Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Кузбасский государственный технический университет
имени Т. Ф. Горбачева»

Кафедра информационных и автоматизированных
производственных систем

Составители
Е. А. Рябова Е. А. Игнатьева

ИНФОРМАТИКА

Рекомендовано учебно-методической комиссией специальности
15.05.01
«Проектирование технологических машин и комплексов»
профиль 11 «Проектирование механообрабатывающих
и инструментальных комплексов в машиностроении»
в качестве электронных изданий
для использования в учебном процессе

Кемерово 2019

Рецензенты:

А. Н. Коротков – доктор технических наук, профессор, заведующий кафедрой металлорежущих станков и инструментов

И. В. Чичерин – кандидат технических наук, доцент, заведующий кафедрой информационных и автоматизированных производственных систем

Рябова Елена Анатольевна
Игнатьева Елена Александровна

Информатика [Электронный ресурс]: методические указания к лабораторным работам для студентов обучающихся 15.05. Проектирование технологических машин и комплексов всех форм обучения / сост. Е. А Рябова, Е. А. Игнатьева; КузГТУ. – Электрон. дан. – Кемерово, 2019. – Систем. требования : Pentium IV; ОЗУ 8 Мб; Windows 95; мышь. – Загл. с экрана.

В данных методических указаниях изложено содержание лабораторных работ, порядок их выполнения и контрольные вопросы к ним.

Лабораторная работа № 1
Основы позиционных систем счисления

1. ЦЕЛЬ РАБОТЫ

Целью работы является ознакомление студентов с позиционными системами счисления, с компьютерным представлением чисел, алгоритмами перевода чисел из одной системы счисления в другую и получение практических навыков перевода чисел.

2. ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Система счисления – это способ записи чисел с помощью заданного набора специальных знаков (цифр).

Система счисления должна обладать следующими свойствами:

– простота способа записи чисел на материальном носителе;

– удобство выполнения арифметических операций над числами в предложенной записи;

– наглядность обучения основам работы с числами.

Классификация позиционных систем счисления

Все многообразие систем счисления можно разделить на две группы:

1. Аддитивные (непозиционные) системы счисления.

2. Позиционные системы счисления.

Аддитивные системы счисления – системы записи чисел, в которых каждой цифре в записи числа соответствует величина, не зависящая от местонахождения этой цифры в записи числа.

Наиболее известной из аддитивных систем счисления является римская система счисления. В ней для обозначения чисел используются буквы латинского алфавита: I, V, X, L, C, D и M. Число в римской системе счисления обозначается набором стоящих подряд знаков. Значение числа определяется следующим образом:

1) суммируются значения идущих подряд нескольких одинаковых знаков (группа первого вида);

2) вычитаются значения двух знаков, если слева от большего знака стоит меньший, то есть от значения большего знака отнимается значение меньшего (группа второго вида).

Позиционные системы счисления – системы записи чисел, в которых значение цифры в записи числа зависит от ее позиции или местонахождения в числе.

Совокупность различных цифр, используемых в позиционной системе счисления для записи чисел, называется алфавитом системы счисления.

Базис позиционной системы счисления – это последовательность чисел, каждое из которых задает значение цифры по ее месту в записи числа, то есть «вес» каждого разряда.

Приведем в качестве примера базисы некоторых традиционных систем счисления:

Десятичная система счисления: 1, 10, 10^2, 10^3, …, 10^n.

Двоичная система счисления: 1, 2, 2^2, 2^3, …, 2^n.

Восьмеричная система счисления: 1, 8, 8^2, 8^3, …, 8^n.

Позиционные системы счисления бывают традиционными и нетрадиционными.

Системы счисления, базис которых образуют члены геометрической прогрессии, называют традиционными.

К традиционным системам счисления относятся двоичная, восьмеричная, десятичная и шестнадцатеричная системы счисления. Для записи чисел в десятичной системе счисления используются цифры 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. В восьмеричной системе счисления: 0, 1, 2, 3, 4, 5, 6, 7, а восьмерка записывается двумя цифрами 10. Для записи чисел в шестнадцатеричной системе счисления необходимо располагать уже 16-ю различными символами, используемыми как цифры. В качестве первых десяти шестнадцатеричных цифр используются те же, что и в десятичной системе (0, 1, …, 9), для обозначения остальных шести цифр (в десятичной системе они соответствуют числам 10, 11, 12, 13, 14, 15) используют буквы латинского алфавита – A, B, C, D, E, F.

Позиционные системы счисления принято делить на следующие три класса:

1. Системы с целочисленным основанием:

- единичная (унарная) система счисления, простейшая из позиционных систем счисления;
- двоичная (применяется в дискретной математике, информатике и программировании);
- троичная система счисления;
- восьмеричная (применяется в программировании);
- десятичная система счисления;
- двенадцатеричная (широко использовалась в древности, в некоторых областях используется и сейчас);
- шестнадцатеричная (наиболее распространена в программировании, а также в шрифтах);
- сорокаичная система счисления (применялась в древности («сорок сороков = 1600»));
- шестидесятеричная (применяется при измерении углов и, в частности, долготы и широты).

2. Системы с отрицательным основанием (нега-позиционные):

- нега-двоичная система счисления (с основанием -2);
- нега-десятичная система счисления (с основанием -10).

Системы с нецелочисленным основанием:

2,71… = e – e-ричная система счисления с основанием, равным числу Эйлера (применяется в натуральных логарифмах).

В общем виде для традиционных позиционных систем счисления базис можно записать в следующем виде:

$$P^{-m},\ P^{-m+1},\ \ldots,\ P^{-2},\ P^{-1},\ 1,\ P,\ P^{2},\ldots,\ P^{n-1},\ P^{n}.$$

Знаменатель P геометрической прогрессии, члены которой образуют базис традиционной системы счисления, называется основанием системы счисления.

К нетрадиционным системам счисления относятся факториальная, фибоначчиевая и уравновешенная. Базисы этих систем счисления выглядят следующим образом:

– факториальная – 1!, 2!, …, (n-1)!, n!;

– фибоначчиевая – 1, 2, 3, 5, 8, 13, 21, …;

– уравновешенная троичная – -1, 0, 1.

Преобразование чисел

В общем случае перевод любого числа, состоящего из целой и дробной частей

$$A_P = \pm(a_n P^n + a_{n-1}P^{n-1} + \ldots + a_1 P^1 + a_0 P^0 + \\ + a_{-1}P^{-1} + \ldots + a_{-m}P^{-m}), \quad (1)$$

из системы счисления с основанием P_1 в систему с основанием P_2 может быть выполнен по универсальному алгоритму, который нетрудно определить, приведя целую и дробную части выражения к видам:

$$A_P^{u} = \pm((\ldots((a_n P + a_{n-1})P + a_{n-2})P + \cdots)P + a_0) \quad (2)$$

и

$$A_P^{др} = \pm P^{-1}(a_{-1} + P^{-1}(a_{-2} + \ldots + P^{-1}(a_{-(m-1)} + a_{-m}P^{-1})\ldots)) \quad (3)$$

Согласно этому алгоритму перевод числа, включающего в себя целую и дробную части, состоит из вычислительных процессов двух видов:

1) последовательного деления нацело целой части и образующихся целых частных на основание новой системы счисления;

2) последовательного умножения дробной части и дробных частей получающихся произведений на то же новое основание, записанное, как и в первом случае, цифрами исходной системы счисления.

При переводе целой части числа в ходе процесса последовательного деления получаются остатки, выраженные цифрами исходной системы счисления. Они представляют собой цифры a_0, a_1, …, a_n целой части числа, переведенного в новую систему счисления. Последний остаток является старшей цифрой переведенного числа. Процесс деления продолжается до тех пор, пока частное не станет равным нулю.

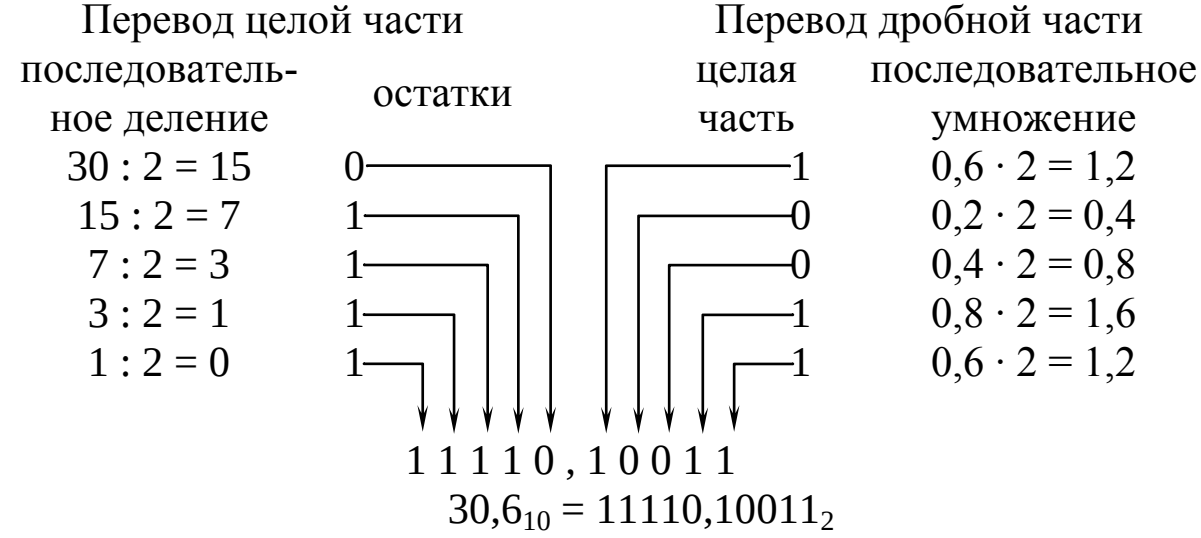
При переводе дробной части числа при каждом умножении получаются целые части, которые исключаются из последующих умножений. Эти целые части, изображенные цифрами исходной системы счисления, представляют собой цифры дробной части, переведенной в новую систему счисления. Значение первой целой части является первой цифрой после запятой переведенного числа. Процесс умножения продолжаем до тех пор, пока дробная часть произведения не станет равной нулю или не выделится период.

Следует отметить, что все арифметические действия осуществляются по правилам исходной системы счисления.

Пример 1.

Требуется перевести число 30,6 в десятичной системе счисления в двоичную систему счисления.

Согласно вышесказанному переводим отдельно целую и дробную части числа.



$30{,}6_{10} = 11110{,}10011_2$

Пример 2.

Требуется перевести число 111101,01 в двоичной системе счисления в десятичную систему счисления.

Перевод целой части		Перевод дробной части	
последовательное деление	остатки	целая часть	последовательное умножение
111101:1010 = 110	001	010	0,0100·1010=010,1000
110 : 1010 = 0	110	101	0,1000·1010=101,0000

6 1 , 2 5

$111101{,}01_2 = 61{,}25_{10}$

6 1 , 2 5

$111101{,}01_2 = 61{,}25_{10}$

По универсальному алгоритму наиболее просто переводить числа из десятичной системы счисления в любую другую.

Для перевода числа из любой системы счисления в десятичную систему можно воспользоваться представлением исходного числа как суммы произведений цифр данного числа и степеней основания исходной системы счисления. Алгоритм такого перевода выглядит следующим образом (действия проводятся в десятичной системе счисления):

1) цифры исходного числа нумеруются справа налево начиная с нуля – для целой части числа и слева направо начиная с **–1** – для дробной части;

2) каждая цифра числа переводится в число в десятичной системе;

3) десятичное число, соответствующее каждой цифре, умножается на основание исходной системы счисления в пронумерованной степени, и результаты складываются.

Пример 3.

Требуется перевести число 111101,01 в двоичной системе счисления в десятичную систему счисления:

$$1^5 1^4 1^3 1^2 0^1 1^0 , 0^{-1} 1^{-2}{}_2 =$$

$$= 1\cdot 2^5 + 1\cdot 2^4 + 1\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 + 0\cdot 2^{-1} + 1\cdot 2^{-2} = 61{,}25_{10}$$

По универсальному алгоритму наиболее просто переводить числа из десятичной системы счисления в любую другую.

Для перевода числа из любой системы счисления в десятичную систему можно воспользоваться представлением исходного числа как суммы произведений цифр данного числа и степеней основания исходной системы счисления. Алгоритм такого перевода выглядит следующим образом (действия проводятся в десятичной системе счисления):

1) цифры исходного числа нумеруются справа налево начиная с нуля – для целой части числа и слева направо начиная с -1 – для дробной части;

2) каждая цифра числа переводится в число в десятичной системе;

3) десятичное число, соответствующее каждой цифре, умножается на основание исходной системы счисления в пронумерованной степени, и результаты складываются.

Пример 4.

Требуется перевести число 111101,01 в двоичной системе счисления в десятичную систему счисления:

$$1^5 1^4 1^3 1^2 0^1 1^0 , 0^{-1} 1^{-2}{}_2 =$$

$$= 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = 61{,}25_{10}$$

Пример 5А.

Требуется перевести число 3Е5А1 в шестнадцатеричной системе счисления в десятичную систему счисления:

$$3^4 E^3 5^2 A^1 1^0{}_{16} = 3 \cdot 16^4 + E \cdot 16^3 + 5 \cdot 16^2 + A \cdot 16^1 + 1 \cdot 16^0 =$$

$$= 3 \cdot 16^4 + 14 \cdot 16^3 + 5 \cdot 16^2 + 10 \cdot 16^1 + 1 \cdot 16^0 = 255393_{10}$$

Еще более простой алгоритм применяется для работы с системами счисления, основание которых представляет собой ту или иную степень числа 2. К таким системам счисления относятся: двоичная, восьмеричная, шестнадцатеричная и т. д. Иногда подобные системы счисления называют смешанными. Для записи любой цифры числа, представленного в системе счисления, основание которой 2 в любой степени, необходимо количество двоичных цифр, равное значению данной степени. Например, для записи восьмеричного числа – три двоичные цифры, для записи шестнадцатеричного – четыре.

Для того чтобы перевести двоичное число в восьмеричную систему счисления, преобразуемое число разделяют справа налево на группы по три двоичные цифры, при этом самая левая группа может содержать меньше трех двоичных цифр. Затем каждую группу двоичных цифр выражают в виде восьмеричной цифры с учетом того, что:

$000_2 = 0_8$ $011_2 = 3_8$ $110_2 = 6_8$
$001_2 = 1_8$ $100_2 = 4_8$ $111_2 = 7_8$
$010_2 = 2_8$ $101_2 = 5_8$

Аналогично преобразуют двоичное число в шестнадцатеричное с той лишь разницей, что преобразуемое двоичное число делят на группы по 4 двоичных цифры в каждой, поскольку для записи любой цифры шестнадцатеричного числа необходимы четыре двоичные цифры:

$0000_2 = 0_{16}$ $0100_2 = 4_{16}$ $1000_2 = 8_{16}$ $1100_2 = C_{16}$
$0001_2 = 1_{16}$ $0101_2 = 5_{16}$ $1001_2 = 9_{16}$ $1101_2 = D_{16}$
$0010_2 = 2_{16}$ $0110_2 = 6_{16}$ $1010_2 = A_{16}$ $1110_2 = E_{16}$
$0011_2 = 3_{16}$ $0111_2 = 7_{16}$ $1011_2 = B_{16}$ $1111_2 = F_{16}$

Преобразование восьмеричного или шестнадцатеричного числа в двоичное осуществляется простым переводом каждой цифры исходного числа в группу из трех (для восьмеричного числа) или четырех (для шестнадцатеричного числа) двоичных цифр.

Пример 5Б.

$$1101111011_2 = 1\ \ 101\ \ 111\ \ 011 = 1573_8$$
$$1101111011_2 = 11\ \ 0111\ \ 1011 = 37B_{16}$$
$$123_8 = 001\ \ 010\ \ 011 = 1010011_2$$
$$A17_{16} = 1010\ \ 0001\ \ 0111 = 101000010111_2$$

Арифметические операции

Арифметические действия над числами (сложение, вычитание, умножение и деление) в любой позиционной системе счисления производятся по тем же правилам, что и десятичной системе, так как все они основываются на правилах выполнения действий над соответствующими многочленами. При этом нужно

только пользоваться теми таблицами сложения и умножения, которые соответствуют данному основанию b системы счисления. Операции сложения и умножения производятся по тем же правилам, что и в десятичной системе счисления.

Пример 6.

Требуется сложить два числа в двоичной системе счисления:

$$\begin{array}{r} 1\ 0\ 1\ 1\ 1 \\ +\ 1\ 0\ 1\ 1\ 0 \\ \hline 1\ 0\ 1\ 1\ 0\ 1 \end{array}$$

Пример 7.

Требуется найти произведение двух чисел в двоичной системе счисления:

$$\begin{array}{r} 1\ 0\ 1\ 1 \\ \times\quad 1\ 0\ 1 \\ \hline 1\ 0\ 1\ 1 \\ +\ 1\ 0\ 1\ 1\ \ \\ \hline 1\ 1\ 0\ 1\ 1\ 1 \end{array}$$

Вычитание чисел в двоичной системе счисления может осуществляться двумя способами:

1) из большего по абсолютной величине числа вычитается меньшее, и у результата ставится соответствующий знак;

2) вычитаемое предварительно преобразуется в дополнительный код (перед преобразованием количество разрядов в числах выравнивается), после чего оба числа суммируются.

Для получения дополнительного кода отрицательного числа необходимо:

1) значения всех разрядов изменить на противоположные, т.е. все нули заменить единицами, а единицы нулями (получить обратный код исходного числа);

2) к полученному обратному коду прибавить единицу в младшем разряде.

Пример 8.

Требуется найти разность двух чисел в двоичной системе счисления: $10111_2 - 1101_2$

Первый способ:

$$\begin{array}{rrrrrr} & 1 & 0 & 1 & 1 & 1 \\ - & & 1 & 1 & 0 & 1 \\ \hline & & 1 & 0 & 1 & 0 \end{array}$$

Второй способ:

Получим дополнительный код числа 1101_2

1101 $\Rightarrow$ 01101 (*добавили один пустой разряд справа*)

01101 $\Rightarrow$ 10010 (*заменили* 0 $\Leftrightarrow$ 1)

10010 $\Rightarrow$ 10011 (*прибавили* 1 *в младшем разряде*)

$$\begin{array}{rrrrrr} & 1 & 0 & 1 & 1 & 1 \\ + & 1 & 0 & 0 & 1 & 1 \\ \hline \cancel{1} & 0 & 1 & 0 & 1 & 0 \end{array}$$

Игнорируем 1 переноса из старшего разряда и получаем результат, равный 01010_2.

При делении столбиком приходится в качестве промежуточных вычислений выполнять действия умножения и вычитания.

Пример 9.

Требуется найти частное от деления двух чисел в двоичной системе счисления:

$$\begin{array}{rrrrrr|rrr} & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ \cline{7-9} - & 1 & 1 & 0 & & & 1 & 0 & 1 \\ \cline{1-3} & & & 1 & 1 & 0 & & & \\ & & - & 1 & 1 & 0 & & & \\ \cline{3-5} & & & & & 0 & & & \end{array}$$

Представление данных в памяти ЭВМ

Основные положения

Любая информация (числа, команды, записи и т.п.) представляется в ЭВМ в виде двоичных кодов фиксированной или переменной длины. Отдельные элементы двоичного кода, имеющие значение 0 или 1, называют разрядами или битами. Двоичный код, состоящий из 8 разрядов, носит название байта. Для записи чисел также используют 32-разрядный формат (машинное слово), 16-разрядный формат (полуслово) и 64-разрядный формат (двойное слово).

В целях упрощения выполнения арифметических операций в ЭВМ применяют специальные коды для представления чисел, что позволяет свести операцию вычитания чисел к арифметическому сложению кодов этих чисел. Применяются так называемые прямой, обратный и дополнительный коды чисел.

Прямой код

Прямой код – способ представления двоичных чисел с фиксированной запятой в компьютерной арифметике. Главным образом используется для записи положительных чисел.

При записи числа в прямом коде старший разряд является знаковым. Если его значение равно 0 – то число положительное, если 1 – то отрицательное. В остальных разрядах, называемых цифровыми разрядами, записывается двоичное представление модуля числа.

Функция кодирования двоичных чисел (в том числе целых чисел и смешанных дробей) в прямом коде имеет вид:

$$[A]_{ПР} = \begin{cases} A, & A \geq 0 \\ 2^n + A, & A < 0 \end{cases}, \qquad (4)$$

где n – номер знакового разряда. В частности, при кодировании правильных двоичных дробей (то есть чисел $-1 < A < 1$), $n = 0$ и функция кодирования принимает вид:

$$[A]_{ПР} = \begin{cases} A, & A \geq 0 \\ 1 + A, & A < 0 \end{cases} \qquad (5)$$

Величина числа *A* в прямом коде определяется по следующей формуле:

$$A = (1 - 2a_{sign}) \sum_{i=-k}^{n} a_i p^i , \qquad (6)$$

где: a_{sign} – значение знакового разряда; число *A* имеет *k* разрядов справа от запятой (дробная часть) и *n* разрядов слева (целая часть), тут учитываются только цифровые разряды.

Как видно из последней формулы, знаковый разряд в прямом коде не имеет разрядного веса. При выполнении арифметических операций это приводит к необходимости отдельной обработки знакового разряда в прямом коде.

(*n* + 1)-разрядный прямой код (*n* цифровых разрядов и один знаковый) позволяет представлять целые числа в диапазоне $\left[-(2^n - 1); (2^n - 1)\right]$ и правильные двоичные дроби в диапазоне $\left[-(1 - 2^n); (1 - 2^n)\right]$.

Примеры представления чисел в прямом коде приведены в табл. 3.

Таблица 3

Представление чисел в прямом коде

Десятичный код	Двоичный код	8-разрядный прямой код
0	0	00000000 – положительный ноль
0	0	10000000 – отрицательный ноль
4	100	00000100
10	1010	00001010
-5	-101	10000101
-16	-10000	10010000
9/16	0.1001	0.1001000
-9/16	-0.1001	1.1001000
105/128	0.1101001	0.1101001
-5/128	-0.0000101	1.0000101

В информатике прямой код используется главным образом для записи неотрицательных целых чисел. Его легко получить из

представления целого числа в любой другой системе счисления. Для этого достаточно перевести число в двоичную систему счисления, а затем заполнить нулями свободные слева разряды разрядной сетки машины.

Однако у прямого кода есть два недостатка:

1. В прямом коде есть два варианта записи числа 0 (например, 00000000 и 10000000 в восьмиразрядном представлении).

2. Использование прямого кода для представления отрицательных чисел в памяти компьютера предполагает или выполнение арифметических операций центральным процессором в прямом коде, или перевод чисел в другое представление (например, в дополнительный код) перед выполнением операций и перевод результатов обратно в прямой код.

Выполнение арифметических операций над числами в прямом коде затруднено, т. к. даже для сложения чисел с разными знаками требуется, кроме сумматора, иметь специальный блок-«вычитатель». Кроме того, при выполнении арифметических операций требуется особо обрабатывать значащий разряд, так как он не имеет веса. Также необходима обработка «отрицательного нуля». Таким образом, выполнение арифметических операций над числами в прямом коде требует сложной архитектуры центрального процессора и является малоэффективным.

Гораздо более удобным для выполнения арифметических операций является дополнительный код.

Обратный код

Обратный код – метод вычислительной математики, позволяющий вычесть одно число из другого, используя только операцию сложения над натуральными числами. Ранее метод использовался в механических калькуляторах (арифмометрах). В настоящее время используется в основном в современной компьютерной технике.

Обратный *n*-разрядный двоичный код положительного целого числа состоит из одноразрядного кода знака (двоичной цифры 0), за которым следует (*n*–1)-разрядное двоичное представление модуля числа (обратный код совпадает с прямым кодом).

Обратный n-разрядный двоичный код отрицательного целого числа состоит из одноразрядного кода знака (двоичной циф-

ры 1), за которым следует (n-1)-разрядное двоичное число, представляющее собой инвертированное (n-1)-разрядное представление модуля числа.

У числа 0 имеется два обратных кода: «положительный нуль» 00000000 и «отрицательный нуль» 11111111 (приведены 8-разрядные обратные коды).

n-разрядный обратный код позволяет представить числа в диапазоне $\left[-(2^{n-1}-1);(2^{n-1}-1)\right]$.

Примеры представления чисел в обратном коде представлены в табл. 4.

Таблица 4

Примеры представления чисел в обратном коде

Десятичный код	Двоичный код	8-разрядный обратный код
0	0	00000000
-0	0	11111111
5	101	00000101
-5	-101	11111010

Дополнительный код

Дополнительный код – наиболее распространенный способ представления отрицательных целых чисел в компьютерной арифметике. Он позволяет заменить операцию вычитания на операцию сложения, чем упрощает архитектуру ЭВМ. Дополнительный код отрицательного числа получается инвертированием двоичного числа и прибавлением к нему единицы.

При записи числа в дополнительном коде, старший разряд является знаковым. Если его значение равно 0, то в остальных разрядах записано положительное двоичное число, совпадающее с прямым кодом. Если же знаковый разряд равен 1, то в остальных разрядах записано отрицательное двоичное число, преобразованное в дополнительный код. Для получения значения отрицательного числа все разряды, кроме знакового, инвертируются, а к результату добавляется единица. Обратное преобразование, то есть перевод из дополнительного кода в прямой, осуществляется аналогично.

Двоичное 8-разрядное число может представлять любое целое в диапазоне от –128 до +127. Если старший разряд равен нулю, то наибольшее целое число, которое может быть записано в оставшихся 7 разрядах равно 2^{7-1}.

В табл. 5 приведены примеры представления чисел в прямом и обратном коде.

Таблица 5

Примеры представления чисел в прямом и обратном коде

Десятичный код	8-разрядный двоичный код	
	прямой код	дополнительный код
0	00000000	00000000
1	00000001	00000001
-1	10000001	11111111
-10	10001010	11110110

Модифицированные обратный и дополнительный коды

При переполнении разрядной сетки, происходит перенос единицы в знаковый разряд. В этом случае положительное число, получившееся в результате арифметической операции, может восприниматься как отрицательное, так как в знаковом разряде появляется 1.

В модифицированном обратном коде под знак числа отводится не один, а два разряда. Соответственно знаковые разряды будут 00 – для положительных чисел и 11 – для отрицательных чисел.

Любая другая комбинация («01» или «10»), получившаяся в знаковых разрядах в ходе арифметических операций служит признаком переполнения разрядной сетки. Сложение чисел в модифицированном обратном коде ничем не отличается от сложения в обычном обратном коде.

Модифицированный дополнительный код также рассматривает два знаковых разряда, а во всем остальном ничем не отличается от обычного дополнительного кода.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Ознакомиться с основными теоретическими положениями.
2. Получить у преподавателя вариант задания (см. приложение).
3. Выбрать наиболее рациональный метод перевода чисел для каждого задания.
4. Перевести числа из одной системы счисления в другую с помощью выбранного метода.

4. СОДЕРЖАНИЕ ОТЧЕТА

1. Цель работы.
2. Задание.
3. Перевод чисел со всеми промежуточными расчетами.
4. Выводы по работе.

5. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Чем отличаются позиционные системы счисления от аддитивных? Приведите примеры.
2. Чем характеризуется позиционная система счисления?
3. Какие системы счисления относятся к нетрадиционным? Приведите примеры.
4. Каким образом осуществляется перевод по универсальному алгоритму?
5. Как можно перевести числа из любой системы счисления в десятичную?
6. Назовите недостатки представления чисел в двойном коде.
7. Каким образом получается дополнительный код двоичного числа?
8. Что такое экспоненциальная форма записи числа?
9. Что подразумевается под понятиями: машинное слово, полуслово и двойное слово

ПРИЛОЖЕНИЕ

Варианты заданий

Вариант 1

1. Перевести в 10-ную систему счисления:
1100001.11_2, 3402.1_5, 346.7_8, 5212_7, $184.B_{16}$
2. Перевести в 2-ную систему счисления:
627_{10}, 20341_8, $A1DF4_{16}$
3. Перевести в 8-ную систему счисления:
532_{10}, 1010110111_2, $A4DC8_{16}$
4. Перевести в 16-ную систему счисления:
430_{10}, 362143_8, 1000111101010_2
5. Перевести в 2-ную систему счисления:
87.85_{10}, 230.3_5
6. Перевести в 8-ную систему счисления:
230.3_4
7. Выполнить следующие действия:
$11010101_2 + 1110_2$
$11011011_2 – 110101110_2$

Вариант 2

1. Перевести в 10-ную систему счисления:
1011001.11_2, 214.41_5, 761.5_8, 6212_7, $1AC4.5_{16}$
2. Перевести в 2-ную систему счисления:
587_{10}, 7415_8, $D8F5A_{16}$
3. Перевести в 8-ную систему счисления:
608_{10}, 1010111001_2, $ABCDE_{16}$
4. Перевести в 16-ную систему счисления:
346_{10}, 360721_8, 1011001010101_2
5. Перевести в 3-ную систему счисления:
84.55_{10}, 103.2_4
6. Перевести в 7-ную систему счисления:
240.1_5
7. Выполнить следующие действия:
$111011101_2 + 101110_2$
$110011011_2 – 1100001110_2$

Продолжение приложения

Вариант 3

1. Перевести в 10-ную систему счисления:
10000101.01_2, 3131.2_5, 274.2_8, 5624_7, $13B.A_{16}$
2. Перевести в 2-ную систему счисления:
593_{10}, 65470_8, $A3F26_{16}$
3. Перевести в 8-ную систему счисления:
833_{10}, 1010011001_2, $E8A42_{16}$
4. Перевести в 16-ную систему счисления:
820_{10}, 521647_8, 1001100100100_2
5. Перевести в 2-ную систему счисления:
39.55_{10}, 160.4_7
6. Перевести в 8-ную систему счисления:
121.1_3
7. Выполнить следующие действия:
$1011101_2 + 11101101_2$
$11010101_2 – 1110_2$

Вариант 4

1. Перевести в 10-ную систему счисления:
1000110.01_2, 3442.1_5, 705.5_8, 3540_7, $14A.F_{16}$
2. Перевести в 2-ную систему счисления:
327_{10}, 67424_8, $CAF90_{16}$
3. Перевести в 8-ную систему счисления:
642_{10}, 10111111101_2, $FA479_{16}$
4. Перевести в 16-ную систему счисления:
716_{10}, 565137_8, 1000101111110_2
5. Перевести в 2-ную систему счисления:
81.35_{10}, 203.2_4
6. Перевести в 5-ную систему счисления:
220.1_3
7. Выполнить следующие действия:
$11011101_2 + 1010110_2$
$110011_2 – 1001110_2$

Продолжение приложения

Вариант 5

1. Перевести в 10-ную систему счисления:
1010100.11_2, 3400.3_5, 360.4_8, 3701_8, $14B.F_{16}$
2. Перевести в 2-ную систему счисления:
347_{10}, 65403_8, $A1F94_{16}$
3. Перевести в 8-ную систему счисления:
820_{10}, 111111101101_2, 61370_{16}
4. Перевести в 16-ную систему счисления:
628_{10}, 521347_8, 10110000000011_2
5. Перевести в 3-ную систему счисления:
62.75_{10}, 130.4_5
6. Перевести в 8-ную систему счисления:
323.2_4
7. Выполнить следующие действия:
$1110111_2 + 1110_2$
$1100110011_2 – 11001110_2$

Вариант 6

1. Перевести в 10-ную систему счисления:
1000001.101_2, 2402.3_5, 127.6_8, 6541_7, $AC5.F_{16}$
2. Перевести в 2-ную систему счисления:
757_{10}, 47032_8, $BCD41_{16}$
3. Перевести в 8-ную систему счисления:
289_{10}, 1111100000111_2, $A4B8C41_{16}$
4. Перевести в 16-ную систему счисления:
806_{10}, 457561_8, 11110000001000_2
5. Перевести в 6-ную систему счисления:
159.48_{10}, 452.3_8
6. Перевести в 3-ную систему счисления:
10101011.1_2
7. Выполнить следующие действия:
$11101101101_2 + 101001110_2$
$1100011_2 – 110000110_2$

Продолжение приложения

Вариант 7

1. Перевести в 10-ную систему счисления:
10001101.01_2, 3222.2_5, 752.4_8, 6115_7, $84AC.E_{16}$
2. Перевести в 2-ную систему счисления:
967_{10}, 345011_8, $CD8F4_{16}$
3. Перевести в 8-ную систему счисления:
286_{10}, 10010101001_2, $4DEC8_{16}$
4. Перевести в 16-ную систему счисления:
875_{10}, 572163_8, 1101010101010_2
5. Перевести в 2-ную систему счисления:
104.25_{10}, 411.4_6
6. Перевести в 7-ную систему счисления:
121.2_3
7. Выполнить следующие действия:
$11101_2 + 10110_2$
$11001011_2 – 1001110_2$

Вариант 8

1. Перевести в 10-ную систему счисления:
10011111.01_2, 2302.2_5, 701.5_8, 5622_7, $2AD.B_{16}$
2. Перевести в 2-ную систему счисления:
197_{10}, 60321_8, $ADF48_{16}$
3. Перевести в 8-ную систему счисления:
402_{10}, 100011011_2, $4DB87_{16}$
4. Перевести в 16-ную систему счисления:
1050_{10}, 321643_8, 10101101011110_2
5. Перевести в 3-ную систему счисления:
800.2_{10}, 543.2_7
6. Перевести в 9-ную систему счисления:
424.5_6
7. Выполнить следующие действия:
$111011100101_2 + 1011110_2$
$1100110110_2 – 110000111001_2$

Продолжение приложения

Вариант 9

1. Перевести в 10-ную систему счисления:
11011001.101_2, 1111.1_5, 3572.2_8, 61112_7, $20C.A1_{16}$
2. Перевести в 2-ную систему счисления:
275_{10}, 72143_8, $A51D4E_{16}$
3. Перевести в 8-ную систему счисления:
824_{10}, 1111000001011_2, $C8F13_{16}$
4. Перевести в 16-ную систему счисления:
432_{10}, 621443_8, 1000100000010_2
5. Перевести в 2-ную систему счисления:
721.54_{10}, 4443.1_5
6. Перевести в 7-ную систему счисления:
1020.1_3
7. Выполнить следующие действия:
$11010110101_2 + 101110_2$
$110011011_2 – 1111011_2$

Вариант 10

1. Перевести в 10-ную систему счисления:
1000001.1101_2, 2442.3_5, 276.5_8, 1612_7, $4CD.5_{16}$
2. Перевести в 2-ную систему счисления:
147_{10}, 20571_8, $A18BF8_{16}$
3. Перевести в 8-ную систему счисления:
842_{10}, 111111000111_2, $DE52C_{16}$
4. Перевести в 16-ную систему счисления:
2081_{10}, 621142_8, 1011101111010_2
5. Перевести в 4-ную систему счисления:
104.25_{10}, 240.4_5
6. Перевести в 8-ную систему счисления:
432.4_5
7. Выполнить следующие действия:
$11111101_2 + 10001110_2$
$11001100011_2 – 101110_2$

Продолжение приложения

Вариант 11

1. Перевести в 10-ную систему счисления:
101111101.01_2, 3402.24_5, 3211.2_8, 5641_7, $1AD4.8_{16}$
2. Перевести в 2-ную систему счисления:
407_{10}, 34100_8, $E00DF_{16}$
3. Перевести в 8-ную систему счисления:
602_{10}, 1011010110001_2, $40D2C_{16}$
4. Перевести в 16-ную систему счисления:
654_{10}, 620103_8, 1010110101010_2
5. Перевести в 2-ную систему счисления:
79.24_{10}, 431.3_5
6. Перевести в 16-ную систему счисления:
214.1_5
7. Выполнить следующие действия:
$1101_2 + 1011010_2$
$11001111_2 – 1101110_2$

Вариант 12

1. Перевести в 10-ную систему счисления:
11001.101_2, 3342.2_5, 372.7_8, 45621_7, $A018.04_{16}$
2. Перевести в 2-ную систему счисления:
806_{10}, 10741_8, $1A0F4_{16}$
3. Перевести в 8-ную систему счисления:
627_{10}, 1010000111_2, $4FD08_{16}$
4. Перевести в 16-ную систему счисления:
860_{10}, 621743_8, 1010110101010_2
5. Перевести в 3-ную систему счисления:
72.55_{10}, 241.62_7
6. Перевести в 6-ную систему счисления:
202.12_3
7. Выполнить следующие действия:
$1011101101_2 + 10110010_2$
$1010011011_2 – 110000_2$

Продолжение приложения

Вариант 13

1. Перевести в 10-ную систему счисления:
1010001.01_2, 2024.2_5, 366.1_8, 2562_7, $1C4.E_{16}$
2. Перевести в 2-ную систему счисления:
278_{10}, 27301_8, $1DAF4_{16}$
3. Перевести в 8-ную систему счисления:
822_{10}, 10111100111_2, $AF0C2_{16}$
4. Перевести в 16-ную систему счисления:
387_{10}, 621543_8, 1010110101010_2
5. Перевести в 2-ную систему счисления:
70.255_{10}, 240.32_5
6. Перевести в 16-ную систему счисления:
401.2_5
7. Выполнить следующие действия:
$1000001_2 + 101111_2$
$1001001_2 – 1001110_2$

Вариант 14

1. Перевести в 10-ную систему счисления:
1000100.11_2, 3102.3_5, 372.5_8, 5602_7, $10C4.2_{16}$
2. Перевести в 2-ную систему счисления:
524_{10}, 20061_8, $A17F01_{16}$
3. Перевести в 8-ную систему счисления:
1052_{10}, 10111010111_2, $A400C8_{16}$
4. Перевести в 16-ную систему счисления:
651_{10}, 621043_8, 10001010101010_2
5. Перевести в 2-ную систему счисления:
97.95_{10}, 224.4_5
6. Перевести в 9-ную систему счисления:
323.4_5
7. Выполнить следующие действия:
$1010111011101_2 + 1011101_2$
$100111000_2 – 11000110_2$

Лабораторная работа № 2
Работа с файлами и директориями в операционной системе MS DOS

1. ЦЕЛЬ РАБОТЫ

Целью работы является изучение функционирования операционной системы MS DOS и приобретение практических навыков работы в ней с файлами и директориями при помощи основных команд.

2. ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Основные понятия

Назначение операционной системы

При работе пользователя на компьютере часто возникает необходимость выполнить операции с прикладной программой в целом, организовать работу внешних устройств, проверить работу различных блоков, скопировать информацию и т.п. Из всего многообразия подобных операций выделяют типовые и реализуют их с помощью специализированных программ, принимаемых в качестве стандартных средств и поставляемых вместе с аппаратной частью.

Программы, организующие работу устройств и не связанные со спецификой решаемой задачи, входят в состав комплекса программ, называемого операционной системой (ОС).

ОС *– совокупность программных средств, обеспечивающая управление аппаратной частью компьютера и прикладными программами, а также их взаимодействие между собой и пользователем.*

ОС образует автономную среду, не связанную ни с одним из языков программирования. Любая же прикладная программа связана с ОС и может эксплуатироваться только на тех компьютерах, где имеется аналогичная системная среда. Для работы с ОС необходимо овладеть языком этой среды – совокупностью команд, структура которых определяется синтаксисом этого языка.

ОС выполняет следующие функции:

– управление работой каждого блока компьютера и их взаимодействием;

– управление выполнением программ;

– организацию хранения информации во внешней памяти;

– взаимодействие пользователя с компьютером, т.е. поддержка интерфейса пользователя.

Обычно ОС хранится на жестком диске или на специальном гибком диске, называемом системным. При включении компьютера ОС автоматически загружается с диска в оперативную память и занимает в ней определенное место. ОС создается не для отдельной модели компьютера, а для серии компьютеров.

DOS (ДОС – дисковая ОС) – это имя, используемое фирмой IBM для обозначения основной ОС, которая работает на персональных компьютерах (ПК) семейства IBM PC. DOS была создана для IBM фирмой Microsoft, одной из ведущих фирм по производству программного обеспечения для ПК. Фирма Microsoft также создала версии DOS для компьютеров других семейств. Эти другие версии обычно называют MS DOS (сокращение от Microsoft DOS).

Понятие файла

В основе любой ОС лежит принцип организации работы внешнего устройства для хранения информации. Несмотря на то, что внешняя память может быть реализована на разных материальных носителях, их объединяет принятый в ОС принцип организации хранения логически связанных наборов информации в виде так называемых файлов.

Файл *– логически связанная совокупность данных, для размещения которой во внешней памяти выделяется именованная область.*

Файл служит учетной единицей информации в ОС. Любые действия с информацией в MS DOS осуществляются над файлами: запись на диск, вывод на экран, ввод с клавиатуры, печать и прочее.

В файлах могут храниться разнообразные виды и формы представления информации: тексты, рисунки, чертежи, числа, программы, таблицы и т. п. Особенности конкретных файлов определяются *форматом*, под которым понимается элемент язы-

ка, в символическом виде описывающий представление информации в файле.

Для характеристики файла используются следующие параметры:

– полное имя файла;
– объем файла в байтах;
– дата создания файла;
– время создания файла;
– специальные атрибуты файла: R – только для чтения, H – скрытый файл, S – системный файл, A – архивированный файл.

С понятием файла в MS DOS тесно связано понятие логического диска. Логический диск создается и управляется специальной программой (драйвером). Он имеет уникальное имя, например, C, D, E, F. Логический диск может быть организован на жестком, гибком, лазерном дисках, в оперативной памяти (электронный диск) и т.п. На одном физическом диске может быть создано несколько логических дисков. В дальнейшем изложении под *диском* будем понимать *логический диск*.

Способы обращения к файлу

К файлу можно обращаться с помощью *имени* или с помощью *полного имени*. Рассмотрим эти варианты.

Правило образования имени. Имя файла уникально и служит для отличия одного файла от другого. *Имя файла* в MS DOS образуется не более чем из восьми символов, причем используются *только буквы латинского алфавита и цифры*, а первым символом обязательно должна быть буква.

В качестве имени файла можно использовать символьное имя устройства:

– PRN или LPT1 (2, 3) – принтер или любое устройство, подключенное к параллельному порту;
– CON – консоль (клавиатура при вводе и дисплей при выводе);
– COM1 (2, 3, 4) – какое-либо внешнее устройство, подключенное к последовательному порту;
– NUL – фиктивное устройство; вывод в файл NUL никуда не направляется, а просто уничтожается.

Правило образования полного имени. *Полное имя* более подробно характеризует файл и образуется из имени файла и типа (расширения), разделенных точкой.

Тип файла служит для характеристики хранящейся в файле информации и образуется не более чем из трех символов, причем используются (как и при образовании имени) только буквы латинского алфавита.

Пример. PRIMER.PAS – файл PRIMER для хранения программ на Паскале; STRAN.TXT – файл STRAN для хранения текста; COPY.COM – файл COPY, содержащий программу ОС по копированию файлов.

При работе на ПК установлен ряд соглашений по заданию типа файла. Часть таких соглашений приведена в таблице.

Соглашения по типу файлов

Тип	Назначение
.ARJ	Архивный файл
.BAK	Копия файла, создаваемая при перезаписи файла
.BAT	Командный файл
.COM	Командный системный файл, исполняемый файл
.DAT	Файл данных
.EXE	Исполняемый файл
.HLP	Файл для справочной информации
.LIB	Библиотека программ
.PIC	Данные выводимого на экран изображения
.PRN	Листинг (распечатка программы)
.SYS	Файлы, расширяющие возможности ОС (драйверы)
.TXT	Текстовый файл

Часто возникает ситуация, когда надо работать не с одним файлом, а с группой файлов. Например, необходимо выполнить операции: копирование группы файлов с одного диска на другой; удаление группы файлов; перемещение группы файлов на другой диск; поиск группы файлов заданного типа и т. п.

Такие операции легко выполнять, пользуясь при формировании имен и типов файлов *шаблоном*.

Шаблон имени файла – специальная форма, в которой в полях имени и типа файла используются символы **'*'** и **'?'**.

Символ **'*'** служит для замены любой последовательности символов. В шаблоне может быть использовано в поле имени и типа по одному символу **'*'.**

Пример. Задав имя **'*.**TXT', вы обратитесь ко всем текстовым файлам. Задав имя 'SD***.***', вы обратитесь ко всем файлам, имя которых начинается с букв SD.

Символ **'?'** служит для замены одного символа на месте, где стоит вопросительный знак. В шаблоне может быть использовано несколько таких символов.

Пример. Имя 'RT**??**.PAS' позволит обратиться ко всем файлам типа PAS, имя которых состоит из четырех символов, причем первые два символа обязательно RT, а третий и четвертый – любые.

Характеристика MS DOS

Организация доступа к файлу

Способ хранения файлов на диске и организацию доступа к ним можно сравнить соответственно с организацией хранения книг в библиотеке и процедурой поиска нужной книги по ее шифру из каталога.

***Доступ** – процедура установления связи с памятью и размещенным в ней файлом для записи и чтения данных.*

***Директория (каталог)** – справочник (список) файлов с указанием месторасположения на диске.*

Различают два состояния директории – активное (текущее) и пассивное. MS DOS помнит текущую директорию на каждом логическом диске.

***Текущая директория** – это директория, в которой работа пользователя производится в текущее машинное время.*

***Пассивная директория** – это директория, с которой в данный момент времени не имеется связи.*

В ОС MS DOS принята *иерархическая структура организации директорий (каталогов)*. На каждом диске всегда имеется главная (корневая) директория. Она находится на нулевом (высшем) уровне иерархии и обозначается символом '\'. Корневая директория создается при форматировании

(инициализации, разметке) диска и не может быть удалена средствами MS DOS. В корневую директорию могут входить другие директории (директории первого уровня) и файлы, которые создаются и удаляются командами ОС. В свою очередь, в директории первого уровня могут входить поддиректории (директории второго уровня) и т. д.

Родительская директория *– это директория, имеющая поддиректории.*

Поддиректория *– это директория, которая входит в другую директорию.*

Как правило, употребляют термин "директория" ("каталог"), подразумевая поддиректорию (подкаталог) или родительскую директорию (родительский каталог) в зависимости от контекста.

Правила наименования директорий такие же, как и правила наименования файлов. Для формального отличия от файлов обычно директориям присваивают только имена.

Доступ к содержимому файла организован из главной директории через цепочку соподчиненных директорий *n*-го уровня. В директориях любого уровня могут храниться записи как о файлах, так и о директориях нижнего уровня.

Описанный принцип организации доступа к файлу через директорию является основой файловой системы.

