

ФГОС

8



Л. Л. Босова  
А. Ю. Босова

ИНФОРМАТИКА



ИЗДАТЕЛЬСТВО

БИНОМ

**ФГОС**

**Л. Л. Босова, А. Ю. Босова**

# **ИНФОРМАТИКА**

**Учебник  
для 8 класса**

**2-е издание, исправленное**

Рекомендовано  
Министерством образования и науки  
Российской Федерации  
к использованию в образовательном процессе  
в имеющих государственную аккредитацию  
и реализующих образовательные программы  
общего образования образовательных учреждениях

**爱  
谢谢**



Москва  
БИНОМ. Лаборатория знаний  
2014

# Введение

## *Уважаемые восьмиклассники!*

Мы живём во время стремительных перемен, когда для человека важна способность к постоянному развитию, готовность к освоению новых, в том числе информационных, технологий. Необходимость подготовки к быстро наступающим переменам в окружающем мире требует от человека развитого мышления, умения организации собственной учебной деятельности, ориентации на деятельностную жизненную позицию. Формирование таких качеств личности невозможно без фундаментального базового образования.

В курсе информатики 8 класса большое внимание уделяется фундаментальным (теоретическим) вопросам информатики. Вы будете изучать математические основы информатики, освоите базовые алгоритмические конструкции, познакомитесь с языком программирования Паскаль.

Мы надеемся, что знания и умения, полученные на уроках информатики, вы сможете применять в дальнейшем при решении разнообразных жизненных задач, предполагающих такие этапы, как:

- **целеполагание** — постановка задачи на основе соотнесения того, что уже известно, и того, что требуется установить;
- **планирование** — определение последовательности промежуточных целей с учётом конечного результата, разбиение задачи на подзадачи, разработка последовательности и структуры действий, необходимых для достижения цели с помощью фиксированного набора средств;
- **прогнозирование** — предвосхищение результата;
- **контроль** — интерпретация полученного результата, его соотнесение с имеющимися данными с целью установления соответствия или несоответствия (обнаружения ошибки);
- **коррекция** — внесение необходимых дополнений и исправлений в план действий в случае обнаружения ошибки;
- **оценка** — осознание человеком того, насколько качественно им решена задача.











Как и ранее, в этом учебнике кроме основной информации содержатся многочисленные ссылки на образовательные ресурсы сети Интернет, в том числе на такие порталы, как:

- 1) Единая коллекция цифровых образовательных ресурсов (<http://sc.edu.ru/>);
- 2) Федеральный центр информационных образовательных ресурсов (<http://fcior.edu.ru/>);
- 3) сайт методической службы издательства (<http://metodist.Lbz.ru/>).

На страницах учебника подробно рассмотрены решения типовых задач по каждой изучаемой теме. В конце каждой главы учебника приведены тестовые задания, которые помогут вам оценить, хорошо ли вы освоили теоретический материал и можете ли применять свои знания для решения возникающих проблем.

Изучая теоретический материал, работая с дополнительными материалами, отвечая на вопросы, решая задачи и выполняя практические задания на компьютере, вы сможете полностью подготовиться к сдаче выпускного экзамена по курсу информатики и в форме государственной итоговой аттестации (ГИА), требования к которой размещены на сайте <http://fipi.ru>.

В работе с учебником вам помогут навигационные значки:

-  — важное утверждение или определение;
-  — интересная информация;
-  — пример решения задачи;
-  — информация, полезная для решения практических задач;
-  — ссылка на ресурс в Интернете;
-  — дополнительный материал к параграфу, содержащийся в электронном приложении к учебнику (<http://metodist.Lbz.ru/>);
-  — вопросы в тексте параграфа, вопросы и задания для самоконтроля;
-  — задания для подготовки к итоговой аттестации;
-  — домашний проект или исследование;
-  — задания для практических работ на компьютере.

Желаем успехов в изучении информатики!

---

# Глава 1

## МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ

### § 1.1

#### Системы счисления

**Ключевые слова:**

- система счисления
- цифра
- алфавит
- позиционная система счисления
- основание
- развёрнутая форма записи числа
- свёрнутая форма записи числа
- двоичная система счисления
- восьмеричная система счисления
- шестнадцатеричная система счисления

#### 1.1.1. Общие сведения о системах счисления

---

**Система счисления** — это знаковая система, в которой приняты определённые правила записи чисел. Знаки, с помощью которых записываются числа (рис. 1.1), называются **цифрами**, а их совокупность — **алфавитом** системы счисления.

---

В любой системе счисления цифры служат для обозначения чисел, называемых *узловыми*; остальные числа (*алгоритмические*) получаются в результате каких-либо операций из узловых чисел.

**Пример 1.** У вавилонян узловыми являлись числа 1, 10, 60; в римской системе счисления узловые числа — это 1, 5, 10, 50, 100, 500 и 1000, обозначаемые соответственно I, V, X, L, C, D, M.



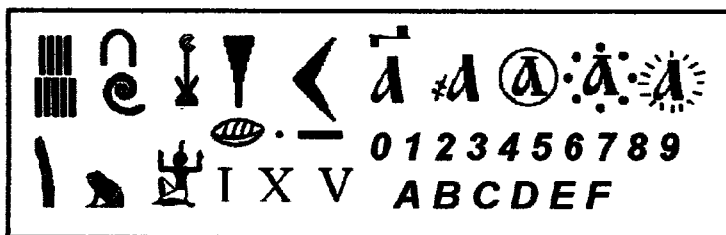


Рис. 1.1. Знаки, используемые для записи чисел в различных системах счисления

Системы счисления различаются выбором узловых чисел и способами образования алгоритмических чисел. Можно выделить следующие виды систем счисления:

- 1) унарная система;
- 2) непозиционные системы;
- 3) позиционные системы.

Простейшая и самая древняя система — так называемая **унарная система счисления**. В ней для записи любых чисел используется всего один символ — палочка, узелок, зарубка, камушек. Длина записи числа при таком кодировании прямо связана с его величиной, что роднит этот способ с геометрическим представлением чисел в виде отрезков. Именно унарная система лежит в фундаменте арифметики, и именно она до сих пор вводит первоклассников в мир счёта. Унарную систему ещё называют системой бирок.



Система счисления называется **непозиционной**, если количественный эквивалент (количественное значение) цифры в числе не зависит от её положения в записи числа.

В большинстве непозиционных систем счисления числа образуются путём сложения узловых чисел.

**Пример 2.** В древнеегипетской системе счисления числа 1, 2, 3, 4, 10, 13, 40 обозначались соответственно следующим образом:



Те же числа в римской системе счисления обозначаются так: I, II, III, IV, X, XIII, XL. Здесь алгоритмические числа получаются путём сложения и вычитания узловых чисел с учётом следующего правила: каждый меньший знак, поставленный справа от большего, прибавляется к его значению, а каждый меньший знак, поставленный слева от большего, вычитается из него.

Система счисления называется **позиционной**, если количественный эквивалент цифры зависит от её положения (позиции) в записи числа.

**Основание** позиционной системы счисления равно количеству цифр, составляющих её алфавит.

Десятичная система записи чисел, которой мы привыкли пользоваться в повседневной жизни, с которой мы знакомы с детства, в которой производим все наши вычисления, — пример позиционной системы счисления. Алфавит десятичной системы составляют цифры 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Алгоритмические числа образуются в ней следующим образом: значения цифр умножаются на «веса» соответствующих разрядов, и все полученные значения складываются. Это отчётливо прослеживается в числительных русского языка, например: «три-ста пять-десят семь».

Основанием позиционной системы счисления может служить любое натуральное число  $q > 1$ . Алфавитом произвольной позиционной системы счисления с основанием  $q$  служат числа 0, 1, ...,  $q-1$ , каждое из которых может быть записано с помощью одного уникального символа; младшей цифрой всегда является 0.

Основные достоинства любой позиционной системы счисления — простота выполнения арифметических операций и ограниченное количество символов, необходимых для записи любых чисел.

В позиционной системе счисления с основанием  $q$  любое число может быть представлено в виде:

$$A_q = \pm (a_{n-1} \cdot q^{n-1} + a_{n-2} \cdot q^{n-2} + \dots + a_0 \cdot q^0 + a_{-1} \cdot q^{-1} + \dots + a_{-m} \cdot q^{-m}). \quad (1)$$

Здесь:

$A$  — число;

$q$  — основание системы счисления;

$a_i$  — цифры, принадлежащие алфавиту данной системы счисления;

$n$  — количество целых разрядов числа;

$m$  — количество дробных разрядов числа;

$q^i$  — «вес»  $i$ -го разряда.

Запись числа по формуле (1) называется **развёрнутой формой записи**. **Свёрнутой формой** записи числа называется его представление в виде<sup>1</sup>  $\pm a_{n-1}a_{n-2}\dots a_1a_0,a_{-1}\dots a_{-m}$ .

<sup>1</sup> Далее будут рассматриваться только положительные целые числа.



**Пример 3.** Рассмотрим десятичное число 14351,1. Его свёрнутая форма записи настолько привычна, что мы не замечаем, как в уме переходим к развёрнутой записи, умножая цифры числа на «веса» разрядов и складывая полученные произведения:

$$1 \cdot 10^4 + 4 \cdot 10^3 + 3 \cdot 10^2 + 5 \cdot 10^1 + 1 \cdot 10^0 + 1 \cdot 10^{-1}.$$

## 1.1.2. Двоичная система счисления

**Двоичной системой счисления** называется позиционная система счисления с основанием 2. Для записи чисел в двоичной системе счисления используются только две цифры: 0 и 1.

На основании формулы (1) для целых двоичных чисел можно записать:

$$a_{n-1}a_{n-2}\dots a_1a_0 = a_{n-1} \cdot 2^{n-1} + a_{n-2} \cdot 2^{n-2} + \dots + a_0 \cdot 2^0. \quad (1')$$

Например:

$$10011_2 = 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 2^4 + 2^1 + 2^0 = 19_{10}.$$



Такая форма записи «подсказывает» правило перевода натуральных двоичных чисел в десятичную систему счисления: *необходимо вычислить сумму степеней двойки, соответствующих единицам в свёрнутой форме записи двоичного числа.*



Получим правило перевода целых десятичных чисел в двоичную систему счисления из формулы (1').

Разделим  $a_{n-1} \cdot 2^{n-1} + a_{n-2} \cdot 2^{n-2} + \dots + a_0 \cdot 2^0$  на 2. Частное будет равно  $a_{n-1} \cdot 2^{n-2} + \dots + a_1$ , а остаток будет равен  $a_0$ .

Полученное частное опять разделим на 2, остаток от деления будет равен  $a_1$ .

Если продолжить этот процесс деления, то на  $n$ -м шаге получим набор цифр:

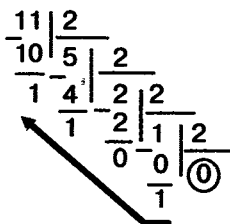
$$a_0, a_1, a_2, \dots, a_{n-1},$$

которые входят в двоичное представление исходного числа и совпадают с остатками при его последовательном делении на 2.

Таким образом, для перевода целого десятичного числа в двоичную систему счисления нужно последовательно выполнять деление данного числа и получаемых целых частных на 2 до тех пор, пока не получим частное, равное нулю. Исходное число в двоичной системе счисления составляется последовательной записью полученных остатков, начиная с последнего.



**Пример 4.** Переведём десятичное число 11 в двоичную систему счисления. Рассмотренную выше последовательность действий (алгоритм перевода) можно изобразить так:



Выписывая остатки от деления в направлении, указанном стрелкой, получим:  $11_{10} = 1011_2$ .

**Пример 5.** Если десятичное число достаточно большое, то более удобен следующий способ записи рассмотренного выше алгоритма:

363	181	90	45	22	11	5	2	1
1	1	0	1	0	1	1	0	1

←

$$363_{10} = 101101011_2$$

### 1.1.3. Восьмеричная система счисления

Восьмеричной системой счисления называется позиционная система счисления с основанием 8. Для записи чисел в восьмеричной системе счисления используются цифры: 0, 1, 2, 3, 4, 5, 6, 7.

На основании формулы (1) для целого восьмеричного числа можно записать:

$$a_{n-1}a_{n-2}\dots a_1a_0 = a_{n-1} \cdot 8^{n-1} + a_{n-2} \cdot 8^{n-2} + \dots + a_0 \cdot 8^0. \quad (1'')$$

Например:  $1063_8 = 1 \cdot 8^3 + 0 \cdot 8^2 + 6 \cdot 8^1 + 3 \cdot 8^0 = 563_{10}$ .

Таким образом, для перевода целого восьмеричного числа в десятичную систему счисления следует перейти к его развёрнутой записи и вычислить значение получившегося выражения.

Для перевода целого десятичного числа в восьмеричную систему счисления следует последовательно выполнять деление данного числа и получаемых целых частных на 8 до тех пор, пока не получим частное, равное нулю. Исходное число в новой системе счисления составляется последовательной записью полученных остатков, начиная с последнего.

**Пример 6.** Переведём десятичное число 103 в восьмеричную систему счисления.

$$\begin{array}{r|l}
 103 & 8 \\
 \hline
 8 & 12 \\
 \hline
 23 & 8 \\
 \hline
 16 & 4 \\
 \hline
 7 & 0 \\
 \hline
 & 1
 \end{array}
 \begin{array}{l}
 \\
 \\
 \\
 \\
 \\
 \textcircled{0}
 \end{array}$$

$$103_{10} = 147_8$$

### 1.1.4. Шестнадцатеричная система счисления

Основание:  $q = 16$ .

Алфавит: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Здесь только десять цифр из шестнадцати имеют общепринятое обозначение 0, ..., 9. Для записи цифр с десятичными количественными эквивалентами 10, 11, 12, 13, 14, 15 обычно используются первые пять букв латинского алфавита.

Таким образом, запись  $3AF_{16}$  означает:

$$3AF_{16} = 3 \cdot 16^2 + 10 \cdot 16^1 + 15 \cdot 16^0 = 768 + 160 + 15 = 943_{10}.$$

**Пример 7.** Переведём десятичное число 154 в шестнадцатеричную систему счисления.

$$\begin{array}{r|l}
 154 & 16 \\
 \hline
 144 & 9 \\
 \hline
 10 & 0 \\
 \hline
 (A) & 9 \\
 \hline
 & \textcircled{0}
 \end{array}$$

$$154_{10} = 9A_{16}$$

### 1.1.5. Правило перевода целых десятичных чисел в систему счисления с основанием $q$

Для перевода целого десятичного числа в систему счисления с основанием  $q$  следует:

- 1) последовательно выполнять деление данного числа и получаемых целых частных на основание новой системы счисления до тех пор, пока не получим частное, равное нулю;
- 2) полученные остатки, являющиеся цифрами числа в новой системе счисления, привести в соответствие с алфавитом новой системы счисления;

3) составить число в новой системе счисления, записывая его, начиная с последнего полученного остатка.

Представим таблицу соответствия десятичных, двоичных, восьмеричных и шестнадцатеричных чисел от 0 до  $20_{10}$ .

Десятичная система	Двоичная система	Восьмеричная система	Шестнадцатеричная система
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14



В Единой коллекции цифровых образовательных ресурсов (<http://sc.edu.ru/>) размещена интерактивная анимация «Преобразование десятичного числа в другую систему счисления» (135050). С её помощью можно понаблюдать за переводом произвольного целого числа от 0 до 512 в позиционную систему счисления, основание которой не превышает 16.

В размещённой там же виртуальной лаборатории «Цифровые весы» (135009) вы сможете освоить ещё один способ перевода целых десятичных чисел в другие системы счисления — метод разностей.

## 1.1.6. Двоичная арифметика

Арифметика двоичной системы счисления основывается на использовании следующих таблиц сложения и умножения:

+	0	1
0	0	1
1	1	10

×	0	1
0	0	0
1	0	1



**Пример 8.** Таблица двоичного сложения предельно проста. Так как  $1 + 1 = 10$ , то 0 остаётся в младшем разряде, а 1 переносится в старший разряд.

+	1	0	0	1
	1	0	1	0
	1	0	1	1

+	1	1	1	1
				1
	1	0	0	0



**Пример 9.** Операция умножения двоичных чисел выполняется по обычной схеме, применяемой в десятичной системе счисления, с последовательным умножением множимого на очередную цифру множителя.

			1	0	1	1
				1	0	1
				1	0	1
				1	0	1
1	0		1	1		
1	1	0	1	1	1	1

Таким образом, в двоичной системе счисления умножение сводится к сдвигам множимого и сложениям.

### 1.1.7. «Компьютерные» системы счисления

В компьютерной технике используется двоичная система счисления, обеспечивающая ряд преимуществ по сравнению с другими системами счисления:

- двоичные числа представляются в компьютере с помощью достаточно простых технических элементов с двумя устойчивыми состояниями;
- представление информации посредством только двух состояний надёжно и помехоустойчиво;
- двоичная арифметика наиболее проста;
- существует математический аппарат, обеспечивающий логические преобразования двоичных данных.

Обмен информацией между компьютерными устройствами осуществляется путём передачи двоичных кодов. Пользоваться такими кодами из-за их большой длины и зрительной однородности человеку неудобно. Поэтому специалисты (программисты, инженеры) на некоторых этапах разработки, создания, настройки вычислительных систем заменяют двоичные коды на эквивалентные им величины в восьмеричной или шестнадцатеричной системах счисления. В результате длина исходного слова сокращается в три, четыре раза соответственно. Это делает информацию более удобной для рассмотрения и анализа.

С помощью ресурса «Интерактивный задачник, раздел «Системы счисления»» (128659), размещённого в Единой коллекции цифровых образовательных ресурсов, можно проверить, насколько прочно вы усвоили изученный в этом параграфе материал.



## САМОЕ ГЛАВНОЕ

Система счисления — это знаковая система, в которой приняты определённые правила записи чисел. Знаки, с помощью которых записываются числа, называются цифрами, а их совокупность — алфавитом системы счисления.

Система счисления называется позиционной, если количественный эквивалент цифры зависит от её положения (позиции) в записи числа. Основание позиционной системы счисления равно количеству цифр, составляющих её алфавит.

Основанием позиционной системы счисления может служить любое натуральное число  $q > 1$ .



В позиционной системе счисления с основанием  $q$  любое число может быть представлено в виде:

$$A_q = \pm (a_{n-1} \cdot q^{n-1} + a_{n-2} \cdot q^{n-2} + \dots + a_0 \cdot q^0 + a_{-1} \cdot q^{-1} + \dots + a_{-m} \cdot q^{-m}).$$

Здесь:

$A$  — число;

$q$  — основание системы счисления;

$a_i$  — цифры, принадлежащие алфавиту данной системы счисления;

$n$  — количество целых разрядов числа;

$m$  — количество дробных разрядов числа;

$q^i$  — «вес»  $i$ -го разряда.



## Вопросы и задания



1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Что вы можете сказать о формах представления информации в презентации и в учебнике? Какими слайдами вы могли бы дополнить презентацию?



2. Найдите дополнительную информацию об унарной, позиционных и непозиционных системах счисления. Чем они различаются? Приведите примеры.

3. Цифры каких систем счисления приведены на рис. 1.1?

4. Объясните, почему позиционные системы счисления с основаниями 5, 10, 12 и 20 называют системами счисления анатомического происхождения.

5. Как от свёрнутой формы записи десятичного числа перейти к его развёрнутой форме?

6. Запишите в развёрнутой форме числа:

а)  $143,511_{10}$ ;

б)  $143511_8$ ;

в)  $143511_{16}$ ;

г)  $1435,11_8$ .

7. Вычислите десятичные эквиваленты следующих чисел:

а)  $172_8$ ;

б)  $2EA_{16}$ ;






в)  $101010_2$ ;

г)  $10,1_2$ ;

д)  $243_6$ .

8. Укажите, какое из чисел  $110011_2$ ,  $111_4$ ,  $35_8$  и  $1B_{16}$  является:
- наибольшим;
  - наименьшим.
9. Какое минимальное основание имеет система счисления, если в ней записаны числа 123, 222, 111, 241? Определите десятичный эквивалент данных чисел в найденной системе счисления.
10. Верны ли следующие равенства?
- $33_4 = 21_7$ ;
  - $33_8 = 21_4$ .
11. Найдите основание  $x$  системы счисления, если:
- $14_x = 9_{10}$ ;
  - $2002_x = 130_{10}$ .
12. Переведите целые числа из десятичной системы счисления в двоичную:
- 89;
  - 600;
  - 2010.
13. Переведите целые числа из десятичной системы счисления в восьмеричную:
- 513;
  - 600;
  - 2010.
14. Переведите целые числа из десятичной системы счисления в шестнадцатеричную:
- 513;
  - 600;
  - 2010.
15. Заполните таблицу, в каждой строке которой одно и то же число должно быть записано в системах счисления с основаниями 2, 8, 10 и 16.

Основание 2	Основание 8	Основание 10	Основание 16
101010			
	127		
		321	
			2A

-  16. Выполните операцию сложения над двоичными числами:
- а)  $101010 + 1101$ ;
  - б)  $1010 + 1010$ ;
  - в)  $10101 + 111$ .
-  17. Выполните операцию умножения над двоичными числами:
- а)  $1010 \cdot 11$ ;
  - б)  $111 \cdot 101$ ;
  - в)  $1010 \cdot 111$ .
18. Расставьте знаки арифметических операций так, чтобы были верны следующие равенства в двоичной системе:
- а)  $1100 ? 11 ? 100 = 100000$ ;
  - б)  $1100 ? 10 ? 10 = 100$ ;
  - в)  $1100 ? 11 ? 100 = 0$ .
-  19. Вычислите выражения:
- а)  $(1111101_2 + AF_{16}) : 36_8$ ;
  - б)  $125_8 + 101_2 \cdot 2A_{16} - 141_8$ .
- Ответ дайте в десятичной системе счисления.
20. Какими преимуществами и недостатками обладает двоичная система счисления по сравнению с десятичной?
21. Разработайте таблицы сложения и умножения для восьмеричной системы счисления.
22. Постройте граф, отражающий разновидности систем счисления.
-   23. Подготовьте небольшое сообщение об одной из систем счисления (когда и где применялась, какие символы использовались и т. д.). Можете воспользоваться материалами электронного приложения к учебнику.



## Представление чисел в компьютере

### Ключевые слова:

- разряд
- беззнаковое представление целых чисел
- представление целых чисел со знаком
- представление вещественных чисел

### 1.2.1. Представление целых чисел

Оперативная память компьютера состоит из ячеек, каждая из которых представляет собой физическую систему, состоящую из некоторого числа однородных элементов. Эти элементы обладают двумя устойчивыми состояниями, одно из которых соответствует нулю, а другое — единице. Каждый такой элемент служит для хранения одного из битов — разряда двоичного числа. Именно поэтому каждый элемент ячейки называют битом или разрядом (рис. 1.2).

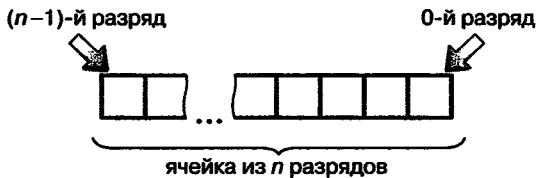


Рис. 1.2. Ячейка памяти

Для компьютерного представления целых чисел используется несколько различных способов, отличающихся друг от друга количеством разрядов (под целые числа обычно отводится 8, 16, 32 или 64 разряда) и наличием или отсутствием знакового разряда. Беззна-

ковое представление можно использовать только для неотрицательных целых чисел, отрицательные числа представляются только в знаковом виде.

Беззнаковое представление используется для таких объектов, как адреса ячеек, всевозможные счётчики (например, число символов в тексте), а также числа, обозначающие дату и время, размеры графических изображений в пикселях и т. д.

Максимальное значение целого неотрицательного числа достигается в случае, когда во всех разрядах ячейки хранятся единицы. Для  $n$ -разрядного представления оно будет равно  $2^n - 1$ . Минимальное число соответствует  $n$  нулям, хранящимся в  $n$  разрядах памяти, и равно нулю.

Ниже приведены максимальные значения для беззнаковых целых  $n$ -разрядных чисел:

Количество битов	Минимальное значение	Максимальное значение
8	0	255 ( $2^8 - 1$ )
16	0	65 535 ( $2^{16} - 1$ )
32	0	4 294 967 295 ( $2^{32} - 1$ )
64	0	18 446 744 073 709 551 615 ( $2^{64} - 1$ )

Для получения компьютерного представления беззнакового целого числа достаточно перевести число в двоичную систему счисления и дополнить полученный результат слева нулями до стандартной разрядности.



**Пример 1.** Число  $53_{10} = 110101_2$  в восьмиразрядном представлении имеет вид:

0	0	1	1	0	1	0	1
---	---	---	---	---	---	---	---

Это же число 53 в шестнадцати разрядах будет записано следующим образом:

0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

При представлении со знаком самый старший (левый) разряд отводится под знак числа, остальные разряды — под само число. Если число положительное, то в знаковый разряд помещается 0, если число отрицательное — 1. Такое представление чисел называется прямым кодом. В компьютере прямые коды используются для хранения положительных чисел в запоминающих устройствах, для выполнения операций с положительными числами.

На сайте Федерального центра информационно-образовательных ресурсов (<http://fcior.edu.ru/>) размещён информационный модуль «Число и его компьютерный код». С помощью этого ресурса вы можете получить дополнительную информацию по изучаемой теме.

Для выполнения операций с отрицательными числами используется **дополнительный код**, позволяющий заменить операцию вычитания сложением. Узнать алгоритм образования дополнительного кода вы можете с помощью информационного модуля «Дополнительный код», размещённого на сайте Федерального центра информационно-образовательных ресурсов (<http://fcior.edu.ru/>).

## 1.2.2. Представление вещественных чисел

Любое вещественное число  $A$  может быть записано в экспоненциальной форме:

$$A = \pm m \cdot q^p,$$

где:

$m$  — мантисса числа;

$q$  — основание системы счисления;

$p$  — порядок числа.

Например, число 472 000 000 может быть представлено так:  $4,72 \cdot 10^8$ ,  $47,2 \cdot 10^7$ ,  $472,0 \cdot 10^6$  и т. д.

С экспоненциальной формой записи чисел вы могли встречаться при выполнении вычислений с помощью калькулятора, когда в качестве ответа получали записи следующего вида:  $4.72E+8$ .

Здесь знак «Е» обозначает основание десятичной системы счисления и читается как «умножить на десять в степени».

Из приведённого выше примера видно, что положение запятой в записи числа может изменяться.

Для единообразия мантиссу обычно записывают как правильную дробь, имеющую после запятой цифру, отличную от нуля. В этом случае число 472 000 000 будет представлено как  $0,472 \cdot 10^9$ .



Вещественное число может занимать в памяти компьютера 32 или 64 разряда. При этом выделяются разряды для хранения знака мантиссы, знака порядка, порядка и мантиссы.

Пример:

0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
Знак и порядок										Знак и мантисса																											

Диапазон представления вещественных чисел определяется количеством разрядов, отведённых для хранения порядка числа, а точность определяется количеством разрядов, отведённых для хранения мантиссы.

Максимальное значение порядка числа для приведённого выше примера составляет  $1111111_2 = 127_{10}$ , и, следовательно, максимальное значение числа:

$$0,1111111111111111111111111111 \cdot 10^{1111111}$$



Попробуйте самостоятельно выяснить, каков десятичный эквивалент этой величины.

Широкий диапазон представления вещественных чисел важен для решения научных и инженерных задач. Вместе с тем следует понимать, что алгоритмы обработки таких чисел более трудоёмки по сравнению с алгоритмами обработки целых чисел.

## САМОЕ ГЛАВНОЕ

Для компьютерного представления целых чисел используются несколько различных способов, отличающихся друг от друга количеством разрядов (8, 16, 32 или 64) и наличием или отсутствием знакового разряда.

Для представления беззнакового целого числа его следует перевести в двоичную систему счисления и дополнить полученный результат слева нулями до стандартной разрядности.

При представлении со знаком самый старший разряд отводится под знак числа, остальные разряды — под само число. Если число положительное, то в знаковый разряд помещается 0, если число отрицательное, то 1. Положительные числа хранятся в компьютере в прямом коде, отрицательные — в дополнительном.

При хранении в компьютере вещественных чисел выделяются разряды на хранение знака порядка числа, самого порядка, знака мантиссы и мантиссы. При этом любое число записывается так:

$$A = \pm m \cdot q^p,$$

где:  $m$  — мантисса числа;

$q$  — основание системы счисления;

$p$  — порядок числа.

