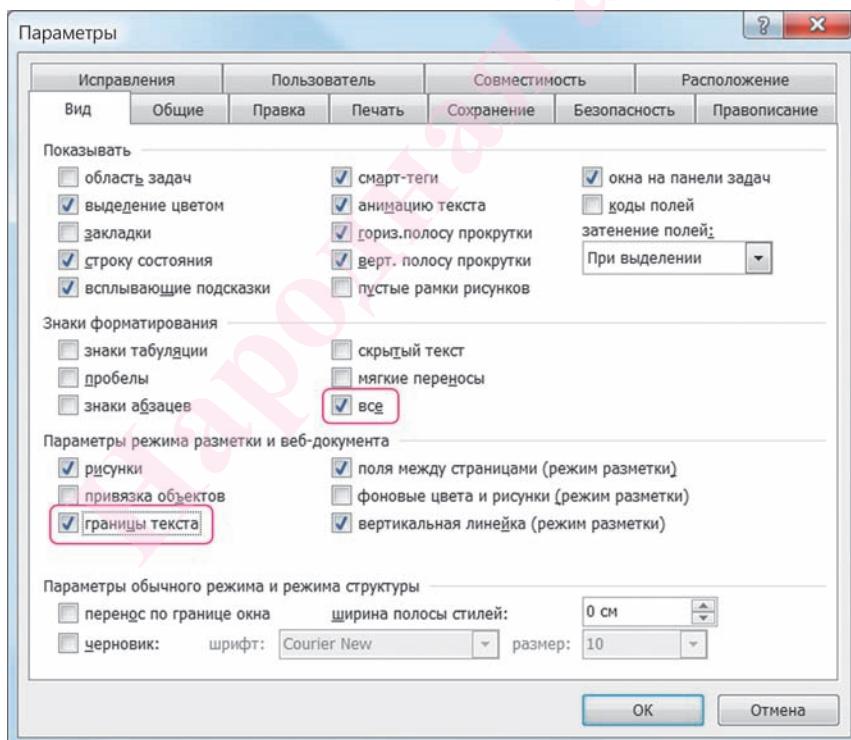


Рис. 2.39

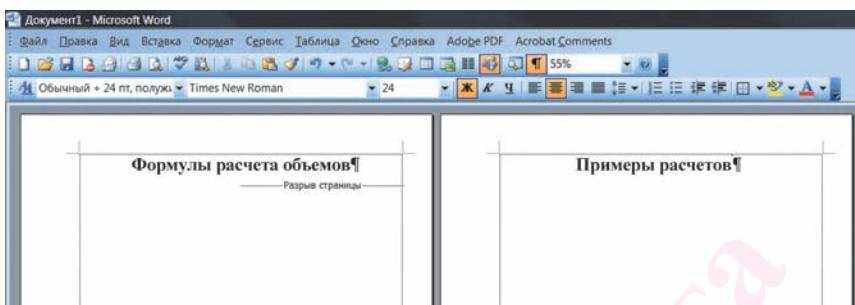


Рис. 2.40

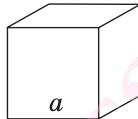
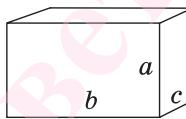
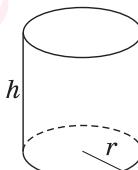
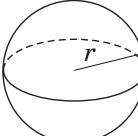
Наберем два абзаца текста: заголовки «Формулы расчета объемов» и «Примеры расчетов». Выделим набранные абзацы. Командой **Формат** → **Шрифт...** откроем диалоговое окно **Шрифт**. Выберем **Начертание: полужирный** и **Размер: 20**. Затем командой **Формат** → **Абзац...** откроем окно **Абзац**. Выберем выравнивание **По центру** и междустroчный интервал — **Полуторный**. На вкладке **Положение на странице** включим флажок **с новой страницы**. Закроем это окно и сохраним изменения в файле, используя комбинацию клавиш Shift + F12. Заготовка документа создана (рис. 2.40).

13.2. Набор и форматирование элементов документа

Качество создаваемого документа и время, затрачиваемое на его разработку, зависят не только от знания команд используемого текстового процессора, но и от выбора рациональной последовательности действий по набору, редактированию и форматированию документа.

Пример 2. Набрать текст, выполнить форматирование и подготовить документ к печати в двух экземплярах с разбором по копиям.

Формулы расчета объемов

Тело	Формула	Чертеж
Обозначения элементов		
Куб, a — ребро куба	$V = a^3$	
Прямоугольный параллелепипед, a, b, c — ребра параллелепипеда	$V = abc$	
Цилиндр, r — радиус основания цилиндра, h — высота цилиндра	$V = \pi r^2 h$	
Шар, r — радиус шара	$V = \frac{4}{3} \pi r^3$	

Примеры расчетов

Задача 1. Определить объем воды в аквариуме, имеющем форму шара радиуса R_1 , если аквариум заполнен водой на 75 %.

Решение. Аквариум представляет собой шар, и его объем равен $V = \frac{4}{3} \pi R_1^3$. Объем воды составляет $V = 0,75 \frac{4}{3} \pi R_1^3 = \pi R_1^3$.

Ответ: $V = \pi R_1^3$.

Задача 2. Определить объем воды в аквариуме размером $a \times b \times c$, если аквариум заполнен водой на 90 %.

Решение. Аквариум представляет собой прямоугольный параллелепипед, и его объем равен $V = abc$. Объем воды составляет $V = 0,9abc$.

Ответ: $V = 0,9abc$.

Используем файл с заготовкой документа из примера 1 этого параграфа. Откроем этот файл командой **Файл → Открыть...** .

Поставим текстовый курсор в конец первого заголовка и нажмем клавишу Enter. Создастся новый абзац на новой странице. Выделим этот абзац и выберем команду **Формат → Абзац...** . На вкладке **Положение на странице** выключим флажок **с новой страницы**. В результате абзац вернется на страницу с заголовком.

Добавим таблицу из 3 столбцов и 3 строк с автоподбором ширины столбцов **по содержимому**. Выровняем таблицу по центру страницы. Две первые строки таблицы будем использовать для заголовков столбцов. Наберем их. Третья строка предназначена для основного текста таблицы.

После каждого этапа работы над документом обязательно будем сохранять его, используя комбинацию клавиш Ctrl + S. Это нужно для того, чтобы при сбоях в работе компьютера или программного обеспечения не потерять всю сделанную работу.

Выделим всю таблицу и назначим размер шрифта 16 пт (команда **Формат → Шрифт...**) и полуторный межстрочный интервал (команда **Формат → Абзац...**). Затем в третьей строке выделим первый

и второй столбцы. Назначим этим столбцам выравнивание по левому краю и отступ слева 0,5 см (команда **Формат → Абзац...**). Наберем в две строки текст «Куб, *a* — ребро куба». Для того чтобы быстро набрать «*a*» курсивом, перед набором символа нажмем комбинацию клавиш Ctrl + I. Набрав символ, снова нажмем комбинацию клавиш Ctrl + I. Для набора тире используем комбинацию клавиш Ctrl + Alt + «минус», причем «минус» нажмем на цифровой клавиатуре. Для набора показателя степени в формуле $V = a^3$ перед вводом цифры «3» нажмем комбинацию клавиш Ctrl + Shift + «плюс».

В третий столбец вставим рисунок. Если нет файлов с готовыми чертежами, то их можно нарисовать в редакторе Paint и перенести в документ копированием через буфер обмена. Для этого в редакторе Paint выделим готовый рисунок и нажмем комбинацию клавиш Ctrl + C. Перейдем к документу Word и нажмем комбинацию клавиш Ctrl + V.

Добавим и заполним остальные строки таблицы. В последней строке для набора формулы объема шара используем объект Microsoft Equation.

Перейдем к созданию второй части документа. Поставим текстовый курсор в конец заголовка «Примеры расчетов» и нажмем клавишу Enter. Создастся новый абзац на новой странице. Выделим этот абзац. Выберем команду **Формат → Абзац...**. Установим выравнивание **По ширине** и межстрочный интервал — **Полутонный**. На вкладке **Положение на странице** выключим флајок **с новой страницы**.

Затем выберем команду **Формат → Шрифт...**. Зададим размер шрифта **16** и начертание **Обычный**. Наберем и отформатируем текст решения за-

дач. Чтобы быстро набрать нижний индекс в записи R_1 , перед вводом «1» нажмем комбинацию клавиш Ctrl + «плюс». Набрав «1», повторно нажмем эту комбинацию клавиш и возвратимся к набору обычного текста. Формулы удобно копировать и редактировать, как это делалось при заполнении таблицы.

Добавим нумерацию страниц, используя команду **Вставка → Номера страниц...**. Сохраним документ и проверим в режиме предварительного просмотра.

Откроем окно **Печать** (команда **Файл → Печать...**). В разделе **Страницы** выберем вариант **все**. Включим флашок **разобрать по копиям**. Зададим число копий: **2**. Щелкнем мышью по кнопке **Параметры...**. В открывшемся окне выключим флашок **черновой** и включим флашок **графические объекты**. Закроем окно щелчком по кнопке **OK**. Настройки для печати выполнены.

- ?
1. Почему в ходе работы надо регулярно сохранять документ?
 2. Можно ли копировать формулы в документе?

Упражнения

1. Наберите текст. Подготовьте документ к печати. Установите размеры полей: левое — 4 см, правое — 2,5 см, верхнее и нижнее — по 2 см. Добавьте нумерацию страниц справа вверху.

Старая батарейка дает слабый ток, и поэтому лампочка горит тускло. Почему это происходит? Разность потенциалов V на контактах и старой, и новой батарейки одинакова.

Сопротивление цепи складывается из сопротивления лампочки R_L и внутреннего сопротивления батарейки r . В цепи с новой батарейкой сила

тока составляет $I_{\text{нов}} = \frac{V}{R_{\text{л}} + r_{\text{нов}}}$, где $r_{\text{нов}}$ — внутреннее сопротивление новой батарейки. Для старой батарейки $I_{\text{ст}} = \frac{V}{R_{\text{л}} + r_{\text{ст}}}$, где $r_{\text{ст}}$ — внутреннее сопротивление старой батарейки. Внутреннее сопротивление новой батарейки намного меньше, чем старой: $r_{\text{нов}} \ll r_{\text{ст}}$. Следовательно, $I_{\text{ст}} \ll I_{\text{нов}}$, и лампочка от новой батарейки должна гореть заметно ярче, чем от старой.

2. Наберите текст. Подготовьте документ к печати в пяти экземплярах в черновом варианте. Установите размеры полей: левое — 2,5 см, правое, верхнее и нижнее — по 1,5 см.

Задача. В прямоугольной пластиине размером $a \times b$ вырезали круглое отверстие радиуса r_1 . В это отверстие вставили заподлицо втулку с внутренним диаметром d_2 . Плотность материала пластины ρ_1 , а материала втулки — ρ_2 . Определить массу получившейся детали, если толщина пластины h .

Решение. Площадь пластины с отверстием составит $S_1 = ab - \pi r_1^2$, а ее объем $V_1 = (ab - \pi r_1^2)h$.

Площадь втулки составит $S_2 = \pi r_1^2 - \frac{\pi}{4}d_2^2$, а объем $V_2 = \left(\pi r_1^2 - \frac{\pi}{4}d_2^2\right)h$. Масса детали составит $M = V_1\rho_1 + V_2\rho_2$.

Выполнив подстановки, получим

$$M = (ab - \pi r_1^2)h\rho_1 + \left(\pi r_1^2 - \frac{\pi}{4}d_2^2\right)h\rho_2.$$

$$\text{Ответ: } M = (ab - \pi r_1^2)h\rho_1 + \left(\pi r_1^2 - \frac{\pi}{4}d_2^2\right)h\rho_2.$$

3. Оформите документ по теме «Вычисление площади», как описано в параграфе. Предъявите учителю установки для печати документа в трех экземплярах с разбором по копиям.

Формулы расчета площадей

Фигура Обозначения элементов	Формула	Чертеж
Круг, r — радиус круга	$S = \pi r^2$	
Квадрат, a — сторона квадрата	$S = a^2$	
Прямоугольник, a, b — стороны прямоугольника	$S = ab$	
Треугольник, a — основание треугольника, h — высота треугольника	$S = \frac{ah}{2}$	

Примеры расчетов

Задача. Определить площадь квадратной пластины со стороной a и вырезом в виде полукруга радиуса $\frac{a}{2}$.

Решение. Площадь пластины с вырезом составит $S = S_1 - S_2$, где S_1 — площадь квадрата, S_2 — площадь выреза (полукруга радиуса $\frac{a}{2}$). Найдя

площадь квадрата $S_1 = a^2$ и площадь полукруга

$$S_2 = \frac{1}{2}\pi\left(\frac{a}{2}\right)^2, \text{ получим:}$$

$$S = a^2 - \frac{1}{2}\pi\left(\frac{a}{2}\right)^2 = a^2\left(1 - \frac{\pi}{8}\right).$$

Ответ: $S = a^2\left(1 - \frac{\pi}{8}\right)$.

4. Создайте и подготовьте к печати документ со сведениями из биографии одного из белорусских писателей. Фамилию, имя, годы жизни и портрет разместите по центру листа. Биографию разместите в две колонки. Раздел «Творчество» представьте в виде таблицы. Раздел «Награды» оформите в виде списка. Пример выполнения упражнения показан на рисунке.

Название	Год
1. Стихотворения	1922
2. Песни	1923
3. Повести	1924
4. Драмы	1925
5. Сказки	1926
6. Песни на зарубежные темы	1929
7. Песни на зарубежные темы, написанные для фестиваля «Славянка» (до 1926), включая «Джинн»	1931
8. Песни на зарубежные темы, написанные для фестиваля «Славянка»	1931
9. Песни на зарубежные темы	1932
10. Песни на зарубежные темы	1933
11. Песни на зарубежные темы	1934
12. Песни на зарубежные темы	1935
13. Песни на зарубежные темы	1936
14. Песни на зарубежные темы	1937
15. Песни на зарубежные темы	1938
16. Песни на зарубежные темы	1939
17. Песни на зарубежные темы	1940
18. Песни на зарубежные темы	1941
19. Песни на зарубежные темы	1942
20. Песни на зарубежные темы	1943
21. Песни на зарубежные темы	1944
22. Песни на зарубежные темы	1945

Узнаграды

- Герой Социалистической Труда (1973)
- заслуженный артист БССР
- орден Красного Знамени
- орден Чаремского Октября
- орден Адрианова оружия II степени
- орден Дружбы народов I степени

Глава 3

Работа с векторной графикой

§ 14. Понятие векторного изображения

14.1. Векторная и растровая графика

Цифровые фотоаппараты, встроенные видео- и фотокамеры сотовых телефонов, широкое распространение Интернета сделали работу с изображениями на компьютере повседневным занятием. Компьютерная графика стала неотъемлемой частью профессиональной деятельности дизайнеров, инженеров, модельеров, архитекторов...

Компьютерная графика — это изображения, подготовленные при помощи компьютера. Компьютер может обрабатывать информацию, представленную только в цифровой форме. Значит, изображение надо представить в цифровой форме. Наиболее распространены два способа представления изображений: **растровый** и **векторный**. Для получения **растрового** представления изображение делят на точки и записывают цвет каждой точки. Для получения **векторного** представления изображение делят на простые фигуры, описывают их характеристики и взаимное расположение.

Так, например, для описания рисунка в виде синего круга на белом фоне в растровом представлении указывается цвет каждой точки как круга, так и фона. Векторное представление использует математическое описание фигуры. Достаточно указать радиус круга, координаты центра, цвет и толщину линии, цвет заливки. В результате размер файла с векторным изображением значительно меньше.