Файловая система *– часть ОС, управляющая размещением и доступом к файлам и директориям на диске.*

С понятием файловой системы связано понятие *файловой структуры* диска, под которой понимают, как размещаются на диске директории, файлы, ОС, а также какие для них выделены объемы памяти.

Доступ к файлу можно организовать следующим образом:

– если имя файла зарегистрировано в текущей директории, то достаточно указать только его имя (полное имя);

– если имя файла зарегистрировано в пассивной директории, то, находясь в текущей директории, нужно указать еще и путь.

Путь *– цепочка соподчиненных директорий, которую необходимо пройти по иерархической структуре к директории, где зарегистрирован искомый файл.*

При задании пути имена директорий записываются в порядке следования и отделяются друг от друга символом '\'.

Взаимодействие пользователя с ОС осуществляется с помощью командной строки, индицируемой на экране дисплея. В начале командной строки всегда имеется *приглашение*, которое заканчивается символом '>'. В приглашении может быть отражено: имя текущего диска, имя текущей директории, символы-разделители, текущее время и дата, путь.

Приглашение ОС *– индикация на экране дисплея информации, означающей готовность ОС к вводу команд пользователя.*

Возможны три варианта организации пути доступа к файлу в зависимости от места его расположения. Файл находится в текущей директории (путь отсутствует). При организации доступа к файлу достаточно указать его полное имя.

1. Файл находится в пассивной директории одного из нижних уровней, подчиненного текущей директории. При организации доступа к файлу необходимо указать путь, в котором перечислены имена всех директорий нижнего уровня, лежащих на этом пути (включая директорию, в которой находится данный файл).

2. Файл находится в пассивной директории на другой ветке по отношению к местонахождению текущей директории. Здесь необходимо указать путь, начиная с корневой директории, то есть с символа '\'. Горизонтальные переходы из директории в директорию недопустимы.

Модульная структура MS DOS
Модули ОС MS DOS

Понятие модуля широко используется применительно как к аппаратной, так и к программной части компьютера.

Модуль *– унифицированная самостоятельная функциональная часть системы, имеющая законченное оформление и средства сопряжения с другими функциональными узлами и модулями.*

Структуру ОС MS DOS образуют следующие модули:

1) BIOS (Basic Input/Output System) – базовая система ввода-вывода;

2) модуль расширения –EM BIOS (Extension Module BIOS) в виде файла с именем IO.SYS;

3) системный загрузчик (SB – System Bootstrap);

4) внешние драйверы – файлы с расширением .COM, .EXE, .SYS;

5) базовый модуль (BM – Basic Module) в виде файла с именем MSDOS.SYS;

6) командный процессор или интерпретатор команд (CI – Command Interpreter) в виде файла с именем COMMAND.COM;

7) внешние команды, утилиты – файлы с расширением .COM, .EXE, .SYS;

8) инструментальные средства DOS: система программирования MS DOS QBASIC; текстовый редактор MS DOS EDITOR; отладчик DEBUG для тестирования и отладки исполняемых файлов.

Первые четыре модуля составляют *машинозависимую* часть ОС, а последние четыре модуля – *машинонезависимую* часть ОС.

Система прерываний

Основным механизмом функционирования MS DOS является система прерываний.

Прерывания *– это процедуры, которые компьютер вызывает для выполнения определенной задачи.*

Различают аппаратные, логические и программные прерывания.

Аппаратные прерывания инициируются аппаратурой, например, сигналом от принтера, нажатием клавиши на клавиатуре, сигналом от таймера и т. д.

Логические прерывания возникают при нестандартных ситуациях в работе микропроцессора, например, деление на ноль, переполнение регистров и т. д.

Программные прерывания инициируются программами, т. е. возникают, когда программа ждет получения сервиса со стороны

другой программы, например, доступ к определенным аппаратным средствам.

Функции и назначение базовой системы ввода-вывода

BIOS находится в постоянной памяти, которая входит в комплект поставки ПК. Тип ОС может изменяться, а BIOS остается постоянным.

BIOS устанавливает связь между техническими средствами и стандартизированным программным обеспечением (ПО), а именно ОС. BIOS содержит специальные программы (драйверы) по управлению работой стандартными внешними устройствами. Назначение BIOS состоит в выполнении наиболее простых и универсальных функций ОС, связанных с вводом-выводом. BIOS содержит также: тест функционирования ПК, проверяющий работу памяти и устройств после включения питания, программу загрузки ОС. BIOS – общая (неизменяемая) часть всех ОС для данной модели ПК. Системный загрузчик считывает в оперативную память модуль расширения BIOS и модуль обработки прерываний.

Функции и назначение модуля расширения BIOS

Модуль расширения BIOS придает гибкость ОС, позволяет управлять с ее помощью набором аппаратных средств ПК. Этот модуль можно модифицировать с учетом необходимых нужд конкретной версии MS DOS.

Модуль позволяет перекрыть функции BIOS в постоянном запоминающем устройстве и обеспечивает возможность подключения дополнительных драйверов (программ обслуживания внешних устройств). Основная функция этого модуля – это увеличение возможностей BIOS.

Функции и назначение базового модуля

Основная функция базового модуля – управление ресурсами ПК, файловой системой, работой программ при помощи системы прерываний. Функциями базового модуля на этапе загрузки являются: считывание в память и запуск командного процессора, инициализация векторов прерываний верхнего уровня.

Функции и назначение командного процессора

Командный процессор на диске может занимать любое место и, по сути, представляет собой выполнимую программу. Командный процессор выполняет в ПК следующие функции:

– прием и разбор команд с клавиатуры или из командного файла;

– выполнение команд MS DOS, находящихся внутри файла COMMAND.COM;

– загрузка и выполнение внешних команд MS DOS (утилит) и прикладных программ, хранящихся в виде файлов с расширением COM и EXE.

Программы с расширением COM не требуют настройки адресов после их загрузки в оперативную память, а с расширением EXE – настраиваются по месту размещения (для них задаются соответствующие адреса сегментов).

При загрузке в оперативную память командный процессор распадается на две части:

– резидентную, постоянно размещаемую в оперативной памяти;

– нерезидентную (транзитную), периодически изменяемую путем передачи данных между оперативной памятью и диском.

Резидентная часть содержит подпрограммы стандартной обработки прерываний. Здесь же находятся: программа подзагрузки нерезидентной части в оперативную память и подпрограмма, обрабатывающая файл AUTOEXEC.BAT при запуске ПК.

Назначение загрузчика

Загрузчик BOOT RECORD (модуль начальной загрузки) всегда размещается на диске в нулевом секторе. Основное назначение загрузчика – поиск и перезапись (загрузка) с диска в оперативную память двух файлов IO.SYS и MSDOS.SYS, а также запуск модуля расширения базовой системы ввода-вывода.

Утилиты, внешние команды и драйверы

Утилиты – обслуживающие программы, поставляемые вместе с ОС в виде файлов и предоставляющие пользователю сервисные услуги (форматирование дискет, проверку дисков и т. д.).

Внешней командой принято считать программу, выдающую пользователю ряд простых запросов или выполняющуюся автоматически без специально организованного интерфейса пользователя. MS DOS имеет определенный перечень внешних команд.

Внешние драйверы – программы, дополняющие систему ввода-вывода и обеспечивающие обслуживание новых устройств или нестандартное использование имеющихся устройств. Драйверы загружаются в оперативное запоминающее устройство при загрузке ОС, а их имена указываются в файле конфигурации CONFIG.SYS.

Загрузка MS DOS в оперативную память с диска

ОС хранится во внешней памяти на жестком или (реже) на гибком диске. Для работы ПК необходимо, чтобы основные модули ОС находились в оперативной памяти. Поэтому после включения ПК организована автоматическая перезапись (загрузка) ОС с диска в оперативную память.

Запуск ПК и подготовка ОС к работе включает следующие шаги:

1. При включении ПК управление передается базовой системе ввода-вывода BIOS. BIOS выполняет тестирование памяти, проверку состояния аппаратуры и инициализирует устройства. Параметры конфигурации ПК извлекаются из так называемой энергонезависимой памяти. При нажатии клавиши <Del> перед инициализацией устройств можно передать управление программе изменения параметров конфигурации.

2. Управление конфигурацией ПК (задание параметров жесткого диска, указание системного диска, задание пароля) выполняется с помощью программы Setup.

3. Вызов загрузчика (BOOT RECORD) и загрузка с его помощью в оперативную память модуля расширения IO.SYS и базового модуля MSDOS.SYS.

4. Загрузка командного процессора COMMAND.COM.

5. Обработка файла конфигурации CONFIG.SYS, содержащего команды подключения необходимых драйверов.

6. Обработка командного файла AUTOEXEC.BAT. С помощью этого файла можно произвести настройку параметров ОС. Например, создать виртуальный диск, обеспечить смену режимов печати, загрузить вспомогательные программы и т.д.

Технология работы в MS DOS

Общие сведения о командах

Работа в ОС MS DOS организуется *командами*. Они вызывают определенное действие: организуют передачу информации, вырабатывают необходимый управляющий сигнал, подключают внешнее устройство для организации процесса ввода-вывода информации и т.д.

Команда технически реализована программой в машинных кодах и хранится либо в файле на диске (внешняя команда), либо входит в состав командного процессора COMMAND.COM (внутренняя команда). По порядку запуска внутренние и внешние команды не различаются. При запуске внешних команд необходимо удостовериться, что файлы, в которых они находятся, существуют на диске и находятся на "видимой" (компьютеру) директории. Как и любая другая программа, команда имеет уникальное имя и всегда имеет тип COM или EXE.

Ввод команды осуществляется в *командной строке* в соответствии с определенными правилами, заданными в виде *формата*.

Командная строка – строка экрана дисплея, начинающаяся с приглашения ОС. Командная строка состоит из информации подсказки, указателя ввода и курсора. Обычная информация подсказки указывает на диск и директорию, где в это время производится работа.

Формат команды – правило формирования команды пользователем с клавиатуры.

При формировании команды в соответствии с установленным форматом необходимо соблюдать следующие правила:

1) формат команды состоит из имени команды (латинскими буквами без указания типа) и отделенных от него одним пробелом параметров, уточняющих действие команды;

2) в большинстве случаев параметры между собой пробелом не разделяются, а в качестве разделителя часто используется символ '/';

3) параметрами могут быть: имя логического диска, путь, имя файла, тип файла, латинские буквы, символы, цифры;

4) параметры в формате могут и отсутствовать, что указывается с помощью квадратных скобок '[' и ']'.

Обобщенный формат команды можно представить в следующем виде: <имя команды> [<параметры>].

Пример: C:\>DIR D:\USER*.TXT/P

Здесь:

C:\> – приглашение ОС MS DOS;

DIR – имя команды;

D:\USER*.TXT/P – параметры.

Эта команда вызывает с помощью параметра D:\USER*.TXT/P на экран записи обо всех файлах типа 'TXT' из директории первого уровня 'USER' логического диска 'D'. Вызов записей производится постранично, на что указывает параметр '/P'.

Процедура ввода команды состоит в следующем:

1) в соответствии с форматом в командной строке набирают имя команды и необходимые параметры;

2) нажимают клавишу ввода, что служит сигналом начала анализа структуры набранной команды. При отсутствии ошибок в формате команды она будет выполнена, иначе на экран выдается сообщение: *Bad command or filename* (Неверная команда или имя файла);

3) при невыполнении команды просматривают вводимую конструкцию и вновь вводят ее, но уже в откорректированном варианте.

Порядок действий при выполнении команды MS DOS

После ввода команды с клавиатуры MS DOS выполняет следующие действия:

1. MS DOS анализирует первое слово командной строки (последовательность символов до первого пробела) с целью выяснить – задано ли просто имя, неполный адрес или точный адрес. Основной ключ к анализу – наличие символов ':' и '\'.

2. Если задано просто имя, MS DOS ищет его сначала в файле COMMAND.COM, затем в текущей директории, затем в директориях, перечисленных в команде PATH, записанной в файле автозапуска AUTOEXEC.BAT. Если расширение в имени

опущено, поиск ведется по собственному имени с подстановкой расширения в следующем порядке: COM, EXE, BAT.

3. Если задан неполный адрес, MS DOS ищет программу либо на текущем диске, либо начиная с текущей директории. Например, если указано: \SIMP\REM.EXE, то поиск файла REM.EXE ведется в каталоге SIMP текущего диска. Если указано: C\SIMP\REM.EXE, то MS DOS ищет файл REM.EXE в поддиректории 'C' текущей директории.

4. Если задан точный адрес, то MS DOS просто следует по указанному пути, не обращая внимания ни на текущий каталог, ни на директории, перечисленные в команде PATH.

5. Найдя программу, MS DOS загружает ее и передает ей в качестве параметров все, что набрано в командной строке.

6. После завершения программы на экран вновь выводится приглашение MS DOS.

7. Если программа не найдена, на дисплей поступает сообщение: *Bad command or filename* (имя команды или файла указано неверно), и выдается приглашение MS DOS.

Команды MS DOS общего назначения

По мере необходимости пользователь может использовать следующие команды, называемые командами общего назначения:

1. CLS – очистка экрана от выведенной до этого информации.

2. ECHO <сообщение> – печать сообщения на экране. Команды ECHO OFF и ECHO ON соответственно запрещают и разрешают печать сообщения других команд.

3. DATE – вывод на экран или установка текущей даты в формате "мм-дд-гг".

4. TIME – вывод на экран или установка системного времени в формате "чч:мм".

5. PROMPT $<тип информации>$<вид указателя> – определение системной подсказки. Тип информации задается символами: D – текущая дата, P – текущий диск и путь, N – только текущий диск, T – текущее время. Вид указателя задается символами: G (на экране появится символ '>') или L (на экране появится символ '<'). Обычно эту команду используют в следующем виде: PROMPT PG.

6. VER – вывод на экран номера версии ОС на этом ПК.

Основные команды для работы с директориями

Команда DIR – просмотр директории

Работа на ПК, как правило, начинается с просмотра директории, например, чтобы убедиться в том, что нужный вам файл существует. Часто необходимо просмотреть содержимое пассивной директории.

В зависимости от параметров, допустимых в структуре команды, можно просмотреть записи директории в стандартной или усеченной форме с выводом только полных имен файлов, а также при большом содержании директории выводить ее постранично.

Формат команды:

DIR [Имя дисковода:][Путь\][Имя файла][Параметры]

Если имя дисковода и/или путь отсутствуют в команде, то подразумевается текущий дисковод и текущая директория. Параметры (ключи) задают порядок вывода списка файлов и директорий.

Назначение основных параметров (ключей):

/P – постраничный вывод содержимого директории на экран. Для продолжения вывода следует нажать любую клавишу;

/W – вывод только полных имен файлов и директорий;

/A – индикация содержимого директорий с атрибутами;

/O – задание порядка сортировки выводимых сведений.

Примеры:

C:\>DIR	Вывод содержимого корневой текущей директории на экран
C:\USER1>DIR *.BAK	Вывод на экран всех имен файлов типа BAK из текущей директории первого уровня USER1
C:\>DIR A:	Вывод на экран содержимого пассивного дисковода A
C:\B1>DIR B2	Вывод на экран содержимого пассивной директории второго уровня B2, находящейся в директории первого уровня B1
C:\USER1>DIR \B1\B2/P	Вывод на экран постранично содержимого пассивной директории B2.

	Эта директория находится в другой ветке иерархической структуры директории, чем текущая директория USER1
C:\B1>DIR /W	Вывод на экран записей текущей директории B1 в усеченном формате (только полные имена файлов и директорий)

Команда MD – создание директории

Новую директорию можно создать командой MD в текущей директории или, если указан путь, в пассивной директории.

Формат команды:

MD [Имя дисковода:][Путь\]Имя директории

Примеры:

C:\>MD USER1	Создание (в текущей корневой директории) директории первого уровня USER1
C:\>MD USER1\USER2	Создание (в директории первого уровня USER1) директории второго уровня USER2
C:\T1\T2>MD USER1\USER2	Создание директории второго уровня USER2, если ОС находится в другом каталоге второго уровня T2

Команда RD – уничтожение директории

Эта команда уничтожает только пустую директорию. Предварительно необходимо удалить из нее командой DEL все файлы, а затем командой DIR убедиться в том, что она пустая.

Формат команды:

RD [Имя дисковода:][Путь\]Имя директории

Примеры:

C:\>RD USER1	Удаление директории USER1 из корневой директории
C:\B1>RD \USER1\USER2	Удаление пассивной директории второго уровня USER2, если ОС находится в текущей директории

B1

Команда CD – переход в другую директорию

Иногда необходимо перейти в другую директорию и сделать ее текущей. В этом случае следует воспользоваться командой CD.

Формат команды:

CD [Имя дисковода:][Путь\]Имя директории

Для перехода в родительскую директорию достаточно вместо имени директории задать '**..**' (две точки). Для перехода в корневую директорию задают символ '\'.

Примеры:

C:\>CD USER1	Переход в директорию USER1 из корневой директории. После ввода команды приглашение примет вид: C:\USER1>
C:\B1\B2>CD \USER1	Переход из директории второго уровня B2 в директорию первого уровня USER1, находящуюся в другой ветви иерархической структуры. После ввода команды приглашение примет вид: C:\USER1>
A:\>CD C:\USER1	Переход из корневой директории диска A в директорию первого уровня диска C. После ввода команды приглашение примет вид: C:\USER1>
C:\M1\M2\M3>CD ..	Переход в родительскую директорию. После ввода команды приглашение примет вид: C:\M1\M2>
C:\F1\F2>CD \	Переход в корневую директорию. После ввода команды приглашение примет вид: C:\>

Основные команды для работы с файлами

Команда TYPE – просмотр текстового файла

Командой TYPE удобно пользоваться для просмотра содержимого текстового файла на экране дисплея или на принтере. После запуска команды текст (содержимое файла) выводится не-

прерывным потоком, причем скорость смены кадров с текстом на экране настолько велика, что прочесть его практически невозможно. Для приостановки вывода текста нажимают одновременно две клавиши: <CTRL> и <S>. Нажатие затем любой клавиши возобновит вывод текста.

Формат команды для вывода на экран:

TYPE [Имя дисковода:][Путь\]Полное имя файла

Формат команды для печати на принтере:

TYPE [Имя дисковода:][Путь\]Полное имя файла>PRN

Примеры:

C:\>TYPE ROK.TXT	Вывод на экран содержимого файла ROK.TXT, расположенного в корневой директории активного дисковода
C:\>TYPE A:\RED\LOT.TXT	Вывод на экран содержимого файла LOT.TXT, расположенного в директории RED первого уровня пассивного дисковода A
C:\>TYPE \B1\BOOK.TXT>PRN	Печать на принтере содержимого файла BOOK.TXT, расположенного в директории B1 первого уровня текущего дисковода C

Команда DEL – удаление файлов

Можно удалять как один файл, так и группу файлов, указывая для группы в шаблоне имени файла символы '*' или '?'.

Формат команды:

DEL [Имя дисковода:][Путь\]Полное имя файла>[/P]

Параметр [/P] служит для вывода на экран запроса на подтверждение удаления.

При вводе команды удаления всех файлов (DEL *.*) ОС задает вопрос *Are You sure (Y/N)? (Вы уверены?)*. Если вы не передумали, нажмите клавишу <Y>, в противном случае – клавишу <N>.

Примеры:

C:\>DEL TOST.TXT	Удаление файла TOST.TXT из корне-

	вой директории текущего дисковода C
C:\>DEL A:\AR\B.TXT	Удаление файла B.TXT из директории первого уровня AR пассивного дисковода A
C:\>DEL \A1\A2*.BAS	Удаление всех файлов типа BAS из директории второго уровня A2 текущего дисковода C
C:\F1>DEL F2*.*/P	Удаление с подтверждением всех файлов из директории второго уровня F2 текущего дисковода C, подчиненной текущей директории F1

Команда COPY – копирование файлов

Команда используется для создания копий существующих файлов, вывода содержимого файла на внешнее устройство, объединения содержимого нескольких файлов.

Команда COPY допускает выполнение функций над группой файлов, и тогда в шаблоне имени файла используются символы '*' или '?'.

Формат команды для копирования файлов:

COPY [Имя дисковода-источника:][Путь\]Полное имя файла-источника[Имя дисковода-приемника:][Путь\][Полное имя файла-приемника][N]

Обязательным параметром является только полное имя файла-источника. Если копируется файл (группа файлов) с тем же именем (именами), то достаточно указать только полное имя файла-источника. Имя дисковода и путь нужно указывать при работе с пассивным дисководом и директорией.

Примеры:

C:\>COPY ROK.PAS A:	Копирование файла ROK.PAS из корневой директории текущего дисковода C на диск A с тем же именем
C:\>COPY A1\A2\P.TXT \B1\B2\B3	Копирование файла P.TXT из директории A2 второго уровня в директорию третьего уровня B3 с тем же име-

	нем
C:\>COPY A:ROST.BAS	Копирование файла ROST.BAS с пассивного дисковода A в корневую директорию текущего дисковода C с тем же именем
C:\>COPY A:T.TXT A1\A2\S.TXT	Копирование файла T.TXT с диска пассивного дисковода в директорию второго уровня A2 текущего дисковода. Полученной копии файла присваивается новое имя S.TXT
C:\A1>COPY *.BAS B:/V	Копирование всех файлов типа BAS из текущей директории первого уровня A1 дисковода C на диск пассивного дисковода B с одновременным контролем процесса копирования

Формат команды объединения нескольких файлов:

COPY [Имя дисковода:][Путь\]Полное имя файла + [Имя дисковода:][Путь\]Полное имя файла + … [Имя дисковода:][Путь\]Полное имя файла-приемника

Имена объединяемых файлов перечисляются в команде COPY через знак '+'.

Имя результирующего файла записывается последним и отделяется от имен объединяемых файлов пробелом.

Содержимое результирующего файла представляет собой подсоединенное друг за другом содержимое исходных файлов в соответствии с порядком следования их имен.

Примеры:

C:\>COPY M1.TXT+ M2.TXT \K1\SUM.TXT	Объединение двух текстовых файлов M1.TXT и M2.TXT в файл SUM.TXT, который будет записан в директорию первого уровня K1
C:\>COPY T1.TXT+T2.TXT	Объединение двух текстовых файлов T1.TXT и T2.TXT. К со-

держимому файла T1.TXT добавляется содержимое файла T2.TXT, и результат объединения будет храниться в файле с именем T1.TXT

Форматы команд для обмена данными между внешним устройством и файлом, хранящимся на диске:

COPY Имя внешнего устройства (откуда) [Имя дисковода:][Путь\]Полное имя файла (куда)

COPY [Имя дисковода:][Путь\]Полное имя файла (откуда) Имя внешнего устройства (куда)

COPY Имя внешнего устройства (откуда) Имя внешнего устройства (куда)

Под внешним устройством здесь понимается любое устройство, кроме системного блока и дисководов. В ОС приняты соглашения относительно имен внешних устройств. К наиболее употребительным относятся:

- CON – клавиатура и дисплей (консоль);
- PRN или LPT1 – основной принтер.
- **Примеры:**

C:\>COPY T.TXT PRN	Печать содержимого текстового файла на принтере
C:\>COPY CON S.TXT	Заполнение файла S.TXT поступающими с клавиатуры символами
C:\>COPY CON PRN	Все символы, набираемые с клавиатуры, печатаются, минуя центральную часть компьютера, т.е. компьютер используется как пишущая машинка. Одновременно компьютер может обрабатывать информацию в соответствии с программой, где не требуется обращение к принтеру

Образец варианта задания

ВАРИАНТ № 1

Выполнить задания в ОС MS DOS

1. На диске С создать директорию ТЕСТ. В этой директории создать директорию КАТЕР, а в ней создать директорию КОРВЕТ. Внутри последней директории создать две поддиректории КОМ и РЕМАРКА.

2. Скопировать в поддиректорию КОМ все файлы с расширением TPU из директории C:\TP или TP7\BIN. Скопировать в поддиректорию РЕМАРКА файлы с расширением PAS, имеющие 7 и менее символов в своем имени и находящиеся в директории TP или TP7 или ее поддиректориях.

3. Соединить три файла в поддиректории РЕМАРКА в один файл с именем СТАКАН.TXT.

4. Переименовать файл с наименьшим размером в поддиректории КОМ в файл с именем СОРТ.TXT.

5. Удалить поддиректорию КОМ.

3. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое файл, характеристики файла?
2. Понятие имени файла и полного имени файла.
3. Каковы правила при задании имени файла в MS DOS?
4. Какие символы используются в шаблоне имени файла?
5. Доступ и три способа организации доступа к файлу.

Лабораторная работа № 3
Основы работы в текстовом редакторе Microsoft Word

ЦЕЛЬ РАБОТЫ

Целью работы является ознакомление студентов с основными операциями создания и обработки текстовых документов с помощью Microsoft Word.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Обработка текстов – один из наиболее распространенных видов работ, выполняемых на персональном компьютере. Существует большое количество текстовых редакторов, различающихся набором предоставляемых функций, легкостью освоения, операционными системами, в которых они могут работать, быстродействием, распространенностью и другими параметрами. В большинстве текстовых редакторов используются одни и те же принципы работы. Текстовый редактор **Microsoft Word** представляет документ, как совокупность символов, логически связанных между собой и образующих завершенную лексическую конструкцию. Текстовый редактор **Microsoft Word** входит в пакет **Microsoft Office.** Вид значка Microsoft Word показан на рисунке1.

Рис. 1. Значок **Microsoft Word**

Работа в Microsoft Word

Создание документа

Для того чтобы создать новый документ необходимо щелкнуть на кнопке **Microsoft Office**  , откроется меню, показанное на рисунке 2, выбрать команду создать, появится окно **Создание документа**, щелкнуть **Новый документ**, откроется окно программы Microsoft Word как показано на рисунке 3.

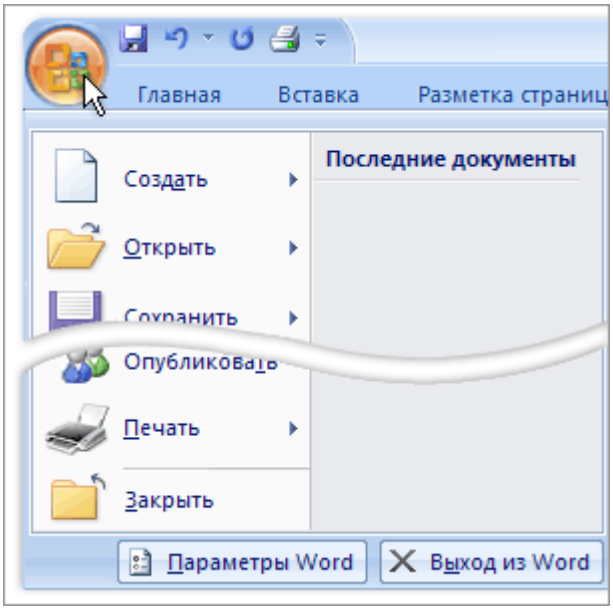


Рис. 2. Меню приложения

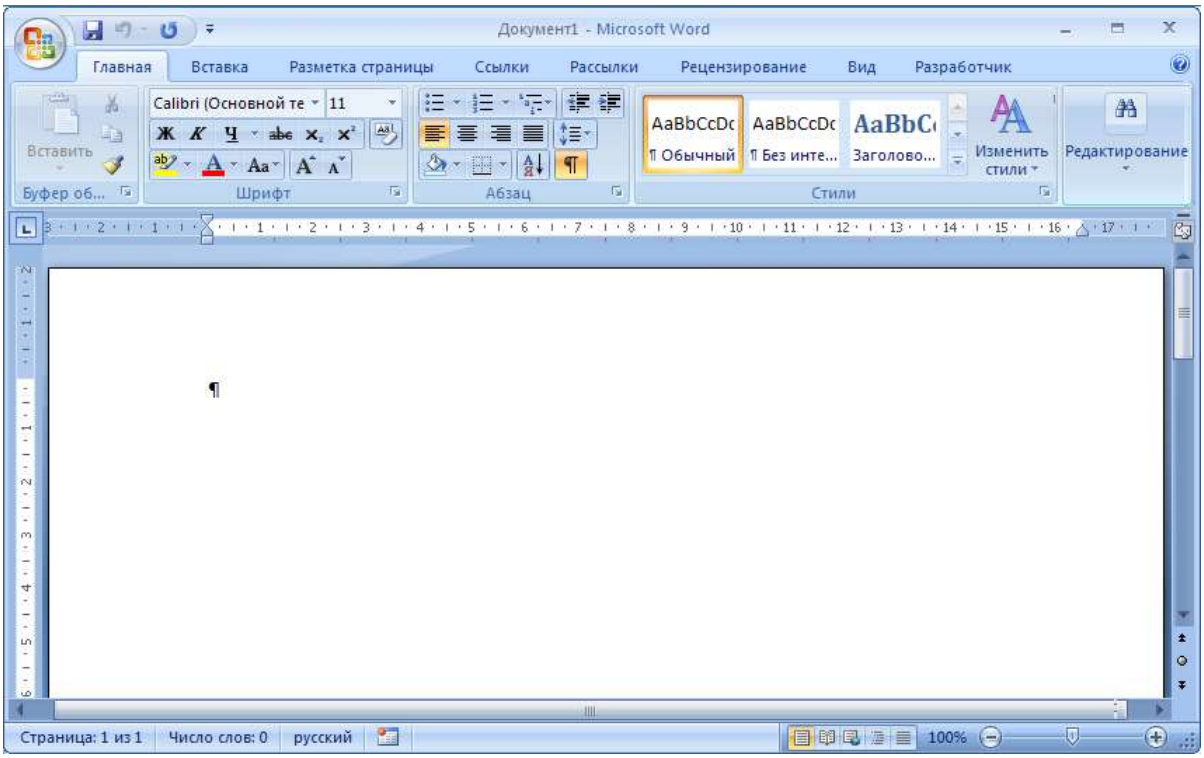


Рис. 3. Окно программы Microsoft Word

Многооконная организация Microsoft Word позволяет работать с несколькими документами, каждый из которых расположен в своем окне. Для перехода между окнами можно выбрать его имя на панели задач или выбрать во всплывающем меню по-

сле нажатия кнопки **Microsoft Office** . Для того, чтобы отобразить два документа одновременно необходимо открыть оба файла, на вкладке Вид в группе **Окно** щелкните **Просмотреть рядом** (рис. 4).

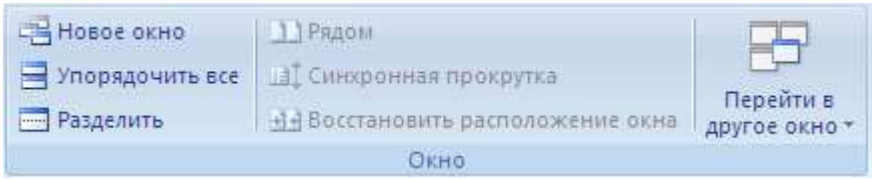
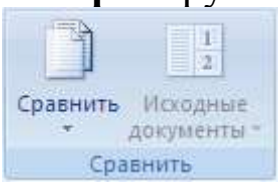


Рис. 4. Группа Окно

Чтобы прокручивать одновременно оба документа, на вкладке **Режим**, в группе **Окно** нужно нажать **Синхронная прокрутка**. Если кнопка **Синхронная прокрутка** не отображается, то на вкладке **Режим** выбрать **Окно**, а затем щелкнуть **Синхронная прокрутка**. Чтобы закрыть режим просмотра **Рядом**, на вкладке **Режим**, в группе **Окно** нажать кнопку **Рядом**. Если кнопка **Рядом** не отображается, то на вкладке **Режим** выбрать **Окно**, а затем **Рядом.**

Можно синхронизировать изменения в документе, попарно комбинируя друг с другом, если изменения производили несколько редакторов, для этого на вкладке **Обзор** в группе **Сравнение версий** выбрать команду **Сравнить** 

. Далее, после выбора команды **Объединение исправлений от нескольких авторов**, в группе **Исходный документ** нужно выбрать имя документа, в котором следует объединить исправления из нескольких источников.

Перемещение по документу

При вводе данных перемещаться по документу можно с помощью курсора ввода или указателя мыши. **Курсор ввода** представляет собой мигающую вертикальную полоску |. Он указывает

место, в которое будет вводиться текст. Для его перемещения используются клавиши управления курсором (табл. 1) или мышь. При перемещении курсора с помощью мыши достаточно установить курсор в нужную позицию и щелкнуть правой кнопкой мыши.

Таблица 1

Перемещение курсора с помощью клавиатуры

Клавиша	Перемещение
↑, ↓,	На одну строку вверх, вниз
←, →	На одну позицию влево, вправо
Ctrl+↑, Ctrl+↓	На один абзац вверх, вниз
Ctrl+←, Ctrl+→	На одно слово влево, вправо
PgUp, PgDn	На один экран вверх, вниз
End, Home	В конец строки, в начало строки
Ctrl+Home, Ctrl+End	В начало документа, в конец документа

Перемещаться по документу можно также используя полосы прокрутки (вертикальную и горизонтальную) (рис. 5).

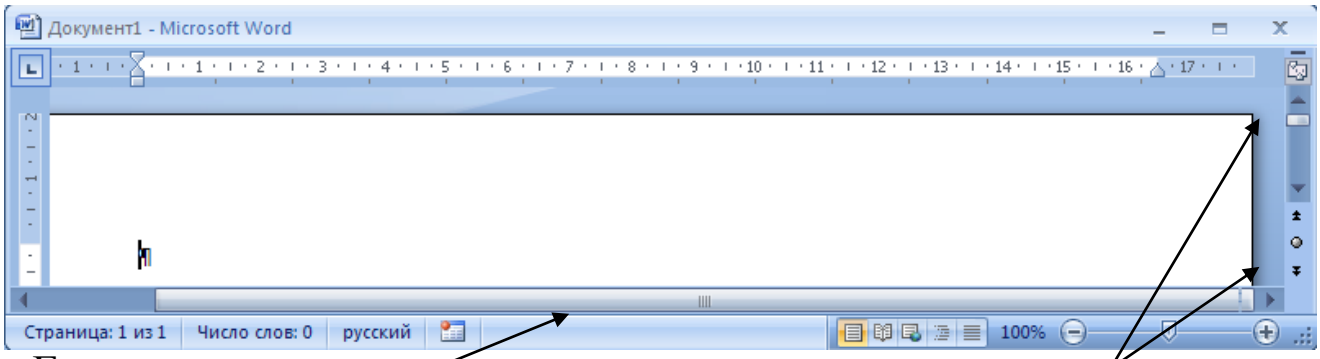


Рис. 5. Прокрутка по документу

Пользовательский интерфейс

Лента.

Лента разработана для облегчения доступа к командам и состоит из вкладок, связанных с определенными целями или объек-

тами. Каждая вкладка, в свою очередь, состоит из нескольких групп взаимосвязанных элементов управления (рис. 6).

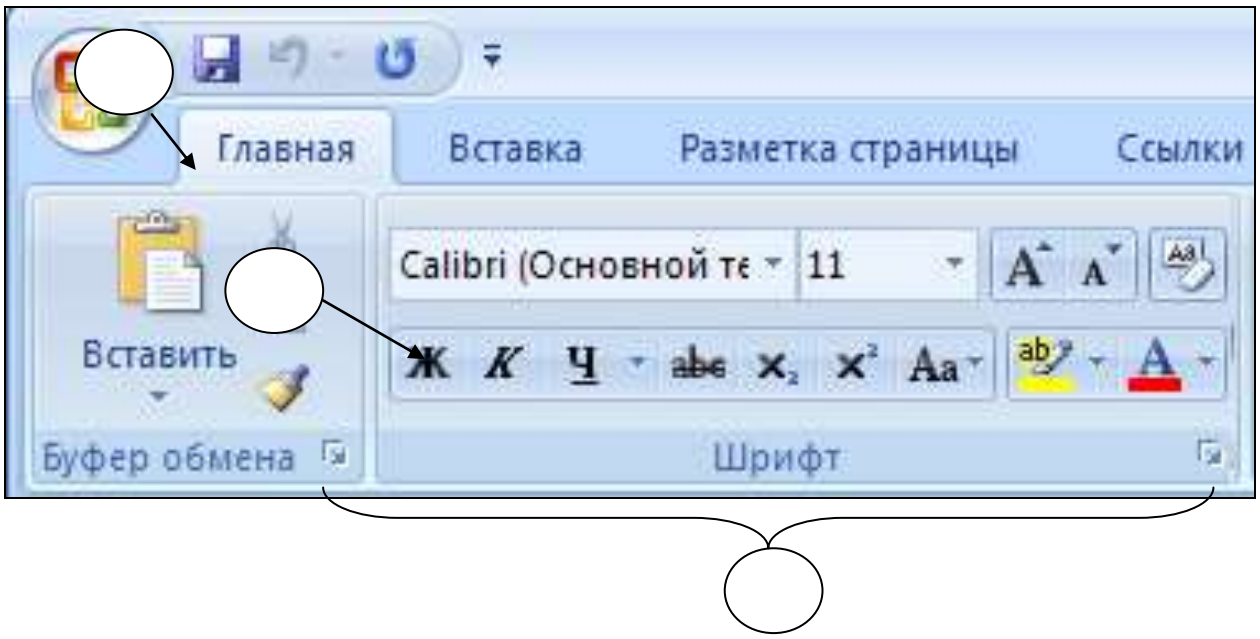


Рис. 6. Пользовательский интерфейс **Лента**
1 – Вкладки ориентированы на выполнение задач
2 – Группы на каждой вкладке разбивают задачу на ее составляющие
3 – Кнопки команд в каждой группе служат для выполнения команд или отображения меню команд

Кроме стандартного набора вкладок на ленте имеются вкладки еще двух типов, которые отображаются в интерфейсе в зависимости от выполняемой задачи.

Контекстные инструменты.

Контекстные инструменты позволяют работать с элементом, который выделен на странице, например, с таблицей, изображением или графическим объектом. Если щелкнуть такой элемент, относящийся к нему набор контекстных вкладок, выделенный цветом, появится рядом со стандартными вкладками (рис. 7). Для того чтобы контекстное меню отобразилось, необходимо выделить элемент в документе, при этом названия контекстных инструментов выделяются цветом, а контекстные вкладки отображаются рядом со стандартными вкладками. Контекстные вкладки содержат команды для работы с выделенным в документе элементом.

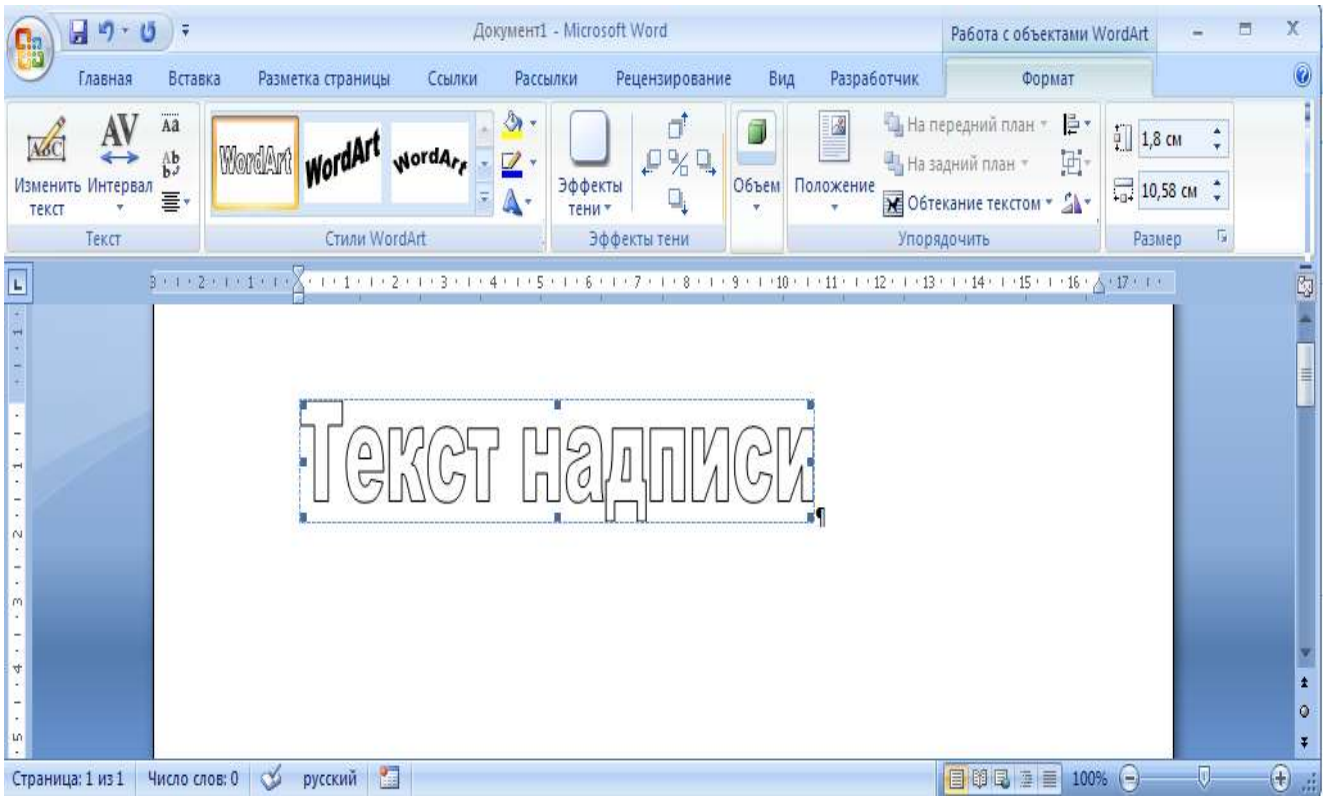


Рис. 7. Контекстное меню

Вкладки приложений.

Вкладки приложений заменяют стандартный набор вкладок при переходе в определенные представления или режимы создания содержимого, например, «Предварительный просмотр».

Панель быстрого доступа.

Панель быстрого доступа по умолчанию расположена в верхней части окна приложения Word и предназначена для быстрого доступа к наиболее часто используемым функциям (рис. 8).

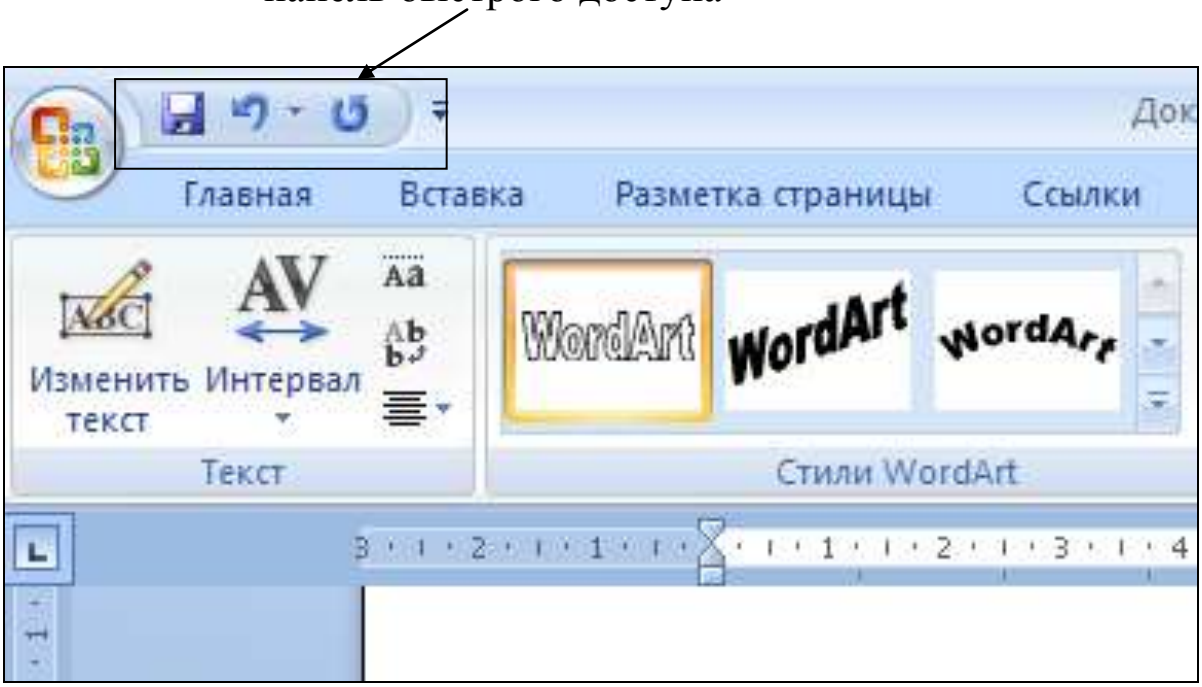


Рис. 8. Панель быстрого доступа

Панель быстрого доступа можно настраивать, добавляя в нее новые команды (рис. 9.). Для этого необходимо нажать кнопку **Microsoft Office**  , а затем щелкнуть **Параметры Word**. В списке слева выбрать пункт **Настройка**. В раскрывающемся списке **Выбрать команды из** щелкните **Все команды.**

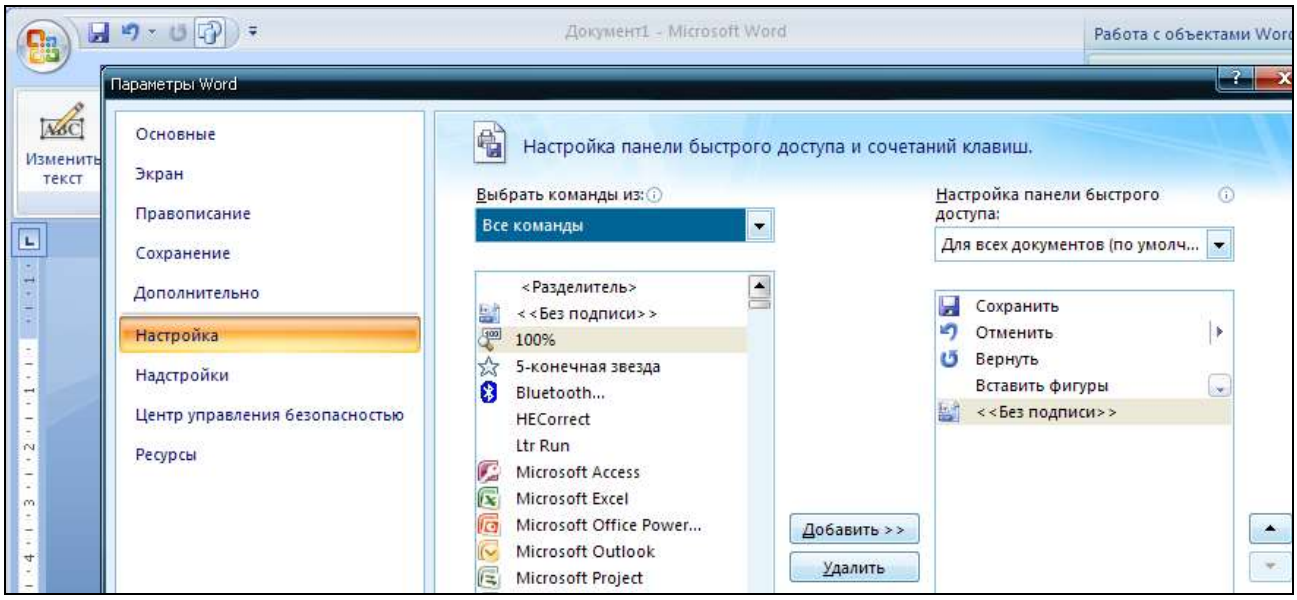


Рис. 9. Настройка панели быстрого доступа

Кнопки вызова диалоговых окон

Кнопки вызова диалоговых окон – это маленькие значки, которые могут отображаться в некоторых группах. По нажатию такой кнопки открывается соответствующее диалоговое окно или область задач, содержащая дополнительные параметры, связанные с данной группой (рис. 10).

