



**МИНИСТЕРСТВО ОБРАЗОВАНИЯ  
РЕСПУБЛИКИ БЕЛАРУСЬ**

**Белорусский национальный  
технический университет**

---

**Кафедра «Технология и методика преподавания»**

**Н. И. Астапчик**

**В. В. Пенкрат**

# **ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ**

*Учебно-методическое пособие*

**Минск  
БНТУ  
2018**

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  
Белорусский национальный технический университет

---

Кафедра «Технология и методика преподавания»

Н. И. Астапчик  
В. В. Пенкрат

## ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

Учебно-методическое пособие для студентов  
специальности 1-08 01 01 «Профессиональное обучение  
(по направлениям)», направления специальности 1-08 01 01-01  
«Профессиональное обучение (машиностроение)», 1-08 01 01-05  
«Профессиональное обучение (строительство)», 1-08 01 01-07  
«Профессиональное обучение (информатика)»

*Рекомендовано учебно-методическим объединением  
по профессионально-техническому обучению*

Минск  
БНТУ  
2018

УДК 004.42(075.8)

ББК 32.973я7

А91

Р е ц е н з е н т ы :

кафедра информатики Минского государственного высшего  
радиотехнического колледжа;  
канд. пед. наук, доцент *С. И. Зенько*

**Астапчик, Н. И.**

А91 Основы алгоритмизации и программирования: учебно-методическое пособие для студентов специальности 1-08 01 01 «Профессиональное обучение (по направлениям)», направления специальности 1-08 01 01-01 «Профессиональное обучение (машиностроение)», 1-08 01 01-05 «Профессиональное обучение (строительство)», 1-08 01 01-07 «Профессиональное обучение (информатика)» / Н. И. Астапчик, В. В. Пенкрат – Минск : БНТУ, 2018. – 183 с.  
ISBN 978-985-550-681-3.

Учебно-методическое пособие содержит теоретические материалы, примеры решения задач, контрольные вопросы, а также задания для самостоятельного выполнения.

Издание предназначено студентам специальности 1-08 01 01 «Профессиональное обучение (по направлениям)».

УДК 004.42(075.8)

ББК 32.973я7

ISBN 978-985-550-681-3

© Астапчик Н. И.,  
Пенкрат В. В., 2018

© Белорусский национальный  
технический университет, 2018

## ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ .....	5
Лабораторная работа №1. ВЫРАЖЕНИЯ, ИХ ТИПЫ .....	6
Теоретические сведения.....	6
Примеры выполнения заданий.....	12
Контрольные Вопросы .....	14
Задания для самостоятельного выполнения .....	15
Лабораторная работа № 2. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЛИНЕЙНОЙ СТРУКТУРЫ.....	17
Теоретические сведения.....	17
Примеры выполнения заданий.....	26
Контрольные вопросы.....	32
Задания для самостоятельного выполнения .....	34
Лабораторная работа № 3. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ РАЗВЕТВЛЯЮЩЕЙСЯ СТРУКТУРЫ.....	39
Теоретические сведения.....	39
Примеры выполнения заданий.....	42
Контрольные вопросы.....	45
Задания для самостоятельного выполнения .....	47
Лабораторная работа № 4. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЦИКЛИЧЕСКОЙ СТРУКТУРЫ.....	52
Теоретические сведения.....	52
Примеры выполнения заданий.....	56
Контрольные вопросы.....	62
Задания для самостоятельного выполнения .....	64
Лабораторная работа № 5. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ С ИСПОЛЬЗОВАНИЕМ ВЛОЖЕННЫХ ЦИКЛОВ .....	69
Теоретические сведения.....	69
Примеры выполнения заданий.....	69
Контрольные вопросы.....	73
Задания для самостоятельного выполнения .....	74
Лабораторная работа № 6. СТРОКОВЫЕ ПЕРЕМЕННЫЕ .....	77
Теоретические сведения.....	77

Примеры выполнения заданий .....	80
Контрольные вопросы.....	90
Задания для самостоятельного выполнения .....	91
<b>Лабораторная работа № 7. ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ПОЛЬЗОВАТЕЛЬСКИХ ПРОЦЕДУР И ФУНКЦИЙ .....</b>	
Теоретические сведения.....	94
Примеры выполнения заданий .....	99
Контрольные вопросы.....	103
Задания для самостоятельного выполнения .....	106
<b>Лабораторная работа №8. СТРУКТУРИРОВАННЫЙ ТИП ДАННЫХ. ОДНОМЕРНЫЕ МАССИВЫ .....</b>	
Теоретические сведения.....	113
Примеры выполнения заданий .....	124
Контрольные вопросы.....	131
Задания для самостоятельного выполнения .....	131
<b>Лабораторная работа № 9. ДВУМЕРНЫЕ МАССИВЫ. ....</b>	
Теоретические сведения.....	136
Примеры выполнения заданий .....	140
Контрольные вопросы.....	147
Задания для самостоятельного выполнения .....	148
<b>Лабораторная работа 10. КОМБИНИРОВАННЫЙ ТИП ДАННЫХ: ЗАПИСИ .....</b>	
Теоретические сведения.....	153
Примеры выполнения заданий .....	155
Контрольные вопросы.....	161
Задания для самостоятельного выполнения .....	161
<b>Лабораторная работа 11. ТИП ДАННЫХ: ФАЙЛЫ ДАННЫХ ...</b>	
Теоретические сведения.....	166
Текстовые файлы .....	168
Типизированные файлы .....	171
Примеры выполнения заданий .....	175
Контрольные вопросы.....	178
Задания для самостоятельного выполнения .....	179
<b>БИБЛИОГРАФИЧЕСКИЙ СПИСОК .....</b>	<b>183</b>

## ВВЕДЕНИЕ

Программирование как учебная дисциплина становится одной из основных при подготовке специалистов разных областей современного производства. У студентов инженерных специальностей разный исходный уровень знаний по программированию, полученных в школе, однако у каждого из них необходимо сформировать навыки алгоритмического мышления и программирования вычислительных процессов.

Любая практическая задача, как правило, вначале формулируется на естественном языке. Затем человек, пытающийся ее решить, ищет закономерности, позволяющие получить результат при имеющемся наборе данных. Этим самым он разрабатывает алгоритм решения задачи, а затем записывает его на каком-либо языке программирования, чтобы эту программу мог выполнить компьютер.

Основной целью данного пособия является обучение студентов основам программирования на языке PascalABC. Пособие содержит 11 лабораторных работ, в начале каждой из них приводится теоретический материал, изучение которого дает возможность выполнить задания, приведенные в каждой лабораторной работе. Задания распределены по 12 вариантам, в каждом из них по 4 задачи. Авторы старались сделать задания по вариантам однотипными. Также в каждой лабораторной работе имеются контрольные вопросы, отвечая на которые студент может самостоятельно проверить свои теоретические знания.

Пособие написано согласно учебным планам и программам учебных дисциплин «Основы алгоритмизации и программирования» для студентов специальности 1-08 01 01-07 «Профессиональное обучение (информатика)», учебной дисциплины «Информатика» для специальностей 1-08 01 01-01 «Профессиональное обучение (машиностроение)», «1-08 01 01-05 «Профессиональное обучение (строительство)».

## Лабораторная работа № 1

### ВЫРАЖЕНИЯ, ИХ ТИПЫ

*Цель работы:* получить первоначальные знания о языке ПаскальABC; научиться писать арифметические и логические выражения, находить их значения и определять тип выражений.

### Теоретические сведения

#### *Алфавит языка*

Языки программирования, как и любые другие языки, имеют свой алфавит. Алфавит языка программирования Pascal состоит из символов трех видов:

1. Прописные и строчные буквы латинского алфавита: A, B, C, D, T, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, причем прописная и строчная буква считаются одним символом.

2. Арабские цифры 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

3. Специальные символы: (, ), [, ], {, }, ' (апостроф), . (точка), , (запятая), : (двоеточие), ; (точка с запятой), + (плюс), - (минус), \* (звездочка), / (правый слеш), = (равно), > (больше), < (меньше), \_ (знак подчеркивания), пробел (не имеет начертания).

Из символов алфавита составляются слова. Слова бывают двух типов:

– **зарезервированные (служебные)** – это слова, которые придумали разработчики языка программирования, вложили в них определенный смысл, который остается постоянным для всех программ языка Паскаль (например, *begin, while, For, Repeat*);

– **идентификаторы (имена)** – это слова, которые придумывает программист, смысл этих слов сохраняется только в рамках одной программы. Идентификатор может состоять из букв и цифр, но на первом месте должна стоять буква. Также в именах можно использовать знак подчеркивания, но другие специальные символы использовать нельзя. Количество символов в них не должно превышать 126. Однако пользоваться очень длинными именами неудобно, поэтому на практике используют короткие имена.

**Числовые константы** – это:

1. Целые числа со знаком или без него. Например: 3458, –876, 37 105;

2. Вещественные числа, в которых целая часть от дробной отделяется *точкой*. Например, 3.5, –0.8713, 9801.003.

**Переменная** – это величина, которая *может* менять свое значение при выполнении программы. Каждой переменной программист дает свое имя – идентификатор. **Константа** – это величина, которая остается постоянной. Различают *стандартные и пользовательские* константы. Стандартные константы встроены в язык программирования. К ним можно отнести *PI* ( $\pi$ ), *MaxInt* (максимальное целое число 2 147 483 647), *MaxReal* (максимальное вещественное число  $1,7 \cdot 10^{308}$ ), *MinReal* (минимальное положительное вещественное число  $4,94065645841247 \cdot 10^{-324}$ ). Для констант пользователя в каждой программе дается имя и значение этой константы.

### ***Арифметические выражения***

К арифметическим операциям: + (сложение), – (вычитание), \* (умножение), / (деление), mod (нахождение остатка при целочисленном делении), div (нахождение частного при целочисленном делении).

*Тип результата операции* зависит от типов величин (операндов), входящих в эту операцию.

1. Для операций mod и div операнды могут быть только целого типа, тип результата также целый.

2. Для операций *сложения, вычитания и умножения* справедливо следующее:

– если оба операнда целые, то тип результата целый;

– если оба или хотя бы один операнд вещественный, то тип результата вещественный.

3. Для операции *деления* тип результата *всегда* вещественный, независимо от типов операндов.

Числовые константы и числовые переменные, соединенные знаками арифметических операций и стандартных функций, образуют *арифметические выражения*.



Арифметические выражения могут быть довольно сложными, поэтому при их записи должны соблюдаться следующие *правила*:

1. Все величины, входящие в арифметическое выражение, должны быть записаны в одну строку.
2. Все знаки операций должны быть обязательно проставлены;
3. Для указания порядка выполнения действий используют круглые скобки с учетом обще принятой математической иерархии действий.
4. Нельзя писать два знака операций один за другим.
5. Нельзя писать две константы одна за другой.
6. Нельзя писать две переменные одна за другой.

При вычислении значений арифметических выражений необходимо знать числовые значения всех величин, входящих в это выражение и их типы. Тип значения арифметического выражения может быть:

– *целым*, если все операнды этого выражения целого типа и используются любые арифметические операции, кроме деления (/), а также стандартные функции, у которых значение аргумента и значение функции могут быть только целого типа;

– *вещественным*, если хотя бы один операнд вещественного типа или есть хотя бы одна операция деления (/) или одна функция, значением которой может быть только вещественное число.

### ***Некоторые стандартные функции, типы аргумента и результата.***

В языке Pascal очень много стандартных математических функций, в табл. 1.1 приведены только некоторые из них. При записи стандартной функции аргумент обязательно должен заключаться в круглые скобки. Аргументом функции может быть числовая константа, числовая переменная и арифметическое выражение. Значением функции является число.