Растровое представление графики обычно используется для изображений фотографического типа с

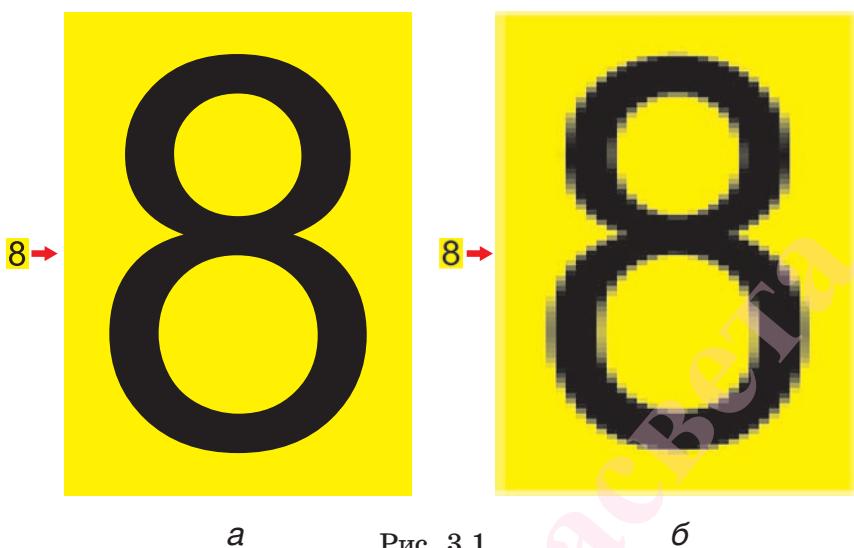


Рис. 3.1

б

большим количеством деталей и оттенков. Масштабирование таких картинок ухудшает их качество. Векторная графика удобна для рисунков, которые не нуждаются в фотorealизме. Масштабирование векторной графики происходит без потери качества. Сравните, например, увеличенное в 16 раз векторное (рис. 3.1, а) и раcтровое (рис. 3.1, б) изображения цифры 8.

14.2. Представление о цветовых моделях

Цвет — это ощущение, которое возникает в сознании человека при воздействии света на его орган зрения. То есть цвет — это прежде всего ощущение, и восприятие цвета зависит от человека. Человеческий глаз различает миллионы цветовых оттенков. Как их описать и воспроизвести? Текстильная и лакокрасочная промышленность, полиграфия и компьютерные технологии нуждаются в единых стандартных способах описания цвета.

Художники для получения нужного оттенка смешивают на палитре множество красок. Однако все многообразие оттенков можно получить смешением нескольких цветов. Это могут быть, например, красный, зеленый и синий цвета или голубой, пурпурный, желтый и черный. Задача состоит в том, чтобы с помощью базовых цветов смоделировать необходимое цветовое ощущение. При воспроизведении цвета используют **цветовые модели** как способ описания цветовых ощущений. Использование цветовых моделей решает задачу единых способов описания цвета. Наиболее популярные цветовые модели — RGB, CMYK, HSB.

В модели **RGB** (Red, Green, Blue) основными являются три цвета: красный, зеленый и синий. Все цветовые оттенки получаются за счет сложения в разных пропорциях этих цветов. Например, монитор компьютера воспроизводит цвета на основе модели RGB. Цвет каждой точки монитора формируется сложением света от трех элементов, каждый из которых светится своим цветом — красным, зеленым и синим. Цвет точки зависит от того, насколько ярко светится каждый из этих элементов. Если яркость всех трех элементов равна нулю, то глаз человека воспринимает цвет точки на мониторе как черный. Если яркость всех трех элементов максимальна, то цвет воспринимается как белый (рис. 3.2). Если яркость одного из элементов равна нулю, то получается желтый, пурпурный или голубой цвет (см. рис. 3.2). Так, желтый цвет формируется максимальной яркостью красного и зеленого элементов и нулевой яркостью синего элемента.

В компьютерных программах значения компонентов RGB изменяются от 0 до 255. Значение 255 соответствует максимальной яркости компонента.

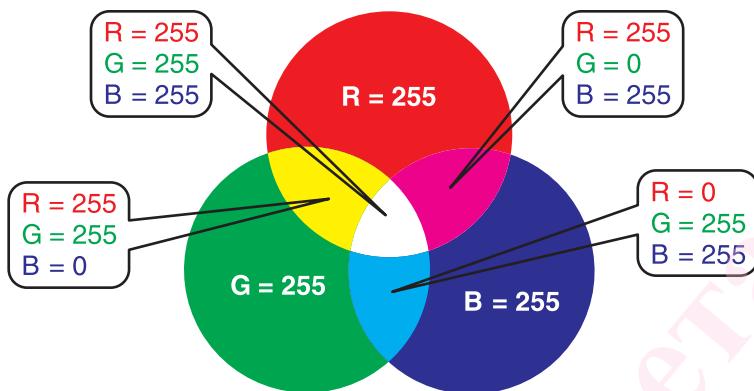
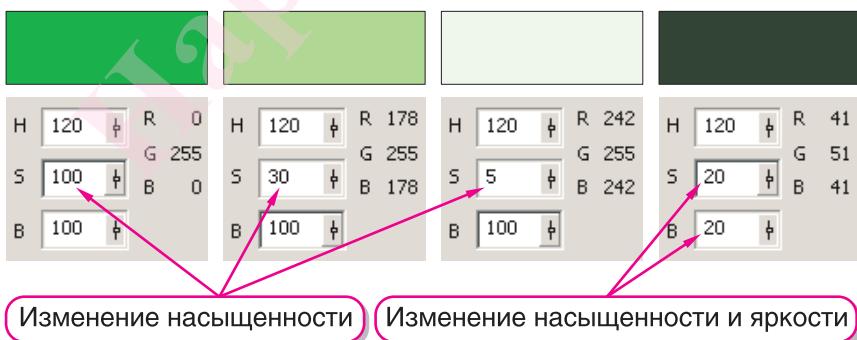


Рис. 3.2

Модель **HSB** наиболее точно соответствует словесному способу описания цвета. В этой модели управляют цветовым тоном (**Hue**), насыщенностью (**Saturation**) и яркостью (**Brightness**). Сравните, как изменяется зеленый цвет при уменьшении насыщенности со 100 % до 30 %, затем до 5 % (рис. 3.3, а). Одновременное уменьшение насыщенности и яркости приближает цвет к черному (рис. 3.3, б).

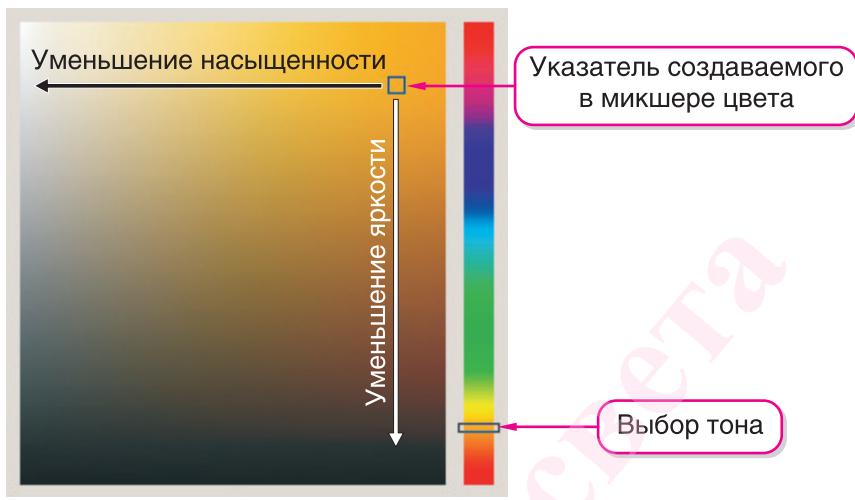
Для описания отраженного и излученного света используют разные цветовые модели. Излученный свет попадает в глаз человека прямо от источника,



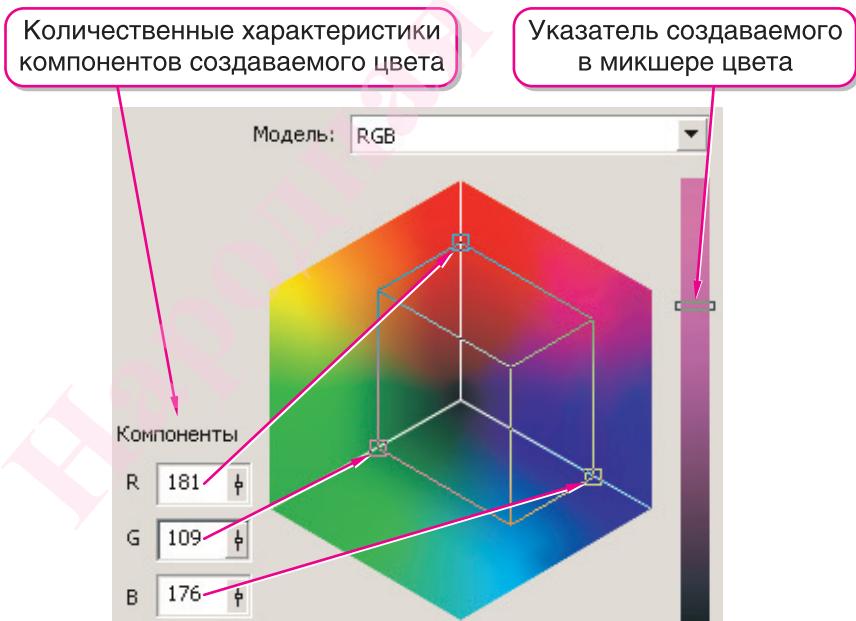
а

Рис. 3.3

б



a



б

Рис. 3.4

например компьютерного монитора или экрана телевизора. Модели RGB, HSB используются для описания излученного света.

В полиграфии работают с отраженным от поверхности (например, бумаги) светом. В этом случае используют другие цветовые модели, например CMYK. В этой модели нужный оттенок получают печатью четырьмя красителями. Цвета красителей — голубой (Cyan), пурпурный (Magenta), желтый (Yellow) и черный (KeyColor — Black). Последовательное нанесение различного количества этих красителей на поверхность изменяет цвет отраженного света и формирует миллионы оттенков.

В графических редакторах предусмотрена возможность выбора необходимой пользователю цветовой модели. При этом нужный цвет можно подбирать из готового набора стандартных цветов — цветовой палитры или создавать самостоятельно. Для самостоятельного создания цвета можно воспользоваться цветовым микшером. Варианты внешнего вида микшеров представлены на рисунке 3.4.

- ?
- 1. Что называют векторной графикой?
- 2. В чем отличие векторной графики от растровой?
- 3. Что называют палитрой цветов графического редактора?
- 4. Какие цвета использует модель RGB для получения нужного оттенка?

§ 15. Векторный графический редактор

15.1. Назначение графического редактора

Напомним, что программное средство для создания и обработки изображений называется **графическим редактором**. Редакторов создано много, но все

они разными средствами решают одни и те же типовые задачи:

- создание изображения;
- редактирование изображения;
- сохранение изображения в виде файла;
- загрузка изображения из файла в оперативную память;
- вывод изображения на носитель, например бумагу, пластик или ткань.

Какой графический редактор выбрать из всего многообразия программ? Ответ будет зависеть от поставленной задачи. Если необходимо обработать фотографию или нарисовать пейзаж, удобно работать в растровом графическом редакторе, например Adobe Photoshop, Corel PHOTO-PAINT. С простым растровым редактором Microsoft Paint мы познакомились в 6-м классе.

Для создания графических компьютерных моделей удобен редактор векторной графики. Наиболее известные векторные графические редакторы — CorelDRAW, Adobe Illustrator, Inkscape, Adobe Flash. Рассмотрим основные приемы создания и редактирования векторных изображений на примере редактора **CorelDRAW**. Одно из его достоинств состоит в том, что интерфейс и основные функции мало отличаются при переходе от одной версии редактора к другой. Мы будем использовать **CorelDRAW X4**.

15.2. Элементы интерфейса графического редактора

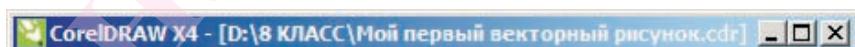
Запустить программу можно двойным щелчком по ярлыку на рабочем столе  или с помощью команды Пуск → Программы → **CorelDRAW Graphics**

Suite X4 → CorelDRAW X4. При запуске программы окно приветствия CorelDRAW предлагает варианты **Быстрого старта** (Quick Start). Можно открыть один из недавно сохраненных файлов (слева от списка имен файлов есть окошко для просмотра содержимого выбранного файла). Можно создать новый пустой документ или открыть заготовку-шаблон для продолжения работы на ее основе.

Для создания нового изображения следует выбрать вариант **Новый пустой документ** (New blank document). После этого открывается рабочее окно редактора CorelDRAW (рис. 3.5). Познакомимся с его основными элементами.

Область, которая занимает практически все пространство окна редактора, — **Рабочее поле**. Оно служит для создания, редактирования и временно-го хранения элементов изображения. На печать выводится только часть изображения, ограниченная **Страницей рисования** — прямоугольником в центре рабочего поля.

Верхняя часть окна — **строка заголовка** с именем документа. Имя новому документу программма присваивает автоматически, например **Рисунок1** (Graphic1). В строке заголовка сохраненного документа отображается полный путь к файлу:



Первая строка под заголовком — **строка меню** с текстовыми командами. Некоторые команды управления документом вынесены в виде кнопок на следующую строку — **Стандартную панель**. Некоторые команды главного меню также доступны через контекстные меню выделенных элементов. В пунктах

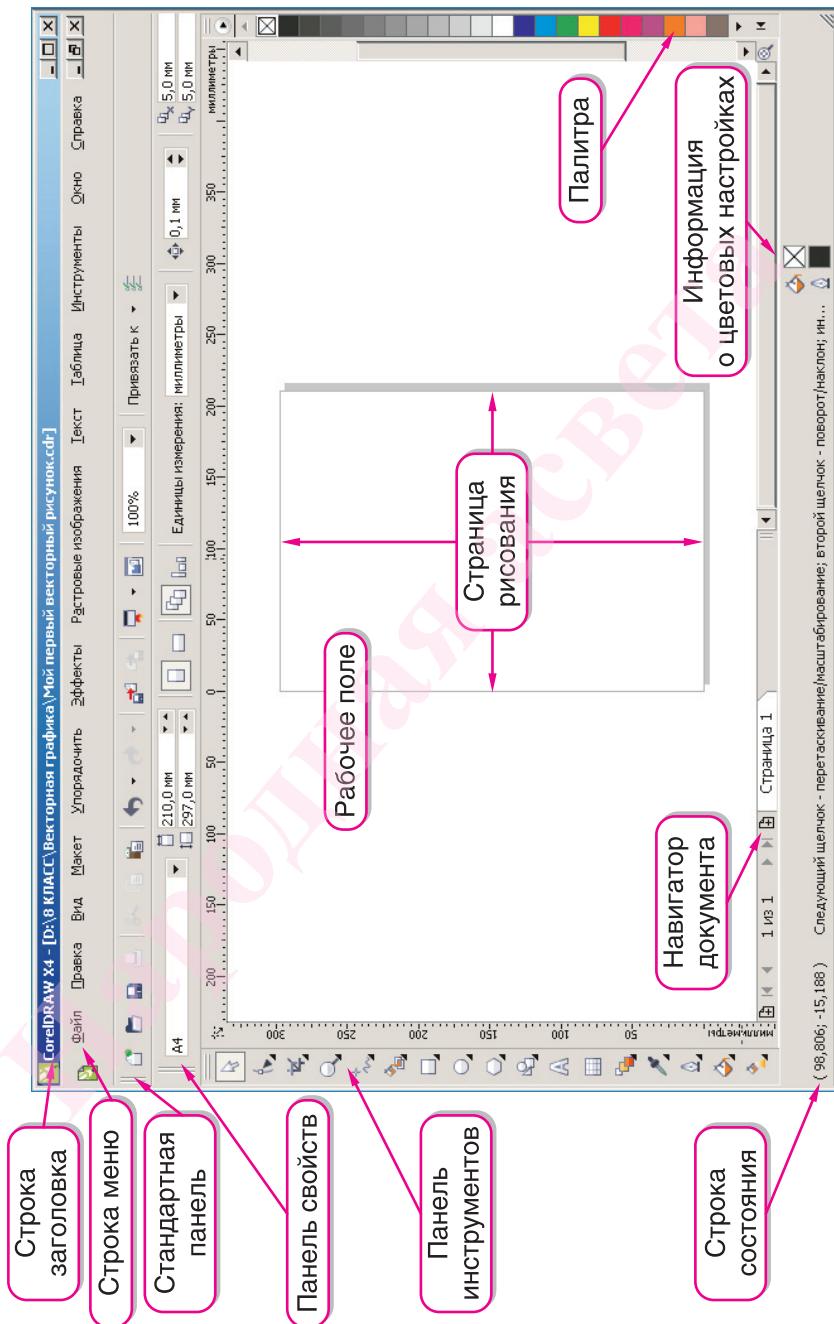


Рис. 3.5

меню указаны клавиатурные сокращения для быстрого вызова некоторых команд.