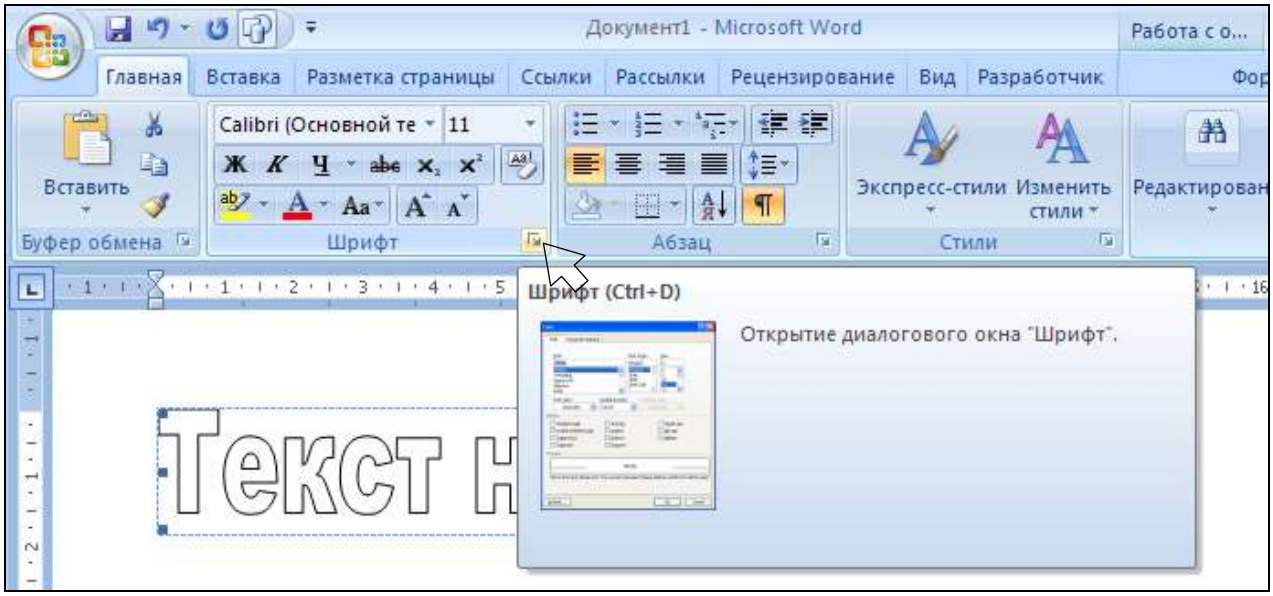


Рис. 10. Кнопки вызова диалоговых окон

Строка состояния.

Строка состояния отображается в нижней части рабочей области (рис. 11).

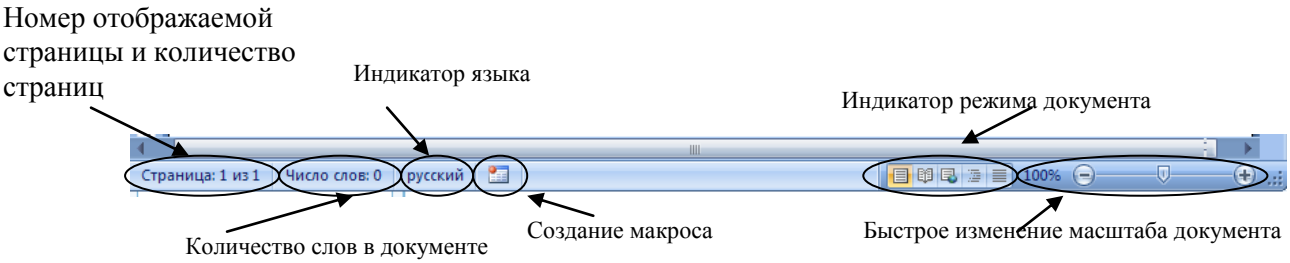


Рис. 11. Строка состояния

При щелке правой кнопкой мыши по строке состояния появляется меню Настройка строки состояния (рис. 12), где можно выбрать необходимые параметры строки состояния.

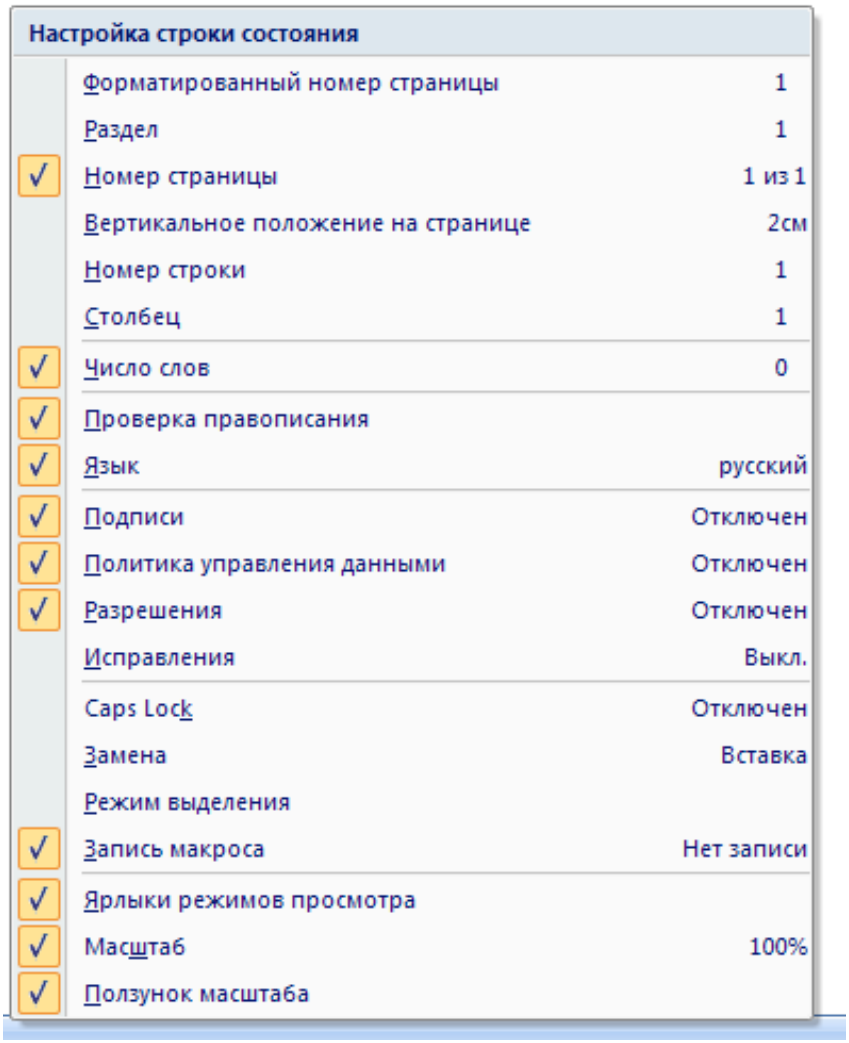


Рис. 12. Строка состояния

Редактор Microsoft Word позволяет просматривать документ в различных режимах.

1. Разметка страниц – отображает документ в том виде, в каком он будет выведен на печать.
2. Режим чтения – наиболее удобный режим для чтения документа на экране компьютера.
3. Веб-документ – просмотр документа в виде Веб-страницы.
4. Структура – просмотр документа в виде структуры.
5. Черновик – данный режим используется для быстрого редактирования текста.

Изменить режим просмотра документа можно на строке состояния или, выбрав вкладку **Вид**, в группе **Режимы** просмотра документа.

Масштаб документа также выбирается на вкладке **Вид**, в группе **Масшта**б, или на строке состояния.

Мини-панель инструментов.

Мини панель инструментов появляется при выделении текста или после правого щелчка мыши (рис. 13). Панель помогает работать со шрифтами, стилями и размерами шрифтов, выравниванием, цветом текста, уровнями отступов и маркерами. Настраивать ее нельзя, можно лишь изменять ее прозрачность.

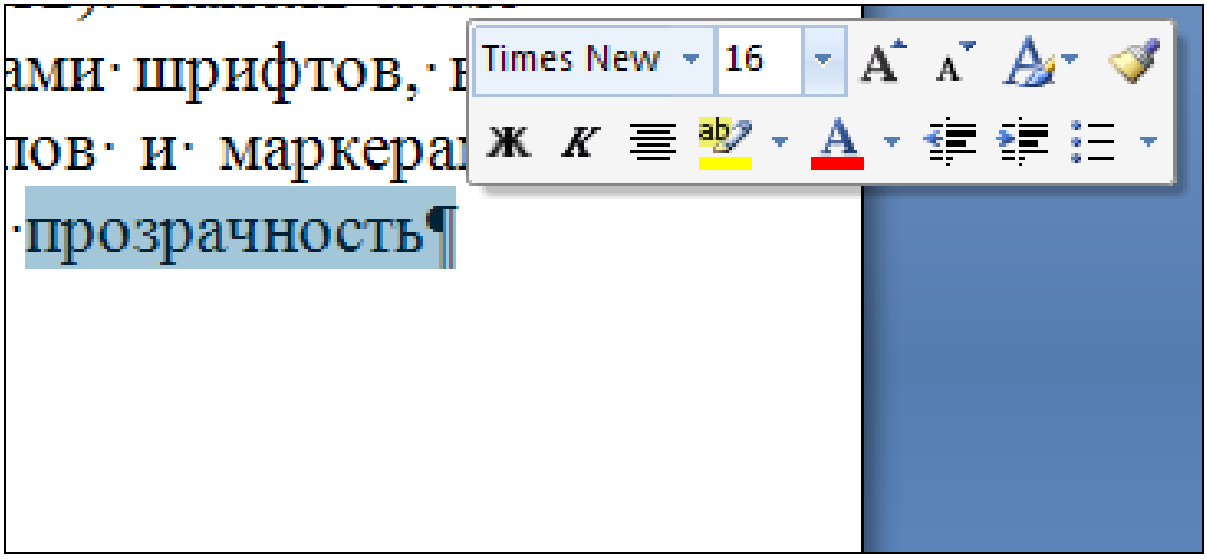


Рис. 13. Мини-панель инструментов

Контекстное меню

Для вызова контекстного меню следует щелкнуть правой клавишей мыши на обрабатываемом объекте. Контекстное меню появляется возле указателя мыши, и содержит команды для обработки выделенного объекта. На рисунке 14 показано контекстное меню, всплывающее при правом щелчке по объекту.

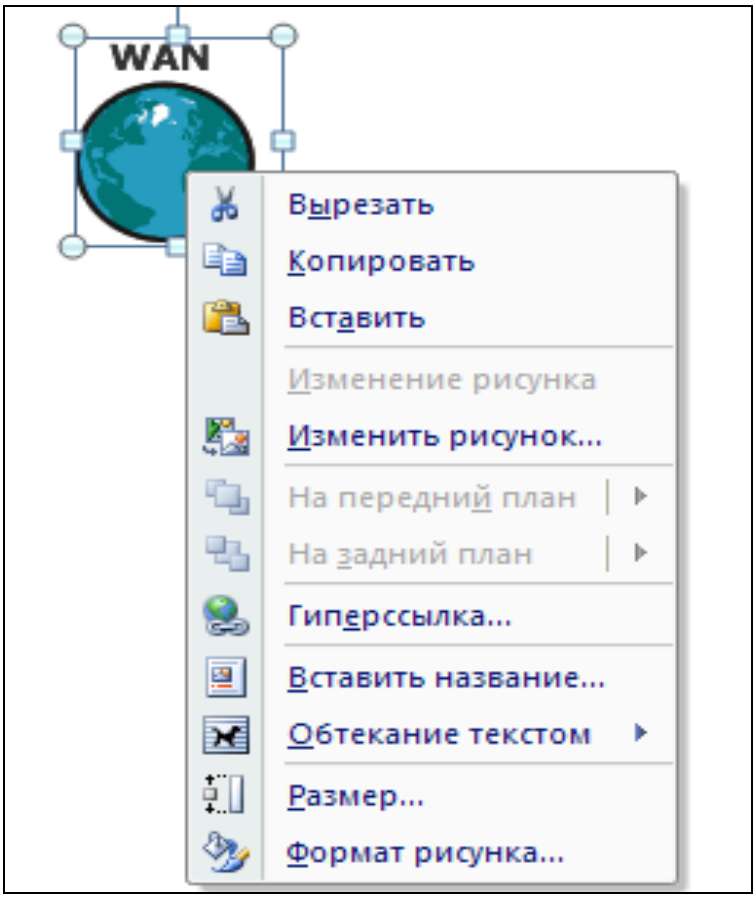


Рис. 14. Контекстное меню

Открытие документа

Для открытия документа, созданного в Microsoft Word 2007, необходимо щелкнуть на кнопке **Microsoft Office** , откроется меню (рис. 2), выбрать команду **Открыть**, появится окно **Открытие документа**, выбрать нужный, нажать на кнопке **Открыть**. Открыть документ можно также при двойном щелчке по документу. Если документ был создан в более ранних версиях редактора, то автоматически включается режим совместимости, а в строке заголовка окна документа отображается надпись **Режим ограниченной функциональности**. Включение режима совместимости гарантирует, что при работе с документом не будут использоваться возможности Office Word 2007 и пользователи более ранних версий Microsoft Word смогут редактировать любую часть этого документа.

В Microsoft Office Word 2007 можно открывать и сохранять файлы в других форматах. При выборе документа щелкнуть не на кнопке **Открыть**, а рядом по стрелке, появится меню (рис. 15), и выбрать необходимое действие.

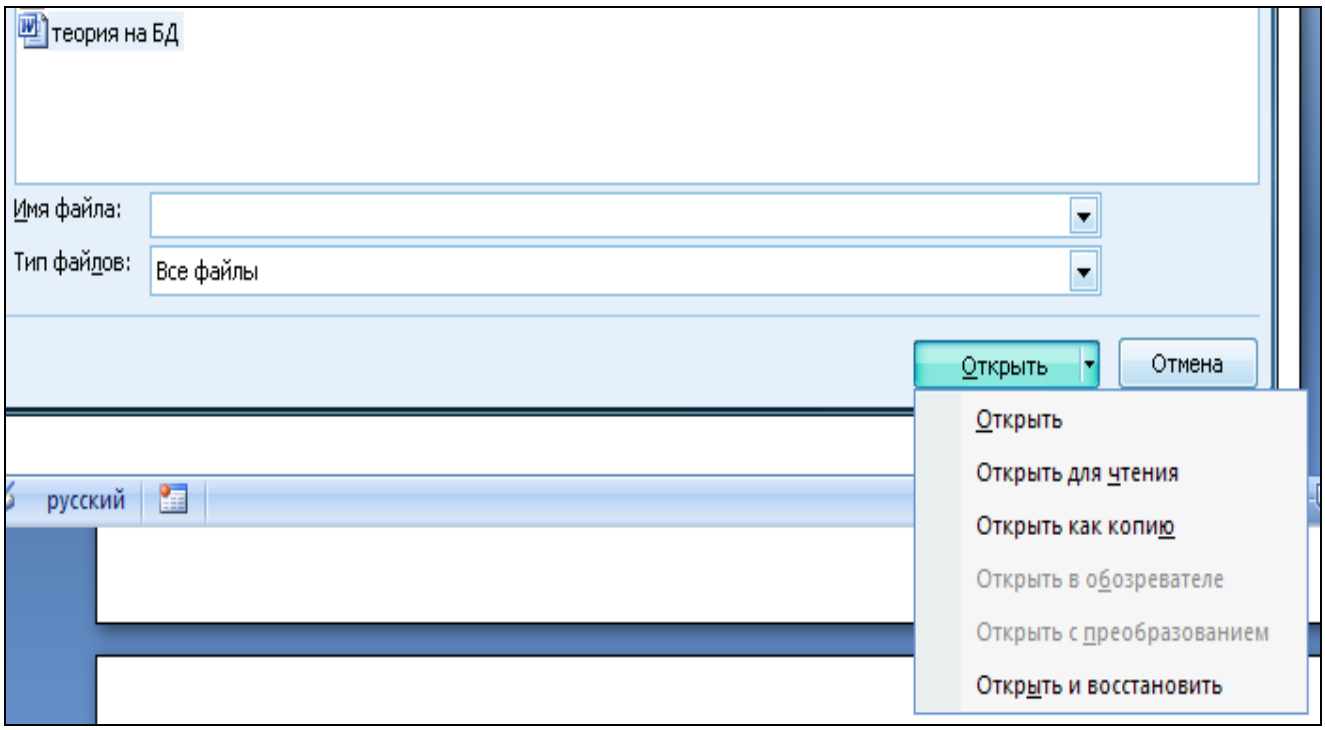


Рис. 15. Открытие документа

Сохранение документа

В Office Word 2007 можно сохранить файл в одном из нескольких форматов. Для этого необходимо щелкнуть значок , и выбрать команду **Сохранить как** (если выбрать команду **Сохранить**, документ будет сохранен в формате документа Microsoft Word 2007 с расширением “docx”). В диалоговом окне **Сохранить как** выберать в списке **Тип файла** нужный тип файла и ввести имя файла в поле Имя файла, нажать кнопку **Сохранить.**

Сохранить документ можно также щелкнув по кнопке на **Панели быстрого доступа.**

Выход из Microsoft Word

Для завершения работы с Microsoft Word необходимо закрыть окно программы, можно также щелкнуть значок  , и выбрать команду **Закрыть.**

Работа с текстом

Ввод текста

Курсор указывает место, в которое будет вводиться текст. Достигнув края страницы, курсор автоматически переходит в начало следующей строки. Для перехода в начало следующего абзаца следует нажать **Enter.**

Существует два режима ввода текста – вставки и замены. В режиме вставки при вводе новых символов, текст, содержащийся в документе, перемещается вправо от места ввода. В режиме замены старый текст заменяется новым. Переключение между режимами осуществляется выбором во всплывающем меню – Настройка строки состояния, или после щелчка по значку  , **Параметры Word**, Дополнительно в группе **Парамерты правки** путем снятия или установки флажка **Использовать режим замены.**

Выделение фрагмента текста

Прежде чем выполнить какую-нибудь операцию над фрагментом текста, его необходимо выделить одним из следующих способов:

– установить указатель мыши в левое поле (он превратится в стрелку, направленную вправо), при нажатии клавиши мыши выделится одна строка, при двойном нажатии – абзац, при тройном – весь документ;

– установить указатель мыши в левое поле напротив первой строки фрагмента (или в начало фрагмента), нажать клавишу мыши и, не отпуская ее, растянуть выделение на весь фрагмент;

– для выделения одного предложения следует нажать клавишу **Ctrl** и щелкнуть мышью в предложении;

– для выделения всего текста следует нажать клавишу **Ctrl** и щелкнуть мышью в левом поле;

– чтобы выделить фрагмент текста с помощью клавиатуры, необходимо установить курсор в начало фрагмента и, нажав клавишу **Shift**, клавишами управления курсором растянуть выделение на весь фрагмент.

Снять выделение можно щелчком мыши в любом месте текста. При выделении нового фрагмента предыдущее выделение снимается.

Редактирование текста

Символ справа от курсора удаляется клавишей **Delete**, символ слева от курсора – клавишей **Backspace**. Для удаления фрагмента текста следует выделить его и нажать клавишу **Delete** или **Backspace**. Если выделить фрагмент текста и набрать на клавиатуре новый текст, он вставится вместо выделенного фрагмента.

Чтобы разделить абзац на два, необходимо установить курсор в предполагаемый конец первого абзаца и нажать клавишу **Enter.**

Для включения/отключения режима просмотра непечатаемых символов используется кнопка  .

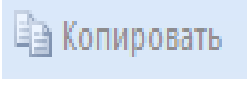
Отмена операций над текстом

Для отмены последней операции редактирования необходимо щелкнуть кнопку  . Чтобы вернуть последнюю отмененную операцию – щелкнуть кнопку  или  . Если щелкнуть на стрелке между кнопками, то откроется список операций, выполненных в текущем сеансе. Щелкнув на имени одной операции, можно отменить ее и все операции выполненные после нее.

Копирование текста

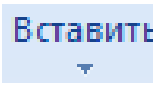
Для копирования фрагмента текста необходимо:

– выделить фрагмент текста;

– щелкнуть кнопку 

на вкладке **Главная** в группе **Буфер обмена;**

– установить курсор в место, куда следует вставить фрагмент;

– щелкнуть кнопку  или щелкнуть  и выбрать во всплывающем меню параметры вставки.

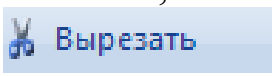
В процессе этой операции копия выделенного фрагмента текста помещается в буфере промежуточного хранения **Clipboard**, а потом попадает в документ.

При копировании текста можно также воспользоваться **Контекстным меню.**

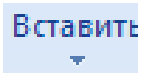
Перемещение текста

Для перемещения фрагмента текста необходимо:

– выделить фрагмент текста;

– щелкнуть кнопку  ;

– установить курсор в место, куда следует вставить фрагмент;

– щелкнуть кнопку  или щелкнуть  и выбрать во всплывающем меню параметры вставки.

При перемещении текста можно также воспользоваться **Контекстным меню.**

Также переместить фрагмент текста можно путем перетягивания выделенного фрагмента в нужное место (место вставки указывает знак |).

Если при перетягивании выделения держать нажатой клавишу **Ctrl**, то фрагмент будет скопирован.

Вставка символа

Для вставки в текст символа, отсутствующего на клавиатуре, необходимо: выбрать вкладку **Вставка**, группу **Символы** и нажать кнопку  .

Вставка формул

Вставить в текст формулу можно двумя способами:

– Вкладка **Вставка**→группа **Символ**→**Формула.**

– Вкладка **Вставка**→группа **Текст**→**Объект**→**Microsoft Equation 3.0.**

Форматирование текста

Выбор темы

Темы документов, применяемые непосредственно, влияют на на вкладке **Разметка страницы** в группе **Темы** нажмите кнопку **Темы** (рис. 16).

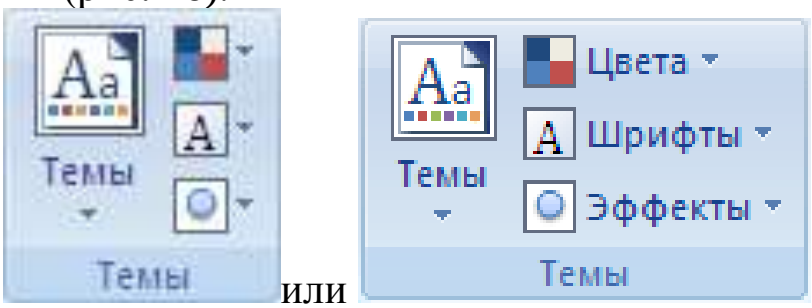


Рис. 16. Темы документа

Для применения определенной темы документа, нужно выбрать ее в группе **Встроенные**. Чтобы применить пользовательскую тему – выбрать тему в группе **Пользовательские.**

Отдельно стиль выделенного фрагмента или абзаца, где установлен курсор, изменить можно вкладке **Главная** в группе **Стили**. Окно **Стили** отображается, щелкнув в нижнем правом углу группы **Стили** кнопку вызова диалоговых окон, чтобы получить доступ к **Библиотеке стилей**. Если курсор установлен в абзаце, то при наведении на кнопки в группе **Стили**, текст абзаца примет указанный стиль, если в документе часть текста выделена, то стиль изменится только в этой части.

Установка позиций табуляции

Для ручной установки позиций табуляции можно использовать линейку, расположенную слева, справа и в центре документа. Если горизонтальная линейка не отображается, На вкладке **Вид** в группе **Показать** или скрыть установить галочку напротив **Линейка.**

Доступны следующие типы табуляции:

– Значок **По левому краю** задает начальную позицию текста таким образом, чтобы при вводе текста он сдвигался вправо.

– Значок **По центру** задает позицию в середине текста таким образом, чтобы при вводе текста он центрировался относительно этой точки.

– Значок **По правому краю** задает правую (конечную) позицию текста так, что при вводе текста он будет сдвигаться влево.

– Значок **По разделителю** задает позицию выравнивания чисел относительно десятичного разделителя. Независимо от количества цифр десятичный разделитель будет оставаться на этой позиции. (Числа могут выравниваться только по отношению к знаку десятичного разделителя. Этот параметр табуляции нельзя использовать для выравнивания чисел относительно другого символа, например, дефиса или амперсанда.)

– Параметр **С чертой** не предназначен для выравнивания текста. При выборе этого значка в позиции табуляции будет вставлена вертикальная черта.

Для определения позиции табуляции с особой точностью, недоступной при помощи линейки, или вставки определенного символа (заполнителя) перед табуляцией следует использовать диалоговое окно **Табуляция**. Для открытия этого диалогового окна дважды щелкните любую из позиций табуляции на линейке.

Изменение параметров шрифта и иабзаца

Для изменения параметров символов можно использовать мини-панель, воспользоваться вкладкой **Главная**, группа **Абзац** (рис. 17).

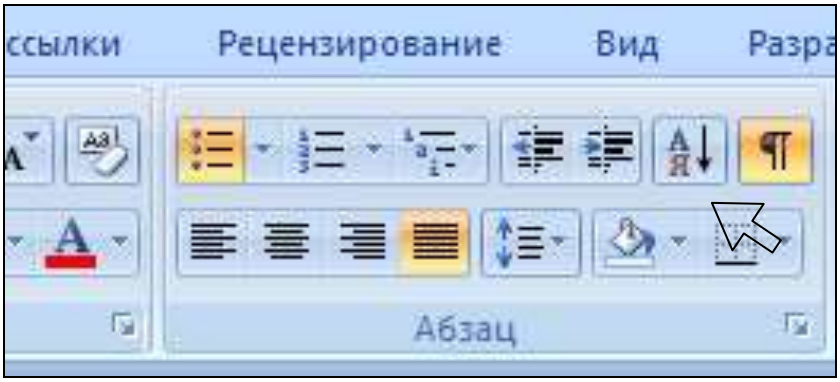


Рис. 17. Группа Абзац

Изменение параметров абзаца происходит путем открытия диалогового окна **Абзац** (рис. 18).

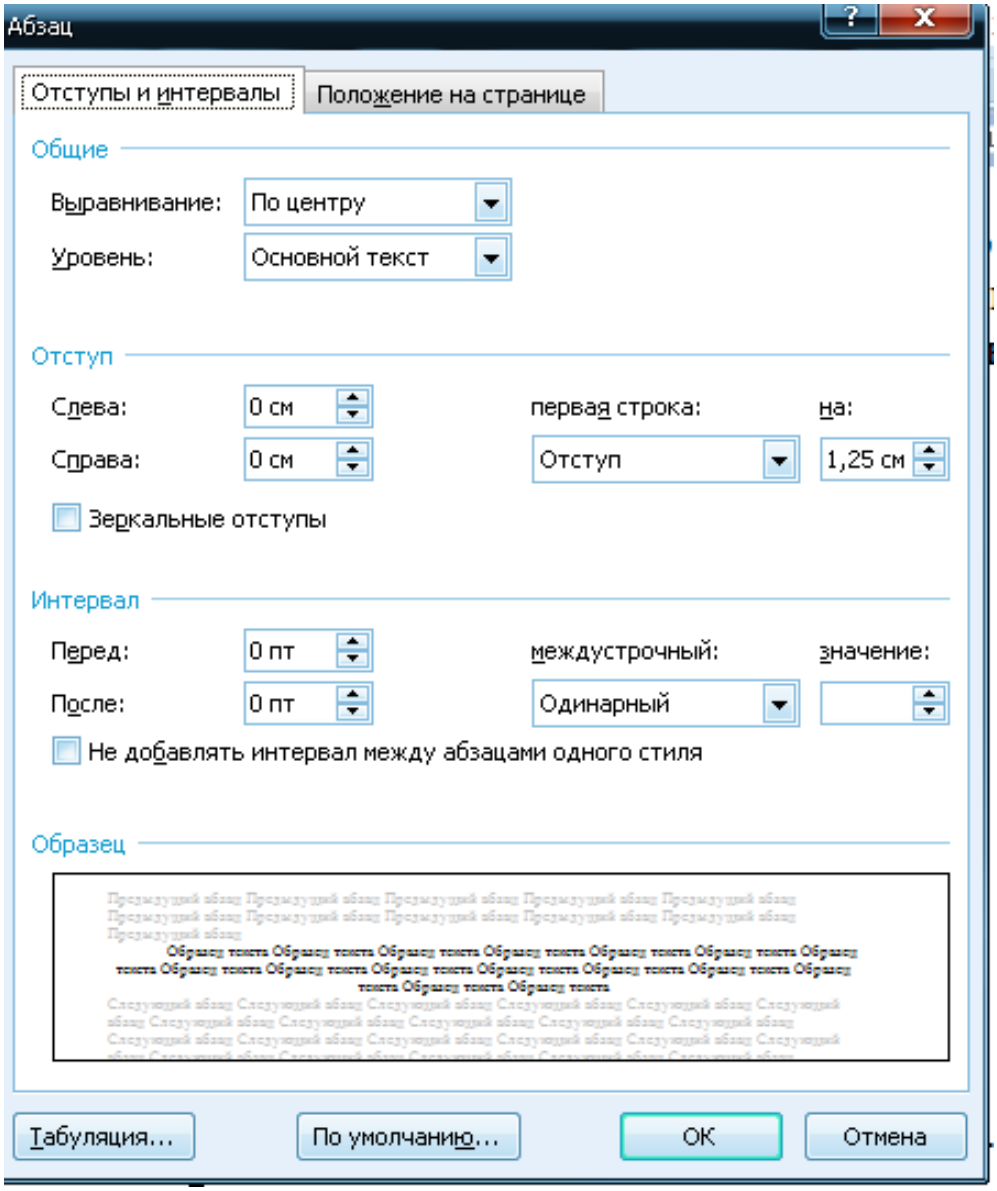


Рис. 18. Диалоговое окно Абзац

Группа кнопок  изменяет положение текста в абзаце: **По левому краю, По центру, По правому краю, По ширине.**

Кнопка  устанавливает междустрочный интервал, можно также настроить интервал, добавляемый перед абзацами и после них.

Кнопка  изменяет цвет фона выделенного текста и абзаца.

Кнопка  выделяет границы текста или абзаца.

Группа кнопок  изменяет отступы абзаца, соответственно уменьшает и увеличивает отступ.

Кнопка  вызывает диалоговое окно сортировка текста.
Кнопка  показывает знаки, не отображающие при печати

Изменение интервала и положения символов

Для изменения интервала и положения символов используется вкладыш **Интервал** диалогового окна **Шрифт** (рис. 19).

В поле **Масштаб** выбирается степень растяжения или сжатия символов. Масштаб можно изменить на вкладке **Вид.**

В поле **Интервал** устанавливается межсимвольный интервал.

В поле **Смещение** устанавливается вертикальное положение символов.

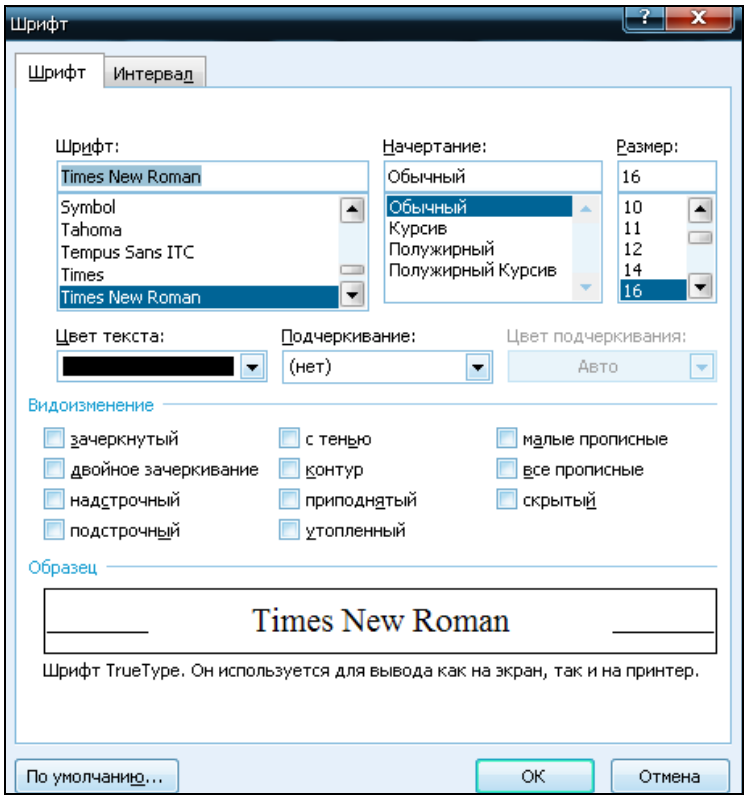


Рис. 19. Диалоговое окно Шрифт

Создание и редактирование списков

Microsoft Word позволяет быстро составлять списки с маркерами, нумерацией и многоуровневые списки с нумерацией. Элементом списка считается абзац текста.

Для создания списка необходимо выделить абзацы, которые следует сделать элементами списка или установить курсор в тот абзац, с которого будет начинаться список, выбрать на вкладке **Главная** в группе **Абзац** нужный тип списка из существующих стилей или создать новый  .

Различные стили маркеров и форматы нумерации можно найти, щелкнув стрелку рядом с полями **Маркеры** или **Нумерация** на вкладке **Главная** в группе **Абзац.**

Можно передвигать весь список влево или вправо. Щелкните маркер или цифру в списке и перетащите на новое место. Весь список переместится за указателем. Уровни нумерации не изменятся.

Иногда при создании списка его элементы имеют разный отступ, и список необходимо выровнять.

1. Щелкнуть перед элементом списка, который требуется выровнять.
2. Щелкнуть правой кнопкой мыши и выберите команду **Изменить отступы в списке.**
3. Чтобы изменить выравнивание, необходимо изменять значения в полях:

- **Положение номера –** Выберите положение номера или маркера.
- **Отступ текста** – Определите расположение текста.
- **Символ после номера** – Укажите расстояние между выбранными номерами или маркерами и текстом **Знак табуляции,** указав соответствующий параметр: **Пробел** – один пробел между выбранными номерами или маркерами и текстом – или **(нет),** чтобы не оставлять места между выбранными номерами или маркерами и текстом.

Не рекомендуется использовать линейку для настройки отступов списка.

Оформление страниц документа

В приложении Microsoft Word задать параметры страницы можно несколькими способами. Можно использовать поля страницы, установленные по умолчанию, или задать собственные.

1. **Выделение дополнительного пространства страницы под переплет.** Для этого используется поле переплета, которое можно установить у бокового или верхнего края страницы (рис. 20). Поле переплета гарантирует сохранность текста при брошюровке документа.

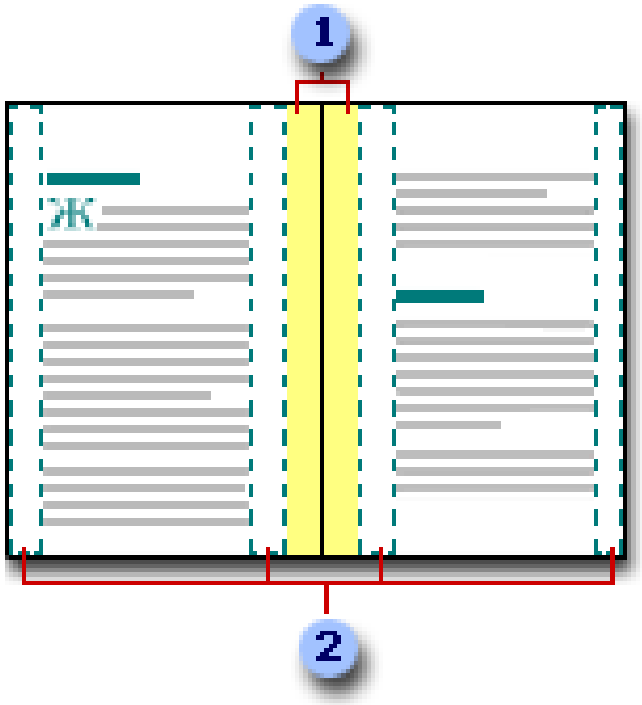


Рис. 20. Поле переплета
1 – Поля переплета для брошюровки
2 – Зеркальные поля для нечетных страниц

2. **Установка полей для четных и нечетных страниц.** Для задания параметров четных и нечетных страниц в документах с двусторонней печатью, например, в книгах или журналах, используются зеркальные поля. В данном случае поля левой страницы являются зеркальным отражением полей правой страницы (т. е. для страниц устанавливаются одинаковые внутренние и внешние поля).

3. **Примечание.** Поля переплета можно задать и для документа с зеркальными полями, если в этом документе требуется выделить дополнительное пространство для переплета.

4. **Добавление брошюры**. В диалоговом окне **Параметры** страницы выберите пункт **Брошюра**. Этот тип параметров страницы позволяет печатать меню, приглашения, программы меро-

приятий и другие типы документов, оформляемые в виде брошюры.

Изменение и установка полей страницы

На вкладке **Разметка страницы** в группе **Параметры страницы** выбрать команду **Поля** (рис. 21).

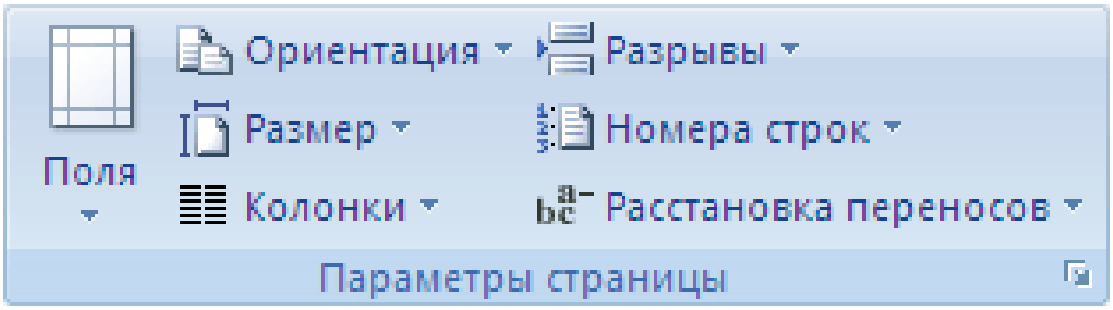


Рис. 21. Параметры страницы

Выберите нужный тип полей. Для установки наиболее часто используемого размера полей выберите в списке значение Обычное. Выбранный тип полей автоматически применяется ко всему документу.

Параметры полей можно задать самостоятельно. Перейдите на вкладку **Поля**, нажмите кнопку **Настраиваемые поля**, а затем введите новые значения в полях **Верхнее**, **Нижнее**, **Левое** и **Правое**. Установить поля страницы можно также с помощью координатных линеек для этого необходимо установить указатель мыши на границу серого и белого участка (он будет иметь вид двунаправленной стрелки) и перетянуть ее в нужное место.

Изменение ориентации страниц

Для всего документа или для его части можно выбрать либо книжную (вертикальную), либо альбомную (горизонтальную) ориентацию страниц. При изменении ориентации страниц коллекции готовых страниц и обложек также изменяются в соответствии с вновь выбранной ориентацией страниц.

Для того чтобы изменить ориентацию страниц во всем документе на вкладке **Разметка страницы** в группе **Параметры**

страницы нужно выбрать команду **Ориентация**, а в ней параметр **Книжная** или **Альбомная**.

Если изменить ориентацию необходимо лишь у нескольких страниц, то после выделения текста на этих страницах нужно перейти на вкладку **Разметка страницы**, в группе **Параметры страницы** щелкнуть команду **Поля** и отметить **Альбомная** или **Книжная**, а в списке **Применить** выбрать пункт **К выделенному тексту**.

Вставка разрывов страниц

Microsoft Word автоматически разбивает текст на страницы.

Разрыв страницы можно вставить в любом месте документа. Кроме того, возможна автоматическая расстановка разрывов страниц. При ручной расстановке разрывов страниц в документе большого объема могут потребоваться частые переносы разрывов по мере редактирования документа, для того, чтобы избежать переноса разделов, можно задать параметры определяющие, где автоматически будут вставляться разрывы страниц.

Вставка принудительного разрыва осуществляется следующим образом: щелкнуть место, откуда следует начать новую страницу, а после на вкладке **Вставка** в группе **Страницы** выберите команду **Разрыв страницы** (рис. 22).

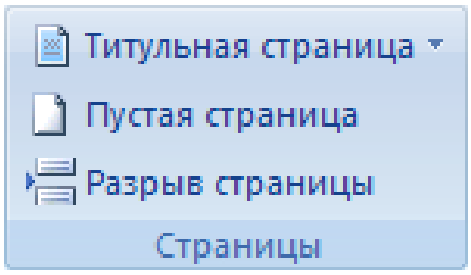


Рис. 22. Вставка принудительного разрыва

Можно запретить вставку разрыва страницы в середину абзаца, для этого нужно выбрать абзац, для которого необходимо запретить вставку разрыва страницы. Перейти на вкладку **Главная** и в группе **Абзац** нажать кнопку запуска диалогового окна (рис. 17), а затем щелкнуть вкладку **Положение на странице**, установить флажок **Не разрывать абзац** (рис. 23).

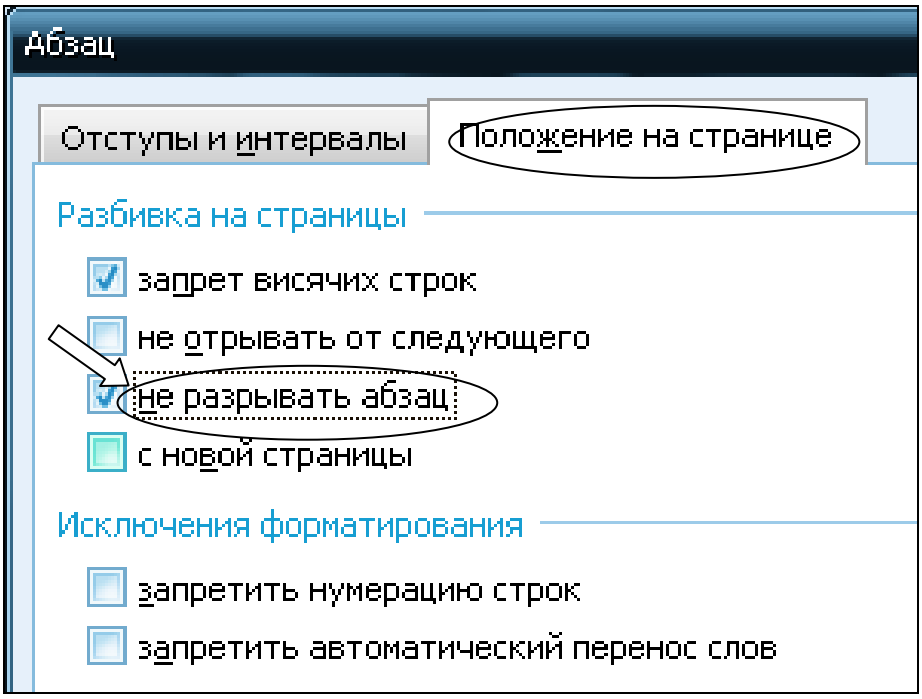


Рис. 23. Запрет разрыва абзаца

Можно запретить ставить разрыв раздела между абзацами, для этого следует выделить абзацы, которые должны находиться на одной странице, затем диалоговое окно **Абзац**, как описано ранее, и в появившимся диалоговом окне установить флажок **Не отрывать от следующего** (рис. 24).

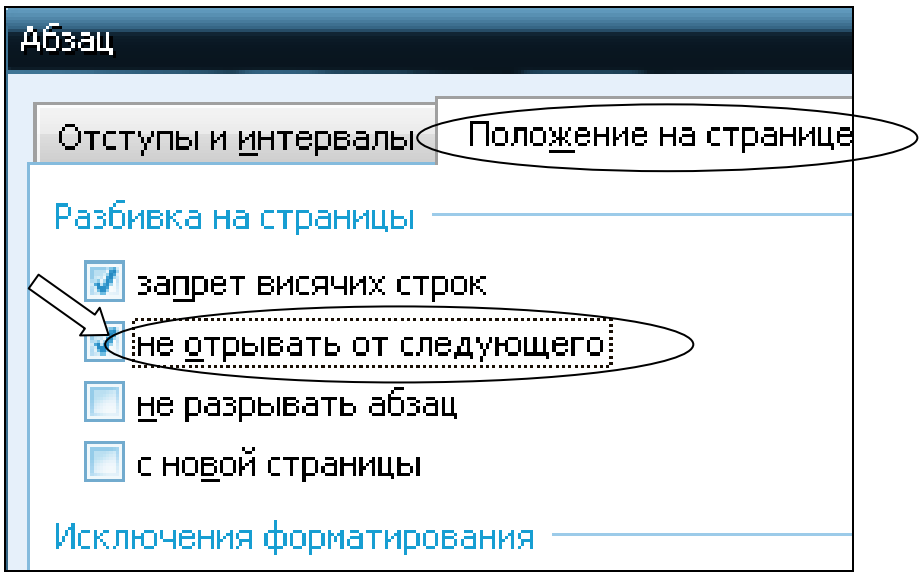


Рис. 24. Запрет разрыва нескольких абзацев

Для того чтобы добавить разрыв раздела между абзацами:

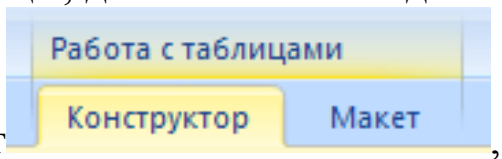
1) щелкнуть абзац, перед которым необходимо вставить разрыв страницы;

2) на вкладке **Главная** в группе **Абзац** нажать кнопку запуска диалогового окна, а затем щелкнуть вкладку **Положение на странице**.