Таблица 1.1

Обозначение функции	Тип аргумента $x$	Тип результата	Операция
Abs( $x$ )	Целый, вещественный	Целый, вещественный	Вычисление абсолютного значения $x$
Sqr( $x$ )	Целый, вещественный	Целый, вещественный	Вычисление $x^2$

Продолжение табл. 1.1

Обозначение функции	Тип аргумента $x$	Тип результата	Операция
Sqrt( $x$ )	Целый, вещественный	Вещественный	Вычисление квадратного корня из $x$
Exp( $x$ )	Целый, вещественный	Вещественный	Вычисление $e^x$
Frac( $x$ )	Вещественный	Вещественный	Вычисление дробной части числа $x$
Int( $x$ )	Вещественный	Вещественный	Вычисление целой части числа $x$
Trunc( $x$ )	Вещественный	Целый	Нахождение целой части числа $x$ по следующим правилам: 1) если $x \geq 0$ , то результат будет $\leq x$ ; 2) если $x < 0$ , то результат будет $> x$
Succ( $x$ )	Целый	Целый	Выдает следующее за $x$ значение
Pred( $x$ )	Целый	Целый	Выдает предыдущее $x$ значение
Ln( $x$ )	Целый, вещественный	Вещественный	Вычисление Ln( $x$ )
Round( $x$ )	Вещественный	Целый	Округлить до ближайшего целого числа
Sin( $x$ )	Целый, вещественный	Вещественный	Вычисление Sin( $x$ )
Cos( $x$ )	Целый, вещественный	Вещественный	Вычисление Cos( $x$ )
ArcTan( $x$ )	Целый, вещественный	Вещественный	Вычисление ArcTg( $x$ )
Odd( $x$ )	Целый	Логический	Выдает значение «Истина», если число $x$ нечетное; значение «Ложь», если $x$ – четное
Random( $x$ )	Целый	Целый	Возвращает случайное целое число в диапазоне $[0; x - 1]$

Обозначение функции	Тип аргумента $x$	Тип результата	Операция
Random	–	Вещественный	Возвращает случайное число в диапазоне [0; 1)
Uprcase( $c$ )	Символьный	Символьный	Возвращает символ $c$ , преобразованный к верхнему регистру
LowCase( $c$ )	Символьный	Символьный	Возвращает символ $c$ , преобразованный к нижнему регистру
Power( $x, y$ )	Вещественный	Вещественный	Возвращает $x$ в степени $y$
Ord( $x$ )	Символьный	Целый	Возвращает код символа $x$
Chr( $x$ )	Целый	Символьный	Возвращает символ с кодом $x$

Для вычисления значений других функций следует пользоваться тождествами:

$$\arcsin x = \operatorname{arctg} \frac{x}{\sqrt{1-x \cdot x}};$$

$$\arccos x = \frac{\pi}{2 - \operatorname{arctg} \frac{x}{\sqrt{1-x \cdot x}}};$$

$$\operatorname{arctg} x = \frac{\pi}{2 - \operatorname{arctg} nx};$$

$$\log_b a = \frac{\ln a}{\ln b};$$

$$a^x = e^{x \cdot \ln a}, \text{ если } (a > 0).$$

## *Операции отношений и логические выражения*

К операциям отношения относятся:  $>$  (больше),  $\geq$  (больше или равно),  $<$  (меньше),  $\leq$  (меньше или равно),  $=$  (равно),  $\neq$  (не равно). С помощью этих операций составляют выражения, в которых обязательно должно быть два операнда. Операндами могут быть числовые константы, переменные и арифметические выражения. Например,  $A > B$ ,  $X + Y \leq 7$ ,  $3 < 9$ ,  $C = D + 2$ . Выражения, записанные с помощью операций отношения, являются *простыми логическими выражениями*. Значение такого выражения зависит от числовых значений, входящих в выражение, и может быть либо **TRUE** (истина), либо **FALSE** (ложь).

Простые логические выражения с помощью логических операций можно объединять в *сложные*.

*Логические операции* приведены в табл. 1.2.

Таблица 1.2

Запись операции на <i>Pascal</i>	Суть операции	Значение переменной <i>A</i>	Значение переменной <i>B</i>	Значение операции
$Not(A)$	Отрицание (нет)	True False	– –	False True
$(A) \text{ and } (B)$	Логическое умножение (союз и)	True True False False	True False True False	True False False False
$(A) \text{ or } (B)$	Логическое сложение (союз или)	True True False False	True False True False	True True True False
$(A) \text{ xor } (B)$	Исключенное или	True True False False	True False True False	False True True False

Примеры сложных логических выражений:  $(A > B) \text{ and } (A > C)$ ,  $(A = D) \text{ or } (X = Y)$ .

## Приоритет операций

Приоритетом (иерархией) называется очередность выполнения операций в выражении. Выполнение каждой операции происходит с учетом ее приоритета (табл. 1.3).

Таблица 1.3

Операция	Приоритет
Not (отрицание или изменение знака)	Первый (высший)
*, /, Div, Mod, And	Второй
+, -, or, xor	Третий
=, >, >=, <, <=, <>	Четвертый

## Примеры выполнения заданий

1. Укажите ошибки в следующих именах переменных или констант пользователя:

- End (это служебное слово, оно не может быть именем);
- 34Sum (здесь на 1 месте стоит цифра);
- Summa Chisel (в имени есть пробел);
- злг (в имени есть русская буква);
- ABC (нельзя сказать, есть ли здесь ошибка, так как нельзя понять, это русские буквы или латинские, однако компьютер четко различает, на каком языке набраны символы);
- z\_t\_w (правильная запись имени);
- Akwa (правильная запись имени);
- hRk (правильная запись имени);
- wW12 (правильная запись имени).

2. Запишите следующие математические выражения на языке ПаскальABC:

- $\frac{a^2 - b^2}{\sqrt{a - b}}$  на языке ПаскальABC. Ответ:  $(\text{sqr}(a) - \text{sqr}(b))/\text{sqrt}(a - b)$ ;
- $e^{x^2} - \frac{\sin(x)}{\text{tg}(x)}$  на языке ПаскальABC. Ответ:  $\text{exp}(\text{sqr}(x)) - \sin(x)/(\sin(x)/\cos(x))$ ;

–  $\log_2(x-5) + \sqrt{4ac}$  на языке ПаскальABC. Ответ:  $\ln(x - 5)/\ln(2) + \text{sqrt}(4 * a * c)$ ;

–  $\frac{(x-6)(x-4)}{(x+8)(x+2)}$  на языке ПаскальABC. Ответ:  $(x-6) * (x - 4) / ((x+8) * (x+2))$ .

3. Выражения, записанные на языке ПаскальABC, представьте в виде математических выражений:

–  $(-b + \text{sqrt}(\text{sqrt}(b) - 4 * a * c)) / (2 * a)$ . Ответ:  $\frac{-b + \sqrt{b^2 - 4ac}}{2a}$ ;

–  $(x-5) / ((y+3) * (z-2))$ . Ответ:  $\frac{x-5}{(y+3)(z-2)}$ .

4. Определите тип результатов арифметических выражений:

–  $B + 5$ , если  $B$  целое число, то в результате получится целое число; если  $B$  вещественное число, то в результате получится вещественное число;

–  $(X + 5) / (\text{sqrt}(y) + 2)$ , независимо от значений  $X$  и  $Y$  результат будет всегда вещественный, так как имеется операция  $/$  (деление).

5. Найдите значения выражений:

–  $467 \bmod 23$ . Ответ: при делении 467 на 23 остаток будет равен 7;

–  $467 \text{ div } 23$ . Ответ: при делении частное будет равно 20;

–  $(49/7+3) > 15$ . Ответ: значением этого выражения будет *False*;

–  $(35 \bmod 8) < (345 \bmod 10)$ . Ответ: значением этого выражения будет *True*.

6. При помощи формул языка Паскаль ABC запишите следующие математические утверждения:

– область определения следующего выражения:  $y = \sqrt{x-7}$ . Ответ:  $x - 7 \geq 0$ ;

–  $x \in [-5, 7]$ . Ответ:  $(x \geq -5) \text{ and } (x \leq 7)$ ;

– у равностороннего треугольника все стороны  $a$ ,  $b$ ,  $c$  равны. Ответ:  $(a = b) \text{ and } (a = c)$ ;

– у равнобедренного треугольника две стороны равны. Ответ:  $(a = b) \text{ or } (a = c) \text{ or } (b = c)$ ;

– треугольник существует, если длина одной стороны меньше суммы длин двух других сторон. Ответ:  $(a < b + c)$  and  $(b < a + c)$  and  $(c < a + b)$ ;

– запишите отрицание следующего выражения:  $4 < x < 12$ . Ответ:  $\text{not}((x > 4) \text{ and } (x < 12))$ ;

– область определения следующего выражения:  $\frac{x+5}{x+7}$ . Ответ:  $(x+7) \neq 0$ .

### Контрольные вопросы

1. Из каких символов состоит алфавит языка программирования Паскаль ABC?

2. Что является зарезервированным словом?

3. Что является идентификатором?

4. Зачем нужны идентификаторы?

5. Что представляет собой константа?

6. Что представляет собой переменная?

7. Какие операции относятся к арифметическим?

8. Что обозначает операция `mod`?

9. Что обозначает операция `div`?

10. Что такое арифметическое выражение?

11. Правило записи арифметических выражений.

12. Для чего предназначена функция *ODD*? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

13. Для чего предназначены функции *INT*, *FRAC*? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

14. Для чего предназначены функции *ROUND*, *TRUNC*? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

15. Для чего предназначена функция *ABS*? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

16. Для чего предназначена функция *RANDOM*? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

17. Для чего предназначены функции *ORD*, *CHR*? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

18. Для чего предназначена функция *UPCASE*? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

19. Для чего предназначены функции *SQR*, *SQRT*? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

20. Какие существуют операции отношения?

21. Что называется простым логическим выражением?

22. Какие логические операции вы знаете?

23. Как называется операция  $(A)$  and  $(B)$ ? Какие значения она может принимать?

24. Как называется операция  $(A)$  or  $(B)$ ? Какие значения она может принимать?

25. Как называется операция  $(A)$  xor  $(B)$ ? Какие значения она может принимать?

26. Как называется операция Not $(A)$ ? Какие значения она может принимать?

27. Каков порядок выполнения операций при вычислении значений выражений?

28. Записать тождество определения  $\arcsin(x) = \dots$

29. Записать тождество определения  $\arctg(x) = \dots$

30. Записать тождество определения  $\arccos(x) = \dots$

31. Записать тождество определения  $\text{arccotg}(x) = \dots$

32. Записать тождество определения  $\log_b a = \dots$

33. Записать тождество определения  $a^x = \dots$

34. Записать тождество определения  $a^2 = \dots$

35. Записать тождество определения  $a^{2/3} = \dots$

### Задания для самостоятельного выполнения

1. Укажите ошибки в следующих именах переменных:

3YTR; NO UN; NBVЩ; VFD 9; ФАН; S\_U L;  
RU,OP; WER.; XYZ(A; A15B+C; SWET\_#; -TY;  
(-1)KA; SPOL<sub>1</sub>; A<sup>2</sup>\$

2. Запишите математические выражения на языке ПаскальABC:

$$- \frac{\sqrt{a^2 + b^2}}{x^4 + y^4};$$



- $(a + b) \leq (a + c)$ ;
- $a = b = c$ ;
- $e^x + x_2 - \frac{1}{c + d}$ ;
- $1 + x + \arcsin x$ ;
- $x^{\frac{2}{3}}$ ;
- $\log_{10} x$ .

3. Выражения, записанные на языке ПаскальABC, представьте в виде математических выражений:

- $\text{Sqrt}(p * (p - a) * (p - b) * (p - c))$ ;
- $\text{sqrt}(\text{sqr}(x1 - x2) + \text{sqr}(y1 - y2))$ ;
- $m * \text{sqr}(d)/g$ ;
- $\text{exp}(\text{sqr}(x)) * \text{sqr}(y)$ ;
- $\text{exp}(3 * \ln(7))$ ;
- $\text{pi}/(2 - \arctan(x))$ .

4. Вычислите значения следующих выражений:

- $45 \bmod 37$ ;
- $22 \text{ div } 13$ ;
- $(59 \bmod 10) < (71 \text{ div } 7)$ ;
- $15 \bmod 10$ ;
- $63 \text{ div } 3$ ;
- $(95 \bmod 5) > (84 \text{ div } 4)$ ;
- $(34 \bmod 2) < (91 \text{ div } 10)$ ;
- $61 \bmod 20$ ;
- $98 \text{ div } 38$ .

## Лабораторная работа № 2

### ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЛИНЕЙНОЙ СТРУКТУРЫ

*Цель работы:* выработать практические навыки работы с системой PascalABC, научиться создавать, вводить в компьютер, выполнять и исправлять простейшие программы на языке *Pascal* в режиме диалога, познакомиться с диагностическими сообщениями компилятора об ошибках при выполнении линейных программ.

#### Теоретические сведения

##### *Окно редактора кода*

Большую часть рабочей области, ее верхнюю часть занимает окно редактора кода. В него вводится исходный текст программы.

Горячие клавиши, которые можно использовать при работе с текстом программы:

F2, Ctrl-S – сохранить файл;

F3, Ctrl-O – загрузить файл;

F12 – сохранить файл под новым именем;

Ctrl-Shift-S – сохранить все открытые файлы;

Ctrl-Tab, Ctrl-Shift-Tab – перейти к следующему/предыдущему окну редактора;

Ctrl-Shift-I – увеличить отступ выделенного блока;


Ctrl-Shift-U – уменьшить отступ выделенного блока.

##### *Окно вывода*

Под окном редактора расположено окно вывода. Оно предназначено для вывода данных процедурами *write* и *writeln*, а также для вывода сообщений об ошибках и предупреждений во время работы программы.

Окно вывода может быть скрыто. Клавиша F5 показывает/скрывает окно вывода. Для скрытия окна вывода используется также клавиша Esc.

Окно вывода обязательно открывается при любом выводе в него.

Для очистки окна вывода следует нажать комбинацию клавиш *Ctrl-Del* или кнопку .

### **Окно ввода**

Окно ввода открывается при выполнении процедур `read` и `readln` в ходе работы программы.

Ввод данных в окно ввода сопровождается эхо-выводом в окно вывода (рис 2.1). После нажатия клавиши `Enter` данные из окна ввода попадают в соответствующие переменные, окно ввода закрывается, программа работает дальше.