Панель в левой части окна содержит **инструменты** для рисования и редактирования изображений. Нужный инструмент выбирается щелчком мыши по кнопке с его изображением.

Схожие инструменты собраны в группы. На некоторых кнопках в правом нижнем углу есть темный треугольник. Если по нему щелкнуть мышью, то раскроется панель с инструментами группы. На рисунках 3.6 и 3.7 показаны группы **Основные фигуры** (Basic Shapes) и **Многоугольник** (Polygon Tool).

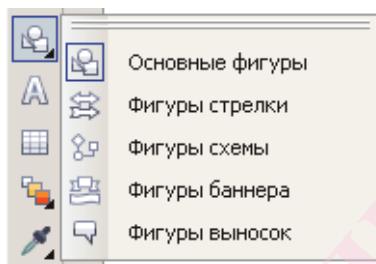


Рис. 3.6

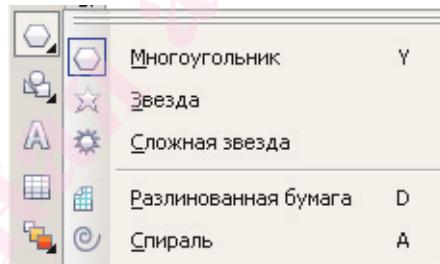


Рис. 3.7

Управлять параметрами выбранного инструмента позволяет **Панель свойств**. Содержимое этой панели изменяется в зависимости от выполняемой операции.

Строка состояния внизу окна содержит сведения о выбранном элементе и координаты курсора. Начало координат — левый нижний угол страницы рисования.

Полезная деталь интерфейса программы — всплывающие подсказки. Наведя курсор мыши на любую кнопку на панелях, можно увидеть подсказку с крат-



Интеллектуальная заливка



Форма (F10)

Рис. 3.8

ким пояснением назначения кнопки (рис. 3.8). В подсказках указываются и клавиатурные сокращения.

- ! Команды управления редактором и изображением можно вызывать одним из четырех способов: из главного меню, из контекстного меню (вызывается щелчком правой кнопкой мыши), кнопками на панелях, клавиатурными сокращениями.
Знание и использование клавиатурных сокращений значительно ускоряет работу пользователя.

- ? 1. Для чего предназначены графические редакторы?
2. Перечислите основные элементы окна программы CorelDRAW.
3. Что означает темный треугольник в правом нижнем углу некоторых кнопок на панели инструментов?

Упражнения

1. Откройте приложение CorelDRAW. Внесите изменения в интерфейс редактора.

Для этого:

а) командой **Справка** → **советы** откройте окно **Советы**, добавьте к нему окно **Диспетчер объектов** (команда **Окно** → **Окна настройки** → **Диспетчер объектов** (Window → Dockers → Object Manager)). Вернитесь к окну **Советы** щелчком по его вкладке.

б) Правой кнопкой мыши вызовите контекстное меню панели инструментов и измените ее внешний вид, выполнив команду: **Настройка** → **Панель инструментов** → **Средний размер кнопок** (Customize → → Toolbox Toolbar → Medium Button Size).

в) Выберите единицы измерения в документе — сантиметры.

г) Верните все настройки панелей редактора в первоначальное состояние. Для этого закройте окно **Диспетчер объектов** щелчком по кнопке ; примените команду контекстного меню **Настройка** → **Панель инструментов** → **Настройка по умолчанию** (Customize → Toolbox Toolbar → Reset to Default).

2. С помощью справки найдите и прочитайте сведения об интерфейсе CorelDRAW (команда **Справка** → **Вызов справки**).

§ 16. Создание векторного изображения

Векторное изображение создается из отдельных элементов. Удобная последовательность действий для создания простого векторного изображения следующая:

- 1) создать контур первого элемента;
- 2) настроить его размеры, форму, положение на странице;
- 3) задать «внешний вид контура» — стиль и цвет заливки и обводки;
- 4) повторить предыдущие действия для остальных элементов изображения;
- 5) настроить взаимное расположение всех нарисованных элементов.

Рассмотрим подробно каждый этап.

16.1. Объекты CorelDRAW

Любое векторное изображение состоит из **объектов**: точек, прямых, кривых, фигур, текста, импортированных растровых изображений. Линии, лежащие в основе объектов векторного изображения,

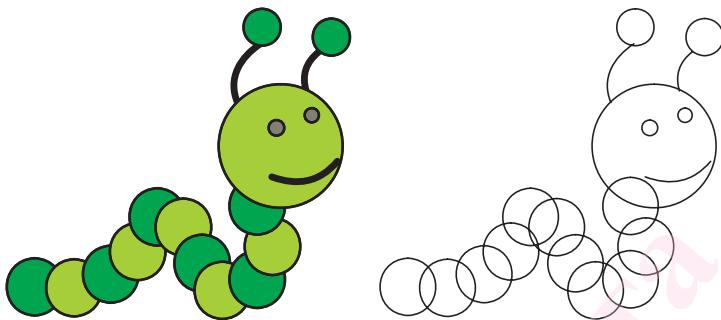


Рис. 3.9

называются **контурами**. Выбирая указателем мыши форму, положение и размер графического объекта, пользователь задает параметры объекта. Контур объекта рассчитывается программой по этим параметрам.

Цвет или текстура добавляются к контуру посредством применения к нему заливки и абриса. **Заливка** (Fill) — это цвет или текстура, заполняющая пространство внутри контура. **Абрис**¹ (OutLine) — это стиль отображения контура объекта. Если назначить объекту отсутствие и заливки, и абриса, то объект на экране станет невидимым. Найти «невидимку» в CorelDRAW можно в режиме просмотра **Каркас** (команда **Вид → Каркас** (View → Wireframe)). На рисунке 3.9 показаны изображения в режимах просмотра **Обычный** (Normal) и **Каркас** (WireFrame).

16.2. Инструменты рисования

Создание объекта начинается с выбора инструмента рисования. Принцип использования инструментов рисования такой же, как в редакторе Paint. Для со-

¹ В графических редакторах чаще применяется другой перевод термина «OutLine» — «обводка».

здания нового объекта на панели инструментов выбирают нужный инструмент рисования. Затем, нажав левую кнопку мыши и не отпуская ее, курсор инструмента перетаскивают по рабочему полю до получения нужного изображения. При нажатой клавише **Ctrl** рисуется правильная фигура (например, круг или квадрат). При нажатой клавише **Shift** рисование и масштабирование выполняются от центра.

При смене инструмента вид курсора изменяется и повторяет рисунок на кнопке выбранного инструмента.

Рассмотрим возможности некоторых инструментов рисования.

Инструментом **Свободная форма** (Freehand Tool)  рисуют прямые и кривые линии. Для создания прямой линии щелчком мыши фиксируют ее начало и конец. Кривая рисуется, как карандашом по бумаге, — перетаскиванием мыши с нажатой левой кнопкой. Нарисованную от руки линию редактор сглаживает, делает более плавной. При этом сохраняется ее натуральность, что придает очертаниям объекта естественность. Изменять толщину и стиль абриса линий можно с помощью панели свойств (рис. 3.10).

Кнопками  обозначены инструменты рисования фигур. **Фигура** (shape) — это объект,

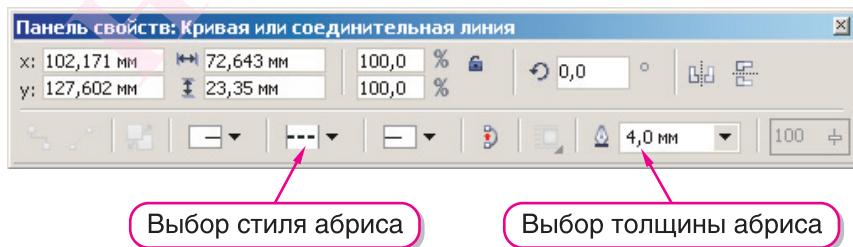


Рис. 3.10

который можно изменять только определенным образом, характерным для каждой фигуры. Например, можно изменять количество лучей звезды или скруглять углы прямоугольника.

По картинкам на кнопках , , , понятно, что они отвечают за создание прямоугольников, эллипсов, многоугольников, набора основных фигур. Перечисленные кнопки — раскрывающиеся (см. рис. 3.6 и 3.7).

Рассмотрим пример рисования фигуры заданного размера.

Пример 1. Нарисовать прямоугольник со сторонами 7 см и 12 см так, чтобы он отступал от левого и верхнего краев страницы на 2 см.

Выберем инструмент **Прямоугольник** (Rectangle Tool) . На рабочем поле нарисуем фигуру.

Уточним размеры нарисованного прямоугольника. Для этого на панели свойств введем горизонтальный и вертикальный размеры (рис. 3.11).

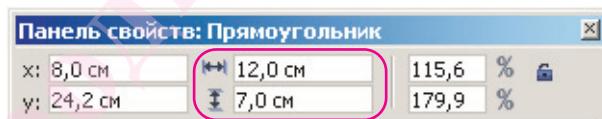


Рис. 3.11

Настроим положение фигуры на странице. Чтобы понять, за какую точку фигуры отвечают окошки x: и y:, впишем x: 0, y: 0. С началом отсчета совместится центр прямоугольника, помеченный крестиком. Это значит, что для получения отступов в 2 см надо ввести x: $12/2 + 2$, y: $29,7 - 2 - 7/2$. Число 29,7 — высота страницы. Программа выполнит вычисления и покажет результаты. Нарисованный прямоугольник переместится на запланированное место.

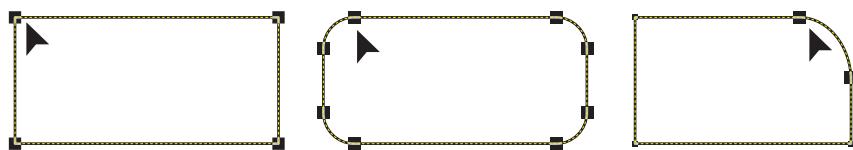


Рис. 3.12



Рис. 3.13

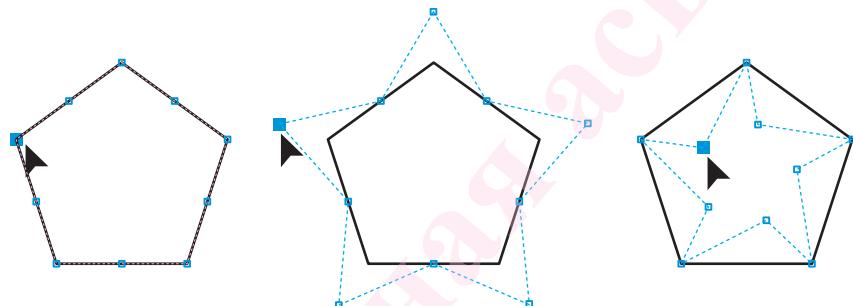


Рис. 3.14

Форму нарисованной фигуры можно изменять характерным для нее образом. Для этого надо выбрать инструмент **Форма** (Shape Tool) . После щелчка по нарисованной фигуре на контуре появятся маркеры управления в виде черных или синих квадратов. Их перемещение изменяет форму контура фигуры. Примеры изменений показаны на рисунках 3.12—3.14.

Пример 2. Изменить форму углов прямоугольника из примера 1 по образцу (рис. 3.15).

Выберем инструмент **Форма** и щелкнем на прямоугольнике — появятся четыре маркера по углам.

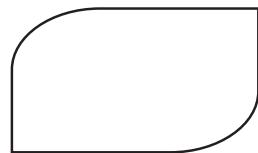


Рис. 3.15

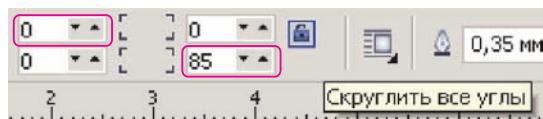


Рис. 3.16

Щелкнем на маркере правого нижнего угла — останется только один маркер. Перетащим этот маркер так, чтобы угол скруглился.

Чтобы получить одинаковую величину скругления, второй угол изменим с помощью панели свойств (рис. 3.16). Для этого введем в левое верхнее окошко то же числовое значение, что видно в правом нижнем. Проверим, «открыт» ли замок на панели свойств, так как «закрытый» замок на кнопке означает, что изменение величины скругления будет применяться ко всем углам одновременно. Нажмем клавишу Enter — фигура готова.



Рис. 3.17

Дополнительную возможность настройки формы контура имеют фигуры, создаваемые с помощью группы инструментов **Основные фигуры** (Basic Shapes). У них при создании и редактировании отображаются глифы — цветные маркеры в форме ромба. Перемещение глифа уточняет форму фигуры (рис. 3.17).

16.3. Заливка и абрис

Создав контур объекта, переходят к следующему шагу — добавлению заливки и абриса. Назначить их стиль и цвет можно несколькими способами. Са-

мый быстрый способ — выбор цвета из палитры. Для этого выделяют объект и щелкают левой кнопкой мыши по образцу в палитре. Так добавляется цвет заливки. Щелчком правой кнопкой мыши добавляют цвет абриса. Если немного задержать кнопку мыши нажатой на образце из палитры, то раскрывается палитра оттенков (рис. 3.18).

Пример 3. Нарисовать круги по образцу, данному на рисунке 3.19.

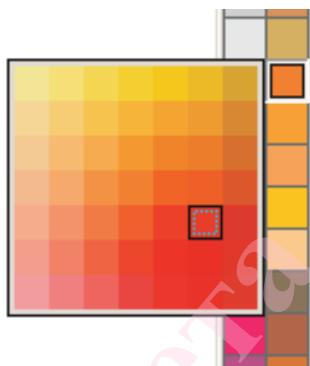


Рис. 3.18



Рис. 3.19

Выберем инструмент Эллипс (Ellips Tool) . При нажатой клавише Ctrl нарисуем первый круг. Правой кнопкой мыши щелкнем по зеленому образцу в цветовой палитре — абрис круга станет зеленым.

Рядом с первым кругом нарисуем второй. Левой кнопкой мыши щелкнем по оранжевому образцу в цветовой палитре — заливка готова. Правой кнопкой мыши щелкнем по клетке Нет в цветовой палитре — абрис станет невидимым.