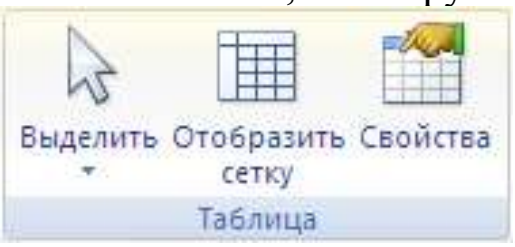
3) установить флажок С новой страницы.

В профессионально оформленном документе страница не завершается первой строкой из нового абзаца и не начинается последней строкой из абзаца предыдущей страницы. Такие строки называются висячими. Чтобы этого избежать висячих строк, на вкладке Положение на странице диалогового окна **Абзац** нужно установить флажок Запрет висячих строк. По умолчанию этот режим включен.

Можно запретить разрывать строки таблицы, если ее размер меньше страницы, для этого на вкладке **Работа с таблицами** выбрать **Макет**



, в группе **Таблица** нажать кнопку **Свойства**



, перейти на вкладку **Строка** и снять флажок **Разрешить перенос строк на следующую страницу**.

Вставить разрыв страницы можно также выбрав команду **Разрыв** на вкладке **Разметка страницы** (рис. 25) или через вкладку **Вставка** в группе **Страницы** выполнить команду **Разрыв страницы** (рис. 26).

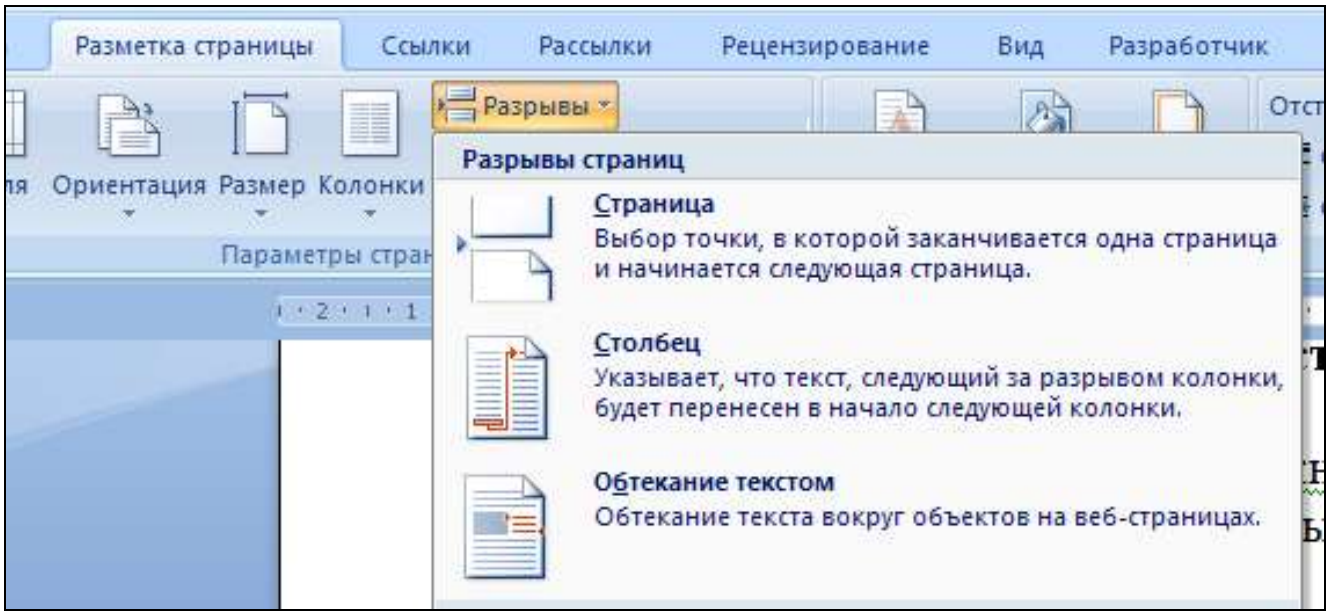


Рис. 25. Вставка разрывов страницы в документ

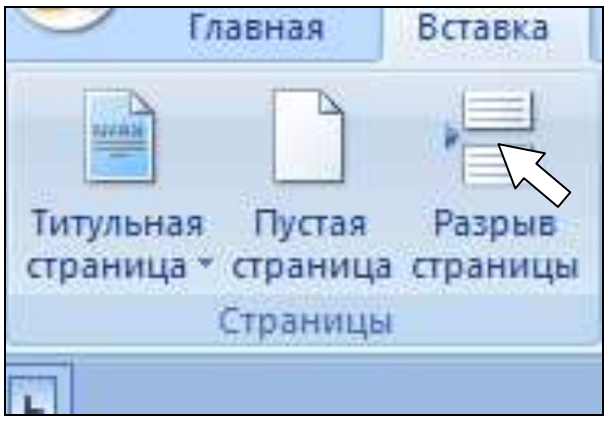


Рис. 26. Вставка разрывов страницы в документ,
используя вкладку **Вставка**

Для изменения разметки и форматирования одной или нескольких страниц документа используются разрывы раздела. Например, можно разметить часть страницы с одной колонкой как имеющую две колонки, разделить главы документа так, чтобы нумерация страниц для каждой из глав начиналась с 1, или задать разные колонтитулы для различных разделов документа. Параметры документа, доступные для изменения при помощи разрывов разделов:

– Поля
– Размер и ориентация бумаги
– Источник бумаги для принтера
– Границы страницы

– Выравнивание текста на странице по вертикали
– Колонтитулы
– Колонки
– Нумерация страниц
– Нумерация строк
– Сноски

Разрывом раздела определяется форматирование текста в предыдущем разделе. При удалении разрыва раздела вместе с ним удаляется форматирование текста в разделе, расположенном перед ним. Этот текст становится частью следующего раздела и принимает соответствующее форматирование. Например, если разделить документ на главы с помощью разрывов разделов, а затем удалить разрыв раздела в начале главы 2, главы 1 и 2 будут объединены в один раздел с форматированием, заданным для главы 2.

Разрыв раздела, определяющий форматирование последней части документа, не отображается. Для изменения форматирования документа щелкните последний абзац документа.

Для того чтобы применить разрыв раздела на вкладке Разметка страницы щелкнуть Разделы и во всплывающем меню выбрать необходимый разрыв (рис. 27).

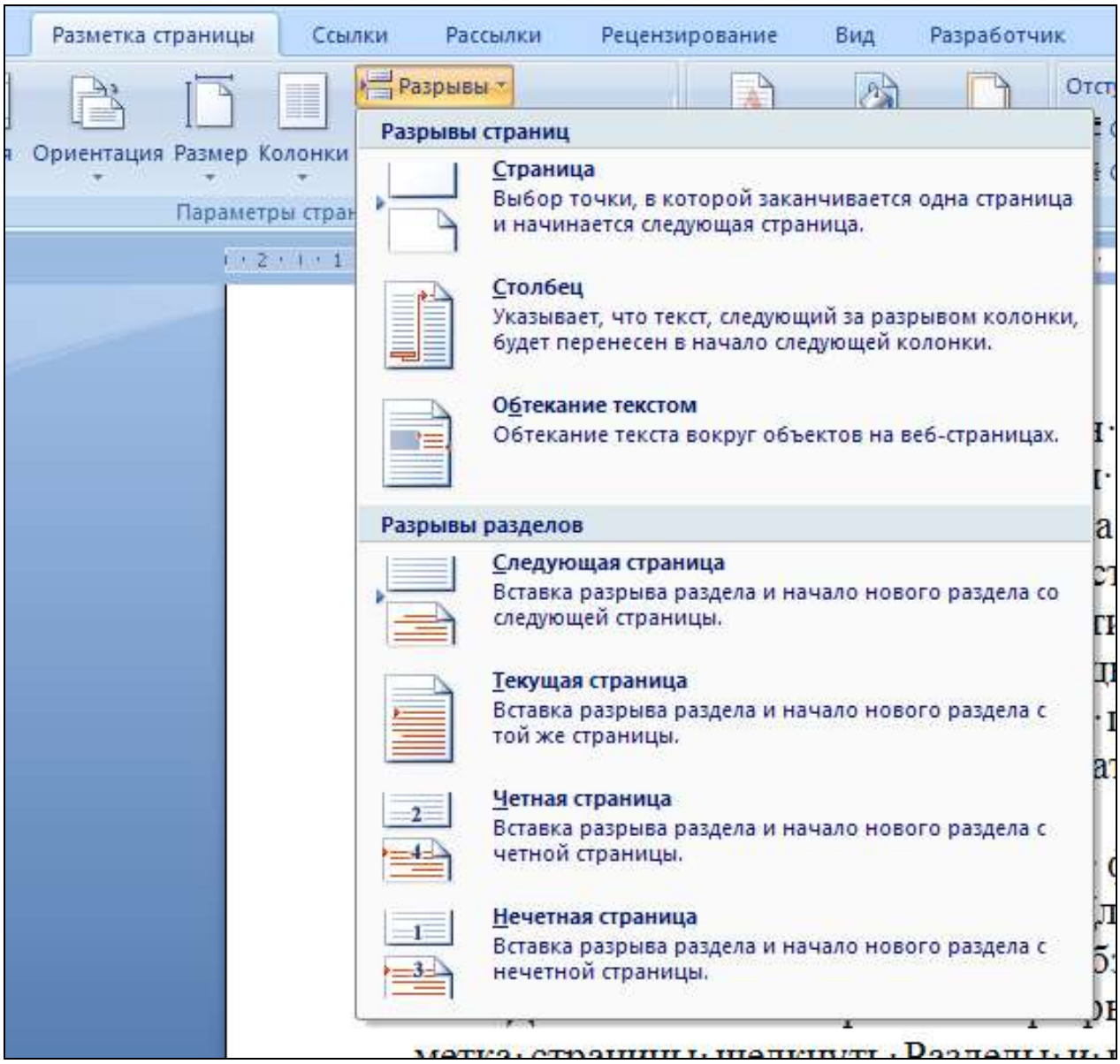


Рис. 27. Вставка разрыва раздела.

Добавление и удаление страницы

Если страница в Microsoft Word до конца заполнена текстом и изображениями, автоматически вставляется разрыв страницы и начинается новая страница. Однако можно добавить в документ пустую страницу или страницу с предварительно заданной разметкой. Ненужные страницы можно удалить из документа путем удаления разрывов страниц.

Для того чтобы добавить пустую страницу в документ нужно щелкнуть в том месте, где требуется вставить таблицу, затем на вкладке **Вставка** в группе **Страницы** выбрать команду

Пустая страница (рис. 28). В документ можно также добавить титульную страницу, выбрав из коллекции, которая появляется после нажатия кнопки **Титульная страница.**

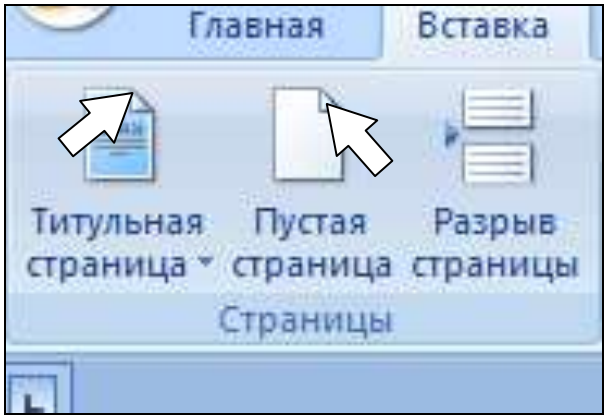


Рис. 28. Вставка страницы в документ

Вставка колонтитулов и нумерация страниц

Колонтитулами называют области, расположенные в верхнем, нижнем и боковом полях каждой из страниц документа (рис. 29).

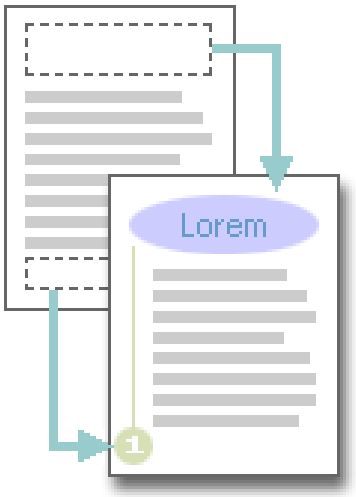


Рис. 29. Колонтитулы

Колонтитулы содержат текст и изображения, которые можно изменять. Например, в колонтитулы можно включать номера страниц, время, дату, эмблему компании, название документа, имя файла, а также фамилию автора.

Для того чтобы вставить готовый колонтитул в документ, нужно щелкнуть необходимый вариант в группе **Колонтитулы** во вкладке **Вставка** и выбрать подходящий из коллекции (рис. 30), если выбрать команду **Изменить колонтитул**, то в открывшемся контекстном меню (рис. 31) можно будет изменять текущий колонтитул.

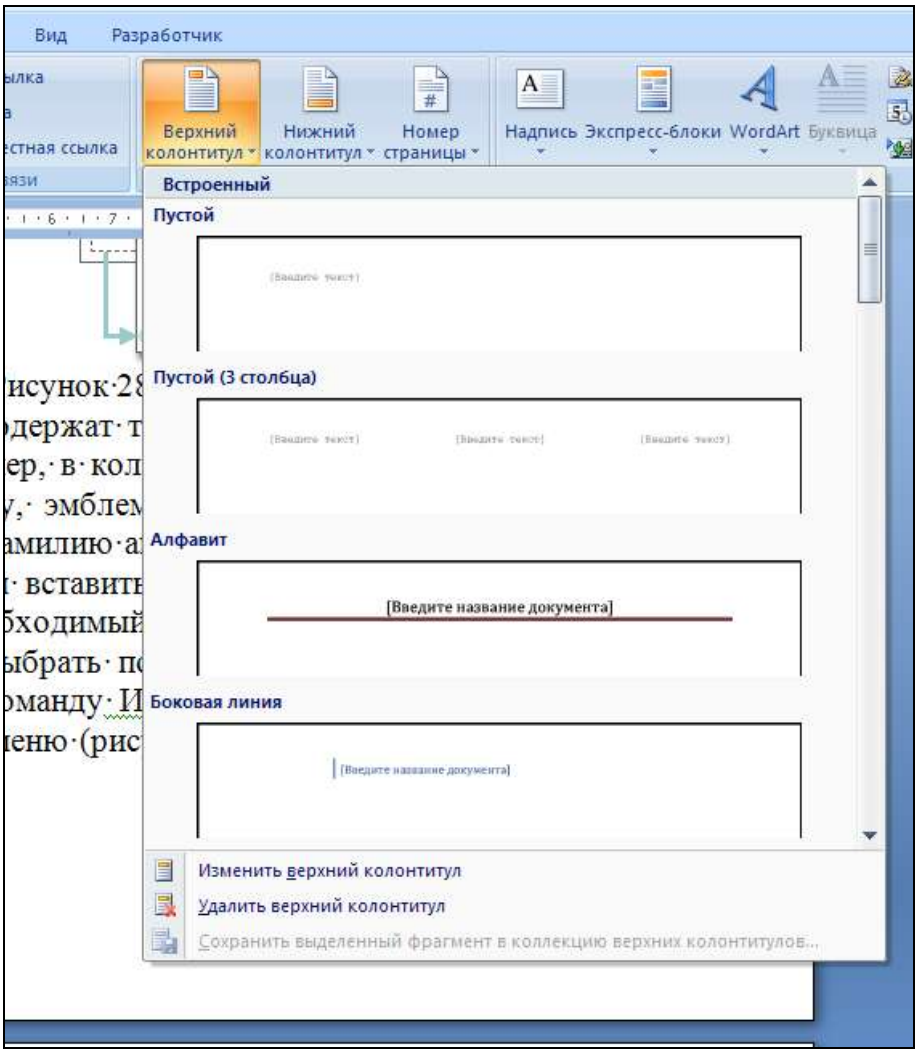


Рис. 30. Выбор колонтитула из коллекции

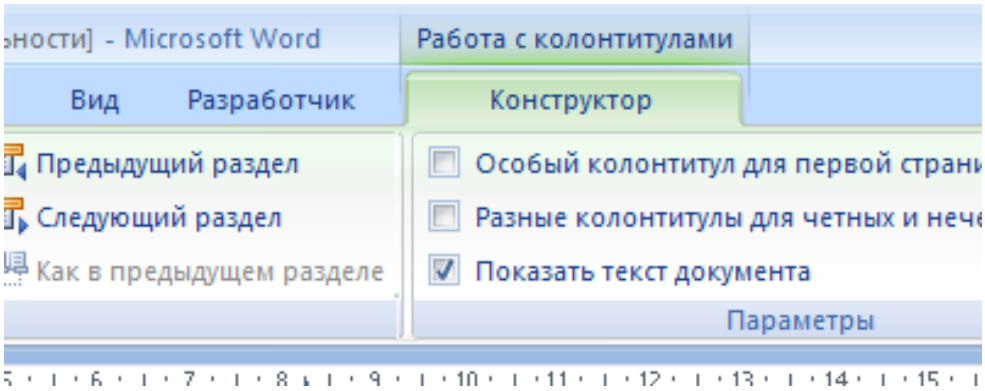


Рис. 31. Работа с колонтитулами

Колонтитул можно назначить один на все разделы документа, а можно, предварительно щелкнув **Колонтитулы** в группе **Переходы** нажмите кнопку **Как в предыдущем разделе** , применить свои параметры колонтитула для выбранного раздела.

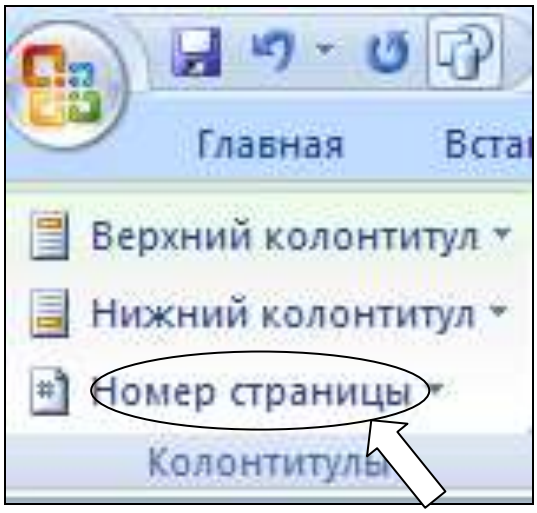
Иногда необходимо различать колонтитулы четных и нечетных страниц, для этого нужно установить флажок **Четных и нечетных страниц** в группе **Параметры** вкладки **Колонтитулы.** Колонтитулы первой страницы (титульной) также можно изменять без изменения колонтитулов документа, для этого нужно установить флажок **Особый колонтитул** для первой страницы в группе **Параметры** вкладки **Колонтитулы.**

Номера страниц, связанные с колонтитулами, добавляются в верхней или нижней части страницы, а также на полях. Сведения в колонтитулах отображаются в затененном виде, их нельзя изменить одновременно с основным текстом документа.

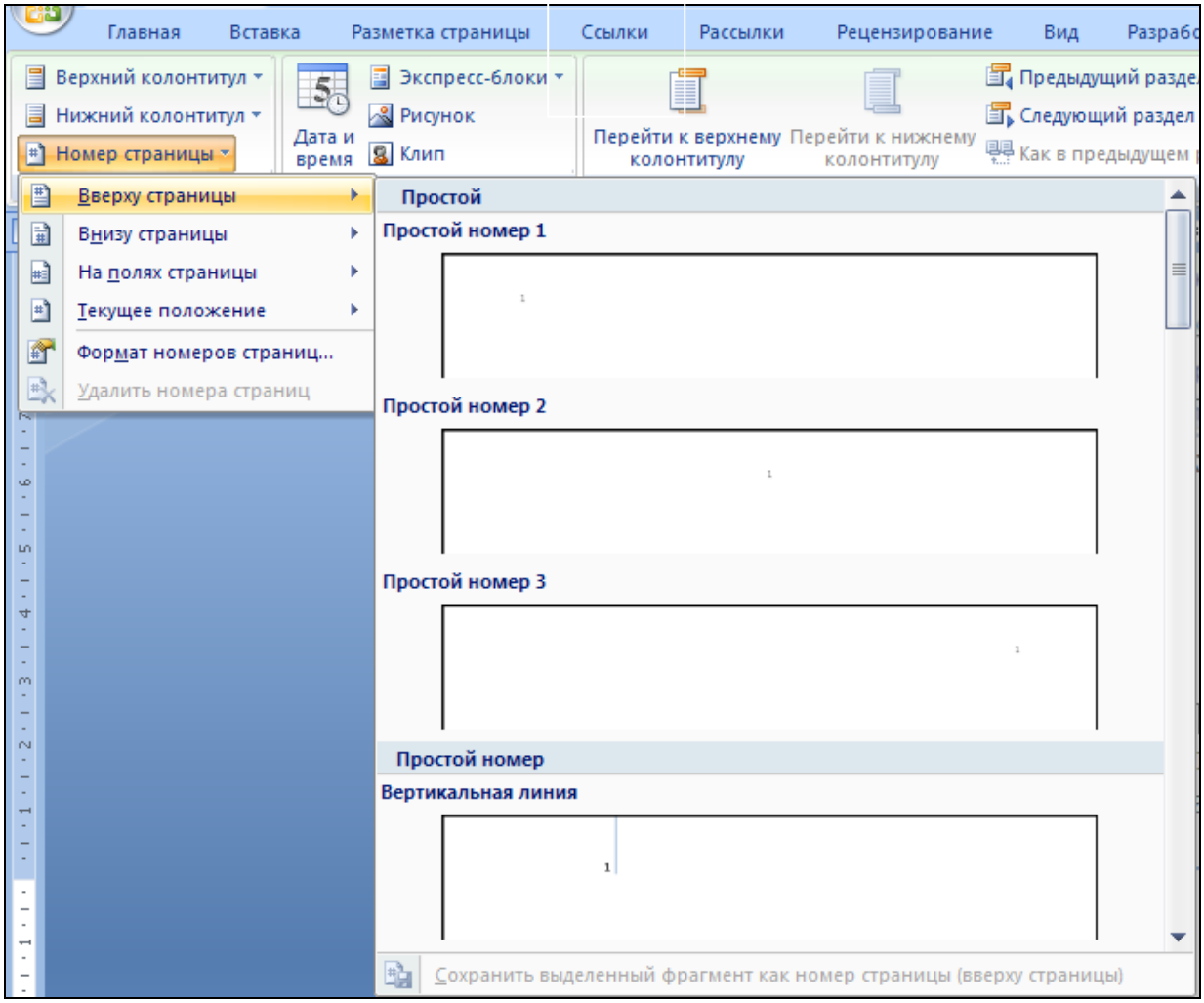
Чтобы изменить колонтитул или сведения на полях страницы, дважды щелкните колонтитул, а затем щелкните вкладку **Колонтитулы** в контекстных инструментах **Работа с колонтитулами.**

Чтобы добавить номер страницы **(**рис. 32 а, б).

1. На вкладке **Вставка** в группе **Колонтитулы** выберите команду **Номер страницы**, рисунок 32 а).
2. Выбор значения **Вверху страницы**, **Внизу страницы** или **На полях страницы** определяет, где в именно документе будут отображаться номера страниц.
3. Выберите в коллекции вид номеров страниц (рис. 32 б).



а)



б)

Рис. 32 (а; б). Вставка номера страницы

Номера страниц можно отформатировать по своему усмотрению, достаточно нажать на **Формат номеров страниц**, который легко заметить на рисунке 32 б), и установить в появившемся меню необходимые параметры, (рис. 33).

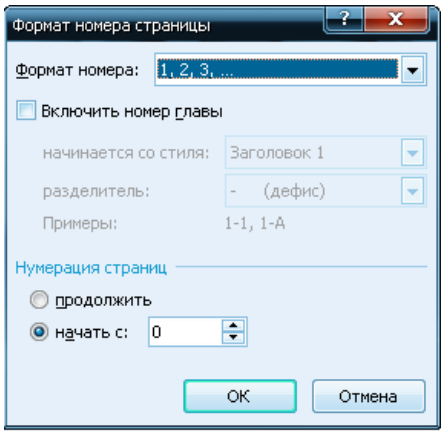


Рис. 33. Изменение формата номера страницы

Здесь в поле **Формат номера** выбирается формат номера; в поле **Нумерация страниц** устанавливается начало нумерации; Включить номер главы – к номеру страницы добавляется номер главы или раздела документа, разделитель также выбирается.

Колонки

Колонки можно применить ко всему документу или к разделу, часто встречаются случаи, когда разделы документа состоят разного числа колонок. Текст вводится в колонки последовательно и переходит к следующей после заполнения предыдущей или принудительно, выбрав на вкладке **Разметка страницы** в группе **Параметры страницы** кнопку **Разрывы**, и во всплывающем меню – **Столбец** (рис. 34).

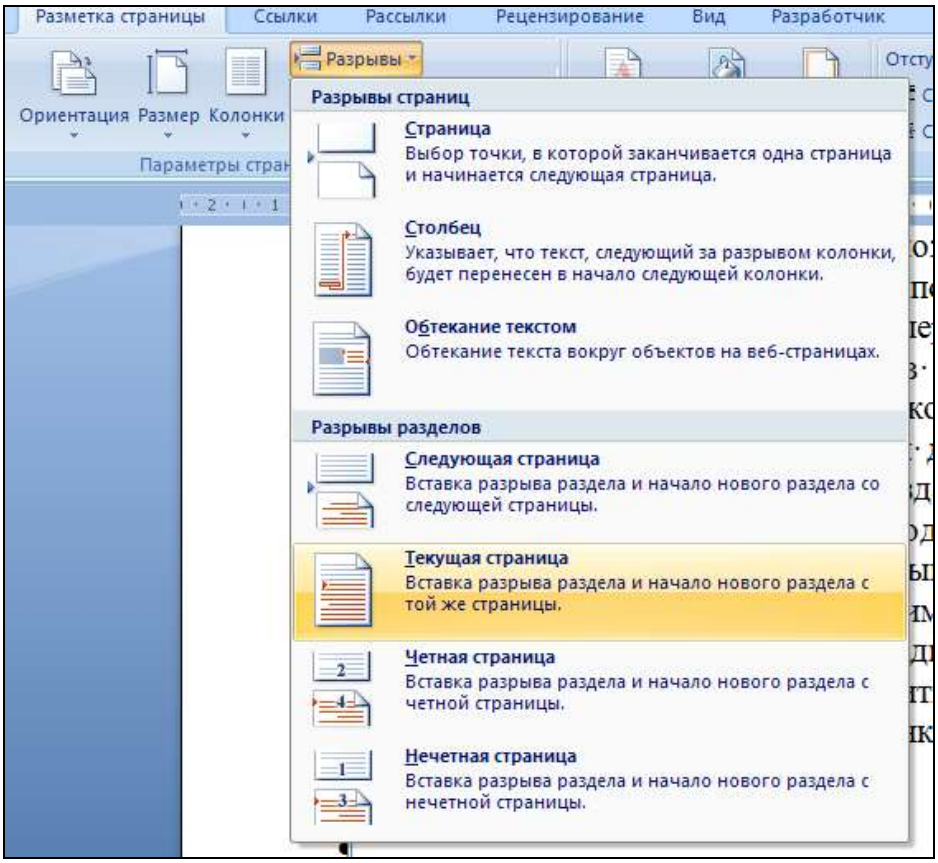


Рис. 34. Вставка разрыва раздела на текущей странице

Создать колонки в документе можно несколькими способами. Если текст, который должен располагаться в колонках, уже создан, необходимо выделить его и перейти на вкладку **Разметка страницы**, а там щелкнуть **Колонки** в группе **Параметры страницы**, во всплывающем меню выбрать количество колонок в документе или щелкнуть другие колонки для открытия диалогового окна **Колонки**. Если текст еще не создан, можно сначала создать новый раздел, установить там необходимое количество колонок, а потом вводить текст: вставить разрыв раздела на текущей странице (рис. 34), выбрать необходимое количество колонок в группе **Параметры страницы** на вкладке **Разметка страницы**. После заполнения колонок снова вставить **Разрыв раздела Текущая страница** и на последующем разделе установить **Колонки** – **Одна**.

Диалоговое окно **Колонки** расширяет возможности пользователя (рис. 35).

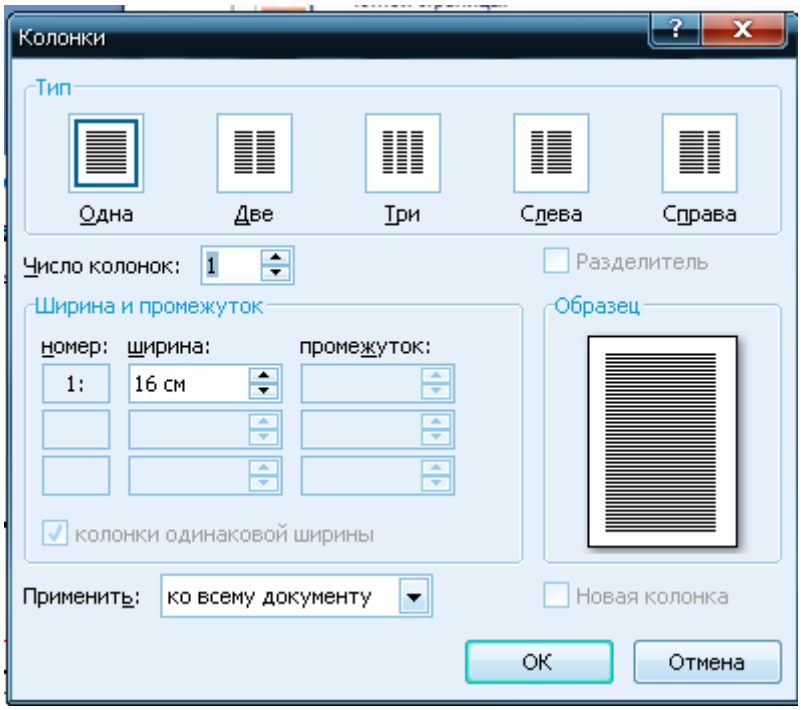


Рис. 35. Диалоговое окно Колонки

Работая в диалоговом окне Колонки, можно выбрать предложенную верстку документа, а можно установить самостоятельно число колонок, ширину колонок и промежуток между ними. В поле применить выбираются границы колонок, можно выбрать ко всему документу, До конца документа. Линия между колонками устанавливается после установки отметки **Разделитель**. Как будет располагаться текст можно увидеть в окне **Образец** (рис. 36).

Изменять ширину колонок и расстояние между ними можно с помощью горизонтальной линейки.

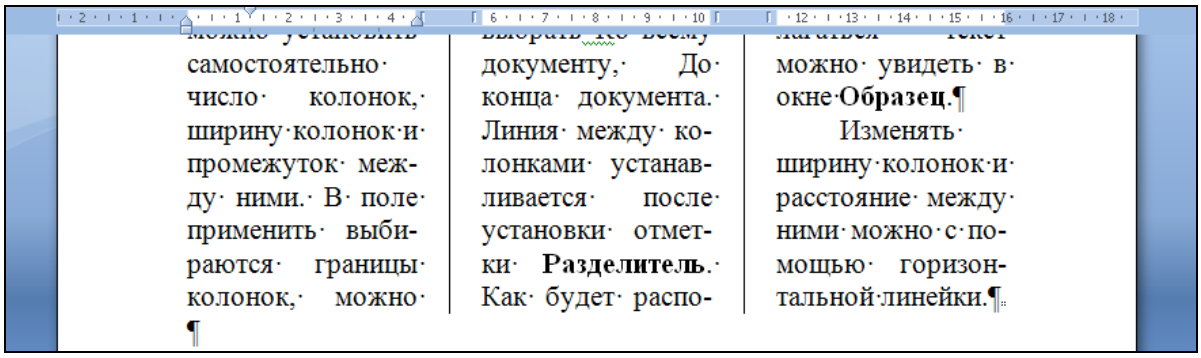


Рис. 36. Колонки

Работа с графическими объектами

Microsoft Word позволяет работать с графическими объектами, созданными как в других программах, так и со встроенными фигурами. На вкладке Вставка располагается группа Иллюстрации, (рис. 37), после щелчка по выбранному объекту открывается диалоговое окно вставки объекта или боковая панель. После добавления объекта в документ, при его выделении открывается контекстная вкладка, где можно установить необходимые параметры объекта.

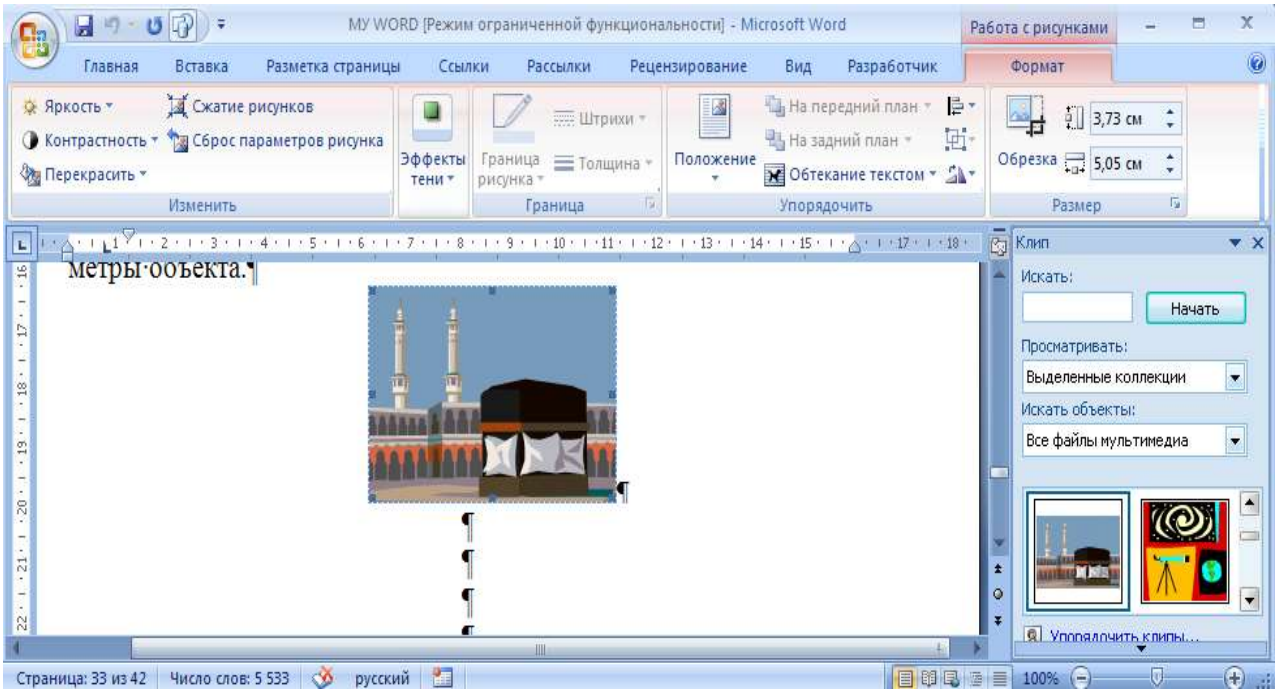


Рис. 37. Вставка клипа в документ

Для удаления рисунка его следует выделить и нажать клавишу **Delete**.

Создание таблиц

В Microsoft Office Word 2007 можно вставить таблицу, выбрав одну из предварительно отформатированных таблиц, заполненных примерными данными, или указав нужное количество строк и столбцов в таблице. Таблица может быть вставлена непосредственно в документ или вложена в другую таблицу, что позволяет создавать сложные таблицы.

Вставка таблицы с использованием шаблона

Шаблоны таблиц содержат примерные данные, которые помогают оценить вид таблицы после того, как в нее будут добавлены данные. Для того чтобы вставить таблицу из коллекции отформатированных таблиц, которая позволяет наглядно оценить вид таблицы после заполнения данными, нужно щелкнуть место, куда вставить новую таблицу, на вкладке **Вставка** в группе **Таблицы** нажать кнопку **Таблица** 

, выделить пункт **Экспресс-таблицы**, выбрать нужный шаблон и заполнить шаблон своими данными, заменив данные шаблона (рис. 38).

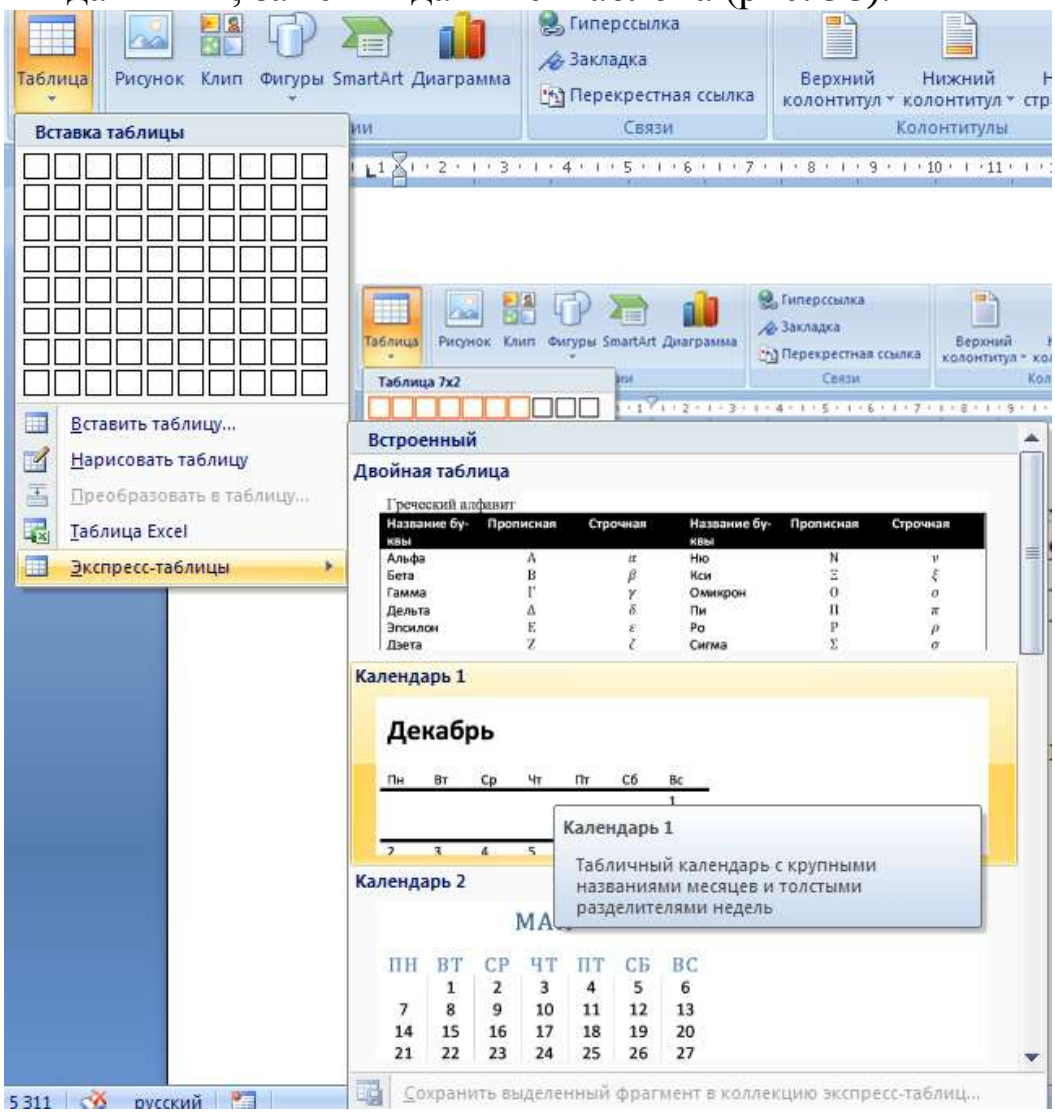
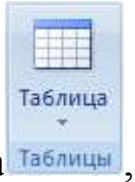


Рис. 38. Вставка таблицы, используя шаблон

Вставка таблицы с использованием меню Таблица

Щелкнуть место, где будет располагаться таблица, перейти на вкладку **Вставка**, в группе **Таблица** нажать кнопку **Таблица**



, а затем в области **Вставить таблицу** путем перетаскивания выберите нужное число строк и столбцов (рис. 39).

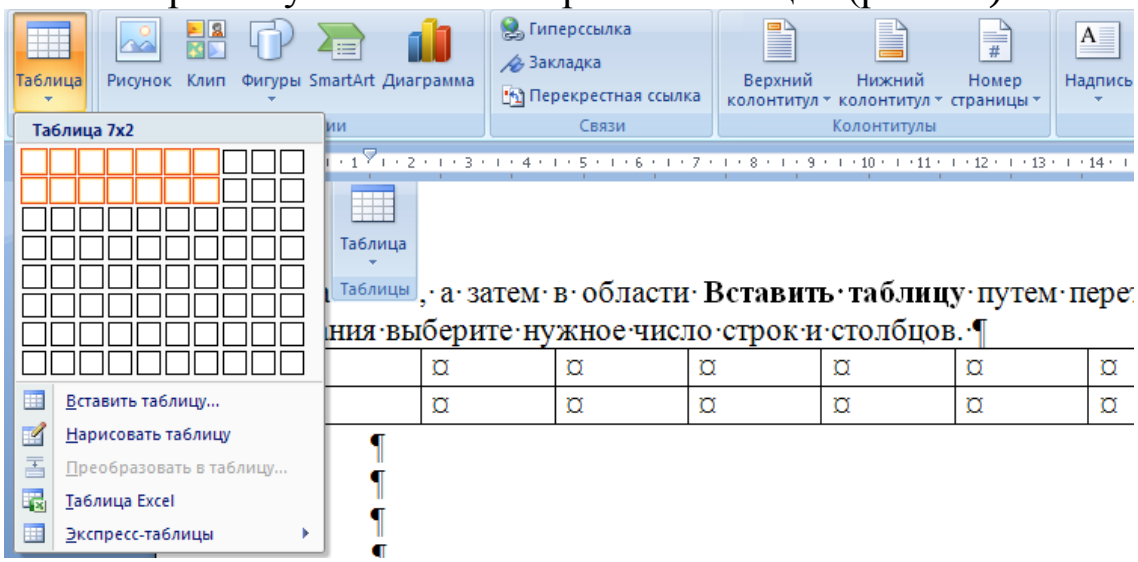


Рис. 39. Вставка таблицы, используя меню Таблица

Вставка таблицы помощью команды Вставить таблицу

Данный способ позволяет перед вставкой таблицы указать ее размеры и формат (рис. 40).

1. Щелкнуть в документе место, куда требуется вставить новую таблицу.
2. На вкладке **Вставка** в группе **Таблицы** щелкнуть **Таблица** и затем выберите **Вставить таблицу**.
3. В области **Размер таблицы** введите количество столбцов и строк.
4. В области **Автоподбор ширины столбцов** выберите параметры подстройки размера таблицы.

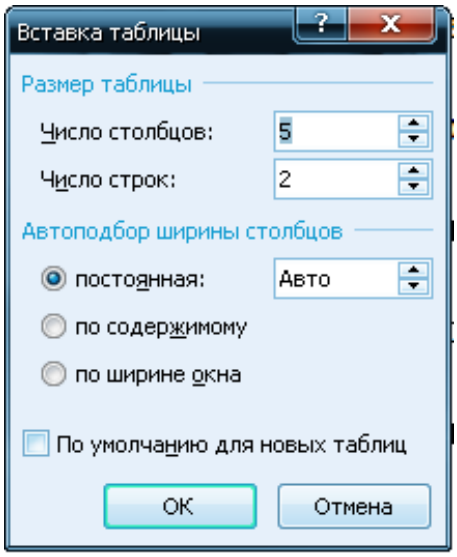


Рис. 40. Вставка таблицы

Рисование таблицы

Сложную таблицу, например таблицу с ячейками разной высоты или с меняющимся числом столбцов в строке, можно нарисовать. Для этого на вкладке **Вставка** в группе **Таблицы** щелкнуть **Таблица,** а затем **Нарисовать таблицу**. После этого указатель мыши примет вид карандаша. Чтобы определить внешние границы таблицы, необходимо нарисовать прямоугольник. Затем внутри этого прямоугольника нарисуйте линии столбцов и строк (рис. 41).

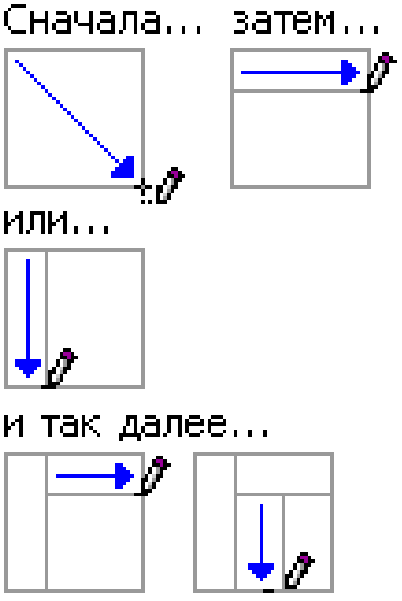


Рис. 41. Рисование таблицы

Чтобы стереть линию или блок линий, на вкладке **Конструктор** контекстных инструментов **Работа с таблицами** в

группе **Нарисовать границы** выбрать **Ластик**, далее щелкнуть линию, которую требуется стереть. После того как таблица нарисована, щелкнуть в ячейку таблицы и начать заполнение.

Преобразование текста в таблицу

В Microsoft Office Word можно преобразовать ранее созданный текст в таблицу, для этого необходимо вставить знаки разделителей (запятая или знак табуляции) в тех местах, где текст должен быть разбит по столбцам. Используя знак абзаца, указать, где должны начинаться новые строки. Выделить текст, на вкладке **Вставка** в группе **Таблицы** выберите пункт **Таблица**, а затем выберите команду **Преобразовать в таблицу**. В диалоговом окне **Преобразовать в таблицу** указать знак разделителя (рис. 42).