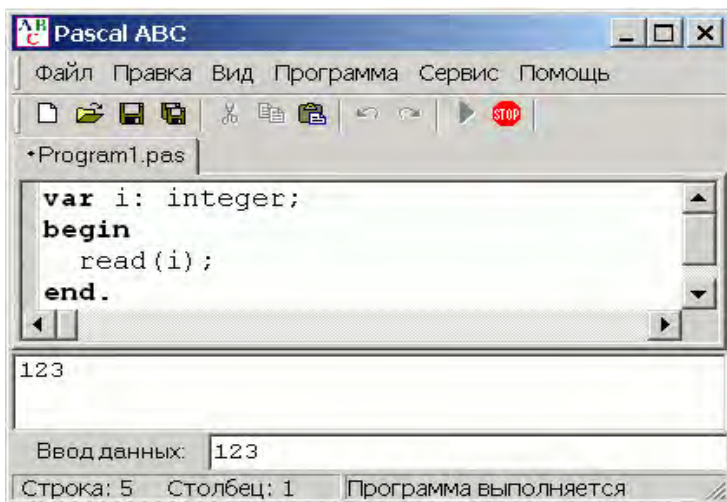




Рис. 2.1. Интерфейс программы PascalABC

### **Запуск и остановка программы**

Для запуска программы в текущем окне редактора следует нажать клавишу `F9` или кнопку  панели инструментов.

Вначале программа компилируется во внутреннее представление, после чего, если не найдены ошибки, она начинает выполняться. При выполнении программы кнопка ее запуска

программы становится неактивной, кнопка остановки программы, наоборот, активна, и в строке статуса отображается информация «Программа выполняется».

Выполнение программы можно в любой момент прервать нажатием комбинации клавиш Ctrl + F2 или кнопки . При этом в окне вывода появится сообщение: **Программа прервана пользователем.**

### *Структура программы*

Программа на языке *Pascal* состоит из заголовка, раздела описаний и раздела операторов (табл. 2.1).

Таблица 2.1

<b>PROGRAM</b> Имя;	Заголовок программы
<b>label</b> – раздел меток <b>const</b> – раздел констант <b>type</b> – раздел типов <b>var</b> – раздел переменных <b>procedure</b> – раздел процедур <b>function</b> – раздел функций	Раздел описаний
<b>BEGIN</b> Оператор 1; Оператор 2; ... Оператор n <b>END.</b>	Раздел операторов (тело программы)

*Заголовок* содержит служебное слово *PROGRAM*, имя программы, задаваемое программистом. Заканчивается заголовок символом «;» (точка с запятой).

*Раздел описаний* предназначен для объявления всех встречающихся в программе данных и их характеристик (имен данных, их типов, возможных значений). Этот раздел содержит следующие подразделы: объявление меток, констант, типов, переменных, объявление процедур и функций. Порядок расположения разделов не важен, описания могут повторяться.

Объявление процедур и функций является одним разделом. Следует заметить, что не все перечисленные разделы обязательны в программе. В простых программах могут потребоваться, например, только разделы описания констант и переменных.

После каждого описания ставится символ «;».

*Раздел операторов* («тело» программы) заключается в операторные скобки вида: *BEGIN* («начать») и *END* («окончить»), при этом после служебного слова *END* обязательно ставится точка. В разделе операторов записывается последовательность исполняемых операторов и каждый выражает действие, которое необходимо выполнить. Исполняемые операторы отделяются друг от друга символом «;».

Хороший стиль программирования требует:

- записывать слова PROGRAM, BEGIN, END с одинаковой позиции строки;

- по отношению к ним описания и операторы принято сдвигать вправо на три символа. Желательно делать сдвиг на одинаковое количество позиций от края или по отношению к предыдущему сдвигу.

### ***Константы и переменные***

В зависимости от способа хранения в компьютере данные можно разделить на две группы: *константы* и *переменные*. В программе они определяются идентификаторами (именами), по которым к ним можно обратиться для получения текущего значения.

*Константы* – это данные, значения которых не изменяются в процессе работы программы. Они «узнаются» компьютером по форме их записи. В языке Pascal используются константы трех видов: числовые, булевские и символьные

*Числовые константы* предназначены для представления числовых данных (целых и вещественных). *Булевские константы* используются для представления данных, имеющих смысл логических высказываний (да – нет, истина – ложь). *Символьные константы* представляют данные, являющиеся последовательностями символов.

Значения *переменных*, в отличие от констант, могут изменяться во время выполнения программы. Переменные «узнаются» компьютером по именам (идентификаторам), более того с каждой переменной программы связывается одна и только одна ее характеристика, называемая *типом*.

*Тип* – это множество значений переменной вместе с множеством операций, которые можно выполнять над элементами этого множества.

Приписывая переменной некоторый тип, явно определяется множество значений, которые можно присвоить этой переменной, а также определяются операции, с помощью которых можно манипулировать ее значениями.

Число отдельных значений, принадлежащих некоторому типу, называется *мощностью типа*. Например, мощность логического типа – 2, так как переменные логического типа могут принимать только два значения True и False.

В языке программирования Pascal все типы данных принято разделять на три группы:

1. Простейшие, или скалярные (к скалярным типам данных относятся типы данных таких величин, значения которых не содержат составных частей, то есть простейшие типы: целочисленный, вещественный, логический, символьный, перечисляемый, интервальный (тип диапазон) и ссылочный).

2. Структурированные (структурированный тип данных, или структуры данных, в своей основе имеет один или несколько скалярных типов данных, к ним относятся: строки, массивы, множества, записи, файлы).

3. Процедурные и типа object (этим типам данных трудно поставить в соответствие данные в обычном понимании этого слова. Их названия отражают их базовые признаки и используются, как правило, в объектно-ориентированном программировании, которое поддержано старшими версиями языка Pascal. Понимание работы с этими типами данных требует определенного опыта и навыков программирования).

Рассмотрим некоторые скалярные (простейшие) типы данных.

### ***Целочисленные типы данных***

1. Диапазон значений типа *Byte*: 0..255. Над данными типа *Byte* допустимы следующие операции:

- сравнения (=, <, <=, >, >=);
- + (сложение); + (унарный плюс);
- – (вычитание); – (унарный минус);

- \* (умножение);
  - DIV (получение целой части частного при целочисленном делении);
  - MOD (получение остатка при целочисленном делении (второй операнд должен быть положительным));
2. Диапазон значений типа *Integer*: –32768..32767. Над данными этого типа допустимы те же операции, что и над данными типа *Byte*.
3. Диапазон значений типа *Word*: 0..65535. Над этим типом *Word* допустимы те же операции, что и над данными типа *Integer*.

### ***Вещественный тип данных***

Диапазон значений типа *Real*: 2.9E-39..1.7E+38. Данные этого типа имеют точность до одиннадцати значащих цифр. Над данными типа *Real* допустимы следующие операции:

- сравнения (=, <, <=, >, >=);
- + (сложение); + (унарный плюс);
- – (вычитание); – (унарный минус);
- \* (умножение); / (деление).

### ***Символьный тип данных***

Тип данных *Char* определяет множество символов с кодами ASCII. Переменные этого типа предназначены для хранения отдельных символов – букв, цифр и специальных знаков. Над данными типа *Char* допустимы операции сравнения (=, <, <=, >, >=).

### ***Логический тип данных***

Данные типа *Boolean* могут принимать два значения: *TRUE* и *FALSE*. Над данными типа *Boolean* допустимы следующие операции: сравнения (=, <, <=, >, >=);

AND (логическое И); OR (логическое ИЛИ);

XOR (логическое исключающее ИЛИ); NOT (логическое отрицание).

Заметим, что вам встретятся и другие простейшие типы данных, не включенные в данную классификацию.

## Процедуры ввода-вывода данных

Для организации ввода и вывода информации в языке программирования *Pascal* служат четыре процедуры: *Read* и *ReadLn*, *Write* и *WriteLn*. Эти процедуры являются «встроенными» в язык *Pascal*, поэтому для того, чтобы ввести информацию в оперативную память компьютера для ее дальнейшей обработки, программисту не надо писать специальную программу, а всего лишь нужно правильно обратиться к процедуре *Read* или *ReadLn*. Соответственно, чтобы вывести информацию на дисплей из оперативной памяти компьютера, необходимо вызвать процедуру *Write* или *WriteLn*.

### Процедура ввода информации

Процедуры ввода (или чтения информации) *Read* или *ReadLn* обеспечивают ввод числовых данных, символов, строк и т. д. для их последующей обработки программой.

*Синтаксис процедуры Read и ReadLn:*

```
Read (X,Y);  
ReadLn (X,Y);
```

где X, Y – переменные, описанные в разделе описания переменных.

Отметим, что указанные процедуры аналогичны, их единственное отличие заключается в том, что после ввода значений переменных, указанных в списке процедуры *ReadLn*, ввод значений переменных, указанных в списке следующей процедуры *ReadLn*, будет производиться с новой строки. Этого не происходит в ходе выполнения процедуры *Read*, то есть ввод осуществляется на той же строке. Таким образом, процедура *ReadLn* не только вводит значения указанных переменных, но еще и переводит курсор на новую строку. Это позволяет делать суффикс *Ln*.

Процедуры ввода в блок-схемах (рис. 2.2) реализуются геометрической фигурой – параллелограммом, внутри которого записываются переменные, которые вводятся, в этот блок входит и выходит одна стрелка.

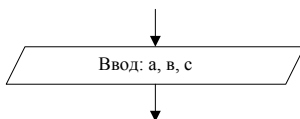


Рис. 2.2. Процедура ввода в блок-схемах



*Процедура вывода информации.* Процедуры вывода (или записи информации) *Write* или *WriteLn* обеспечивают вывод числовых данных, символов, строк на дисплей. В отличие от процедуры ввода, процедура вывода *WriteLn* может не иметь параметров. В этом случае данная процедура обеспечивает лишь перевод курсора на следующую строку.

Процедуры вывода в блок-схемах (рис. 2.3) реализуются геометрической фигурой – параллелограммом, внутри которого записываются переменные, которые выводятся, в этот блок входит и выходит одна стрелка.

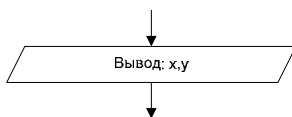


Рис. 2.3. Процедура вывода в блок-схемах

*Синтаксис процедуры Write и WriteLn:*

*Write* (X,Y); или *WriteLn* (X,Y); – «слепой» вывод значений переменных.

*Write* (X+Y); или *WriteLn* (X+Y); – вывод результата выражения.

*Write* (X:6:2); или *WriteLn* (X:6:2); – форматированный вывод значения переменной.

*Write* ('Фраза'); или *WriteLn* ('Фраза'); – вывод некоторой фразы на дисплей.

*Write* ('Фраза',Y); или *WriteLn* ('Фраза',Y); – вывод фразы и «неслепой» вывод значения переменной, где:

1. X, Y – переменные, значения которых нужно вывести на экран;
2. 'Фраза' – некоторый пояснительный текст, который обычно сопровождает значение переменной;
3. X:6:2 – указывается формат значения переменной. Это значит, что всего шесть символов отводится под значение переменной X, причем после десятичной точки будет выведено на экран два символа. Это так называемый форматированный вывод числа с фиксированной точкой.

### ***Оператор присваивания***

Синтаксис оператора присваивания: имя переменной := выражение.

Знак «:=» называется знаком присваивания. Оператор присваивания предписывает выполнить выражение, заданное в его правой

части, и присвоить результат переменной, имя которой указано в его левой части. Переменная и выражение должны быть совместимы по типу.

Примеры:

Y:=2; X:='Строка'; P:=V1; Summa:=V2,

где X, Y, P, Summa – имена переменных, описанных в разделе описания переменных;

2, 'Строка' – некоторые значения;

V1, V2 – выражения, значения которых нужно вычислить.

В блок-схемах (рис. 2.4) оператор присваивания изображается прямоугольником, внутри которого записана выполняемая операция, в него входит и выходит одна стрелка.

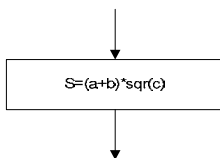


Рис. 2.4. Оператор присваивания в блок-схемах

### ***Пустой оператор***

Пустой оператор не обозначается и не вызывает никаких действий. Пустой оператор – это просто символ «;» (точка с запятой) в программе. Операторы отделяются друг от друга символом «точка с запятой» (;), поэтому ее часто называют *разделителем*. Если разделитель между двумя операторами отсутствует, то это приведет к возникновению ошибки, поскольку компилятор часто не может понять, что же хотел сказать автор программы. Пусть, например, записано

x:=1  
y:=2.

Несмотря на то, что операторы присваивания записаны на разных строках, компилятор будет воспринимать эту запись как

x:=1y:=2.

В итоге получается оператор, в котором используются два знака присваивания и неправильный идентификатор 1y (идентификатор не может начинаться с цифры).

### Примеры выполнения заданий

**Пример 1.** Составьте блок-схему (рис. 2.5) и напишите программу решения следующей задачи: выведите на экран дисплея предложение: «Это наша первая программа на языке Pascal!»

```
PROGRAM Primer_1;  
  BEGIN  
    WriteLn ('Это наша  
первая программа на языке  
Pascal!');  
  END.
```

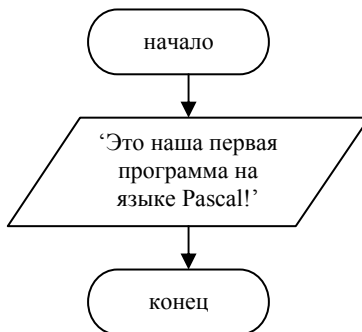


Рис. 2.5. Блок-схема к примеру 1

**Пример 2.** Напишите программу решения следующей задачи: демонстрация операций над данными типа *Word*.

```
PROGRAM Primer_2;  
  var x: Word; {Первый аргумент}  
      y: Word; {Второй аргумент}  
      z: Word; {Результат операции}  
  BEGIN  
    Write ('Введите значение переменной X типа Word...');  
    Read (x);  
    Write ('Введите значение переменной Y типа Word...');  
    Read (y);  
    z:=x+y; WriteLn ('x + y -> ',z);  
    z:=x-y; WriteLn ('x - y -> ',z);  
    z:=x*y; WriteLn ('x * y -> ',z);  
    z:=x DIV 2; WriteLn ('x DIV 2 -> ',z);  
    z:=x MOD 2; WriteLn ('x MOD 2 -> ',z);
```

```

WriteLn ('x=y  -> ',x=y);
WriteLn ('x<>y  -> ',x<>y);
WriteLn ('x>=y  -> ',x>=y)
END.