Создав третий круг, выберем голубую заливку и темно-синий абрис. Затем двойным щелчком по области пера в строке состояния вызовем диалоговое окно Перо абриса (OutLine Pen)

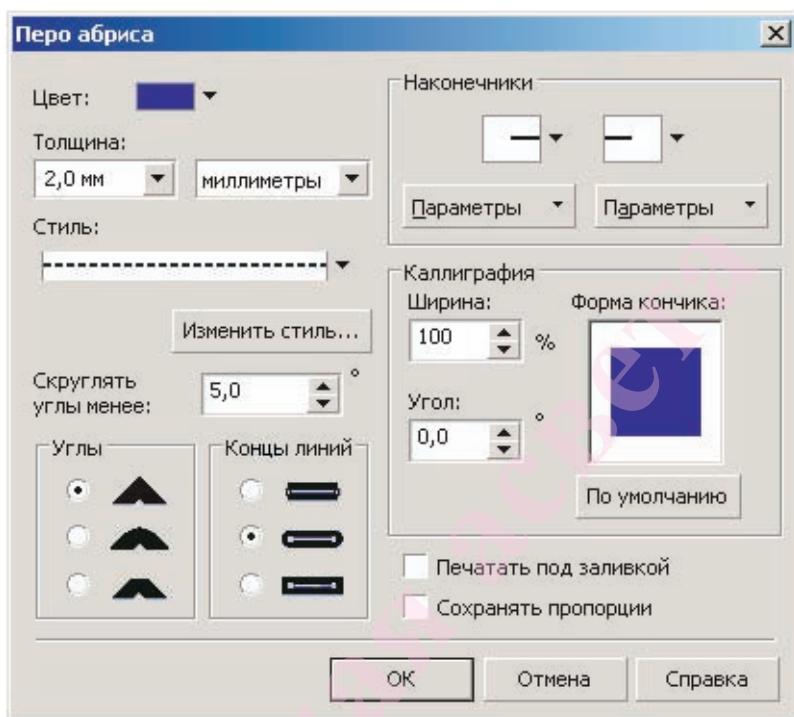


Рис. 3.20

(рис. 3.20). В окне выберем для абриса Толщину (OutLine Pen) — 2 мм и Стиль (Style) — штриховой. Рисунок готов.

16.4. Особенности Интеллектуальной заливки

Для добавления цвета есть особый инструмент — **Интеллектуальная заливка** (Smart Fill Tool) . Он может применяться к замкнутым областям, ограниченным контурами нескольких объектов. При этом заливается не указанная область, а новый объект с контуром в форме границы области. Таким образом, **Интеллектуальная заливка** — это инструмент быстрого создания нового объекта с контуром сложной формы.

Пример 4. Нарисовать фигуру по образцу, данному на рисунке 3.21.



В раскрывающемся меню кнопки **Многоугольник** (Polygon) выберем инструмент **Звезда** (Star) . При нажатой клавише Ctrl нарисуем звезду. На панели свойств увеличим количество лучей звезды до 13: . Инструментом **Прямоугольник** нарисуем поверх звезды прямоугольник, как на рисунке 3.22, а.

Рис. 3.21

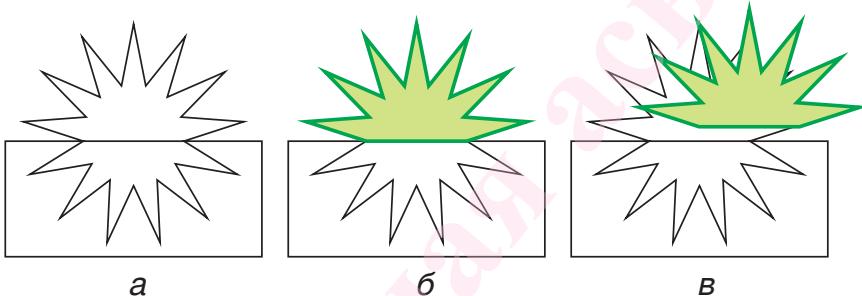


Рис. 3.22

Выберем инструмент **Интеллектуальная заливка** . На панели свойств в **Параметрах заливки** (Fill Options) зададим светло-зеленый цвет заливки, а в **Параметрах абриса** (Outline Options) — толщину абриса 0,5 мм и зеленый цвет абриса (рис. 3.23).

Для применения инструмента **Интеллектуальная заливка** щелкнем мышью внутри верхней половины

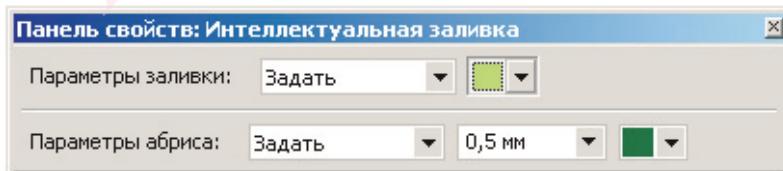


Рис. 3.23

звезды (рис. 3.22, б). Чтобы убедиться, что создан новый объект, курсором инструмента **Указатель** (Pick Tool)  сдвинем закрашенную часть звезды в сторону (рис. 3.22, в). Лишние детали — незакрашенные фигуры — удалим, щелкнув по ним стрелкой **Указатель** и нажимая клавишу Delete. Рисунок готов.

16.5. Стили заливки

Рассмотренные способы заливки относились к стилю **Однородная заливка** (Uniform Fill). Сделать раскраску объекта более выразительной помогает стиль **Фонтанная заливка** (Fountain Fill). В графических редакторах чаще применяется термин «градиентная заливка». В этой заливке используются два или более цвета — цвет плавно «перетекает» из одного в другой. Добавить и точно настроить фонтанную заливку можно инструментом **Фонтанная заливка** из списка раскрывающейся кнопки **Заливка** .

Быстрое редактирование цветового перехода удобно выполнять инструментом **Интерактивная заливка** (Interactive Fill Tool) . Маркеры (рис. 3.24) отве-

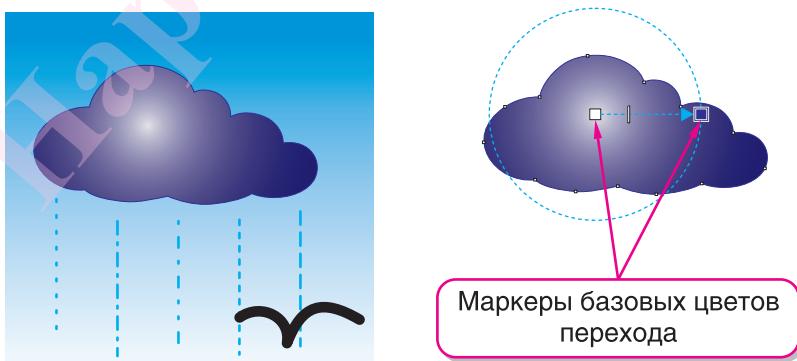


Рис. 3.24

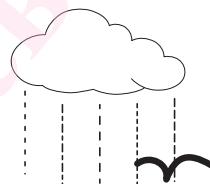
чают за базовые цвета перехода. Изменить цвет выделенного маркера можно щелчком левой кнопкой мыши по образцу из палитры. Положение маркеров изменяется перетаскиванием.

- ? 1. Из чего состоит векторное изображение? Какие объекты, на Ваш взгляд, использовались для создания иллюстраций к тексту этого параграфа?
2. Какие инструменты рисования фигур Вам известны?
3. Что называется заливкой и абрисом?

Упражнения

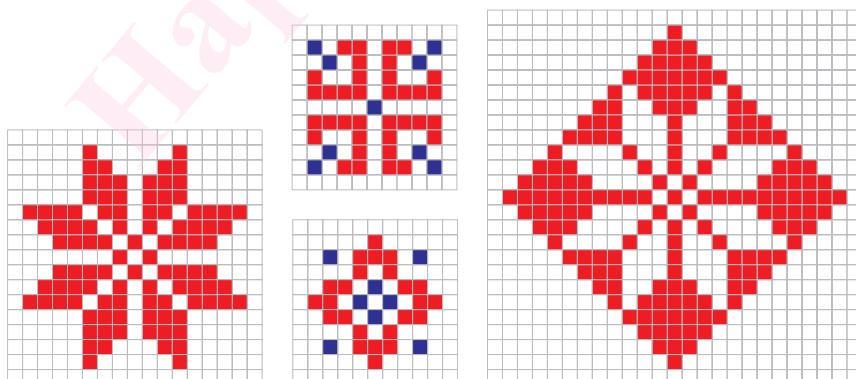
1. Используя инструмент **Свободная форма**, нарисуйте картинку по образцу.

Добавьте однородную заливку к элементам рисунка.



2. Используя инструменты рисования фигур, нарисуйте прямоугольник со сторонами 5 см и 7 см. Закрасьте его зеленым цветом, абрис сделайте черным пунктирным толщиной 4 пункта.

3. С помощью инструмента **Интеллектуальная заливка** создайте на выбор фрагмент белорусского орнамента.



Для этого вначале нарисуйте сетку с квадратными ячейками (инструмент **Разлинованная бумага** (Graph Paper)  при нажатой клавише Ctrl). Выберите инструмент **Интеллектуальная заливка**. На панели свойств задайте красный цвет заливки и отсутствие абриса. Залейте клетки в соответствии с орнаментом. Удалите вспомогательную сетку, выделив ее и нажав клавишу Delete.

§ 17. Редактирование векторного изображения

Редактированием изображения называется внесение в него изменений. Напомним, что векторное изображение состоит из деталей — объектов. Значит, редактировать можно, добавляя, удаляя или изменяя отдельные объекты.

17.1. Выделение

Изменять можно только выделенный объект. Выделение производится инструментом **Указатель** (Pick Tool) . Для этого можно курсором Указателя щелкнуть по объекту или обвести его прямоугольной рамкой. Только что созданный объект всегда выделен. Вокруг выделенного объекта отображается ограничивающий блок из маркеров изменения размера (рис. 3.25).

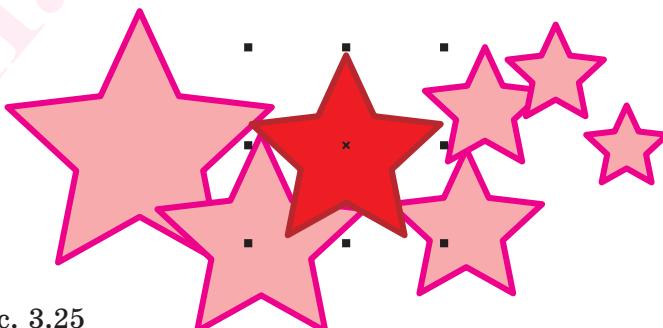


Рис. 3.25

Для выделения нескольких объектов можно, удерживая нажатой клавишу Shift, щелкать на каждый объект, который необходимо выделить.

Снять выделение можно щелчком мыши по рабочему полю.

17.2. Трансформация

Над выделенным объектом можно производить следующие преобразования: перемещение, масштабирование, растягивание, зеркальное отображение, наклон, поворот.

Для пропорционального изменения размера объекта перетаскивают один из угловых маркеров (рис. 3.26, а). При нажатой клавише Shift изменение выполняется от центра. Для растягивания объекта по горизонтали используют правый или левый маркер, для растягивания по вертикали — верхний или нижний (рис. 3.26, б).



Рис. 3.26

Чтобы активировать маркеры поворота и наклона, можно щелкнуть на центральном маркере объекта (крестик в центре). Для поворота объекта перетаскивают один из угловых маркеров (рис. 3.27, а), для наклона — боковые маркеры (рис. 3.27, б).

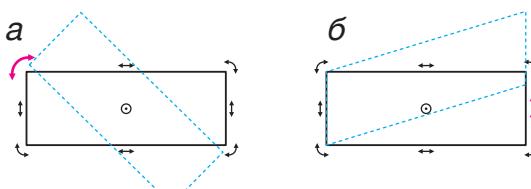


Рис. 3.27

Для получения зеркального отображения можно пользоваться кнопками панели свойств **Отразить по горизонтали** и **Отразить по вертикали** .

Пример 1. Нарисовать модель моста (рис. 3.28).

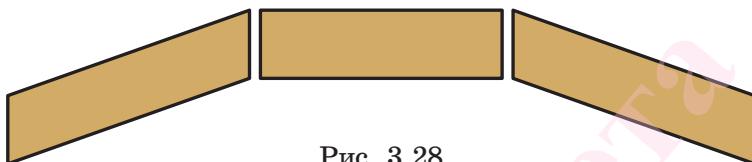


Рис. 3.28

Нарисуем три прямоугольника в ряд.

Выделим левый прямоугольник инструментом **Указатель**. Следующим щелчком мыши активируем маркеры поворота и наклона. Наклоним левый край прямоугольника вниз.

Аналогично наклоним вниз правый край последнего прямоугольника.

17.3. Редактирование контура

Попробуйте из прямоугольника и звезды получить фигуры, как на рисунке 3.29, и окажется, что это невозможно. Контур фигуры «помнит», каким инструментом он был нарисован, поэтому редактируется только в пределах возможностей этого инструмента. Чтобы «освободить» контур и получить возможность произвольно его редактировать, надо превратить

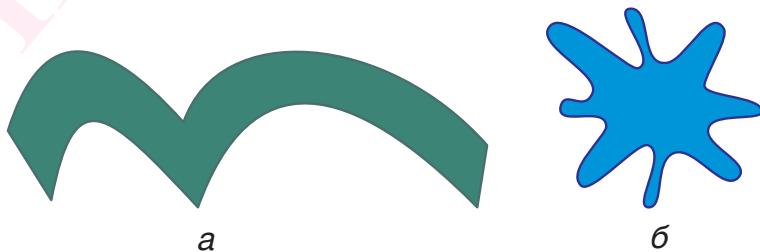


Рис. 3.29

его в кривую. Для этого применяется команда **Упорядочить** → **Преобразовать в кривую** (Arrange → → Convert To Curves) или кнопка  на панели свойств фигур. После этого контур можно произвольно изменять инструментом **Форма** .

Контуры, нарисованные инструментом **Свободная форма** , сразу доступны для редактирования инструментом **Форма**. При выделении контура этим инструментом синими квадратиками выделяются узлы, отвечающие за форму контура. Эти узлы можно удалять, добавлять, перемещать. Участки контура между узлами можно выпрямлять или изгибать. Для этих и других операций есть соответствующие кнопки на панели свойств инструмента **Форма**. Всплывающие подсказки помогают при необходимости разбираться в назначении кнопок.

Пример 2. Изогнуть прямоугольник, как показано на рисунке 3.29, а.

Нарисуем прямоугольник. Преобразуем его в свободно редактируемый контур, нажав кнопку **Преобразовать в кривую** .

Выберем инструмент **Форма** . На углах контура появятся синие квадратики маркеров . При наведении курсора инструмента **Форма** на узел изменится вид маркера узла .

В местах изгиба на верхней и нижней сторонах прямоугольника добавим узлы. Для этого щелчком мыши отметим место, где будет добавляться узел . Нажмем кнопку **Добавить узлы** (Add Node) . На месте пометки-звездочки появится синий квадратик нового узла .

Теперь прямые участки контура можно изгибать. Для этого выделим изгибающий участок контура.

Нажмем кнопку **Преобразовать линию в кривую** (Convert Line To Curve) . Прямой участок контура изогнем курсором инструмента **Форма**. Появившиеся при этом голубые стрелки помогут точнее настроить форму изгиба. Результат получен — контур прямоугольника полностью изменен.

! Редактировать неудачно нарисованный контур сложнее, чем рисовать его заново. Запомните полезное сочетание клавиш Ctrl + Z. Оно отменяет последнее действие. В списке кнопки **Отменить**  на стандартной панели можно выбрать любое количество шагов для отмены.

17.4. Взаимное расположение

Важный этап в работе над векторным изображением — это настройка взаимного расположения объектов.