### Вопросы и задания

1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Используйте эти материалы при подготовке ответов на вопросы и выполнении заданий.
2. Как в памяти компьютера представляются целые положительные и отрицательные числа?
3. Любое целое число можно рассматривать как вещественное, но с нулевой дробной частью. Обоснуйте целесообразность наличия особых способов компьютерного представления целых чисел.
4. Представьте число  $63_{10}$  в беззнаковом 8-разрядном формате.
5. Найдите десятичные эквиваленты чисел по их прямым кодам, записанным в 8-разрядном формате со знаком:
  - а) 01001100;
  - б) 00010101.
6. Какие из чисел  $443_8$ ,  $101010_2$ ,  $256_{10}$  можно сохранить в 8-разрядном формате?
7. Запишите следующие числа в естественной форме:
  - а)  $0,3800456 \cdot 10^2$ ;
  - б)  $0,245 \cdot 10^{-3}$ ;
  - а)  $1,256900E+5$ ;
  - а)  $9,569120E-3$ .
8. Запишите число  $2010,0102_{10}$  пятью различными способами в экспоненциальной форме.
9. Запишите следующие числа в экспоненциальной форме с нормализованной мантиссой — правильной дробью, имеющей после запятой цифру, отличную от нуля:
  - а)  $217,934_{10}$ ;
  - б)  $75321_{10}$ ;
  - в)  $0,00101_{10}$ .
10. Изобразите схему, связывающую основные понятия, рассмотренные в данном параграфе.



## § 1.3

### Элементы алгебры логики

**Ключевые слова:**

- алгебра логики
- высказывание
- логическая операция
- конъюнкция
- дизъюнкция
- отрицание
- логическое выражение
- таблица истинности
- законы логики

#### 1.3.1. Высказывание

Алгебра в широком смысле этого слова — наука об общих операциях, аналогичных сложению и умножению, которые могут выполняться над разнообразными математическими объектами. Многие математические объекты (целые и рациональные числа, многочлены, векторы, множества) вы изучаете в школьном курсе алгебры, где знакомитесь с такими разделами математики, как алгебра чисел, алгебра многочленов, алгебра множеств и т. д.

Для информатики важен раздел математики, называемый алгеброй логики; объектами алгебры логики являются высказывания.



---

**Высказывание** — это предложение на любом языке, содержание которого можно однозначно определить как истинное или ложное.

---

Например, относительно предложений «*Великий русский учёный М. В. Ломоносов родился в 1711 году*» и «*Two plus six is eight*» мож-

но однозначно сказать, что они истинны. Предложение «*Зимой воробьи впадают в спячку*» ложно. Следовательно, эти предложения являются высказываниями.

В русском языке высказывания выражаются повествовательными предложениями. Но не всякое повествовательное предложение является высказыванием.



Например, предложение «*Это предложение является ложным*» не является высказыванием, так как относительно него нельзя сказать, истинно оно или ложно, без того чтобы не получить противоречие. Действительно, если принять, что предложение истинно, то это противоречит сказанному. Если же принять, что предложение ложно, то отсюда следует, что оно истинно.

Относительно предложения «*Компьютерная графика — самая интересная тема в курсе школьной информатики*» также нельзя однозначно сказать, истинно оно или ложно. Подумайте сами почему.



Побудительные и вопросительные предложения высказываниями не являются.



Например, не являются высказываниями такие предложения, как: «*Запишите домашнее задание*», «*Как пройти в библиотеку?*», «*Кто к нам пришёл?*».

Высказывания могут строиться с использованием знаков различных формальных языков — математики, физики, химии и т. п.



Примерами высказываний могут служить:

- 1) «*Na — металл*» (истинное высказывание);
- 2) «*Второй закон Ньютона выражается формулой  $F = m \cdot a$* » (истинное высказывание);
- 3) «*Периметр прямоугольника с длинами сторон  $a$  и  $b$  равен  $a \cdot b$* » (ложное высказывание).

Не являются высказываниями числовые выражения, но из двух числовых выражений можно составить высказывание, соединив их знаками равенства или неравенства. Например:

- 1) « $3 + 5 = 2 \cdot 4$ » (истинное высказывание);
- 2) « $\text{II} + \text{VI} > \text{VIII}$ » (ложное высказывание).

Не являются высказываниями и равенства или неравенства, содержащие переменные. Например, предложение « $X < 12$ » становится высказыванием только при замене переменной каким-либо конкретным значением: « $5 < 12$ » — истинное высказывание; « $12 < 12$ » — ложное высказывание.

Обоснование истинности или ложности высказываний решается теми науками, к сфере которых они относятся. Алгебра логики отвлекается от смысловой содержательности высказываний. Её интересует только то, истинно или ложно данное высказывание. В алгебре логики высказывания обозначают буквами и называют логическими переменными. При этом если высказывание истинно, то значение соответствующей ему логической переменной обозначают единицей ( $A = 1$ ), а если ложно — нулём ( $B = 0$ ). 0 и 1, обозначающие значения логических переменных, называются логическими значениями.



**Алгебра логики** определяет правила записи, упрощения и преобразования высказываний и вычисления их значений.

Опирируя логическими переменными, которые могут быть равны только 0 или 1, алгебра логики позволяет свести обработку информации к операциям с двоичными данными. Именно аппарат алгебры логики положен в основу компьютерных устройств хранения и обработки данных. С применением элементов алгебры логики вы будете встречаться и во многих других разделах информатики.

## 1.3.2. Логические операции

Высказывания бывают простые и сложные. Высказывание называется простым, если никакая его часть сама не является высказыванием. Сложные (составные) высказывания строятся из простых с помощью логических операций.

Рассмотрим основные логические операции, определённые над высказываниями. Все они соответствуют связкам, употребляемым в естественном языке.

Название логической операции	Логическая связка
Инверсия	«не»; «неверно, что»
Конъюнкция	«и»; «а»; «но»; «хотя»
Дизъюнкция	«или»



### Конъюнкция

Рассмотрим два высказывания:  $A = \text{«Основоположником алгебры логики является Джордж Буль»}$ ,  $B = \text{«Исследования Клода Шеннона позволили применить алгебру логики в вычислительной технике»}$ . Очевидно, новое высказывание *«Основоположником алгебры логики является Джордж Буль, и исследования Клода Шеннона позволили применить алгебру логики в вычислительной технике»* истинно только в том случае, когда одновременно истинны оба исходных высказывания.

Самостоятельно установите истинность или ложность трёх рассмотренных выше высказываний.




---

**Конъюнкция** — логическая операция, ставящая в соответствие двум высказываниям новое высказывание, являющееся истинным тогда и только тогда, когда оба исходных высказывания истинны.

---



Для записи конъюнкции используются следующие знаки: И,  $\wedge$ ,  $\cdot$ ,  $\&$ . Например:  $A$  И  $B$ ,  $A \wedge B$ ,  $A \cdot B$ ,  $A\&B$ .

Конъюнкцию можно описать в виде таблицы, которую называют **таблицей истинности**:

$A$	$B$	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

В таблице истинности перечисляются все возможные значения исходных высказываний (столбцы  $A$  и  $B$ ), причём соответствующие им двоичные числа, как правило, располагают в порядке возрастания: 00, 01, 10, 11. В последнем столбце записан результат выполнения логической операции для соответствующих операндов.

Конъюнкцию также называют логическим умножением. Подумайте почему.



## Дизъюнкция

Рассмотрим два высказывания:  $A = \langle \text{Идея использования в логике математической символики принадлежит Готфриду Вильгельму Лейбницу} \rangle$ ,  $B = \langle \text{Лейбниц является основоположником бинарной арифметики} \rangle$ . Очевидно, новое высказывание  $\langle \text{Идея использования в логике математической символики принадлежит Готфриду Вильгельму Лейбницу или Лейбниц является основоположником бинарной арифметики} \rangle$  ложно только в том случае, когда одновременно ложны оба исходных высказывания.



Самостоятельно установите истинность или ложность трёх рассмотренных выше высказываний.



**Дизъюнкция** — логическая операция, которая двум высказываниям ставит в соответствие новое высказывание, являющееся ложным тогда и только тогда, когда оба исходных высказывания ложны.

Для записи дизъюнкции используются следующие знаки: ИЛИ,  $\vee$ ,  $|$ ,  $+$ . Например:  $A$  ИЛИ  $B$ ,  $A \vee B$ ,  $A|B$ ,  $A+B$ .

Дизъюнкция определяется следующей таблицей истинности:

$A$	$B$	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1



Дизъюнкцию также называют логическим сложением. Подумайте почему.

## Инверсия



**Инверсия** — логическая операция, которая высказыванию ставит в соответствие новое высказывание, значение которого противоположно исходному.

Для записи инверсии используются следующие знаки: НЕ,  $\neg$ ,  $\bar{\phantom{A}}$ .  
Например: НЕ  $A$ ,  $\neg A$ ,  $\bar{A}$ .

Инверсия определяется следующей таблицей истинности:

A	$\bar{A}$
0	1
1	0

Инверсию также называют логическим отрицанием.

Отрицанием высказывания «У меня дома есть компьютер» будет высказывание «Неверно, что у меня дома есть компьютер» или, что в русском языке то же самое, «У меня дома нет компьютера». Отрицанием высказывания «Я не знаю китайский язык» будет высказывание «Неверно, что я не знаю китайский язык» или, что в русском языке одно и то же, «Я знаю китайский язык». Отрицанием высказывания «Все юноши 8-х классов — отличники» является высказывание «Неверно, что все юноши 8-х классов — отличники», другими словами, «Не все юноши 8-х классов — отличники».

Таким образом, при построении отрицания к простому высказыванию либо используется речевой оборот «неверно, что ...», либо отрицание строится к сказуемому, тогда к соответствующему глаголу добавляется частица «не».

Любое сложное высказывание можно записать в виде логического выражения — выражения, содержащего логические переменные, знаки логических операций и скобки. Логические операции в логическом выражении выполняются в следующей очерёдности: инверсия, конъюнкция, дизъюнкция. Изменить порядок выполнения операций можно с помощью расстановки скобок.

Логические операции при выполнении имеют следующий приоритет: инверсия, конъюнкция, дизъюнкция.

**Пример 1.** Пусть  $A =$  «На web-странице встречается слово "крейсер"»,  $B =$  «На web-странице встречается слово "линкор"». Рассматривается некоторый сегмент сети Интернет, содержащий 5 000 000 web-страниц. В нём высказывание  $A$  истинно для 4800 страниц, высказывание  $B$  — для 4500 страниц, а высказывание  $A \vee B$  — для 7000 страниц. Для какого количества web-страниц в этом случае будут истинны следующие выражения и высказывание?



- а) НЕ ( $A$  ИЛИ  $B$ );
- б)  $A$  &  $B$ ;
- в) На web-странице встречается слово "крейсер" И не встречается слово "линкор".

*Решение.* Изобразим множество всех web-страниц рассматриваемого сектора сети Интернет кругом, внутри которого разместим два круга: одному из них соответствует множество web-страниц, где истинно высказывание  $A$ , второму — где истинно высказывание  $B$  (рис. 1.3).

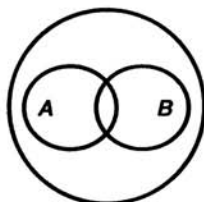


Рис. 1.3. Графическое изображение множеств web-страниц

Изобразим графически множества web-страниц, для которых истинны выражения и высказывание а) – в) (рис. 1.4).

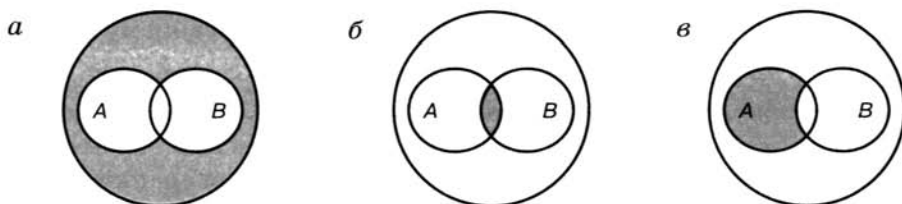


Рис. 1.4. Графические изображения множеств web-страниц, для которых истинны выражения и высказывание а) – в)

Построенные схемы помогут нам ответить на вопросы, содержащиеся в задании.

Выражение  $A$  ИЛИ  $B$  истинно для 7000 web-страниц, а всего страниц 5 000 000. Следовательно, выражение  $A$  ИЛИ  $B$  ложно для 4 993 000 web-страниц. Иначе говоря, для 4 993 000 web-страниц истинно выражение НЕ ( $A$  ИЛИ  $B$ ).

Выражение  $A \vee B$  истинно для тех web-страниц, где истинно  $A$  (4800), а также тех web-страниц, где истинно  $B$  (4500). Если бы все web-страницы были различны, то выражение  $A \vee B$  было бы истинно для 9300 (4800 + 4500) web-страниц. Но, согласно условию, таких web-страниц всего 7000. Это значит, что на 2300 (9300 – 7000)

web-страницах встречаются оба слова одновременно. Следовательно, выражение  $A \& B$  истинно для 2300 web-страниц.

Чтобы выяснить, для скольких web-страниц истинно высказывание  $A$  и одновременно ложно высказывание  $B$ , следует из 4800 вычесть 2300. Таким образом, высказывание «На web-странице встречается слово "крейсер" И не встречается слово "линкор"» истинно на 2500 web-страницах.

Самостоятельно запишите логическое выражение, соответствующее рассмотренному выше высказыванию.



На сайте Федерального центра информационно-образовательных ресурсов (<http://fcoir.edu.ru/>) размещён информационный модуль «Высказывание. Простые и сложные высказывания. Основные логические операции». Знакомство с этим ресурсом позволит вам расширить представления по изучаемой теме.



### 1.3.3. Построение таблиц истинности для логических выражений

Для логического выражения можно построить таблицу истинности, показывающую, какие значения принимает выражение при всех наборах значений входящих в него переменных. Для построения таблицы истинности следует:

- 1) подсчитать  $n$  — число переменных в выражении;
- 2) подсчитать общее число логических операций в выражении;
- 3) установить последовательность выполнения логических операций с учётом скобок и приоритетов;
- 4) определить число столбцов в таблице: число переменных + число операций;
- 5) заполнить шапку таблицы, включив в неё переменные и операции в соответствии с последовательностью, установленной в п. 3;
- 6) определить число строк в таблице (не считая шапки таблицы):  $m = 2^n$ ;
- 7) выписать наборы входных переменных с учётом того, что они представляют собой ряд целых  $n$ -разрядных двоичных чисел от 0 до  $2^n - 1$ ;
- 8) провести заполнение таблицы по столбцам, выполняя логические операции в соответствии с установленной последовательностью.



Построим таблицу истинности для логического выражения  $A \vee A \& B$ . В нём две переменные, две операции, причём сначала выполняется конъюнкция, а затем — дизъюнкция. Всего в таблице будет четыре столбца:

$A$	$B$	$A \& B$	$A \vee A \& B$
-----	-----	----------	-----------------

Наборы входных переменных — это целые числа от 0 до 3, представленные в двухразрядном двоичном коде: 00, 01, 10, 11.

Заполненная таблица истинности имеет вид:

$A$	$B$	$A \& B$	$A \vee A \& B$
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

Обратите внимание, что последний столбец (результат) совпал со столбцом  $A$ . В таком случае говорят, что логическое выражение  $A \vee A \& B$  равносильно логической переменной  $A$ .

### 1.3.4. Свойства логических операций

Рассмотрим основные свойства логических операций, называемые также законами алгебры логики.

#### 1. Переместительный (коммутативный) закон:

- для логического умножения:  
 $A \& B = B \& A$ ;

- для логического сложения:  
 $A \vee B = B \vee A$ .

#### 2. Сочетательный (ассоциативный) закон:

- для логического умножения:  
 $(A \& B) \& C = A \& (B \& C)$ ;

- для логического сложения:  
 $(A \vee B) \vee C = A \vee (B \vee C)$ .

При одинаковых знаках операций скобки можно ставить произвольно или вообще опускать.

**3. Распределительный (дистрибутивный) закон:**

- для логического умножения:

$$A \& (B \vee C) = (A \& B) \vee (A \& C);$$

- для логического сложения:

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

**4. Закон двойного отрицания:**

$$\overline{\overline{A}} = A.$$

Двойное отрицание исключает отрицание.

**5. Закон исключённого третьего:**

- для логического умножения:

$$A \& \overline{A} = 0;$$

- для логического сложения:

$$A \vee \overline{A} = 1.$$

Из двух противоречивых высказываний об одном и том же предмете одно всегда истинно, а второе — ложно, третьего не дано.

**6. Закон повторения:**

- для логического умножения:

$$A \& A = A;$$

- для логического сложения:

$$A \vee A = A.$$

**7. Законы операций с 0 и 1:**

- для логического умножения:

$$A \& 0 = 0; A \& 1 = A;$$

- для логического сложения:

$$A \vee 0 = A; A \vee 1 = 1.$$

**8. Законы общей инверсии:**

- для логического умножения:

$$\overline{A \& B} = \overline{A} \vee \overline{B};$$

- для логического сложения:

$$\overline{A \vee B} = \overline{A} \& \overline{B}.$$

Законы алгебры логики могут быть доказаны с помощью таблиц истинности.

Докажем распределительный закон для логического сложения:

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



A	B	C	$B \& C$	$A \vee (B \& C)$	$A \vee B$	$A \vee C$	$(A \vee B) \& (A \vee C)$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

Совпадение значений в столбцах, соответствующих логическим выражениям в левой и правой частях равенства, доказывает справедливость распределительного закона для логического сложения.



**Пример 2.** Найдём значение логического выражения  $(X < 3) \& \overline{(X < 2)}$  для числа  $X = 0$ .



**Решение.** При  $X = 0$  получаем следующее логическое выражение:  $(0 < 3) \& \overline{(0 < 2)}$ . Так как логические выражения  $0 < 3$ ,  $0 < 2$  истинны, то, подставив их значения в логическое выражение, получаем:  $1 \& \bar{1} = 1 \& 0 = 0$ .

## 1.3.5. Решение логических задач

Рассмотрим несколько способов решения логических задач.



**Задача 1.** Коля, Вася и Серёжа гостили летом у бабушки. Однажды один из мальчиков нечаянно разбил любимую бабушкину вазу. На вопрос, кто разбил вазу, они дали такие ответы:



Серёжа: 1) Я не разбивал. 2) Вася не разбивал.

Вася: 3) Серёжа не разбивал. 4) Вазу разбил Коля.

Коля: 5) Я не разбивал. 6) Вазу разбил Серёжа.

Бабушка знала, что один из её внуков, назовём его правдивым, оба раза сказал правду; второй, назовём его шутником, оба раза сказал неправду; третий, назовём его хитрецом, один раз сказал правду, а другой раз — неправду. Назовите имена правдивого, шутника и хитреца. Кто из внуков разбил вазу?

**Решение.** Пусть  $K$  = «Коля разбил вазу»,  $B$  = «Вася разбил вазу»,  $C$  = «Серёжа разбил вазу». Для решения задачи можно соста-



вить таблицу истинности, в которой представить высказывания каждого мальчика. Так как ваза разбита одним внуком, то чтобы выяснить, кто именно это сделал, достаточно фрагмента таблицы истинности, содержащего наборы значений входных переменных: 001, 010, 100.

К	В	С	Утверждения Серёжи		Утверждения Васи		Утверждения Коли	
			$\bar{C}$	$\bar{B}$	$\bar{C}$	К	$\bar{K}$	С
0	0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1	0
1	0	0	1	1	1	1	0	0

Исходя из того, что знает о внуках бабушка, следует искать в таблице строку, содержащую в каком-либо порядке три комбинации значений: 00 (слова шутника), 11 (слова правдивого внука), 01 или 10 (слова хитреца). Такая строка отмечена галочкой. Согласно этой строке, вазу разбил Серёжа, он же оказался хитрецом. Шутником оказался Вася. Имя правдивого внука — Коля.

**Задача 2.** В соревнованиях по гимнастике участвуют Алла, Валя, Сима и Даша. Болельщики высказали предположения о возможных победителях:

- 1) Сима будет первой, Валя — второй;
- 2) Сима будет второй, Даша — третьей;
- 3) Алла будет второй, Даша — четвёртой.

По окончании соревнований оказалось, что в каждом из предположений только одно из высказываний истинно, другое ложно. Какое место на соревнованиях заняла каждая из девушек, если все они оказались на разных местах?

*Решение.* Рассмотрим простые высказывания:

- $S_1$  = «Сима заняла первое место»;
- $B_2$  = «Валя заняла второе место»;
- $S_2$  = «Сима заняла второе место»;
- $D_3$  = «Даша заняла третье место»;
- $A_2$  = «Алла заняла второе место»;
- $D_4$  = «Даша заняла четвёртое место».

Так как в каждом из трёх предположений одно из высказываний истинно, а другое ложно, то можно заключить следующее:



$$1) C_1 + B_2 = 1, C_1 \cdot B_2 = 0;$$

$$2) C_2 + D_3 = 1, C_2 \cdot D_3 = 0;$$

$$3) A_2 + D_4 = 1, A_2 \cdot D_4 = 0.$$

Логическое произведение истинных высказываний будет истинным:

$$(C_1 + B_2) \cdot (C_2 + D_3) \cdot (A_2 + D_4) = 1.$$

На основании распределительного закона преобразуем левую часть этого выражения:

$$(C_1 \cdot C_2 + C_1 \cdot D_3 + B_2 \cdot C_2 + B_2 \cdot D_3) \cdot (A_2 + D_4) = 1.$$

Высказывание  $C_1 \cdot C_2$  означает, что Сима заняла и первое, и второе места. Согласно условию задачи, это высказывание ложно. Ложным является и высказывание  $B_2 \cdot C_2$ . Учитывая закон операций с константой 0, запишем:


$$(C_1 \cdot D_3 + B_2 \cdot D_3) \cdot (A_2 + D_4) = 1.$$

Дальнейшее преобразование левой части этого равенства и исключение заведомо ложных высказываний дают:

$$C_1 \cdot D_3 \cdot A_2 + C_1 \cdot D_3 \cdot D_4 + B_2 \cdot D_3 \cdot A_2 + B_2 \cdot D_3 \cdot D_4 = 1.$$

$$C_1 \cdot D_3 \cdot A_2 = 1.$$

Из последнего равенства следует, что  $C_1 = 1$ ,  $D_3 = 1$ ,  $A_2 = 1$ . Это означает, что Сима заняла первое место, Алла — второе, Даша — третье. Следовательно, Валя заняла четвертое место.

 Познакомьтесь с другими способами решения логических задач, а также принять участие в интернет-олимпиадах и конкурсах по их решению вы сможете на российской странице международного математического конкурса «Кенгуру» (<http://mathkang.ru/>).

На сайте <http://www.kaser.com/> вы сможете скачать демонстрационную версию очень полезной, развивающей логику и умение рассуждать логической головоломки Шерлок.

## 1.3.6. Логические элементы

Алгебра логики — раздел математики, играющий важную роль в конструировании автоматических устройств, разработке аппаратных и программных средств информационных и коммуникационных технологий.

Вы уже знаете, что любая информация может быть представлена в дискретной форме — в виде фиксированного набора отдельных значений. Устройства, которые обрабатывают такие значения (сигналы), называются дискретными. Дискретный преобразователь, который выдаёт после обработки двоичных сигналов значение одной из логических операций, называется логическим элементом.

На рисунке 1.5 приведены условные обозначения (схемы) логических элементов, реализующих логическое умножение, логическое сложение и инверсию.

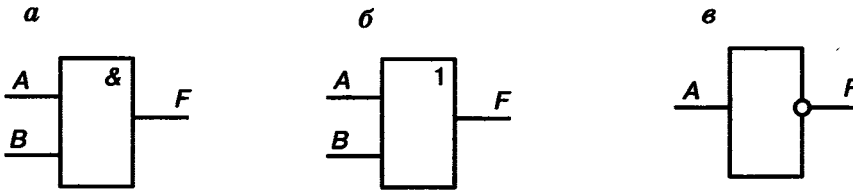


Рис. 1.5. Логические элементы

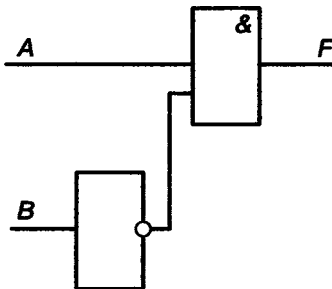
Логический элемент **И** (конъюнктор) реализует операцию логического умножения (рис. 1.5, а). Единица на выходе этого элемента появится только тогда, когда на всех входах будут единицы.

Логический элемент **ИЛИ** (дизъюнктор) реализует операцию логического сложения (рис. 1.5, б). Если хотя бы на одном входе будет единица, то на выходе элемента также будет единица.

Логический элемент **НЕ** (инвертор) реализует операцию отрицания (рис. 1.5, в). Если на входе элемента 0, то на выходе 1 и наоборот.

Компьютерные устройства, производящие операции над двоичными числами, и ячейки, хранящие данные, представляют собой электронные схемы, состоящие из отдельных логических элементов. Более подробно эти вопросы будут раскрыты в курсе информатики 10–11 классов.

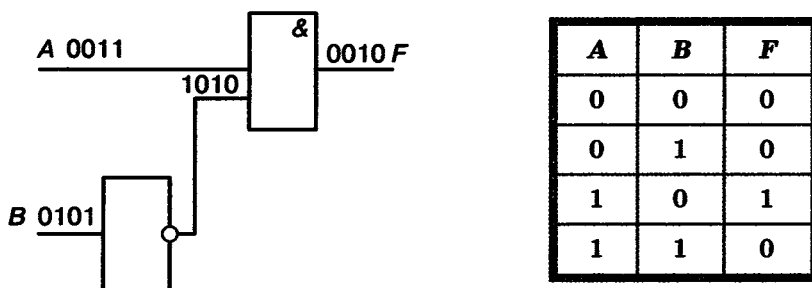
**Пример 3.** Проанализируем электронную схему, т. е. выясним, какой сигнал должен быть на выходе  $F$  при каждом возможном наборе сигналов  $A$  и  $B$  на входах.



**Решение.** Все возможные комбинации сигналов  $A$  и  $B$  на входах внесём в таблицу истинности. Проследим преобразование каждой пары сигналов при прохождении их через логические элементы



и запишем полученный результат в таблицу. Заполненная таблица истинности полностью описывает рассматриваемую электронную схему.



Таблицу истинности можно построить и по логическому выражению, соответствующему электронной схеме. Последний логический элемент в рассматриваемой схеме — конъюнктор. В него поступают сигналы от входа  $A$  и от инвертора. В свою очередь, в инвертор поступает сигнал от входа  $B$ . Таким образом,  $F = A \& \bar{B}$ .

Составить более полное представление о логических элементах и электронных схемах вам поможет работа с тренажёром «Логика» (<http://kpolyakov.narod.ru/prog/logic.htm>).

## САМОЕ ГЛАВНОЕ

Высказывание — это предложение на любом языке, содержание которого можно однозначно определить как истинное или ложное.

Основные логические операции, определённые над высказываниями: инверсия, конъюнкция, дизъюнкция.

Название логической операции	Логическая связка	Обозначение
Инверсия	«не», «неверно, что»	$\neg$ , —
Конъюнкция	«и», «а», «но», «хотя»	$\&$
Дизъюнкция	«или»	$\vee$

Таблицы истинности для основных логических операций:

$A$	$\bar{A}$
0	1
1	0

$A$	$B$	$A \& B$	$A \vee B$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

При вычислении логических выражений сначала выполняются действия в скобках. Приоритет выполнения логических операций:  $\neg$ ,  $\&$ ,  $\vee$ .

Вопросы и задания

1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Дополняет ли презентация информацию, содержащуюся в тексте параграфа?
2. Объясните, почему следующие предложения не являются высказываниями.
  - 1) Какого цвета этот дом?
  - 2) Число  $X$  не превосходит единицы.
  - 3)  $4X + 3$ .
  - 4) Посмотрите в окно.
  - 5) Пейте томатный сок!
  - 6) Эта тема скучна.
  - 7) Рикки Мартин — самый популярный певец.
  - 8) Вы были в театре?
3. Приведите по одному примеру истинных и ложных высказываний из биологии, географии, информатики, истории, математики, литературы.
4. В следующих высказываниях выделите простые высказывания, обозначив каждое из них буквой; запишите с помощью букв и знаков логических операций каждое составное высказывание.
  - 1) Число 376 чётное и трёхзначное.
  - 2) Зимой дети катаются на коньках или на лыжах.
  - 3) Новый год мы встретим на даче или на Красной площади.



- 4) Неверно, что Солнце движется вокруг Земли.
- 5) Земля имеет форму шара, который из космоса кажется голубым.
- 6) На уроке математики старшеклассники отвечали на вопросы учителя, а также писали самостоятельную работу.
5. Постройте отрицания следующих высказываний.
- 1) Сегодня в театре идёт опера «Евгений Онегин».
- 2) Каждый охотник желает знать, где сидит фазан.
- 3) Число 1 есть простое число.
- 4) Натуральные числа, оканчивающиеся цифрой 0, не являются простыми числами.
- 5) Неверно, что число 3 не является делителем числа 198.
- 6) Коля решил все задания контрольной работы.
- 7) Во всякой школе некоторые ученики интересуются спортом.
- 8) Некоторые млекопитающие не живут на суше.
6. Пусть  $A = \text{«Ане нравятся уроки математики»}$ , а  $B = \text{«Ане нравятся уроки химии»}$ . Выразите следующие формулы на обычном языке:

- 1)  $\overline{A \ \& \ B}$ ;      4)  $A \vee \overline{B}$ ;      7)  $\overline{\overline{A \ \& \ B}}$ ;  
 2)  $\overline{A \ \& \ B}$ ;      5)  $\overline{A \ \vee \ B}$ ;      8)  $\overline{\overline{A \ \vee \ B}}$ ;  
 3)  $A \ \& \ \overline{B}$ ;      6)  $\overline{A \ \vee \ B}$ ;      9)  $(A \ \& \ B)$ .