```

**Пример 3.** Составьте блок-схему (рис. 2.6) и напишите программу решения следующей задачи: демонстрация операций над типом *Real*. Операции с переменными типа *Real* реализуются программно, то есть математический сопроцессор в этом случае не используется.

```

PROGRAM Primer_3;
  var x: Real; {Первый аргумент}
      y: Real; {Второй аргумент}
      z: Real; {Результат операции}
BEGIN
  Write ('Введите значение
переменной X типа Real...');
  Read (x);
  Write ('Введите значение
переменной Y типа Real...');
  Read (y);
  z:=x+y; WriteLn ('x + y
=',z);
  z:=x-y; WriteLn ('x - y
=',z);
  z:=x*y; WriteLn ('x * y
=',z);
  z:=x/y; WriteLn ('x/y
=',z);
  WriteLn ('x=y      ',x=y);
  WriteLn ('x<>y    ',x<>y);
  WriteLn ('x>=y    ',x>=y)
END.

```

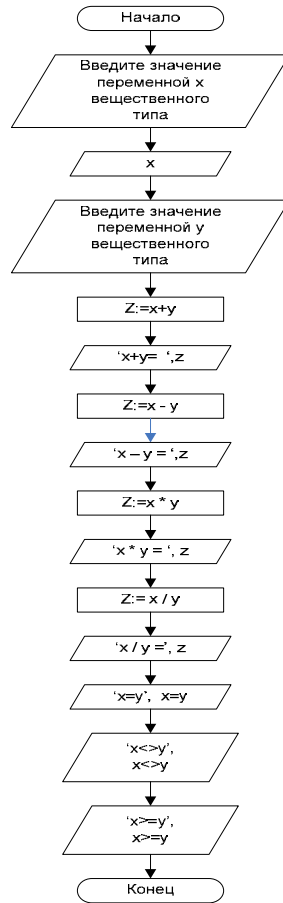


Рис. 2.6. Блок-схема к примеру 3

**Пример 4.** Напишите программу решения следующей задачи:  
демонстрация операций над типом *Integer*.

```
PROGRAM Primer_4;
  var x: Integer; {Первый аргумент}
      y: Integer; {Второй аргумент}
      z: Integer; {Результат операции}
BEGIN
  Write ('Введите значение переменной X типа Integer...');
  Read (x);
  Write ('Введите значение переменной Y типа Integer...');
  Read (y);
  z:=x+y; WriteLn ('x + y -> ',z);
  z:=x-y; WriteLn ('x - y -> ',z);
  z:=x*y; WriteLn ('x * y -> ',z);
  z:=x DIV 2; WriteLn ('x DIV 2 -> ',z);
  z:=x MOD 2; WriteLn ('x MOD 2 -> ',z);
  WriteLn ('x=y -> ', x=y);
  WriteLn ('x<>y -> ', x<>y);
  WriteLn ('x>=y -> ', x>=y)
END.
```

**Пример 5.** Напишите программу решения следующей задачи:  
демонстрация операций над данными типа *Byte*.

```
PROGRAM Primer_5;
  var x: Byte; {Первый аргумент}
      y: Byte; {Второй аргумент}
      z: Byte; {Результат операции}
BEGIN
  Write ('Введите значение переменной X типа Byte...');
  Read (x);
  Write ('Введите значение переменной Y типа Byte...');
  Read (y);
  z:=x+y; WriteLn ('x + y =',z);
  z:=x-y; WriteLn ('x - y =',z);
  z:=x*y; WriteLn ('x * y =',z);
  z:=x DIV 2; WriteLn ('x DIV 2 =', z);
  z:=x MOD 2; WriteLn ('x MOD 2 =', z);
```

```

WriteLn ('x=y   ',x=y);
WriteLn ('x<>y   ',x<>y);
WriteLn ('x>=y   ',x>=y)
END.

```

**Пример 6.** Напишите программу решения следующей задачи: демонстрация операций над типом *Char*.

```

PROGRAM Primer_6;
  var x,y: Char;
BEGIN
  Write ('Введите значение переменной X типа Char...');
  Read (x);
  Write ('Введите значение переменной Y типа Char...');
  Read (y);
  WriteLn ('x = ',x); WriteLn ('y = ',y);
  WriteLn ('x=y   ',x=y);
  WriteLn ('x<>y   ',x<>y);
  WriteLn ('x<y   ',x<y)
END.

```

**Пример 7.** Напишите программу решения следующей задачи: демонстрация операций над типом *Boolean*.

```

PROGRAM Primer_7;
  var x,y,z: Boolean;
BEGIN
  x:=TRUE; y:=FALSE; WriteLn ('x = ',x); WriteLn ('y = ',y);
  z:=NOT x; WriteLn ('NOT x  -> ',z);
  z:=x AND y; WriteLn ('x AND y -> ',z);
  z:=x OR y; WriteLn ('x OR y  -> ',z);
  WriteLn ('x=y   ',x=y);
  WriteLn ('x<>y   ',x<>y);
  WriteLn ('x>y   ',x>y)
END.

```

**Пример 8.** Напишите программу решения следующей задачи: демонстрация вычисления значений стандартных функций: *UpCase*, *Ord* и *Chr*.

```

PROGRAM Primer_8;
  var x: Char;
BEGIN
  Write ('Введите значение аргумента типа Char: ');
  ReadLn (x);
  WriteLn ('UpCase (x) ',UpCase (x));
  WriteLn ('Ord (x)  ',Ord (x));
  WriteLn ('Chr (65)  ',Chr (65))
END.

```

**Пример 9.** Напишите программу решения следующей задачи: демонстрация вычисления значений элементарных функций: Max-Int, Trunc(X), Int(X), Frac(X), Abs(X), Sqrt(X), Exp(X), Ln(X), Sin(X), Cos(X), ArcTan(X),  $X^Y$ , Odd(Z).

```

PROGRAM Primer_9;
(*Демонстрация вычисления значений элементарных функций*)
  var X: Real; {Аргумент}
      Y: Real; {Аргумент}
      Z: Integer; {Аргумент}
      F: Real; {Результат}
BEGIN
  WriteLn ('Значение MaxInt: ',MaxInt);
  WriteLn ('Значение Пи: ',PI);
  Write ('Введите значение аргумента: ');
  ReadLn (X);
  WriteLn ('Trunc (X)= ',Trunc (X));
  WriteLn ('Int (X)= ',Int (X));
  WriteLn ('Frac (X)= ',Frac (X));
  F:=Abs (X); WriteLn ('Значение функции Abs : ',F);
  F:=Sqrt (X); WriteLn ('Значение функции корень: ',F);
  F:=Exp (X); WriteLn ('Значение функции Exp : ',F);
  F:=Ln (X); WriteLn ('Значение функции Ln : ',F);
  {Аргумент тригонометрических функций выражен в радианах}
  F:=Sin (X); WriteLn ('Значение функции Sin : ',F);
  F:=Cos (X); WriteLn ('Значение функции Cos : ',F);
  F:=ArcTan (X); WriteLn ('Значение функции ArcTg: ',F);

```

```

{-----}
Write ('Введите значение аргумента: '); ReadLn (Y);
F:=Exp (Y*Ln(X));
WriteLn ('Значение функции X^Y  : ',F);
{-----}
Write ('Введите значение аргумента типа Integer: '); ReadLn (Z);
WriteLn ('Odd (Z)  =',Odd (Z))
END.

```

**Пример 10.** Составьте блок-схему (рис. 2.7) и напишите программу решения следующей задачи: дано положительное трехзначное целое число. Вывести цифры этого числа по одной в строке.

```

Program Primer_10;
Var x,a,b,c:Integer;
Begin
Writeln('Ввести положительное трехзначное число');
Readln(x);
a:=x mod 10; {последняя цифра числа}
b:=(x div 10) mod 10; {средняя цифра числа}
c:=x div 100; {первая цифра числа}
Writeln('первая цифра = ',c); {вывод первой цифры числа}
Writeln('вторая цифра = ',b); {вывод второй цифры числа}
Writeln('третья цифра = ',a); {вывод третьей цифры числа}
End.

```

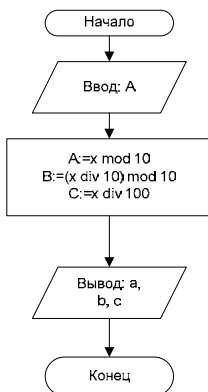


Рис. 2.7. Блок-схема к примеру 10



**Пример 11.** Напишите программу, используя оператор присваивания, в результате выполнения которого логическая переменная *T* принимает значение True, если выполняется указанное условие, и значение False в противном случае: а) числа *x*, *y*, *z* равны между собой; б) из чисел *x*, *y*, *z* только два равны между собой.

```
PROGRAM Primer_1;
  var x,y,z:word; t:boolean;
BEGIN
  writeln (введите x,y,z');  readln(x,y,z);
  {Для пункта а}
  t:=(x=y) and (y=z);  writeln ('числа x, y, z равны между собой ',t);
  {Для пункта б}
  t:=(x=y) or (x=z) or (y=z); writeln (' из чисел x, y, z только два
равны между собой ',t);
END.
```

### Контрольные вопросы

1. Из каких разделов состоит программа на языке Паскаль?
2. Для чего предназначен раздел описания? Из каких подразделов он состоит?
3. Что представляет собой раздел операторов?
4. Каким символом заканчивается программа на языке Паскаль? В каких случаях в программе не ставится «;»?
5. Какой символ является разделителем в Паскале?
6. Какова структура заголовка?
7. Чего требует хороший стиль программирования?
8. Что представляет собой константа?
9. Перечислите типы констант. Приведите примеры объявления констант различных типов.
10. Что представляет собой переменная? Примеры объявления переменных различных типов.
11. Какая характеристика связана с каждой переменной? Что такое мощность типа?
12. Приведите примеры объявления переменных целочисленных, логического, вещественного, символьного типа.
13. На какие группы делятся типы данных в Паскале?
14. Какие разделы могут отсутствовать в программе?

15. Охарактеризуйте тип Byte. Какие операции допустимы над данными этого типа? Почему при вычитании  $a - b$ , если  $a < b$ , получается неверный результат?

16. Охарактеризуйте тип Integer. Какие операции допустимы над данными этого типа?

17. Охарактеризуйте тип Word. Какие операции допустимы над данными этого типа? Почему при вычитании  $a - b$ , если  $a < b$ , получается неверный результат?

18. Охарактеризуйте тип Real. Какие операции допустимы над данными этого типа?

19. Охарактеризуйте тип Char. Какие операции допустимы над данными этого типа?

20. Охарактеризуйте тип Boolean. Какие операции допустимы над данными этого типа?

21. Приведите примеры встроенных констант. Чем они отличаются от пользовательских?

22. Для чего предназначена функция ODD? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

23. Для чего предназначены функции INT, FRAC? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

24. Для чего предназначены функции ROUND, TRUNC? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

25. Для чего предназначена функция ABS? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

26. Для чего предназначены функции ORD, CHR? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

27. Для чего предназначена функция UPCASE? Какого типа должен быть аргумент и результат какого типа возвращает эта функция?

28. Какая процедура предназначена для ввода информации? Ее синтаксис.

29. Какая процедура предназначена для вывода информации? Ее синтаксис.

30. Форматированный и неформатированный вывод значений на экран. Приведите примеры.

31. «Слепой» и «неслепой» вывод значений переменной на экран. Приведите примеры.
32. Организовать вывод на экран фразы «Защита лабораторной работы № 1» и значения выражения  $X*Y$ .
33. Оператор присваивания. Его синтаксис.
34. Пустой оператор. Его назначение.

### Задания для самостоятельного выполнения

1. В каждой задаче варианта составьте блок-схему и напишите программу решения задач.
2. Оформите отчет по лабораторной работе, который должен состоять из следующих разделов: номера лабораторной работы и ее названия, условия каждой задачи, блок-схемы и программы.

#### Вариант № 1

1. Стороны прямоугольника  $a$  и  $b$ . Найдите периметр  $p$ , площадь  $s$  и диагональ  $d$  этого прямоугольника.
2. Дано трехзначное целое число. Найдите цифры этого числа.
3. Известны координаты двух точек  $A(x_1, y_1)$  и  $B(x_2, y_2)$ . Найдите расстояния от начала координат до этих точек.
4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: данное натуральное число  $A$  кратно числу  $B$ , но не кратно числу  $C$ .

#### Вариант № 2

1. В треугольнике  $ABC$  известны длины сторон  $a, b, c$ . Вычислите длину высоты, проведенной из вершины  $A$ .

$$h = \frac{2\sqrt{p(p-a)(p-b)(p-c)}}{a}$$

2. Дано целое трехзначное число. Найдите сумму цифр этого числа.
3. Определите количество теплоты  $Q$ , требуемое на для нагревания  $V$ , л, жидкости, взятой при температуре  $T_1$ , до температуры кипения  $T_2$ , если известна удельная теплоемкость жидкости  $q$ .