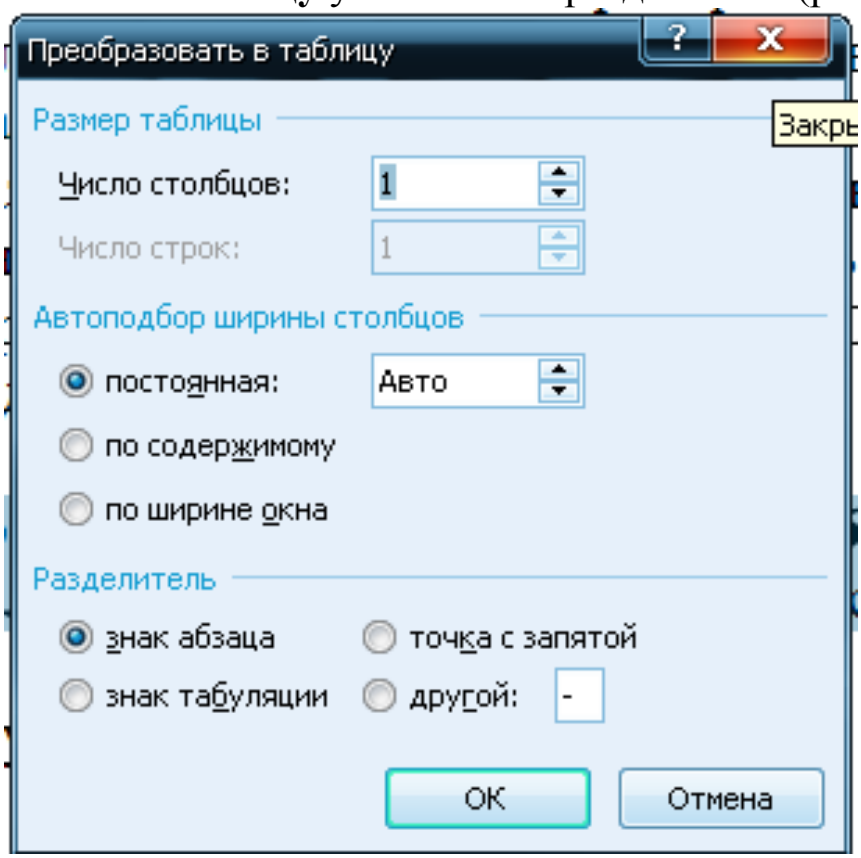


Рис. 42. Преобразование текста в таблицу

Каждая ячейка таблицы рассматривается как абзац. Форматировать текст в ячейке можно используя вкладку **Главная** на **Ленте**, мини-панель **Форматирование** или **Контекстное меню**.

4. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Ознакомиться с теоретическими сведениями.
2. Получить задание у преподавателя.
3. Выполнить задание на ПК.
4. Показать преподавателю.
5. Ответить на вопросы.

5. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Создание нового документа в Microsoft Word, варианты сохранения.
2. Особенности пользовательского интерфейса.
3. Каким образом осуществляется ввод и редактирование текста?
4. Вставка символа и формул.
5. Выбор темы для всего документа и редактирование стиля части документа.
6. Как изменить параметры шрифта и параметры абзаца?
7. Вставка нумерации страниц и работа с колонтитулами.
8. Особенности работы с графическими объектами.
9. Способы создания таблиц.

Лабораторная работа № 4
Табличный процессор MS EXCEL.Создание таблиц и диаграмм. Статистическая обработка данных

1. ЦЕЛЬ РАБОТЫ

Целью работы является ознакомление студентов с возможностями и основными объектами приложения Microsoft Excel; освоение приемов занесения данных и формул в таблицы и способы построения диаграмм.

2. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Табличный процессор Microsoft Excel и текстовый редактор Microsoft Word имеют много общего, речь идет о выполнении основных действий (запуск программы, открытие, создание, сохранение документов).

С помощью MS Excel можно:

- создавать, редактировать, выводить на печать отредактированные и оформленные таблицы;
- производить вычисления по различным формулам;
- создавать списки, в столбцах которых содержатся однотипные данные;
- производить различные операции не только над числами, но и над текстом;
- создавать разнообразные диаграммы;
- создание рисунков и схем;
- автоматизация сложных задач.

Помимо этого таблицы Microsoft Excel могут быть встроены во многие документы, в том числе и в Microsoft Word.

Объекты документа Excel

Электронная таблица, так же как и обычная таблица, представляет собой набор числовых и текстовых данных, расположенных в ячейках. Документ, созданный в MS Excel, называется **Рабочей книгой** или просто **книгой** (Workbook). В состав книги входят листы электронных таблиц – **Рабочие листы** (Workbook). Новая книга обычно содержит 3 пустых листа (рис. 1), листы можно добавлять (для этого достаточно щелкнуть по пустой вкладке за листами, или правой кнопкой мыши по имени любого

листа и во всплывающем меню выбрать **Вставить**, а в появившемся диалоговом окне выбрать **Лист**), переименовывать (щелкнуть правой кнопкой мыши и выбрать **Переименовать**), изменять последовательность расположения (достаточно просто перетащить в нужное место, или через диалоговое окно **Переместить** или скопировать указать его местоположение).

Лист книги состоит из строк (1048576) и столбцов (16384).

На экране ячейки листа таблицы разделяются линиями сетки. Строки обозначаются цифрами от 1 до 1048576, а столбцы буквами латинского алфавита A, B, C,…Z, AA, AB, …AZ, BA, BB, …XFD.

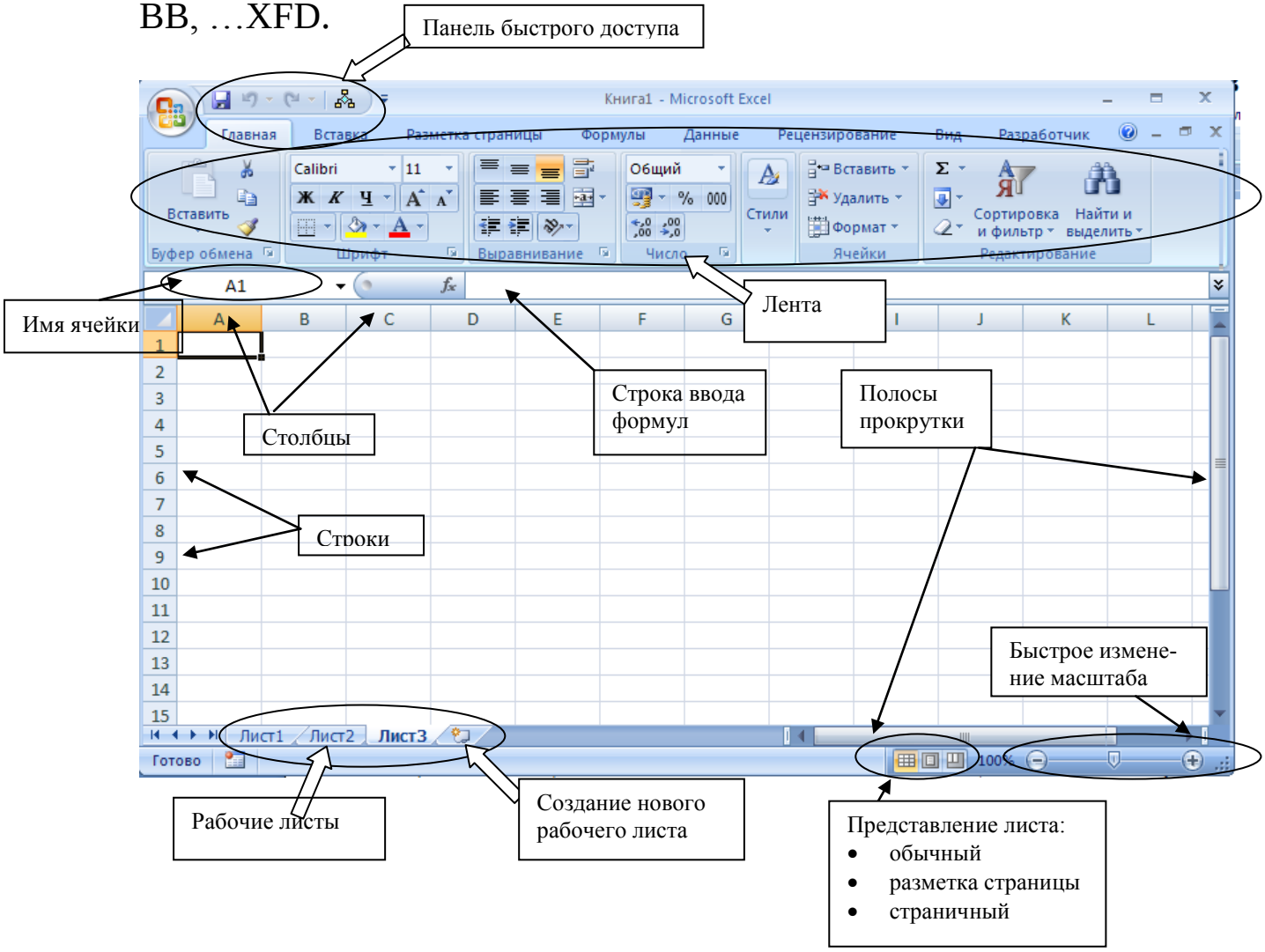
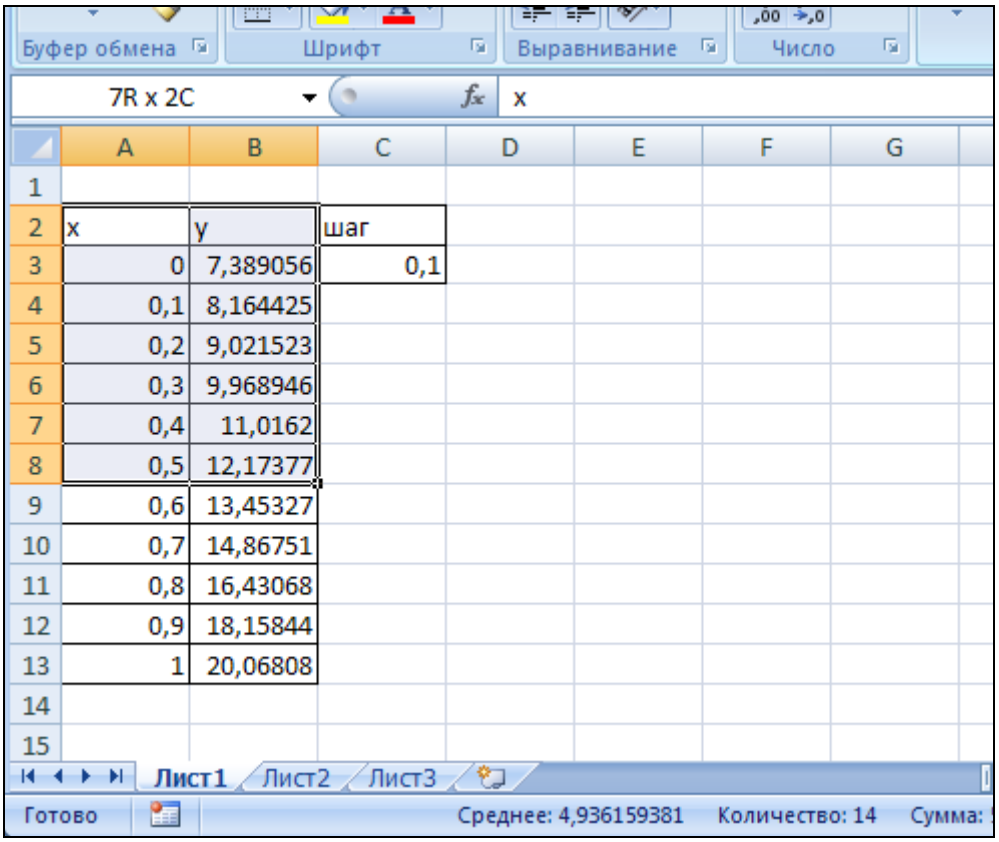


Рис. 1. Рабочий лист Microsoft Excel

Диапазон – это группа ячеек (рис. 2), чтобы задать адрес диапазона, нужно указать адреса его левой верхней и правой нижней ячеек, разделив их двоеточием.



	A	B	C	D	E	F	G
1							
2	x	y	шаг				
3	0	7,389056	0,1				
4	0,1	8,164425					
5	0,2	9,021523					
6	0,3	9,968946					
7	0,4	11,0162					
8	0,5	12,17377					
9	0,6	13,45327					
10	0,7	14,86751					
11	0,8	16,43068					
12	0,9	18,15844					
13	1	20,06808					
14							
15							

Рис. 2. Диапазон ячеек

Например, C2 – диапазон, состоящий из одной ячейки, A1:D4 – диапазон из 16 ячеек, расположенных в четырех строках и столбцах.

Для того чтобы производить какие-либо действия над диапазоном его необходимо выделить. В Microsoft Excel можно выделить строку или столбец целиком, щелкнув на номере строки или имени столбца, соответственно, чтобы обратиться к диапазону ячеек – щелкнуть одну левую верхнюю ячейку и перетащить указатель, выделяя диапазон, либо ввести имя диапазона в поле **Имя**. Для того чтобы выделить *несмежные* диапазоны (ячейки или диапазоны расположенные не рядом друг с другом) – нажать клавишу Ctrl и указать диапазоны, или в поле **Имя** ввести имена диапазонов, отделяя их запятыми.

Наиболее часто встречающиеся операции при работе с диапазоном ячеек – это копирование и перемещение. При этом различают:

1) копирование содержимого одной ячейки в другую ячейку;

2) копирование содержимого ячейки в диапазон, при этом содержимое исходной ячейки копируется в каждую ячейку диапазона.

3) копирование содержимого диапазона в другой диапазон, при этом оба диапазона должны иметь одинаковые размеры;

4) перемещение содержимого диапазона в другой диапазон.

При копировании содержимое исходного диапазона сохраняется, а при перемещении – удаляется.

В общем случае эти две процедуры сводятся к двум этапам:

1) выделить необходимый диапазон и скопировать его в буфер обмена (при перемещении диапазон не копируется, а вырезается);

2) поместить курсор в ячейку (диапазон ячеек) и вставить данные из буфера обмена.

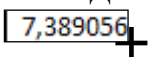
Для упрощения Microsoft Excel предлагает несколько вариантов выполнения операций копирования и перемещения.

1. Копирование с помощью ленточных команд. **Главная ⇨ Буфер обмена ⇨ Копировать**, а затем **Главная ⇨ Буфер обмена ⇨ Вставить.**

2. Копирование с помощью команд контекстного меню. Щелкнуть на диапазоне правой кнопкой мыши, а затем в контекстном меню выбрать команду **Копировать** (или **Вырезать**). Для того, чтобы вставить скопированный диапазон, щелкнуть правой кнопкой мыши и выбрать в контекстном меню команду **Вставить.**

3. Копирование с помощью клавиатуры. <Ctrl+C> – Копирует выбранные ячейки в буферы Windows и Office. <Ctrl+X> – Вырезает выбранные ячейки в буферы Windows и Office. <Ctrl+V> – Вставляет содержимое буфера обмена в выбранную ячейку или диапазон ячеек.

4. Копирование и перемещение с помощью перетаскивания. Выделить ячейку (диапазон ячеек), переместить указатель мыши к границе ячейки (диапазона), когда указатель примет вид стрелки нажать клавишу Ctrl, (к указателю добавится знак +), перетащить ячейку(диапазон) в нужное место, оставляя зажатой клавишу Ctrl. Для перемещения выполнить все действия без нажатия клавиши Ctrl.

5. Копирование в соседние ячейки. **Главная ⇨ Редактирование ⇨ Заполнить ⇨ Вниз. Главная ⇨ Редактирование ⇨ Заполнить ⇨ Вверх. Главная ⇨ Редактирование ⇨ Заполнить ⇨ Влево. Главная ⇨ Редактирование ⇨ Заполнить ⇨ Вправо.** Заполняется выбранный диапазон снизу сверху, слева, справа соответственно. При копировании в соседние ячейки можно воспользоваться маркером заполнения 

: подвести указатель к нижнему правому углу, щелкнуть и растянуть на необходимый диапазон.

6. Копирование диапазона на соседние листы. Все вышеописанные способы можно применять и для копирования выбранного диапазона на соседние листы или в другую рабочую книгу, для этого перед выбором операции **Вставить**, необходимо перейти на нужный рабочий лист в рабочей книге.

Форматирование диапазона

Ввод данных

Ниже панели инструментов расположена строка формул (рисунок 1). Содержимое активной (выделенной в данный момент) ячейки Excel всегда появляется в строке формул. В процессе ввода или редактирования данных в ячейке, в строке формул появляются три кнопки:

- кнопка отмены (крестик );
- кнопка ввода (галочка );
- кнопка изменения формулы (знак функции ).

Информацию можно вводить как непосредственно в ячейку, так и в строку формул.

В ячейке могут находиться данные одного из трех типов:

- числовое значение,
- текст,
- формула.

На рабочем листе Excel могут находиться также графики, диаграммы, рисунки, изображения, кнопки и другие объекты, они располагаются на невидимом слое листа, расположенным поверх листа (графический уровень).

Для облегчения ввода набора числовых значений или текстовых элементов в Microsoft Excel существует специальная воз-

можность **Автозаполнение**. Процесс заполнения диапазона ячеек данным способом рассмотрен ранее. Для того, чтобы заполнить последовательность от 1 до 15 с шагом 1, нужно в начальную ячейку ввести значение первого элемента, в ячейку рядом – второго, выделить ячейки, подвести указатель к правому нижнему углу (при этом появится маркер автозаполнения) и растянуть на необходимый диапазон, при этом последовательность примет вид, показанный на рисунке 3.

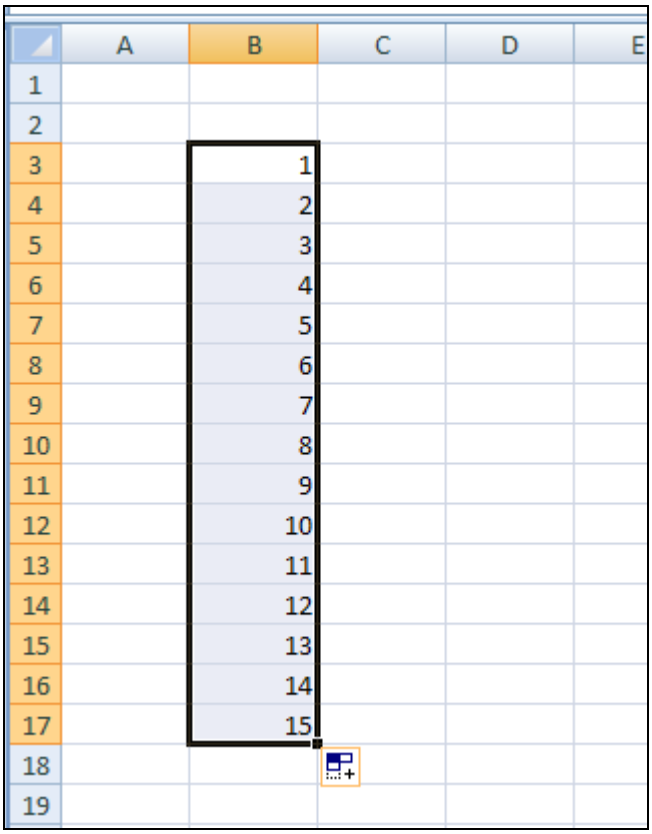


Рис. 3. Автозаполнение ячеек

При вводе слишком длинного текста в ячейку иногда необходимо отображение текста в нескольких строках, чтобы перейти на следующую строку ячейки можно нажать комбинацию клавиш <Alt+Enter>, или вызвать диалоговое окно **Формат ячеек** и установить галочку напротив **Переносить по словам**, как показано на рисунке 4.

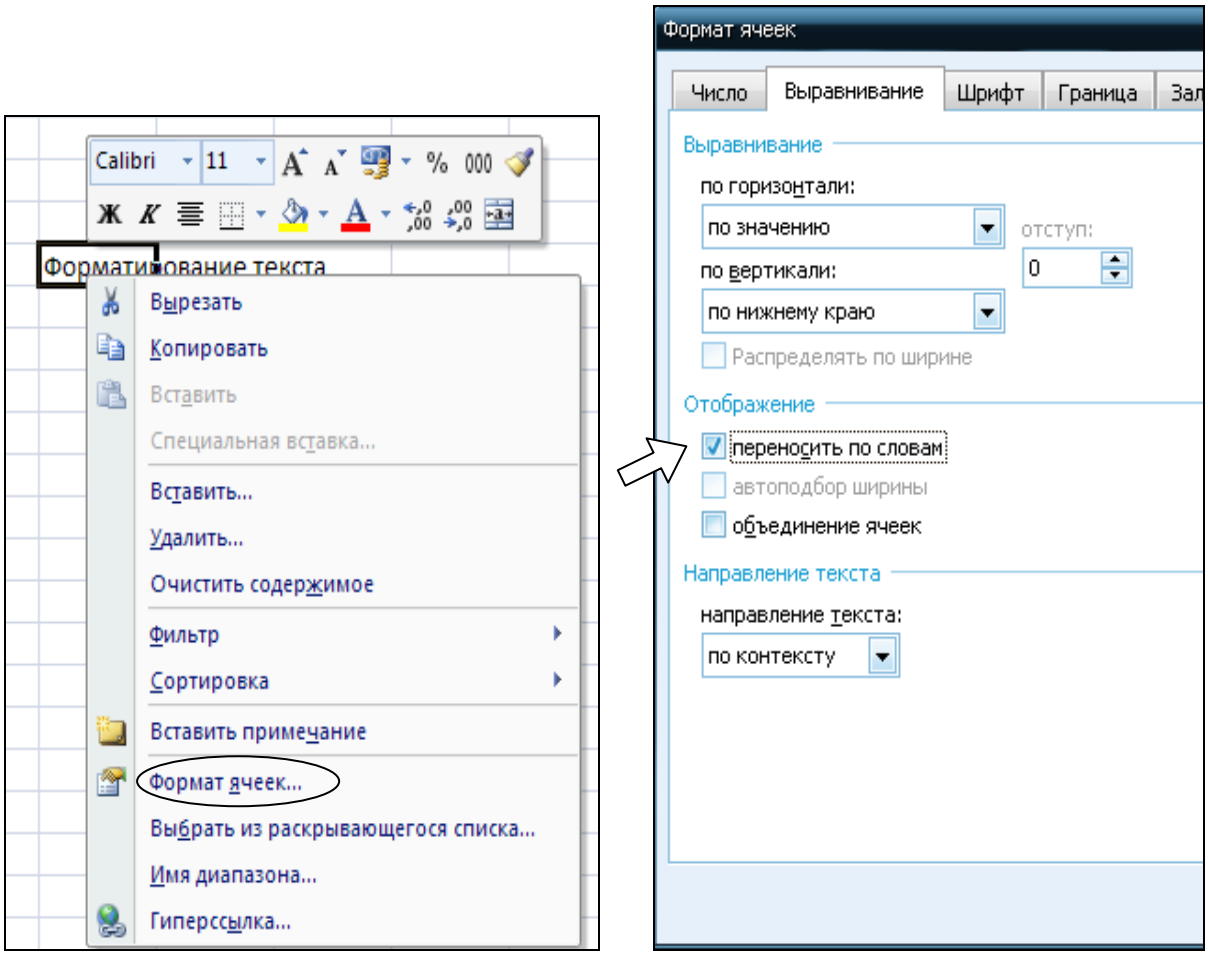


Рис. 4. Создание дополнительных строк ячейки

Для того, чтобы ввести текущую дату и время в ячейку достаточно нажать <Ctrl+Shift+; (точка с запятой)> и < Ctrl+Shift+: (двоеточие)> соответственно.

Форматирование ячеек

Форматирование ячеек – это изменение внешнего вида ячеек, без изменения значений.

После ввода числовые значения, как правило, никак не отформатированы, в таблице 1 показаны числа в форматированном и неформатированном виде.

Таблица 1

Форматы данных

Неформатированные	Форматированные	Формат
23436	23436	Общий
23436	23436,00	Числовой
23436	23 436,00р.	Денежный
23436	23 436,00р.	Финансовый
06.01.2012	06 января 2012 г.	Дата
12:45	12:45 PM	Время
0,234	23,40%	Процентный
0,75	3/4	Дробный
326998367	3,27E+08	Экспоненциальный
856,563 руб.	856,563 руб.	Текстовый

Некоторые операции форматирования Microsoft Excel выполняет самостоятельно, именно поэтому в табл. 1 дата 06.01.2012 в неформатированном виде показана как 06.01.2012, а не 40914.

Даты в Microsoft Excel хранятся в виде последовательности (порядковых номеров), их можно складывать, вычитать и использовать в других выражениях. Например, день 1 января 1900 г. имеет номер 1, а 1 января 2012 г. — 40 909, т. к. интервал меду этими датами составляет 40 909 дней. Временные значения хранится в виде дробной части этого числа, поскольку они рассматриваются как доли суток. Microsoft Excel поддерживает 2 системы дат (рис. 5), которая выбирается автоматически.

Система дат	Первая дата	Последняя дата
1900	1 января 1900 г. (значение 1)	31 декабря 9999 г. (значение 2958465)
1904	2 января 1904 г. (значение 1)	31 декабря 9999 г. (значение 2957003)

Рис. 5. Системы дат

Форматировать данные в ячейках можно:

1) форматирование ячеек с помощью **Ленты** (вкладка **Главная**, группа **Число**) (рис. 6);

2) комбинацией клавиш (рис. 7);

3) с помощью диалогового окна **Формат ячеек** (рис. 8).

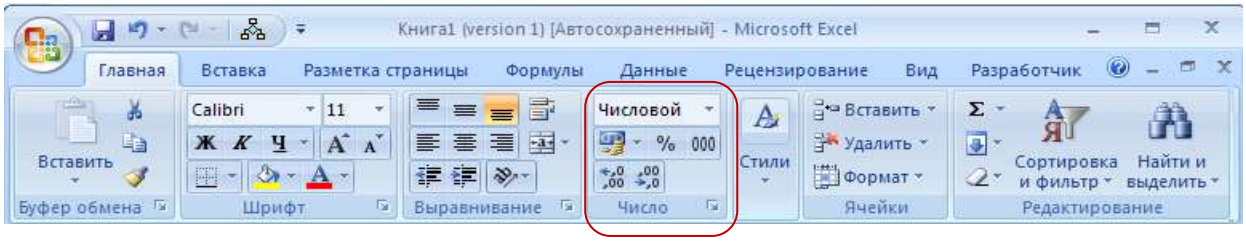


Рис. 6. Форматирование ячеек с помощью **Ленты**

Комбинация клавиш	Применяемое форматирование
<Ctrl+Shift+~>	Общий числовой формат (т.е. неотформатированные значения)
<Ctrl+Shift+$>	Денежный формат с двумя знаками после запятой (отрицательные числа записываются в скобках)
<Ctrl+Shift+%>	Процентный формат без десятичных разрядов после запятой
<Ctrl+Shift+^>	Экспоненциальный формат чисел с двумя знаками после запятой
<Ctrl+Shift+#>	Формат дат с указанием дня, месяца и года
<Ctrl+Shift+@>	Формат дат с отображением часов и минут
<Ctrl+Shift+!>	Числовой формат с двумя знаками после запятой, разделителем разрядов и дефисом для отрицательных значений

Рис. 7. Форматирование с помощью комбинации клавиш

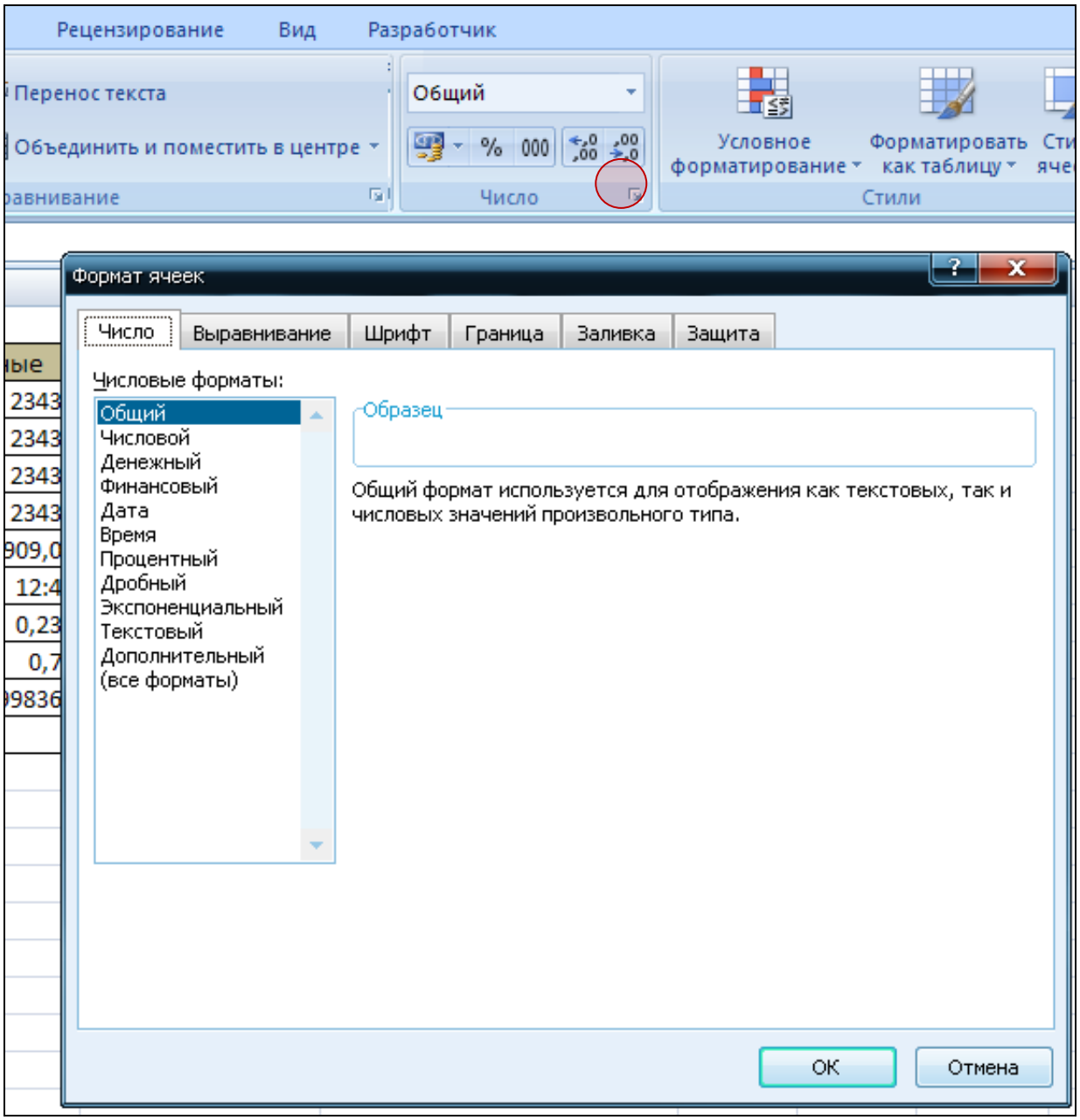


Рис. 8. Форматирование ячеек
с помощью диалогового окна **Формат ячеек**

2.4. Ввод и использование формул

В Microsoft Excel под понятием формула скрывается математическое выражение, с помощью которого вычисляется значение какой-либо ячейки. В формулах можно использовать как числовые значения, так и адреса ячеек, значения которых используются в формуле вместо адреса (рис. 9).

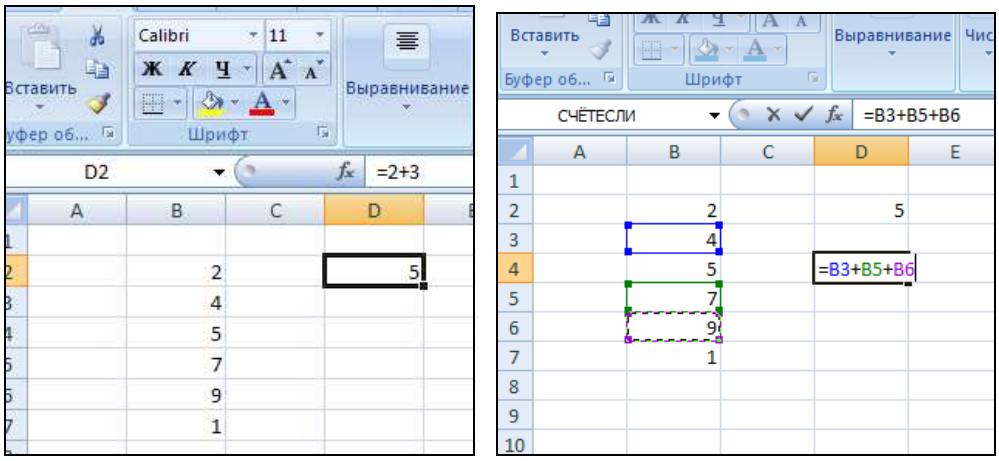


Рис. 9. Ввод формул

Как показано на рисунке 8, любая формула начинается со знака =, если в ячейке, на которую присутствует ссылка в формуле, нет записи, то будет вставлено нулевое значение. При указании имен ячеек в формуле можно вместо ввода с клавиатуры, указывать их щелчком мыши, что помогает избежать ошибок. В формуле используются знаки математических операций (-, +, *, /), знаки соотношений (<, >, <=, >=, <>), скобки и встроенные функции.

Использование встроенных функций существенно расширяет возможности Microsoft Excel, (рис. 10). Функция предопределяет какую-либо встроенную формулу.

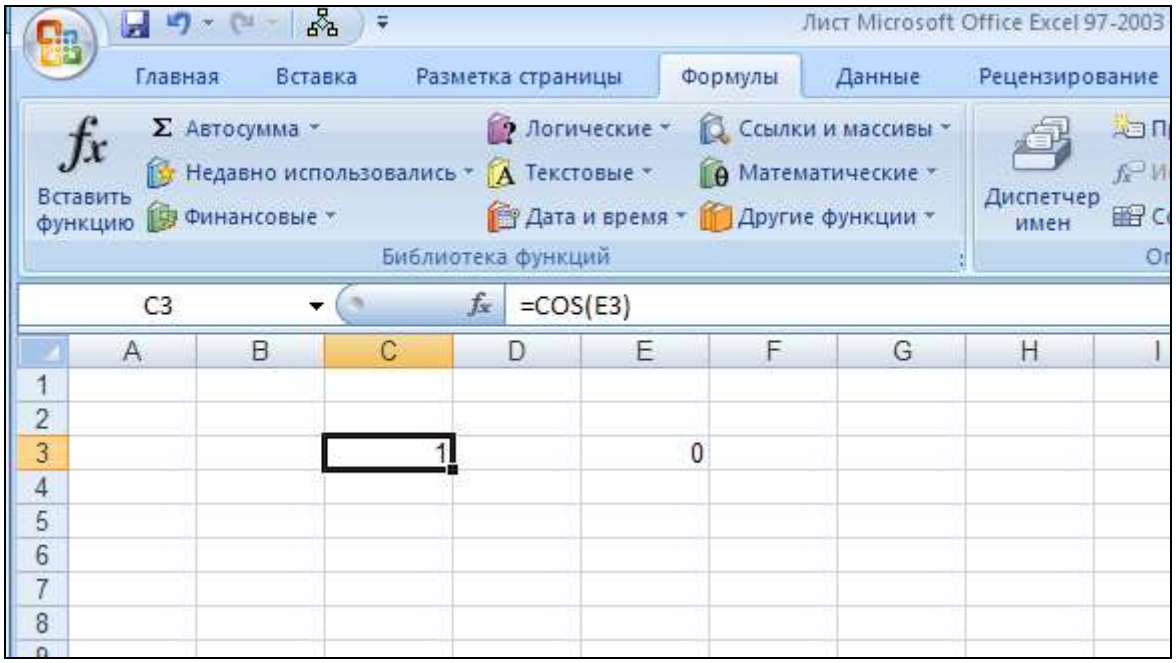


Рис. 10. Формулы, встроенные в Microsoft Excel

Для ввода той или иной функции можно воспользоваться несколькими способами:

а) в строке формул после знака = или, например +, ввести первую букву имени формулы и Excel предложит список функций, начинающихся с этой буквы;

б) на вкладке **Формулы** найти соответствующую функцию (рис. 9);

в) вызвать мастер функций, нажатием **Вставить функцию**, или щелкнуть по пиктограмме f_x, (рис. 11).

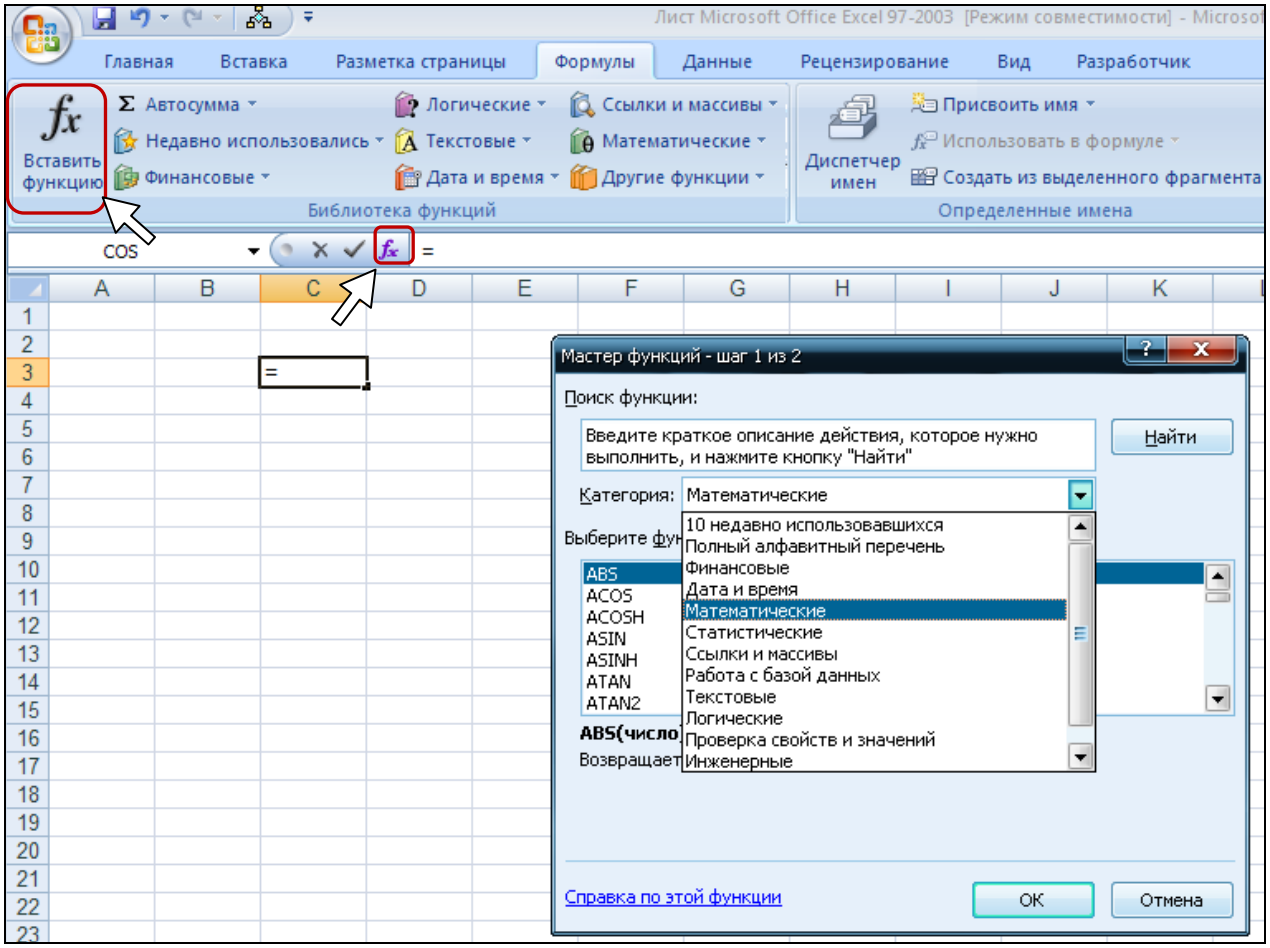


Рис. 11. Работа с Мастером функций

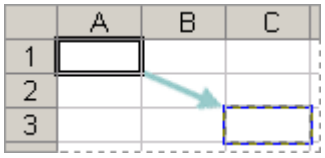
По умолчанию Excel использует стиль ссылок на ячейки A1, непосредственно указывая столбец и строку ячейки.

Относительная ссылка основана на относительной позиции ячейки, при изменении позиции ячейки изменяется и ссылка, например: A1=B2+C4, при копировании или заполнении ссылка корректируется: B3=C4+D6.

Абсолютная ссылка указывает точный адрес ячейки в формуле независимо от положения ячейки в формуле, например: A1=$B$2+C4, при копировании или заполнении ссылка на данную ячейку не корректируется: B3=$B$2+D6.

Таблица 2

Абсолютная, относительная и смешанная адресация

Формула	Ссылка	Итог
	$A$1 – Абсолютная ссылка	$A$1
	A1– Относительная ссылка	C3
	$A1 – Смешанная ссылка (абсолютный столбец, относительная строка	$A3
	A$1– Смешанная ссылка (относительный столбец, абсолютная строка	C$1

В Microsoft Excel можно посмотреть, какие ячейки связаны с данной, для этого на вкладке **Формулы** нажать кнопку **Влияющие ячейки** (рис. 12).

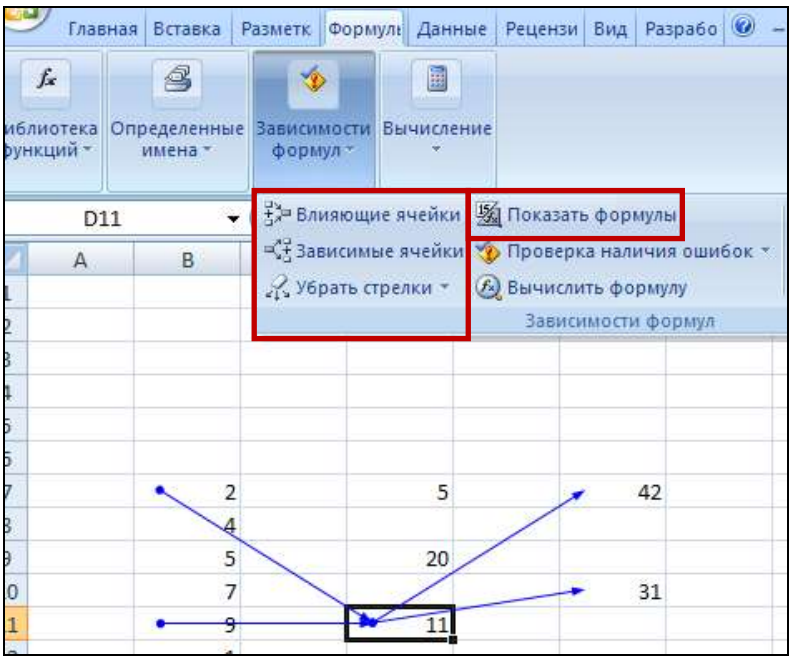


Рис. 12. Отображение связей

Если нажать кнопку **Зависимые ячейки**, отобразятся связи с ячейками, для вычисления которых использовалась данная ячейка. Чтобы убрать связи – щелкнуть по кнопке **Убрать связи**,

при этом можно убрать как все связи, так и по выбору, только влияющие, или только зависимые ячейки. Для того, чтобы отображалась формула в ячейке, нажать **Показать формулы**, при повторном нажатии, формулы отображаться не будут (рис. 11).

Нередко требуется, чтобы в формуле на данном листе использовались данные с других рабочих листов, для этого можно прописать в строке формул имена листов и адреса ячеек, или указать мышкой соответствующие ячейки, на рабочих листах (рис. 13).

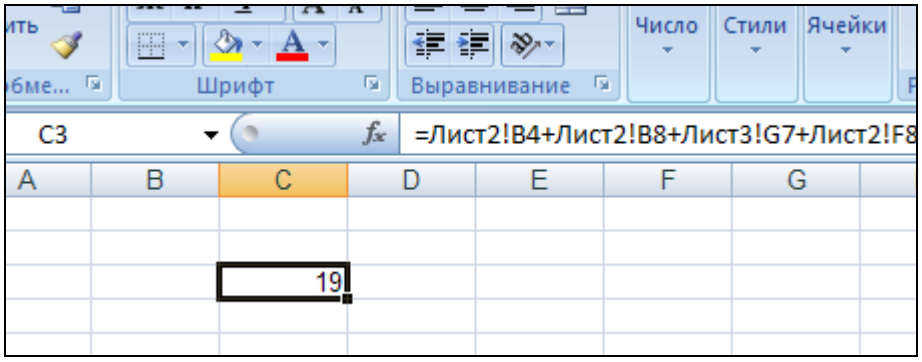
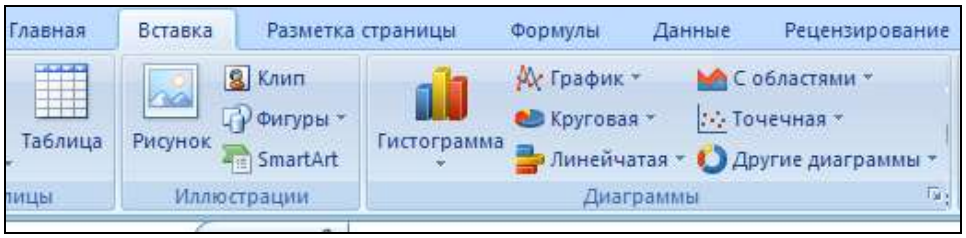


Рис. 13. Ссылка на ячейки из разных рабочих листов

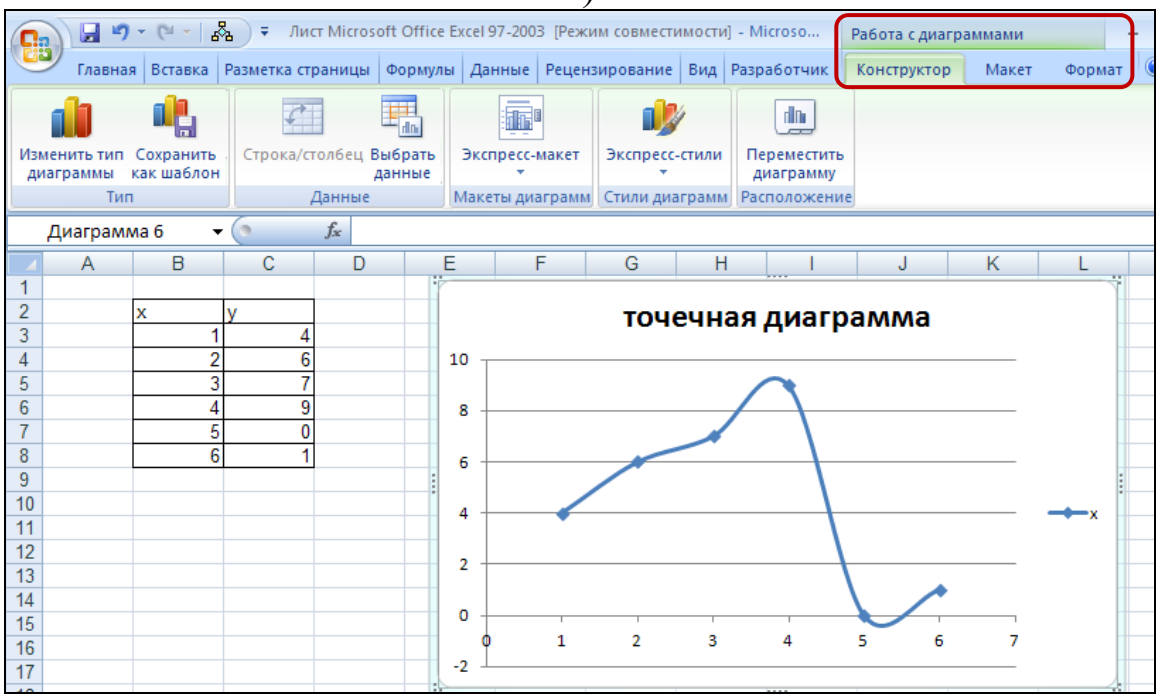
Можно также ссылаться на ячейки, расположенные в другой рабочей книге, для этого в строке формул, помимо адреса ячейки необходимо прописать путь к рабочей книге, здесь проще указывать ячейки мышкой в рабочей книге.