7. Некоторый сегмент сети Интернет состоит из 1000 сайтов. Поисковый сервер в автоматическом режиме составил таблицу ключевых слов для сайтов этого сегмента. Вот её фрагмент:

Ключевое слово	Количество сайтов, для которых данное слово является ключевым
сомики	250
меченосцы	200
гулли	500

По запросу *сомики & гулли* было найдено 0 сайтов, по запросу *сомики & меченосцы* — 20 сайтов, а по запросу *меченосцы & гулли* — 10 сайтов.

Сколько сайтов будет найдено по запросу *сомики | меченосцы | гулли*?

Для скольких сайтов рассматриваемого сегмента ложно высказывание «*Сомики — ключевое слово сайта ИЛИ меченосцы — ключевое слово сайта ИЛИ гуппи — ключевое слово сайта*»?

8. Постройте таблицы истинности для следующих логических выражений:



1)  $B \& (A \vee B)$ ;

2)  $A \& (B \vee \overline{B})$ ;

3)  $A \& (A \vee B \vee C)$ ;

4)  $A \vee B \vee \overline{C}$ .

9. Проведите доказательство рассмотренных в параграфе логических законов с помощью таблиц истинности.



10. Даны три числа в десятичной системе счисления:  $A = 23$ ,  $B = 19$ ,  $C = 26$ . Переведите  $A$ ,  $B$  и  $C$  в двоичную систему счисления и выполните поразрядно логические операции  $(A \vee B) \& C$ . Ответ дайте в десятичной системе счисления.



11. Найдите значения выражений:



1)  $(1 \vee 1) \vee (1 \vee 0)$ ;

2)  $((1 \vee 0) \vee 1) \vee 1$ ;

3)  $(0 \& 1) \& 1$ ;

4)  $1 \& (1 \& 1) \& 1$ ;

5)  $((1 \vee 0) \& (1 \& 1)) \& (0 \vee 1)$ ;

6)  $((1 \& 1) \vee 0) \& (0 \vee 1)$ ;

7)  $((0 \& 0) \vee 0) \& (1 \vee 1)$ ;

8)  $(A \vee 1) \vee (B \vee 0)$ ;

9)  $((1 \& A) \vee (B \& 0)) \vee 1$ ;

10)  $1 \vee A \& 0$ .

12. Найдите значение логического выражения  $\overline{(X < 3)} \& \overline{(X < 2)}$  для указанных значений числа  $X$ :



1) 1

2) 2

3) 3

4) 4

13. Пусть  $A =$  «Первая буква имени — гласная»,  $B =$  «Четвёртая буква имени согласная». Найдите значение логического выражения  $\overline{A} \vee B$  для следующих имён:



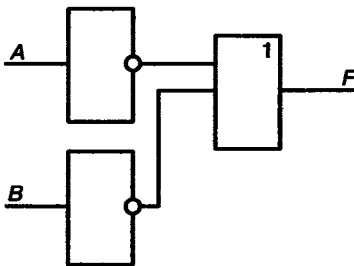
1) ЕЛЕНА

2) ВАДИМ

3) АНТОН

4) ФЁДОР

14. Разбирается дело Джона, Брауна и Смита. Известно, что один из них нашёл и утаил клад. На следствии каждый из подозреваемых сделал два заявления:  
Смит: «Я не делал этого. Браун сделал это».  
Джон: «Браун не виновен. Смит сделал это».  
Браун: «Я не делал этого. Джон не делал этого».  
Суд установил, что один из них дважды солгал, другой дважды сказал правду, третий один раз солгал, один раз сказал правду. Кто из подозреваемых должен быть оправдан?
15. Алёша, Боря и Гриша нашли в земле старинный сосуд. Рассматривая удивительную находку, каждый высказал по два предположения:  
1) Алёша: «Это сосуд греческий и изготовлен в V веке».  
2) Боря: «Это сосуд финикийский и изготовлен в III веке».  
3) Гриша: «Это сосуд не греческий и изготовлен в IV веке».  
Учитель истории сказал ребятам, что каждый из них прав только в одном из двух предположений. Где и в каком веке изготовлен сосуд?
16. Выясните, какой сигнал должен быть на выходе электронной схемы при каждом возможном наборе сигналов на входах. Составьте таблицу работы схемы. Каким логическим выражением описывается схема?





## Тестовые задания для самоконтроля



1. Совокупность знаков, с помощью которых записываются числа, называется:
  - а) системой счисления
  - б) цифрами системы счисления
  - в) алфавитом системы счисления
  - г) основанием системы счисления
  
2. Чему равен результат сложения двух чисел, записанных римскими цифрами:  $MCM + LXVIII$ ?
  - а) 1168
  - б) 1968
  - в) 2168
  - г) 1153
  
3. Число 301011 может существовать в системах счисления с основаниями:
  - а) 2 и 10
  - б) 4 и 3
  - в) 4 и 8
  - г) 2 и 4
  
4. Двоичное число 100110 в десятичной системе счисления записывается как:
  - а) 36
  - б) 38
  - в) 37
  - г) 46



5. В классе  $110010_2\%$  девочек и  $1010_2$  мальчиков. Сколько учеников в классе?
- 10
  - 20
  - 30
  - 40
6. Сколько цифр 1 в двоичном представлении десятичного числа 15?
- 1
  - 2
  - 3
  - 4
7. Чему равен результат сложения чисел  $110_2$  и  $12_8$ ?
- $6_{10}$
  - $10_{10}$
  - $10000_2$
  - $17_8$
8. Ячейка памяти компьютера состоит из однородных элементов, называемых:
- кодами
  - разрядами
  - цифрами
  - коэффициентами
9. Количество разрядов, занимаемых двухбайтовым числом, равно:
- 8
  - 16
  - 32
  - 64
10. В знаковый разряд ячейки для отрицательных чисел заносится:
- +
  - 
  - 0
  - 1

11. Вещественные числа представляются в компьютере в:
- а) естественной форме
  - б) развёрнутой форме
  - в) экспоненциальной форме с нормализованной мантиссой
  - г) виде обыкновенной дроби
12. Какое предложение не является высказыванием?
- а) Никакая причина не извиняет невежливость.
  - б) Обязательно стань отличником.
  - в) Рукописи не горят.
  - г)  $1011_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$
13. Какое высказывание является ложным?
- а) Знаком  $\vee$  обозначается логическая операция ИЛИ.
  - б) Логическую операцию ИЛИ также называют логическим сложением.
  - в) Дизъюнкцию также называют логическим сложением.
  - г) Знаком  $\vee$  обозначается логическая операция конъюнкция.
14. Для какого из указанных значений числа  $X$  истинно высказывание  $((X < 5) \vee (X < 3)) \wedge ((X < 2) \vee (X < 1))$  ?
- а) 1
  - б) 2
  - в) 3
  - г) 4
15. Для какого символического выражения верно высказывание: «НЕ (Первая буква согласная) И НЕ (Вторая буква гласная)»?
- а) abcde
  - б) bcade
  - в) babas
  - г) cabab
16. Некоторый сегмент сети Интернет состоит из 1000 сайтов. Поисковый сервер в автоматическом режиме составил таблицу ключевых слов для сайтов этого сегмента. Вот её фрагмент:

Ключевое слово	Количество сайтов, для которых данное слово является ключевым
сканер	200
принтер	250
монитор	450

Сколько сайтов будет найдено по запросу *принтер | сканер | монитор*, если по запросу *принтер | сканер* было найдено 450 сайтов, по запросу *принтер & монитор* — 40, а по запросу *сканер & монитор* — 50?

- а) 900
- б) 540
- в) 460
- г) 810



17. Какому логическому выражению соответствует следующая таблица истинности?

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

- а)  $A \& B$
- б)  $A \vee B$
- в)  $\overline{A \& B}$
- г)  $A \& \overline{B}$



18. Когда сломался компьютер, его хозяин сказал: «Оперативная память не могла выйти из строя». Сын хозяина компьютера предположил, что вышел из строя процессор, а жёсткий диск исправен. Пришедший специалист по обслуживанию сказал, что, скорее всего, с процессором всё в порядке, а оперативная память неисправна. В результате оказалось, что двое из них сказали всё верно, а третий — всё неверно. Что же сломалось?

- а) оперативная память
- б) процессор
- в) жёсткий диск
- г) процессор и оперативная память



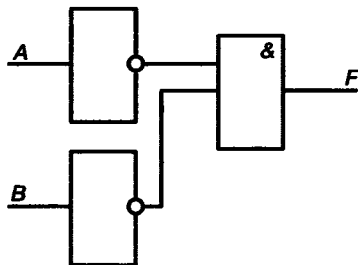
19. На перекрёстке произошло дорожно-транспортное происшествие, в котором участвовали автобус (А), грузовик (Г), легковой автомобиль (Л) и маршрутное такси (М). Свидетели происшествия дали следующие показания. Первый свидетель считал, что первым на перекрёсток выехал автобус, а маршрутное так-



си было вторым. Другой свидетель полагал, что последним на перекрёсток выехал легковой автомобиль, а вторым был грузовик. Третий свидетель уверял, что автобус выехал на перекрёсток вторым, а следом за ним — легковой автомобиль. В результате оказалось, что каждый из свидетелей был прав только в одном из своих утверждений. В каком порядке выехали машины на перекрёсток? В вариантах ответов перечислены подряд без пробелов первые буквы названий транспортных средств в порядке их выезда на перекрёсток:

- а) АМЛГ
- б) АГЛМ
- в) ГЛМА
- г) МЛГА

20. Какое логическое выражение соответствует следующей схеме?



- а)  $A \& B$
- б)  $A \vee B$
- в)  $\overline{A \& B}$
- г)  $\overline{A \& \overline{B}}$

Для проверки знаний и умений по теме «Математические основы информатики» вы можете воспользоваться интерактивным тестом к главе 1, содержащимся в электронном приложении к учебнику.



# Глава 2

## ОСНОВЫ АЛГОРИТМИЗАЦИИ

### § 2.1

#### Алгоритмы и исполнители

**Ключевые слова:**

- алгоритм
- свойства алгоритма
  - дискретность
  - понятность
  - определённости
  - результативность
  - массовость
- исполнитель
- характеристики исполнителя
  - круг решаемых задач
  - среда
  - режим работы
  - система команд
- формальное исполнение алгоритма

#### 2.1.1. Понятие алгоритма

Каждый человек в повседневной жизни, в учёбе или на работе решает огромное количество задач самой разной сложности. Сложные задачи требуют длительных размышлений для нахождения решения; простые и привычные задачи человек решает не задумываясь, автоматически. В большинстве случаев решение каждой задачи можно разбить на простые этапы (шаги). Для многих таких задач (установка программного обеспечения, сборка шкафа, создание сай-

та, эксплуатация технического устройства, покупка авиабилета через Интернет и т. д.) уже разработаны и предлагаются пошаговые инструкции, при последовательном выполнении которых можно прийти к желаемому результату.

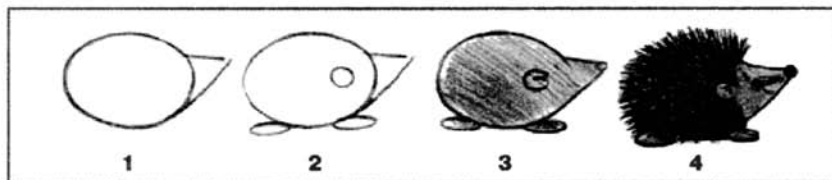
**Пример 1.** Задача «Найти среднее арифметическое двух чисел» решается в три шага:

- 1) задумать два числа;
- 2) сложить два задуманных числа;
- 3) полученную сумму разделить на 2.

**Пример 2.** Задача «Внести деньги на счёт телефона» подразделяется на следующие шаги:

- 1) подойти к терминалу по оплате платежей;
- 2) выбрать оператора связи;
- 3) ввести номер телефона;
- 4) проверить правильность введённого номера;
- 5) вставить денежную купюру в купюроприёмник;
- 6) дождаться сообщения о зачислении денег на счёт;
- 7) получить чек.

**Пример 3.** Этапы решения задачи «Нарисовать весёлого ёжика» представлены графически:



Нахождение среднего арифметического, внесение денег на телефонный счёт и рисование ежа — на первый взгляд совершенно разные процессы. Но у них есть общая черта: каждый из этих процессов описывается последовательностями кратких указаний, точное следование которым позволяет получить требуемый результат. Последовательности указаний, приведённые в примерах 1–3, являются алгоритмами решения соответствующих задач. Исполнитель этих алгоритмов — человек.

Алгоритм может представлять собой описание некоторой последовательности вычислений (пример 1) или шагов нематематического характера (примеры 2–3). Но в любом случае перед его разработкой должны быть чётко определены начальные условия (исходные

данные) и то, что предстоит получить (результат). Можно сказать, что алгоритм — это описание последовательности шагов в решении задачи, приводящих от исходных данных к требуемому результату.

В общем виде схему работы алгоритма можно представить следующим образом (рис. 2.1).



Рис. 2.1. Общая схема работы алгоритма

Алгоритмами являются изучаемые в школе правила сложения, вычитания, умножения и деления чисел, многие грамматические правила, правила геометрических построений и т. д.



Анимации «Работа с алгоритмом» (193576), «Наибольший общий делитель» (170363), «Наименьшее общее кратное» (170390) помогут вам вспомнить некоторые алгоритмы, изученные на уроках русского языка и математики (<http://sc.edu.ru/>).



**Пример 4.** Некоторый алгоритм приводит к тому, что из одной цепочки символов получается новая цепочка следующим образом:

1. Вычисляется длина (в символах) исходной цепочки символов.
2. Если длина исходной цепочки нечётна, то к исходной цепочке справа приписывается цифра 1, иначе цепочка не изменяется.
3. Символы попарно меняются местами (первый — со вторым, третий — с четвёртым, пятый — с шестым и т. д.).
4. Справа к полученной цепочке приписывается цифра 2.

Получившаяся таким образом цепочка является результатом работы алгоритма.

Так, если исходной была цепочка  $A\#B$ , то результатом работы алгоритма будет цепочка  $\#A1B2$ , а если исходной цепочкой была  $ABV@$ , то результатом работы алгоритма будет цепочка  $BA@B2$ .

### 2.1.2. Исполнитель алгоритма

Каждый алгоритм предназначен для определённого исполнителя.



**Исполнитель** — это некоторый объект (человек, животное, техническое устройство), способный выполнять определённый набор команд.



Различают формальных и неформальных исполнителей. Формальный исполнитель одну и ту же команду всегда выполняет одинаково. Неформальный исполнитель может выполнять команду по-разному.

Рассмотрим более подробно множество формальных исполнителей. Формальные исполнители необычайно разнообразны, но для каждого из них можно указать следующие характеристики: круг решаемых задач (назначение), среду, систему команд и режим работы.

**Круг решаемых задач.** Каждый исполнитель создаётся для решения некоторого круга задач — построения цепочек символов, выполнения вычислений, построения рисунков на плоскости и т. д.

**Среда исполнителя.** Область, обстановку, условия, в которых действует исполнитель, принято называть средой данного исполнителя. Исходные данные и результаты любого алгоритма всегда принадлежат среде того исполнителя, для которого предназначен алгоритм.

**Система команд исполнителя.** Предписание исполнителю о выполнении отдельного законченного действия называется командой. Совокупность всех команд, которые могут быть выполнены некоторым исполнителем, образует систему команд данного исполнителя (СКИ). Алгоритм составляется с учётом возможностей конкретного исполнителя, иначе говоря, в системе команд исполнителя, который будет его выполнять.

**Режимы работы исполнителя.** Для большинства исполнителей предусмотрены режимы *непосредственного управления* и *программного управления*. В первом случае исполнитель ожидает команд от человека и каждую поступившую команду немедленно выполняет. Во втором случае исполнителю сначала задаётся полная последовательность команд (программа), а затем он выполняет все эти команды в автоматическом режиме. Ряд исполнителей работает только в одном из названных режимов.

Рассмотрим примеры исполнителей.

**Пример 5.** Исполнитель Черепашка перемещается на экране компьютера, оставляя след в виде линии. Система команд Черепашки состоит из следующих команд:

Вперёд  $n$  (где  $n$  — целое число) — вызывает передвижение Черепашки на  $n$  шагов в направлении движения — в том направлении, куда развёрнута её голова и корпус;



Направо  $m$  (где  $m$  — целое число) — вызывает изменение направления движения Черепашки на  $m$  градусов по часовой стрелке.

Запись Повтори  $k$  [ $\langle$ Команда1 $\rangle$   $\langle$ Команда2 $\rangle$  ...  $\langle$ Команда $n$  $\rangle$ ] означает, что последовательность команд в скобках повторится  $k$  раз.



Подумайте, какая фигура появится на экране после выполнения Черепашкой следующего алгоритма.

Повтори 12 [Направо 45 Вперёд 20 Направо 45]



**Пример 6.** Система команд исполнителя **Вычислитель** состоит из двух команд, которым присвоены номера:

- 1 — вычти 1
- 2 — умножь на 3

Первая из них уменьшает число на 1, вторая увеличивает число в 3 раза. При записи алгоритмов для краткости указываются лишь номера команд. Например, алгоритм 21212 означает следующую последовательность команд:

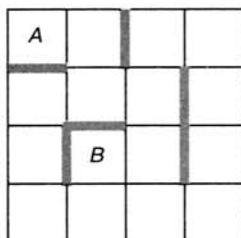
- умножь на 3
- вычти 1
- умножь на 3
- вычти 1
- умножь на 3

С помощью этого алгоритма число 1 будет преобразовано в 15:  
 $((1 \cdot 3 - 1) \cdot 3 - 1) \cdot 3 = 15$ .



**Пример 7.** Исполнитель **Робот** действует на клетчатом поле, между соседними клетками которого могут стоять стены. Робот передвигается по клеткам поля и может выполнять следующие команды, которым присвоены номера:

- 1 — вверх
- 2 — вниз
- 3 — вправо
- 4 — влево



При выполнении каждой такой команды Робот перемещается в соседнюю клетку в указанном направлении. Если же в этом направлении между клетками стоит стена, то Робот разрушается.

Что произойдёт с Роботом, если он выполнит последовательность команд 32323 (здесь цифры обозначают номера команд), начав движение из клетки *A*? Какую последовательность команд следует выполнить Роботу, чтобы переместиться из клетки *A* в клетку *B*, не разрушившись от встречи со стенами?



При разработке алгоритма:

- 1) выделяются фигурирующие в задаче объекты, устанавливаются свойства объектов, отношения между объектами и возможные действия с объектами;
- 2) определяются исходные данные и требуемый результат;
- 3) определяется последовательность действий исполнителя, обеспечивающая переход от исходных данных к результату;
- 4) последовательность действий записывается с помощью команд, входящих в систему команд исполнителя.



Можно сказать, что алгоритм — модель деятельности исполнителя алгоритмов.

### 2.1.3. Свойства алгоритма

Не любая инструкция, последовательность предписаний или план действий может считаться алгоритмом. Каждый алгоритм обязательно обладает следующими свойствами: дискретность, понятность, определённости, результативность и массовость.

**Свойство дискретности** означает, что путь решения задачи разделён на отдельные шаги (действия). Каждому действию соответствует предписание (команда). Только выполнив одну команду, исполнитель может приступить к выполнению следующей команды.

**Свойство понятности** означает, что алгоритм состоит только из команд, входящих в систему команд исполнителя, т. е. из таких команд, которые исполнитель может воспринять и по которым может выполнить требуемые действия.

**Свойство определённости** означает, что в алгоритме нет команд, смысл которых может быть истолкован исполнителем неоднозначно; недопустимы ситуации, когда после выполнения очередной команды исполнителю неясно, какую команду выполнять следующей. Благодаря этому результат алгоритма однозначно определяется набором исходных данных: если алгоритм несколько раз применяется к одному и тому же набору исходных данных, то на выходе всегда получается один и тот же результат.

**Свойство результативности** означает, что алгоритм должен обеспечивать получение результата после конечного, возможно, очень большого, числа шагов. При этом результатом считается не только

обусловленный постановкой задачи ответ, но и вывод о невозможности продолжения по какой-либо причине решения данной задачи.

Свойство массовости означает, что алгоритм должен обеспечивать возможность его применения для решения любой задачи из некоторого класса задач. Например, алгоритм нахождения корней квадратного уравнения должен быть применим к любому квадратному уравнению, алгоритм перехода улицы должен быть применим в любом месте улицы, алгоритм приготовления лекарства должен быть применим для приготовления любого его количества и т. д.



**Пример 8.** Рассмотрим один из методов нахождения всех простых чисел, не превышающих некоторое натуральное число  $n$ . Этот метод называется «решето Эратосфена» по имени предложившего его древнегреческого учёного Эратосфена (III в. до н. э.).

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

- 1) выписать подряд все натуральные числа от 2 до  $n$  (2, 3, 4, ...,  $n$ );
- 2) заключить в рамку 2 — первое простое число;
- 3) вычеркнуть из списка все числа, делящиеся на последнее найденное простое число;
- 4) найти первое неотмеченное число (отмеченные числа — зачёркнутые числа или числа, заключённые в рамку) и заключить его в рамку — это будет очередное простое число;
- 5) повторять шаги 3 и 4 до тех пор, пока не останется неотмеченных чисел.



Более наглядное представление о методе нахождения простых чисел вы сможете получить с помощью размещённой в Единой коллекции цифровых образовательных ресурсов анимации «Решето Эратосфена» (180279).

Рассмотренная последовательность действий является алгоритмом, так как она удовлетворяет свойствам:

- дискретности — процесс нахождения простых чисел разбит на шаги;
- понятности — каждая команда понятна ученику 8 класса, выполняющему этот алгоритм;
- определённости — каждая команда трактуется и выполняется исполнителем однозначно; имеются указания об очередности выполнения команд;
- результативности — через некоторое число шагов достигается результат;

- массовости — последовательность действий применима для любого натурального  $n$ .

Рассмотренные свойства алгоритма позволяют дать более точное определение алгоритма.

---

**Алгоритм** — это предназначенное для конкретного исполнителя описание последовательности действий, приводящих от исходных данных к требуемому результату, которое обладает свойствами дискретности, понятности, определённости, результативности и массовости.

---



#### 2.1.4. Возможность автоматизации деятельности человека

Разработка алгоритма — как правило, трудоёмкая задача, требующая от человека глубоких знаний, изобретательности и больших временных затрат.

Решение задачи по готовому алгоритму требует от исполнителя только строгого следования заданным предписаниям.

**Пример 9.** Из кучки, содержащей любое, большее трёх, количество каких-либо предметов, двое играющих по очереди берут по одному или по два предмета. Выигрывает тот, кто своим очередным ходом сможет забрать все оставшиеся предметы.

Рассмотрим алгоритм, следуя которому первый игрок наверняка обеспечит себе выигрыш.

1. Если число предметов в кучке кратно 3, то уступить ход противнику, иначе начать игру, взяв 1 или 2 предмета так, чтобы осталось количество предметов, кратное 3.
2. Своим очередным ходом каждый раз дополнять число предметов, взятых соперником, до 3 (число оставшихся предметов должно быть кратно 3).

Исполнитель может не вникать в смысл того, что он делает, и не рассуждать, почему он поступает так, а не иначе, т. е. он может действовать формально. Способность исполнителя действовать формально обеспечивает возможность автоматизации деятельности человека. Для этого:

- 1) процесс решения задачи представляется в виде последовательности простейших операций;
- 2) создаётся машина (автоматическое устройство), способная выполнять эти операции в последовательности, заданной в алгоритме;



- 3) человек освобождается от рутинной деятельности, выполнение алгоритма поручается автоматическому устройству.

### САМОЕ ГЛАВНОЕ

**Исполнитель** — некоторый объект (человек, животное, техническое устройство), способный выполнять определённый набор команд.

**Формальный исполнитель** одну и ту же команду всегда выполняет одинаково. Для каждого формального исполнителя можно указать: круг решаемых задач, среду, систему команд и режим работы.

**Алгоритм** — предназначенное для конкретного исполнителя описание последовательности действий, приводящих от исходных данных к требуемому результату, которое обладает свойствами дискретности, понятности, определённости, результативности и массовости.

Способность исполнителя действовать формально обеспечивает возможность автоматизации деятельности человека.







### Вопросы и задания



1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Дополняет ли презентация информацию, содержащуюся в тексте параграфа? Какими слайдами вы могли бы дополнить презентацию?
2. Что называют алгоритмом?
3. Подберите синонимы к слову «предписание».
4. Приведите примеры алгоритмов, изучаемых вами в школе.
5. Кто может быть исполнителем алгоритма?
6. Приведите пример формального исполнителя. Приведите пример, когда человек выступает в роли формального исполнителя.
7. От чего зависит круг решаемых задач исполнителя «компьютер»?
8. Рассмотрите в качестве исполнителя текстовый процессор, имеющийся на вашем компьютере. Охарактеризуйте круг решаемых этим исполнителем задач и его среду.
9. Что такое команда, система команд исполнителя?



10. Какие команды должны быть у робота, выполняющего функции: а) кассира в магазине; б) дворника; в) охранника?
11. Перечислите основные свойства алгоритма.
12. К чему может привести отсутствие какого-либо свойства у алгоритма? Приведите примеры.
13. В чём важность возможности формального исполнения алгоритма?
14. Последовательность чисел строится по следующему алгоритму: первые два числа последовательности принимаются равными 1; каждое следующее число последовательности принимается равным сумме двух предыдущих чисел. Запишите 10 первых членов этой последовательности. Выясните, как называется эта последовательность. 
15. Некоторый алгоритм получает из одной цепочки символов новую цепочку следующим образом. Сначала записывается исходная цепочка символов, после нее записывается исходная цепочка символов в обратном порядке, затем записывается буква, следующая в русском алфавите за той буквой, которая в исходной цепочке стояла на последнем месте. Если в исходной цепочке на последнем месте стоит буква «Я», то в качестве следующей буквы записывается буква «А». Получившаяся цепочка является результатом работы алгоритма. Например, если исходная цепочка символов была «ДОМ», то результатом работы алгоритма будет цепочка «ДОММОДН». Дана цепочка символов «КОМ». Сколько букв «О» будет в цепочке символов, которая получится, если применить алгоритм к данной цепочке, а затем ещё раз применить алгоритм к результату его работы? 
16. Найдите в сети Интернет анимацию шагов алгоритма Эратосфена. С помощью алгоритма Эратосфена найдите все простые числа, не превышающие 50.
17. Что будет результатом исполнения Черепашкой (см. пример 5) алгоритма?  
Повтори 8 [Направо 45 Вперёд 45] 
18. Запишите алгоритм для исполнителя Вычислитель (см. пример 6), содержащий не более 5 команд:  
а) получения из числа 3 числа 16;  
б) получения из числа 1 числа 25. 



- 19.** Система команд исполнителя Конструктор состоит из двух команд, которым присвоены номера:

1 — приписать 2

2 — разделить на 2

По первой из них к числу приписывается справа 2, по второй число делится на 2. Как будет преобразовано число 8, если исполнитель выполнит алгоритм 22212? Составьте алгоритм в системе команд этого исполнителя, по которому число 1 будет преобразовано в число 16 (в алгоритме должно быть не более 5 команд).



- 20.** В какой клетке должен находиться исполнитель Робот (пример 7), чтобы после выполнения алгоритма 3241 в неё же и вернуться?



## § 2.2

### Способы записи алгоритмов

**Ключевые слова:**

- словесное описание
- построчная запись
- блок-схема
- школьный алгоритмический язык

Существуют различные способы записи алгоритмов. Основными среди них являются:

- словесные;
- графические;
- на алгоритмических языках.

Теоретические исследования нашего соотечественника Андрея Андреевича Маркова (младшего) (1903–1979), выполненные в середине прошлого века, показали, что в общем случае алгоритмы должны содержать предписания двух видов:

- 1) предписания, направленные на непосредственное преобразование информации (функциональные операторы);
- 2) предписания, определяющие дальнейшее направление действий (логические операторы).

Именно эти операторы положены в основу большинства способов записи алгоритмов.



#### 2.2.1. Словесные способы записи алгоритма

**Словесное описание.** Самой простой является запись алгоритма в виде набора высказываний на обычном разговорном языке. Сло-

весное описание имеет минимум ограничений и является наименее формализованным. Однако все разговорные языки обладают неоднозначностью, поэтому могут возникнуть различные толкования текста алгоритма, заданного таким образом. Алгоритм в словесной форме может оказаться очень объёмным и трудным для восприятия.