Объекты векторной графики располагаются на разных уровнях — одни находятся выше других в воображаемой стопке. Каждому объекту выделен собственный уровень. Вновь создаваемый объект располагается поверх ранее нарисованных и может их частично или полностью закрывать. При этом нижние объекты не исчезают и не изменяются. Порядок объектов можно увидеть в окне **Диспетчер объектов** (рис. 3.30), выполнив команду **Инструменты** → **Диспетчер объектов** (Tools → Object Manager). Изменяется порядок объектов командами из списка **Порядок** (Order) пункта меню **Упорядочить** (Arrange). Быстро изменить порядок можно с помощью кнопок **На передний план слоя** (To Front Of Layer)  и **На задний план слоя** (To Back Of Layer) .



а

Рис. 3.30

б

Пример 3. Преобразовать рисунок 3.30, а в рисунок 3.30, б.

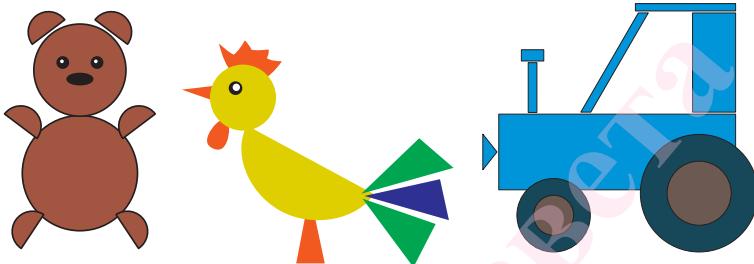
Выделим звезду и нажмем кнопку **На задний план слоя** .

Выделим прямоугольник и нажмем кнопку **На передний план слоя** .

- ?
1. Как выделить один объект, несколько объектов?
 2. Как изменить размер объекта, как повернуть и наклонить объект?

Упражнения

1. Используя инструменты рисования фигур, поворот и наклон, нарисуйте на выбор один из рисунков.



2. Отредактируйте контуры восьмиугольника так, чтобы получилась фигура рисунка 3.29, б.

§ 18. Операции над объектами векторного изображения

Для ускорения и облегчения работы пользователя в редакторе CorelDRAW предусмотрена возможность выполнения ряда операций над объектами и группами объектов. Это операции копирования и дублирования, распределения и выравнивания, группировки и формирования. Рассмотрим способы использования этих операций.

18.1. Дублирование

Для получения копии существующего изображения можно использовать копирование или дублирование. Для копирования нужно к выделенному объекту применить команду Правка → Копировать (Edit → Copy), а затем команду Правка → Вставить (Edit → Paste). Копия размещается точно поверх исходного изображения.

Дублирование удобно тем, что выполняется одной операцией **Правка** → **Дублировать** (Edit → Duplicate) или сочетанием клавиш Ctrl + D. Копия выделенного объекта вставляется непосредственно на рабочее поле, минуя буфер обмена. При дублировании объекта можно задать расстояние вдоль осей *x* и *y* между исходным объектом и его дубликатом. Это расстояние называется смещением.

Пример 1. Нарисовать фрагмент пчелиных сот (рис. 3.31).

Нарисуем правильный шестиугольник. Установим на панели свойств шестиугольника горизонтальный размер 1 см. Это потребуется для числовой настройки смещения дубликата.

Не снимая выделение, нажмем комбинацию клавиш Ctrl + D. В открывшемся окне **Смещение дубликата** (Duplicate Offset) введем горизонтальное смещение: 1 см и вертикальное смещение: 0 см. Нажмем комбинацию клавиш Ctrl + D еще пять раз — получим ряд шестиугольников, точно совмещенных боковыми сторонами.

Для дублирования целого ряда зададим новое смещение. Для этого снимем выделение, щелкнув мышью на пустом поле. На панели свойств введем значения смещения по оси *x*: 0,5 см и по оси *y*: -0,86 см (рис. 3.32).



Рис. 3.31

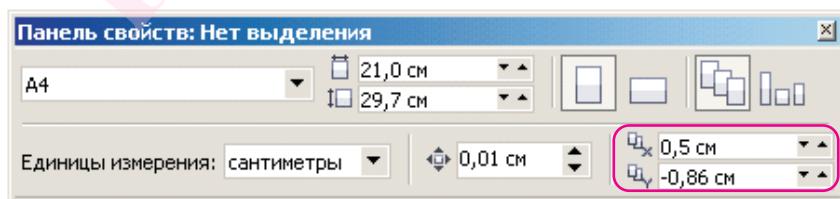


Рис. 3.32

Выделим весь ряд и нажатием комбинации клавиш Ctrl + D продублируем его несколько раз. Рисунок готов.

18.2. Выравнивание, распределение

«Помощником» пользователя можно назвать набор операций выравнивания и распределения объектов на рабочем поле. Эти операции применяются к группе объектов.

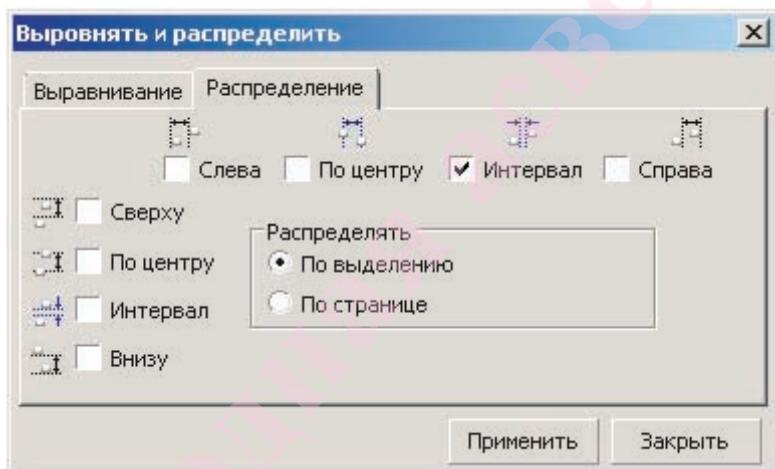


Рис. 3.33

Для применения операций выделяют необходимые объекты. Командой Упорядочить → Выровнять и распределить (Arrange → Align and Distribute) вызывают список возможностей. В открывшемся списке выбирают нужную команду. Выбрав строку Выровнять и распределить, открывают соответствующее диалоговое окно (рис. 3.33).

18.3. Формирование

Посмотрите на рисунок 3.34. Почти всю стрекозу можно нарисовать с использованием инструментов рисования фигур. Но как нарисовать хвост? Помочь решить эту задачу может набор операций, объединенный названием **Формирование** (Shaping).

При выделении нескольких объектов на панели свойств появляются кнопки операций. Нужная нам кнопка — **Объединить** (Weld) (рис. 3.35).

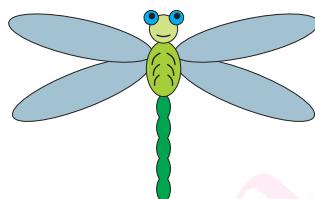


Рис. 3.34

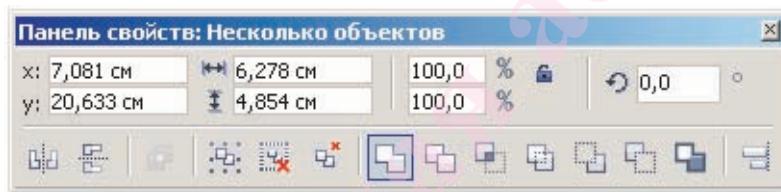


Рис. 3.35

Пример 2. Нарисовать хвост стрекозы, изображенной на рисунке 3.34.

Нарисуем эллипс. Нажмем комбинацию клавиш **Ctrl + D** четыре раза — получим пять одинаковых эллипсов. Переместим их так, чтобы они частично перекрывали друг друга.

Выделим все пять эллипсов. Командой **Упорядочить** → **Выровнять и распределить** вызовем окно **Выровнять и распределить** (см. рис. 3.33). На вкладке **Выравнивание** выберем **По верху** и нажмем **Применить**. На вкладке **Распределение** выберем горизонтальный **Интервал** и нажмем **Применить**.

Не снимая выделение, нажмем кнопку **Объединить**. Фигура готова.

18.4. Группировка

Различные объекты можно объединять в группы. Это делают для того, чтобы применять операции сразу ко всем участникам группы. Кроме того, группировка предотвращает случайные изменения положения объекта относительно других объектов. Для выполнения группировки надо выделить все объекты будущей группы и применить команду **Упорядочить** → **Группировать** (Arrange → Group). При перемещении и трансформации группа ведет себя как один объект. При этом можно воздействовать отдельно на каждый элемент в группе. Для этого его выделяют Указателем с нажатой клавишей Ctrl. Группировка при этом не отменяется.

Для отмены группировки надо выделить группу и применить команду **Упорядочить** → **Разгруппировать** (Arrange → Ungroup).

- ?** 1. Можно ли получить несколько копий элемента изображения? Как это можно сделать?
- 2. Можно ли сгруппировать несколько объектов? Для чего это нужно?

Упражнения

1. Составьте орнамент по образцу, используя фрагмент, созданный Вами в упражнении 3 к § 16.



Для удобства обращения с фрагментом сгруппируйте его. Для точного размещения фрагментов в орнаменте используйте операцию выравнивания по

верхнему краю и числовые настройки смещения дубликата.

2. Нарисуйте стрекозу, как на рисунке 3.34.

§ 19. Работа с текстом

В рисунки можно добавлять текст двух видов — фигурный и простой. Фигурный текст используется для добавления коротких строк текста. К нему можно применить множество эффектов, например тени. Простой текст используется для больших блоков текста, которые нуждаются в точном форматировании. Текст в CorelDRAW можно создавать, форматировать и редактировать. Для управления работой с текстом в меню предусмотрен отдельный пункт **Текст**.

19.1. Создание

Текст можно добавлять непосредственно на рабочее поле. Для этого применяется инструмент **Текст** (Text)  . **Фигурный текст** набирают сразу после щелчка инструментом **Текст** по полю.

Для добавления **простого текста** сначала создается рамка текста (рис. 3.36). Для этого выбирают инструмент **Текст** и рисуют прямоугольник. После этого щелкают мышью внутри рамки и вводят текст.



Рис. 3.36

19.2. Форматирование

В графическом редакторе доступны те же параметры форматирования шрифта и абзаца, что и в текстовом процессоре. Для изменения формата текстового фрагмента можно использовать панель свойств. Для этого выбирают инструмент **Текст**, выделяют нужный текст и изменяют его параметры на панели (рис. 3.37).

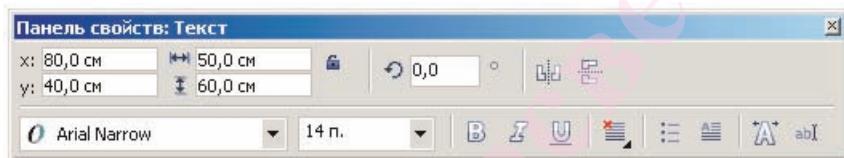


Рис. 3.37

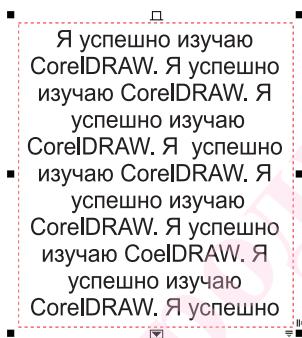


Рис. 3.38

Для форматирования простого текста можно использовать возможности рамки текста. Так, если не весь текст вместился в рамку, то он «прятается» за ее границей. Рамка в этом случае становится красной (рис. 3.38). Вместить весь текст в рамку можно разными способами:

- 1) изменить размер рамки, потянув за маркеры текстовой рамки;
- 2) изменить размер шрифта;
- 3) применить команду **Текст → Простой текст → Текст в рамку** (**Text → Paragraph Text Frame → Fit Text to Frame**). При этом пропорционально изменятся все параметры форматирования: размер шрифта, интервал между буквами, интервал между строками — и текст полностью займет рамку.

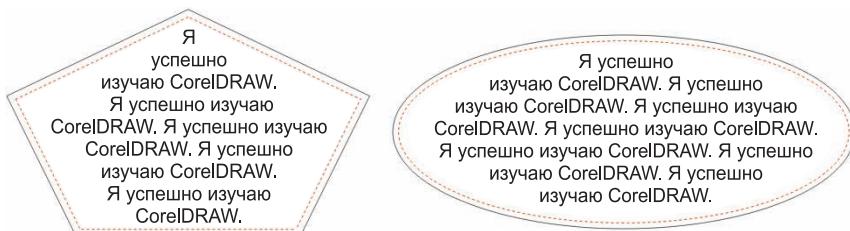


Рис. 3.39

В качестве рамки можно использовать любой замкнутый контур (рис. 3.39). Для этого следует нарисовать контур, выбрать инструмент **Текст**, подвесить курсор к контуру (курсор примет вид курсора ввода текста). Щелкнуть мышью внутри контура и напечатать текст.

Фигурный текст можно располагать вдоль открытого или замкнутого контура. Простой текст размещается только вдоль незамкнутого контура. При этом он сохраняет возможность форматирования с помощью рамки.

Чтобы расположить текст вдоль контура, можно использовать следующую последовательность действий.

1. С помощью инструмента **Текст** ввести текст (простой или фигурный).

2. Нарисовать контур, вдоль которого будет размещаться текст.

3. Инструментом **Указатель** выделить введенный текст и выбрать команду **Текст → Текст вдоль пути** (**Text → Fit Text to Path**). Курсор примет вид толстой стрелки , которой надо указать на контур. Не отпуская кнопку мыши, можно настроить положение перемещаемого текста относительно контура — расстояние до контура, смещение относительно начала контура (рис. 3.40, а).

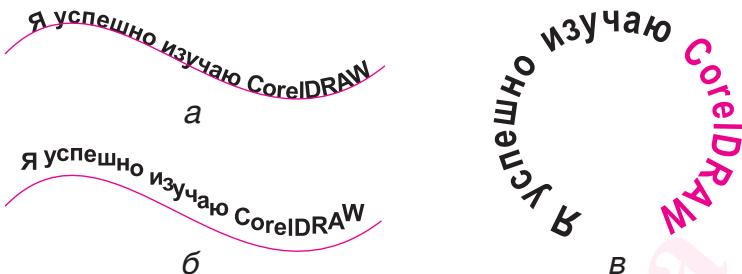


Рис. 3.40

4. Не снимая выделение с текста, можно осуществить его точную настройку с помощью панели свойств. Например, уточнить ориентацию букв (рис. 3.40, *б*).

Рамки текста и контуры, вдоль которых расположен текст, обычно являются вспомогательными. Их можно удалять или делать невидимыми. При этом текст сохраняет заданную форму (рис 3.40, *в*).

! Главное при работе с текстом — не допускать, чтобы применяемые эффекты мешали удобству чтения.

19.3. Редактирование

Текст можно редактировать непосредственно в рамке. Удобнее работать в специальном окне **Редактирование текста** (Edit text). Окно открывается командой Текст → Редактирование текста (Text → Edit Text) или щелчком по кнопке . Это окно напоминает обычновенный текстовый редактор. В окне выполняются операции копирования, перемещения, вставки, удаления фрагментов текста. Параметры форматирования текста также можно задавать в этом окне, но отобразятся они только на рабочем поле. Отметим, что в окне Редактирование текста работает автоматическая проверка орфографии.

- ?
- 1. Можно ли добавить текст в изображение?
- 2. Как создать текстовый блок в форме квадрата со стороной 6 см?

Упражнения

1. Создайте на выбор одно из изображений, показанных на рисунках 3.39 и 3.40.
2. Нарисуйте в графическом редакторе блок-схему (например, рисунок 1.14 из § 3).