4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: сумма двух действительных чисел  $a$  и  $b$  является целым числом, то есть дробная часть суммы равна нулю.

### Вариант № 3

1. В треугольнике  $ABC$  известны длины сторон  $a$ ,  $b$ ,  $c$ . Вычислить длину медианы проведенной из вершины  $A$ .

$$m = \frac{\sqrt{2b^2 + 2c^2 - a^2}}{2}.$$

2. Дано целое число  $x$ . Найдите остаток от деления этого числа на другое данное число  $y$ .

3. Определите расстояние, пройденное физическим телом за время  $t$ , если тело движется с постоянным ускорением  $a$  и в начальный момент времени имеет скорость  $V_0$ .

4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: произведение натуральных чисел  $a$  и  $b$  кратно числу  $c$ .

### Вариант № 4

1. Сутки на Марсе на 40 мин длиннее земных. Вычислите, на сколько часов марсианская неделя длиннее земной.

2. Дано трехзначное натуральное число  $A$ . Результат – двузначное число  $B$ , которое получается из  $A$  вычеркиванием средней цифры. Получите число  $B$ .

3. Определите время свободного падения физического тела с высоты  $H$ .

4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: сумма двух натуральных чисел кратна двум.

### Вариант № 5

1. Даны длины оснований трапеции  $x$ ,  $y$  и высота  $h$ . Найдите площадь этой трапеции.

2. Дано трехзначное целое число  $A$ . Получите из него двузначное целое число путем отбрасывания первой цифры.

3. Камень бросили в колодец и через  $t$ ,  $s$ , послышался всплеск воды. На какой глубине вода в колодце?

4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: данное четырехзначное число читается одинаково слева направо и справа налево.

### Вариант № 6

1. В треугольнике  $ABC$  известны длины сторон  $a$ ,  $b$ ,  $c$ . Вычислите длину биссектрисы, проведенной из вершины  $A$ .

$$l = \frac{2\sqrt{bc p(p-a)}}{b+c}.$$

2. Дано целое трехзначное число  $A$ . Получите из него двузначное путем вычеркивания последней цифры.

3. Вычислите напряжение на каждом из последовательно соединенных участков электрической цепи сопротивлением  $R_1$ ,  $R_2$ ,  $R_3$ , Ом, если сила тока при  $U$ , В, составляет  $I$ , А.

4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: число  $c$  является средним арифметическим чисел  $a$  и  $b$ .

### Вариант № 7

1. Дан радиус круга. Вычислите длину окружности и площадь данного круга.

2. Дано целое двухзначное число. Найдите сумму цифр этого числа.

3. Найти расстояние между двумя точками, если известны координаты этих точек  $A(x_1, y_1)$ ,  $B(x_2, y_2)$ .

4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: данные числа  $c$  и  $d$  являются соответственно квадратом и кубом числа  $a$ .

### Вариант № 8

1. Найдите целую и дробную часть данного вещественного числа  $A$ .
2. Найдите сумму цифр трехзначного натурального числа  $B$ .
3. Вычислите величину работы, совершенной при равномерном подъеме груза массой  $m$ , кг, на высоту  $h$ , м. Ускорение свободного падения опишите как константу  $g = 9,81$ .
4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: цифра  $M$  входит в десятичную запись четырехзначного числа  $N$ .

### Вариант № 9

1. Поменяйте местами значения переменных  $X$  и  $Y$ . Выполните задание двумя способами: с использованием промежуточной переменной и без нее.
2. Дано целое двузначное  $A$ . Образуйте новое двузначное число, поменяв цифры в числе  $A$  местами.
3. Вычислите значение силы тока  $I$  на участке, состоящем из двух параллельно соединенных резисторов сопротивлением  $R_1$  и  $R_2$ , если напряжение на концах этого участка равно  $U$ .
4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае:  $(x_1, y_1)$  и  $(x_2, y_2)$  – координаты левой верхней и правой нижней вершин прямоугольника; точка  $A(x, y)$  лежит внутри этого прямоугольника или на одной из его сторон.

### Вариант № 10

1. Вычислите объем шара радиусом  $r$ .
2. Дано трехзначное целое число  $A$ . Поменяйте в этом числе первую и вторую цифры.
3. Вычислите давление столбца жидкости плотностью  $R$  и высотой  $h$  на дно сосуда.
4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: данные числа  $x, y$  являются координатами точки, лежащей в первой координатной четверти.

### Вариант № 11

1. Вычислите периметр квадрата  $P$  по указанному значению его площади  $S$ .
2. Дано трехзначное целое число  $A$ . Поменяйте в этом числе первую и последнюю цифры.
3. Вычислите выталкивающую силу, действующую на тело объемом  $V$ , наполовину погруженное в жидкость плотностью  $R$ .
4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: данная тройка натуральных чисел  $a, b, c$  является тройкой Пифагора, то есть  $c^2 = a^2 + b^2$ .

### Вариант № 12

1. Коммерсант, имея стартовый капитал  $K$  рублей, занялся торговлей, которая ежемесячно увеличивает капитал на  $P, \%$ . Какая сумма будет у коммерсанта через полгода?
2. Дано целое трехзначное число  $A$ . Определите остаток от деления этого числа на другое данное число  $B$ .
3. Пусть смешаны  $V_1$ , л, воды температурой  $t_1$  с  $V_2$ , л, воды температуры  $t_2$ . Вычислите температуру и объем образовавшейся смеси.
4. Составьте программу, печатающую значение TRUE, если указанное высказывание является истинным, и FALSE – в противном случае: даны три стороны одного и три стороны другого треугольника. Эти треугольники равновеликие, то есть имеют равные площади.

## Лабораторная работа № 3

### ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ РАЗВЕТВЛЯЮЩЕЙСЯ СТРУКТУРЫ

*Цель работы:* овладение практическими навыками разработки и программирования разветвляющихся процессов с использованием условных операторов (If...then... else... и Case).

#### Теоретические сведения

##### *Оператор условного перехода If*

Операторы условного перехода позволяют выбрать для исполнения один из нескольких операторов-компонентов в зависимости от условия.

Синтаксис оператора **If**:

**If**  $A$     **then**  $ST_1$     **else**  $ST_2$ ,

где  $A$  – булевское выражение (возвращающее TRUE или FALSE);

$ST_1$ ,  $ST_2$  – операторы (возможно пустые или составные).

В блок-схемах оператор условного перехода реализуется следующей структурой (рис. 3.1).

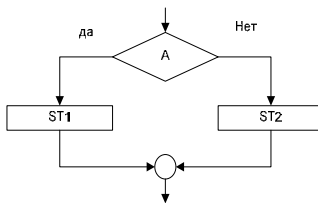


Рис. 3.1. Структура оператора условного перехода в блок-схемах

Если значение выражения  $A$  – TRUE, то выполняется оператор  $ST_1$ ; если значение выражения  $A$  – FALSE, то выполняется оператор  $ST_2$ . Затем в обоих случаях управление передается следующему оператору программы.



Допускается и краткая форма условного оператора:

**If A then ST;**

где  $A$  – булевское выражение;

$ST$  – оператор (возможно составной).

В блок-схемах (рис. 3.2) краткая форма оператора условного перехода реализуется структурой, изображенной на рис. 3.2.

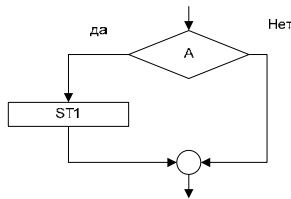


Рис. 3.2. Краткая форма оператора условного перехода в блок-схемах

Если значение  $A$  – TRUE, то выполняется оператор  $ST$ . Если значение  $A$  – FALSE, то управление передается оператору, следующему за оператором IF.

### ***Оператор выбора Case***

При написании программы не рекомендуется использовать многократно вложенные друг в друга условные операторы – программа становится громоздкой и трудно понимаемой. Считается, что число уровней вложения не должно превышать двух-трех. Но как быть, если необходимо проверять достаточно много условий и в зависимости от них выполнять те или иные действия? Для этих целей в языке ПаскальABC существует специальный оператор выбора **Case**.

Синтаксис оператора выбора **Case**:

**Case P of**

a:  $S_1$ ;

b:  $S_2$ ;

...

n:  $S_n$   
else  $S_{n+1}$   
end;

где **Case** («случай») – служебное слово;

$P$  – переменная или выражение, называемое селектором или переключателем;

$a, b$  – константы выбора;

$S_1, S_2, S_n, S_{n+1}$  – операторы, могут быть и составными.

В блок-схемах (рис. 3.3) оператор Case реализуется следующей структурой:

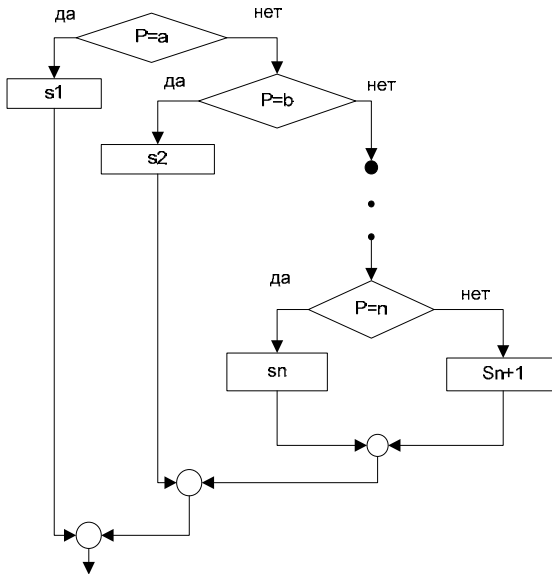


Рис. 3.3

При выполнении этого оператора сначала вычисляется некоторое выражение  $p$ , называемое селектором выбора, а затем в зависимости от полученного значения (если оно равно одной из констант  $a, b, \dots, n$ , которые называются константами выбора), выполняется один из операторов:  $S_1, S_2, \dots, S_n$ , помеченные соответствующей кон-

стантой. Если значение выражения  $p$  не совпадает ни с одной из констант выбора, выполняется оператор  $S_{n+1}$ , содержащийся после ключевого слова **else**, причем ветвь **else** в операторе **Case** необязательна. Используемые здесь зарезервированные слова **Case**, **of**, **else**, **end** имеют смысл **вариант**, **из**, **иначе**, **конец**.

Выражение, играющее роль селектора, должно принадлежать порядковому типу данных (то есть типу, имеющему конечное число значений). Например: *Integer*, *Boolean*, *Char*.

Кроме одиночных констант, в вариантах оператора **Case** могут использоваться диапазоны значений и списки. Например:

```
Case x of  
1..5: S1;  
2, 3, 8: S2;  
4, 6, 9..13: S3  
else S4  
end;
```

### *Составной оператор*

Составной оператор представляет собой группу из произвольного числа операторов, отделенных друг от друга точкой с запятой, и ограниченную операторными скобками – зарезервированными словами в языке ПаскальABC – **begin** и **end**:

```
begin S1; S2; ...; Sn end;
```

Поэтому приведенный оператор можно мысленно представить в таком виде:  $(S_1; S_2; \dots S_n)$ . Составной оператор используется в том случае, если необходимо выполнить последовательность операторов внутри какой-либо конструкции как единое целое.

### **Примеры выполнения заданий**

**Пример 1.** Составьте блок-схему (рис. 3.4) и напишите программу решения следующей задачи: определите большее из двух целых чисел  $X$  и  $Y$ .

```

PROGRAM Primer_1;
var X : Integer; {Пер-
вый аргумент}
    Y Integer;
    {Второй аргумент}
    Bolshee: Inte-
ger; {Результат}
BEGIN
    Write ('Введите пер-
вое число: ');
    ReadLn (X);
    Write ('Введите вто-
рое число: '); ReadLn
(Y);
    If X>=Y
        then Bolshee:=X
        else Bolshee:=Y;
    WriteLn ('Результат:
',Bolshee)
    END.

```

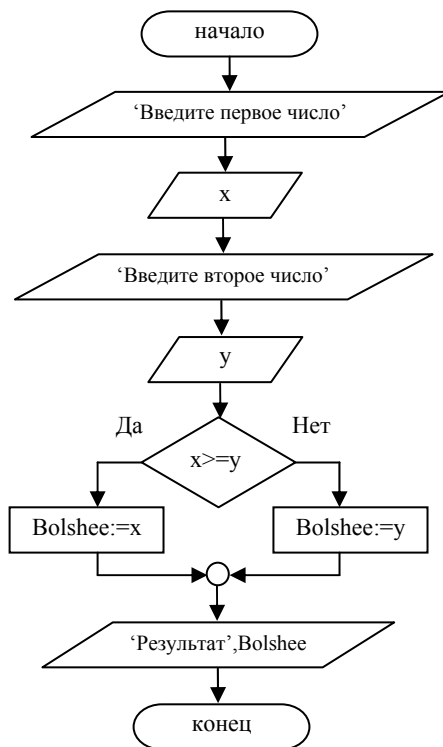


Рис. 3.4

**Пример 2.** Напишите программу решения следующей задачи: определите пору года по номеру месяца.

```

PROGRAM primer_2;
var mes:byte;
BEGIN
    writeln ('Введите номер месяца '); readln (mes);
    case mes of
        1,2,12: writeln ('Зима');
        3..5: writeln ('Весна');
        6..8: writeln ('Лето');
        9..11: writeln ('Осень');
    end;
END.