**Пример 1.** Словесное описание алгоритма нахождения наибольшего общего делителя (НОД) пары натуральных чисел (алгоритм Евклида).

Чтобы найти НОД двух чисел, составьте таблицу из двух столбцов и назовите столбцы  $X$  и  $Y$ . Запишите первое из заданных чисел в столбец  $X$ , а второе — в столбец  $Y$ . Если данные числа не равны, замените большее из них на результат вычитания из большего числа меньшего. Повторяйте такие замены до тех пор, пока числа не окажутся равными, после чего число из столбца  $X$  считайте искомым результатом.

**Построчная запись.** Это запись на естественном языке, но с соблюдением некоторых дополнительных правил:

- каждое предписание записывается с новой строки;
- предписания (шаги) алгоритма нумеруются;
- исполнение алгоритма происходит в порядке возрастания номеров шагов, начиная с первого (если не встречается никаких специальных указаний).

Кроме слов естественного языка предписания могут содержать математические выражения и формулы.

**Пример 2.** Построчная запись алгоритма Евклида.

1. Обозначить первое из заданных чисел  $X$ , второе обозначить  $Y$ .
2. Если  $X = Y$ , то перейти к п. 8.
3. Если  $X > Y$ , то перейти к п. 4, иначе перейти к п. 6.
4. Заменить  $X$  на  $X - Y$ .
5. Перейти к п. 2.
6. Заменить  $Y$  на  $Y - X$ .
7. Перейти к п. 2.
8. Считать  $X$  искомым результатом.

Построчная запись алгоритма позволяет избежать ряда неопределённостей; её восприятие не требует дополнительных знаний.

Вместе с тем использование построчной записи требует от человека большого внимания.

### 2.2.2. Блок-схемы

Наилучшей наглядностью обладают графические способы записи алгоритмов; самый распространённый среди них — блок-схема.

Блок-схема представляет собой графический документ, дающий представление о порядке работы алгоритма. Здесь предписания изображаются с помощью различных геометрических фигур, а последовательность выполнения шагов указывается с помощью линий, соединяющих эти фигуры. Направления линий связи *слева направо* и *сверху вниз* считаются стандартными, соответствующие им линии связи можно изображать *без стрелок*. Линии связи *справа налево* и *снизу вверх* изображаются *со стрелками*.

Рассмотрим некоторые условные обозначения, применяемые в блок-схемах.

Выполнение алгоритма всегда начинается с блока начала и оканчивается при переходе на блок конца (рис. 2.2, а). Из начального блока выходит одна линия связи; в конечный блок входит одна линия связи.

Внутри блока данных (рис. 2.2, б) перечисляются величины, значения которых должны быть введены (исходные данные) или выведены (результаты) в данном месте схемы. В блок данных входит одна линия связи, и из блока исходит одна линия связи.

В блоке обработки данных (рис. 2.2, в) содержится описание тех действий, которые должны быть выполнены при переходе на этот блок (выполнение определённой операции или группы операций, приводящее к изменению значения, формы или размещения информации). В блок обработки данных входит одна линия связи, и из блока исходит одна линия связи.

Проверка условия изображается с помощью блока принятия решения, внутри которого записывается это условие (рис. 2.2, г). В блок принятия решения входит одна линия, а выходят две линии, около которых записываются результаты проверки условия.

Комментарии (рис. 2.2, д) используются для добавления пояснительных записей, делающих блок-схему более понятной.

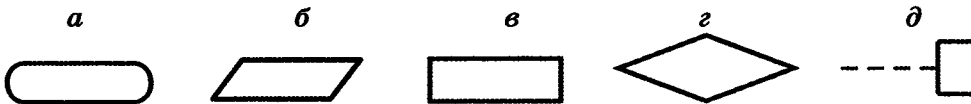


Рис. 2.2. Обозначения на блок-схемах



**Пример 3.** Запись алгоритма Евклида с помощью блок-схемы (рис. 2.3).

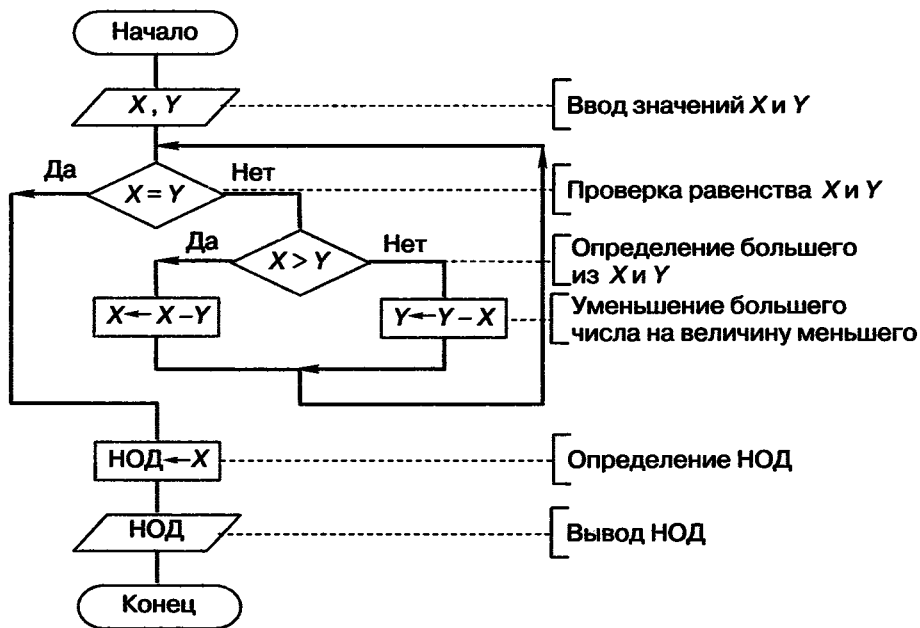


Рис. 2.3. Запись алгоритма Евклида с помощью блок-схемы

Создание детальной блок-схемы сложного алгоритма — трудоёмкая задача. Кроме того, блок-схема, не уместающаяся на одном стандартном листе, теряет своё основное преимущество — наглядность. При разработке сложных алгоритмов блок-схемы удобно использовать в качестве средства для наглядного представления решения задачи в общем виде.

### 2.2.3. Алгоритмические языки

Алгоритмические языки — формальные языки, предназначенные для записи алгоритмов. Каждый из них характеризуется:

- алфавитом — набором используемых символов;
- синтаксисом — системой правил, по которым из символов алфавита образуются правильные конструкции языка;
- семантикой — системой правил, строго определяющей смысл и способ употребления конструкций языка.

Класс алгоритмических языков очень широк. При изучении курса информатики в школах используются различные версии школьного (учебного) алгоритмического языка.

**Школьный алгоритмический язык.** Для записи алгоритмов на школьном алгоритмическом языке используется некоторое ограниченное множество слов, смысл и способ употребления которых заданы раз и навсегда. Это так называемые служебные слова: **алг** (алгоритм), **дано**, **надо**, **нач** (начало), **кон** (конец), **арг** (аргумент), **рез** (результат) и др. При записи алгоритмов в книгах служебные слова выделяются жирным шрифтом, в тетради и на доске — подчёркиванием.

В общем виде программу на школьном алгоритмическом языке можно представить так:

```
алг <название алгоритма>  
нач  
    <последовательность команд>  
кон
```

**Пример 4.** Алгоритм, позволяющий из полного сосуда ёмкостью 12 л отлить половину, пользуясь двумя пустыми сосудами ёмкостью 8 и 5 л.

```
алг переливания  
нач
```

```
    наполнить сосуд ёмкостью 8 л из сосуда ёмкостью 12 л  
    наполнить сосуд ёмкостью 5 л из сосуда ёмкостью 8 л  
    вылить всё из сосуда ёмкостью 5 л в сосуд ёмкостью 12 л  
    вылить всё из сосуда ёмкостью 8 л в сосуд ёмкостью 5 л  
    наполнить сосуд ёмкостью 8 л из сосуда ёмкостью 12 л  
    долить из сосуда ёмкостью 8 л в сосуд ёмкостью 5 л  
    вылить всё из сосуда ёмкостью 5 л в сосуд ёмкостью 12 л
```

```
кон
```

По ссылке <http://www.niisi.ru/kumir/> вы можете скачать систему КуМир (Комплект учебных Миров), в которой используется школьный алгоритмический язык, со встроенными исполнителями Робот, Чертёжник, Водолей и др. Кумир работает в операционных системах Windows и Linux.

Далее, говоря об алгоритмическом языке, мы будем иметь в виду именно школьный алгоритмический язык.



### САМОЕ ГЛАВНОЕ

Существуют различные способы записи алгоритмов: словесное описание, построчная запись, блок-схемы, школьный алгоритмический язык и др. Каждый из этих способов обладает своими достоинствами и недостатками.



### Вопросы и задания



1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Что вы можете сказать о формах представления информации в презентации и в учебнике? Какими слайдами вы могли бы дополнить презентацию?
2. Каковы основные способы записи алгоритмов?
3. Чем вызвано существование многих способов записи алгоритмов?
4. Дайте словесное описание алгоритма сложения двух обыкновенных дробей  $a/b$  и  $c/d$ .
5. Представьте в виде построчной записи алгоритм решения следующей задачи: «Имеются четыре арбуза различной массы. Как, пользуясь чашечными весами без гирь, путём не более пяти взвешиваний расположить их по возрастанию веса?».
6. Представьте с помощью блок-схемы алгоритм решения следующей задачи: «Из трёх монет одинакового достоинства одна фальшивая (более лёгкая). Как её найти с помощью одного взвешивания на чашечных весах без гирь?».
7. Запишите на алгоритмическом языке алгоритм построения окружности заданного радиуса  $r$ , проходящей через заданные точки  $A$  и  $B$ .
8. В среде КуМир запишите и выполните алгоритм переливаний (пример 4) для исполнителя Водолей.
9. Подготовьте краткую биографическую справку о Маркове А. А. (младшем).



## § 2.3

# Объекты алгоритмов

### **Ключевые слова:**

- величина
- константа
- переменная
- тип
- имя
- присваивание
- выражение
- таблица

### 2.3.1. Величины

Алгоритмы описывают последовательность действий, производимых над некоторыми объектами, определёнными условием задачи. Например, при решении задачи о начислении зарплаты сотрудникам предприятия такими объектами могут быть табельный номер сотрудника, его фамилия, имя, отчество, оклад, отработанное время и т. д.

---

В информатике отдельный информационный объект (число, символ, строка, таблица и др.) называется **величиной**.

---



Величины делятся на постоянные (константы) и переменные. **Постоянной (константой)** называется величина, значение которой указывается в тексте алгоритма и не меняется в процессе его исполнения. **Переменной** называется величина, значение которой меняется в процессе исполнения алгоритма. При исполнении алгоритма в каждый момент времени переменная обычно имеет значение, называемое текущим значением.

**Пример 1.** Величины, выражающие количество дней в неделе, ускорение свободного падения, количество дней в первой декаде месяца, являются константами. Величины, выражающие количество дней в месяце, пульс человека, количество дней в третьей декаде месяца, являются переменными.

В алгоритмах над величинами выполняются некоторые операции. Например:

- арифметические операции  $+$ ,  $-$ ,  $*$  (умножение),  $/$  (деление);
- операции отношения  $<$ ,  $>$ ,  $<=$ ,  $>=$ ,  $=$ ,  $;$ ;
- логические операции И, ИЛИ, НЕ.

Объекты, над которыми выполняются операции, называются операндами. Не всякий объект может быть операндом для выполнения любой операции. Например, текст не может быть объектом для выполнения арифметических операций; отрицательное число не может быть операндом для извлечения квадратного корня и т. д.

Множество величин, объединённых определённой совокупностью допустимых операций, называют величинами определённого типа. При составлении алгоритмов используют величины числового (целого и вещественного), символьного, литерного и логического типов.

В математике и физике оперируют числовыми величинами — натуральными, целыми, действительными числами. При составлении алгоритмов чаще всего используют числовые величины целого и вещественного<sup>1</sup> типов, которые в алгоритмическом языке обозначаются **цел** и **вещ** соответственно.

В задачах, возникающих в повседневной жизни, встречаются и нечисловые величины, значениями которых являются символы, слова, тексты и др. При составлении алгоритмов обработки текстовой информации используют величины символьного (**сим**) и литерного (**лит**) типов. Значением символьной величины является один символ: русская или латинская буква, цифра, знак препинания или другой символ. Значением литерной величины является последовательность символов. Иногда эту последовательность называют строкой или цепочкой. Литерные значения в алгоритме записывают в кавычках, например: 'алгоритм', 'литерная величина', '2011'.

Величины логического (**лог**) типа могут принимать всего два значения:

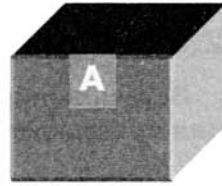
- ДА (ИСТИНА, TRUE, 1);
- НЕТ (ЛОЖЬ, FALSE, 0).

<sup>1</sup> Термин «вещественный» принято использовать наряду с термином «действительный».



Для ссылок на величины используют их имена (идентификаторы). Имя величины может состоять из одной или нескольких латинских букв, из латинских букв и цифр:  $A1$ ,  $M$ ,  $AP$ . Рекомендуется выбирать мнемонические имена, т. е. имена, отражающие суть объектов решаемой задачи, например, *SUMMA*, *PLAN*, *CENA* и т. д.

Если величину представить как ящик, содержащий которого является некоторое значение, то имя величины — это ярлык, повешенный на ящик.



### 2.3.2. Выражения

**Выражение** — языковая конструкция для вычисления значения с помощью одного или нескольких операндов.

Выражения состоят из операндов (констант, переменных, функций), объединённых знаками операций. Выражения записываются в виде линейных последовательностей символов (без подстрочных и надстрочных символов, обыкновенных дробей и т. д.); знаки операций пропускать нельзя. Порядок выполнения операций определяется скобками и приоритетом (старшинством) операций; операции одинакового приоритета выполняются слева направо.

Различают арифметические, логические и строковые выражения.

**Арифметические выражения** служат для определения числового значения. Например,  $2 * x + 3$  — арифметическое выражение, значение которого при  $x = 1$  равно пяти, а при  $x = -1$  — единице. Выражение  $\text{sqrt}(x)$  служит для обозначения операции извлечения квадратного корня из  $x$  ( $\sqrt{x}$ ).

**Логические выражения** описывают некоторые условия, которые могут удовлетворяться или не удовлетворяться. Логическое выражение может принимать одно из двух значений — ИСТИНА или ЛОЖЬ. Например, логическое выражение  $(x > 5)$  и  $(x < 10)$  определяет принадлежность точки  $x$  интервалу  $(5; 10)$ :



При  $x = 6$  значение этого выражения — ИСТИНА, а при  $x = 12$  — ЛОЖЬ.

**Строковые выражения** состоят из величин (констант, переменных) символьного и литерного типов, соответствующих функций и операций сцепления (присоединения). Операция сцепления обозначается знаком «+» и позволяет соединить в одну последовательность несколько последовательностей символов. Значениями стро-

ковых выражений являются последовательности символов. Например, если  $A = \text{'том'}$ , то значение строкового выражения  $\text{'a'+A}$  есть  $\text{'атом'}$ .

### 2.3.3. Команда присваивания

Задать конкретное значение величины можно с помощью операции присваивания, которая записывается так:

$\langle \text{имя переменной} \rangle := \langle \text{выражение} \rangle$

Знак  $\langle := \rangle$  читается: «присвоить». Например, запись  $A := B + 5$  читается так: «переменной  $A$  присвоить значение выражения  $B$  плюс 5».

Знаки присваивания  $\langle := \rangle$  и равенства  $\langle = \rangle$  — разные знаки:

- знак  $\langle = \rangle$  означает равенство двух величин, записанных по обе стороны от этого знака;
- знак  $\langle := \rangle$  предписывает выполнение операции присваивания.

Например, запись  $A := A + 1$  выражает не равенство значений  $A$  и  $A + 1$ , а указание увеличить значение переменной  $A$  на единицу.

При выполнении команды присваивания сначала вычисляется значение выражения, стоящего справа от знака  $\langle := \rangle$ , затем результат присваивается переменной, стоящей слева от знака  $\langle := \rangle$ . При этом тип выражения должен быть совместим с типом соответствующей переменной.

Свойства присваивания:

- 1) пока переменной не присвоено значение, она остаётся неопределённой;
- 2) значение, присвоенное переменной, сохраняется в ней вплоть до выполнения следующего присваивания этой переменной нового значения;
- 3) если мы присваиваем некоторой переменной очередное значение, то предыдущее её значение теряется безвозвратно.

**Пример 2.** Составим алгоритм, в результате которого переменные  $A$  и  $B$  литерного типа обменяются своими значениями.

Решение вида

$A := B$

$B := A$

неверно, так как после выполнения первой команды присваивания первоначальное значение переменной  $A$  будет безвозвратно утеряно. Вторая команда присвоит переменной  $B$  текущее значение переменной  $A$ . В результате обе переменные получат одно и то же значение.

Для поиска правильного решения воспользуемся аналогией. Если требуется перелить жидкость из сосуда 1 в сосуд 2, а из сосуда 2 — в сосуд 1, то без дополнительного сосуда 3 здесь не обойтись. Алгоритм переливаний представлен на рис. 2.4.

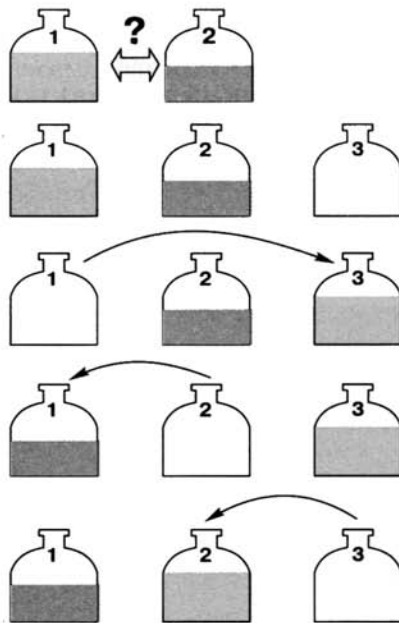


Рис. 2.4. Алгоритм переливаний жидкостей

Для решения исходной задачи введём промежуточную переменную  $M$ . Алгоритм обмена значениями переменных  $A$  и  $B$  запишем так:

**алг** обмен значениями (лит  $A$ ,  $B$ )

**арг**  $A$ ,  $B$

**рез**  $A$ ,  $B$

**нач** лит  $M$

$M := A$

$A := B$

$B := M$

**кон**

Если  $A$  и  $B$  — числовые величины, то обмен их значениями можно организовать и без промежуточной переменной, например так:

$A := A + B$

$B := A - B$

$A := A - B$

### 2.3.4. Табличные величины

В практической деятельности человек часто использует всевозможные таблицы. Это, например, список учащихся в классном журнале, табель успеваемости, таблица результатов спортивных соревнований и т. д. Чаще всего встречаются линейные и прямоугольные таблицы.

**Линейная таблица (одномерный массив)** представляет собой набор однотипных данных, записанных в одну строку или один столбец. Элементы строки (столбца) всегда нумеруются. Например, с помощью линейной таблицы могут быть представлены дни недели (рис. 2.5, а) или количество уроков, пропущенных учеником в течение 5-дневной учебной недели (рис. 2.5, б).

<i>a</i>		<i>b</i>										
1	Понедельник											
2	Вторник											
3	Среда											
4	Четверг											
5	Пятница											
6	Суббота											
7	Воскресенье											
	Васечкин	<table style="border-collapse: collapse; text-align: center;"> <tr> <td style="border: 1px solid black; padding: 2px;">1</td> <td style="border: 1px solid black; padding: 2px;">2</td> <td style="border: 1px solid black; padding: 2px;">3</td> <td style="border: 1px solid black; padding: 2px;">4</td> <td style="border: 1px solid black; padding: 2px;">5</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">6</td> <td style="border: 1px solid black; padding: 2px;">6</td> <td style="border: 1px solid black; padding: 2px;">1</td> <td style="border: 1px solid black; padding: 2px;">0</td> <td style="border: 1px solid black; padding: 2px;">0</td> </tr> </table>	1	2	3	4	5	6	6	1	0	0
1	2	3	4	5								
6	6	1	0	0								

Рис. 2.5. Примеры линейных таблиц

**Прямоугольная таблица (двумерный массив)** — это упорядоченный некоторым образом набор строк (столбцов), содержащих одинаковое количество элементов. Строки прямоугольных таблиц имеют свою нумерацию, столбцы — свою. Например, с помощью прямоугольной таблицы можно представить количество уроков, пропущенных всеми учениками 8 класса в течение 5-дневной учебной недели (рис. 2.6).

	1	2	3	4	5
1. Васечкин	6	6	1	0	0
2. Ионов	0	0	0	0	6
3. Радугина	0	0	1	0	0
...	...	...	...	...	...
19. Чабанюк	0	0	0	0	0

Рис. 2.6. Пример прямоугольной таблицы

Всей совокупности элементов табличной величины даётся одно имя. Элементы различают по их номерам, называемым индексами. Индекс записывается в квадратных скобках сразу за именем таблицы.

Если первую из рассмотренных нами таблиц (см. рис. 2.5, а) назвать *WEEK*, то  $WEEK[1] = \text{'понедельник'}$ ,  $WEEK[6] = \text{'суббота'}$ . Назовём третью из рассмотренных таблиц *LES*. Тогда  $LES[1,1] = 6$ ,  $LES[2,5] = 6$ ,  $LES[3,4] = 0$ .

Образно линейная и прямоугольная таблицы показаны на рис. 2.7.

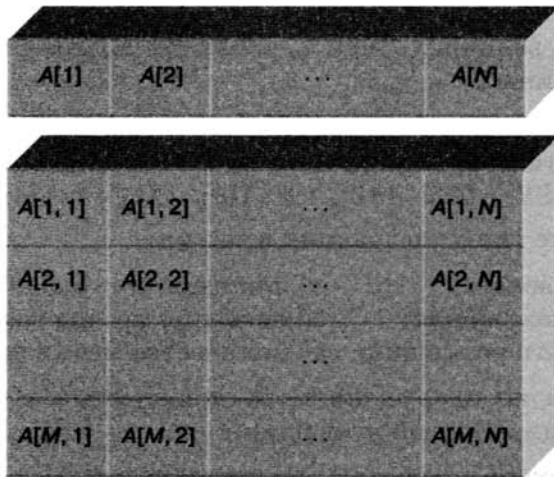


Рис. 2.7. Образное представление линейной и прямоугольной таблиц

## САМОЕ ГЛАВНОЕ

В информатике отдельный информационный объект (число, символ, строка, таблица и др.) называется величиной.

Величины делятся на постоянные (их значения указываются в тексте алгоритма и не меняются в процессе его исполнения) и переменные (их значения меняются в процессе исполнения алгоритма). При составлении алгоритмов используют величины целого, вещественного, логического, символического и литерного типов.

Для ссылок на величины используют их имена (идентификаторы). Имя величины может состоять из одной или нескольких латинских букв, из латинских букв и цифр.

Таблица (массив) — набор некоторого числа однотипных элементов, которым присвоено одно имя. Положение элемента в таблице однозначно определяется его индексами.



### Вопросы и задания



1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Используйте эти материалы при подготовке ответов на вопросы и выполнении заданий.
2. Что такое величина? Чем отличаются постоянные и переменные величины?
3. Величины каких типов используются при записи алгоритмов?
4. Укажите тип величины, если её значение равно: 2010; 14.48; 'ДА'; FALSE, -125; '142';  $1,4 \cdot 10^5$ ; .123E-2; 'пять'.
5. Определите типы следующих величин:
  - а) вес человека;
  - б) марка автомобиля;
  - в) год вашего рождения;
  - г) площадь фигуры;
  - д) название месяца года;
  - е) количество мест в самолёте.
6. Приведите по одному примеру допустимых и недопустимых значений для каждой из величин:
  - а) температура человека;
  - б) скорость автомашины;
  - в) площадь страны;
  - г) название дня недели.
7. Для чего предназначена команда присваивания? Каковы её основные свойства?
8. Какие команды присваивания составлены правильно?
  - а)  $A := B$
  - б)  $A = B$
  - в)  $A = B + 1$
  - г)  $A + 1 := A$
9. Придумайте свой алгоритм обмена значениями числовых переменных  $A$  и  $B$ .
10. Сколько промежуточных переменных потребуется для того, чтобы переменной  $A$  было присвоено значение переменной  $B$ , переменной  $B$  — значение переменной  $C$ , а переменной  $C$  — значение переменной  $A$ ? Запишите соответствующий алгоритм на алгоритмическом языке.



11. После выполнения команды присваивания  $x:=x+y$  значение переменной  $x$  равно 3, а значение переменной  $y$  равно 5. Чему были равны значения переменных  $x$  и  $y$  до выполнения указанной команды присваивания?



12. Что называют выражением? Каковы основные правила записи выражений?

13. Переведите из линейной записи в общепринятую:

а)  $a * b / c$ ;

г)  $(a + b) / c$ ;

б)  $a / b * c$ ;

д)  $a + b / c + d$ ;

в)  $a + b / c$ ;

е)  $(a + b) / (c + d)$ .

14. Запишите на алгоритмическом языке:

а)  $ax^2 + bx + c$ ;

б)  $v + \frac{at^2}{2}$ ;

в)  $\frac{1}{2}(a + b)h$ ;

г)  $\frac{1 + x_1x_2}{b^2c}$ ;

д)  $\sqrt{a^2 + b^2}$ .

15. Запишите логическое выражение, истинное при выполнении указанного условия и ложное в противном случае:



а)  $x$  принадлежит отрезку  $[0, 1]$ ;

б)  $x$  лежит вне отрезка  $[0, 1]$ ;

в) каждое из чисел  $x, y$  положительно;

г) хотя бы одно из чисел  $x, y$  положительно;

д) ни одно из чисел  $x, y$  не является положительным;

е) только одно из чисел  $x, y$  положительно.

16. Изобразите в декартовой прямоугольной системе координат область, в которой и только в которой истинно следующее логическое выражение:

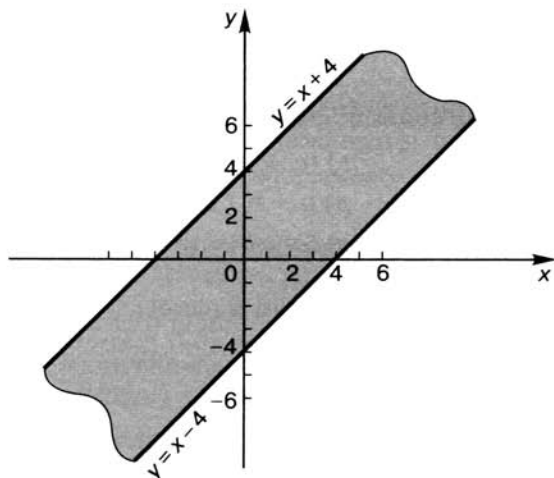


а)  $(x > -1) \text{ и } (x <= 1) \text{ и } (y > -1) \text{ и } (y <= 1)$ ;

б)  $(y >= x) \text{ и } (y >= -x) \text{ и } (y <= 1)$ .



17. Запишите логическое выражение, принимающее значение TRUE, когда точка с координатами  $(x, y)$  принадлежит закрашенной области.



18. Запишите команду присваивания, в результате выполнения которой логическая переменная  $t$  получает значение TRUE, если выполняется указанное условие, и значение FALSE в противном случае:
- $x$  — положительное число;
  - хотя бы одно из чисел  $x, y, z$  равно нулю;
  - числа  $x, y, z$  равны между собой.
19. Какие из приведённых ниже величин целесообразно представлять с помощью таблиц?

*Величины:* список учеников класса, рост учеников класса, средний рост учеников класса, оценка ученика по физике, средний балл ученика по физике, оценки учеников за контрольную работу по информатике, длины сторон треугольника, длины сторон нескольких треугольников, названия дней недели, имя человека, площадь фигуры, периметры нескольких прямоугольников, самая холодная температура воздуха в январе, количество девочек в классе, самая дождливая декада июня.



## Основные алгоритмические конструкции

### Ключевые слова:

- следование
- ветвление
- повторение
- линейные алгоритмы
- разветвляющиеся алгоритмы
- циклические алгоритмы

Человеку в жизни приходится решать множество различных задач. Решение каждой из них описывается своим алгоритмом, и разнообразие этих алгоритмов очень велико. Вместе с тем для записи любого алгоритма достаточно трёх основных алгоритмических конструкций (структур): следования, ветвления, повторения. Это положение выдвинул и доказал Э. Дейкстра в 70-х гг. прошлого века.



Эдсгер Вибе Дейкстра (1930–2002) — выдающийся нидерландский учёный, идеи которого оказали огромное влияние на развитие компьютерной индустрии.

### 2.4.1. Следование

**Следование** — алгоритмическая конструкция, отображающая естественный, последовательный порядок действий. Алгоритмы, в которых используется только структура «следование», называются **линейными алгоритмами**.