§ 20. Сохранение и загрузка изображений

20.1. Сохранение изображения

Результат работы в графическом редакторе — изображение на экране и в оперативной памяти компьютера. Полученное изображение следует сохранить в виде файла. Поскольку изображение в редакторе подготавливается чаще всего для дальнейшего использования в других приложениях, например для вставки иллюстрации в текстовый документ, необходимо сохранять рисунок в формате, который поддерживается выбранным приложением.

Документ записывается на диск командой **Файл** → → **Сохранить как...** (File → Save as...). Раскрывающееся окно **Сохранение документа** (Save Drawing) похоже на аналогичные окна других приложений Windows. В окне выбирается папка для хранения файла. В поле **Имя файла** (File name) стандартное имя заменяется или дополняется. Имена файлам лучше давать осмысленные, чтобы впоследствии ориентироваться в их содержании. Далее выбирается тип файла. Собственный тип файлов приложения CorelDRAW — **.cdr**.

! Чтобы не потерять сделанную работу при различных сбоях компьютера или из-за собственных ошибочных действий, регулярно сохраняйте результаты на диск. Для этого сохраните файл сразу после создания нового документа. Затем в процессе работы периодически (например, каждые 10 мин) сохраняйте внесенные изменения командой **Файл → Сохранить** (File → Save) или комбинацией клавиш Ctrl + S.

20.2. Типы файлов

Все типы файлов, доступные графическому редактору CorelDRAW, можно увидеть в выпадающем списке при выполнении команд **Открыть** (Open) или **Сохранить как...**.

Рассмотрим назначение некоторых из них.

- **.pdf** (Portable Document Format — переносимый формат документов) удобен для распространения документов в электронном виде, создания иллюстрированной документации, передачи в типографию.
- **.svg** (Scalable Vector Graphics — масштабируемая векторная графика) используется для размещения векторных рисунков на интернет-страницах.
- **.ai** (Adobe Illustrator) выбирается, если предполагается доработка изображения в редакторе Adobe Illustrator.
- **.emf** (Enhanced Metafile Windows — расширенный метафайл Windows) используется, если рисунок создается для вставки в текстовый документ Word. После вставки рисунок можно редактировать средствами текстового редактора.

20.3. Загрузка изображения

Загрузить изображение можно одним из трех способов:

- командой меню **Файл → Открыть** (File → Open);
- кнопкой  **Открыть** на стандартной панели управления;
- выбором файла из списка **Быстрого запуска** окна приветствия (кнопка  на стандартной панели).

Каждый из этих способов открывает диалоговое окно **Открытие документа** (Open Drawing) (рис. 3.41). Вид и правила работы в окне знакомы Вам по другим приложениям Windows. Удобным дополнением

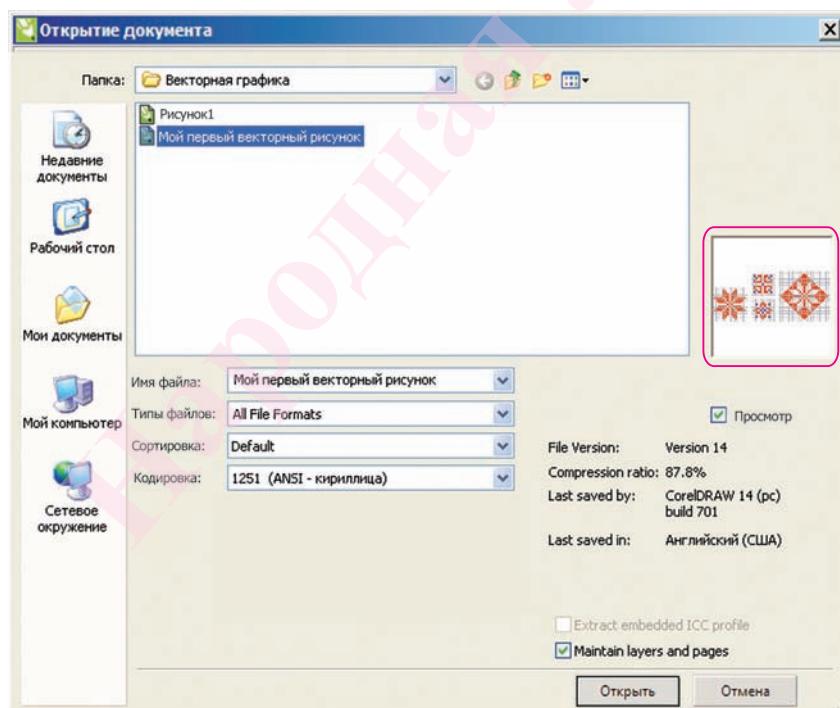


Рис. 3.41

является возможность просмотра первой страницы документа. После выбора нужной папки и искомого файла рисунок загружается в оперативную память и появляется на экране.

20.4. Свойства документа

Если рисунок готовится для печати, то сначала необходимо настроить параметры страницы, выводимой на печать, и проверить, попадает ли нужная часть изображения на распечатываемую страницу. Для этого выполняются следующие действия:

- делаются видимыми границы страницы (если они не отображались) командой Вид → Показать → → Границы страницы и Печатаемая область (View → Show → Page Border и Printable Area);
- настраиваются параметры страницы на панели свойств (рис. 3.42).

В случае необходимости рисунок масштабируется так, чтобы он вместился в границы печатаемой области.

Если предполагается использовать рисунок в редакторе растровой графики, то изображение можно экспорттировать. Экспорт выполняется командой Файл → Экспорт (File → Export).

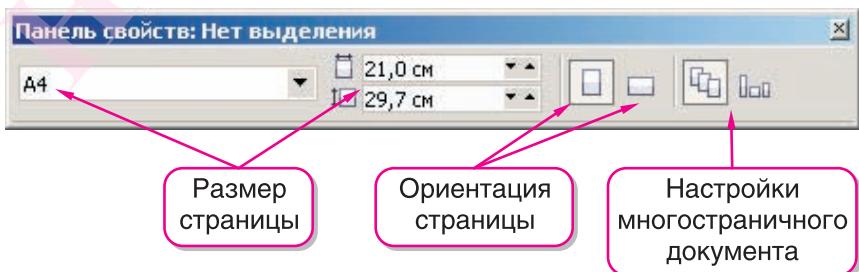


Рис. 3.42

Окно Экспорт (Export) похоже на окно Сохранение документа. Для экспорта удобно пользоваться типами файлов .jpg и .gif. В некоторых случаях может потребоваться тип .bmp.

Если на рабочем поле выделить объект, то можно экспорттировать только его, а не весь рисунок. Для этого в окне Экспорт надо включить флажок Выделенные объекты.

- ? 1. Какое расширение может иметь файл с векторным рисунком?
- 2. Какие команды в графическом редакторе используются для сохранения изображения, для загрузки изображения?

Упражнения

1. Создайте новый документ CorelDRAW с размерами страницы рисования, соответствующими размерам обложки тетради по информатике, бейджа, почтовой марки.
2. Откройте изображение, указанное учителем, и сохраните его с расширением .emf. Вставьте созданное изображение в документ Word.

Глава 4

Вредоносное программное обеспечение и защита информации

§ 21. Виды вредоносного программного обеспечения

Хранящаяся на компьютерах информация должна быть защищена от вирусов, шпионских программ, киберпреступников, нежелательных и вредоносных сообщений электронной почты. Из перечня видно, что для нанесения вреда могут использоваться разные идеи и технологии, но их объединяет одно — злой умысел или злые намерения.

Нанести вред информации, хранящейся на компьютере, могут целенаправленные действия человека или вредоносные программы. Основной угрозой при этом является получение доступа к компьютеру для повреждения информации, кражи, вымогательства, шантажа, контроля над зараженным компьютером или использования компьютера в качестве промежуточного звена для атаки.

К целенаправленным действиям человека относят **сетевые (компьютерные) атаки**. Такие атаки возможны из-за ошибок в программах или ошибок в построении защиты компьютера. Целями могут быть:

- хулиганство;
- шутка;
- вандализм;
- стремление доказать свою квалификацию;
- мошенничество. Например, обман пользователей с целью выманить их ценные личные данные. Для этого мошенники рассылают письма от имени известных сервисов, например социальных се-

тей (**Одноклассники**, **В Контакте**). В письме содержится прямая ссылка на сайт злоумышленника и просьба подтвердить сведения о себе. Оказавшись на таком сайте, пользователь может сообщить мошенникам ценную информацию. Такие действия называют фишинг-атакой¹;

- виртуальный терроризм. Например, специальной программой заражаются многие компьютеры. Затем по команде злоумышленника с этих компьютеров реализуются одновременные обращения к атакуемому серверу, что приводит к его перегрузке и выходу из строя. Так совершаются DDoS-атаки².

Вредоносной программой называется любое программное обеспечение, предназначенное для получения несанкционированного доступа к информации, хранимой на компьютере, с целью причинения вреда владельцу информации или владельцу компьютера.

К вредоносным программам относятся: *вирусы, черви, троянские, шпионские и рекламные программы*.

Компьютерным вирусом называется программа, которая заражает другие программы, копирует себя и может выполнять вредоносные действия на компьютере. Заражается компьютер при запуске программы-вируса либо программы, которая заражена вирусом.

Вирусы могут быть *файловыми, загрузочными, сетевыми и макровирусами*.

¹ Фишинг (*phishing*) — рыбная ловля, выуживание.

² DDoS — сокращение английских слов *Distributed Denial of Service* — распределенный отказ в обслуживании.

Файловые вирусы поражают исполняемые файлы, например с расширениями .exe и .com. После запуска зараженной программы вирус проникает в оперативную память компьютера и может «приписывать» себя к другим файлам. При этом могут быть заражены даже звуковые или графические файлы.

Загрузочные вирусы записывают себя в загрузочный сектор диска. Они запускаются при загрузке компьютера. В последнее время получили большое распространение *вирусы автозапуска*, такие как вирус Autorun. Передается вирус при подключении к компьютеру зараженного съемного носителя (флешдиска, CD-диска).

Сетевые вирусы — это вирусы, распространяющиеся по компьютерной сети.

Макровирусы поражают файлы приложения Microsoft Office, например документы Word, электронные таблицы Excel. После открытия зараженного документа макровирусы постоянно присутствуют в памяти компьютера и могут заражать другие документы. Угроза заражения исчезает только после закрытия приложения.

Существуют также специфические вредоносные программы, которые распространяют свои копии по локальным и глобальным сетям. Это могут быть **черви** и **Троянские программы (Трояны)**. Они различаются по действиям, которые производят на компьютере. Активизация червя может вызвать уничтожение программ и данных. А Трояны осуществляют тайные действия по сбору, изменению и передаче информации злоумышленнику.

Рассмотрим подробнее эти вредоносные программы.

Черви, которые распространяются через электронную почту и сеть Интернет, называются *Интернет-червями*. К ним относятся *почтовые черви* и *Веб-черви*.

Почтовые черви распространяются через сообщения электронной почты, содержащие прикрепленный файл, который может оказаться вредоносной программой. Опасными являются письма, содержащие ссылки на файл, в котором спрятан червь. Если перейти по такой ссылке, то сразу начнется загрузка и активизация червя. Больше всего вредоносного программного обеспечения распространяется под видом виртуальных открыток или приглашений получить выигрыш.

Веб-черви используют для своего распространения Веб-сервисы. Заразиться ими можно при посещении зараженного сайта. Часто веб-черви прячутся в активные элементы веб-страниц или скрипты. (Пассивные элементы веб-страницы — это рисунки, видеофрагменты, звуковые клипы. Активные элементы веб-страницы содержат программы на языках JavaScript или VBScript.)

Троянские программы проникают на компьютер, как сетевые черви. Бывают *Трояны удаленного управления*; *Трояны-шионы*; *Трояны, ворующие информацию*. Так, Троянские программы бэкдор¹ предназначены для скрытого удаленного управления компьютером. Они, в отличие от сетевых червей, не распространяются по сети самопроизвольно. Программа подчиняется специальной команде «хозяин-

¹ Бэкдор (от англ. Back door) — черный ход.

на», управляющего данной копией троянской программы.

Трояны-шпионы ведут электронный шпионаж за пользователем зараженного компьютера. Эти программы могут, например, записывать нажатия клавиш на клавиатуре или копировать изображение на экране. Полученная информация часто используется для кражи информации пользователей о различных платежах в банковских системах.

Трояны, ворующие информацию, ищут и передают информацию о пользователе или компьютере, например регистрационные данные к лицензионному программному обеспечению.

Шпионские и рекламные программы относят к потенциально опасным программам. Они не всегда причиняют вред человеку. **Шпионские программы** — программное обеспечение, которое тайно устанавливается и используется для доступа к информации, хранимой на компьютере. **Рекламные программы** отображают рекламные объявления, загружаемые из Интернета. Часто эти объявления появляются в отдельных окнах на рабочем столе или поверх рабочего окна. Даже при закрытии этих рекламных окон программа продолжает работать и следить за действиями пользователя в Интернете.

Наряду с вредоносными сообщениями электронной почты существуют нежелательные сообщения. К ним относят **спам** — анонимную массовую незапрошенную рассылку.

- ?
- 1. Что называют компьютерным вирусом?
- 2. Какие Вы знаете виды вредоносных программ?

§ 22. Способы защиты от вредоносного программного обеспечения

22.1. Антивирусные программы

Антивирусные программы — это программы для обеспечения комплексной защиты данных на компьютере от вредоносных программ и способов их проникновения на компьютер. К наиболее популярным антивирусным программам относят: Avast!, NOD32, Антивирус Касперского, Dr. Web [Доктор Веб], Norton AntiVirus, ВирусБлокАда (VBA32).

Антивирусные программы выполняют следующие функции.

- *Постоянная защита* (антивирусный монитор, или резидентный сканер). Слово «резидентный» означает «постоянно присутствующий». Программа постоянно контролирует процессы в памяти компьютера и проявляет активность только при обнаружении действий вредоносных программ.
- *Защита по требованию пользователя* (антивирусный сканер). Программа проверяет оперативную память, а также ищет вредоносные программы на жестких и сетевых дисках. Она запускается по заранее настроенному расписанию или по требованию пользователя в любое время.

Многие антивирусные программы сочетают в себе функции постоянной защиты и защиты по требованию пользователя. Так, для комплексной защиты компьютера от вирусов и всех других типов вредоносных программ, а также от хакерских атак и спа-

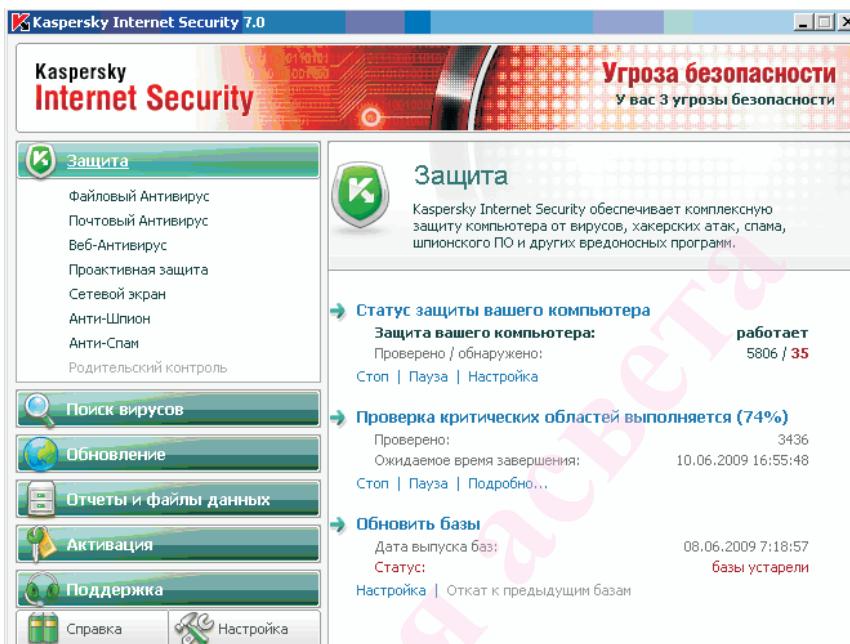


Рис. 4.1

ма предназначена, например, программа Kaspersky Internet Security (рис. 4.1).