Построение диаграмм

Microsoft Excel, помимо различных вычислений, позволяет наглядно отобразить изменение какого-либо показателя в таблице или просто содержащиеся в ней данные. Для этого в Excel целая группа инструментов, расположенных на вкладке **Вставка** в разделе **Диаграммы** (рис 14 а), в свою очередь после выбора типа диаграммы появляется контекстная вкладка **Работа** с диаграммами (рис 14 б).



а)



б)

Рис. 14. Построение и работа с диаграммами

Для построения диаграммы, показанной на рисунке 14 б), нужно.

1. Выделить диапазон ячеек, в которых содержатся необходимые данные, в данном случае B3:C8.

2. Перейти на вкладку **Вставка** и в группе **Диаграммы** выбрать **Точечная**, и во всплывающем меню выбрать **Точечная** с гладкими кривыми и маркерами.

3. Сразу после выбора диаграмма будет построена. **Далее**, выбирая вкладки **Конструктор**, **Макет**, **Формат**, можно изменять или добавлять данные, легенды и т. п.

3. СОДЕРЖАНИЕ И ПОРЯДОК ВЫПОЛНЕНИЯРАБОТЫ

Создание электронной таблицы и заполнение формулами

Задание 1

1. Создайте таблицу следующего образца:

	Март		
Наименование товара			
Количество			
Закупочная цена			
Процентная надбавка			
Розничная цена			
Сумма закупки			
Сумма продажи			
Прибыль			

2. Заполните исходную таблицу таким образом, чтобы в столбцах помимо наименования товара были соответствующие цифры а в ячейках **Розничная цена**, **Сумма закупки**, **Сумма продажи**, **Прибыль** – соответствующие формулы

Розничная цена = Закуп. Цена × Процент. надбавка

Сумма закупки = Закуп. Цена × Количество

Сумма продажи = Закуп. Цена × Количество

Прибыль = Сумма продажи - Сумма закупки

Помните, что все данные в ячейки заносятся только через ссылки на ячейку, в которой они находятся, результаты покажите преподавателю.

Задание 2

1. Составьте электронную таблицу следующего вида:

	A	B	C	D	E	F	G	H
1	№ п/п	Наименование материала	Цена	Количество			Сумма на конец месяца	Доля в общем объеме
2				На начало месяца	На конец месяца	Израсходовано		
3	1							
4	2							
5	3							
6	4							
7	5							
8	6							
9	7							
10	8							
11	9							
12	10							
13	11							
14	12							
15	Итого							
16	Средняя сумма расхода							
17	Минимальная доля в общем объеме							

2. Заполните исходную таблицу.

Расчеты в таблице производятся по следующим формулам Excel со ссылками на ячейки:

Израсходовано = Кол. На конец месяца – Кол. На начало месяца;

Сумма на конец месяца = Цена × Кол. На конец месяца;

Доля в общем объеме = (*Сумма на конец месяца – Итого Сумма на конец месяца) × 100;*

Результаты покажите преподавателю.

Задание 3

Составьте электронную таблицу, вычисляющую *n*-й член и сумму арифметической прогрессии по формулам (1), (2)

$$\grave{a}_n = a_1 + d(n-1); \quad (1)$$

$$S_n = (a_1 + a_n)\cdot\frac{n}{2}, \quad (2)$$

где a_1 — первый член прогрессии, d — разность арифметический прогрессии.

1. В ячейку A1 и введите заголовок таблицы "Вычисление *n*-го члена и суммы арифметической прогрессии". Заголовок будет размещен в одну строчку и займет несколько ячеек правее A1.

	A	B	C	D
1	Вычисление *n*-го члена и суммы арифметической прогрессии			
2	d	n	a_n	S_n
3	0,725	1	-2	-2
4	0,725	2	-1,275	-3,275
5	0,725	3	-0,55	-3,825
6	0,725	4	0,175	-3,65
7	0,725	5	0.9	2.75
8	0,725	6	1,625	-1,125
9	0,725	7	2,35	1,225
10	0,725	8	3.075	4,3
11	0,725	9	3,8	8,1
12	0,725	10	4.525	12,625

2. В ячейку A2 введите "*d*", в ячейку B2 — "n", в C2 — "a_n", в D2 — "S_n". Для набора нижних индексов воспользуйтесь командой "*Формат*" → "*Ячейки*" ..., выберите вкладку "*Шрифт*" и включите нижний индекс. Выровняйте по центру и примените полужирный шрифт.

3. В ячейку A3 введите величину разности арифметической прогрессии.

4. Следующий столбик заполните последовательностью чисел от 1 до 10.

5. Введите в ячейку С3 значение первого члена арифметической прогрессии.

6. Выделите ячейку С4 и наберите в ней формулу =С3+А3, зафиксируйте ее нажатием Enter, в ячейке окажется результат вычисления по формуле, а в **Строке формул** – сама формула.

7. Заполните формулой, "протащив" маркер заполнения вниз, ряд ячеек ниже С4. Ссылки в формуле изменились относительно смещения формулы.

8. Аналогично введите в ячейку D3 формулу =($C$3+C3)**B3*/2 для подсчета суммы *n* первых членов арифметической прогрессии, заполните формулами нижние ячейки.

9. Выделите ячейку А1 и примените полужирное начертание символов к содержимому ячейки.

Задание для защиты: редактирование таблиц

Задание 1

1. Откройте первую таблицу, внесите следующие изменения:

- уберите строку *Процентная надбавка*;
- вставьте месяцы *январь* и *февраль*;
- скопируйте в них нужные формулы и заполните (установите *формат денежный*);
- вставьте перед первым столбцом столбец с нумерацией;
- вставьте строку со словами "*Расчет рентабельности*" и отформатируйте ее полужирным шрифтом изменив цвет шрифта, строку "Наименование товара" залить цветом, столбец от слов "*Количество*" до "*Процентная надбавка*" стилем курсив; цифры, "*Розничная цена*" до "*Прибыль*" – стилем на ваш вкус.

2. Последовательность действий покажите преподавателю.

Задание 2

1. Откройте вторую таблицу.
2. Вставьте перед столбиком "*Сумма на конец месяца*" столбец с "*Процентным количеством израсходованных материалов*" от общего количества (использование абсолютной ссылки).
3. Отформатируйте текст различными стилями по вашему усмотрению.
4. Выровняйте содержимое диапазонов с цифрами по центру.
5. Сохраните результат и покажите преподавателю.

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

4.1. Какие виды работ позволяет выполнить табличный процессор Excel?

4.2. Что такое диапазон данных?

4.3. Как выделить несмежные диапазоны для совместного их форматирования?

4.4. Какие виды диаграмм можно построить в Excel?

4.5. Как Excel работает с датами?

4.6. Приведите примеры использования абсолютной и относительной адресации.

4.7. Какое расширение имеют файлы, созданные с помощью Excel?

Лабораторная работа № 5
Работа с макросами в табличном процессоре MS EXCEL

1. ЦЕЛЬ РАБОТЫ

Целью работы является приобретение практических навыков по созданию макросов в виде таблиц в MS EXCEL.

2. ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Некоторые действия в Microsoft Excel могут носить повторяющийся характер. Например, нам нужно выполнить одинаковыми заголовками целый ряд рабочих документов или одним и тем же способом отформатировал все заголовки. Вместо того чтобы каждый раз вводить одни и те же данные или выполнять команды форматирования, вы можете создать макрос и затем, запуская его, выполнять такие операции автоматически. За каждой кнопкой панели инструментов закреплена небольшая программа – макрос.

Макрос представляет собой последовательность макрокоманд и макрофункций, написанных на языке Visual Basic.

Создание макросов с помощью макрорекордера.

Существует два различных способа создания макроса.

1. Необходимо выполнить задачу, которую вы хотите автоматизировать, при этом компьютер записывает все ваши действия для последующего их воспроизведения.
2. Написать макрос "с нуля", используя язык программирования Visual Basic for Applications (VBA).

Макрорекордер – это встроенный в Microsoft Excel инструмент, который может отслеживать выполнения вами задачи и после этого автоматически повторять те же шаги.

Процесс записи макроса можно свести к следующим шагам.

1. Сообщите Microsoft Excel, что хотите начать запись макроса.

2. Дайте имя макросу.

3. Выберите параметры для макроса, такие как описание и где он должен быть сохранен.

4. Запустите макрорекордер.

5. Выполняйте задачу (задачи), которую хотите автоматизировать.

6. Остановите макрорекордер.

Запись макроса

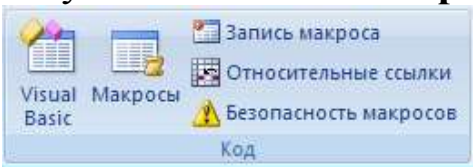
При записи макроса все шаги, необходимые для выполнения действий, записываются программой записи макроса. Перемещение по ленте не включается в записанные шаги.

1. Если вкладка **Разработчик** недоступна, выполните следующие действия для ее отображения:

- щелкните значок **Кнопка Microsoft Office**  , а затем щелкните **Параметры Excel**.
- в категории **Личная настройка** в группе **Основные параметры работы с Excel** установите флажок **Показывать вкладку "Разработчик" на ленте**, а затем нажмите кнопку **ОК**.

2. Для установки уровня безопасности, временно разрешающего выполнение всех макросов, выполните следующие действия:

- на вкладке **Разработчик** в группе **Код** нажмите кнопку **Безопасность макросов**.



- в группе **Параметры макросов** выберите переключатель **Включить все макросы (не рекомендуется, возможен запуск опасной программы)**, а затем дважды нажмите кнопку **ОК**.

ПРИМЕЧАНИЕ. Для предотвращения запуска потенциально опасных программ по завершении работы с макросами рекомендуется вернуть параметры, отключающие все макросы.

3. На вкладке **Разработчик** в группе **Код** нажмите кнопку **Запись макроса**.

4. В поле **Имя макроса** введите имя макроса.

5. В списке **Сохранить в** выберите книгу, в которой необходимо сохранить макрос.

6. Для присоединения описания макроса введите нужный текст в поле **Описание**.

7. Для начала записи макроса нажмите кнопку **ОК**.

8. Выполните действия, которые нужно записать.

9. На вкладке **Разработчик** в группе **Код** нажмите кнопку **Остановить запись** .

СОВЕТ. Можно также нажать кнопку **Остановить запись** слева от строки состояния

Чтобы использовать макрос в любом месте рабочего стола, перед началом записи макроса необходимо выделить начальную ячейку. Если выделить ячейку после начала записи макроса, то процедура выделения будет воспринята как часть макроса.

Использование записанных макросов.

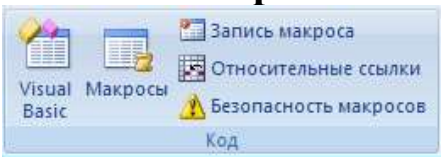
Чтобы выполнить ранее созданный макрос:

1. Если вкладка **Разработчик** недоступна, выполните следующие действия для ее отображения:

- щелкните значок **Кнопка Microsoft Office** , а затем щелкните **Параметры Excel**.
- в категории **Основные** в группе **Основные параметры работы с Excel** установите флажок **Показывать вкладку "Разработчик" на ленте**, а затем нажмите кнопку **ОК**.

2. Для установки уровня безопасности, временно разрешающего выполнение всех макросов, выполните следующие действия.

- на вкладке **Разработчик** в группе **Код** нажмите кнопку **Безопасность макросов**.



- в категории **Параметры макросов** в группе **Параметры макросов** нажмите кнопку **Включить все макросы (не рекомендуется, возможен запуск опасной программы)**, а затем нажмите кнопку **ОК**.

ПРИМЕЧАНИЕ. Для предотвращения запуска потенциально опасных программ по завершении работы с макросами рекомендуется вернуть параметры, отключающие все макросы.

3. Откройте книгу, содержащую нужный макрос.
4. На вкладке **Разработчик** в группе **Код** нажмите кнопку **Макросы**.
5. В поле **Имя макроса** введите имя макроса, который нужно выполнить.
6. Выполните одно из следующих действий:

- для запуска макроса в книге Excel нажмите кнопку **Выполнить**.

СОВЕТ. для запуска макроса можно также нажать клавиши CTRL+F8. Для прекращения выполнения макроса нажмите клавишу ESC.

- для запуска макроса из модуля Microsoft Visual Basic нажмите кнопку **Изменить**, а затем в меню **Run** выберите команду **Run Sub/UserForm**  или нажмите клавишу F5.

СОВЕТ. При работе в редакторе Visual Basic можно запускать разные макросы

Запуская макрос, вы выполняете записанные в нем действия. В некоторых случаях перед началом запуска макроса необходимо выделить ячейку или несколько ячеек, которые подлежат обработке с помощью макроса.

Опции доступные при записи макроса

В диалоговом окне запись макроса имеется несколько опций

Имя и описание макроса

С одной стороны, не имеет значения, какое имя вы даете макросу, поскольку макрос, как его не назови, будет выполнять те же самые действия. Но, с другой стороны, можно назвать макрос так, чтобы его имя напоминало вам, что он делает.

Имеется несколько правил, которые необходимо соблюдать при задании имени макросу.

1. Имя макроса не должно содержать более 255 символов.
2. Имя должно начинаться с буквы, хотя может включать и цифры. Можно использовать символ подчеркивания, если хотите разделить слова.
3. В имени макроса нельзя использовать пробел и другие

специальные символы.

4. Следует избегать использования ключевых слов VBA, имеющих специальное значение.

Поле *Описание* позволяет разместить более подробное описание того, что делает макрос. Это описание потом отображается в диалоговом окне *Макрос* при выборе имени макроса. Здесь можно ввести несколько строк, но не следует вводить слишком длинные описания.

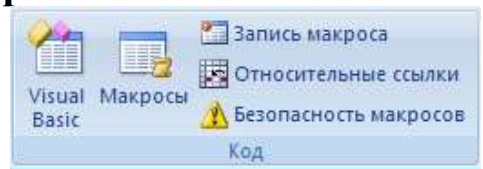
Быстрый запуск макросов

Так как макросы предназначены для ускорения вашей работы, их использование должно быть как можно более удобным. *Microsoft Excel* предоставляет 3 простых способа доступа к макросам.

1. С помощью комбинации клавиш
2. Нажатием кнопки на панели быстрого доступа
3. Запуск макроса щелчком области графического объекта
4. Кнопкой на рабочем листе

Запуск макроса нажатием клавиши CTRL
в сочетании с клавишей быстрого вызова

1. На вкладке **Разработчик** в группе **Код** нажмите кнопку **Макросы**.



2. В поле **Имя макроса** выберите макрос, которому нужно назначить сочетание клавиши CTRL с клавишей быстрого вызова.
3. Нажмите кнопку **Параметры**.
4. В поле **Сочетание клавиш** введите нужную прописную или строчную буквы.

ПРИМЕЧАНИЕ. Выбранное сочетание клавиш заменяет все совпадающие стандартные сочетания клавиш Excel на то время, пока открыта книга, содержащая данный макрос. Список уже заданных сочетаний клавиши CTRL с клавишами быстрого вызова см. в статье **Клавиши быстрого вызова** и функциональные клавиши Excel.

5. Для добавления описания к макросу введите нужный текст в поле **Описание**.
6. Нажмите кнопку **ОК**, а затем — кнопку **Отмена**.

Запуск макроса нажатием кнопки на панели быстрого доступа

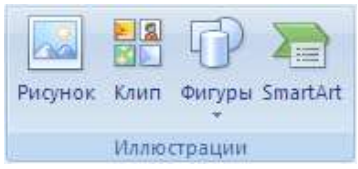
1. Щелкните значок **Кнопка Microsoft Office**  , а затем щелкните **Параметры Excel**.
2. В категории **Настройка** в списке **Выбрать команды из** выберите вариант **Часто используемые команды**.
3. В поле списка выберите вариант **Макросы**, а затем нажмите кнопку **Добавить**.
4. Нажмите кнопку **ОК**.

СОВЕТ. Кнопка **Макросы** будет добавлена на **панель быстрого доступа**.

5. На **панели быстрого доступа** нажмите кнопку **Макросы**.
6. В поле **Имя макроса** выберите макрос, который нужно запустить, а затем нажмите кнопку **Выполнить**

Запуск макроса щелчком области графического объекта

1. Выделите на листе графический объект (рисунок, клип, фигуру или рисунок SmartArt).
2. Для создания активной области на существующем объекте нажмите кнопку **Фигуры** в группе **Иллюстрации** на вкладке **Вставка**, выберите одну из фигур и нарисуйте ее на существующем объекте.



3. Щелкните созданную активную область правой кнопкой мыши, а затем выберите пункт **Назначить макрос** в контекстном меню.
4. Выполните одно из следующих действий:

- чтобы назначить графическому объекту существующий макрос, дважды щелкните нужный макрос или введите его имя в поле **Имя макроса**.

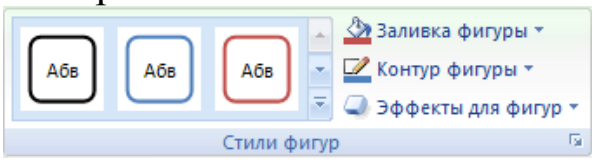
- чтобы записать новый макрос для назначения выделенному графическому объекту, нажмите кнопку **Записать**. После завершения записи макроса нажмите кнопку **Остановить запись** на вкладке **Разработчик** в группе **Код**.

СОВЕТ. Можно также нажать кнопку **Остановить запись** слева от строки состояния.

- для редактирования существующего макроса щелкните его имя в поле **Имя макроса**, а затем нажмите кнопку **Изменить**.

5. Нажмите кнопку **ОК**.
6. На листе выберите активную область.

Отобразится панель **Рисование** и вкладка **Формат**.



7. На вкладке **Формат** в группе **Стили фигур** выполните следующие действия:

- щелкните стрелку рядом с кнопкой **Заливка фигуры** и выберите вариант **Нет заливки**.
- щелкните стрелку возле кнопки **Контур фигуры** и выберите вариант **Нет контура**.

Запуск макроса кнопкой на рабочем листе

Вы можете поместить кнопку непосредственно на рабочий лист.

1. Выберите **Разработчик**.
2. Щелкните **Вставить** вберите значок кнопки, после этого щелкните в каком-либо месте рабочего листа и растяните границы кнопки.
3. Выполните правый щелчок кнопки и выберите во всплывающем меню пункт **Исходный текст**, чтобы открыть код события Click (Щелчок) для этой кнопки.
4. Напечатайте в месте установки курсора следующую строку:

Call ***имя макроса***

При этом программный код примет вид:

Private Sub CommandButton1_Click()
Call ***имя макроса***
End sub

5. Закройте окно редактора кода и выйдите из режима разработки (закройте окно Microsoft Visual Basic for Application)
6. Щелкните кнопку **Выход из режима конструктора**.

Теперь созданная вами кнопка будет запускать макрос.

Для того чтобы удалить созданную вами кнопку: на панели инструментов **Элементы управления** щелкните кнопку **Режим конструктора**, выполните правый щелчок по вашей кнопке и выберите команду **Вырезать**.

Выбор места для хранения макросов.

Имеется три места для сохранения макроса в рабочей книге. Макрос можно хранить в текущей, новой, или особой книге, называемой **Личная книга макросов**. По умолчанию запустить макрос можно только в том случае, если рабочая книга, в которой создан открыта. Для того, чтобы иметь доступ к макросу в любое время макрос нужно сохранять в **Личной книге макросов**. Личная книга макросов – скрытый файл, который Microsoft Excel создает при выборе соответствующей опции, и при каждом последующем запуске Microsoft Excel этот файл открывается автоматически.

Использование относительных ссылок

Например, вы хотите создать макрос, который записывает ваше имя в любую выделенную ячейку, а фамилию в ячейке непосредственно под ней.

Поскольку нужно, чтобы макрос воздействовал на любую выделенную ячейку, необходимо выбрать эту ячейку до включения макрорекордера. Предположим, что это ячейка Е6. После того, как вы начали запись макроса и ввели свое имя в выделенную ячейку, вам надо будет выделить ячейку Е7 и ввести свою фамилию.

Если вы хотите выполнить данный макрос в другом столбце таблицы, то вы выделяете, например ячейку D2, а затем запускаете макрос. При этом ваше имя записывается в ячейку D2, а фамилия в ячейку Е7.

Данную проблему позволяют решить относительные ссылки.*

Прежде чем вносить в выделенную ячейку свое имя, на панели инструментов **Остановка записи** необходимо щелкнуть на кнопке **Относительная ссылка** (кнопка должна быть утоплена). Теперь, когда вы будете записывать макрос, все выделенные ячейки запишутся относительно заданной позиции.

Для многих макросов не имеет значения, отмечена опция **Относительная ссылка** или нет. На работу макросов, которые не производят изменений в отдельных ячейках или группах ячеек, установка этих опций не влияет. Например, макрос, который отключает линии сетки и изменяет расположение страницы, будет работать одинаково, независимо от состояния опции **Относительные ссылки.**

3. ПОРЯДОК ВЫПОЛНЕНИЯ

1. Получить задание у преподавателя.
2. Создать макрос.
3. Назначить способ быстрого запуска макроса в соответствии с вариантом.

4. ЗАДАНИЕ

Вариант 1

Таблица умножения											
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10
2	0	2	4	6	8	10	12	14	16	18	20
3	0	3	6	9	12	15	18	21	24	27	30
4	0	4	8	12	16	20	24	28	32	36	40
5	0	5	10	15	20	25	30	35	40	45	50
6	0	6	12	18	24	30	36	42	48	54	60
7	0	7	14	21	28	35	42	49	56	63	70
8	0	8	16	24	32	40	48	56	64	72	80
9	0	9	18	27	36	45	54	63	72	81	90
10	0	10	20	30	40	50	60	70	80	90	100

4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое макрос?
2. Как создать макрос с помощью макрорекордера?
3. Как запустить макрос?
4. Как влияет опция относительная ссылка на создание макроса?
5. Какие действия необходимо предпринять, чтобы макрос выполнялся в любом месте рабочего листа?

Лабораторная работа № 6
Работа в Microsoft Access

1. ЦЕЛЬ РАБОТЫ

Целью работы является приобретение умений в создании баз данных с помощью СУБД Access

2. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

2.1. Основные понятия баз данных

Компьютерные базы данных являются основой большинства автоматизированных информационных систем и служат средством рационального и эффективного хранения информации. Базы данных обеспечивают надежную защиту данных от случайной потери или порчи, экономно используют ресурсы (людские и технические) и снабжаются механизмами поиска информации.

В широком смысле под базой данных понимают совокупность сведений о конкретных объектах реального мира в какой-либо предметной области. В узком смысле база данных – это именованная и определённым образом организованная совокупность данных, отражающая состояние объектов и их отношений в рассматриваемой предметной области.

Под предметной областью принято понимать часть реального мира, подлежащая изучению. Поскольку предметная область обычно неупорядочена и имеет очень сложную структуру, то для создания баз данных используют упрощенные описания предметной области в виде модели данных.

Модель данных – формально определённая структура, которая используется для представления данных. Модели данных разделяют на иерархические, сетевые и реляционные. Иерархическая модель данных организует данные в виде древовидной структуры, сетевая – в виде сетевой, реляционная модель – в виде таблиц (отношений).

Сущностью (информационным объектом) называют элемент предметной области, информация о котором интересует пользователя. Каждый информационный объект описывается рядом основных свойств – атрибутов. Атрибутом называют именованную характеристику объекта. Он показывает, какая информация об

объекте интересует пользователя и должна храниться в базе данных.

Например, предметная область – высшее учебное заведение; сущности – студент и преподаватель; атрибуты – фамилия студента, его адрес проживания, группа, фамилия преподавателя, дисциплина, по которой он проводит занятия, учёное звание и учёная степень.

Для централизованного управления массивами данных в базе используют систему управления базой данных. Системой управления базой данных (СУБД) называют комплекс программных и языковых средств, предназначенных для создания, ведения и совместного применения базы данных любыми пользователями. Иначе говоря, СУБД выступает в качестве интерфейса между пользователями и базой данных. Существующие СУБД по типам моделей данных делятся на реляционные, иерархические и сетевые.

2.2. Реляционные модели данных

В последнее время в большинстве СУБД используется реляционная модель данных, в которой данные представляются в виде совокупности взаимосвязанных таблиц. Такое взаимоотношение между таблицами называется связью.

Между записями двух таблиц (например, таблиц A и B) могут существовать следующие виды связей:

1) связь "один к одному" ($1 \leftrightarrow 1$), при которой каждой записи из A соответствует одна определённая запись из B (например, в каждом государстве существует одна политическая столица);

2) связь "один ко многим" ($1 \leftrightarrow \infty$), при которой каждой записи из A соответствует несколько записей из B, (например, на станке осуществляется обработка несколькими инструментами);

3) связь "многие к одному" ($\infty \leftrightarrow 1$), при которой множеству записей из A соответствует одна определённая запись из B (например, несколько жильцов живут в одном доме);

4) связь "многие ко многим" ($\infty \leftrightarrow \infty$), при которой множеству записей из A соответствует множество записей из B, (например, у нескольких студентов занятия ведут разные преподаватели).

Связи устанавливаются посредством ключевых полей или ключей. Первичным ключом называется поле, содержащее только уникальные записи. Внешний ключ – это поле, содержащее значения ключевого поля другой таблицы. Внешние ключи используются для установления логических связей между таблицами. Связь между таблицами устанавливается путём присваивания значений внешнего ключа одной таблицы значениям ключа другой таблицы.

2.3. Основные объекты СУБД Access

СУБД Access при обработке информации рассматривает базу данных как набор нескольких структурных элементов, каждый из которых может включать один или несколько объектов. Среди основных составляющих базы данных с точки зрения Access можно выделить следующие объекты:

1. Таблицы, представляющие собой объекты, которые создаются пользователем для хранения информации о сущностях предметной области в определённой структуре. Любая таблица состоит из полей (столбцов) и записей (строк).

2. Запросы, являющиеся объектами, которые предназначены для получения требуемых данных из имеющихся в базе данных таблиц. При помощи запросов можно создавать выборки данных, добавлять или удалять информацию в определённой таблице. Кроме того, с помощью запроса возможно также создание новых таблиц на основании одной или нескольких имеющихся в базе данных.

Можно выделить следующие основные виды запросов в Access:

- Запрос на выборку, позволяющий выбирать из таблиц содержащуюся в них информацию по заранее заданному условию.

- Параметрический запрос, представляющий собой запрос на выборку, в котором условие отбора определяется во время его выполнения.

- Запрос с вычисляемым полем, позволяющий не только выбирать из таблиц содержащуюся в них информацию, но также производить вычисления и отображать результат вычисления в результирующей таблице.

- Запрос на обновление записей в таблице.

- Запрос на добавление записей в таблицу, переносящий (добавляющий) данные из одной таблицы в другую.

- Запрос на создание таблицы.

- Перекрёстный запрос, представляющий собой сводную таблицу, в которой данные могут быть сгруппированы определённым образом, а также может подсчитываться среднее, максимальное, минимальное и другие значения.

3. Формы, представляющие собой объекты, которые используются для разработки интерфейса, при помощи которого происходит ввод данных пользователем, а также отображение на экране имеющейся в базе данных информации.

4. Отчеты, являющиеся объектами, которые используются для подведения каких-либо итогов на основании имеющихся данных, и вывода этих итогов в определённом формате на печать.

5. Макросы, являющиеся объектами, которые предназначены для выполнения определенных действий при возникновении того или иного события. Например, с помощью макросов можно создавать запросы, формировать отчеты, открывать таблицы, обрабатывать формы и т. д.

2.4. Типы полей в Access

При создании структуры таблицы необходимо задать типы полей, которые будут в ней использоваться. Тип поля определяется типом данных, которые в нем хранятся. Можно выделить следующие типы полей в Access:

1) Текстовый – используется для хранения строковых данных с длиной до 255 символов;

2) Поле MEMO – тоже поле текстового типа, которое может содержать до 65535 символов и предназначено для хранения комментариев (не может быть ключевым полем);

3) Числовой – хранит числовую информацию, длина поля зависит от значения свойства Размер поля;

4) Дата/время – хранит значение времени и даты, с помощью свойства Формат поля задаётся способ представления значения;

5) Денежный – используется для проведения денежных расчётов с точностью до 15 знаков в целой и 4 знаков в дробной частях;

6) Счётчик – по умолчанию значения поля уникальные целые числа, последовательно возрастающие на единицу; значение этого поля нельзя изменить или удалить (ввести); используется в качестве первичного ключа таблицы;

7) Логический – хранит логические данные, имеющие одно из двух возможных значений: да/нет, истина/ложь, вкл./выкл.;

8) Поле объекта OLE – используется для вставки объекта другого приложения, например, рисунка, фотографии, звука и т. д. (меню Вставка, Объект);

9) Гиперссылка – можно указать путь к файлу на жёстком диске или адрес (универсальный указатель ресурса) Web-страницы в сети Интернет;

10) Мастер подстановки – выбор этого типа данных запускает Мастер подстановок, обычно используется для внешних ключей, реализует первый способ создания поля подстановки (поле с раскрывающимся списком) в режиме конструктора таблиц.

1. ПОРЯДОК ВЫПОЛНЕНИЯ

1. Получить задание у преподавателя.
2. Создать таблицы.
3. Установить связи между таблицами.
4. Включить "Обеспечение целостности данных" для всех связей".
5. Создать формы для заполнения таблиц.
6. Заполнить таблицы.
7. Создать запросы.
8. Создать отчёт.

2. ЗАДАНИЕ

Задание выдается преподавателем индивидуально после беседы и опроса студента.

Лабораторная работа № 7
Массивы. Элементарные операции с матрицами

1. ЦЕЛЬ РАБОТЫ

Целью работы является изучение массивов и получение практических навыков при работе с одномерными и двумерными массивами.

2. ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Основные положения

Массивом называется последовательность или таблица переменных одного типа, называемых элементами массива. В обращении к элементу указывается имя массива и один (если массив одномерный) или несколько (если массив многомерный) индексов.

Одномерный массив аналогичен строке или столбцу таблицы и вектору в математике (рис.1).

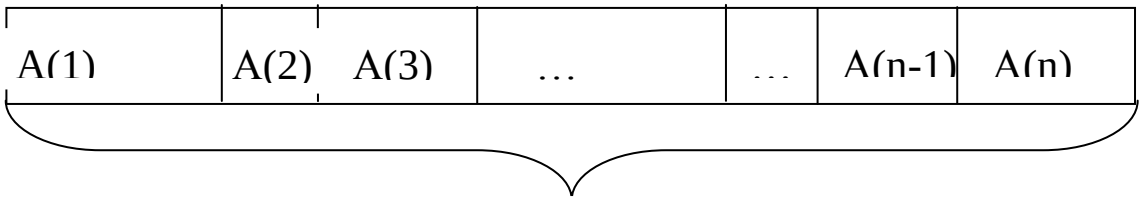


Рис. 1. Одномерный массив, где A(1), A(2), …, A(n) – элементы одномерного массива A, состоящего из n элементов

Двумерный массив аналогичен прямоугольному диапазону ячеек таблицы Excel и матрице в математике (рис 2).

Прежде чем использовать массив, его следует описать (объявить). Кроме того, для каждого индекса должны быть определены нижняя и верхняя границы, в пределах которых индекс может меняться. Значения элементам присваивается по порядку. Если элементов в массиве больше, чем инициализаторов, то элементы, для которых значения не указаны обнуляются.

B(1,1)	B(1,2)	…	…	B(1,m)
B(2,1)	B(2,2)	…	…	B(2,m)
…	…	…	…	…
B(n,1)	B(n,2)	…	…	B(n,m)

Массив

Рис. 2. Двумерный массив,
где B(1,1), B(1,2), B(n,m) –
элементы двумерного массива B размерностью n × m

Объем оперативной памяти, требуемый для массива, равен произведению количества байтов для одного элемента массива на количество элементов.

Размерность массива определяется количеством элементов массива, поэтому размерность может быть задана только целой положительной константой или константным выражением. Размерность массивов предпочтительнее задавать с помощью именованных констант, так как при таком подходе для ее изменения достаточно изменить значение константы всего лишь в одном месте программы.

Существует два вида массивов – статические и динамические.

Статические массивы.

При описании статического одномерного массива задаются верхняя и нижняя границы для индекса, определяющие количества элементов массива, причем заданные границы не могут быть изменены в программе.

Динамические массивы.

Динамическим называется массив, размер которого может меняться во время исполнения программы. Динамические массивы дают возможность более гибкой работы с данными, так как позволяют не прогнозировать хранимые объёмы данных, а регулировать размер массива в соответствии с реально необходимыми объёмами.

Описание массивов

Массивы в коде программы объявляются так же, как и переменные. Место, в котором объявляется массив, определяет, где его можно использовать (его область видимости). Если массив объявлен локально для процедуры, то его можно использовать только в этой процедуре. Если массив объявлен в верхней части формы, то его можно использовать во всей форме. Если массив объявлен открытым в стандартном модуле, его можно использовать в любом месте проекта.

Описание статических массивов

Статические массивы на языке программирования Visual Basic for Applications (VBA) описываются в программе при помощи ключевых слов Dim и As. Границами являются целые числа в скобках. Между верхней и нижней границами ставится ключевое слово To.

Одномерные массивы имеют один индекс.

Базовый синтаксис одномерного статического массива следующий:

Dim Имя_массива (граница1 To граница 2) As тип_данных.

Здесь важную роль играют следующие аргументы.

- Dim – это ключевое слово, для объявления массива. Если массив помещается в стандартный модуль, то вместо Dim используется слово Public.
- Имя_массива – это имя переменной массива, которое будет использовано для представления массива в программе. Имена массивов подчиняются тем же правилам, что и имена переменных.
- Граница1 – это нижняя граница массива.
- Гранца2 – это верхняя граница массива.
- Тип_данных – это тип данных, хранимых в массиве. В большинстве случаев все переменные в массиве относятся к одному и тому же типу данных. Если в массиве будут храниться данные разных типов, то можно указать тип Object.

Например:

Dim A(1 To 10) As Integer

Dim B(-10 To 10) As String

Если в скобках указано только одно целое число, то это – верхняя граница. При этом нижняя граница равна нулю.

Например:

Dim C(10) As Byte.

Этот оператор эквивалентен оператору Dim C(0 To 10) As Byte.

Если нужно, чтобы нижней границей массива была единица, то перед первой строкой программы необходимо набрать строку Option Base 1.

При этом оператор последнего примера будет эквивалентен следующему:

Dim C(1 To 10) As Byte.

Значения границ не должны выходить за пределы диапазона значений для данных типа Long.

Для описания многомерных массивов (имеющих несколько индексов), указывается несколько границ, через запятую.

Например:

Dim A(4,4) As Byte

Dim B(1 To 5, -5 To -1) As Byte.

В данных примерах объявленные массивы A и B являются двумерными и содержат одинаковое количество элементов, равное $5 \times 5 = 25$.

Число размерностей массива может достигать 60.

Описание динамических массивов

Динамические массивы используются в том случае, когда количество элементов массива заранее не известно, а определяется в процессе выполнения программы. По окончании работы с динамическим массивом можно освободить память, которую он занимает. Это важно для задач, требующих большого объема оперативной памяти.

Описание динамического массива осуществляется в два этапа:

1. Объявляется массив с использованием оператора Dim, но без указания размерности. Признаком массива являются скобки после его имени.

2. В нужном месте программы описывается данный массив с указанием размерности при помощи оператора ReDim, причем в качестве границ можно использовать не только целые числа, но и арифметические выражения. Важно, чтобы к выполнению оператора ReDim все переменные в этих арифметических выражениях имели числовые значения.

При помощи оператора ReDim можно задавать любые размерности массива. Например:

```
Dim A() As Byte
ReDim A(1, 1)          ′ двумерный массив
A(0, 0)=13
A(1, 1)=14
ReDim A(3, 3, 1 To 3)          ′ трехмерный массив.
```

Следует иметь в виду, что при каждом выполнении оператора ReDim (то есть при каждом переопределении массива) значения элементов будут потеряны, так как оператор ReDim обнуляет все элементы массива.

Чтобы при переопределении массива значения элементов не пропали, используется ключевое слово Preserve при изменении верхней границы одномерного массива или при изменении верхней границы последней размерности многомерного массива.

Например:

```
Sub Сохранение()
Dim I As Integer
Dim J As Integer
Dim A() As Integer          ′объявление массива
ReDim A(2, -5 To 1)         ′указание размерности
For I = 0 To 2
  For J = -5 To 1
    A(I, J) = (I+1)*J^2
  Next J
Next I
ReDim Preserve A(2, -5 To 4)    ′указание размерности
For I = 1 To 2
  For J = 2 To 4
```

```
    A(I, J) = (I + 1)*J^3
  Next J
Next I
End Sub.
```

Работа с массивами

Для определения значений нижней и верхней границ массива любой размерности используются функции LBound и UBound соответственно.

Например: Используя функции LBound и UBound определить границы одномерного массива A.

```
Dim Low As Integer
Dim Up As Integer
Low = LBound(A)     'Low – нижняя граница массива
Up = UBound(A)      'Up – верхняя граница массива.
```

Функция UBound необходима, например, когда значение верхней границы одномерного массива неизвестно и при этом необходимо увеличить это значение на определенной число.

Для освобождения оперативной памяти, занимаемой динамическим массивом, используется оператор Erase.

Например:

```
Sub Память()
  Dim A() As Byte
  Dim B() As Byte
  ReDim A(8)          'Память для A: 9 байт
  Erase A             'Память для A: 0 байт
  ReDim B(2, 2)       'Память для B: 3*3 = 9 байт
End Sub.
```

Для того чтобы просмотреть выполнение данной программы необходимо после ввода программы "Память" выполнить следующие действия:

1. Дважды выполнив Debug (отладка) > Add Watch (добавить контрольное значение), в окне Watches (контрольное значение) сгенерировать строки, соответствующие массива A и B.

2. Кликнуть левой кнопкой мыши в любом месте программы для установки там мигающего курсора.

3. Произвести пошаговое выполнение программы "Память", наблюдая за распределением памяти с помощью окна Watches.

Для решения многих математических задач требуются массивы случайных чисел. Случайные числа рассчитываются с помощью функции Rnd.

Перед серией обращений к функции Rnd должен находиться оператор Randomize.

Например: программа, рассчитывающая 10 случайных чисел от 0 до 1:

```
Sub случайные_числа()
Dim N As Integer
Dim I As Integer
Dim S() As Single
N = 10
ReDim S(1 To N)
Randomize
For I = 1 To N
  S(I) = Rnd
Next I
End Sub.
```

Операции с матрицами

Основные виды матриц

В математике матрицей называют прямоугольную таблицу значений, упорядоченных по строкам и столбцам, например, матрица A размера n × m имеет вид:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix},$$

где a_{ij} – элемент матрицы A, расположенный в i-ой строке j-ого столбца; m, n – количество строк и столбцов матрицы соответственно.

Матрица характеризуется размерностью, то есть произведением числа строк на число столбцов.

Элементы a_{ii} (i = j) образуют главную диагональ матрицы A.

Если количество строк и столбцов матрицы совпадают, т.е. n = m то матрицу называют квадратной, если не совпадают – прямоугольной матрицей.

Матрица размером $1 \times m$ называется вектором-строкой, размера $n \times 1$ – вектором столбцом.

Нулевой называется матрица у которой элементы $a_{ij} = 0$.

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Единичной называется квадратная матрица, у которой на главной диагонали стоят 1, а все остальные элементы равны 0.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Диагональной является квадратная матрица, где все элементы (кроме элементов, расположенных на главной диагонали) равны 0, т.е. $a_{ij} = 0$ при $i \neq j$.

$$\begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 9 \end{pmatrix}$$

Квадратная матрица A размерностью $n \times n$ называется симметричной (симметрической), если $a_{ij} = a_{ji}$.

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 5 & 6 & 7 \\ 3 & 6 & 8 & 9 \\ 4 & 7 & 9 & 0 \end{pmatrix}$$

Треугольной называется матрица порядка $n \times n$, у которой одна часть элементов (либо над главной диагональю, либо под ней) равна 0.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & 3 & 0 & 0 \\ 4 & 5 & 6 & 0 \\ 7 & 8 & 9 & 10 \end{pmatrix} \quad \begin{pmatrix} 1 & 2 & 3 & 4 \\ 0 & 5 & 6 & 7 \\ 0 & 0 & 8 & 9 \\ 0 & 0 & 0 & 10 \end{pmatrix}$$

Транспонированной матрицей A^T, называется матрица у которой строки полностью совпадают со столбцами исходной матрицы A, а столбцы – со строками матрицы A.

Основные операции с матрицами

Операции над матрицами определяются с помощью операций над их элементами:

1. Две матрицы A и B размерностью $n \times m$ равны друг другу ($A = B$) в том случае, если $a_{ij} = b_{ij}$;

2. Сумма матриц A и B размерностью $n \times m$ есть матрица $C(n \times m)$, то есть $C = A + B = (a_{ij} + b_{ij}) = c_{ij}$, где $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, m$;

3. Произведение матрицы A на скаляр α – есть матрица $C = \alpha \cdot A = (\alpha \cdot a_{ij}) = c_{ij}$;

4. Произведение матрицы A размерностью $n \times m$ на матрицу B размерностью $m \times r$ – есть матрица C размерностью $n \times r$, то есть $c_{ij} = \sum_{k=1}^{m} \left(a_{ik} \cdot b_{kj}\right)$, где $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, m$.

Ввод матриц

1. Блок схема формирования произвольной матрицы A размерностью $(n \times m)$ приведена на рис 3.

2. Блок-схема формирования нулевой матрицы A размерностью $(n \times m)$ приведена на рис. 4.

3. Блок-схема формирования единичной матрицы A размерностью $(n \times n)$ приведена на рис. 5.

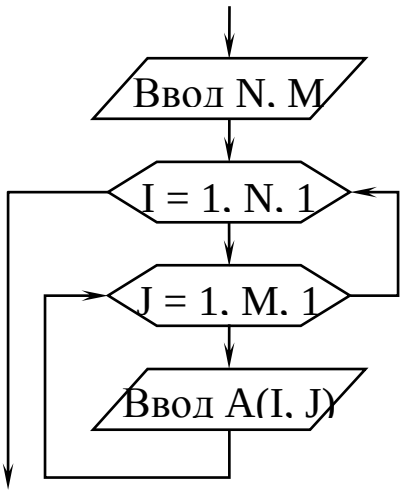


Рис. 3. Блок-схема
формирования произвольной матрицы A(n × m)

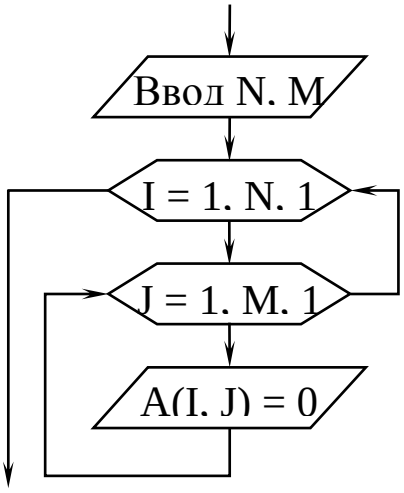


Рис. 4. Блок-схема формирования нулевой матрицы A(n × m)

4. Блок-схема формирования диагональной матрицы A размерностью (n × n) приведена на рис. 6.

5. Блок-схема формирования симметричной матрицы A размерностью (n × n) представлена на рис. 7.

6. Блок-схема формирования треугольной матрицы A размерностью (n × n) представлена на рис. 8, 9.