```

**Пример 3.** Напишите программу, которая определяет число дней в месяце, если даны: номер месяца  $N$  – целое число от 1 до 12; целое число  $A$ , равное 1, если год високосный, и 0 – в противном случае.

```
PROGRAM Primer_3;
  var n,a,k:Integer;
  BEGIN
    writeln ('Введите месяц');  readln(n);
    case n of
      1,3,5,7,8,10,12: begin k:=31; writeln (' в ', n,'  месяце ', k,'
дней'); end;
      4,6,9,11: begin k:=30; writeln (' в ', n,'  месяце ',k,' дней');end;
      2: begin
          writeln ('Если год високосный введите 1, иначе 0');
          readln(a);
          if a=1 then k:=29 else k:=28;
          writeln (' в ', n,'  месяце ',k,' дней');
          end
      else writeln ('Такого месяца нет');
    end;
  END.
```

**Пример 4.** Составьте блок-схему (рис. 3.5) и напишите программу, переводящую школьные отметки в оценки.

```
PROGRAM Primer_4;
  var BALL : Integer;
  BEGIN
    Write ('Введите величину отметки: ');
    Read (BALL);
    WriteLn;
    Case BALL of
      {Перевод отметки в оценку}
      1: WriteLn ('Очень плохо...');
      2: WriteLn ('Плохо...');
      3: WriteLn ('Удовлетворительно');
      4: WriteLn ('Хорошо!');
      5: WriteLn ('Отлично!')
    end;
  END.
```

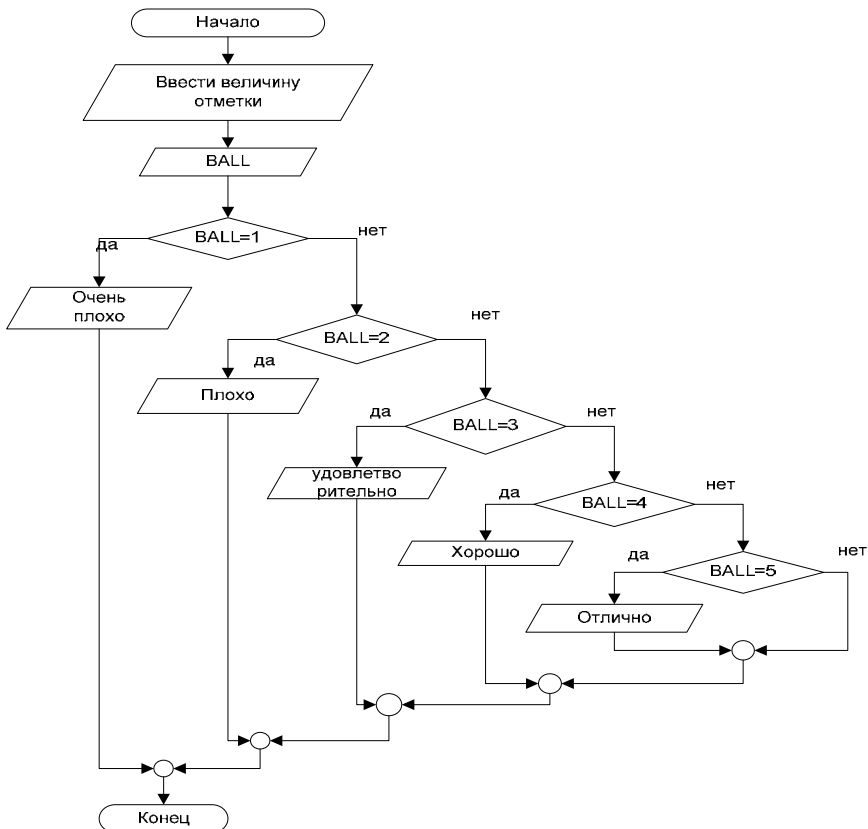


Рис. 3.5

### Контрольные вопросы

1. Перечислите операторы условного перехода.
2. Синтаксис оператора IF.
3. Как работает оператор IF (полная форма)?
4. Краткая форма оператора IF.
5. Как работает оператор IF (краткая форма)?
6. Что представляет собой составной оператор?
7. В каком случае используется составной оператор?
8. Что такое операторные скобки?

9. Сколько допускается уровней вложенности условных операторов?

10. В каком случае применяется оператор выбора Case?

11. Код программы выглядит следующим образом:

**Case P of**

a: S<sub>1</sub>;

b: S<sub>2</sub>;

...

n: S<sub>n</sub>

**else S<sub>n+1</sub>**

**end;**

поясните что обозначает:

Case – ...;

P – ...;

a, b, ..., n – ...;

S<sub>1</sub>, S<sub>2</sub>, S<sub>n</sub>, S<sub>n+1</sub> – ...

12. Запишите синтаксис оператора выбора.

13. Как работает оператор выбора?

14. Какому типу данных должен принадлежать селектор?

15. Что может использоваться в вариантах оператора выбора?

16. Какая ветвь в операторе Case может отсутствовать?

17. Какому оператору передается управление после окончания работы условных операторов?

18. Объясните, в чем разница между операторами слева и справа. Каково будет итоговое значение переменной X для каждого из них?

X:=1;

If X>=1 then X:=X+1

Else if X>=0 then X:=X+2;

X:=1;

If X>=1 then X:=X+1;

if X>=0 then X:=X+2;

19. Перечислите операции отношения.

20. Перечислите логические операции.

21. Каков результат применения логической операции TRUE and TRUE = ...?

22. Каков результат применения логической операции TRUE and FALSE = ...?

23. Каков результат применения логической операции FALSE and TRUE = ...?

24. Каков результат применения логической операции FALSE and FALSE = ...?

25. Каков результат применения логической операции TRUE or TRUE = ...?

26. Каков результат применения логической операции TRUE or FALSE = ...?

27. Каков результат применения логической операции FALSE or TRUE = ...?

28. Каков результат применения логической операции FALSE or FALSE = ...?

29. Каков результат применения логической операции not TRUE = ...?

30. Каков результат применения логической операции not FALSE = ...?

### **Задания для самостоятельного выполнения**

1. В каждой задаче составьте блок-схему и напишите программу решения задач.

2. Оформите отчет по лабораторной работе, который должен состоять из следующих разделов: номера лабораторной работы и ее названия, условия каждой задачи, ее блок-схемы и программы.

#### **Вариант № 1**

1. Вычислите значение выражения  $y = \sqrt{x-5}$  (используйте полную форму оператора IF).

2. На трех бензоколонках имеется  $a$ ,  $b$ ,  $c$  литров бензина. На какую бензоколонку вести новую партию бензина? (Используйте сложное логическое выражение и неполную форму оператора IF).

3. Дано трехзначное целое число  $A$ . Ответьте на вопрос, есть ли в записи этого числа цифра 5?

4. 1 января 2006 года было в воскресенье. Введите любую дату января 2006 года. Определите, на какой день недели эта дата приходится (используйте оператор Case).



### Вариант № 2

1. Вычислите значение выражения  $y = \sqrt{7-x}$  (используйте полную форму оператора IF).

2. В трех магазинах один и тот же товар имеет разные цены: в первом магазине  $a$  руб., во втором –  $b$  руб., в третьем –  $c$  руб. Определите в каком магазине товар самый дешевый (используйте сложное логическое выражение и неполную форму оператора IF).

3. Студент сдал четыре экзамена и получил соответственно  $a, b, c, d$  баллов по каждому из них. Ответьте на вопрос, средний балл студента меньше 8 или нет?

4. 31 декабря 2005 года было в субботу. Введите любую дату декабря 2005 года. Определите, на какой день недели эта дата приходится (используйте оператор Case)?

### Вариант № 3

1. Вычислите значение выражения  $y = \frac{x^2 + 4}{x - 6}$  (используйте полную форму оператора IF).

2. В трех магазинах один и тот же товар имеет разные цены: в первом магазине  $a$  руб., во втором –  $b$  руб., в третьем –  $c$  руб. Определите, в каком магазине товар самый дорогой (используйте сложное логическое выражение и неполную форму оператора IF).

3. Дано четырехзначное целое число  $A$ . Ответьте на вопрос, есть ли в записи этого числа цифра 0?

4. Введите номер месяца. Дайте этому месяцу наименование (используйте оператор Case).

### Вариант № 4

1. Вычислите значение выражения  $y = \sqrt{x+9}$  (используйте полную форму оператора IF).

2. Дан рост трех человек: первого –  $x$  см, второго –  $y$  см, третьего –  $z$  см. Определите самого высокого (используйте сложное логическое выражение и неполную форму оператора IF).

3. Дано трехзначное целое число  $A$ . Ответьте на вопрос, есть ли в записи этого числа четные цифры?

4. Введите номер месяца 2006 года. Определите, сколько дней в этом месяце (используйте оператор Case).

### Вариант № 5

1. Имеются два человека: у одного вес  $x$  пудов, у второго –  $y$  фунтов. Кто из них тяжелее? (используйте полную форму оператора IF).
2. Дан рост трех человек: первого –  $x$  см, второго –  $y$  см, третьего –  $z$  см. Определите самого низкого (используйте сложное логическое выражение и неполную форму оператора IF).
3. Дано трехзначное целое число  $A$ . Ответьте на вопрос, есть ли в записи этого числа нечетные цифры?
4. Введите номер месяца. Определите время года, которому он соответствует (используйте оператор Case).

### Вариант № 6

1. Известно, что 500 сажений = 3500 футов = 1066,8 м. С клавиатуры введите  $x$  сажений,  $y$  футов. Какое из этих значений меньше? (используйте полную форму оператора IF).
2. Дан вес трех человек: первого –  $x$  кг, второго –  $y$  кг, третьего –  $z$  кг. Определите самого легкого человека (используйте сложное логическое выражение и неполную форму оператора IF).
3. Дано четырехзначное целое число  $A$ . Ответьте на вопрос, превосходит ли сумма цифр этого числа данное число  $X$ .
4. Известно, что 2006 год – год Собаки. Введите номер года от 2000 до 2023. Определите, какой это год по китайскому календарю (используйте оператор Case). *Справка.* Знаки идут в следующем порядке: Крыса, Бык, Тигр, Кот, Дракон, Змея, Лошадь, Коза, Обезьяна, Петух, Собака, Свинья.

### Вариант № 7

1. Вычислить значение выражения  $y = \sqrt{\cos(x)}$  (используйте полную форму оператора IF).
2. Даны оценки по одному экзамену трех студентов: первый получил  $a$  баллов, второй –  $b$  баллов, третий –  $c$  баллов. Определите, кто сдал экзамен лучше всех (используйте сложное логическое выражение и неполную форму оператора IF).
3. Дано четырехзначное целое число  $A$ . Найдите сумму нечетных и сумму четных цифр этого числа. Ответьте на вопрос, какая из найденных сумм больше?
4. Имеется  $N$  корзин количеством от 30 до 50. Дайте этому количеству корзин правильное наименование (используйте оператор Case).

### Вариант № 8

1. Даны координаты точки  $A(x_1, y_1)$ . Определите, принадлежит ли эта точка 1 координатной четверти (используйте полную форму оператора IF).

2. Студент сдал четыре экзамена и получил соответственно  $a, b, c, d$  баллов по каждому из них. Какой предмет он сдал лучше других? (используйте сложное логическое выражение и неполную форму оператора IF).

3. Дано трехзначное целое число  $A$ . Ответьте на вопрос, равны ли первая и последняя цифра этого числа?

4. Имеется стадо из  $N$  слонов от 15 до 30 штук. Дать числу  $N$  правильное наименование от слова «слон» (используйте оператор Case).

### Вариант № 9

1. Даны координаты точки  $A(x_1, y_1)$ . Определите, принадлежит ли эта точка какой-нибудь оси координат (используйте полную форму оператора IF).

2. В магазине имеется три вида товара:  $a, b, c$ . Определите, какой вид товара необходимо завести в магазин (используйте сложное логическое выражение и неполную форму оператора IF).

3. Дано трехзначное целое число  $A$ . Ответьте на вопрос, равны ли последняя и средняя цифра этого числа?

4. В библиотеке одного человека имеется  $N$  книг от 20 до 50. Дайте этому количеству книг правильное наименование от слова «книга» (используйте оператор Case).

### Вариант № 10

1. Даны координаты точки  $A(x_1, y_1)$ . Определите, принадлежит ли эта точка 2 координатной четверти (используйте полную форму оператора IF).

2. В магазине имеется три вида товара:  $a, b, c$ . Определите, какой вида товара больше всего (используйте сложное логическое выражение и неполную форму оператора IF).

3. Дано трехзначное целое число  $A$ . Ответьте на вопрос, равны ли первая и средняя цифра этого числа?

4. 1 марта 2006 года – это день недели среда. Введите любую дату марта 2006 года. Определите день недели (используйте оператор Case).

### Вариант № 11

1. Даны координаты точки  $A(x_1, y_1)$ . Определите, принадлежит ли эта точка 3 координатной четверти (используйте полную форму оператора IF).

2. Студент сдал четыре экзамена и получил соответственно  $a, b, c, d$  баллов по каждому из них. Какой предмет он сдал хуже остальных? (используйте сложное логическое выражение и неполную форму оператора IF).