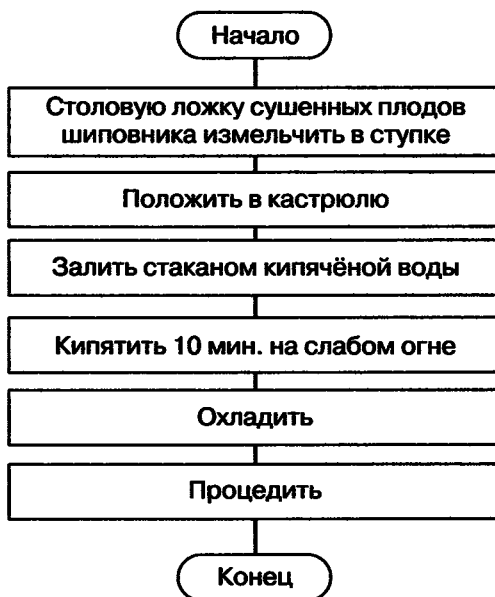


Графическое представление алгоритмической конструкции «следование» приведено на рис. 2.8.



Рис. 2.8. Алгоритмическая конструкция «следование»

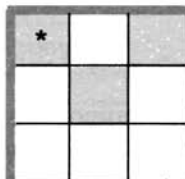
**Пример 1.** Линейный алгоритм приготовления отвара шиповника.



Обратите внимание, что многие из предписаний этого алгоритма могут потребовать детализации — представления в виде некоторой совокупности более мелких предписаний.

**Пример 2.** У исполнителя Робот есть четыре команды перемещения (вверх, вниз, влево и вправо), при выполнении каждой из них Робот перемещается на одну клетку в соответствующем направлении. По команде закрасить Робот закрашивает клетку, в которой он находится. Запишем линейный алгоритм, исполняя который

Робот нарисует на клетчатом поле следующий узор и вернётся в исходное положение, обозначенное звёздочкой:



```

алг узор
нач
    закрасить
    вправо
    вправо
    закрасить
    вниз
    влево
    закрасить
    вверх
    влево
кон
    
```

**Пример 3.** Дан фрагмент линейного алгоритма:

```

x:=2
y:=x*x
y:=y*y
x:=y*x
s:=x+y
    
```

Выясним, какое значение получит переменная  $s$  после выполнения этого фрагмента алгоритма. Для этого составим таблицу значений переменных, задействованных в алгоритме:

Шаг алгоритма	Переменные		
	$x$	$y$	$s$
1	2	–	–
2		4	–
3		16	–
4	32		–
5			48

Составленная нами таблица значений переменных моделирует работу исполнителя этого алгоритма.





**Пример 4.** Некоторый исполнитель может выполнять над целыми числами кроме операций сложения, вычитания, умножения и деления ещё две операции: с помощью операции `div` вычисляется целое частное, с помощью операции `mod` — остаток.

Например:  $5 \text{ div } 2 = 2$ ;  $5 \text{ mod } 2 = 1$ ;  $2 \text{ div } 5 = 0$ ;  $2 \text{ mod } 5 = 2$ .

Покажем, как с помощью этих операций можно реализовать алгоритм работы кассира, выдающего покупателю сдачу ( $s$ ) наименьшим количеством банкнот по 500 ( $k500$ ), 100 ( $k100$ ), 50 ( $k50$ ) и 10 ( $k10$ ) рублей.

```
k500:=s div 500
s:=s mod 500
k100:=s div 100
s:=s mod 100
k50:=s div 50
s:=s mod 50
k10:=s div 10
```



Исполните алгоритм для  $s = 745$  и  $s = 1864$ . Составьте соответствующие таблицы значений переменных.



Ознакомьтесь с имеющимся в Единой коллекции цифровых образовательных ресурсов модулем для коллективной работы «Линейные алгоритмы» (217039). Совместно с друзьями постарайтесь составить алгоритмы для имеющихся в модуле задач. Пройдите тестирование.

### 2.4.2. Ветвление



**Ветвление** — алгоритмическая конструкция, в которой в зависимости от результата проверки условия («да» или «нет») предусмотрен выбор одной из двух последовательностей действий (ветвей). Алгоритмы, в основе которых лежит структура «ветвление», называют **разветвляющимися**.

Блок-схема ветвления представлена на рис. 2.9. Каждая ветвь может быть любой степени сложности (рис. 2.9, а), а может вообще не содержать предписаний (рис. 2.9, б).

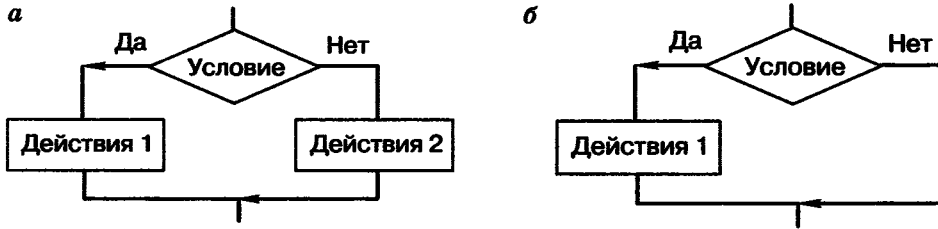


Рис. 2.9. Структура «ветвление»: а — полная форма ветвления; б — неполная форма ветвления

На алгоритмическом языке команда ветвления записывается так:

Полная форма ветвления:

```
если <условие>
    то <действия 1>
    иначе <действия 2>
все
```

Неполная форма ветвления:

```
если <условие>
    то <действия 1>
все
```

### Пример 5

```
алг правописание приставок НЕ, НИ
нач
    если приставка под ударением
        то писать НЕ
        иначе писать НИ
    все
кон
```

### Пример 6

```
алг сборы на прогулку
нач
    если на улице дождь
        то взять зонтик
    все
кон
```



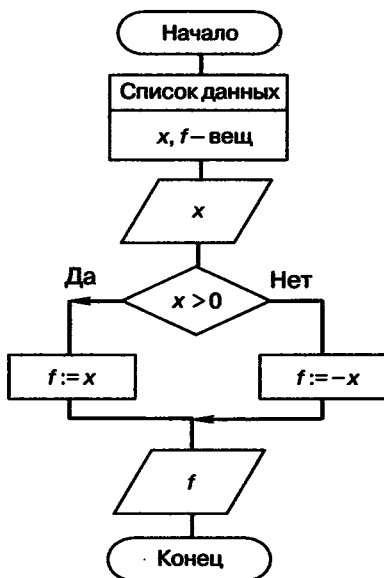
Для записи условий, в зависимости от результатов проверки которых выбирается та или иная последовательность действий, используются операции сравнения:

- $A < B$  —  $A$  меньше  $B$ ;
- $A \leq B$  —  $A$  меньше или равно  $B$ ;
- $A = B$  —  $A$  равно  $B$ ;
- $A > B$  —  $A$  больше  $B$ ;
- $A \geq B$  —  $A$  больше или равно  $B$ ;
- $A \neq B$  —  $A$  не равно  $B$ .

Здесь буквы  $A$  и  $B$  можно заменять на любые переменные, числа и арифметические выражения. Приведённые операции сравнения допускаются и для символьных переменных.



**Пример 7.** Алгоритм вычисления функции  $f(x) = |x|$  для произвольного числа  $x$ .

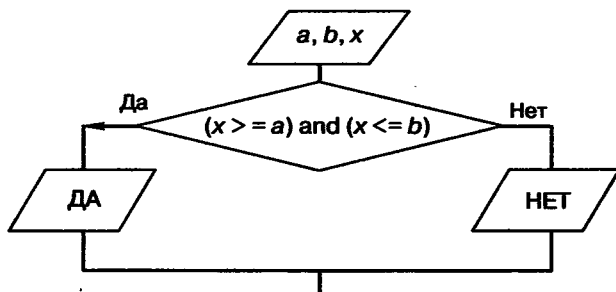


Обратите внимание на второй блок этой блок-схемы. В нём представлены имена и типы величин (данных), обрабатываемых в алгоритме.

Условия, состоящие из одной операции сравнения, называются простыми. В качестве условий при организации ветвлений можно использовать и составные условия. Составные условия получаются из простых с помощью логических связок **and** (**и**), **or** (**или**), **not** (**не**): **and** означает одновременное выполнение всех условий, **or** — выполнение хотя бы одного условия, а **not** означает отрицание условия, записанного за словом **not**.

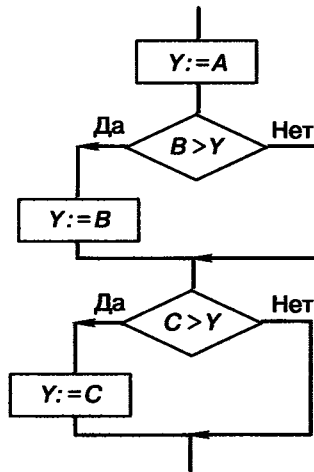


**Пример 8.** Алгоритм определения принадлежности точки  $x$  отрезку  $[a, b]$ . Если точка  $x$  принадлежит данному отрезку, то выводится ответ ДА, в противном случае — НЕТ.



Существует достаточно много ситуаций, в которых приходится выбирать не из двух, а из трёх и более вариантов. Есть разные способы построения соответствующих алгоритмов. Один из них — составить комбинацию из нескольких ветвлений.

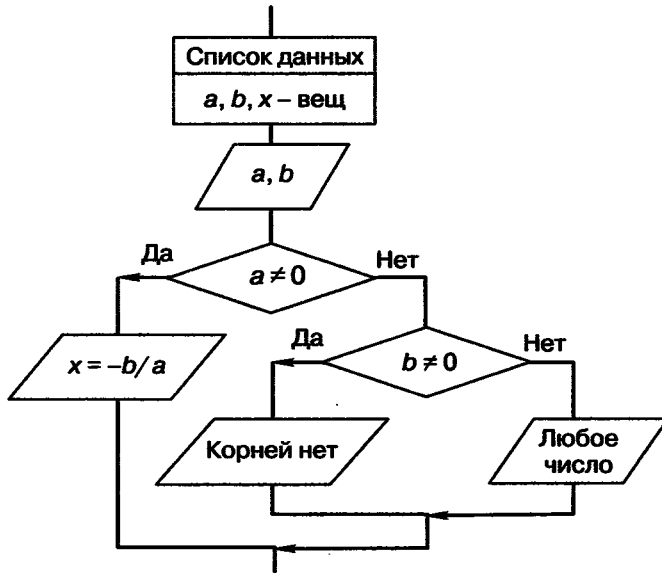
**Пример 9.** Алгоритм, в котором переменной  $Y$  присваивается значение большей из трёх величин  $A$ ,  $B$  и  $C$ .



Пусть  $A = 10$ ,  $B = 30$  и  $C = 20$ . Тогда процесс выполнения алгоритма можно представить в следующей таблице:

Шаг алгоритма	Константы			Переменная $Y$	Условие
	$A$	$B$	$C$		
1	10	30	20	10	
2					$30 > 10$ (Да)
3				30	
4					$20 > 30$ (Нет)

**Пример 10.** Алгоритм решения линейного уравнения  $ax + b = 0$ .



**Пример 11.** Исполнитель Робот может выполнять ту или иную последовательность действий в зависимости от выполнения следующих простых условий:

справа свободно  
слева свободно  
сверху свободно  
снизу свободно  
клетка чистая

справа стена  
слева стена  
сверху стена  
снизу стена  
клетка закрашена

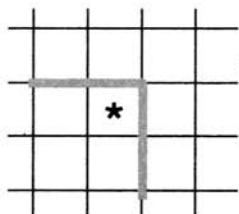
Также Робот может действовать в зависимости от выполнения составных условий.

**?** Подумайте, в какую клетку переместится Робот из клетки, обозначенной звёздочкой, при выполнении следующего фрагмента алгоритма.

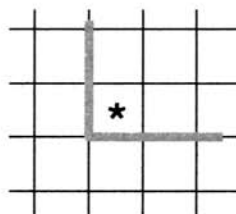
**если** справа свободно **или** снизу свободно  
**то** закрасить  
**все**  
**если** справа стена  
**то** влево  
**все**  
**если** слева стена  
**то** вправо  
**все**



а



б



Ознакомьтесь с размещённым в Единой коллекции цифровых образовательных ресурсов модулем для коллективной работы «Алгоритмы с ветвящейся структурой» (217044). Совместно с друзьями постарайтесь составить алгоритмы для имеющихся в модуле задач. Пройдите тестирование.



### 2.4.3. Повторение

**Повторение** — алгоритмическая конструкция, представляющая собой последовательность действий, выполняемых многократно. Алгоритмы, содержащие конструкцию повторения, называют **циклическими** или **циклами**. Последовательность действий, многократно повторяющаяся в процессе выполнения цикла, называется **телом цикла**.



В зависимости от способа организации повторений различают три типа циклов:

- 1) цикл с заданным условием продолжения работы;
- 2) цикл с заданным условием окончания работы;
- 3) цикл с заданным числом повторений.

**Цикл с заданным условием продолжения работы (цикл-ПОКА, цикл с предусловием).** Логика работы этой конструкции описывается схемой, показанной на рис. 2.10.

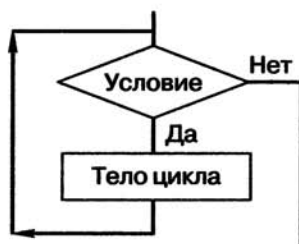


Рис. 2.10. Цикл с предусловием

На алгоритмическом языке эта конструкция записывается так:

```
нц пока <условие>  
  <тело цикла (последовательность действий)>  
кц
```

Выполняется цикл-ПОКА следующим образом: 1) проверяется условие (вычисляется значение логического выражения); 2) если условие удовлетворяется (Да), то выполняется тело цикла и снова осуществляется переход к проверке условия; если же условие не удовлетворяется, то выполнение цикла заканчивается. Возможны случаи, когда тело цикла не будет выполнено ни разу.

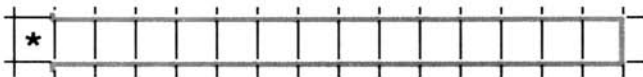


**Пример 12.** Алгоритм, по которому из всех имеющихся кирпичей отбираются целые кирпичи и складываются в машину.

```
алг отбор  
нач  
  нц пока есть кирпичи  
    взять один кирпич  
    если кирпич целый  
      то положить кирпич в машину  
    иначе отложить кирпич в сторону  
кц  
кон
```



**Пример 13.** Правее Робота (клетка со звездочкой) расположен коридор неизвестной длины. Необходимо, чтобы Робот закрасил все клетки этого коридора.



Пока будет выполняться условие справа свободно, Роботу следует выполнять команды:

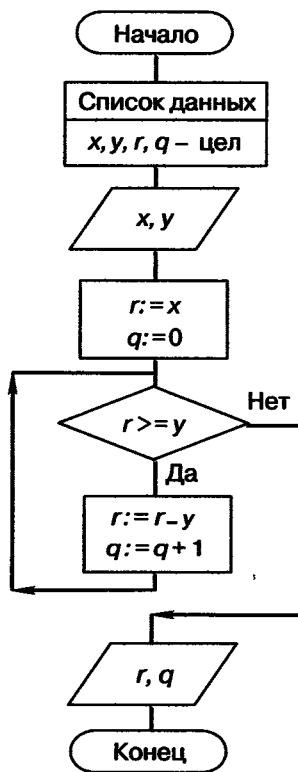
```
вправо  
закрась
```

Соответствующий алгоритм для Робота будет иметь вид:

```
нц пока справа свободно  
  вправо  
  закрась  
кц
```

**Пример 14.** Требуется, не пользуясь операцией деления, получить частное  $q$  и остаток  $r$  от деления натурального числа  $x$  на натуральное число  $y$ .

Представим операцию деления как последовательные вычитания делителя из делимого. Причём вычитать будем до тех пор, пока результат вычитания не станет меньше вычитаемого (делителя). В этом случае количество вычитаний будет равно частному от деления  $q$ , а последняя разность — остатку от деления  $r$ .



Исполним этот алгоритм для  $x = 23$  и  $y = 5$ .

Шаг алгоритма	Операция	Переменная				Условие $r \geq y$
		x	y	r	q	
1	Ввод $x$	23	-	-	-	
2	Ввод $y$		5	-	-	
3	$r := x$			23	-	
4	$q := 0$				0	
5	$r \geq y$					23 $\geq$ 5 (Да)
6	$r := r - y$			18		
7	$q := q + 1$				1	
8	$r \geq y$					18 $\geq$ 5 (Да)
9	$r := r - y$			13		
10	$q := q + 1$				2	

Шаг алгоритма	Операция	Переменная				Условие $r \geq y$
		x	y	r	q	
11	$r \geq y$					$13 \geq 5$ (Да)
12	$r := r - y$			8		
13	$q := q + 1$				3	
14	$r \geq y$					$8 \geq 5$ (Да)
15	$r := r - y$			3		
16	$q := q + 1$				4	
17	$r \geq y$					$3 \geq 5$ (Нет)
18	Вывод $r$			3		
19	Вывод $q$				4	



Ознакомьтесь с размещённым в Единой коллекции цифровых образовательных ресурсов модулем для коллективной работы «Циклические алгоритмы с предусловием» (217033). Совместно с друзьями постарайтесь составить алгоритмы для имеющихся в модуле задач. Пройдите тестирование.

Цикл с заданным условием окончания работы (цикл-ДО, цикл с постусловием). Логика работы этой конструкции описывается схемой, показанной на рис. 2.11.

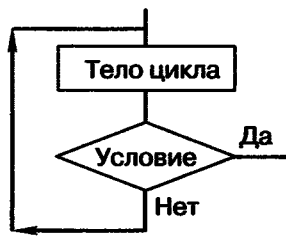


Рис. 2.11. Цикл с постусловием

На алгоритмическом языке эта конструкция записывается так:

**нц**

<тело\_цикла (последовательность действий)>

**кц при** <условие>

Выполняется цикл-ДО следующим образом: 1) выполняется тело цикла; 2) проверяется условие (вычисляется значение логического выражения); если условие не удовлетворяется («Нет»), то снова выполняется тело цикла и осуществляется переход к проверке условия; если же условие удовлетворяется, то выполнение цикла заканчивается. В любом случае тело цикла будет выполнено хотя бы один раз.

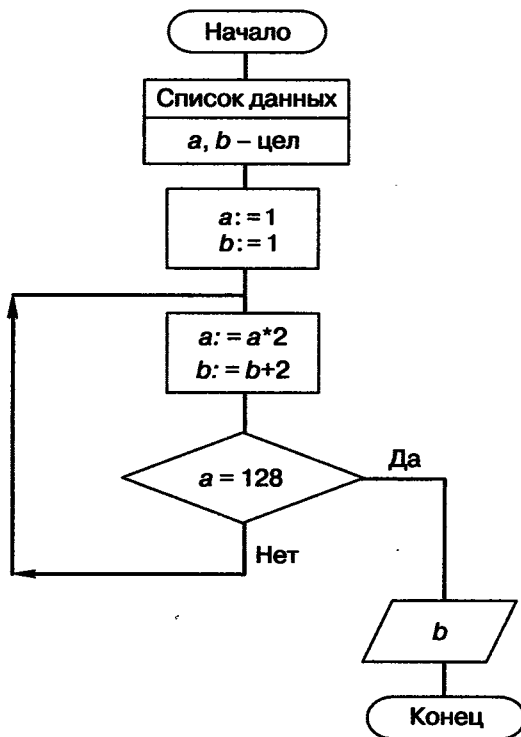
**Пример 15.** Алгоритм по выучиванию наизусть четверостишия.

```

алг четверостишие
нач
  нц
    прочитать четверостишие по книге 1 раз
    рассказать четверостишие
  кц при не сделал ошибку
кон
    
```



**Пример 16.** Вычислим значение переменной  $b$  согласно следующему алгоритму:



## Глава 2. Основы алгоритмизации

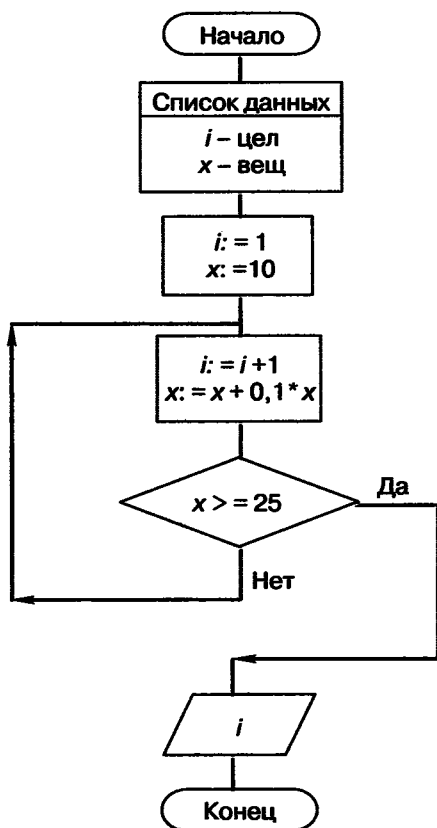
Составим таблицу значений переменных, задействованных в алгоритме:

Шаг алгоритма	Операция	Переменные		Условие $a = 128$
		$a$	$b$	
1	$a := 1$	1	–	
2	$b := 1$		1	
3	$a := a * 2$	2		
4	$b := b + a$		3	
5	$a = 128$			$2 = 128$ (Нет)
6	$a := a * 2$	4		
7	$b := b + a$		7	
8	$a = 128$			$4 = 128$ (Нет)
9	$a := a * 2$	8		
10	$b := b + a$		15	
11	$a = 128$			$8 = 128$ (Нет)
12	$a := a * 2$	16		
13	$b := b + a$		31	
14	$a = 128$			$16 = 128$ (Нет)
15	$a := a * 2$	32		
16	$b := b + a$		63	
17	$a = 128$			$32 = 128$ (Нет)
18	$a := a * 2$	64		
19	$b := b + a$		127	
20	$a = 128$			$64 = 128$ (Нет)
21	$a := a * 2$	128		
22	$b := b + a$		255	
23	$a = 128$			$128 = 128$ (Да)
24	Вывод $b$		255	

Ответ:  $b = 255$ .

**Пример 17.** Спортсмен приступает к тренировкам по следующему графику: в первый день он должен пробежать 10 км; каждый следующий день следует увеличивать дистанцию на 10% от нормы предыдущего дня. Как только дневная норма достигнет или превысит 25 км, необходимо прекратить её увеличение и далее пробегать ежедневно ровно 25 км. Начиная с какого дня спортсмен будет пробегать 25 км?

Пусть  $x$  — количество километров, которое спортсмен пробежит в некоторый  $i$ -й день. Тогда в следующий  $(i + 1)$ -й день он пробежит  $x + 0,1x$  километров ( $0,1x$  — это 10% от  $x$ ).



Ознакомьтесь с размещённым в Единой коллекции цифровых образовательных ресурсов модулем для коллективной работы «Циклические алгоритмы с постусловием» (217037). Совместно с друзьями постарайтесь составить алгоритмы для имеющихся в модуле задач. Пройдите тестирование.

**Цикл с заданным числом повторений (цикл-ДЛЯ, цикл с параметром).** Логика работы этой конструкции описывается схемой, показанной на рис. 2.12.

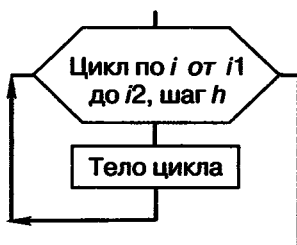


Рис. 2.12. Цикл с параметром

На алгоритмическом языке эта конструкция записывается так:

```
нц для i от i1 до i2 шаг R
  <тело цикла (последовательность действий)>
кц
```

В цикле-ДЛЯ всегда есть параметр цикла — величина целого типа, изменяющаяся в ходе выполнения цикла от своего начального значения  $i_1$  до конечного значения  $i_2$  с шагом  $R$ .

Выполняется цикл-ДЛЯ следующим образом: 1) параметру цикла присваивается начальное значение; 2) параметр цикла сравнивается с конечным значением; если параметр цикла не превышает конечное значение, то выполняется тело цикла, увеличивается значение параметра цикла на шаг и снова осуществляется проверка параметра цикла; если же параметр цикла превышает конечное значение, то выполнение цикла заканчивается.

Если величина шага в цикле с параметром равна единице, то шаг не указывают. Мы ограничимся рассмотрением именно таких циклов.

В отличие от двух предыдущих конструкций (цикл-ПОКА, цикл-ДО) цикл-ДЛЯ имеет строго фиксированное число повторений, что позволяет избежать заикливания, т. е. ситуации, когда тело цикла выполняется бесконечно.



**Пример 18.** Алгоритм переправы через реку воинского отряда из пяти человек. Солдаты могут воспользоваться помощью двух мальчиков — хозяев небольшой лодки, в которой может переправиться или один солдат, или два мальчика.



**алг** переправа

**нач**

**нц** для  $i$  от 1 до 5

два мальчика переправляются на противоположный берег  
 один мальчик высаживается на берег, другой плывёт обратно  
 солдат переправляется через реку  
 мальчик возвращается на исходную позицию

**кц**

**кон**

**Пример 19.** Составим алгоритм вычисления степени с натуральным показателем  $n$  для любого вещественного числа  $a$ .



По определению:

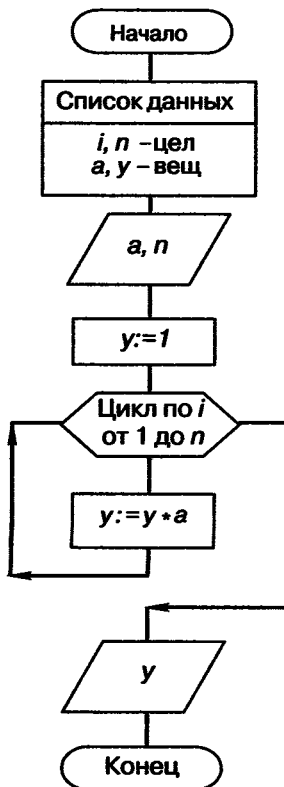
$$a^1 = a, a^n = \underbrace{a \cdot a \cdot \dots \cdot a}_n, \quad a \in R, n \in N, n \geq 2.$$

$n$  сомножителей

При составлении алгоритма воспользуемся единой формулой, в которой число умножений равно показателю степени:

$$a^n = 1 \cdot \underbrace{a \cdot a \cdot \dots \cdot a}_n$$

$n$  умножений



Исполним этот алгоритм для  $a = 4$  и  $n = 3$ .

Шаг алгоритма	Операция	Переменные				Условие
		$a$	$n$	$y$	$i$	$i \leq n$
1	Ввод $a, n$	4	3	–	–	
2	$y := 1$			1	–	
3	$i := 1$				1	
4	$i \leq n$					1 ≤ 3 (Да)
5	$y := y \cdot a$			4		
6	$i := i + 1$				2	
7	$i \leq n$					2 ≤ 3 (Да)
8	$y := y \cdot a$			16		
9	$i := i + 1$				3	
10	$i \leq n$					3 ≤ 3 (Да)
11	$y := y \cdot a$			64		
12	$i := i + 1$				4	
13	$i \leq n$					4 ≤ 3 (Нет)
14	Вывод $y$			64		



**Пример 20.** Для исполнителя Робот цикл с известным числом повторений реализуется с помощью следующей конструкции:

```

нц <число повторений> раз
    <тело цикла>
кц
    
```

Так, если правее Робота не встретится препятствий, то, выполнив приведённый ниже алгоритм, он переместится на пять клеток вправо и закрасит эти клетки:

```

алг
нач
    нц 5 раз
        вправо; закрасить
    кц
кон
    
```

Ознакомьтесь с размещённым в Единой коллекции цифровых образовательных ресурсов модулем для коллективной работы «Циклические алгоритмы с параметром» (217024). Совместно с друзьями постарайтесь составить алгоритмы для имеющихся в модуле задач. Пройдите тестирование.

## САМОЕ ГЛАВНОЕ

Для записи любого алгоритма достаточно трёх основных алгоритмических конструкций (структур): следования, ветвления, повторения.

**Следование** — алгоритмическая конструкция, отображающая естественный, последовательный порядок действий. Алгоритмы, в которых используется только структура «следование», называются линейными.

**Ветвление** — алгоритмическая конструкция, в которой в зависимости от результата проверки условия («да» или «нет») предусмотрен выбор одной из двух последовательностей действий (ветвей). Алгоритмы, в основе которых лежит структура «ветвление», называют разветвляющимися.

**Повторение** — алгоритмическая конструкция, представляющая собой последовательность действий, выполняемых многократно. Алгоритмы, содержащие конструкцию «повторение», называют циклическими или циклами. Последовательность действий, многократно повторяющаяся в процессе выполнения цикла, называется телом цикла. В зависимости от способа организации повторений различают три типа циклов:

- 1) цикл с заданным условием продолжения работы;
- 2) цикл с заданным условием окончания работы;
- 3) цикл с заданным числом повторений.