В некоторых программах окна резидентного сканера и обычного сканера могут быть разделены. Например, в программе Avast! контроль электронной почты и защиту от сетевых червей будет выполнять резидентный антивирусный монитор Avast!. Авторы этой программы называют его сканером доступа (рис. 4.2). А проверить жесткие диски поможет обычная антивирусная программа Avast! (рис. 4.3).

Антивирусные программы ищут и обнаруживают вредоносное программное обеспечение, используя так называемые антивирусные базы с информацией о возможных вирусах. Такая информация хранится в виде сигнатур. *Сигнатура* — некоторая метка, со-

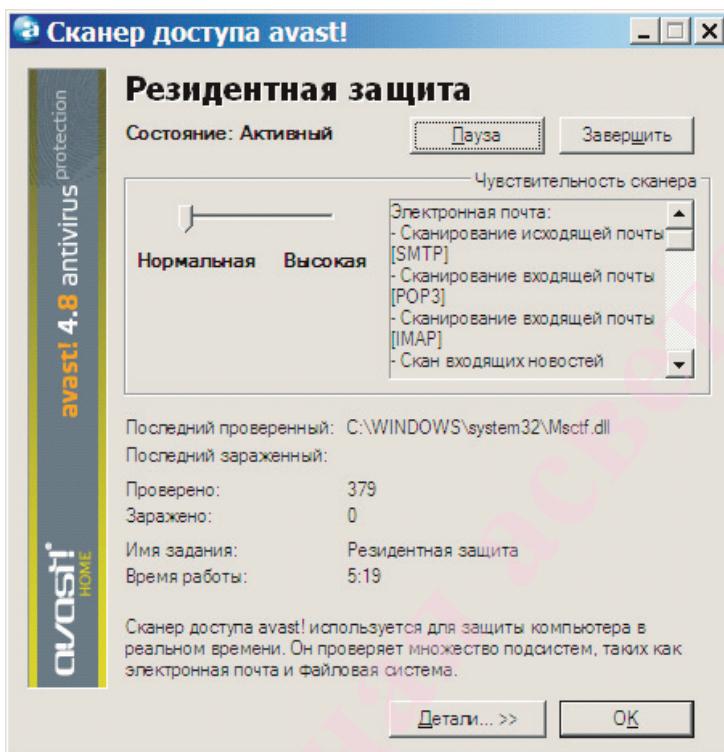


Рис. 4.2

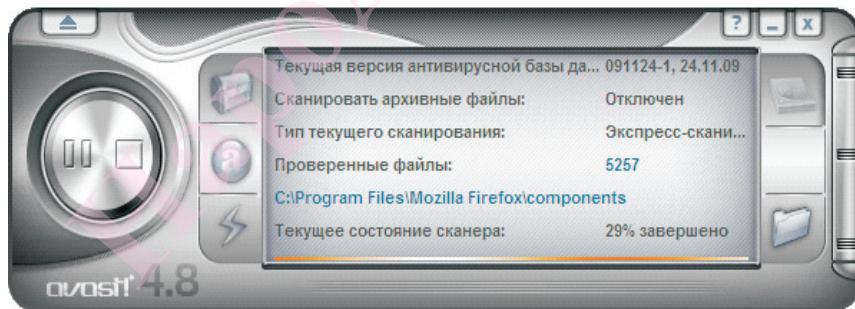


Рис. 4.3

ответствующая конкретной вредоносной программе. Понятно, что для поиска новых вирусов старых баз недостаточно, поэтому антивирусные базы нуждаются в обновлении.

ся в постоянном обновлении. Некоторые антивирусные программы ищут вирусы с помощью алгоритмов *эвристического анализа*. Тогда в ходе анализа отслеживаются все действия, характерные для вирусов и вредоносных программ других типов. В таком случае антивирусные базы не нужны.

Если антивирусная программа обнаружила «подозрительные» объекты, то пользователю предлагаются возможные действия на выбор:

1. Только отчет.
2. Лечить зараженные файлы. Если вылечить невозможно, то:
 - удалять зараженные файлы;
 - переименовывать, перемещать зараженные вирусами файлы;
 - запрашивать пользователя о дальнейших действиях.
3. Сразу удалять зараженные файлы.

Недопустимо одновременно использовать на одном компьютере несколько антивирусных программ с комплексной защитой.

22.2. Защита информации

С самого начала следует уяснить один факт: ничего не может быть по-настоящему безопасным. Полной защиты от всех вредоносных программ не существует. Что же делать? Как защитить компьютер? Ведь в современном мире уже невозможно отказаться от работы на компьютере, поиска информации в Интернете, общения с друзьями в сети. Начнем с первого правила: надежность работы вашего компьютера во многом зависит от применяемых Вами мер защиты.

Основным средством защиты информации является резервное копирование наиболее ценных данных. Отдельно надо сохранять все пароли для доступа к услугам Интернета. Их следует хранить на внешнем диске, например на флеш-диске. После просмотра паролей надо сразу отключать этот диск.

Опишем тактику борьбы с вредоносными программами:

- 1) не допускать, чтобы вредоносные программы попадали на ваш компьютер;
- 2) если они к Вам попали, ни в коем случае не запускать их на выполнение;
- 3) если вредоносные программы все же запустились, принять меры, чтобы они не причинили вреда.

Не допускать попадания вредоносных программ помогут следующие действия:

- использование лицензионных программных продуктов;
- проверка на наличие вирусов всех поступающих извне данных, в том числе через диски, флеш-диски, а также по любым сетям;
- периодическая проверка всех дисков вашего компьютера на наличие вирусов;
- использование антивирусных программных продуктов известных производителей и регулярное обновление вирусной базы;
- ограничение доступа к компьютеру посторонних лиц, например друзей с компьютерными играми неизвестного происхождения;
- использование двух профилей на своем компьютере: пользовательского и администраторского. Новые программы устанавливаются под профилем

администратора, а работа выполняется с правами пользователя. Это не позволит большинству вредоносных программ инсталлироваться на вашем компьютере;

- отключение автозапуска со сменных носителей;
- настройка по «зонам безопасности» браузера, например Internet Explorer. Прежде чем открыть сайт в Интернете, браузер проверит его соответствие заданной зоне безопасности: низкий, средний или высокий уровня защиты. Такая настройка поможет бороться с веб-червями;
- использование персонального сетевого экрана. Его называют также брандмауэр или файервол (Firewall). Эта программа фильтрует входящую и исходящую информацию, вырезает рекламу. Она поможет защитить ваш компьютер от интернет-мошенников.

! Не запускайте на выполнение неизвестные программы и подозрительные ссылки в почтовом сообщении!

Какие следует принять меры, если произошло любое неожиданное событие, и Вы утратили важные данные на компьютере? Напомним, что основным средством защиты информации является резервное копирование ваших данных. Копии должны быть не менее двух, и храниться они должны в разных местах.

Надо сказать, что большинство случаев заражения компьютера не носит фатальных последствий для данных, хранящихся на компьютере. В особо тяжелых случаях жесткие диски переформатируют и готовят к новой эксплуатации. На «чистый» отфор-

матированный диск устанавливают операционную систему, затем устанавливают все необходимое программное обеспечение. Восстановление компьютера завершается восстановлением данных, которые берут с резервных носителей.

После установки всего программного обеспечения можно создать образ жесткого диска на внешнем носителе. При любых последующих сбоях на компьютере будет достаточно восстановить сохраненный образ диска.

- ! Основным средством защиты информации является резервное копирование наиболее ценных данных.
- ? 1. Для чего предназначены антивирусные программы?
2. Приведите примеры антивирусных программ.
3. Назовите способы защиты от вредоносного программного обеспечения.

Упражнения

1. На сайте <http://www.kaspersky.ru> представлен аналитический обзор вредоносного программного обеспечения. Просмотрите рейтинг вредоносных программ за последний месяц. Определите наиболее опасные программы и программы-вирусы, которые скачиваются из сети Интернет чаще всего.
2. Проверьте указанный учителем файл на наличие вирусов.

Если Вы не знаете, какая антивирусная программа установлена у вас на компьютере и как с ней работать, можно проверить файл следующим образом:

- щелкнуть по значку файла (папки, диска) правой кнопкой мыши;

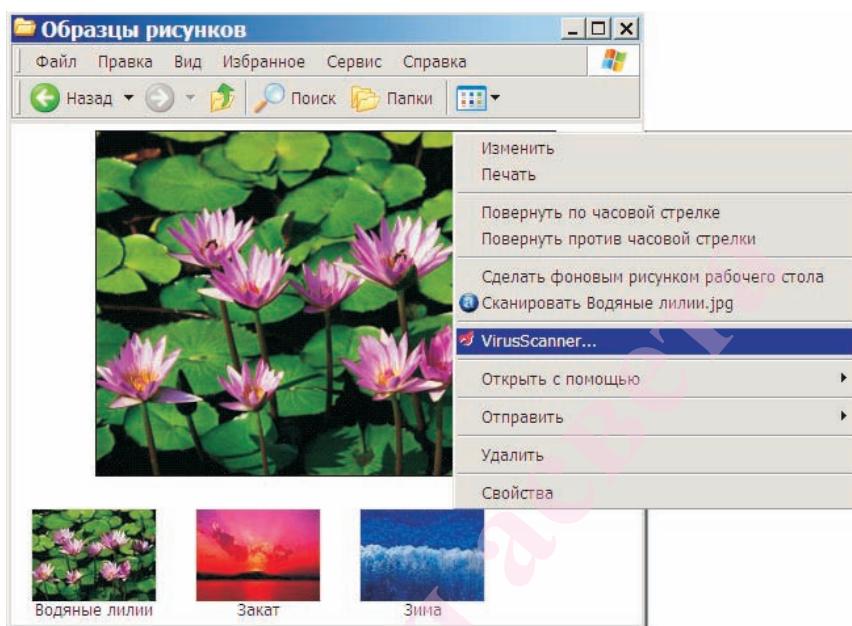


Рис. 4.4

- в контекстном меню выбрать команду Проверить на вирусы или аналогичную ей: Сканировать файл, VirusScanner... (название команды зависит от установленной антивирусной программы) (рис. 4.4).

Файл будет проверен той антивирусной программой, которая установлена на вашем компьютере.

3. Исследуйте папку, указанную учителем, на наличие вирусов с помощью антивирусной программы. Последовательность действий описана в упражнении 2.

4. Исследуйте флеш-диск на наличие вирусов. Последовательность действий описана в упражнении 2.

Глава 5

Работа с электронной почтой

§ 23. Понятие об электронной почте. Программы для работы с электронной почтой

23.1. Понятие о почтовом сервере. Адрес электронной почты

Электронная почта, или e-mail (от англ. electronic mail) — способ отправки и получения сообщений с помощью компьютерной сети. Электронное сообщение или письмо состоит из передаваемых данных и заголовка. В заголовок помещают сведения, необходимые для передачи.

Электронная почта построена по принципу клиент-сервер. Это означает, что на отдельный компьютер в сети устанавливают серверную часть программы электронной почты. Этот компьютер называют **почтовым сервером**. Он, подобно обычному почтовому отделению, обеспечивает прием, хранение и передачу писем, только в электронном виде. Клиентом же называется программа, расположенная на любом компьютере, который обращается к серверу и забирает оттуда почту.

В Интернете письмо доставляется адресату через цепочку промежуточных почтовых серверов. Механизм передачи почты следующий:

**КЛИЕНТ ОТПРАВИТЕЛЯ → СЕРВЕР 1 →
→ СЕРВЕР 2 → ... → СЕРВЕР N →
→ КЛИЕНТ ПОЛУЧАТЕЛЯ.**

Основные функции почтового сервера: создание и ведение почтовых ящиков, взаимодействие с клиентами (прием и передача писем), взаимодействие с другими серверами.

Сервер функционирует без перерывов и выходных. Пользователь может подключиться к нему в любое время, забрать и отправить письма. Для хранения писем разных типов на сервере заведены специальные папки **Входящие**, **Отправленные**, **Корзина**.

В папке **Входящие** хранятся файлы полученной почтовой корреспонденции. Содержание папки пополняется после обращения к почтовому ящику с запросом новой почты. Папка **Отправленные** хранит отправленные письма, а папка **Корзина** содержит удаленные письма перед их окончательным удалением.

Чтобы письмо попало в конкретный почтовый ящик, на сервере заведены учетные записи на всех пользователей данного сервера. Обязательная часть такой записи — **учетное имя**, которое является названием почтового ящика.

Например, для учащегося средней школы № 27 г. Гомеля Ивана Иванова заведен почтовый ящик с именем *Ivan_Ivanov*. Адрес электронной почты в данном случае будет выглядеть так:

Ivan_Ivanov@school_27.gomel.by

Кому

Куда

Адрес состоит из двух частей: «Кому» и «Куда». Разделяются они значком @, который обозначает предлог at. Адрес можно прочесть так: почтовый ящик Ивана Иванова находится по адресу *school_27.gomel.by*.

Часть «Куда» является доменным адресом. Он состоит из доменов разного уровня. Уровни считают справа налево:

by (Беларусь) — домен первого уровня,
gomel (Гомель) — домен второго уровня,
school_27 (школа № 27) — домен третьего уровня.

«Учетное имя», «имя пользователя», «идентификатор пользователя», «имя почтового ящика», «логин», «username» — все эти термины являются синонимами. Они обозначают имя, которое будет использовать человек при входе на почтовый сервер. Такое имя состоит из любых латинских букв и может разделяться некоторыми знаками, например точкой или нижним подчеркиванием. Его вид может быть самым разнообразным. Чаще всего используют личные данные: фамилию, имя, отчество. Например:

Ivan_Ivanov	I_Ivanov	I.Ivanov
Ivanov_I	IvanovI	Ivanov.Ivan

23.2. Почтовые клиенты

К своему почтовому ящику пользователь может обратиться с помощью программы-клиента. Почтовым клиентом называется программа, позволяющая работать с электронной почтой — получать, читать, составлять, отправлять и хранить письма. Выделяют *два способа взаимодействия* с сервером и соответственно два типа почтовых клиентов.

В первом случае компьютер пользователя соединяется с компьютером поставщика услуг. После установки соединения почтовый клиент принимает и отправляет почту. Такое взаимодействие называют *удаленным*. Для него подходят программы-клиенты, устанавливаемые на удаленных компьютерах. Наиболее популярные программы: Microsoft Outlook Express — поставляется вместе с операционной сис-

темой; Microsoft Outlook — входит в состав офисного пакета Microsoft Office; и независимые программы: The Bat!, Eudora, Mozilla Thunderbird.

Во втором случае пользователь подключается к Интернету и работает со своим почтовым ящиком, как с обычной веб-страницей, т. е. получатель читает почту прямо с сервера через сайт в Интернете. Такой способ называют **онлайн-взаимодействием**. Этот способ позволяет человеку с любого компьютера любого города посещать сайт, где зарегистрирован его почтовый ящик.

Интерфейс почтового клиента зависит от способа взаимодействия и используемой программы. Заголовок окна, меню и панель инструментов различных почтовых программ похожи на аналогичные элементы других программ Windows или браузеров. Внешний вид, состав сервисов или другие элементы программ могут отличаться в зависимости от настроек. Но основные элементы остаются неизменными.