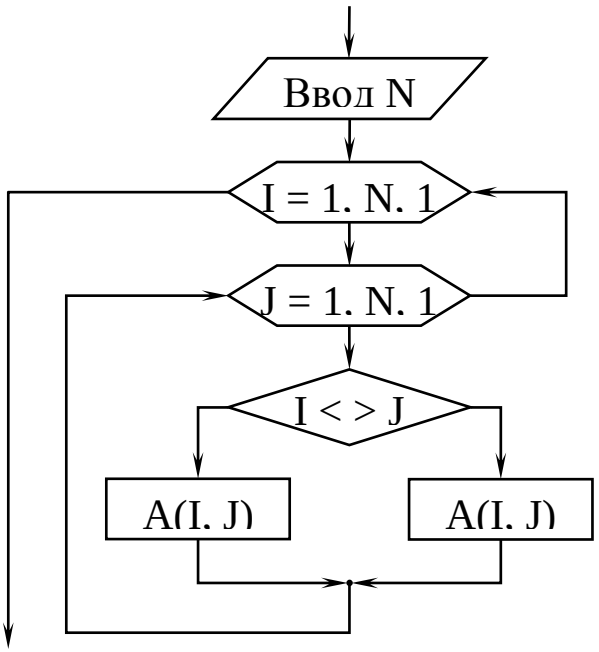


Рис. 5. Блок-схема формирования единичной матрицы A(n × n)

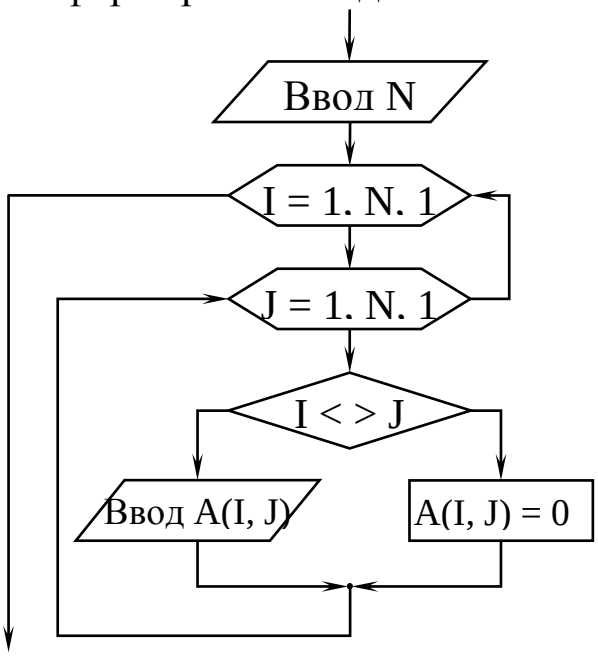


Рис. 6. Блок-схема формирования диагональной матрицы A(n × n)

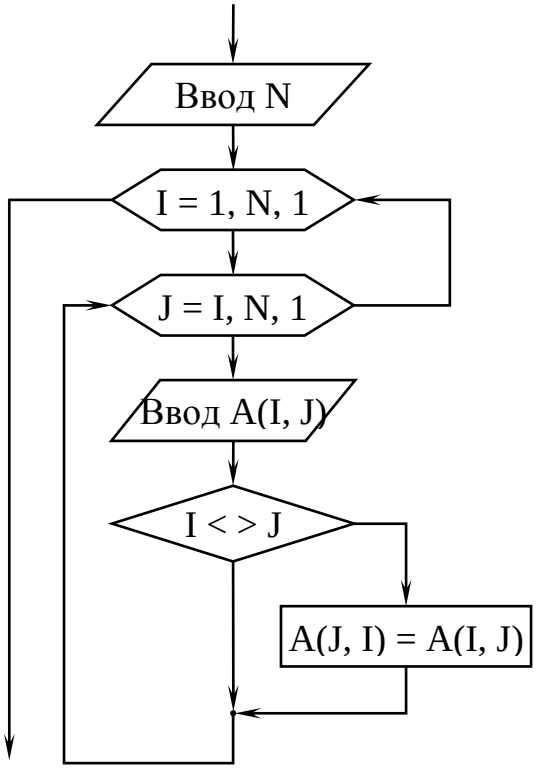


Рис. 7. Блок-схема
формирования симметричной матрицы A(n × n)

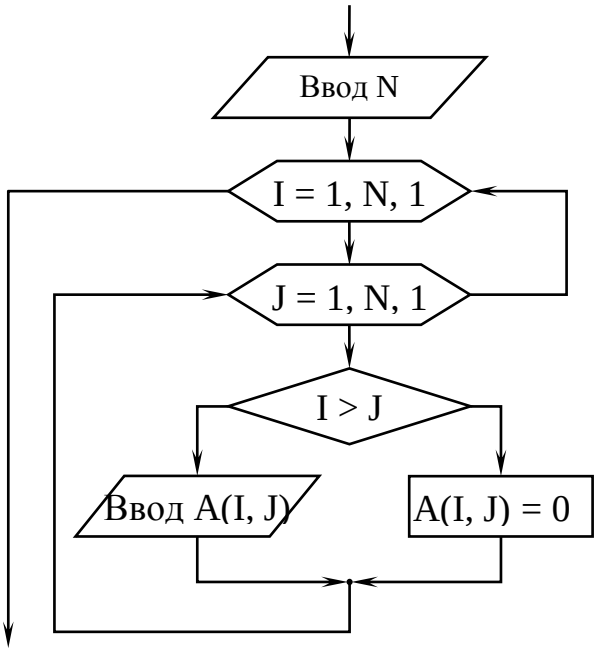


Рис. 8. Блок-схема формирования треугольной матрицы A(n × n)
с нулями под главной диагональю

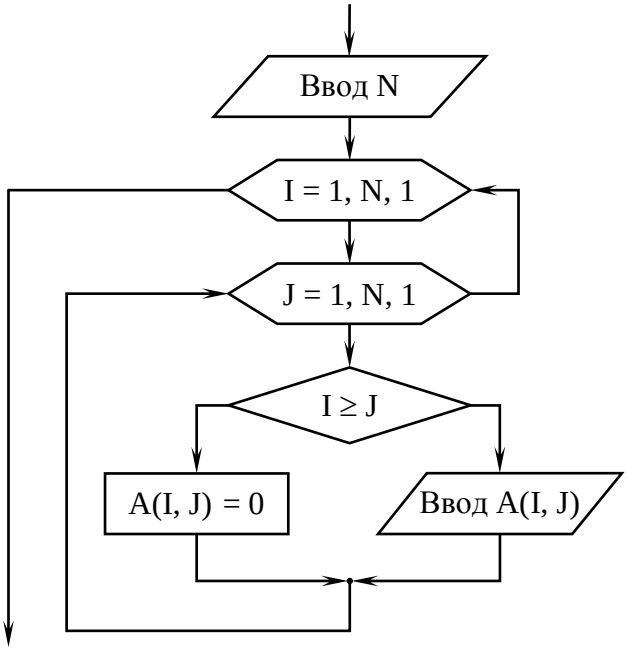


Рис. 9. Блок-схема формирования треугольной матрицы A(n × n) с нулями над главной диагональю

Вывод матриц

На рис. 10 представлена блок-схема вывода матрицы A(n × n).

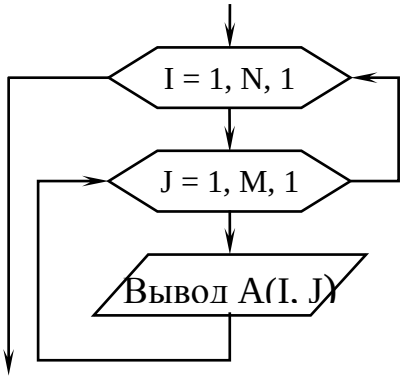


Рис. 10. Блок-схема вывода матрицы A(n × n)

Операции над матрицами

1. Блок-схема умножения матрицы A размерностью (n × m) на константу C и получения результирующей матрицы B представлена на рис. 11.

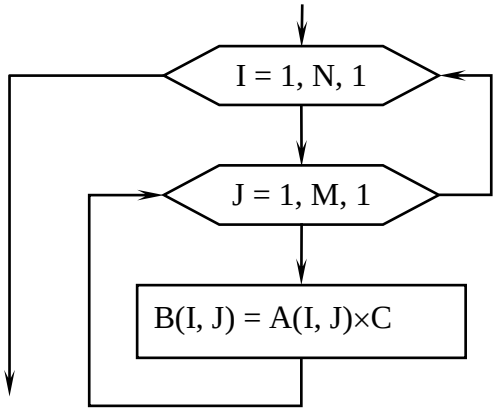


Рис. 11. Блок-схема умножения матрицы A(n × m) на константу C и получения результирующей матрицы B(n × m)

2. Блок-схема транспонирования матрицы A размерностью (n × n) представлена на рис. 12.

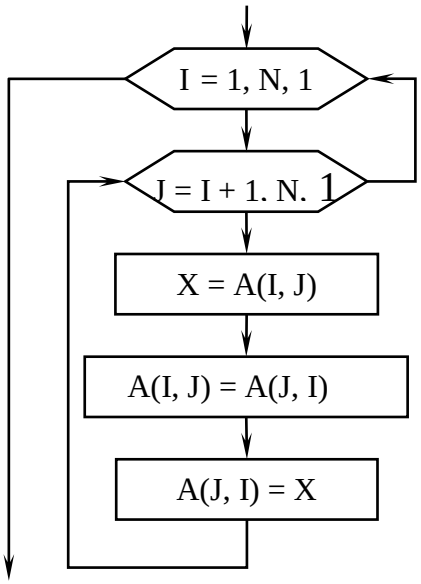


Рис. 12. Блок-схема транспонирования матрицы A(n × n)

3. Блок-схема сложения матриц A и B размерностями (n × m) и получения результирующей матрицы C той же размерности представлена на рис. 13.

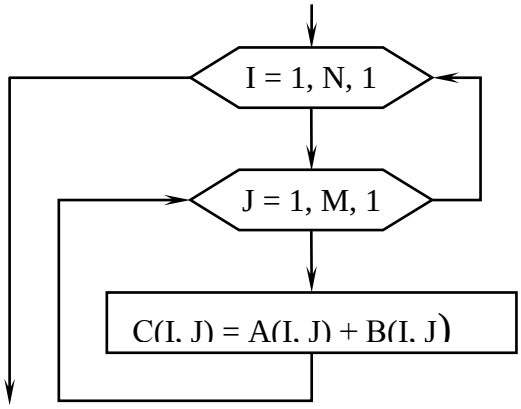


Рис. 13. Блок-схема A(n × m) и B(n × m)
и получения результирующей матрицы C(n × m)

4. Блок-схема умножения матриц A(m × n) и B(n × l) и получения результирующей матрицы C размерностью (m × l) представлена на рис. 14.

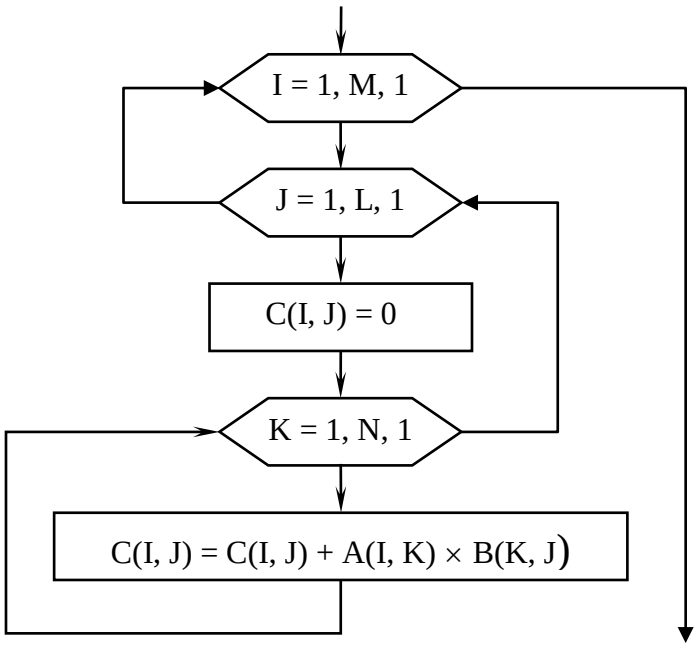


Рис. 14. Блок-схема умножения матриц A(m × n) и B(n × l)
и получения результирующей матрицы C(m × l)

3. ПОРЯДОК ВЫПОЛНЕНИЯ

1. Получить задание у преподавателя.
2. Выполнить задание в соответствии с вариантом.
3. Ответить на контрольные вопросы.

4. ЗАДАНИЕ

Варианты заданий для лабораторной работы по теме "массивы"

1. Дан массив натуральных чисел. Найти количество и сумму элементов, кратных заданному числу К.

2. В целочисленной последовательности есть нулевые элементы. Создать массив из номеров этих элементов.

3. Дана последовательность натуральных чисел. Создать массив из четных чисел заданной последовательности. Если таких чисел нет, то вывести сообщение об этом.

4. Дана последовательность действительных чисел. Заменить все ее члены, большие заданного числа К, этим числом. Подсчитать количество замен.

5. Дан массив действительных чисел. Поменять местами наибольший и наименьший элементы заданного массива.

6. Дан целочисленный массив. Напечатать те его элементы, индексы которых являются степенями двойки (1, 2, 4, 8 и т. д.).

7. В массиве целых чисел найти наиболее часто встречающееся число. Если таких чисел несколько, то определить наименьшее из них.

8. Дана последовательность целых чисел. Найти сумму ее членов, расположенных между максимальным и минимальным элементами (включая оба эти числа).

9. Дана последовательность целых чисел. Вывести произведения всех пар соседних чисел.

10. В заданной последовательности определить максимальный элемент. Подсчитать количество элементов равных максимальному.

11. В массиве, содержащем номер месяца рождения каждого студента группы, подсчитать количество всех студентов, которые родились в заданном месяце К, и напечатать их номера.

12. В области 10 районов. Заданы площади, засеваемые в каждом районе пшеницей, и урожай, собранный в каждом районе. Определить среднюю урожайность пшеницы по каждому району и по области в целом.

13. Известны значения ежедневной температуры воздуха в марте. Найти среднюю температуру и количество теплых дней (выше 0).

14. Даны натуральные числа помощью X1, X2, . . ., X10; Y1, Y2, . . . , Y10. Определить количество точек с координатами (Xi, Yi), которые лежат в круге с центром (125, 96) и радиусом 50.

15. Многочлен N-ой степени задан массивом своих коэффициентов. Определить коэффициенты первой производной заданного многочлена.

16. Дана последовательность действительных чисел. Указать те ее элементы, которые принадлежат отрезку [c, d].

17. Даны целые числа a_1, a_2, . . . , a_n. Определить только те числа, для которых выполняется условие $a_i \le i$.

Варианты заданий для лабораторной работы по теме "элементарные операции с матрицами"

1. $\mathbf{0.5 \times (A^2 + A^T)}$
2. $\mathbf{(A^2 + E) \times 6}$
3. $\mathbf{T \times T^T + E}$
4. $\mathbf{3 \times (C^2 + E)}$
5. $\mathbf{(A^2 - E) \times A^T}$
6. $\mathbf{(2 \times T + E) \times T^T}$
7. $\mathbf{2 \times A + (A + E)^2}$
8. $\mathbf{C \times (5 \times C - E)}$
9. $\mathbf{(3 \times T^2 + E)^T}$
10. $\mathbf{C^2 + 2 \times C + E}$
11. $\mathbf{A \times (4 \times A^T + E)}$
12. $\mathbf{(E + 7 \times T^T) \times T}$
13. $\mathbf{5 \times (A + E)^2}$
14. $\mathbf{8 \times C^2 + E}$
15. $\mathbf{T \times (3 \times T – E)}$
16. $\mathbf{(7 \times A^2 + E)^T}$

17. $\mathbf{C + (E - 6 \times C^2)}$
18. $\mathbf{A^2 - 2 \times (A + E)}$
19. $\mathbf{3 \times T^T + E + T^2}$
20. $\mathbf{7 \times (A^T + E)^2}$
21. $\mathbf{(E+C) \times 2 \times C}$
22. $\mathbf{A \times (5 \times A^T + E)}$
23. $\mathbf{T - (T^T + E)^2}$
24. $\mathbf{A^T \times (9 \times A - E)}$
25. $\mathbf{C^2 - 8 \times (C + E)}$
26. $\mathbf{A^2 + E - 3 \times A^T}$
27. $\mathbf{2 \times T^2 - E + T^T}$
28. $\mathbf{2 \times (A - E)^T \times A}$

Обозначения: **A**, **T**, **C**, **E** – квадратные матрицы порядка **N**;
A – произвольная матрица;
T – треугольная матрица;
C – симметричная матрица;
E – единичная матрица.

5. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое массив?
2. Одномерные и двумерные массивы.
3. Статические и динамические массивы.
4. Описание статических массивов.
5. Описание динамических массивов.
6. Функция Rnd.
7. Что такое матрица?
8. Расскажите об основных видах матриц.
9. Блок-схема формирования единичной матрицы.
10. Блок-схема формирования симметричной матрицы
11. Блок-схема транспонирования матрицы.
12. Блок-схема умножения матрицы A(m × n) на матрицу B(n × l).

Лабораторная работа № 8
Методы сортировки данных

1. ЦЕЛЬ РАБОТЫ

Целью работы является изучение методов сортировки и получение практических навыков сортировки массивов.

2. ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Основные положения

В общем случае сортировку следует понимать как процесс перегруппировки заданного множества объектов в некотором определенном порядке.

Цель сортировки – облегчить последующий поиск элементов в таком отсортированном множестве.

Мы встречаемся с отсортированными объектами в телефонных книгах, в оглавлениях книг, в библиотеках, в словарях, на складах – почти везде, где нужно искать хранимые объекты. Даже малышей учат держать свои вещи "в порядке", и они уже сталкиваются с некоторыми видами сортировки задолго до того, как ознакомятся с азами арифметики.

Сортировка – пример задачи, которую можно решать с помощью многих различных алгоритмов, имеющих и свои достоинства, и свои недостатки. И выбирать алгоритмы, нужно исходя из конкретной постановки задачи.

Введем некоторые обозначения и понятия.

Если у нас есть элементы A_1, A_2,…, A_N, то сортировка есть перестановка этих элементов так, чтобы при некоторой упорядочивающей функции F выполнялись отношения: $F(A_{K1}) \le F(A_{K2}) \dots \le F(A_{KN})$. Если сортировка происходит по убыванию, то знак отношения меняется на противоположный "≥". При решении практических задач упорядочивание массива, как правило, сопровождается некоторыми дополнительными действиями. Например, если A_1,Y_1; A_2,Y_2;…; A_N,Y_N – это значения аргумента A и некоторой функции Y = f(A), то перестановка этих аргумен-

тов должна сопровождаться одновременной перестановкой и значений функции.

Рассмотрим методы сортировки.

Сортировка методом прямого включения

Такой метод широко используется при игре в карты. Элементы (карты) мысленно делятся на уже "готовую" последовательность A_1, A_2,…, A_{i-1}, и "оставшуюся" (не сортированную) часть: A_i, A_{i+1},…, A_N.

Суть метода заключается в том, что при каждом i-м шаге (начиная с i = 2), из неотсортированной части извлекается i-й элемент и помещается в "готовую" часть, при этом он вставляется на нужное место.

Текстовый алгоритм метода:

1. Начало.

2. Выполнить цикл, пока i имеет значения от 2 до N, шаг = 1:

а) i-ый элемент (A(i)) поместить в ячейку A(0);

б) присвоить j = -1, то есть j равно номеру элемента, находящегося слева от испытуемого (i-го) и таким образом стоящего в "готовой" последовательности;

в) если A(0) ≥ A(j), то элемент A(0) поместить в ячейку A(j+1), иначе элемент A(j) поместить в ячейку A(j+1), уменьшить значение j на единицу и вновь выполнить пункт в).

3. Конец.

На рис. 1 представлена блок-схема сортировки методом прямого включения.

Метод работает следующим образом: на i-ом шаге (начиная с i = 2) i-й элемент помещается в свободную ячейку (например, A(0)). Этот элемент сравнивается со стоящим в "готовой" части слева от него элементом. Если элемент A(0) меньше, то происходит сдвиг вправо сравниваемого (j-го элемента) на одну позицию, после чего для сравнения берется следующий элемент. Если же элемент A(0) при сравнении оказывается не меньше, то он помещается на место, стоящее сразу за сравниваемым элементом.

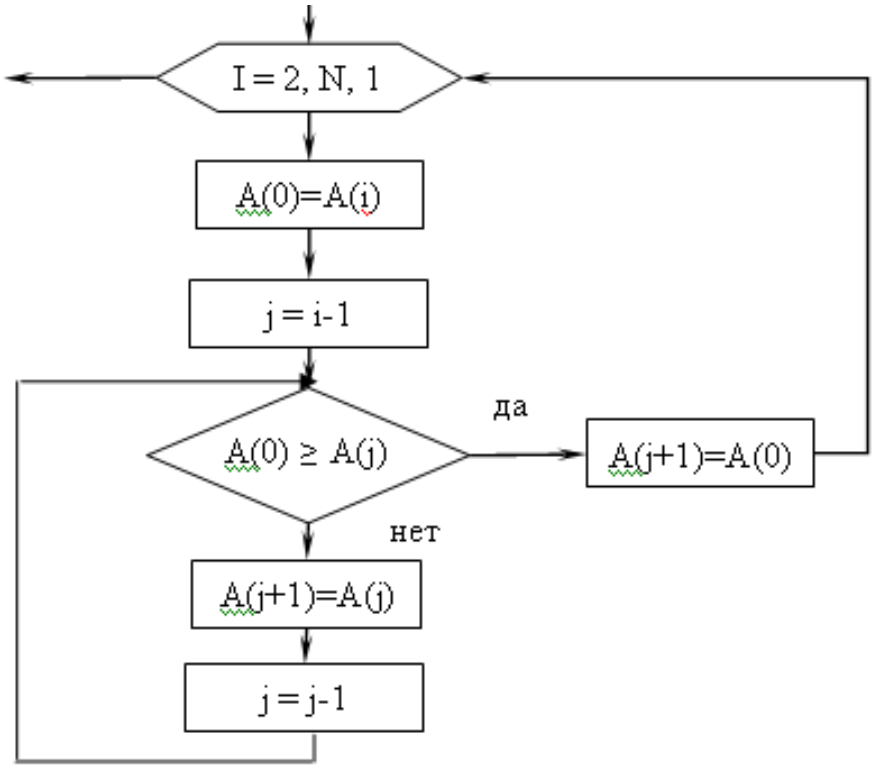


Рис. 1. Блок-схема сортировки методом прямого включения

На рис. 2 приведен пример выполнения сортировки методом прямого включения.

Исходная последовательность		4	5	2	2	4	8	6	7
	A (0)								
I									
I									
I									
I									
I									
I									
I									
Результат									

Рис. 2. Пример сортировки методом прямого включения

Сортировка прямым включением больше подходит для случая, когда сортируемые данные поступают последовательно (одно за другим).

Сортировка методом прямого выбора

Суть метода заключается в следующем. Выбирается наименьший элемент в "оставшейся" (неотсортированной) части и меняется местами с первым элементом (в этой же неотсортированной части). После этого длина неотсортированной части уменьшается на один элемент (на первый), и весь процесс продолжается уже с (n – 1) элементами, затем с (n – 2) элементами и т.д., до тех пор, пока не останется один, самый большой элемент.

Этот метод в некотором смысле противоположен методу прямого включения. В методе прямого включения на каждом шаге рассматривается только один очередной элемент и все элементы уже "готовой" части последовательности, среди которых отыскивается точка включения этого очередного элемента. А в методе прямого выбора для поиска одного (минимального) элемента просматривают все элементы неотсортированной части, и этот минимальный элемент помещается как очередной элемент в уже "готовую" часть.

Текстовый алгоритм метода:

1. Начало.

2. Выполнить цикл, пока i имеет значения от 1 до N – 1, шаг = 1:

а) поместим текущий (i-ый) элемент в какую-нибудь ячейку памяти (X) и запомним порядковый номер (i) текущего элемента (в переменной K);

б) выполнить цикл, пока j имеет значения от i + 1 (то есть от следующего за i элемента) до N, шаг = +1:

тело цикла: если X > A(j), то помещаем в ячейку X элемент A(j) и запоминаем его номер в ячейке K;

в) присвоить A(K) = A(i) и A(i) = X.

3. Конец.

На рис. 3 приведен пример выполнения сортировки методом прямого выбора.

Исходная последовательность	4	5	2	2	4	8	6	7
I = 1								
I = 2								
I = 3								
I = 4								
I = 5								
I = 6								
I = 7								

Рис. 3. Пример сортировки методом прямого выбора

На рисунке 4 приведена блок-схема сортировки методом прямого выбора.

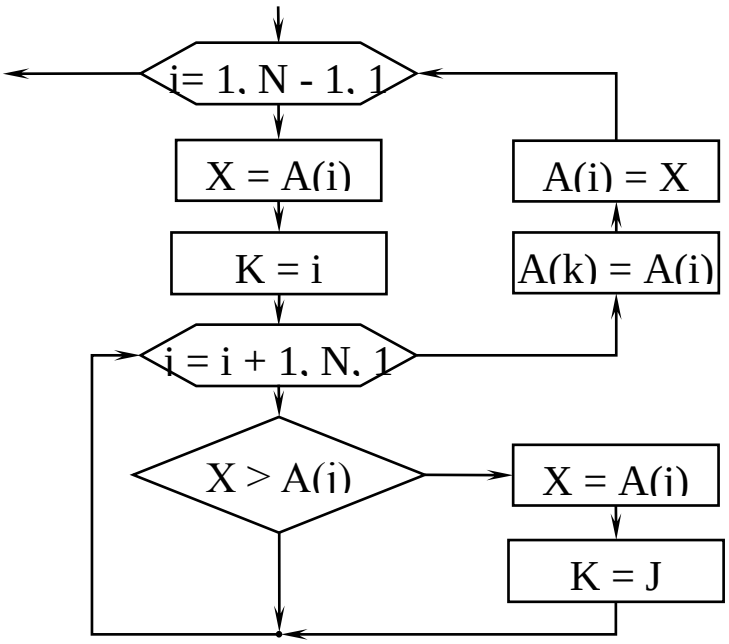


Рис. 4. Блок-схема сортировки методом прямого выбора

Сортировка прямым выбором подходит для случая, когда все данные находятся в памяти, а отсортированные данные последовательно выводятся.

Сортировка методом прямого обмена

Оба разбиравшихся выше метода можно тоже рассматривать как "обменные" сортировки.

Для рассматриваемого ниже метода обмен местами двух элементов представляет собой характерную особенность процесса.

Изложенный ниже алгоритм прямого обмена основан на сравнении и смене мест для пары соседних элементов и продолжении этого процесса до тех пор, пока не будут упорядочены все элементы.

Как и в методе прямого выбора, мы повторяем проходы по массиву, сдвигая каждый раз наименьший элемент "оставшейся" части последовательности к левому концу массива.

Если мы будем рассматривать массивы как вертикальные, а не горизонтальные построения, то элементы можно интерпретировать как пузырьки в чане с водой, причем вес каждого пузырька соответствует значению элемента. В этом случае при каждом проходе один пузырек как бы поднимается до уровня, соответствующего его весу. Такой метод широко известен под именем "пузырьковая сортировка".

Текстовый алгоритм:

1. Начало.

2. Выполнить цикл, пока i имеет значения от 2 до N, шаг = +1:

а) выполнить цикл, пока j имеет значения от N до i, шаг = -1:

тело цикла: если A(j-1) > A(j), то меняем местами эти два элемента.

3. Конец.

На рисунке 5 представлена блок-схема сортировки методом прямого обмена.

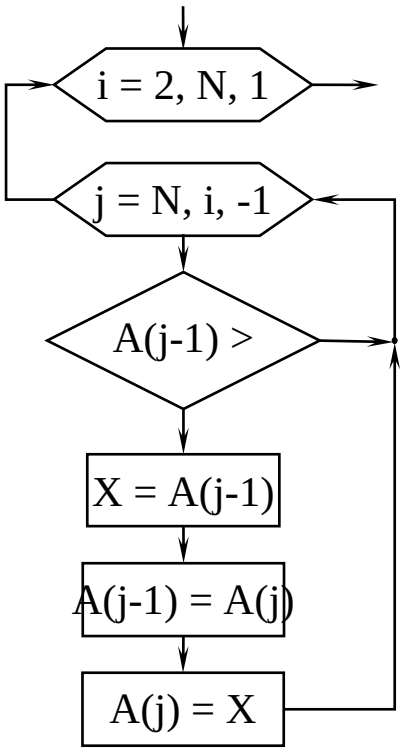


Рис. 5. Блок-схема сортировки методом прямого обмена

Если сравнивать свойства алгоритмов первых двух методов, то оказывается, что при сортировке прямым выбором число производимых сравнений больше, а число перемещений элементов – меньше, чем при сортировке прямым включением. В целом же большой разницы в эффективности этих методов нет.

Сортировка методом прямого обмена имеет меньшую эффективность. Число шагов просмотра с обменом $\leq (n - 2)$, и из-за того, что "легкие" элементы "всплывают" по несколько сразу, то кажется, что сортировка происходит быстро. Но с другой стороны, "тяжелые" элементы, если они находятся наверху, перемещаются вниз на одну позицию при каждом шаге, что увеличивает время сортировки. Число обменов элементов оказывается довольно большим.

На рисунке 6 приведен пример сортировки методом прямого обмена.

Исходная последовательность	= 1	= 2	= 3	= 4	= 5	= 6	= 7	= 8
44								
55								
12								
42								
94								
18								
06								
67								

Рис. 6. Пример сортировки методом прямого обмена

Сортировка бинарными включениями

Идея метода сходна с сортировкой прямыми включениями. Так же из неотсортированной части на i-ом шаге извлекается i-ый элемент, которому ищется место в уже "готовой" части последовательности. Однако процесс поиска места включения протекает в несколько раз быстрее.

Суть метода заключается в следующем:

Часть последовательности до испытуемого (i-го) элемента ("готовая" часть) делится пополам и i-й элемент сравнивается с элементом, стоящим на середине, после чего границы поиска уменьшаются в два раза. Получившийся полуинтервал делится пополам, и процесс повторяется до тех пор, пока не будет определено место включения i-го элемента. Затем происходит сдвиг вправо на одну позицию тех элементов, которые расположены от места включения до i-го элемента, освобождая таким образом позицию для i-ого элемента.

Текстовый алгоритм:

1. Начало.

2. Выполнить цикл, пока I имеет значение от 2 до N с шагом = 1

а) X = A(i), l = 1, r = i-1

б) Если l > r, то:

1) выполнить цикл, пока j имеет значение от (i-1) до l с шагом = -1

тело цикла: A(j + 1) = A(j)

2) присвоить A(l) = X

иначе:

1) присвоить m = (l + r) \ 2

2) если X < A(m), то r = m – 1 иначе l = m + 1

3) перейти к пункту б).

3. Конец.

На рисунке 7 приведен пример выполнения сортировки бинарными включениями.

Исходная последовательность	4	5	2	2	4	8	6	7
I = 2								
I = 3								
I = 4								
I = 5								
I = 6								
I = 7								
I = 8								
Результат								

Рис. 7. Пример выполнения сортировки
бинарными включениями

На рисунке 8 представлена блок-схема сортировки бинарными включениями.

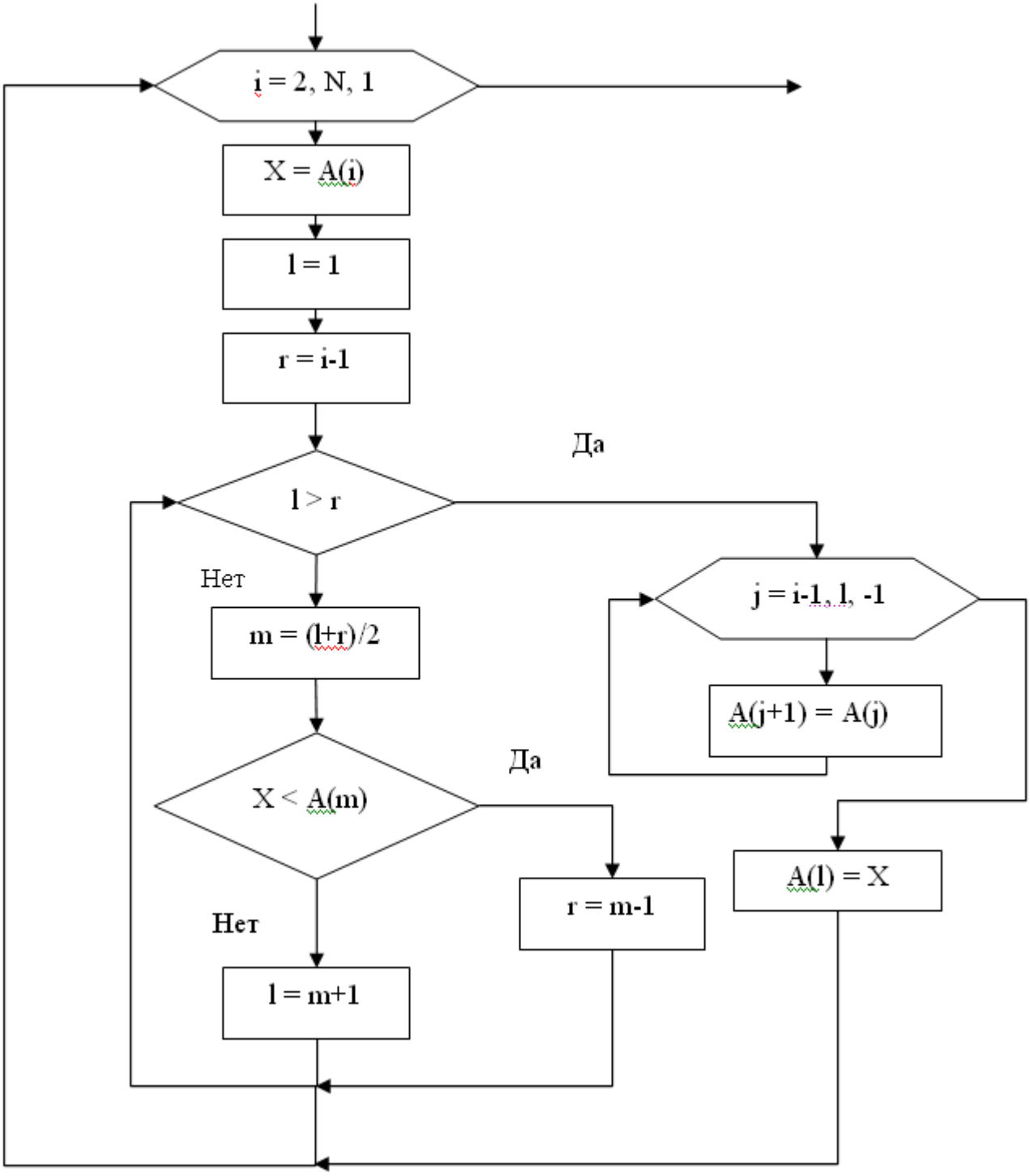


Рис. 8. Блок-схема сортировки бинарными включениями

Шейкер-сортировка

Как и алгоритм сортировки прямого обмена, алгоритм шейкер-сортировки основан на сравнении и смене мест пары соседних элементов. Однако в рассматриваемом методе каждый шаг состоит из двух этапов.

На первом этапе наименьший элемент неотсортированной части последовательности сдвигается к левому краю этой части, а наибольший элемент из оставшейся неотсортированной части сдвигается к правому краю этой части массива. После выполнения данных этапов неотсортированная часть массива уменьшается на два элемента. Шаги выполняются, пока не будет отсортирован весь массив.

Текстовый алгоритм:

1. Начало.
2. Присвоить переменной t (слева массива) значение 2, переменной r (справа массива) и переменной k – значение количества элементов массива.
3. Выполнить цикл, пока i имеет значение от r до t с шагом = -1:

а) если A(i-1) > A(i), то меняем местами эти два элемента и переменной k присваиваем значение = i.

4. Присвоить переменной t значение = k + 1.
5. Выполнить цикл, пока i имеет значение от t до r с шагом = 1:

а) если A(i-1) > A(i), то меняем местами эти два элемента и переменной k присваиваем значение = i.

6. Присвоить переменной r значение = k – 1.
7. Если t > r, то идти к пункту 8, иначе идти к пункту 3.
8. Конец.

На рисунке 9 представлен пример выполнения шейкер – сортировки по шагам.

Исходная последовательность		4	5	2	2	4	8	6	7
1-й шаг	1-й этап	06	44	55	12	42	94	18	67
	2-й этап	06	44	12	42	55	18	67	94
2-й шаг	1-й этап	06	12	44	18	42	55	67	94
	2-й этап	06	12	18	42	44	55	67	94
3-й шаг	1-й этап	06	12	18	42	44	55	67	94
Результат									

Рис. 9. Пример выполнения шейкер-сортировки

На рис. 10 приведена блок-схема шейкер-сортировки.

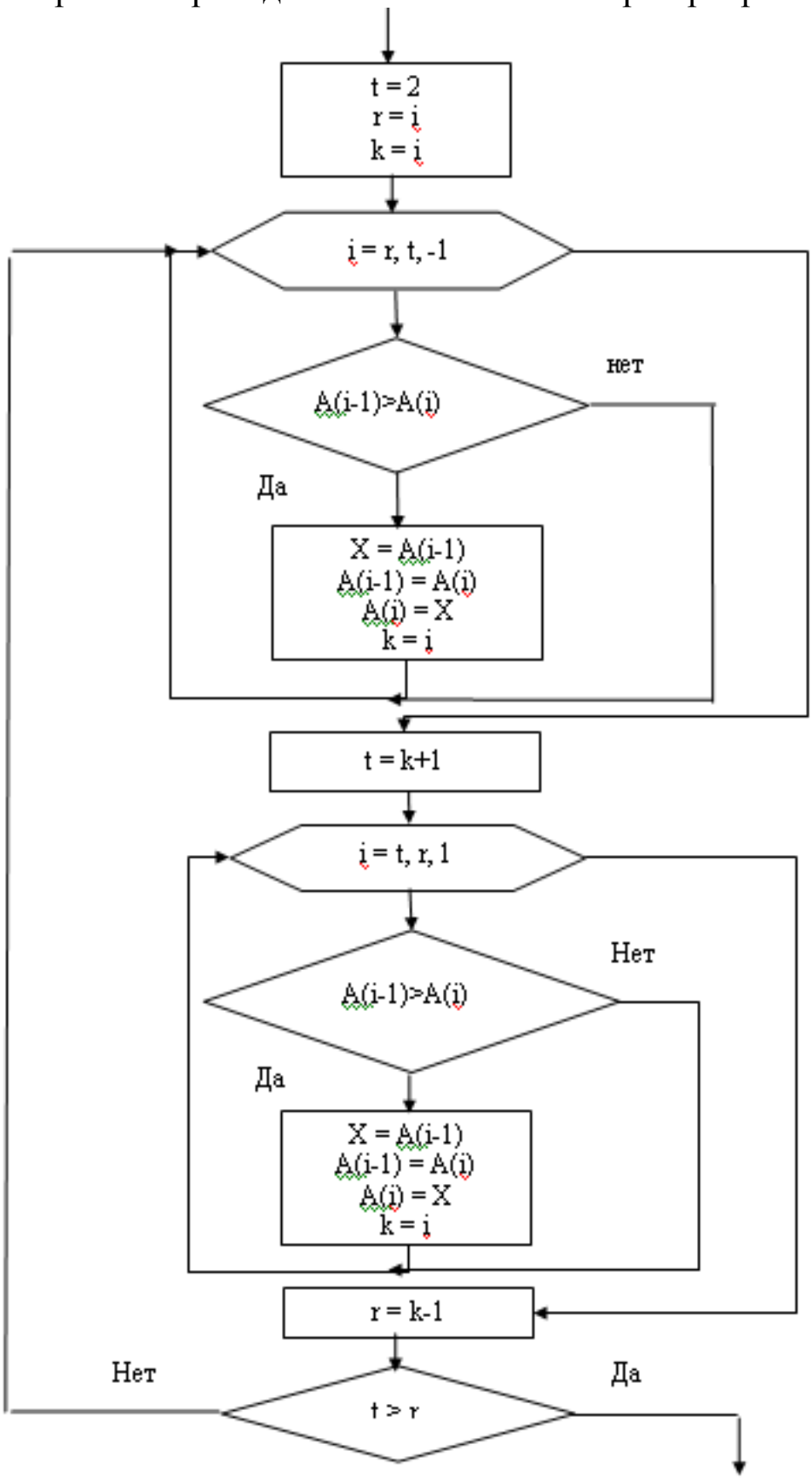


Рис. 10. Блок-схема шейкер-сортировки

Из примера, приведенного на рисунке 9 видно, что после первого шага длина неотсортированной части уменьшилась на два элемента, а после второго шага длина неотсортированной части вместо двух уменьшилась сразу на 4 элемента. Это дополнительное уменьшение обеспечивает переменная k, показывающая при каком значении i был совершен последний обмен местами двух элементов. Благодаря переменной k быстрее увеличивается и быстрее уменьшается левая (t = k + 1) и правая (r = k – 1) границы неотсортированной части.

3. ПОРЯДОК ВЫПОЛНЕНИЯ

1. Получить задание у преподавателя.
2. Выполнить задание в соответствии с вариантом.
3. Ответить на контрольные вопросы.

4. ЗАДАНИЕ

Выполнить сортировку одномерного массива, на языке программирования, указанного преподавателем в соответствии с заданным вариантом.

1. Известен список фамилий и рост учеников класса. Напечатать в порядке возрастания роста список детей, используя метод шейкер-сортировки.
2. Известен список спортсменов и результат их прыжков в длину. Напечатать общий список в порядке возрастания результата, используя метод сортировки бинарными включениями.
3. Известен список биатлонистов и результаты их стрельбы на двух огневых рубежах (попадания). На каждом рубеже пять мишеней. Напечатать общий список биатлонистов в порядке убывания результата на втором огневом рубеже, используя метод прямого выбора.
4. Известен список студентов группы и количество пропущенных часов каждым из студентов. Напечатать список студентов в порядке возрастания количества пропущенных часов (если пропуски имеют место), используя сортировку прямыми включениями.

5. Известен список рабочих и их месячный заработок. Напечатать список рабочих в порядке убывания зарплаты, используя метод сортировки прямыми включениями.

6. Задан числовой массив, состоящий из J элементов. Напечатать отрицательные числа в порядке возрастания по модулю, используя шейкер – сортировку.

7. Задан числовой массив, состоящий из J элементов. Напечатать положительные числа в порядке убывания, используя метод прямого выбора.

8. Известен список фамилий и вес студентов вашей группы.

Напечатать в порядке возрастания список студентов, вес которых не меньше среднего веса всей группы. Для этого использовать метод сортировки бинарными включениями.

9. Дан одномерный массив из I целых чисел. Напечатать отрицательные числа в порядке возрастания по модулю, а положительные числа в порядке убывания, используя метод прямого обмена.

10. Известен список студентов группы и количество пропущенных часов каждым из студентов. Напечатать в порядке возрастания тех, кто пропустил более 10 часов, используя метод сортировки прямыми включениями.

11. Известен список спортсменов и результат их прыжков в длину. Напечатать в порядке убывания тех, чей результат меньше 3 метров, используя сортировку прямого выбора.

12. Известен список спортсменов и результат их прыжков в длину. Напечатать в порядке возрастания тех, чей результат больше 3 метров, используя шейкер-сортировку.

13. Известен список фамилий и рост учеников класса. Напечатать в порядке убывания тех, чей рост меньше 160 см, используя сортировку бинарными включениями.

14. Известен список фамилий и рост учеников класса. Напечатать в порядке возрастания тех, чей рост больше 160 см, используя сортировку прямого обмена.

5. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое сортировка?
2. Объясните суть метода сортировки методом прямого включения.
3. Объясните суть метода сортировки методом прямого выбора.
4. Объясните суть сортировки методом прямого обмена.
5. Объясните суть сортировки бинарными включениями.
6. Объясните суть шейкер – сортировки.
7. Что обеспечивает дополнительное уменьшение неотсортированной части в шейкер – сортировки?

Лабораторная работа № 9
Работа со строковыми данными

1. ЦЕЛЬ РАБОТЫ

Изучение синтаксиса стандартных строковых процедур и функций, встроенных в язык программирования высокого уровня Visual Basic for Applications (VBA), получение навыков применения строковых процедур и функций в программах пользователя при работе с данными строкового типа.

2. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Общие положения

Описание и создание процедур

В офисном программировании целью работы, обычно, является создание системы документов. С системой документов связан программный проект, представляющий совокупность программных проектов, связанных с каждым отдельным документом. Проекты разных документов связаны между собой и могут иметь общие данные и общие процедуры и функции, доступные из разных проектов. Каждый проект состоит из модулей разного типа. Каждый модуль содержит объявления переменных, процедур и функций. Процедуры и функции являются минимальными модульными конструкциями, из которых строится программный проект.

Процедура (функция) – это программная единица VBA, включающая операторы описания ее локальных данных и исполняемые операторы. Обычно в процедуру объединяют регулярно выполняемую последовательность действий, решающую отдельную задачу или подзадачу. Особенность процедур VBA в том, что они могут использовать в качестве элементарных действий большое количество встроенных методов и функций, оперирующих с разнообразными объектами этой системы. Поэтому структура управления типичной процедуры прикладной офисной си-

стемы довольно проста: она состоит из последовательности вызовов встроенных процедур и функций, управляемой небольшим количеством условных операторов и циклов. Обычно ее размеры не должны превышать нескольких десятков строк.

Классификация процедур

Процедуры VBA можно классифицировать по нескольким признакам: по способу использования (вызова) в программе, по способу запуска процедуры на выполнение, по способу создания кода процедуры, по месту нахождения кода процедуры в проекте.

Процедуры VBA подразделяются на подпрограммы и функции. Первые описываются ключевым словом Sub, вторые – Function. Различие между этими видами процедур только синтаксическое, так как преобразовать процедуру одного вида в эквивалентную процедуру другого вида не сложно.

По способу создания процедуры делятся на:

– обычные, разрабатываемые "вручную";

– процедуры, код которых создается автоматически генератором макросов (MacroRecoder).

Это разделение можно считать условным, так как многие процедуры создаются генератором макросов, а затем изменяются и дописываются вручную.

По способу запуска процедур на выполнение можно выделить в отдельную группу процедуры, запускаемые автоматически при наступлении того или иного события (процедуры обработки событий).

По положению в проекте различаются процедуры:

– находящиеся в специальных программных единицах – стандартных модулях, модулях классов и модулях, связанными с объектами;

– реагирующие на события.

Еще один специальный тип процедур – процедуры-свойства Property Let, Property Set и Property Get. Они служат для задания и получения значений закрытых свойств класса.

Главное назначение процедур во всех языках программирования состоит в том, что при их вызове они изменяют состояние

программного проекта, – изменяют значения переменных (свойства объектов), описанных в модулях проекта. У процедур VBA сфера действия шире. Их главное назначение состоит в изменении состояния системы документов, частью которого является изменение состояния самого программного проекта. Поэтому процедуры VBA оперируют, в основном, с объектами Office.

Есть два способа, с помощью которых процедура получает и передает информацию, изменяя тем самым состояние системы документов. Первый и основной способ состоит в использовании параметров процедуры. При вызове процедуры ее аргументы, соответствующие входным параметрам получают значение, то есть процедура получает информацию от внешней среды, и в результате работы процедуры формируются значения выходных параметров, переданных ей по ссылке. Тем самым изменяется состояние проекта и документов. Второй способ состоит в использовании процедурой глобальных переменных и объектов, как для получения, так и для передачи информации.