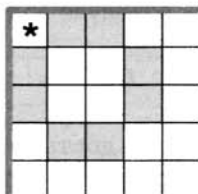
## Вопросы и задания

1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Используйте эти материалы при подготовке ответов на вопросы и выполнении заданий.
2. Какие алгоритмы называются линейными?
3. Приведите пример линейного алгоритма:
  - а) из повседневной жизни;
  - б) из литературного произведения;
  - в) из любой предметной области, изучаемой в школе.





4. Запишите линейный алгоритм, исполняя который Робот нарисует на клетчатом поле следующий узор и вернётся в исходное положение:



5. По алгоритму восстановите формулу.

```

a1:=1/x
a2:=a1/x
a3:=a2/x
a4:=a3/x
y:=a1+a2
y:=y+a3
y:=y+a4
    
```



6. Какое значение получит переменная  $y$  после выполнения алгоритма?

```

x:=1
y:=2*x
y:=y+3
y:=y*x
y:=y+4
y:=y*x
y:=y+5
    
```

Восстановите формулу вычисления  $y$  для произвольного значения  $x$ .

7. Для заданного количества суток ( $t$  в  $fh$ ) требуется определить количество часов ( $h$ ), минут ( $m$ ) и секунд ( $s$ ).
8. Известно, что 1 миля = 7 вёрст, 1 верста = 500 сажень, 1 сажень = 3 аршина, 1 аршин = 28 дюймов, 1 дюйм = 25,4 мм. Пользуясь этой информацией, составьте линейный алгоритм перевода расстояния  $X$  миль в километры.
9. Исходное данное — целое трёхзначное число  $x$ . Выполните для  $x = 125$  следующий алгоритм.

```

a:=x div 100
b:=x mod 100 div 10
c:=x mod 10
s:=a+b+c
    
```

Какой смысл имеет результат  $s$  этого алгоритма?

10. Определите значение целочисленных переменных  $x$  и  $y$  после выполнения алгоритма.

```
x:=336
y:=8
x:=x div y
y:=x mod y
```

11. Какие алгоритмы называют разветвляющимися?

12. Приведите пример разветвляющегося алгоритма:

- а) из повседневной жизни;
- б) из литературного произведения;
- в) из любой предметной области, изучаемой в школе.



13. Дополните алгоритм из примера 9 так, чтобы с его помощью можно было найти наибольшую из четырёх величин  $A$ ,  $B$ ,  $C$  и  $D$ .

14. Составьте алгоритм, с помощью которого можно определить, существует ли треугольник с длинами сторон  $a$ ,  $b$ ,  $c$ .



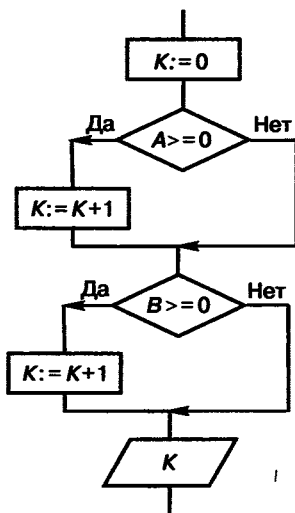
15. Составьте алгоритм, с помощью которого можно определить, является ли треугольник с заданными длинами сторон  $a$ ,  $b$ ,  $c$  равносторонним.



16. Составьте алгоритм возведения чётного числа в квадрат, а нечётного — в куб.



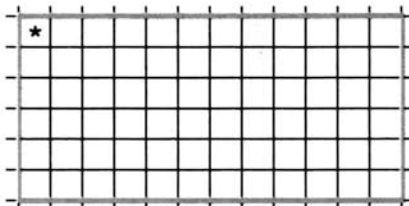
17. Какая задача решается с помощью следующего алгоритма?






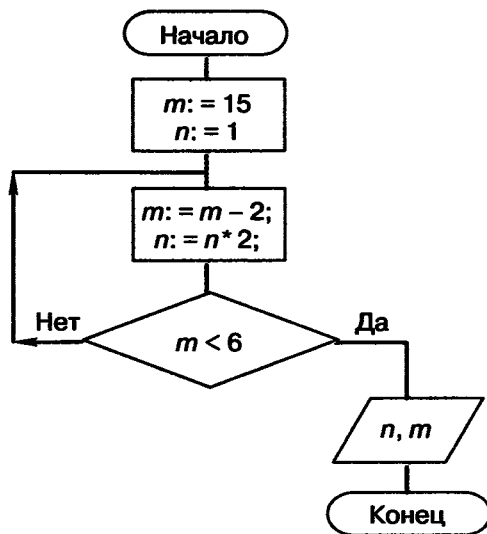
18. Составьте блок-схему алгоритма определения количества чётных чисел среди заданных целых чисел  $A$ ,  $B$  и  $C$ .
19. Составьте блок-схему алгоритма определения принадлежности точки  $X$  отрезку  $[A, B]$  (пример 8) с использованием комбинации из двух ветвлений.
20. Составьте блок-схему алгоритма правописания приставок, оканчивающихся на букву «з».
21. Известно, что 31 января 2011 года было понедельником. Какие значения должны быть присвоены литерной переменной  $y$  в алгоритме, определяющем день недели для произвольного числа (*chislo*) января 2011 года?

```
chislo:= chislo mod 7
если chislo=3 то y:='...'
если chislo=4 то y:='...'
если chislo=5 то y:='...'
если chislo=6 то y:='...'
если chislo=0 то y:='...'
если chislo=1 то y:='...'
если chislo=2 то y:='...'
```

22. Даны две точки на плоскости. Определите, какая из них находится ближе к началу координат.
23. Определите, есть ли среди цифр заданного целого трёхзначного числа одинаковые.
24. Приведите пример циклического алгоритма:
  - из повседневной жизни;
  - из литературного произведения;
  - из любой предметной области, изучаемой в школе.
25. Напишите алгоритм, под управлением которого Робот обойдёт прямоугольную область, обнесённую стеной, по периметру и закрасит угловые клетки. Размеры области неизвестны.



26. Запас рыбы в пруду оценён в  $A$  тонн. Ежегодный прирост рыбы составляет 15%. Ежегодный план отлова —  $B$  тонн. Наименьший запас рыбы составляет  $C$  тонн. (Запас ниже  $C$  тонн уже не восстанавливается.) Составьте блок-схему алгоритма для подсчёта количества лет, в течение которых можно выдерживать заданный план.
27. Дана последовательность 5, 9, 13, 17, ... . Составьте блок-схему алгоритма для определения числа слагаемых, сумма которых равна 324. 
28. Составьте алгоритм для определения количества цифр в записи произвольного натурального числа. 
29. Сумма 10 000 рублей положена в сберегательный банк, при этом прирост составляет 5% годовых. Составьте алгоритм, определяющий, через какой промежуток времени первоначальная сумма увеличится в два раза.
30. Одноклеточная амёба каждые три часа делится на 2 клетки. Составьте алгоритм вычисления времени, через которое будет  $X$  амёб.
31. Определите значения переменных  $n$  и  $m$  после выполнения алгоритма. 



32. Составьте алгоритм нахождения произведения  $z$  двух натуральных чисел  $x$  и  $y$  без использования операции умножения.

33. Население города  $N$  увеличивается на 5% ежегодно. В текущем году оно составляет 40 000 человек. Составьте блок-схему алгоритма вычисления предполагаемой численности населения города через 3 года. Составьте таблицу значений переменных, задействованных в алгоритме.
34. Каждая бактерия делится на две в течение 1 минуты. В начальный момент имеется одна бактерия. Составьте блок-схему алгоритма вычисления количества бактерий через 10 минут. Исполните алгоритм, фиксируя каждый его шаг в таблице значений переменных.



## Тестовые задания для самоконтроля



1. Алгоритмом можно считать:
  - а) описание процесса решения квадратного уравнения
  - б) расписание уроков в школе
  - в) технический паспорт автомобиля
  - г) список класса в журнале
2. Как называется свойство алгоритма, означающее, что данный алгоритм применим к решению целого класса задач?
  - а) понятность
  - б) определённости
  - в) результативность
  - г) массовость
3. Как называется свойство алгоритма, означающее, что он всегда приводит к результату через конечное, возможно, очень большое, число шагов?
  - а) дискретность
  - б) понятность
  - в) результативность
  - г) массовость
4. Как называется свойство алгоритма, означающее, что он задан с помощью таких предписаний, которые исполнитель может воспринимать и по которым может выполнять требуемые действия?
  - а) дискретность
  - б) понятность
  - в) определённости
  - г) массовость

5. Как называется свойство алгоритма, означающее, что путь решения задачи разделён на отдельные шаги?
- а) дискретность
  - б) определённость
  - в) результативность
  - г) массовость
6. Как называется свойство алгоритма, означающее, что путь решения задачи определён вполне однозначно, на любом шаге не допускаются никакие двусмысленности и недомолвки?
- а) дискретность
  - б) понятность
  - в) определённость
  - г) результативность
7. Исполнителю Черепашка был дан для исполнения следующий алгоритм:
- Повтори 10 [Вперед 10 Направо 72]
- Какая фигура появится на экране?
- а) незамкнутая ломаная линия
  - б) правильный десятиугольник
  - в) фигура, внутренние углы которой равны  $72^\circ$
  - г) правильный пятиугольник
8. Исполнитель Робот передвигается по клетчатому полю, выполняя команды, которым присвоены номера: 1 — на клетку вверх, 2 — на клетку вниз, 3 — на клетку вправо, 4 — на клетку влево. Между соседними клетками поля могут стоять стены. Если при выполнении очередного шага Робот сталкивается со стеной, то он разрушается. В результате выполнения программы 3242332411 Робот успешно прошёл из точки *A* в точку *B*. Какую программу необходимо выполнить, чтобы вернуться из точки *B* в точку *A* по кратчайшему пути и не подвергнуться риску разрушения?
- а) 41
  - б) 4131441322
  - в) 2231441314
  - г) 241314
  - д) 14

9. Система команд исполнителя Вычислитель состоит из двух команд, которым присвоены номера:

1 — вычти 2

2 — умножь на 3

Первая из них уменьшает число на 2, вторая увеличивает число в 3 раза. При записи алгоритмов для краткости указываются лишь номера команд. Запишите алгоритм, содержащий не более пяти команд, с помощью которого из числа 11 будет получено число 13.

10. Некоторый алгоритм строит цепочки символов следующим образом:



- первая цепочка состоит из одного символа — цифры 1;
- в начало каждой из последующих цепочек записывается число — номер строки по порядку, далее дважды подряд записывается предыдущая строка.

Вот первые 3 строки, созданные по этому правилу:

(1) 1

(2) 211

(3) 3211211

Сколько символов будет в седьмой цепочке, созданной по этому алгоритму?

11. Наибольшей наглядностью обладает следующая форма записи алгоритмов:

- а) словесная
- б) рекурсивная
- в) графическая
- г) построчная

12. Величины, значения которых меняются в процессе исполнения алгоритма, называются:

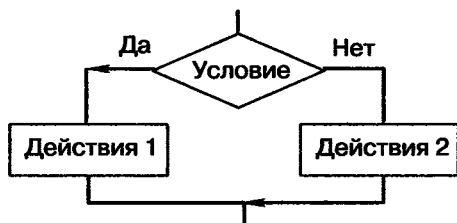
- а) постоянными
- б) константами
- в) переменными
- г) табличными

13. величиной целого типа является:
- а) количество мест в зрительном зале
  - б) рост человека
  - в) марка автомобиля
  - г) площадь государства
14. Какое логическое выражение истинно, если  $x \in [-10, 10]$ ?
- а)  $(x > 10)$  И  $(x < -10)$
  - б)  $(x > 10)$  ИЛИ  $(x < -10)$
  - в)  $(x < 10)$  ИЛИ  $(x \geq -10)$
  - г)  $(x \geq -10)$  И  $(x \leq 10)$
15. Укажите правильный вариант записи условия « $x$  — двузначное число»:
- а)  $x \text{ div } 10 \leq 9$
  - б)  $(x \geq 10)$  И  $(x < 100)$
  - в)  $x \text{ div } 100 = 0$
  - г)  $x \text{ mod } 100 = 99$
16. Какая команда присваивания должна следовать за командами  $A := A + B$  и  $B := A - B$ , чтобы последовательное выполнение всех трёх команд вело к обмену значениями переменных  $A$  и  $B$ ?
- а)  $A := A + B$
  - б)  $A := A - B$
  - в)  $B := A + B$
  - г)  $B := B - A$
17. К какому виду алгоритмов можно отнести алгоритм, схема которого представлена ниже?



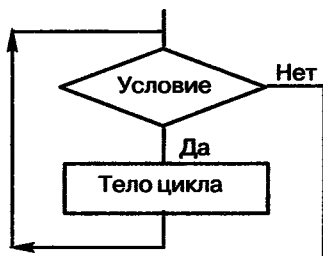
- а) линейный
- б) разветвляющийся
- в) циклический
- г) вспомогательный

18. К какому виду алгоритмов можно отнести алгоритм, схема которого представлена ниже?



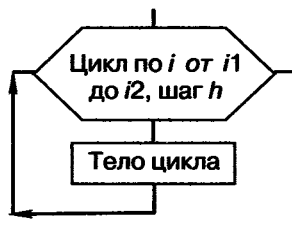
- а) линейный
- б) разветвляющийся с неполным ветвлением
- в) разветвляющийся с полным ветвлением
- г) циклический

19. К какому виду алгоритмов можно отнести алгоритм, схема которого представлена ниже?

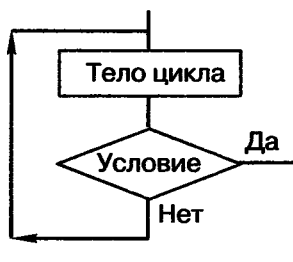


- а) цикл с параметром
- б) цикл с заданным условием продолжения работы
- в) цикл с заданным условием окончания работы
- г) цикл с заданным числом повторений

20. К какому виду алгоритмов можно отнести алгоритм, схема которого представлена ниже?



- а) цикл с заданным условием продолжения работы
  - б) цикл с заданным условием окончания работы
  - в) цикл с постусловием
  - г) цикл с заданным числом повторений
21. К какому виду алгоритмов можно отнести алгоритм, схема которого представлена ниже?



- а) цикл с заданным условием продолжения работы
  - б) цикл с заданным условием окончания работы
  - в) цикл с заданным числом повторений
  - г) цикл с предусловием
22. Сергей, Антон, Таня и Надя, гуляя по лесу, наткнулись на овраг, который можно перейти по шатковому мосту. Сергей может перейти его за минуту, Антон — за две, Таня — за три, Надя — за четыре. Фонарик у группы только один, и он обязательно нужен для перехода по мосту, который выдерживает только двоих человек. Когда два человека вместе идут по мосту, то идут они со скоростью более медлительного из них. Ребята смогли разработать алгоритм перехода на другой берег за минимально возможное время. Какое время она затратили на его исполнение?
- а) 10 минут
  - б) 11 минут
  - в) 12 минут
  - г) 13 минут
23. Дан фрагмент линейного алгоритма.

$a := 8$

$b := 6 + 3 * a$

$a := b / 3 * a$

Чему равно значение переменной  $a$  после его исполнения?

24. Исполните следующий фрагмент линейного алгоритма для  $a = x$  и  $b = y$ .



$a := a + b$

$b := b - a$

$a := a + b$

$b := -b$

Какие значения присвоены переменным  $a$  и  $b$ ?

а)  $y, x$

б)  $x + y, x - y$

в)  $x, y$

г)  $-y, x$

25. Определите значение целочисленных переменных  $x$  и  $y$  после выполнения алгоритма.



$x := 11$

$y := 5$

$t := y$

$y := x \bmod y$

$x := t$

$y := y + 2 * t$

а)  $x = 11, y = 5$

б)  $x = 5, y = 11$

в)  $x = 10, y = 5$

г)  $x = 5, y = 10$

26. Среди четырёх монет есть одна фальшивая. Неизвестно, легче она или тяжелее настоящей. Какое минимальное количество взвешиваний необходимо сделать на весах с двумя чашками без гирь, чтобы определить фальшивую монету?

а) 2

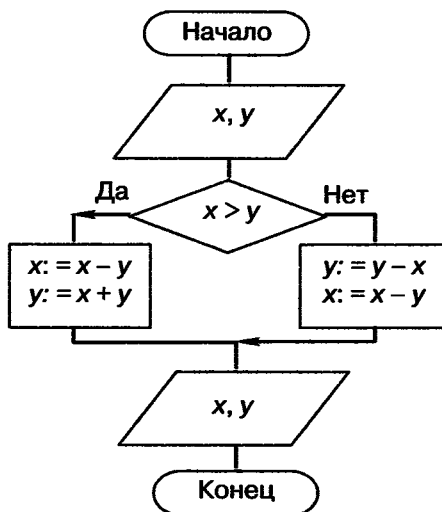
б) 3

в) 4

г) 5



27. Выполните алгоритм при  $x = 10$  и  $y = 15$ .

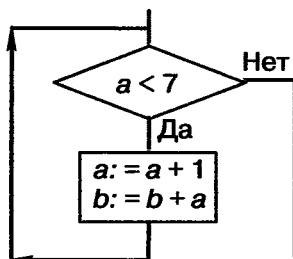


Какие значения будут получены в результате его работы?

- а)  $-5, 10$
- б)  $5, 20$
- в)  $10, 15$
- г)  $5, 5$
- д)  $-5, 5$



28. Выполните фрагмент алгоритма при  $a = 2$  и  $b = 0$ .



Определите значение переменной  $b$  после выполнения фрагмента алгоритма.



29. Определите значение переменной  $f$  после выполнения фрагмента алгоритма.



```
f:=1
нц для i от 1 до 5
  f:=f*i
кц
```

30. Определите значение переменной  $s$  после выполнения фрагмента алгоритма.



```
s:=0
нц для i от 1 до 5
  s:=s+i*i
кц
```

---

Для проверки знаний и умений по теме «Основы алгоритмизации» вы можете воспользоваться интерактивным тестом к главе 2, содержащимся в электронном приложении к учебнику.



# Глава 3

## НАЧАЛА ПРОГРАММИРОВАНИЯ

### § 3.1

#### Общие сведения о языке программирования Паскаль

*Ключевые слова:*

- язык программирования
- программа
- алфавит
- служебные слова
- типы данных
- структура программы
- оператор присваивания



**Языки программирования** — это формальные языки, предназначенные для записи алгоритмов, исполнителем которых будет компьютер. Записи алгоритмов на языках программирования называются **программами**.



Существует несколько тысяч языков программирования. Мы с вами познакомимся с языком программирования **Паскаль**, который был разработан в 70-х годах прошлого века Никлаусом Виртом (Швейцария). Своё название этот язык получил в честь французского учёного Блеза Паскаля, известного не только своими достижениями в математике, физике и философии, но и созданием первой в мире механической машины, выполнявшей сложение двух чисел.

Язык Паскаль считается универсальным языком программирования, так как он может применяться для записи алгоритмов решения самых разных задач (вычислительных, обработки текстов, построения графических изображений, поиска информации и т. д.).

Он поддерживает *процедурный стиль программирования*, в соответствии с которым программа представляет собой последовательность операторов, задающих те или иные действия<sup>1</sup>.

**Никлаус Вирт** (род. в 1934 г.) — швейцарский учёный, специалист в области информатики, один из известнейших теоретиков в области разработки языков программирования, профессор компьютерных наук. Разработчик языка Паскаль и ряда других языков программирования.



Рекомендуем вам зайти на сайт [pascalabc.net](http://pascalabc.net). Здесь вы найдёте много полезной информации для начинающих программистов, сможете скачать систему программирования PascalABC.NET.



### 3.1.1. Алфавит и словарь языка

Основой языка программирования Паскаль, как и любого другого языка, является **алфавит** — набор допустимых символов, которые можно использовать для записи программы. Это:

- латинские прописные буквы (A, B, C, ..., X, Y, Z);
- латинские строчные буквы (a, b, c, ..., x, y, z);
- арабские цифры (0, 1, 2, ..., 7, 8, 9);
- специальные символы (знак подчёркивания; знаки препинания; круглые, квадратные и фигурные скобки; знаки арифметических операций и др.).

В качестве неделимых элементов (составных символов) рассматриваются следующие последовательности символов:

- := (знак операции присваивания);
- >= и <= (знаки  $\geq$  и  $\leq$ );
- (\* и \*) (начало и конец комментария).

В языке существует также некоторое количество различных цепочек символов, рассматриваемых как единые смысловые элементы с фиксированным значением. Такие цепочки символов называются **служебными словами**. В таблице 3.1 приведены основные служебные слова, которые мы будем использовать при записи программ на языке Паскаль.

<sup>1</sup> С другими стилями программирования вы познакомитесь при изучении курса информатики в 10–11 классах.

### Служебные слова языка Паскаль

Служебное слово языка Паскаль	Значение служебного слова
<code>and</code>	и
<code>array</code>	массив
<code>begin</code>	начало
<code>do</code>	выполнить
<code>else</code>	иначе
<code>for</code>	для
<code>if</code>	если
<code>of</code>	из
<code>or</code>	или
<code>procedure</code>	процедура
<code>program</code>	программа
<code>repeat</code>	повторять
<code>then</code>	то
<code>to</code>	до (увеличивая до)
<code>until</code>	до (до тех пор, пока)
<code>var</code>	переменная
<code>while</code>	пока

Для обозначения констант, переменных, программ и других объектов используются имена — любые отличные от служебных слов последовательности букв, цифр и символа подчёркивания, начинающиеся с буквы или символа подчёркивания.

Прописные и строчные буквы в именах не различаются.

Длина имени может быть любой. Для удобства мы будем пользоваться именами, длина которых не превышает 8 символов.

#### 3.1.2. Типы данных, используемые в языке Паскаль

В языке Паскаль используются различные типы данных. Мы будем пользоваться некоторыми из так называемых простых типов данных (табл. 3.2).

Таблица 3.2

## Некоторые типы данных в языке Паскаль

Название	Обозначение	Допустимые значения	Область памяти
Целочисленный	integer <sup>1</sup>	-32 768 .. 32 767	2 байта со знаком
Вещественный	real	$\pm(2,9 \cdot 10^{-39} \dots 1,7 \cdot 10^{+38})$	6 байтов
Символьный	char	Произвольный символ алфавита	1 байт
Строковый	string	Последовательность символов длиной меньше 255	1 байт на символ
Логический	boolean	true и false	1 байт

В вещественном числе целая часть от дробной отделяется точкой, при этом перед точкой и после неё должно быть, по крайней мере, по одной цифре. Пробелы внутри числа недопустимы.

## 3.1.3. Структура программы на языке Паскаль

В программе, записанной на языке Паскаль, можно выделить:

- 1) заголовок программы;
- 2) блок описания используемых данных;
- 3) блок описания действий по преобразованию данных (программный блок).

Заголовок программы состоит из служебного слова **program** и имени программы. После имени программы ставится точка с запятой.

Блок описания данных состоит из раздела описания констант (**const**), раздела описания переменных (**var**) и некоторых других разделов<sup>2</sup>. В разделе описания переменных указываются имена используемых в программе переменных и их типы. Имена переменных одного типа перечисляются через запятую, затем после двоеточия ука-

<sup>1</sup> integer — основной, но не единственный тип для работы с целочисленными данными. Дополнительную информацию по этому вопросу вы можете найти в справочниках по программированию на языке Паскаль.

<sup>2</sup> В 8 классе мы ограничимся рассмотрением разделов описания констант и переменных, оставив изучение других разделов для старшей школы.

зывается их тип; описание каждого типа заканчивается точкой с запятой. Ниже приведён пример раздела описания переменных:

```
var i, j: integer; x: real; a: char;
```

Целый тип                      Вещественный тип                      Символьный тип

Программа может не иметь заголовка; в ней может отсутствовать блок описания данных. Обязательной частью программы является программный блок. Он содержит команды, описывающие алгоритм решения задачи. Программный блок начинается со слова **begin** и заканчивается словом **end** с точкой.

Ниже приведён общий вид программы:

```
program <имя программы>;
  const <список постоянных значений>;
  var <описание используемых переменных>;
begin <начало программного блока>
  <оператор 1>;
  <оператор 2>;
  ...
  <оператор n>
end.
```

**Операторы** — языковые конструкции, с помощью которых в программах записываются действия, выполняемые над данными в процессе решения задачи.

Точка с запятой служит разделителем между операторами, а не является окончанием соответствующего оператора.

Перед оператором **end** точку с запятой ставить не нужно.

### 3.1.4. Оператор присваивания

Основное преобразование данных, выполняемое компьютером, — присваивание переменной нового значения, что означает изменение содержимого области памяти; оно осуществляется **оператором присваивания**, аналогичным команде присваивания алгоритмического языка. Общий вид оператора:

```
<имя переменной> := <выражение>
```

Операция присваивания допустима для всех приведённых в табл. 3.2 типов данных. Выражения в языке Паскаль конструируются по рассмотренным ранее правилам для алгоритмического языка.

Рассмотрим процесс выполнения операторов присваивания на следующем примере:

```
a:=10;
b:=5;
s:=a+b
```

При выполнении оператора  $a:=10$  в ячейку оперативной памяти компьютера с именем  $a$  заносится значение 10; при выполнении оператора  $b:=5$  в ячейку оперативной памяти компьютера с именем  $b$  заносится значение 5. При выполнении оператора  $s:=a+b$  значения ячеек оперативной памяти с именами  $a$  и  $b$  переносятся в процессор, где над ними выполняется операция сложения. Полученный результат заносится в ячейку оперативной памяти с именем  $s$  (рис. 3.1).

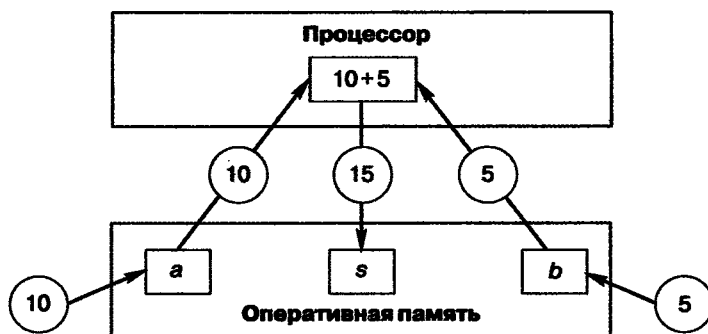


Рис. 3.1. Процесс выполнения оператора присваивания

## САМОЕ ГЛАВНОЕ

Паскаль — универсальный язык программирования, получивший своё название в честь выдающегося учёного Блеза Паскаля.

В языке Паскаль используются различные типы данных: целочисленный (integer), вещественный (real), символьный (char), строковый (string), логический (boolean) и другие.

В программе, записанной на языке Паскаль, можно выделить:

- 1) заголовок программы;
- 2) описание используемых данных;
- 3) описание действий по преобразованию данных (программный блок).

Общий вид программы:

```
program <имя программы>;  
  const <список постоянных значений>;  
  var <описание используемых переменных>;  
begin  
  <оператор 1>;  
  <оператор 2>;  
  ...  
  <оператор N>  
end.
```



### Вопросы и задания



1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Дополняет ли презентация информацию, содержащуюся в тексте параграфа? Какими слайдами вы могли бы дополнить презентацию?
2. В честь кого назван язык программирования Паскаль? Подготовьте краткую биографическую справку об этом учёном.
3. Почему язык программирования Паскаль считается универсальным?
4. Что входит в состав алфавита языка Паскаль?
5. Каких требований следует придерживаться при выборе имён для различных объектов в языке Паскаль?
6. Указывая название, обозначение, диапазон и занимаемую область памяти, опишите известные вам типы данных, используемые в языке Паскаль.
7. В чём разница между числами 100 и 100.0 в языке Паскаль?
8. Какую структуру имеет программа, записанная на языке Паскаль?
9. Как записывается раздел описания переменных?
10. Запишите раздел описания переменных, необходимых для вычисления:
  - а) значения функции  $y = x^2$ ;
  - б) площади прямоугольника;
  - в) стоимости покупки, состоящей из нескольких тетрадей и такого же количества обложек;
  - г) стоимости покупки, состоящей из нескольких тетрадей, нескольких ручек и нескольких карандашей.



11. Опишите процесс выполнения операторов присваивания.

`a:=3; b:=4; a:=a+b`

12. Запишите оператор для:

а) вычисления среднего арифметического переменных  $x_1$  и  $x_2$ ;

б) уменьшения на единицу значения переменной  $k$ ;

в) увеличения на единицу значения переменной  $i$ ;

г) вычисления стоимости покупки, состоящей из нескольких тетрадей, нескольких ручек и нескольких карандашей.

## § 3.2

### Организация ввода и вывода данных

**Ключевые слова:**

- оператор вывода write
- формат вывода
- оператор ввода read

#### 3.2.1. Вывод данных

В предыдущем параграфе мы познакомились со структурой программы на языке Паскаль, научились описывать данные, рассмотрели оператор присваивания. Этого достаточно для того, чтобы записать программу преобразования данных. Но результат этих преобразований нам виден не будет.