Перечислим наиболее часто используемые операции программ-клиентов: прочитать письмо, написать новое, ответить, переслать. При работе с помощью удаленного взаимодействия добавляются операции: получить и отправить почту.

- ?
- 1. Что такое электронная почта?
- 2. Как доставляется адресату письмо в Интернете?
- 3. Что хранится в паках **Входящие**, **Отправленные**, **Корзина**?

§ 24. Создание электронного почтового ящика

Отправлять или принимать электронную почту можно как с помощью компьютера, так и с помощью мобильного телефона. Для этого надо создать элект-

ронный ящик в Интернете на любом из почтовых серверов. Чтобы получить бесплатный почтовый ящик на таком сервере, необходимо зарегистрироваться.

Шаг 1. Запустите программу Microsoft Internet Explorer командой Пуск → Программы → Internet Explorer или с помощью значка на Рабочем столе. В адресной строке браузера введите адрес нужного сайта. Выберите ссылку **Зарегистрироваться**.

Шаг 2. Ознакомьтесь с правилами пользования услугами электронной почты.

Шаг 3. Заполните регистрационную анкету. Обычно вопросов в анкете не много. Поясним некоторые из них.

Заранее следует придумать имя пользователя и пароль. Имя (логин) требуется для идентификации пользователя, а пароль — для подтверждения этого имени. Имя пользователя должно быть уникальным. Можно выбрать то имя, которое предложит почтовый сервер. Минимальная длина пароля — 6 символов. Обязательно запомните пароль! Желательно сразу записать свой адрес и пароль в записную книжку.

Легкий «секретный вопрос» поможет в будущем, если Вы забудете пароль. Таким секретным вопросом может быть: девичья фамилия матери, любимое блюдо, дата рождения бабушки и т. п.

Шаг 4. Подтвердите данные, нажав кнопку **Зарегистрироваться**. После успешной регистрации появится Ваш личный электронный адрес.

§ 25. Элементы электронного письма

25.1. Элементы электронного письма

Сообщение электронной почты состоит из двух разделов: заголовка сообщения и основного раздела. Заголовок имеет поля: Кому, Копия, Тема, Информация о наличии вложенных файлов.

Основной раздел сообщения обычно содержит: обращение, текст сообщения и подпись. Следует осмысленно подписывать свои письма. Можно использовать автоматически создаваемую подпись, содержащую некоторую информацию об авторе письма: полное имя, контактную информацию.

25.2. Действия пользователя при работе с клиентом

Рассмотрим действия пользователя по отправке и приему электронной почты.

Создание новых сообщений. Для создания нового письма надо открыть свой почтовый ящик, щелкнуть мышью на ссылке **Написать письмо** и заполнить основные поля нового сообщения.

Сначала нужно указать, кому предназначено письмо. В поле **Кому** записывается электронный адрес получателя. В поле **Копия** вводятся адреса людей, которым предназначена копия данного письма. Все адреса полученных ранее писем сохраняются в адресной книге. При указании, кому предназначено письмо, достаточно выбрать имя адресата.

Особо важным является поле **Тема**. Здесь следует четко и коротко описывать содержание или назначение письма. Если в приходящей почте есть письма с заполненным непонятно или пустым полем **Тема**, то

возможно это — вредоносное сообщение. Такие письма желательно удалять, не читая.

В следующей таблице приведен пример заголовка сообщения, которое адресовано Иванову и Петрову с краткой формулировкой темы письма.

Кому	Ivan_Ivanov@sch27.gomel.by
Копия	Petrov@sch27.gomel.by
Тема	Приглашение на бал-маскарад
Информация о наличии вложенных файлов	Программа бала-маскарада

В основном разделе пишется текст письма. Некоторые программы-клиенты позволяют форматировать текст, изменяя его шрифт и цвет.

Прикрепление файлов к письму. Механизм почтовых вложений позволяет пересылать вместе с текстовыми сообщениями мультимедийные, архивные и любые другие файлы. Для этого надо щелкнуть по ссылке **Прикрепить файл** и выбрать нужный файл. Значок «скрепка» покажет, что у полученного сообщения есть вложенный файл.

- !
 - Не прикрепляйте к письму много файлов.
 - Не пересылайте крупные файлы без предварительного согласования с получателем.

Отправка сообщения. После указания адреса и создания письма следует отправить сообщение адресату. Для этого надо нажать кнопку **Отправить**. Если в это время Вы работали с подключением к Интернету, письмо сразу будет отправлено. Если Вы работали без подключения к Интернету, то при нажатии

кнопки **Отправить** появится предложение установить соединение. После выхода в Интернет начнется процесс отправки почты.

- ! Итак, для того чтобы создать и отправить сообщение электронной почты, надо:
- открыть свой почтовый ящик;
 - щелкнуть левой кнопкой мыши на ссылке **Написать**;
 - заполнить поле **Кому**;
 - заполнить поле **Тема**;
 - написать текст письма;
 - проверить правописание;
 - подписать письмо (указать свою фамилию, имя, класс);
 - щелкнуть на ссылке **Отправить**;
 - проверить, отправлено ли письмо.

Вместо отправки письма можно сохранить его в папке **Черновики**.

Получение сообщения. Электронные письма, которые поступают на Ваш адрес, хранятся в личном почтовом ящике на почтовом сервере. Для доставки писем предназначена кнопка **Проверить почту**. После установления соединения с Интернетом ваша почта будет отправлена и появится оповещение о прибытии новой почты. Открыть пришедшее сообщение можно, дважды щелкнув по нему мышью.

Прочитанные сообщения могут быть подвергнуты обработке. Для этого созданы специальные элементы управления. Кнопка **Ответить** избавляет от необходимости вводить адрес того, кому Вы отвечаете. А кнопка **Переслать** позволяет переслать письмо кому-то еще.

Пересылка сообщений. Для пересылки полученного письма необходимо нажать на ссылку **Переслать**. Далее вводится или выбирается из адресной книги нужный адрес. К пересылаемому письму желательно добавить свои комментарии. Подготовленное письмо пересылается кнопкой **Отправить**.

Сохранение сообщений. Недописанное письмо можно сохранить в папке **Черновики**, нажав кнопку **Сохранить как черновик**. Позже это письмо можно будет дописать и отправить.

25.3. Сетевой этикет и меры безопасности при переписке

Сетевой этикет — понятие, возникшее с появлением электронной почты. В самом общем виде оно объединяет правила поведения, общения в Интернете, оформления писем в электронной почте. Правила сетевого этикета просты и похожи на правила поведения в реальной жизни.

Приведем небольшую **памятку о правилах поведения** при пользовании электронной почтой.

1. При подготовке сообщений заполняйте все поля заголовка.

2. При составлении ответа на письмо цитируйте некоторые его отрывки. Это поможет адресату вспомнить, о чем он писал.

3. Обязательно отвечайте на личные письма. Хотя бы сообщите, что письмо получено.

4. Не пишите все письмо **ПРОПИСНЫМИ** буквами. Такая запись воспринимается как крик. К тому же такой текст плохо читается.

5. Архивируйте файлы, которые отправляются как вложение. Тогда адресат быстрее загрузит их.

6. Соблюдайте правила вежливости. Помните, что Вы говорите с человеком. Избегайте отсылки писем, составленных под влиянием эмоций.

7. В личной переписке можно использовать смайлики, обозначающие эмоции. Например:

Текстовый код	Выражаемая эмоция
:)	Smile — Улыбка
;)	Wink — Подмигивание
:oops:	Oops — Ой
:("	Sad — Грустный

Полезной будет также **памятка по обеспечению безопасности** во время работы с электронной почтой.

1. Регистрируя почтовый ящик, придумывайте сложные пароли и секретные вопросы. Часто меняйте пароль.

2. Никогда не отвечайте на сообщение из неизвестного источника.

3. Никогда не отвечайте на письма с просьбой или требованием прислать на указанный в письме адрес свой логин и пароль.

4. Вводите пароли вручную. Если Вы разрешите **Сохранить пароль**, он будет храниться на диске вашего компьютера в файле со стандартным расширением и может быть украден путем получения доступа к диску.

5. После работы на чужом компьютере вручную очистите информацию, которую браузер автоматически сохраняет на диске. Очистить следует «Историю», «кэш» и «Автозаполнение» браузера, после

чего надо закрыть все окна. В Internet Explorer это выполняется так:

Сервис → Свойства обозревателя → Журнал → Общие → Очистить;

Сервис → Свойства обозревателя → Временные файлы Интернета → Удалить.

6. Не включайте в письмо очень личную информацию, например пароли.

7. Никогда не отправляйте исполняемые файлы, например с расширениями .exe, .com, .bat. Так распространяются опасные вирусы. Если все же необходимо переслать такой файл, то заархивируйте его.

8. Перед загрузкой писем из ящика просматривайте их список через веб-интерфейс почтового сервиса и удаляйте письма, содержащие вирусы и спам.

9. Используйте антивирусные программы, проверяя ими все вложения в присылаемых вам письмах.

10. Зарегистрируйте несколько почтовых ящиков. Один для доверенных людей, другой — для всех остальных.

! Каждое ваше слово в письме — это слово, обращенное к человеку. По вашим словам о Вас будут судить другие люди. Помните об этом.

- ? 1. Из каких разделов состоит сообщение электронной почты?
2. Для чего в сообщении указывается тема?
3. Каковы основные этические правила при общении по электронной почте?

Упражнение

Рассмотрите предложенный текст.

Добрый день.

Бал-маскарад начнется в 17.00.

С уважением, Мария Ивановна.

> Уважаемая Мария Ивановна!

> Спасибо за приглашение. :)

> Уточните, пожалуйста, время начала бала.

> Всего Вам доброго! С уважением, Ваня Иванов.

>> Дорогие ребята!

>> Приглашаю Вас на бал-маскарад, который состоится 15 декабря.

>> Программу высылаю вложенным файлом.

>> С уважением, Мария Ивановна.

Сколько сообщений в данной переписке?

Кто первый отправил письмо?

Какие элементы писем можно сделать автоматически создаваемой подписью?

СОДЕРЖАНИЕ

От авторов	3
Глава 1. Основы алгоритмизации и программирования	
§ 1. Алгоритмическая конструкция ВЕТВЛЕНИЕ	5
1.1. Принятие решений в программе	—
1.2. Простые условия	6
1.3. Составные условия	8
1.4. Полная и сокращенная формы условного оператора	10
1.5. Составной оператор	12
§ 2. Составление и реализация алгоритмов с ветвлением	17
§ 3. Алгоритмическая конструкция ПОВТОРЕНИЕ	25
3.1. Цикл (повторение)	—
3.2. Оператор цикла с параметром	27
3.3. Оператор цикла с предусловием	34
§ 4. Составление и реализация алгоритмов с повторением	42
Глава 2. Технология обработки текстовых документов	
§ 5. Поиск и замена в тексте, проверка правописания	59
5.1. Функции поиска и замены	—
5.2. Проверка правописания	61
§ 6. Создание и форматирование списков	65
6.1. Маркированный список	—
6.2. Нумерованный список	67
6.3. Вложенные списки	—
§ 7. Создание и форматирование таблиц	70
7.1. Таблица	—
7.2. Создание и редактирование таблиц	71
7.3. Оформление таблицы	76
§ 8. Создание и форматирование колонок	79
§ 9. Вставка декоративного текста	82
9.1. Создание декоративного текста	—
9.2. Использование декоративного текста	84
§ 10. Рисунки в документе	87
10.1. Вставка рисунков в документ	—
10.2. Использование коллекции клипов	89

§ 11. Вставка формул	93
11.1. Простые формулы	—
11.2. Объект Microsoft Equation	95
§ 12. Нумерация страниц и подготовка документов к печати	100
12.1. Формат бумаги и поля	—
12.2. Колонтитулы и нумерация страниц	102
12.3. Средство расстановки переносов	103
12.4. Предварительный просмотр документа	104
12.5. Настройка параметров печати	106
§ 13. Применение текстового процессора в разработке документов из различных областей	110
13.1. Подготовка к работе с новым документом	—
13.2. Набор и форматирование элементов документа	112

Глава 3. Работа с векторной графикой

§ 14. Понятие векторного изображения	120
14.1. Векторная и растровая графика	—
14.2. Представление о цветовых моделях	121
§ 15. Векторный графический редактор	125
15.1. Назначение графического редактора	—
15.2. Элементы интерфейса графического редактора	126
§ 16. Создание векторного изображения	131
16.1. Объекты CorelDRAW	—
16.2. Инструменты рисования	132
16.3. Заливка и абрис	136
16.4. Особенности интеллектуальной заливки	138
16.5. Стили заливки	140
§ 17. Редактирование векторного изображения	142
17.1. Выделение	—
17.2. Трансформация	143
17.3. Редактирование контура	144
17.4. Взаимное расположение	146
§ 18. Операции над объектами векторного изображения	148
18.1. Дублирование	—
18.2. Выравнивание, распределение	150

18.3. Формирование	151
18.4. Группировка	152
§ 19. Работа с текстом	153
19.1. Создание	—
19.2. Форматирование	154
19.3. Редактирование	156
§ 20. Сохранение и загрузка изображений	157
20.1. Сохранение изображения	—
20.2. Типы файлов	158
20.3. Загрузка изображения	159
20.4. Свойства документа	160

Глава 4. Вредоносное программное обеспечение и защита информации

§ 21. Виды вредоносного программного обеспечения	162
§ 22. Способы защиты от вредоносного программного обеспечения	167
22.1. Антивирусные программы	—
22.2. Защита информации	170

Глава 5. Работа с электронной почтой

§ 23. Понятие об электронной почте. Программы для работы с электронной почтой	175
23.1. Понятие о почтовом сервере. Адрес электронной почты	—
23.2. Почтовые клиенты	177
§ 24. Создание электронного почтового ящика	178
§ 25. Элементы электронного письма	180
25.1. Элементы электронного письма	—
25.2. Действия пользователя при работе с клиентом	—
25.3. Сетевой этикет и меры безопасности при переписке	183

Учебное издание

**Миняйлова Елена Леонидовна
Вербовиков Дмитрий Александрович
Коледа Наталья Ремовна
Якунина Нина Владимировна**

ИНФОРМАТИКА

Учебное пособие для 8 класса
общеобразовательных учреждений
с русским языком обучения

Зав. редакцией *В. Г. Бехтина*. Редактор *Н. М. Алганова*. Художественный
редактор *Л. А. Дашикевич*. Технический редактор *Г. А. Дудко*. Корректоры
Д. Р. Лосик, З. Н. Гришели, В. С. Бабеня, А. В. Алешко.

Подписано в печать 22.04.2010. Формат 60×90¹/₁₆. Бумага офсетная.
Гарнитура школьная. Офсетная печать. Усл. печ. л. 12. Уч.-изд. л. 8.
Тираж 91 150 экз. Заказ .

Издательское республиканское унитарное предприятие
«Народная асвета» Министерства информации Республики Беларусь.
ЛИ № 02330/0494083 от 03.02.2009.
Пр. Победителей, 11, 220004, Минск.

Республиканское унитарное предприятие
«Минская фабрика цветной печати».
ЛП № 02330/0494156 от 03.04.2009.
Ул. Корженевского, 20, 220024, Минск.

Правообладатель Народная асвета

**Информатика : учеб. пособие для 8-го кл. общеобразо-
И74 ват. учреждений с рус. яз. обучения / Е. Л. Миняйлова
[и др.]. — Минск : Нар. асвета, 2010. — 189 с. : ил.**

ISBN 978-985-03-1342-3.

**УДК 004(075.3=161.1)
ББК 32.81я721**

Народная асвета

(Название и номер школы)

Учебный год	Имя и фамилия ученика	Состояние учебного пособия при получении	Оценка ученику за пользование учебным пособием
20 /			
20 /			
20 /			
20 /			
20 /			