3. Дано трехзначное целое число  $A$ . Ответьте на вопрос, равны ли последняя и средняя цифра этого числа?

4. 1 мая 2006 года – понедельник. Введите любую дату месяца мая. Определите день недели (используйте оператор Case).

### Вариант № 12

1. Даны координаты точки  $A(x_1, y_1)$ . Определите, принадлежит ли эта точка какой-нибудь оси координат (используйте полную форму оператора IF).

2. Студент сдал четыре экзамена и получил соответственно  $a, b, c, d$  баллов по каждому из экзаменов. Какой предмет он сдал лучше остальных? (используйте сложное логическое выражение и неполную форму оператора IF).

3. Дано четырехзначное целое число  $A$ . Ответьте на вопрос, равны ли последняя и вторая цифра этого числа?

4. Имеется  $N$  парт количеством от 1 до 30. Дайте этому количеству парт правильное наименование (используйте оператор Case).

## ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЦИКЛИЧЕСКОЙ СТРУКТУРЫ

*Цель работы:* закрепить практические навыки работы с системой PascalABC, овладеть практическими навыками разработки, программирования циклических процессов, познакомиться с диагностическими сообщениями компилятора об ошибках при выполнении программ, содержащих операторы цикла While, Repeat и For.

### Теоретические сведения

Операторы цикла служат для того, чтобы выполнять входящие в их состав операторы (так называемые операторы тела цикла) несколько раз (в частном случае – один раз или ни разу). В языке PascalABC имеется три вида операторов цикла: **While**, **Repeat** и **For**.

#### *Оператор цикла While*

Синтаксис оператора цикла While:

**While** A **do** ST;

где A – булевское выражение (или условие);

ST – оператор (возможно составной).

Значение выражения A вычисляется перед каждым выполнением оператора ST, поэтому цикл While называют еще циклом с предусловием.

Если значение A есть TRUE, то оператор ST выполняется и управление передается на вычисление значения выражения A; если значение выражения A есть FALSE, то оператор ST не выполняется и происходит выход из цикла, то есть выполняется оператор, который записан в программе далее.

В блок-схеме (рис. 4.1) оператор While реализуется следующей структурой

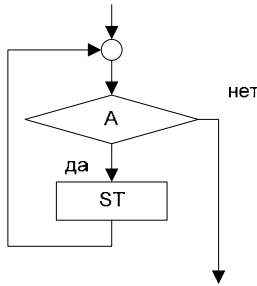


Рис. 4.1

*Замечания:*

1. Если первоначальное значение выражения  $A$  – FALSE, то оператор  $ST$  не будет выполнен ни разу.

2. В операторе цикла While точка с запятой никогда не ставится после зарезервированного слова do.

3. Чтобы избежать бесконечного повторения (зацикливания), необходимо хотя бы одну переменную, входящую в выражение  $A$ , изменять в теле оператора цикла. Более того, эти изменения должны быть такими, чтобы булевское выражение  $A$  рано или поздно приняло значение FALSE. Если же булевское выражение  $A$  первоначально истинно и ни при каких обстоятельствах не становится ложным, то выполнение оператора цикла никогда не завершится, происходит зацикливание.

Цикл While используется, как правило, в тех случаях, когда заранее неизвестно количество повторений цикла.

***Оператор цикла Repeat***

Цикл Repeat (цикл с постусловием), как правило, используется в тех случаях, когда заранее неизвестно количество повторений операторов тела цикла.

Синтаксис оператора цикла Repeat:

**Repeat  $ST$  until  $A$ ;**

где  $ST$  – оператор (возможно составной);

$A$  – булевское выражение (или условие).

Оператор работает следующим образом: выполняется оператор  $ST$ , вычисляется значение выражения  $A$ ; если его значение – FALSE, то вновь выполняется оператор  $ST$ , если значение выражения  $A$  – TRUE, то цикл заканчивается. Если же значение выражения  $A$  есть TRUE с самого начала, то оператор  $ST$  выполняется лишь один раз. Если выражение  $A$  никогда не принимает значение TRUE, то оператор  $ST$  выполняется бесконечное число раз, то есть происходит заикливание.

Следует отметить, что верхняя граница операторов  $ST$  обозначена словом Repeat, а нижняя – словом until, поэтому нет необходимости заключать составной оператор  $ST$  в операторные скобки begin – end.

Блок-схема оператора **Repeat**  $ST$  **until**  $A$ ; изображена на рис. 4.2.

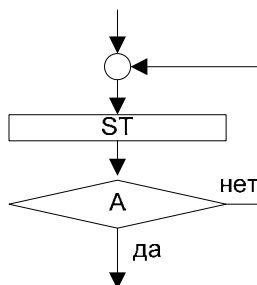


Рис. 4.2

### **Оператор цикла For**

Оператор цикла For служит для организации цикла с параметром и используется в тех случаях, когда заранее известно, сколько раз должна повторяться циклическая часть программы. Синтаксис оператора For по возрастающим значениям параметра  $i$ :

**For**  $i:=N_1$  **to**  $N_2$  **do**  $ST$ ,

где  $i$  – переменная ( параметр ) цикла;

$N_1$  – начальное значение параметра цикла;

$N_2$  – конечное;

$ST$  – оператор (возможно составной).

Переменные  $i$ ,  $N_1$ ,  $N_2$  должны быть одного и того же скалярного типа, но не типа Real, и значение  $N_2$  должно быть больше, чем значение  $N_1$ . Переменная  $i$  принимает последовательные значения данного типа от  $N_1$  до  $N_2$ . В частном случае, когда  $N_1$  и  $N_2$  – целые числа, а  $i$  – переменная типа Integer или Byte, шаг всегда равен единице.

Например, если

**For**  $i:=1$  **to** 20 **do**  $A:=A+1$ ,

то для  $i = 1, 2, 3, \dots, 20$  будет выполняться оператор  $A := A + 1$ .

Если же  $N_1$  и  $N_2$  символьного типа и имеют значения, например, 'A' и 'Z' соответственно, то переменная  $i$  принимает последовательные значения в порядке букв латинского алфавита: 'A', 'B', 'C', ..., 'Z'.

Блок-схема (рис. 4.3 и 4.4) оператора **For**  $i:=N_1$  **to**  $N_2$  **do** ST; может изображаться двумя способами

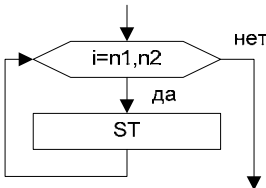


Рис. 4.3

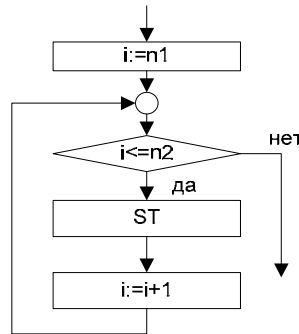


Рис. 4.4

Отметим, что есть возможность организовать цикл по убывающим значениям параметра  $i$ . Для этого используется зарезервированное слово **downto**.

Синтаксис оператора **For** по убывающим значениям параметра  $i$ :

**For**  $i:= N_2$  **downto**  $N_1$  **do** ST;

где  $i$  – переменная ( параметр ) цикла;

$N_2$  – начальное значение параметра цикла;



$N_1$  – конечное;

$ST$  – оператор (возможно составной).

В этом случае параметр  $i$  принимает последовательные убывающие значения данного типа от  $N_2$  до  $N_1$ . Например, если

**For**  $i:=20$  **downto** 1 **do**  $A:=A+1$ ,

то для  $i = 20, 19, 18, \dots, 1$  будет выполняться оператор  $A := A + 1$ .

Блок-схема (рис.4.5 и 4.6) оператора **For**  $i:=N_2$  **downto**  $N_1$  **do**  $ST$ ; может изображаться двумя способами.

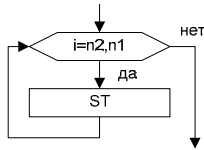


Рис. 4.5

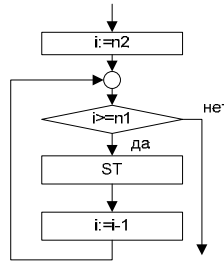


Рис. 4.6

### Примеры выполнения заданий

**Пример 1.** Составьте блок-схему и напишите программу решения следующей задачи: вычислите сумму  $S = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{50}$ .

а) с использованием цикла *For* (рис. 4.7 и 4.8):

```
PROGRAM Primer_1a;
var n: Integer;
s: Real;
BEGIN
S:=0;
For n:=1 to 50 do S:=S+1/n;
WriteLn ('Сумма равна: ',S)
END.
```

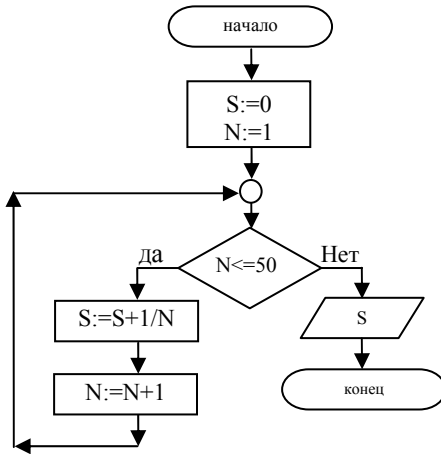


Рис. 4.7

Или в другом виде  
блок-схема

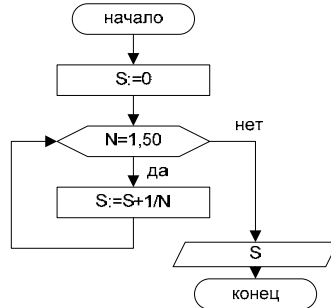


Рис. 4.8

б) С использованием цикла While (рис. 4.9):

```

PROGRAM Pri-
mer_1b;
  var S: Real;
      N: Integer;
BEGIN
  S:=0; N:=1;
  While N<=50 do
  begin
    S:=S+1/N;
    N:=N+1
  end;
  WriteLn ('Сумма
равна: ',S)
END.
  
```

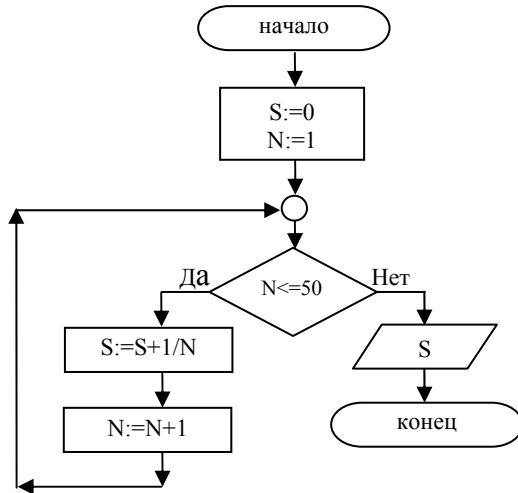


Рис. 4.9.

в) С использованием оператора цикла Repeat (рис. 4.10):

```

PROGRAM Primer_1c;
  var N: Integer;
  S: Real; { Результат суммирования }
BEGIN
  S:=0; N:=1;
  Repeat { Повторить... }
  S:=S+1/N;
  N:=N+1
  until N>50; { пока значение N не превзойдет 50 }
  WriteLn ('Результат суммирования... ',S)
END.

```

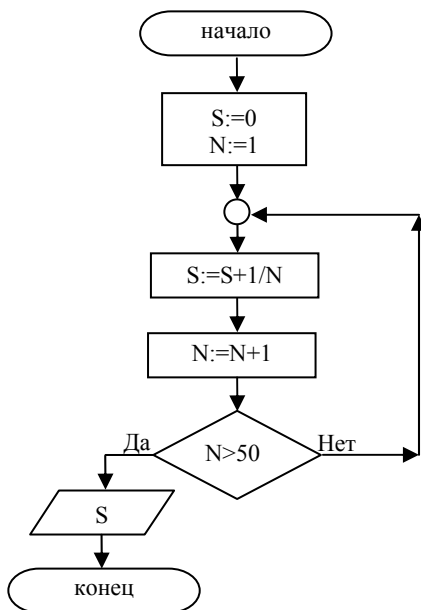


Рис. 4.10

**Пример 2.** Напишите программу, моделирующую микрокалькулятор. После нажатия знака « $\Rightarrow$ » на экране отображается значение выражения.

```

PROGRAM Primer_2;
  var Operacij, Operator: Char; { Символ арифметической операции }
  n,m : Real;
  Answer : Real; { Результат вычисления }
BEGIN
  Answer:=0; { Вначале обнулیم результат... }
  WriteLn ('Введите арифметическое выражение');
  WriteLn (('каждую часть выражения – с новой строки, например:)...');
  Writeln(('первое слагаемое – одна часть'));
  Writeln(('знак "+" – вторая часть'));
  Writeln(('второе слагаемое – третья часть'));
  Writeln(('знак "=" – четвертая часть'));

```

```

Repeat
  ReadLn (n); {Вводимый первый операнд слагаемое}
  ReadLn (Operacij); {Знак вводимой операции}
  ReadLn (m); {Вводимый второй операнд}
  Case Operacij of
    '+': Answer:=n+m;
    '-': Answer:=n-m;
    '*': Answer:=n*m;
    '/': Answer:=n/m
  end;
  ReadLn(Operator);
until Operator='=';
WriteLn (Answer:10:4)
END.