Для вывода данных из оперативной памяти на экран монитора используется оператор вывода write:

```
write (<выражение 1>, <выражение 2>, ..., <выражение N >)
```

список вывода

Здесь в круглых скобках помещается список вывода — список выражений, значения которых выводятся на экран. Это могут быть числовые, символьные и логические выражения, в том числе переменные и константы.

Произвольный набор символов, заключённый в апострофы, считается строковой константой. Строковая константа может содержать любые символы, набираемые на клавиатуре.

**Пример.** Оператор write ('s=', s) выполняется так:

- 1) на экран выводятся символы, заключённые в апострофы: s=
- 2) на экран выводится значение переменной, хранящейся в ячейке оперативной памяти с именем s.

Если значение переменной  $s$  равно 15 и она имеет целочисленный тип, то на экране появится:  $s=15$

Если значение переменной  $s$  равно 15, но она имеет вещественный тип, то на экране появится:  $s=1.5E+01$

При выполнении оператора вывода все элементы списка вывода печатаются непосредственно друг за другом. Так, в результате работы оператора `write (1, 20, 300)` на экран будет выведена последовательность цифр 120300, которая будет восприниматься нами как число 120300, а не как три отдельные числовые константы. Сделать выводимые данные более доступными для восприятия можно разными способами:

Вариант организации вывода	Оператор вывода	Результат
Добавить разделители — запятые	<code>write (1, ',', 20, ',', 300)</code>	1,20,300
Добавить разделители — пробелы	<code>write (1, ' ', 20, ' ', 300)</code>	1 20 300
Указать формат вывода	<code>write (1:3, 20:4, 300:5)</code>	1 20 300

**Формат вывода** — это указываемое после двоеточия целое число, определяющее, сколько позиций на экране должна занимать выводимая величина. Если цифр в числе меньше, чем зарезервированных под него позиций на экране, то свободные позиции дополняются пробелами слева от числа. Если указанное в формате вывода после двоеточия число меньше, чем необходимо, то оно автоматически будет увеличено до минимально необходимого.

Для вывода вещественного числа в списке вывода для каждого выражения указываются два параметра: 1) общее количество позиций, отводимых под число; 2) количество позиций в дробной части числа:

Оператор вывода	Результат выполнения оператора
<code>write ('s=', s:2:0);</code>	$s=15$
<code>write ('s=', s:3:1);</code>	$s= 15.0$
<code>write ('s=', s:5:1);</code>	$s= 15.0$

При выполнении нового оператора `write` вывод продолжается в той же строке. Чтобы осуществить переход к новой строке, ис-

пользуется оператор `writeln`. Других различий между операторами `write` и `writeln` нет.

### 3.2.2. Первая программа на языке Паскаль

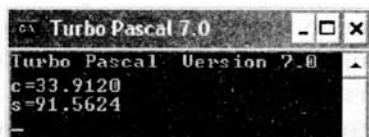
Пользуясь рассмотренными операторами, составим программу, вычисляющую длину окружности и площадь круга радиуса 5,4 см.

Исходным данным в этой задаче является радиус:  $r = 5,4$  см. Результатом работы программы должны быть величины  $c$  — длина окружности и  $s$  — площадь круга.  $c$ ,  $s$  и  $r$  — величины вещественного типа.

Исходные данные и результаты связаны соотношениями, известными из курса математики:  $c = 2\pi r$ ,  $s = \pi r^2$ . Программа, реализующая вычисления по этим формулам, будет иметь вид:

```
program n_1;
  const pi=3.14;
  var r, c, s: real;
begin
  r:=5.4;
  c:=2*pi*r;
  s:=pi*r*r;
  writeln ('c=', c:6:4);
  writeln ('s=', s:6:4)
end.
```

Эта программа верна и решает поставленную задачу. Запустив её на выполнение, вы получите следующий результат:



```
Turbo Pascal 7.0
Turbo Pascal Version 7.0
c=33.9120
s=91.5624
```

И всё-таки составленная нами программа имеет существенный недостаток: она находит длину окружности и площадь круга для единственного значения радиуса (5,4 см).

Для того чтобы вычислить длину окружности и площадь круга для другого значения радиуса, потребуется вносить изменения непосредственно в текст программы, а именно изменять оператор присваивания. Внесение изменений в существующую программу, по меньшей мере, не всегда удобно (например, когда программа большая и операторов присваивания много). Ниже вы познакомитесь с оператором, позволяющим вводить исходные данные в процессе работы программы, не прибегая к изменению текста программы.

### 3.2.3. Ввод данных с клавиатуры

Для ввода в оперативную память значений переменных используется оператор ввода `read`:

```
read (<имя переменной 1>, <имя переменной 2>, <имя переменной N>)
```

список ввода

При выполнении оператора `read` компьютер переходит в режим ожидания данных: пользователь должен ввести данные с клавиатуры и нажать клавишу `Enter`<sup>1</sup>. Несколько значений переменных числовых типов могут вводиться через пробел или через запятую. При вводе символьных переменных пробел и запятая воспринимаются как символы, поэтому ставить их нельзя.

Первое введённое пользователем значение переменной помещается в ячейку памяти, имя которой расположено первым в списке ввода, и т. д. Поэтому типы вводимых значений (входного потока) должны соответствовать типам переменных, указанных в разделе описания переменных.

**Пример.** Пусть .

```
var i, j: integer; x: real; a: char;
```

Присвоим переменным `i`, `j`, `x`, `a` значения 1, 0, 2,5 и 'A'. Для этого воспользуемся оператором

```
read (i, j, x, a)
```

и организуем входной поток одним из следующих способов:

```
1 0 2.5 A<Enter>   1,0,2.5<Enter>   1<Enter>
                   A<Enter>           0<Enter>
                                   2.5<Enter>
                                   A<Enter>
```

Здесь мы не только использовали различные разделители (пробел, запятая), но и представляли входной поток в виде одной, двух и четырёх строк.

Для ввода данных с клавиатуры можно также использовать оператор `readln`. Отличие состоит в том, что после выполнения `readln` осуществляется автоматический переход на новую строку входного потока, даже если в текущей строке остались невведённые символы. Таким образом, `readln` позволяет считать лишь начальную часть введённой пользователем строки и, проигнорировав её окончание, перейти к следующей строке.

<sup>1</sup> Нажатием клавиши `Enter` может сопровождаться ввод каждого значения.

Усовершенствуем программу `n_1`, организовав в ней ввод данных с помощью оператора `read`. А чтобы пользователь знал, для чего предназначена программа, и понимал, какое именно действие ожидает от него компьютер, выведем соответствующие текстовые сообщения с помощью оператора `writeln`:



```
program n_2;
  const pi=3.14;
  var r, c, s: real;
begin
  writeln('Вычисление длины окружности и площади круга');
  write('Введите r>>');
  readln(r);
  c:=2*pi*r;
  s:=pi*r*r;
  writeln ('c=', c:6:4);
  writeln ('s=', s:6:4)
end.
```

Результат работы усовершенствованной программы:

```
Turbo Pascal 7.0
Turbo Pascal Version 7.0 Copyright (c) 1983,92
Вычисление длины окружности и площади круга
Введите r>> 8.5
c=53.3800
s=226.8650
```

Теперь наша программа может вычислить длину окружности и площадь круга для любого значения  $r$ . Иначе говоря, она решает не единичную задачу, а целый класс задач. Кроме того, в программе понятно и удобно организован ввод исходных данных и вывод получаемых результатов. Это обеспечивает дружелюбность пользовательского интерфейса.

### САМОЕ ГЛАВНОЕ

Для ввода в оперативную память значений переменных используются операторы ввода `read` и `readln`.

Для вывода данных из оперативной памяти на экран монитора используются операторы вывода `write` и `writeln`.

Ввод исходных данных и вывод результатов должны быть организованы понятно и удобно; это обеспечивает дружелюбность пользовательского интерфейса.

Вопросы и задания



1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Используйте эти материалы при подготовке ответов на вопросы и выполнении заданий.
2. Запишите оператор, обеспечивающий во время работы программы ввод значения переменной *summa*.
3. Целочисленным переменным *i*, *j*, *k* нужно присвоить соответственно значения 10, 20 и 30. Запишите оператор ввода, соответствующий входному потоку:
  - а) 20 10 30
  - б) 30 20 10
  - в) 10 30 20
4. Опишите переменные, необходимые для вычисления площади треугольника по его трём сторонам, и запишите оператор, обеспечивающий ввод необходимых исходных данных.
5. Что является результатом выполнения оператора?
  - а) write (a)
  - б) write ('a')
  - в) write ('a=', a)
6. Какой тип имеет переменная *f*, если после выполнения оператора write (f) на экран было выведено следующее число?
  - а) 125
  - б) 1.25E+2
7. Каким образом можно вывести на экран вещественное число?
8. Запишите операторы ввода двух чисел и вывода их в обратном порядке.
9. Дан фрагмент программы:  

```
read (a); read (b); c:=a+b; write (a, b); write (c)
```

 Упростите его, сократив число операторов ввода и вывода.
10. Дан фрагмент программы:  

```
a:=10; b:=a+1; a:=b-a; write (a, b)
```

 Какие числа будут выведены на экран компьютера?
11. Напишите программу, которая вычисляет площадь и периметр прямоугольника по длинам двух его сторон.



## § 3.3

### Программирование линейных алгоритмов

**Ключевые слова:**

- вещественный тип данных
- целочисленный тип данных
- символьный тип данных
- строковый тип данных
- логический тип данных

Программы, реализующие линейные алгоритмы, являются простейшими. Все имеющиеся в них операторы выполняются последовательно, один за другим.

Программируя линейные алгоритмы, рассмотрим более подробно целочисленные, логические, символьные и строковые типы данных.

#### 3.3.1. Числовые типы данных

Вы уже знакомы с основными числовыми типами данных `integer` и `real`. К ним применимы стандартные функции, часть из которых приведена в табл. 3.3.

Таблица 3.3

#### Стандартные функции Паскаля

Функция	Назначение	Тип аргумента	Тип результата
<code>abs (x)</code>	Модуль $x$	<code>integer, real</code>	Такой же, как у аргумента
<code>sqr (x)</code>	Квадрат $x$	<code>integer, real</code>	Такой же, как у аргумента



Функция	Назначение	Тип аргумента	Тип результата
sqrt(x)	Квадратный корень из $x$	integer, real	real
round(x)	Округление $x$ до ближайшего целого	real	
int(x)	Целая часть $x$	real	
frac(x)	Дробная часть $x$	real	
random	Случайное число от 0 до 1	-	real
random(x)	Случайное число от 0 до $x$	integer	integer

Исследуем работу функций round, int и frac, применив их к некоторому вещественному  $x$ . Соответствующая программа будет иметь вид:

```

program n_3;
  var x: real;
begin
  writeln ('Исследование функций round, int, frac');
  write ('Введите x>>');
  readln (x);
  writeln ('Округление - ', round(x));
  writeln ('Целая часть - ', int(x));
  writeln ('Дробная часть - ', frac(x))
end.
    
```

Запустите программу несколько раз для каждого  $x \in \{10,2; 10,8; -10,2; -10,8\}$ . Что вы можете сказать о типе результата каждой из этих функций?

### 3.3.2. Целочисленный тип данных

Над целыми числами в языке Паскаль выполняются следующие операции: сложение (+), вычитание (-), умножение (\*), получение целого частного (div), получение целого остатка деления (mod) и деление (/). Результаты первых пяти операций — целые числа. Результатом операции деления может быть вещественное число.

Рассмотрим пример использования операций div и mod, записав на языке Паскаль программу нахождения суммы цифр вводимого с клавиатуры натурального трёхзначного числа.

Используем тот факт, что положительное трёхзначное число можно представить в виде следующей суммы:  $x = a \cdot 100 + b \cdot 10 + c$ , где  $a, b, c$  — цифры числа.



```
program n_4;
  var x, a, b, c, s: integer;
begin
  writeln ('Нахождение суммы цифр трёхзначного числа');
  write ('Введите исходное число>>');
  readln (x);
  a:=x div 100;
  b:=x mod 100 div 10;
  c:=x mod 10;
  s:=a+b+c;
  writeln ('s= ', s)
end.
```



Чему равна сумма цифр числа 123? А числа -123? Совпадают ли ваши результаты с результатами работы программы? Как можно объяснить и исправить ошибку в программе?

### 3.3.3. Символьный и строковый типы данных

Значением символьной величины (тип `char`) в языке Паскаль является любой из символов, который можно получить на экране нажатием на клавиатуре одной из клавиш или комбинации клавиш, а также некоторых других символов, в том числе и невидимых. Множество таких символов состоит из 256 элементов, каждому из которых согласно используемой кодовой таблице поставлен в соответствие код — число 0 до 255.

Символы, соответствующие первым 32 кодам, являются управляющими, а остальные — изображаемыми. К изображаемым символам относится и пробел, имеющий код 32.

Знакам препинания, знакам арифметических операций, цифрам, прописным и строчным латинским буквам соответствуют коды от 33 до 127. Буквам национального алфавита соответствуют коды с номерами 128 и далее.

В тексте программы константу символьного типа можно задать, заключив любой изображаемый символ в апострофы: `'5'`, `'В'`, `'*'`.

Если значение символьной переменной считывается с клавиатуры, то его следует набирать без апострофов.

Чтобы найти код символа, используют функцию `ord`, где в качестве параметра задают символ.

Чтобы по коду узнать символ, используют функцию `chr`, где в качестве параметра указывают код символа.

Значением строковой величины (тип `string`) является произвольная последовательность символов, заключенная в апострофы. В Паскале (как и в алгоритмическом языке) строки можно сцеплять.

**Пример.** Запишем на языке Паскаль программу, в которой для введённой с клавиатуры буквы на экран выводится её код. Затем на экран выводится строка, представляющая собой последовательность из трёх букв используемой кодовой таблицы: буквы, предшествующей исходной; исходной буквы; буквы, следующей за исходной.



```

program n_5;
  var a: char; kod: integer; b: string;
begin
  writeln ('Код и строка');
  write ('Введите исходную букву>>');
  readln (a);
  kod:=ord(a);
  b:=chr(kod-1)+a+chr(kod+1);
  writeln ('Код буквы ', a, '-', kod);
  writeln ('Строка: ', b)
end.
    
```

### 3.3.4. Логический тип данных

Как известно, величины логического типа принимают всего два значения; в Паскале это `false` и `true`. Эти константы определены так, что `false < true`.

Логические значения получаются в результате выполнения операций сравнения числовых, символьных, строковых и логических выражений. Поэтому в Паскале логической переменной можно присваивать результат операции сравнения.

**Пример.** Напишем программу, определяющую истинность высказывания «Число  $n$  является чётным» для произвольного целого числа  $n$ .

Пусть `ans` — логическая переменная, а  $n$  — целая переменная. Тогда в результате выполнения оператора присваивания

$$\text{ans} := n \bmod 2 = 0$$

переменной `ans` будет присвоено значение `true` при любом чётном  $n$  и `false` в противном случае.



```
program n_6;
  var n: integer; ans: boolean;
begin
  writeln ('Определение истинности высказывания
           о чётности числа');
  write ('Введите исходное число>>');
  readln (n);
  ans:=n mod 2 = 0;
  writeln ('Число ', n, ' является чётным - ', ans)
end.
```

Логическим переменным можно присваивать значения логических выражений, построенных с помощью известных вам логических функций **и**, **или**, **не**, которые в Паскале обозначаются соответственно **and**, **or**, **not**.



**Пример.** Напишем программу, определяющую истинность высказывания «Треугольник с длинами сторон *a*, *b*, *c* является равнобедренным» для произвольных целых чисел *a*, *b*, *c*.

```
program n_7;
  var a, b, c: integer; ans: boolean;
begin
  writeln ('Определение истинности высказывания
           о равнобедренном треугольнике');
  write ('Введите значения a, b, c>>');
  readln (a, b, c);
  ans:=(a=b) or (a=c) or (b=c);
  writeln ('Треугольник с длинами сторон ', a, ', ',
          b, ', ', c, ' является равнобедренным - ', ans)
end.
```

### САМОЕ ГЛАВНОЕ

В языке Паскаль используются вещественный, целочисленный, символьный, строковый, логический и другие типы данных. Для них определены соответствующие операции и функции.

Вопросы и задания



1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Используйте эти материалы при подготовке ответов на вопросы и выполнении заданий.



2. Для заданного  $x$  вычислите  $y$  по формуле

$$y = x^3 + 2,5x^2 - x + 1.$$

При этом:

- а) операцию возведения в степень использовать запрещено;
- б) в одном операторе присваивания можно использовать не более одной арифметической операции (сложение, умножение, вычитание);
- в) в программе может быть использовано не более пяти операторов присваивания.

*Подсказка:* преобразуйте выражение к следующему виду:

$$y = ((x + 2,5)x - 1)x + 1.$$



3. По заданным координатам точек  $A$  и  $B$  вычислите длину отрезка  $AB$ .



*Подсказка:* Расстояние  $d$  между точками  $A(x_a, y_a)$  и  $B(x_b, y_b)$  выражается формулой  $d = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}$ .

Пример входных данных	Пример выходных данных
$x_a=2$ $y_a=1$ $x_b=10$ $y_b=7$	$ AB =10.0$

4. Известны длины сторон треугольника  $a, b, c$ . Напишите программу, вычисляющую площадь этого треугольника.



Пример входных данных	Пример выходных данных
$a=3$ $b=4$ $c=5$	$S=6.0$



5. Известны координаты вершин  $A$ ,  $B$ ,  $C$  треугольника. Напишите программу, вычисляющую площадь этого треугольника.

Пример входных данных	Пример выходных данных
$x_a=2$ $y_a=1$ $x_b=6$ $y_b=5$ $x_c=10$ $y_c=1$	$S=16.0$

6. Если сумма налога исчисляется в рублях и копейках, то налоговая служба округляет её до ближайшего рубля (до 50 копеек — с недостатком, свыше 50 копеек (включая 50) — с избытком). Используйте компьютер, чтобы ввести точную сумму налога и вывести, сколько следует уплатить.
7. Исследуйте работу функции `random`, запустив многократно на выполнение программу:

```

program n_8;
  var x, n: integer;
begin
  writeln ('Исследование функции random');
  randomize (*для генерации различных случайных
            чисел при каждом запуске программы *);
  write ('Введите x>>');
  readln (x);
  write ('Введите n>>');
  readln (n);
  writeln ('random(', x, ')=', random(x));
  writeln ('random(', x, ')+', n, '=', random(x)+n)
end.
    
```

Как можно получить случайное число из интервала  $(0, x)$ ?

Как можно получить случайное число из интервала  $(0, x]$ ?

Как можно получить случайное число из интервала  $(n, x + n)$ ?

8. Одна компания выпустила лотерейные билеты трёх разрядов: для молодежи, для взрослых и для пенсионеров. Номера билетов каждого разряда лежат в пределах:

для молодёжи — от 1 до 100;  
 для взрослых — от 101 до 200;  
 для пенсионеров — от 201 до 250.

С помощью компьютера выберите случайным образом лотерейный билет в каждом разряде.

9. Запишите на языке Паскаль программу, которая для произвольного натурального двузначного числа определяет:
  - а) сумму и произведение его цифр;
  - б) число, образованное перестановкой цифр исходного числа.
10. Запишите на языке Паскаль программу, реализующую алгоритм работы кассира, выдающего покупателю сдачу ( $s$ ) наименьшим возможным количеством банкнот по 500 ( $k500$ ), 100 ( $k100$ ), 50 ( $k50$ ) и 10 ( $k10$ ) рублей.



Пример входных данных	Пример выходных данных
845	Следует сдать: банкнот по 500 руб. - 1 шт. банкнот по 100 руб. - 3 шт. банкнот по 50 руб. - 0 шт. банкнот по 10 руб. - 4 шт.

11. Идёт  $k$ -я секунда суток. Разработайте программу, которая по введённой  $k$ -й секунде суток определяет, сколько целых часов  $h$  и целых минут  $m$  прошло с начала суток. Например, если  $k = 13\ 257 = 3 \cdot 3600 + 40 \cdot 60 + 57$ , то  $h = 3$  и  $m = 40$ . Выведите на экран фразу: It is ... hours ... minutes. Вместо многоточий программа должна выводить значения  $h$  и  $m$ , отделяя их от слов ровно одним пробелом.

Пример входных данных	Пример выходных данных
13257	It is 3 hours 40 minutes.

12. Запишите на языке Паскаль программу, которая вычисляет сумму кодов букв в слове «БАЙТ».
13. Запишите на языке Паскаль программу, которая формирует и выводит на экран строку символов, коды которых равны 66, 69, 71, 73, 78.



14. Разработайте программу, которая запрашивает три строковые величины — взаимосвязанные прилагательное, существительное и глагол, а затем выводит все варианты фраз с использованием введённых слов.

Пример входных данных	Пример выходных данных
ЗЕЛЁНЫЕ	ЗЕЛЁНЫЕ ЛИСТЬЯ РАСПУСКАЮТСЯ
ЛИСТЬЯ	ЗЕЛЁНЫЕ РАСПУСКАЮТСЯ ЛИСТЬЯ
РАСПУСКАЮТСЯ	ЛИСТЬЯ ЗЕЛЁНЫЕ РАСПУСКАЮТСЯ
	ЛИСТЬЯ РАСПУСКАЮТСЯ ЗЕЛЁНЫЕ
	РАСПУСКАЮТСЯ ЗЕЛЁНЫЕ ЛИСТЬЯ
	РАСПУСКАЮТСЯ ЛИСТЬЯ ЗЕЛЁНЫЕ



15. Даны значения целочисленных переменных:  $a = 10$ ,  $b = 20$ . Чему будет равно значение логической переменной *rez* после выполнения операции присваивания?
- $rez := (a=10) \text{ or } (b>10)$
  - $rez := (a>5) \text{ and } (b>5) \text{ and } (a<20) \text{ and } (b<30)$
  - $rez := (\text{not } (a<15)) \text{ or } (b>20)$
16. Составьте программу, вводящую true, если высказывание является истинным, и false в противном случае:
- сумма цифр трёхзначного числа  $x$  является чётным числом;
  - треугольник со сторонами  $a$ ,  $b$ ,  $c$  является разносторонним.



## Программирование разветвляющихся алгоритмов

### **Ключевые слова:**

- условный оператор
- неполный условный оператор
- составной оператор
- вложенные ветвления

### 3.4.1. Условный оператор

При записи на языке Паскаль разветвляющихся алгоритмов используют условный оператор. Его общий вид:

```
if <условие> then <оператор_1> else <оператор_2>
```

Для записи неполных ветвлений используется неполная форма условного оператора:

```
if <условие> then <оператор>
```

Слова **if – then – else** переводятся с английского языка на русский как **если – то – иначе**, что полностью соответствует записи ветвления на алгоритмическом языке.

Перед **else** знак «;» не ставится.

В качестве условий используются логические выражения:

- простые — записанные с помощью операций отношения;
- сложные — записанные с помощью логических операций.

**Пример 1.** Запишем на языке Паскаль рассмотренный в п. 2.4.2 (пример 8) алгоритм определения принадлежности точки  $x$  отрезку  $[a, b]$ .



```
program n_9;
  var x, a, b: real;
begin
  writeln ('Определение принадлежности точки отрезку');
  write ('Введите a, b>>');
  readln (a, b);
  write ('Введите x>>');
  readln (x);
  if (x>=a) and (x<=b) then
    writeln ('Точка принадлежит отрезку')
  else writeln ('Точка не принадлежит отрезку')
end.
```



**Пример 2.** Воспользуемся неполным условным оператором для записи на языке Паскаль рассмотренного в п. 2.4.2 (пример 9) алгоритма присваивания переменной  $y$  значения наибольшей из трёх величин  $a$ ,  $b$  и  $c$ .

```
program n_10;
  var y, a, b, c: integer;
begin
  writeln ('Нахождение наибольшей из трёх величин');
  write ('Введите a, b, c>>');
  readln (a, b, c);
  y:=a;
  if (b>y) then y:=b;
  if (c>y) then y:=c;
  writeln ('y=', y)
end.
```



Дополните эту программу так, чтобы её выполнение приводило к присваиванию переменной  $y$  значения большей из четырёх величин  $a$ ,  $b$ ,  $c$  и  $d$ .

### 3.4.2. Составной оператор

В условном операторе и после **then**, и после **else** можно использовать только один оператор. Если при некотором условии требуется выполнить определённую последовательность операторов, то их объединяют в один составной оператор.

Конструкция вида

```
begin <последовательность операторов> end
```

называется **составным оператором**.



**Пример.** Алгоритм решения квадратного уравнения вам хорошо известен. Запишем соответствующую программу на языке Паскаль.

```

program n_11;
  var a, b, c: real;
  var d: real;
  var x, x1, x2: real;
begin
  writeln ('Решение квадратного уравнения');
  write ('Введите коэффициенты a, b, c>>');
  readln (a, b, c);
  d:=b*b-4*a*c;
  if d<0 then writeln ('Корней нет');
  if d=0 then
    begin
      x:=-b/2/a;
      writeln ('Корень уравнения x=', x:9:3)
    end;
  if d>0 then
    begin
      x1:=(-b+sqrt(d))/2/a;
      x2:=(-b-sqrt(d))/2/a;
      writeln ('Корни уравнения:');
      writeln ('x1=', x1:9:3);
      writeln ('x2=', x2:9:3)
    end
  end.

```

### 3.4.3. Многообразие способов записи ветвлений

В качестве оператора после **then** и **else** можно использовать условный оператор. Например, возможна следующая конструкция:

```

if <условие1> then
  if <условие2> then <оператор1>
  else <оператор2>

```

При использовании таких сложных конструкций (их ещё называют вложенными ветвлениями) следует иметь в виду, что **else** всегда относится к ближайшему оператору **if**.

**Пример.** Воспользуемся вложенным ветвлением для записи на языке Паскаль рассмотренного в п. 2.4.2 (пример 10) алгоритма решения линейного уравнения.



```
program n_12;
  var a, b, x: real;
begin
  writeln ('Решение линейного уравнения');
  write ('Введите коэффициенты a, b>>');
  readln (a, b);
  if a<>0 then
    begin
      x:=-b/a;
      writeln ('Корень уравнения x=', x:9:3)
    end
  else if b<>0 then writeln ('Корней нет')
  else writeln ('x - любое число');
end.
```

Как правило, для решения одной и той же задачи можно предложить несколько алгоритмов. Убедимся в этом, записав программу решения линейного уравнения, не прибегая к вложенным ветвлениям.



```
program n_13;
  var a, b, x: real;
begin
  writeln ('Решение линейного уравнения');
  write ('Введите коэффициенты a, b>>');
  readln (a, b);
  if a<>0 then
    begin
      x:=-b/a;
      writeln ('Корень уравнения x=', x:9:3)
    end;
  if (a=0) and (b<>0) then writeln ('Корней нет');
  if (a=0) and (b=0) then writeln ('x - любое число')
end.
```

Возможно, второй вариант программы покажется вам более наглядным. Но и у первого варианта есть свои преимущества: в нём делается меньше проверок.



Используйте вложенные ветвления для записи программы, определяющей принадлежность точки  $x$  отрезку  $[a, b]$ .

## САМОЕ ГЛАВНОЕ

При записи на языке Паскаль разветвляющихся алгоритмов используют условный оператор:

```
if <условие> then <оператор_1> else <оператор_2>
```

Для записи неполных ветвлений используется неполный условный оператор:

```
if <условие> then <оператор>
```

Если при некотором условии требуется выполнить определённую последовательность операторов, то их объединяют в один составной оператор, имеющий вид:

```
begin <последовательность операторов> end
```

## Вопросы и задания



1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Используйте эти материалы при подготовке ответов на вопросы и выполнении заданий.
2. Как на языке Паскаль записывается полное и неполное ветвление?
3. Является ли условным оператором следующая последовательность символов?
  - a) `if x<y then x:=0 else read (y)`
  - b) `if x>=y then x:=0; y:=0 else write (z)`
  - в) `if x<y<z then a:=a+1`
4. Что такое составной оператор? Для чего он используется в условном операторе?
5. Используя составной оператор, упростите следующий фрагмент программы:
 

```
if a>b then c:=1;
if a>b then d:=2;
if a<=b then c:=3;
if a<=b then d:=4
```
6. Дано трёхзначное число. Напишите программу, которая определяет:



- а) есть ли среди цифр заданного целого трёхзначного числа одинаковые;

Пример входных данных	Пример выходных данных
123	Нет
121	Да
222	Да

- б) является ли число «перевёртышем», т. е. числом, десятичная запись которого читается одинаково слева направо и справа налево.

Пример входных данных	Пример выходных данных
122	Нет
121	Перевёртыш
222	Перевёртыш



7. Даны две точки в плоской прямоугольной системе координат. Напишите программу, определяющую, которая из точек находится ближе к началу координат.

Пример входных данных	Пример выходных данных
Координаты 1-й точки >> 1 2 Координаты 2-й точки >> 3 4	1-я точка ближе



8. Даны три натуральных числа. Напишите программу, определяющую, существует ли треугольник с такими длинами сторон. Если такой треугольник существует, то определите его тип (равносторонний, равнобедренный, разносторонний).