Синтаксис процедур и функций

Описание процедуры Sub в VBA имеет следующий вид:

```
[Private | Public] [Static] Sub имя([список-аргументов])
тело-процедуры
End Sub
```

Ключевое слово Public в заголовке процедуры используется, чтобы объявить процедуру общедоступной, то есть дать возможность вызывать ее из всех других процедур всех модулей любого проекта. Если модуль, в котором описана процедура, содержит закрывающий оператор Option Private, процедура будет доступна лишь модулям своего проекта. Альтернативный ключ Private используется, чтобы закрыть процедуру от всех модулей, кроме того, в котором она описана. По умолчанию процедура считается общедоступной.

Ключевое слово Static означает, что значения локальных (объявленных в теле процедуры) переменных будут сохраняться в промежутках между вызовами процедуры (используемые процедурой глобальные переменные, описанные вне ее тела, при этом не сохраняются).

Параметр имя – это имя процедуры, удовлетворяющее стандартным условиям VBA на имена переменных.

Необязательный параметр список-аргументов – это последовательность разделенных запятыми переменных, задающих передаваемые процедуре при вызове параметры. Аргументы (формальные параметры), задаваемые при описании процедуры, всегда представляют только имена (идентификаторы). В то же время при вызове процедуры ее аргументы – фактические параметры могут быть не только именами, но и выражениями.

Последовательность операторов (тело процедуры) задает программу выполнения процедуры. Тело процедуры может включать как "пассивные" операторы объявления локальных данных процедуры (переменных, массивов, объектов и др.), так и "активные" – они изменяют состояния аргументов, локальных и внешних (глобальных) переменных и объектов. В тело может входить оператор Exit Sub, приводящий к немедленному завершению процедуры и передаче управления в вызывающую программу. Каждая процедура в VBA определяется отдельно от других, то есть тело одной процедуры не может включать описания других процедур и функций.

Рассмотрим подробнее структуру одного аргумента из списка-аргументов:

[Optional] [ByVal | ByRef] [ParamArray] переменная[()] [As тип] [= значение-по-умолчанию]

Ключевое слово Optional означает, что заданный им аргумент является возможным, но необязательным, – его необязательно задавать в момент вызова процедуры. Для таких аргументов можно задать значение по умолчанию. Необязательные аргументы всегда помещаются в конце списка аргументов.

Альтернативные ключи ByVal и ByRef определяют способ передачи аргумента в процедуру.

ByVal означает, что аргумент передается по значению, то есть при вызове процедуры будет создаваться локальная копия переменной с начальным передаваемым значением и изменения этой локальной переменной во время выполнения процедуры не отразятся на значении переменной, передавшей свое значение в процедуру при вызове. Передача по значению возможна только для входных параметров, которые передают информацию в про-

цедуру, но не являются результатами. Для таких параметров передача по значению удобнее, чем передача по ссылке, так как в момент вызова аргумент может быть задан сколь угодно сложным выражением. Входные параметры, являющиеся объектами, массивами или переменными пользовательского типа, передаются по ссылке, что позволяет избежать создания копий. Выражения над такими аргументами недопустимы, поэтому передача по значению теряет свой смысл.

ByRef означает, что аргумент передается по ссылке, то есть все изменения значения передаваемой переменной при выполнении процедуры будут непосредственно происходить с переменной-аргументом из вызвавшей данную процедуру программы. В VBA по умолчанию аргументы передаются по ссылке (ByRef). VBA допускает, чтобы фактическое значение аргумента, передаваемого по ссылке, было константой или выражением соответствующего типа. В таком случае этот аргумент рассматривается как передаваемый по значению, и никаких сообщений об ошибке не выдается, даже если этот аргумент встречается в левой части присвоения.

Процедура VBA допускает возможность иметь необязательные аргументы, которые можно опускать в момент вызова. Обобщением такого подхода является возможность иметь переменное, заранее не фиксированное число аргументов. Достигается это за счет того, что один из параметров (последний в списке) может задавать массив аргументов, – в этом случае он задается с описателем ParamArray. Если список-аргументов включает массив аргументов ParamArray, ключ Optional использовать в списке нельзя. Ключевое слово ParamArray. может появиться перед последним аргументом в списке, чтобы указать, что этот аргумент – массив с произвольным числом элементов типа Variant. Перед ним нельзя использовать ключи ByVal, ByRef или Optional.

Переменная – это имя переменной, представляющей аргумент.

Если после имени переменной заданы круглые скобки, то это означает, что соответствующий параметр является массивом.

Параметр тип задает тип значения, передаваемого в процедуру. Он может быть одним из базисных типов VBA (не допускаются только строки String с фиксированной длиной). Обяза-

тельные аргументы могут также иметь тип определенной пользователем записи или класса. Если тип аргумента не указан, то по умолчанию ему приписывается тип Variant.

Для необязательных (Optional) аргументов можно явно задать значение по умолчанию. Это константа или константное выражение, значение которого передается в процедуру, если при ее вызове соответствующий аргумент не задан. Для аргументов типа объект (Object) в качестве значения по умолчанию можно задать только Nothing.

Синтаксис определения процедур-функций похож на определение обычных процедур:

[Public | Private] [Static] Function имя [(список-аргументов)] [As тип-значения]

тело-функции

End Function

Отличие лишь в том, что вместо ключевого слова Sub для объявления функции используется ключевое слово Function, а после списка аргументов следует указать параметр тип-значения, определяющий тип возвращаемого функцией значения. В теле функции должен быть использован оператор присвоения вида:

имя = выражение

Здесь, в левой части оператора стоит имя функции, а в правой – значение выражения, задающего результат вычисления функции. Если при выходе из функции переменной имя значение явно не присвоено, функция возвращает значение соответствующего типа, определенное по умолчанию. Для числовых типов это 0, для строк – строка нулевой длины (""), для типа Variant функция вернет значение Empty, для ссылок на объекты – Nothing.

Чтобы немедленно завершить вычисления функции и выйти из нее, в теле функции можно использовать оператор Exit Function.

Основное отличие процедур от функций состоит в способе их использования в вызывающей программе. Следующая функция возвращает аргумент, возведенный в куб:

```
Function cube(ByVal N As Integer) As Long
cube = N*N*N
End Function
```

Вызов этой функции может иметь вид

```
Dim x As Integer, y As Integer
y = 2
x = cube(y+3)
```

При преобразовании функции в процедуру появляется дополнительный параметр, необходимый для задания результата. Поэтому у эквивалентной процедуры Cube1 появляется еще один аргумент:

```
Sub cube1(ByVal N As Integer, ByRef C As Long)
C = N*N*N              ' получение результата в переменной, заданной по ссылке
End Sub
```

Ее также можно использовать для возведения в куб:

```
cube1(y+3, x)
```

Эта взаимозаменяемость не означает, что безразлично, какой вид процедур использовать в программе. Если бы выражение, в котором участвует функция, было сложнее, например,

```
x = cube(y)+sin(cube(x))
```

то его вычисление с помощью процедуры Cube1 потребовало бы выполнения нескольких операторов и ввода дополнительных переменных:

```
cube1(y,z)
cube1(x,u)
x = z+sin(u)
```

Функции с побочным эффектом

В классическом варианте все аргументы функции являются входными параметрами, и единственный результат вычисления функции – это ее возвращаемое значение, примером является функция Cube. Однако чаще всего, используются функции с побочным эффектом, то есть такие функции, которые помимо получения значения функции изменяют значения некоторых результирующих параметров, передаваемых функции по ссылке. Например:

```
Public Function SideEffect(ByVal X As Integer, ByRef Y As Integer) As Integer
SideEffect = X+Y
Y = Y+1
End Function
```

```
Public Sub TestSideEffect()
Dim X As Integer, Y As Integer, Z As Integer
X = 3: Y = 5
Z = X+Y+SideEffect(X, Y)
Debug.Print X, Y, Z
X = 3: Y = 5
Z = SideEffect(X, Y)+X+Y
Debug.Print X, Y, Z
End Sub
```

Вот результаты вычислений:

```
3          6          16
3          6          17
```

В данном примере результат вычисления суммы трех слагаемых зависит от порядка их записи. Более того, результат вычисления будет непредсказуем, поскольку VBA может для увеличения эффективности изменять порядок действий при вычислении арифметического выражения. Поэтому лучше не использовать в выражении вызов функции с побочным эффектом, изменяющей значения входящих в него переменных.

Создание процедуры

Для создания новой процедуры, текст которой пишется вручную, нужно:

1) открыть в окне проектов "Проект-(VBA)Project" (Project Explorer) папку с модулем (формой, документом, рабочим листом и т.п.), к которому требуется добавить процедуру, и, щелкнув этот модуль, открыть окно редактора с кодами процедур модуля;

2) перейти в редактор, набрать ключевое слово (Sub, Function или Property), имя процедуры и ее аргументы; затем нажмите клавишу Enter, и VBA поместит ниже строку с соответствующим закрывающим оператором (End Sub, End Function, End Property);

3) написать текст процедуры между ее заголовком и закрывающим оператором.

Можно автоматизировать работу, вызвав диалоговое окно "Вставка процедуры" (Insert Procedure). Последовательность действий в этом случае следующая:

1) выбрать в меню "Вставка" (Insert) команду "Процедура" (Procedure);

2) в поле "Имя" (Name) появившегося окна "Вставка процедуры" (Insert Procedure) ввести имя процедуры;

3) указать в группе кнопок-переключателей "Тип" (Type) тип создаваемой процедуры: Подпрограмма (Sub), Функция (Function) или Свойство (Property);

4) указать в группе кнопок-переключателей "Область определения" (Scope) вид доступа к процедуре: Общая (Public) или Личная (Private);

5) пометить, если нужно, флажок "Все локальные переменные считать статическими" (All Local Variables as Statics), чтобы в заголовок процедуры добавился ключ Static;

6) щелкнуть кнопку "OK" – в окне редактора появится заготовка процедуры, состоящая из ее заголовка (без параметров) и закрывающего оператора;

7) добавить параметры в заголовок процедуры и написать текст процедуры между ее заголовком и закрывающим оператором.

Создание процедур обработки событий

VBA является языком, в котором, как и в большинстве современных объектно-ориентированных языков, реализована концепция программирования, управляемого событиями (event driven programming). Здесь нет понятия программы, которая начинает выполняться от Begin до End. Пользователи системы документов и операционная система могут инициировать события в мире объектов. В ответ на возникновение события операционная система посылает сообщение соответствующему объекту. Реакцией объекта на получение сообщения является вызов процедуры – обработчика события. Задача программиста сводится к написанию обработчиков событий для объектов. Причем, все обычные процедуры и функции VBA, о которых мы говорили выше, вызываются прямо или косвенно из процедур обработки событий, если только речь не идет о режиме отладки. Именно эти процедуры приводят к последовательности вызовов обычных процедур и функций.

С каждым из объектов Office связан набор событий, на которые он может реагировать. Процедуры обработки этих событий располагаются в модулях, связанных с объектами, реагирующими на события. Для кнопок меню, у которых есть только один обработчик события, соответствующая процедура может находиться в стандартном модуле или модуле макросов. Office позволяет при описании собственных классов, создаваемых программистом, задать определенный набор событий. Обработчики событий таких объектов создаются по определенной технологии. Например, процедура, которая будет пять раз подавать звуковой сигнал при закрытии документа:

```
Private Sub Document_Close()
Dim I As Integer
For I = 1 To 5              ' 5 раз подается
Beep                ' звуковой сигнал
Next I
End Sub
```

Эта процедура находится в модуле, связанном с объектом ThisDocument, и вызывается в момент закрытия документа.

Вызовы процедур и функций

Вызовы процедур Sub

Вызов обычной процедуры Sub из другой процедуры можно оформить разными способами.

Первый способ:

имя список фактических параметров,

где имя – имя вызываемой процедуры; список фактических параметров – список аргументов, передаваемых процедуре; он должен соответствовать списку аргументов, заданному в описании процедуры.

Задать список параметров можно разными способами.

В простейшем случае в нем перечисляются через запятую значения передаваемых процедуре аргументов в том же порядке, что и в списке аргументов из заголовка процедуры.

Может оказаться, что в одном проекте несколько модулей содержат процедуры с одинаковыми именами. Для различия этих

процедур нужно при их вызове указывать имя процедуры через точку после имени модуля, в котором она определена. Например, если каждый из двух модулей Mod1 и Mod2 содержит определение процедуры ReadData, а в процедуре MyProc нужно воспользоваться процедурой из Mod2, этот вызов имеет вид:

```
Sub Myproc()
Mod2.ReadData
End Sub
```

Если требуется использовать процедуры с одинаковыми именами из разных проектов, нужно добавить к именам модуля и процедуры имя проекта. Например, если модуль Mod2 входит в проект MyBook, тот же вызов можно уточнить так:

```
MyBooks.Mod2.ReadData
```

Второй способ вызова процедур связан с использованием оператора Call. В этом случае вызов процедуры выглядит так:

```
Call имя(список фактических параметров)
```

Обратите внимание на то, что в этом случае список фактических параметров заключен в круглые скобки, а в первом случае – нет. Если процедура VBA имеет только один параметр, то она может быть вызвана без оператора Call и с использованием круглых скобок, не сообщая об ошибке вызова, но возвращает не правильный результат. Например:

```
Public Sub MyInc(ByRef X As Integer)
X = X+1
End Sub

Public Sub TestInc()
Dim X As Integer
X = 1
'Вызов процедуры с параметром, заключенным в скобки,
'синтаксически допустим, но работает не корректно!
MyInc (X)
Debug.Print X
'Корректный вызов
MyInc X
Debug.Print X
'Это тоже корректный вызов
Call MyInc(X)
```

```
Debug.Print X
End Sub
```

Вот результаты ее работы:

```
1
2
3
```

Несмотря на то, что при первом вызове процедура вызывается без ошибок и увеличивает значение результата, но по завершении ее работы значение аргумента не изменяется. В этой ситуации не действует описатель ByRef, вызов идет так, как будто параметр описан с описателем ByVal.

Если же процедура имеет более одного параметра, то попытка вызвать ее, заключив параметры в круглые скобки и не предварив этот вызов ключевым словом Call, приводит к синтаксической ошибке. Например:

```
Public Sub SumXY(ByVal X As Integer, ByVal Y As Integer, ByRef Z As Integer)
Z = X+Y
End Sub

Public Sub TestSumXY()
Dim a As Integer, b As Integer, c As Integer
a = 3: b = 5
'SumXY (a, b, c)        'Синтаксическая ошибка
SumXY a, b, c
Debug.Print c
End Sub
```

В этом примере некорректный вызов процедуры SumXY будет обнаружен на этапе проверки синтаксиса.

Рассмотрим еще одну особенность вызова VBA процедур, связанную с аргументами, передаваемыми по ссылке. Как правило, в языках программирования для таких аргументов возможное значение фактического параметра ограничивается, – он должен быть именем переменной, ссылка на которую передается процедуре. VBA допускает возможность задания для таких аргументов констант и выражений в момент вызова.

Например, процедура CompVal с четырьмя аргументами, которая в зависимости от положительности z возвращает в пере-

менной y либо увеличенное, либо уменьшенное на сто значение x и сообщает об этом в строковой переменной w, определена следующим образом.

```
Sub CompVal(ByVal x As Single, ByRef y As Single, _
ByVal z As Integer, ByRef w As String)
If z > 0 Then              ' увеличение
y = x+100
w = "increase"
Else                       ' уменьшение
y = x-100
w = "decrease"
End If
End Sub
```

Рассмотрим процедуру TestCompVal, в которой несколько раз вызывается процедура CompVal:

```
Sub TestCompVal()
Dim a As Single
Dim b As Single
Dim n As Integer
Dim S As String
n = 5 : a = 7.4                        ' значения параметров
CompVal a, b, n, S                     ' 1-ый вызов
Debug.Print b, S
CompVal 7.4, b, 5, S                   ' 2-ой вызов
Debug.Print b, S
CompVal 0, 0, 0, S                     ' 3-ий вызов
Debug.Print b, S
CompVal 0, 0, 0, "В чем дело?"         ' 4-ый вызов
Debug.Print b, S
End Sub
```

В результате выполнения этой процедуры будут напечатаны следующие результаты:

```
107,4          increase
107,4          increase
107,4          decrease
107,4          decrease
```

Первые два вызова корректны. Следующие два вызова хотя и допустимы в языке VBA, но приводят к тому, что параметры,

переданные по ссылке, не меняют своих значений в ходе выполнения процедуры и, по существу, вызов ByRef по умолчанию заменяется вызовом ByVal.

Вызовы функций

Оформление вызова функции зависит от того, требуется ли использовать ее значение в вызывающей процедуре. Если Вы хотите передать вычисляемое функцией значение в переменную или применить его в выражении правой части оператора присвоения, то вызов пользовательской функции имеет тот же вид, что и вызов встроенной функции, например, sin(x). При вызове указывается имя функции, а после него идет заключенный в круглые скобки список фактических параметров. Например, если заголовок функции MyFunc:

```
Function Myfunc(Name As String, Age As Integer, Newdate As Date) As Integer
```

использовать ее значение можно с помощью вызовов:

```
val= Myfunc("Alex",25, "10/04/97")
```

или

```
x = sqrt(Myfunc("Alex",25, "10/04/97")) + x
```

Если же значение, вычисляемое функцией, нас не интересует и нужно воспользоваться лишь ее побочными эффектами, вызов функции может иметь ту же форму, что и вызов процедуры Sub. Например:

```
Myfunc "Alex",I, "10/04/97"
```

или:

```
Call Myfunc(Myson, 25, DateOfArrival)
```

Использование именованных аргументов

В предыдущих примерах фактические параметры вызова процедуры или функции располагались в том же порядке, что и формальные параметры в ее заголовке. Это не всегда удобно, особенно если некоторые аргументы необязательны (Optional). VBA позволяет указывать значения аргументов в произвольном порядке, используя их имена. При этом после имени аргумента ставятся двоеточие и знак равенства, после которого помещается

значение аргумента (фактический параметр). Например, вызов рассмотренной выше процедуры-функции MyFunc может выглядеть так:

Myfunc Age:= 25, Name:= "Alex", Newdate:= DateOfArrival

Удобство такого способа особенно проявляется при вызове процедур с необязательными аргументами, которые всегда помещаются в конец списка аргументов в заголовке процедуры. Пусть, например, заголовок процедуры ProcEx:

Sub ProcEx(Name As String, Optional Age As Integer, Optional City = "Москва")

Список ее аргументов включает один обязательный аргумент Name и два необязательных: Age и City, – причем для последнего задано значение по умолчанию "Москва". Если при вызове этой процедуры второй аргумент не требуется, то при вызове, не использующем именованных параметров, сам параметр опускается, но, выделяющая его запятая, должна оставаться:

ProcEx "Оля",,"Тверь"

Вместо этого можно использовать вызов с именами аргументов:

ProcEx City:="Тверь", Name:="Оля"

В данном случае не требуется заменять пропущенный аргумент запятыми и соблюдать определенный порядок следования аргументов. Если некий необязательный аргумент не задан при вызове, то вместо него подставляется значение, определенное пользователем по умолчанию, если и такое не определено, подставляется значение, определенное по умолчанию для соответствующего типа. Например, при вызове:

ProcEx Name:="Оля"

в качестве значения аргумента Age в процедуру передастся 0, а в качестве аргумента City – явно заданное по умолчанию значение "Москва".

Чтобы процедура "узнала", что ей при вызове передан необязательный аргумент, можно воспользоваться функцией IsMissing. Она по имени аргумента возвращает логическое значение True, когда значение аргумента не передано в процедуру, и False, если аргумент задан. Это применимо только в том случае, если параметр имеет тип Variant. Для всех остальных типов данных полагается, что в процедуру всегда передано значение пара-

метра, явно или неявно заданное по умолчанию. Поэтому, если такая проверка необходима, то параметр должен иметь тип Variant. Для массива аргументов ParamArray функция IsMissing всегда возвращает False, и для установления его пустоты нужно проверять, что верхняя граница индекса меньше нижней.

Рассмотрим функцию от двух аргументов, второй из которых необязателен:

```
Function TwoArgs(I As Integer, Optional X As Variant) As Variant
If IsMissing(X) Then
' если 2-ой аргумент отсутствует, то вернуть 1-ый.
TwoArgs = I
Else
' если 2-ой аргумент есть, то вернуть их произведение
TwoArgs = I*X
End If
End Function
```

Вот результаты нескольких вызовов этой функции в окне отладки:

```
? TwoArgs(5,7)
35
? TwoArgs(5.5)
6
? TwoArgs(5, 5.5)
27,5
? TwoArgs(5, "6")
30
```

Аргументы, являющиеся массивами

Аргументы процедуры могут быть массивами. Процедуре передается имя массива, а размерность массива определяется встроенными функциями LBound и UBound. Приведем пример процедуры, вычисляющей скалярное произведение векторов:

```
Public Function ScalarProduct(X() As Integer, Y() As Integer) As Integer
'Вычисляет скалярное произведение двух векторов.
'Предполагается, что границы массивов совпадают.
Dim i As Integer, Sum As Integer
```

```
Sum = 0
For i = LBound(X) To UBound(X)
Sum = Sum+X(i)*Y(i)
Next i
ScalarProduct = Sum
End Function
```

Оба параметра процедуры, передаваемые по ссылке, являются массивами, работа с которыми в теле процедуры не представляет затруднений, благодаря тому, что функции LBound и UBound позволяют установить границы массива по любому измерению. Приведем программу, в которой вызывается функция ScalarProduct:

```
Public Sub TestScalarProduct()
Dim A(1 To 5) As Integer
Dim B(1 To 5) As Integer
Dim C As Variant
Dim Res As Integer
Dim i As Integer
C = Array(1, 2, 3, 4, 5)
For i = 1 To 5
A(i) = C(i-1)
Next i
C = Array(5, 4, 3, 2, 1)
For i = 1 To 5
B(i) = C(i-1)
Next i
Res = ScalarProduct(A, B)
Debug.Print Res
End Sub
```

3. ПОРЯДОК ВЫПОЛНЕНИЯ

1. Получить задание у преподавателя.
2. Выполнить задание в соответствии с вариантом.
3. Ответить на контрольные вопросы.

4. ЗАДАНИЕ

Выполнить сортировку одномерного массива в соответствии с заданным вариантом. Алгоритм сортировки выполнить в виде отдельной процедуры.

1. Известен список фамилий и рост учеников класса.

Напечатать в порядке возрастания роста список детей, используя метод шейкер-сортировки.

2. Известен список спортсменов и результат их прыжков в длину.

Напечатать общий список в порядке возрастания результата, используя метод сортировки бинарными включениями.

3. Известен список биатлонистов и результаты их стрельбы на двух огневых рубежах (попадания). На каждом рубеже пять мишеней.

Напечатать общий список биатлонистов в порядке убывания результата на втором огневом рубеже, используя метод прямого выбора.

4. Известен список студентов группы и количество пропущенных часов каждым из студентов.

Напечатать список студентов в порядке возрастания количества пропущенных часов (если пропуски имеют место), используя сортировку прямыми включениями.

5. Известен список рабочих и их месячный заработок.

Напечатать список рабочих в порядке убывания зарплаты, используя метод сортировки прямыми включениями.

6. Задан числовой массив, состоящий из J элементов.

Напечатать отрицательные числа в порядке возрастания по модулю, используя шейкер-сортировку.

7. Задан числовой массив, состоящий из J элементов.

Напечатать положительные числа в порядке убывания, используя метод прямого выбора.

8. Известен список фамилий и вес студентов вашей группы.

Напечатать в порядке возрастания список студентов, вес которых не меньше среднего веса всей группы. Для этого использовать метод сортировки бинарными включениями.

9. Дан одномерный массив из I целых чисел.

Напечатать отрицательные числа в порядке возрастания по модулю, а положительные числа в порядке убывания, используя метод прямого обмена.

10. Известен список студентов группы и количество пропущенных часов каждым из студентов.

Напечатать в порядке возрастания тех, кто пропустил более 10 часов, используя метод сортировки прямыми включениями.

11. Известен список спортсменов и результат их прыжков в длину.

Напечатать в порядке убывания тех, чей результат меньше 3 м, используя сортировку прямого выбора.

12. Известен список спортсменов и результат их прыжков в длину.

Напечатать в порядке возрастания тех, чей результат больше 3 м, используя шейкер – сортировку.

13. Известен список фамилий и рост учеников класса.

Напечатать в порядке убывания тех, чей рост меньше 160 см, используя сортировку бинарными включениями.

14. Известен список фамилий и рост учеников класса.

Напечатать в порядке возрастания тех, чей рост больше 160 см, используя сортировку прямого обмена.

5. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Чем отличается процедура типа "Sub" от процедуры типа "Function"?
2. Что обозначают ключевые слова ByVal и ByRef при описании процедур?
3. Что обозначает ключевое слово Optional при описании процедур?
4. Функции с побочным эффектом.
5. Создание новой процедуры.
6. Вызов процедуры типа "Sub".
7. Вызов процедуры типа "Function".
8. Использование именованных аргументов.
9. Как использовать аргументы, являющиеся массивами?

Лабораторная работа № 10
Процедуры и функции

1. ЦЕЛЬ РАБОТЫ

Изучение синтаксиса стандартных строковых процедур и функций, встроенных в язык программирования высокого уровня Visual Basic for Applications (VBA), получение навыков применения строковых процедур и функций в программах пользователя при работе с данными строкового типа.

2. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Общие положения

VBA – великолепное сочетание одного из самых простых языков программирования BASIC со специальным механизмом, позволяющим программам, написанным на этом языке, обращаться к объектам всех основных приложений Microsoft Office — Excel, Word, Power Point, Access и прочих.

VBA – мощное средство разработки настоящих полнофункциональных программ, работающих в среде приложений Microsoft Office.

VBA поддерживает важнейшие современные концепции разработки программных систем — объектно-ориентированное и событийно-управляемое программирование, а также важнейшие технологии – поддержку элементов управления на базе ActiveX, интеграцию с базами данных, системами электронной почты и Интернетом.

VBA содержит все средства, характерные для современных средств разработки программного обеспечения: интегрированную среду разработки, конструктор форм, мощный отладчик, не уступающий по возможностям отладчикам некоторых современных компиляторов.

Данные в VBA характеризуются своими типами, которые определяют:

- формат представления данных в памяти компьютера;
- область возможных значений;
- множество допустимых операций, применимых к данным.

В свою очередь типы данных делятся на простые (встроенные и определяемые) и на структурные.

Переменную можно объявлять или не объявлять, и тогда тип ей будет присвоен по умолчанию, или по первой букве имени, или по специальному символу объявления типа, которым может заканчиваться имя. Явно объявить переменную можно как в начале блока, так и в том произвольном месте, где возникла необходимость использовать новую переменную.

Соображения повышения надежности программ рекомендуют использовать методику, при которой все переменные объявляются явно и, как правило, в начале блока. При включении в начало модуля оператора Option Explicit (опция "Явно") предварительное объявление переменных становится обязательным.

При объявлении переменной определяется ее тип и область видимости – область, где имя переменной видимо и, значит, возможен доступ к её значению. Переменные можно объявлять на двух уровнях – уровне процедуры и уровне модуля. Для объявления переменных используются операторы Dim, Public, Private и Static. Первый можно использовать на обоих уровнях, Public и Private – на уровне модуля, Static – только на уровне процедуры.

Переменные, объявленные на уровне процедуры, называются локальными по отношению к данной процедуре. Их областью видимости является только та процедура, в которой они объявлены. Локальные переменные можно объявлять в любом месте процедуры, но до выполняемых операторов, использующих эти переменные.

Переменные уровня модуля являются глобальными. Они объявляются в разделе Declarations, который есть у каждого модуля. Область видимости глобальных переменных может распространяться:

- на все процедуры одного модуля, в котором они объявлены; такие глобальные переменные, называемые закрытыми (Private), должны быть объявлены на уровне модуля либо оператором Private либо оператором Dim;
- на все приложение – все процедуры всех модулей данного приложения, такие глобальные переменные, называемые открытыми, должны быть объявлены оператором Public.

Локальные переменные уровня процедуры могут быть объявлены оператором Static, что делает их статическими.

Для локальных переменных, объявленных таким образом, изменяется механизм хранения их (переменных) в оперативной памяти и время существования соответствует времени с момента первого запуска процедуры/функции, в которой определена статическая переменная, до окончания работы программы. Обычные локальные переменные создаются и инициализируются (им присваивается значение) в "своей" процедуре/функции, видимы только в ней и удаляются при завершении этой процедуры/функции (память под переменные отводится при входе в процедуру/функцию, а при выходе она освобождается). Область видимости статической переменной такая же, но время существования иное, так как статическая переменная не удаляется из памяти при завершении процедуры/функции, просто переменная становится временно недоступной. Поэтому при повторном вызове процедуры/функции статические переменные восстанавливают те значения, которые они имели при завершении работы процедуры/функции при предыдущем вызове. Статические переменные – это хранители информации между многократными вызовами одной и той же процедуры/функции. Чтобы статические переменные имели смысл, необходима первоначальная инициализация переменных – они должны иметь хоть какие-то значения уже при первом вхождении в процедуру/функцию. Вот как VBA инициализирует переменные в момент их объявления:

- 0 – для числовых значений;
- пустая строка («») – для строк переменной длины;
- строка, содержащая нули, – для строк фиксированной длины;
- Empty (значение, указывающее на отсутствие инициализации) - для статических переменных типа Variant;
- для массивов каждый элемент инициализируется в соответствии с указанными правилами.

Объявление простых переменных имеет следующий синтаксис: {Dim | Private | Public | Static }<имя переменной> [As <имя типа>] [, <имя переменной> [As <имя типа>]];

Вначале идет имя оператора, а потом список объявлений переменных, где роль разделителя играет запятая. Объявление связывает имя переменной с ее типом, заданным конструкцией As. Когда она опущена, переменная получает тип Variant. Одним оператором Dim можно объявить произвольное число переменных, но конструкция As должна быть указана для каждой из них, иначе переменным без As будет присвоен тип Variant.

Строковый тип данных String

Строкой называется не только последовательность символов, заключённых в кавычки, но и переменная строкового типа, объявленная с помощью ключевого слова String.

Строка как переменная может иметь переменную или постоянную длину.

Строка переменной длины характеризуется тем, что занимаемый ею объём оперативной памяти может меняться в процессе выполнения программы.

Строка постоянной длины занимает фиксированный объём оперативной памяти. При её объявлении после ключевого слова String ставится значок * (звёздочка) и указывается объём памяти (в байтах), выделяемый этой строке.

Пример:

```
Dim A As String
Dim B As String*15
A = "Информатика"
B = "Информатика"
```

В примере переменная A – строка переменной длины, переменная B – строка постоянной длины. После выполнения оператора A = "Информатика" строка A занимает 21 байт, так как размер строки переменной длины равен 10 байт плюс 1 байт на символ. После выполнения оператора B = "Информатика" строка B будет занимать 15 байт, так как объём оперативной памяти, занимаемый строкой постоянной длины не зависит от количества содержащихся в ней символов.

Значение, присваиваемое строке постоянной длины, может содержать количество символов как меньшее указанного в описании, так и большее. В первом случае в конце строки вместо

недостающих символов автоматически будут добавлены пробелы, а во втором случае лишние символы будут автоматически удалены.

Работа с переменными строкового типа

Операции над строками

Строки можно присваивать, объединять и сравнивать. Объединение строк записывается в естественном виде. Если сумма получается длиннее, чем описанная длина левой части оператора присваивания, то излишек отсекается.

Для объединения двух и более строк используется символ "&" (амперсанд) или "+" (сложение). Результатом объединения строк является строка.

Пример 1:

```
Sub строки1()
Dim A As String
Dim B As String
Dim C As String
A="Строковая"
B="переменная"
C=A & B          'Результат: C = "Строковая переменная"
C=A + "переменная"       'Результат: C = "Строковая переменная"
End Sub
```

Первый комментарий соответствует случаю, когда при пошаговом выполнении программы жёлтым выделен оператор C=A & B. Этот комментарий означает следующее: если указатель мышки подвести к C, то высветится C = "Строковая переменная" – результат выполнения оператора, находящегося в одной строке с комментарием.

Аналогичный смысл имеют все комментарии в программах данного методического указания, начинающегося со слова "Результат".

Следует учесть, что при остановах во время выполнения программы жёлтым цветом выделяется первый из невыполненных операторов.

Благодаря операции объединения строк, строку можно рассматривать как выражение, аналогичное арифметическим и логическим выражениям

Сравнение строк происходит посимвольно, начиная от первого символа в строке. Строки равны, если имеют одинаковую длину и посимвольно эквивалентны.

В языке VBA имеется набор стандартных (встроенных) функций для работы со строковыми данными.

Функции для работы со строками

Сравнение строковых выражений

Для сравнения строковых выражений используются следующие функции:

1) Option Compare{Binary | Text | Database}, где

Binary – сравнение идёт побайтно;

Text – сравнение идёт по расположению этих слов в словаре;

Database – работает только с Microsoft Access.

Операция сравнения не зависит от регистра символов.

2) StrComp (string1, string2 [,compare]), где

string1 – строка, которая сравнивается;

string2 – строка, которая сравнивается;

compare – способ сравнения (по умолчанию 0 – Binary, 1 – Text)

Если string1=string2, то функция возвращает 0; если string1<string2, то возвращает 1; если string1>string2, то возвращает -1.

Удаление пробелов

При работе со строками используется три функции удаления пробелов. Описание и назначение этих функций представлено в таблице 1.

Таблица 1

Описание и назначение функций удаления пробелов

Функция	Назначение	Тип аргумента	Тип результата
LTrim(N)	Удаляет все пробелы в начале строки (слева – left)	String	String
RTrim(N)	Удаляет все пробелы в конце строки (справа – right)	String	String
Trim(N)	Удаляет все пробелы в начале и в конце строки	String	String

Пример 2:

```
Sub Строки2()
Dim A As String
Dim B As String
A="    Строковая переменная    "
B=LTrim(A)          'Результат: "Строковая переменная    "
B=RTrim(A)          'Результат: "    Строковая переменная"
B=Trim(A)           'Результат: "Строковая переменная"
End Sub
```

Преобразование числа в строку

Для преобразования числового значения в строковое используется функция Str(Value), где

Value – аргумент функции числового типа.

Результат будет иметь тип String.

Например:

Если переменная n=1000, то в результате S=Str(n) получим S="1000".

Преобразование строки в число

Для преобразования строки в число используется функция Val(St), где

St – аргумент функции строкового типа.

Результат будет иметь числовой тип.

Если функция Val не может сделать преобразование, то она возвращает ноль.

Пример 3:

```
Sub Строки3()
Dim A As String
Dim B As Currency
A="45.77"
B=Val(A)                    'Результат: B=45.77
A=Str(B)                    'Результат: A="45.77"
B=Val("4.7=X")              'Результат: B=4.7
B=Val("X=4.7")              'Результат: B=0
End Sub
```

Возвращение строки, состоящей из пробелов.

Для возвращения в программу строки, состоящей из пробелов, используется функция Space(N), где

N – аргумент функции типа Byte, обозначающий количество пробелов, из которых будет состоять строка.

Пример 4:

```
Sub Строки4()
Dim A As String
Dim B As Currency
Dim C As String
A="X="
B=45.77
C=A & Str(B)                    'Результат: C="X=45.77"
B=Val("45.77=X")                'Результат: B=45.77
B=Val(C)                        'Результат: B=0
C="Строковая" & Space(3) & "переменная"
```

'Результат: C="Строковая переменная"
End Sub

Замена подстроки

Для замены какой-либо части строки или определённых символов используется функция Replace.

Её синтаксис: Replace(expression, find, replace [,start [,count [,compare]]]), где:

expression – исходная строка;

find – какую подстроку заменить;

replace – на какую подстроку заменит;

start – позиция начала поиска (по умолчанию равна 1);

count – количество проводимых замен (по умолчанию равно 1);

compare – способ сравнения (по умолчанию Binary(0) – сравнение идёт побайтно; Text(1) – сравнение идёт по расположению строк в словаре).

Пример 5:

```
Sub строки5()
Dim A As String
Dim B As String
A="Павел Иванов"
B=Replace(A, "Иванов", "Гусев")          'Результат: B="Павел Гусев
End Sub
```

Преобразование букв строки в заглавные или строчные

Для преобразования строки или строковой переменной таким образом, чтобы все буквы стали заглавными (то есть буквами верхнего регистра – upper case) или строчными (то есть буквами нижнего регистра – lower case) используются функции UCase(S) и LCase(S) соответственно.

Описание и назначение этих функций приведено в таблице 2.

Таблица 2

Описание и назначение функций UCase и LCase

Функция	Назначение	Тип аргумента	Тип результата
UCase(S)	Преобразует все буквы строки в заглавные	String	String
LCase(S)	Преобразует все буквы строки в строчные	String	String

Пример 6:

```
Sub Строки6()
Dim A As String
Dim B As String
A="Павел Иванов"
B=UCase(A)              'Результат: "ПАВЕЛ ИВАНОВ"
B=LCase(A)              'Результат: "павел иванов"
End Sub
```

Определение длины строки

Для определения количества символов в строке (не считая кавычек) применяется функция Len(S),где

S – строка символов.

Результат будет иметь тип Integer

Пример 7:

```
Sub Строки7()
Dim S As String
Dim N As Integer
A="Строковая переменная"
N=Len(S)                    'Результат: N=20
End Sub
```

Выделение подстроки из строки символов

Для выделения подстроки из строки используются функции Left(S,N), Right(S,N), Mid(S, Start, N), где:

S – строка;

N – количество выделяемых символов;

Start – позиция начала замены.

Описание и назначение этих функций приведено в таблице 3.

Таблица 3

Описание и назначение функций Left, Right, Mid

Функция	Назначение	Тип аргумента	Тип результата
Left(S,N)	Выделяет N символов слева	S – String N – Byte Start – Byte	String
Right(S,N)	Выделяет N символов справа		
Mid(S, Start, N)	Выделяет N символов начиная с позиции start		

Пример 8:

```
Sub Строки8()
Dim S As String
Dim S1 As String
S=" Моя строковая переменная"
S1=Left(S, 3)                    'Результат: S="Моя"
S1=Right(S, 10)                  'Результат: S="переменная"
S1=mid(S, 5,9)                   'Результат: S="строковая"
End Sub
```

Определение позиции вхождения подстроки в строку

Для определения позиции вхождения подстроки в строку используются функции InStr([start,] S1, S2[, compare]) InStrRev(S1, S2[, start[, compare]]), где:

start – позиция начала поиска;
S1 – строка, в которой будет происходить поиск;
S2 – подстрока, которую будем искать в строке S1;
compare – способ сравнения.

Описание и назначение этих функций приведено в таблице 4.

Таблица 4

Описание и назначение функций InStr и InStrRev

Функция	Назначение	Тип аргумента	Тип результата
InStr([start,] S1, S2[, compare]) InStrRev(S1, S2[, start[, compare]])	Определяет позицию первого вхождения подстроки S2 в строку S1 Определяет позицию последнего вхождения подстроки S2 в строку S1	start – Byte S1 – String S2 – Srting compare – 0 или 1 (по умолчанию 0)	Byte

Пример 9:

```
Sub Строки9()
Dim S As String
Dim S1 As String
Dim N As Byte
S=" Моя строковая переменная"
S1="о"
N= InStr(S, S1)                    'Результат: N=2
N= InStrRev(S, S1)                 'Результат: N=9
End Sub
```

Преобразование элементов массива в строку

Для преобразования элементов массива в строку можно использовать функцию, Join(array[, delimiter]), где:

array – массив, который нужно преобразовать;

delimiter – символ, который будет разделять элементы массива в конечной строке (по умолчанию пробел).

Преобразование строки в массив

Для преобразования строки в массив используется функция Split(S[, delimiter[, limit[, compare]]]), где

S – строка, которую нужно преобразовать в массив

delimiter – символ, который нужно принять за разделитель элементов массива (по умолчанию пробел);

limit – ограничение возвращаемых элементов;

compare – способ сравнения.

3. ПОРЯДОК ВЫПОЛНЕНИЯ

1. Получить задание у преподавателя.
2. Выполнить задание в соответствии с вариантом.
3. Ответить на контрольные вопросы.

4. ЗАДАНИЕ

Выполнить сортировку одномерного массива на языке программирования Visual Basic for Applications.

ВАРИАНТ № 1

Дана строка символов.

1) Определить количество слов, длина которых больше трех и меньше либо равно шести символам.

2) Определить длину самого длинного слова и напечатать его.

3) Определить длину строки с символа с номером *M* по символ с номером *N*.

4) Определить количество слов, начинающихся на сочетание "*ст*".

ВАРИАНТ № 2

Дана строка символов.

1) Напечатать строку символов между второй и третьей запятыми.

2) Определить количество слов, заканчивающихся на "*ь*".

3) Длину самого короткого слова и напечатать его.

4) Определить количество слов, длина которых меньше семи символов.

ВАРИАНТ № 3

Дана строка символов.

1) Напечатать часть строки от первого символа "*5*" до последнего символа "*!*".

2) Определить количество слов, содержащих букву "*й*".

3) Определить количество букв "*а*" во втором слове.

4) Напечатать слова, начинающиеся на "*кр*".

ВАРИАНТ № 4

Дана строка символов.

1) Напечатать часть строки от предпоследней до последней запятой.

2) Определить количество слов, заканчивающихся на сочетание "*ай*".

3) Определить напечатать среднее (по расположению в строке) слово.

4) Определить количество слов, длина которых более *10* символов.

ВАРИАНТ № 5

Дана строка символов.

1) Определить количество символов "=" после предпоследней запятой.

2) Определить и напечатать слова, начинающиеся на "*ст*" и заканчивающиеся на "*нь*".

3) Определить количество букв "*о*" во втором слове.

4) Напечатать второе слово в обратном порядке.

ВАРИАНТ № 6

Дана строка символов.

1) Определить и напечатать слово, содержащее максимальное количество букв "*о*".

2) Определить количество слов, начинающихся на букву "*к*" и длиной от трех до семи символов.

3) Определить количество символов между первым символом "*7*" и последним символом "*?*".

4) напечатать предпоследнее слово в обратном порядке.

ВАРИАНТ № 7

Дана строка символов.

1) Определить количество слов, длина которых больше длины первого слова.

2) Определить длину самого длинного слова и напечатать его.

3) Определить длину строки с символа с номером *M* по вторую встреченную запятую.

4) Определить количество слов, заканчивающихся на "*ок*".

ВАРИАНТ № 8

Дана строка символов.

1) Напечатать строку символов между второй и предпоследней точками.

2) Определить количество слов, в которых встречается буква "*я*".

3) Длину самого короткого слова и напечатать его.

4) Определить количество слов, длина которых меньше 8 символов, но больше длины самого короткого слова.

ВАРИАНТ № 9

Дана строка символов.

1) Напечатать часть строки от третьего символа "*5*" до последнего символа "*;*".

2) Определить количество слов, начинающихся и заканчивающихся на букву "*о*".

3) Определить количество букв "*к*" в предпоследнем слове.

4) Напечатать слова, начинающиеся на "*рос*".

ВАРИАНТ № 10

Дана строка символов.

1) Напечатать часть строки от предпоследней запятой до первой точки.

2) Определить количество слов, заканчивающихся на сочетание "*ин*".

3) Определить напечатать среднее (по расположению в строке) слово.

4) Определить количество слов, длина которых не менее *8* символов.

ВАРИАНТ № 11

Дана строка символов.

1) Определить количество символов "+" после второй точки.

2) Определить и напечатать слова, начинающиеся на "*ст*" и содержащих символ "*5*".

3) Определить количество букв "*о*" в строке.

4) Напечатать последнее слово в обратном порядке.

ВАРИАНТ № 12

Дана строка символов.

1) Определить и напечатать слово, содержащее минимальное количество букв "*о*".

2) Определить количество слов, начинающихся на букву "*т*" и длиной не более семи символов.

3) Определить количество символов между последним символом "*7*" и последним символом "*?*".

4) напечатать предпоследнее слово.

ВАРИАНТ № 13

Дана строка символов.

1) Определить количество слов, содержащих более трех запятых.

2) Определить длину слова, содержащего наибольшее число символов "*о*" и напечатать его.

3) Определить длину строки до третьего с конца строки слова.

4) Определить количество слов, начинающихся на сочетание "*ви*".

ВАРИАНТ № 14

Дана строка символов.

1) Напечатать строку символов между второй запятой и началом последнего слова.

2) Определить количество слов, содержащих "ъ".

3) Длину самого длинного слова и напечатать его.

4) Определить количество слов, длина которых меньше семи символов и больше трех символов.

ВАРИАНТ № 15

Дана строка символов.

1) Напечатать часть строки от первого символа "*к*" до последнего символа "*р*".

2) Определить количество слов, содержащих сочетание "*ай*".

3) Определить количество букв "*е*" в предпоследнем слове.

4) Напечатать слова, начинающиеся на "*11*".

ВАРИАНТ № 16

Дана строка символов.

1) Напечатать часть от начала самого короткого слова до последней запятой.

2) Определить количество слов, заканчивающихся на сочетание "*ть*" и состоящих из 4 символов.

3) Определить напечатать третье с конца строки слово.

4) Определить количество слов, длина которых более *4* символов.

ВАРИАНТ № 17

Дана строка символов.

1) Определить количество символов "=" после второго слова.

2) Определить и напечатать слова, начинающиеся на "*m*".

3) Определить количество букв "*о*" во втором и предпоследнем словах.

4) Напечатать последнее слово в обратном порядке.

ВАРИАНТ № 18

Дана строка символов.

1) Определить и напечатать слово, содержащее максимальное количество символов, но не содержащее букв "*о*".

2) Определить количество слов, длиной пять или восемь символов.

3) Определить количество символов между последним символом "*%*" и третьим символом "*!*".

4) напечатать самое длинное слово в обратном порядке.

ВАРИАНТ № 19

Дана строка символов.

1) Определить и напечатать слово, содержащее *M* или *N* символов.

2) Определить количество слов, содержащих символ "*?*", но не начинающихся на "*о*".

3) Определить количество символов между последним символом "э" и символом под номером *N*.

4) Определить напечатать среднее (по расположению в строке) слово в обратном порядке.

5. КОНТРОЛЬНЫЕ ВОПРОСЫ

Как описывается область видимости и время существования переменных?

2. Чем характеризуется строка переменной длины?

3. Чем характеризуется строка постоянной длины?

4. Какие операции можно выполнять над строками?

5. Какие действия выполняют функции Val(St) и Str(Value)?

6. Какие действия выполняют функции UCase(S) и LCase(S)?

7. Какие функции определяют позицию вхождения подстроки в строку?

Оглавление