Пример входных данных	Пример выходных данных
a b c >> 1 2 1	Не существует
a b c >> 2 2 2	Равносторонний
a b c >> 20 20 30	Равнобедренный
a b c >> 3 4 5	Разносторонний

9. Имеются данные о количестве полных лет трёх призёров спартакиады. Напишите программу, выбирающую и выводящую возраст самого младшего призёра.
10. Напишите программу, определяющую, лежит ли точка  $A(x_a, y_a)$  на прямой  $y = kx + l$ , на ней или под ней.



Пример входных данных	Пример выходных данных
k, l >> -1 5 x <sub>a</sub> , y <sub>a</sub> >> 1 2	Точка лежит под прямой.
k, l >> -1 5 x <sub>a</sub> , y <sub>a</sub> >> 1 10	Точка лежит над прямой.
k, l >> -1 5 x <sub>a</sub> , y <sub>a</sub> >> 1 4	Точка лежит на прямой.

11. Напишите программу, которая производит обмен значений переменных  $x$  и  $y$ , если  $x$  больше  $y$ .



Пример входных данных	Пример выходных данных
x >> 5 y >> 6	x = 5 y = 6
x >> 6 y >> 5	x = 5 y = 6

12. Дан условный оператор:



```

if a < 5 then c := 1
else if a > 5 then c := 2
      else c := 3
    
```

Какое значение имеет переменная  $a$ , если в результате выполнения условного оператора переменной  $c$  присваивается значение 3?

13. Напишите программу, вычисляющую значение функции:

$$y = \begin{cases} -1 & \text{при } x < 0, \\ 0 & \text{при } x = 0, \\ 1 & \text{при } x > 0. \end{cases}$$

Пример входных данных	Пример выходных данных
-5	$y=-1$
0	$y=0$
5	$y=1$

14. Составьте программу для решения задачи № 21 к § 2.4 (определение дня недели).
15. Поле шахматной доски определяется парой натуральных чисел, каждое из которых не превосходит 8. Напишите программу, которая по введённым координатам двух полей ( $k$ ,  $l$ ) и ( $m$ ,  $n$ ) определяет, имеют ли эти поля один цвет.

Пример входных данных	Пример выходных данных
Координаты 1-го поля>>2 2 Координаты 2-го поля>>3 3	Поля одного цвета
Координаты 1-го поля>>2 3 Координаты 2-го поля>>3 3	Поля разного цвета
Координаты 1-го поля>>2 7 Координаты 2-го поля>>5 4	Поля одного цвета

16. Напишите программу, в которой пользователю предлагается дополнить до 100 некоторое целое число  $a$  ( $a$  — случайное число, меньше 100). Ответ пользователя проверяется и комментируется.



## § 3.5

# Программирование циклических алгоритмов

### Ключевые слова:

- **while** (цикл-ПОКА)
- **repeat** (цикл-ДО)
- **for** (цикл с параметром)

### 3.5.1. Программирование циклов с заданным условием продолжения работы

Цикл с заданным условием продолжения работы (цикл-ПОКА) программируется в языке Паскаль с помощью оператора **while**.  
Общий вид оператора:

```
while <условие> do <оператор>
```

Здесь:

<условие> — логическое выражение; пока оно истинно, выполняется тело цикла;

<оператор> — простой или составной оператор, с помощью которого записано тело цикла.

Запишем на языке Паскаль рассмотренный в п. 2.4.3 (пример 14) алгоритм получения частного  $q$  и остатка  $r$  от деления натурального числа  $x$  на натуральное число  $y$  без использования операции деления.

```
program n_14;  
  var x, y, q, r: integer;  
begin  
  writeln ('Частное и остаток');  
  write ('Введите делимое x>>');  
  readln (x);  
  write ('Введите делитель y>>');
```



```
read (y);
r:=x;
q:=0;
while r>=y do
begin
  r:=r-y;
  q:=q+1
end;
writeln ('Частное q=', q);
writeln ('Остаток r=', r)
end.
```



Каким будет результат выполнения программы при  $x = -10$  и  $y = 3$ ? Как вы можете объяснить этот результат?

### 3.5.2. Программирование циклов с заданным условием окончания работы

Цикл с заданным условием окончания работы (цикл-ДО) программируется в языке Паскаль с помощью оператора **repeat**. Общий вид оператора:

```
repeat <оператор1; оператор2; ...; > until <условие>
```

Здесь:

<оператор1>; <оператор2>; ... — операторы, образующие тело цикла;

<условие> — логическое выражение; если оно ложно, то выполняется тело цикла.

Запишем на языке Паскаль рассмотренный в п. 2.4.3 (пример 17) алгоритм решения задачи о графике тренировок спортсмена.



```
program n_15;
var i: integer; x: real;
begin
  writeln ('График тренировок');
  i:=1;
  x:=10;
  repeat
    i:=i+1;
    x:=x+0.1*x;
  until x>=25;
  writeln ('Начиная с ', i, '-го дня спортсмен будет пробегать 25 км')
end.
```

### 3.5.3. Программирование циклов с заданным числом повторений

Цикл с заданным числом повторений (цикл-ДЛЯ) программируется в языке Паскаль с помощью оператора `for`. Его общий вид:

```
for <параметр>:=<начальное_значение> to <конечное_значение>
do <оператор>
```

Здесь:

<параметр> — переменная целого типа;

<начальное\_значение> и <конечное\_значение> — выражения того же типа, что и параметр, вычисляемые перед началом цикла;

<оператор> — простой или составной оператор — тело цикла.

При выполнении этого оператора после каждого выполнения тела цикла происходит увеличение на единицу параметра цикла; условием выхода из цикла является превышение параметром конечного значения.

Запишем на языке Паскаль рассмотренный в п. 2.4.3 (пример 19) алгоритм вычисления степени с натуральным показателем  $n$  для любого вещественного числа  $a$ .

```
program n_16;
  var i, n: integer; a, y: real;
begin
  writeln ('Возведение в степень');
  write ('Введите основание a>>');
  readln (a);
  write ('Введите показатель n>>');
  readln (n);
  y:=1;
  for i:=1 to n do y:=y*a;
  writeln ('y=', y)
end.
```



### 3.5.4. Различные варианты программирования циклического алгоритма

Особенностью программирования является то, что для решения одной и той же задачи могут быть созданы разные программы. Вы могли убедиться в этом, программируя ветвления. Рассмотрим пример, показывающий, что и циклический алгоритм может быть запрограммирован разными способами.



**Пример.** Напишем программу, в которой осуществляется ввод целых чисел (ввод осуществляется до тех пор, пока не будет введён ноль) и подсчёт количества введённых положительных и отрицательных чисел.

Так как здесь в явном виде задано условие окончания работы, то воспользуемся оператором **repeat**.

```
program n_17;
  var n, k1, k2: integer;
begin
  k1:=0;
  k2:=0;
  repeat
    write ('Введите целое число>>');
    readln (n);
    if n>0 then k1:=k1+1;
    if n<0 then k2:=k2+1;
  until n=0;
  writeln ('Введено:');
  writeln ('положительных чисел - ', k1);
  writeln ('отрицательных чисел - ', k2)
end.
```

Имеющееся условие окончания работы можно достаточно просто преобразовать в условие продолжения работы — работа продолжается, пока  $n \neq 0$ . И мы можем воспользоваться оператором **while**:

```
program n_18;
  var n, k1, k2: integer;
begin
  k1:=0;
  k2:=0;
  while n<>0 do
  begin
    writeln ('Введите целое число>>');
    read (n);
    if n>0 then k1:=k1+1;
    if n<0 then k2:=k2+1;
  end;
  writeln ('Введено:');
  writeln ('положительных - ', k1);
  writeln ('отрицательных - ', k2)
end.
```

В рассмотренном примере число повторений тела цикла заранее не известно, и поэтому оператор `for` здесь применить нельзя. Если число повторений тела цикла известно, то лучше воспользоваться оператором `for`. Вместе с тем любая задача, в которой число повторений тела цикла определено заранее, может быть запрограммирована с помощью любого из трёх рассмотренных выше циклов.

## САМОЕ ГЛАВНОЕ

В языке Паскаль имеются три вида операторов цикла: `while` (цикл-ПОКА), `repeat` (цикл-ДО), `for` (цикл с параметром). Если число повторений тела цикла известно, то лучше воспользоваться оператором `for`; в остальных случаях используются операторы `while` и `repeat`.

## Вопросы и задания

1. Ознакомьтесь с материалами презентации к параграфу, содержащейся в электронном приложении к учебнику. Используйте эти материалы при подготовке ответов на вопросы и выполнении заданий.

2. Дана последовательность операторов:

```
a:=1;
b:=2;
while a+b<8 do
begin
  a:=a+1;
  b:=b+2;
end;
s:=a+b
```

Сколько раз будет повторен цикл и какими будут значения переменных  $a$ ,  $b$ ,  $s$  после исполнения этой последовательности операторов?

3. Требовалось написать программу вычисления факториала числа  $n$  (факториал числа  $n$  есть произведение всех целых чисел от 1 до  $n$ ). Программист торопился и написал программу неправильно. Ниже приведён фрагмент его программы, в котором содержатся пять ошибок:

```
k:=1;  
f:=0;  
while k<n do  
  f:=f*k;  
  k:=k+1
```

Найдите ошибки. Допишите необходимые операторы и выполните программу на компьютере.

Пример входных данных	Пример выходных данных
Введите n>>5	5!=120
Введите n>>6	6!=720

4. Проанализируйте следующий цикл:

```
while a<b do  
  c:=a=b;
```

В чём его особенность?

5. Запишите на языке Паскаль программы решения задач № 25–29 из § 2.4. Используйте оператор **while**.
6. Дана последовательность операторов:

```
a:=1;  
b:=1;  
repeat  
  a:=a+1;  
  b:=b*2;  
until b>8;  
s:=a+b
```

Сколько раз будет повторён цикл и какими будут значения переменных *a*, *b*, *s* после исполнения этой последовательности операторов?

7. Напишите программу, в которой осуществляется ввод целых чисел (ввод осуществляется до тех пор, пока не будет введён ноль) и подсчёт суммы и среднего арифметического введённых положительных чисел. Используйте оператор **repeat**.
8. Напишите программу, в которой осуществляется ввод целых чисел (ввод осуществляется до тех пор, пока не будет введён ноль) и определение максимального (наибольшего) из введённых чисел. Используйте оператор **repeat**.

9. Напишите программу вычисления наибольшего общего делителя двух целых чисел:
- а) используйте оператор **repeat**;
  - б) используйте оператор **while**.
10. Сколько раз будет выполнен цикл?
- а) **for i:=0 to 15 do s:=s+1;**
  - б) **for i:=10 to 15 do s:=s+1;**
  - в) **for i:=-1 to 1 do s:=s+1;**
  - г) **for i:=10 to 10 do s:=s+1;**
  - д) **k:=5;**  
**for i:=k-1 to k+1 do s:=s+1;**
11. Напишите программу, которая 10 раз выводит на экран ваше имя и фамилию.
12. Напишите программу, выводящую на экран изображение шахматной доски, где чёрные клетки изображаются звёздочками, а белые — пробелами. Рекомендуемый вид экрана после выполнения программы:

```

*      *      *      *
      *      *      *      *
*      *      *      *
      *      *      *      *
*      *      *      *
      *      *      *      *
*      *      *      *
      *      *      *      *

```

13. Напишите программу, которая вычисляет сумму:
- а) первых  $n$  натуральных чисел;
  - б) квадратов первых  $n$  натуральных чисел;
  - в) всех чётных чисел в диапазоне от 1 до  $n$ ;
  - г) всех двузначных чисел.
14. Напишите программу, которая генерирует 10 случайных чисел в диапазоне от 1 до 20, выводит эти числа на экран и вычисляет их среднее арифметическое.




-  15. Запишите на языке Паскаль программы решения задач № 32, 33 из § 2.4. Используйте оператор `for`.
16. Напишите программу, которая выводит на экран таблицу степеней двойки (от нулевой до десятой). Рекомендуемый вид экрана после выполнения программы:

Таблица степеней двойки:

0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024

-  17. Напишите программу, которая выводит на экран таблицу умножения на  $n$  ( $n$  — целое число в диапазоне от 2 до 10, вводимое с клавиатуры).

Пример входных данных	Пример выходных данных
Введите $n \gg 5$	$5 * 2 = 10$
	$5 * 3 = 15$
	$5 * 4 = 20$
	$5 * 5 = 25$
	$5 * 6 = 30$
	$5 * 7 = 35$
	$5 * 8 = 40$
	$5 * 9 = 45$
	$5 * 10 = 50$

-  18. Какой из трёх рассмотренных операторов цикла является, по вашему мнению, основным, т. е. таким, что им можно заметить два других? Обоснуйте свою точку зрения.



## Тестовые задания для самоконтроля



1. Разработчиком языка Паскаль является:
  - а) Блез Паскаль
  - б) Никлаус Вирт
  - в) Норберт Винер
  - г) Эдсгер В. Дейкстра
2. Что из нижеперечисленного не входит в алфавит языка Паскаль?
  - а) латинские строчные и прописные буквы
  - б) служебные слова
  - в) русские строчные и прописные буквы
  - г) знак подчёркивания
3. Какая последовательность символов не может служить именем в языке Паскаль?
  - а) \_mas
  - б) maS1
  - в) d2
  - г) 2d
4. Вещественные числа имеют тип данных:
  - а) real
  - б) integer
  - в) boolean
  - г) string
5. В программе на языке Паскаль обязательно должен быть:
  - а) заголовок программы
  - б) блок описания используемых данных
  - в) программный блок
  - г) оператор присваивания

6. Какого раздела не существует в программе, написанной на языке Паскаль?
- а) заголовка
  - б) примечаний
  - в) описаний
  - г) операторов
7. Языковые конструкции, с помощью которых в программах записываются действия, выполняемые в процессе решения задачи, называются:
- а) операндами
  - б) операторами
  - в) выражениями
  - г) данными
8. Разделителями между операторами служит:
- а) точка
  - б) точка с запятой
  - в) пробел
  - г) запятая
9. Описать переменную — это значит указать её:
- а) имя и значение
  - б) имя и тип
  - в) тип и значение
  - г) имя, тип и значение
10. В данном фрагменте программы:
- ```
program error;  
begin  
    SuMmA:=25-14;  
end.
```
- ошибкой является:
- а) некорректное имя программы
  - б) не определённое имя переменной
  - в) некорректное имя переменной
  - г) запись арифметического выражения
11. Какая клавиша нажимается после набора последнего данного в операторе read?
- а) Enter
  - б) точка с запятой
  - в) пробел
  - г) Ctrl

12. При присваивании изменяется:
- а) имя переменной
  - б) тип переменной
  - в) значение переменной
  - г) значение константы
13. Для вывода результатов в Паскале используется оператор
- а) begin
  - б) readln
  - в) write
  - г) print
14. Для вычисления квадратного корня из  $x$  используется функция:
- а) abs(x)
  - б) sqr(x)
  - в) sqrt(x)
  - г) int(x)
15. Для генерации случайного целого числа из интервала [10, 20) необходимо использовать выражение:
- а) random\*20
  - б) random(20)
  - в) random(10)+10
  - г) random(10)\*2
16. В каком из условных операторов допущена ошибка?
- а) **if** b=0 **then** writeln('Деление невозможно.');
  - б) **if** a<b **then** min:=a; **else** min:=b;
  - в) **if** a>b **then** max:=a **else** max:=b;
  - г) **if** (a>b) **and** (b>0) **then** c:=a+b;
17. В условном операторе и после **then**, и после **else** нельзя использовать:
- а) оператор вывода
  - б) составной оператор
  - в) несколько операторов
  - г) условный оператор
18. Определите значение переменной  $c$  после выполнения следующего фрагмента программы:
- ```
a:=100;  
b:=30;  
a:=a-b*3;  
if a>b then c:=a-b else c:=b-a;
```



- а) 20
- б) 70
- в) -20
- г) 180

19. Условный оператор

**if** a mod 2=0 **then** write ('Да') **else** write ('Нет')  
позволяет определить, является ли число *a*:

- а) целым
- б) двузначным
- в) чётным
- г) простым

20. Какого оператора цикла не существует в языке Паскаль?

- а) **for**
- б) **while**
- в) **repeat...until**
- г) **loop**

21. Цикл в фрагменте программы

```
p:=2;  
repeat  
  p:=p*0.1  
until p<0.1;  
будет исполнен:
```

- а) 0 раз
- б) 1 раз
- в) 2 раза
- г) бесконечное число раз

22. Цикл в фрагменте программы

```
a:=1;  
b:=1;  
while a+b<8 do  
begin  
  a:=a+1;  
  b:=b+2  
end;  
выполнится:
```

- а) 0 раз
- б) 2 раза
- в) 3 раза
- г) бесконечное число раз

23. Определите значения переменных  $s$  и  $i$  после выполнения фрагмента программы:



```
s:=0; i:=5;
while i>0 do
begin
  s:=s+i;
  i:=i-1;
end;
```

- а)  $s = 0, i = -1$   
б)  $s = 5, i = 0$   
в)  $s = 15, i = 5$   
г)  $s = 15, i = 0$
24. Выберите фрагмент программы, в котором ищется произведение  $1 \cdot 2 \cdot 3 \cdot 4 \cdot 5$ :

а)  $p:=0; i:=1; \text{ while } i \leq 5 \text{ do } i:=i+1; p:=p*i;$   
б)  $p:=1; i:=1; \text{ while } i < 6 \text{ do } i:=i+1; p:=p*i;$   
в)  $p:=1; i:=1; \text{ while } i < 6 \text{ do begin } p:=p*i; i:=i+1 \text{ end};$   
г)  $p:=1; i:=1; \text{ while } i > 5 \text{ do begin } p:=p*i; i:=i+1 \text{ end};$

25. В данном фрагменте программы



```
s:=0;
for i:=1 to 10 do
  s:=s+2*i;
```

вычисляется:

- а) сумма целых чисел от 1 до 10  
б) сумма чётных чисел от 1 до 10  
в) удвоенная сумма целых чисел от 1 до 10  
г) сумма первых десяти натуральных чётных чисел

Для проверки знаний и умений по теме «Начала программирования» вы можете воспользоваться интерактивным тестом к главе 3, содержащимся в электронном приложении к учебнику.



# Ответы и решения к вопросам и заданиям для самостоятельной подготовки

## Глава 1

### § 1.1

3. Приведены знаки, используемые для записи чисел в следующих системах счисления: древнеегипетской, вавилонской, майя, римской, старославянской, двоичной, троичной, ..., шестнадцатеричной.

8. а)  $110011_2$ ; б)  $111_4$ .

9. Минимальное основание 5. 38, 62, 31, 71.

10. а) да; б) нет.

11. а) 5; б)  $2 \cdot x^3 + 0 \cdot x^2 + 0 \cdot x^1 + 2 \cdot x^0 = 130$ ,  
 $2x^3 = 128$ ,  $x^3 = 64$ ,  $x = 4$ .

18. а)  $12 \cdot 3 - 4 = 32$ ; б)  $12 : 2 - 2 = 4$ ; в)  $12 : 3 - 4 = 0$ .

19. а) 10; б) 198.

### § 1.2

4. 00111111.

5. а) +76.

6.  $101010_2$ .

9. а)  $0,217934 \cdot 10^3$ ; б)  $0,75321 \cdot 10^5$ ; в)  $0,101 \cdot 10^{-2}$ .

### § 1.3

7. 920, 80.

10. 18.

12. 1) 0; 2) 0; 3) 1; 4) 1.

13. 1) 1; 2) 1; 3) 0; 4) 1.

14. Смит и Джон.

15. Финикийский сосуд, изготовлен в V веке.

16.  $F = \overline{A} \vee \overline{B}$ .

Глава 2

§ 2.1

14. 1, 1, 2, 3, 5, 8, 13, ...

19. 6, 12212.

§ 2.3

10. Не более одной переменной.

15. г)  $(x > 0)$  или  $(y > 0)$ ;

е)  $((x > 0) \text{ и } (y \leq 0))$  или  $((y > 0) \text{ и } (x \leq 0))$ .

18. а)  $t := x > 0$ ; в)  $t := (x = y)$  и  $(x = z)$ .

§ 2.4

6.  $y = ((2x + 3)x + 4)x + 5$ ;  $y = 14$  при  $x = 1$ .

7.  $h := tfh * 24$ .

21. если  $chislo = 3$  то  $y :=$  понедельник.

Глава 3

§ 3.1

10. в) Пусть  $n$  — количество тетрадей (обложек),  $ct$  — цена одной тетради (целое число),  $co$  — цена одной обложки (целое число),  $s$  — общая стоимость покупки. В этом случае раздел описания переменных будет иметь вид: **var**  $s, n, ct, co$ : **integer**;

12. б)  $k := k - 1$ ;

§ 3.2

10. 111.

§ 3.3

15. а) true; б) true; в) false.

§ 3.4

3. а) да; б) нет; в) нет.

12. 5.

§ 3.5

2. Два раза. 3, 6, 9.

6. Четыре раза. 5, 16, 21.

10. а) 16; б) 6; в) 3; г) 1; д) 3.

# Ключи к тестовым заданиям для самоконтроля

## Глава 1

Задание	1	2	3	4	5	6	7	8	9	10
Ответ	в	б	в	б	б	г	в	б	б	г
Задание	11	12	13	14	15	16	17	18	19	20
Ответ	в	б	г	а	а	г	в	б	б	г

## Глава 2

Задание	1	2	3	4	5	6	7	8	9	10
Ответ	а	г	в	б	а	в	г	д	11121	127
Задание	11	12	13	14	15	16	17	18	19	20
Ответ	в	в	а	г	б	б	а	в	б	г
Задание	21	22	23	24	25	26	27	28	29	30
Ответ	б	б	80	а	б	а	г	25	120	55

## Глава 3

Задание	1	2	3	4	5	6	7	8	9	10
Ответ	б	в	г	а	в	б	б	б	б	б
Задание	11	12	13	14	15	16	17	18	19	20
Ответ	а	в	в	в	в	б	в	а	в	г
Задание	21	22	23	24	25					
Ответ	в	б	г	в	г					



# Оглавление

<b>Введение</b> .....	<b>3</b>
<b>Глава 1. Математические основы информатики</b> .....	<b>5</b>
§ 1.1. Системы счисления .....	5
1.1.1. Общие сведения о системах счисления .....	5
1.1.2. Двоичная система счисления .....	8
1.1.3. Восьмеричная система счисления .....	9
1.1.4. Шестнадцатеричная система счисления .....	10
1.1.5. Правило перевода целых десятичных чисел в систему счисления с основанием $q$ .....	10
1.1.6. Двоичная арифметика .....	12
1.1.7. «Компьютерные» системы счисления .....	13
§ 1.2. Представление чисел в компьютере .....	17
1.2.1. Представление целых чисел .....	17
1.2.2. Представление вещественных чисел .....	19
§ 1.3. Элементы алгебры логики .....	22
1.3.1. Высказывание .....	22
1.3.2. Логические операции .....	24
1.3.3. Построение таблиц истинности для логических выражений .....	29
1.3.4. Свойства логических операций .....	30
1.3.5. Решение логических задач .....	32
1.3.6. Логические элементы .....	34
Тестовые задания для самоконтроля .....	41
<b>Глава 2. Основы алгоритмизации</b> .....	<b>46</b>
§ 2.1. Алгоритмы и исполнители .....	46
2.1.1. Понятие алгоритма .....	46
2.1.2. Исполнитель алгоритма .....	48
2.1.3. Свойства алгоритма .....	51

2.1.4. Возможность автоматизации деятельности человека . . . . .	53
§ 2.2. Способы записи алгоритмов . . . . .	57
2.2.1. Словесные способы записи алгоритма . . . . .	57
2.2.2. Блок-схемы . . . . .	59
2.2.3. Алгоритмические языки . . . . .	60
§ 2.3. Объекты алгоритмов . . . . .	63
2.3.1. Величины . . . . .	63
2.3.2. Выражения . . . . .	65
2.3.3. Команда присваивания . . . . .	66
2.3.4. Табличные величины . . . . .	68
§ 2.4. Основные алгоритмические конструкции . . . . .	73
2.4.1. Следование . . . . .	73
2.4.2. Ветвление . . . . .	76
2.4.3. Повторение . . . . .	81
Тестовые задания для самоконтроля . . . . .	97
<b>Глава 3. Начала программирования. . . . .</b>	<b>106</b>
§ 3.1. Общие сведения о языке программирования Паскаль . . . . .	106
3.1.1. Алфавит и словарь языка . . . . .	107
3.1.2. Типы данных, используемые в языке Паскаль . . . . .	108
3.1.3. Структура программы на языке Паскаль . . . . .	109
3.1.4. Оператор присваивания . . . . .	110
§ 3.2. Организация ввода и вывода данных . . . . .	114
3.2.1. Вывод данных. . . . .	114
3.2.2. Первая программа на языке Паскаль . . . . .	116
3.2.3. Ввод данных с клавиатуры . . . . .	117
§ 3.3. Программирование линейных алгоритмов . . . . .	120
3.3.1. Числовые типы данных . . . . .	120
3.3.2. Целочисленный тип данных . . . . .	121
3.3.3. Символьный и строковый типы данных . . . . .	122
3.3.4. Логический тип данных . . . . .	123
§ 3.4. Программирование разветвляющихся алгоритмов. . . . .	129
3.4.1. Условный оператор . . . . .	129
3.4.2. Составной оператор . . . . .	130
3.4.3. Многообразие способов записи ветвлений . . . . .	131
§ 3.5. Программирование циклических алгоритмов . . . . .	137
3.5.1. Программирование циклов с заданным условием продолжения работы . . . . .	137
3.5.2. Программирование циклов с заданным условием окончания работы . . . . .	138

---

3.5.3. Программирование циклов с заданным числом повторений . . . . .	139
3.5.4. Различные варианты программирования циклического алгоритма . . . . .	139
Тестовые задания для самоконтроля . . . . .	145
Ответы и решения к вопросам и заданиям для самостоятельной подготовки . . . . .	150
Ключи к тестовым заданиям для самоконтроля . . . . .	152

УДК 004.9  
ББК 32.97  
Б85

爱  
谢

**Босова Л. Л.**

**Б85** Информатика : учебник для 8 класса / Л. Л. Босова, А. Ю. Босова. — 2-е изд., испр. — М. : БИНОМ. Лаборатория знаний, 2014. — 160 с. : ил.

ISBN 978-5-9963-1776-9

Учебник предназначен для изучения курса «Информатика» в 8 классе общеобразовательной школы. Он входит в состав учебно-методического комплекта по информатике для 5–9 классов, включающего авторскую программу, учебники, рабочие тетради, электронные приложения и методические пособия.

Выдержан принцип инвариантности к конкретным моделям компьютеров и версиям программного обеспечения. Теоретический материал учебника поддержан развернутым аппаратом организации усвоения изучаемого материала, обеспечивающим подготовку школьников к сдаче экзамена за курс основной школы в формате ГИА.

Предполагается широкое использование ресурсов федеральных образовательных порталов, в том числе Единой коллекции цифровых образовательных ресурсов (<http://sc.edu.ru/>).

Соответствует федеральному государственному образовательному стандарту основного общего образования (2010 г.).

УДК 004.9  
ББК 32.97

---

*Учебное издание*

**Босова Людмила Леонидовна  
Босова Анна Юрьевна**

**ИНФОРМАТИКА**

**Учебник для 8 класса**

Ведущий редактор *О. Полежаева*. Ведущий методист *И. Сретенская*  
Художественное оформление: *И. Марев*  
Художественный редактор *Н. Новак*. Иллюстрации: *Я. Соловцова*  
Технический редактор *Е. Денюкова*. Корректор *Е. Клитина*  
Компьютерная верстка: *С. Янковая*

Подписано в печать 18.02.14. Формат 70×100/16.  
Усл. печ. л. 13,00. Тираж 25 000 экз. Заказ № 35248.

Издательство «БИНОМ. Лаборатория знаний»  
125167, Москва, проезд Аэропорта, д. 3  
Телефон: (499) 157-5272, e-mail: [binom@Lbz.ru](mailto:binom@Lbz.ru)  
<http://www.Lbz.ru>, <http://e-umk.Lbz.ru>, <http://metodist.Lbz.ru>

При участии ООО Агентство печати «Столица»  
[www.apstolica.ru](http://www.apstolica.ru); e-mail: [apstolica@bk.ru](mailto:apstolica@bk.ru)

Отпечатано в соответствии с качеством предоставленных издательством  
электронных носителей в ОАО «Саратовский полиграфкомбинат».  
410004, г. Саратов, ул. Чернышевского, 59. [www.sarpk.ru](http://www.sarpk.ru)

Учебник входит в УМК по информатике для основной школы (5–9 классы).

Соответствует Федеральному государственному образовательному стандарту основного общего образования (2010 г.).

Включён в Федеральный перечень учебников, рекомендованных Министерством образования и науки Российской Федерации.

ISBN 978-5-9963-1776-9