```

**Пример 3.** Напишите программу решения следующей задачи: пусть  $A$  и  $B$  – положительные вещественные числа и  $A > B$ . Найдите такое натуральное  $m$ , что  $m * B > A$ .

```

PROGRAM Primer_3;
  var A,B: Real;      m : Integer;
BEGIN
  Write ('Введите числа A и B... ');
  Read (A);
  Write (' ');
  ReadLn (B); m:=1;
  While m*B<=A do m:=m+1;
  WriteLn ('Результат... ',m:3,'*',B:3:2,' > ',A:6:2)
END.

```

**Пример 4.** Напишите программу решения следующей задачи: определение наименьшего положительного числа Eps такого, что  $1 + Eps > 1$ .

```

PROGRAM Primer_4;
  const C1 = 1.0;      C2 = 2.0;
  var Eps,Eps1: Real;
BEGIN
  Eps:=C1; Eps1:=C2;

```

```

While Eps1>1.0 do begin Eps:=Eps/2; Eps1:=Eps+1 end;
Eps:=Eps*2;
WriteLn ('Приближенное значение Eps равно: ',Eps)
END.

```

**Пример 5.** Напишите программу решения следующей задачи: разложите целое число на простые множители.

```

PROGRAM Primer_5;
var x,m: Integer;
BEGIN
Write ('Введите целое число... '); ReadLn (x);
WriteLn ('Разложение числа ',x,' на простые множители');
m:=2;
While m<=x do
If (x MOD m)=0
then begin Write (' * ',m); x:=x DIV m end
else m:=m+1;
END.

```

**Пример 6.** Напишите программу, позволяющую угадать целое число из отрезка [1, 100].

```

PROGRAM Primer_6;
var x,z : Integer; ugalad: Boolean;
BEGIN
WriteLn ('Угадай число'); WriteLn ('от 1 до 100');
Randomize; z:=Random (100)+1; ugalad:=FALSE;
While NOT ugalad do
begin
Write ('число='); Read (x);
If x=z
then begin ugalad:=TRUE; WriteLn ('Правильно') end
else If x<z
then WriteLn ('Мало...')
else If x>z
then WriteLn ('Много...')
end
END.

```

**Пример 7.** Напишите программу следующей задачи: вычислите целую часть квадратного корня из положительного числа. Идея алгоритма заключается в том, что сумма  $K$  первых нечетных чисел равна, например,  $1 + 3 = 2^2$ ,  $1 + 3 + 5 = 3^2$  и т. д.

```
PROGRAM Primer_7;
  var i,j: Integer;
      x : Real; { Результат }
BEGIN
  Write ('Введите положительное число, из которого ');
  Write ('хотите извлечь корень: '); ReadLn (x);
  i:=-1; j:=0;
  While j<=x do
  begin i:=i+2; j:=j+i end;
  Write ('Результат: ',(i-1) DIV 2)
END.
```

**Пример 8.** Напишите программу решения следующей задачи: найдите все простые числа на заданном отрезке (используйте цикл с параметром)

```
PROGRAM Primer_8;
  var M : Integer; { Нижняя граница отрезка }
      N : Integer; { Верхняя граница отрезка }
      i,j: Integer; { Параметры циклов }
      kl : Integer;
BEGIN
  Write ('Введите нижнюю границу отрезка... '); ReadLn (M);
  Write ('Введите верхнюю границу отрезка... '); ReadLn (N);
  WriteLn ('Все простые числа из отрезка [' ,M,',',N,']');
  For i:=M to N do
    begin kl:=0;
      For j:=2 to Round (Sqrt(i)) do
        If (i MOD j)=0 then kl:=kl+1;
        If kl=0 then Write (i, ' ')
      end
    end
END.
```

## Контрольные вопросы

1. Для чего предназначены операторы цикла?
2. Виды операторов цикла.
3. Синтаксис оператора цикла с предусловием.
4. Каково минимальное количество повторений цикла WHILE?
5. Почему в операторе цикла WHILE после DO не ставится «;»?
6. Как работает оператор цикла WHILE?
7. Как избежать заикливания при организации цикла WHILE?
8. В каких случаях используется цикл WHILE?
9. В каком случае произойдет заикливание оператора WHILE?
10. В каком случае тело цикла WHILE заключается в операторные скобки?
11. В каких случаях используется цикл REPEAT?
12. Синтаксис оператора цикла с постусловием.
13. Как работает оператор цикла REPEAT?
14. Почему тело цикла REPEAT не заключается в операторные скобки?
15. В каком случае произойдет заикливание оператора REPEAT?
16. Как избежать заикливания при организации цикла REPEAT?
17. Каково минимальное количество повторений цикла REPEAT?
18. Синтаксис оператора цикла с параметром по возрастающим значениям параметра.
19. Синтаксис оператора цикла с параметром по убывающим значениям параметра.
20. В каком случае тело цикла FOR заключается в операторные скобки?
21. В каких случаях используется цикл FOR?
22. **While**  $A$  **do**  $ST$ ; Поясните, что такое  $A$  и  $ST$ ?
23. **Repeat**  $ST$  **until**  $A$ ; Поясните что такое  $ST$  и  $A$ ?
24. **For**  $i:=N_1$  **to**  $N_2$  **do**  $ST$ ; Поясните, что такое  $i$ ,  $N_1$ ,  $N_2$ ,  $ST$ ?
25. **For**  $i:=N_1$  **to**  $N_2$  **do**  $ST$ ; К какому типу данных должны принадлежать переменные  $i$ ,  $N_1$ ,  $N_2$ .
26. Верно ли утверждение, что в операторе цикла с параметром по возрастающей начальное значение параметра должно быть меньше конечного?
27. Какому значению равен шаг изменения параметра в цикле **For**  $i:=N_1$  **to**  $N_2$  **do**  $ST$ , если  $i$ ,  $N_1$ ,  $N_2$  – целые числа?

28. Какому значению равен шаг изменения параметра в цикле **For**  $i:=N_1$  **downto**  $N_2$  **do** ST, если  $i$ ,  $N_1$ ,  $N_2$  – целые числа?

29. Как работает оператор цикла с параметром (по возрастающей)?

30. Как работает оператор цикла с параметром (по убывающей)?

31. **For**  $i:=N_1$  **downto**  $N_2$  **do** ST. Какое соотношение верно:  $N_1 > N_2$  или  $N_1 < N_2$ .

32. Из какого количества операторов состоит тело цикла в данном фрагменте программы? Как называется этот цикл?

```
...S:=0;
For i:=1 to 50 do S:=S+1/i;
WriteLn ('Сумма равна: ',S); ...
```

33. Из какого количества операторов состоит тело цикла в данном фрагменте программы? Как называется этот цикл?

```
...S:=0; N:=1;
While N<=50 do
begin S:=S+1/N; N:=N+1 end;
WriteLn ('Сумма равна: ',S)
```

34. Из какого количества операторов состоит тело цикла в данном фрагменте программы? Как называется этот цикл?

```
...ReadLn (B); m:=1;
While m*B<=A do m:=m+1;
WriteLn ('Результат... ',m:3,'*',B:3:2,' > ',A:6:2)
```

35. Из какого количества операторов состоит тело цикла в данном фрагменте программы? Как называется этот цикл?

```
...S:=0; N:=1;
Repeat
S:=S+1/N; N:=N+1
until N>50;
WriteLn ('Результат суммирования... ',S); ...
```



36. Каково условие выхода из цикла в данном фрагменте программы?

```
...While Eps1>1.0 do  
    begin Eps:=Eps/2; Eps1:=Eps+1 end; ...
```

37. Каково условие повторения тела цикла в данном фрагменте программы?

```
...Repeat  
    S:=S+1/N; N:=N+1  
until N>50; ...
```

38. Какому оператору будет передано управление в данном фрагменте программы, если условие  $j \leq x$  станет ложным?

```
...While j<=x do  
    begin i:=i+2; j:=j+i end;  
    Write ('Результат: ',(i-1) DIV 2)  
End.
```

39. Сколько раз выполниться тело цикла в данном фрагменте программы?

```
For i:=1 to 50 do S:=S+1/i;
```

### **Задания для самостоятельного выполнения**

1. В каждой задаче составьте блок-схему и напишите программу решения задач.

2. Оформите отчет по лабораторной работе, который должен состоять из следующих разделов: номера лабораторной работы и ее названия, условия каждой задачи, ее блок-схемы и программы.

3. Для задачи 1 использовать оператор цикла с постусловием, для задачи 2 – оператор цикла с предусловием, для задачи 3 – оператор цикла с параметром, для задачи 4 любой оператор цикла.

### Вариант № 1

1. Составьте таблицу значений функции  $y = x^2 - 2x + 5$  на отрезке  $[a; b]$  с шагом  $h$ .

2. Сумма в  $A$ , руб., лежит на расчетном счете. На сумму начисляется в % каждый год прибыли на ту сумму, которая находится на счете. Прибыль со счета не снимается. Через сколько лет данная сумма удвоится?

3. Дан натуральный отрезок  $[a; b]$ . Выведите из него все четные числа.

4. В каких двузначных числах удвоенная сумма цифр равна их произведению?

### Вариант № 2

1. Составьте таблицу значений функции  $y = (x - 2)/(x^2 + 4)$  на отрезке  $[a; b]$  с шагом  $h$ .

2. Спортсмен марафонец, готовясь к соревнованиям в первый день тренировки пробежал  $A$  км. Каждый следующий день он наращивал норму тренировки на  $B$  % от предыдущего дня. В какой день тренировок норма его пробега превысит  $C$  км?

3. Дан целочисленный отрезок  $[a; b]$ . Выведите из него все отрицательные числа.

4. Найдите двузначное число, равное утроенному произведению его цифр.

### Вариант № 3

1. Составьте таблицу значений функции  $y = x^3 + 2x^2 - 4x + 7$  на отрезке  $[a; b]$  с шагом  $h$ .

2. В 1626 г. индейцы продали остров Манхеттен за 20 долларов. Если бы эти деньги были помещены в банк под 5 % годовых, то сколько бы денег было на счету в 2010 г.? *Справка.* Проценты со счета не снимать. Они начисляются каждый год на ту сумму, которая находится на счету.

3. Дан натуральный отрезок  $[a; b]$ . Выведите из него все числа, которые делятся на данное число  $X$ .

4. Найдите все двузначные числа, сумма квадратов цифр которых делится на 17.

### Вариант № 4

1. Составьте таблицу значений функции  $y = \sin(x) - \cos(x)$  на отрезке  $[a; b]$  с шагом  $h$ .
2. Спортсмен-марафонец, готовясь к соревнованиям, в первый день тренировки пробежал  $A$  км. Каждый следующий день он наращивал норму тренировки на  $B$  % от предыдущего дня. В какой день тренировок общий пробег за все дни превысит  $C$  км?
3. Дан натуральный отрезок  $[a; b]$ . Выведите из него все нечетные числа.
4. Найдите двузначное число, обладающее тем свойством, что куб суммы его цифр равен квадрату самого числа.

### Вариант № 5

1. Составьте таблицу значений функции  $y = (x + 2)^3$  на отрезке  $[a; b]$  с шагом  $h$ .
2. Сумма в  $A$  руб. лежит на расчетном счету. На сумму начисляется  $B$  % каждый год прибыли. Прибыль со счета не снимается. Через сколько лет данная сумма превысит  $C$ , руб.?
3. Найдите сумму всех нечетных двузначных чисел.
4. Найдите все трехзначные числа, которые можно представить разностью между квадратом числа, образованного первыми двумя цифрами и квадратом третьей цифры.

### Вариант № 6

1. Составьте таблицу значений функции  $y = x^4 - 2x + 8$  на отрезке  $[a; b]$  с шагом  $h$ .
2. Количество граждан некоторого города увеличивается ежегодно на  $B$  %. Через сколько лет население города возрастет в два раза, если вначале было  $A$  человек.
3. Найдите сумму всех четных двузначных чисел.
4. Найдите все трехзначные числа, средняя цифра которых равна сумме первой и третьей цифр.

### Вариант № 7

1. Напечатайте таблицу соответствия между расстоянием в футах, ярдах и метрах, если известно, что  $1 \text{ фут} = 0,305 \text{ м}$ ,  $1 \text{ ярд} = 0,914 \text{ м}$ , для значений от 1 до 10 км с шагом 0,5.

2. Мячик упал с высоты  $P$ . Ударился о землю и поднялся на  $2/3$  предыдущей высоты. Через сколько ударов мячик поднимется на высоту  $H$ ?

3. Числа Фибоначчи вычисляются по следующему закону:  $a_1 = 1, a_2 = 1, a_n = a_{n-2} + a_{n-1}$ . Найдите  $k$ -е число Фибоначчи.

4. Подсчитайте  $k$  – количество цифр в десятичной записи целого неотрицательного числа  $N$ .

### Вариант № 8

1. Составьте таблицу значений функции  $y = e^x + 6x - 3$  на отрезке  $[a; b]$  с шагом  $h$ .

2. Гриб за сутки увеличивает свою массу на 40 %. Через сколько суток масса гриба увеличится в 2,5 раза, если первоначально его масса составляла  $A$ ?

3. Найдите сумму всех двузначных чисел кратных 6.

4. Найдите все трехзначные числа, сумма цифр которых равна данному целому числу.

### Вариант № 9

1. Напечатайте таблицу значений  $y = \sin x, y = \cos x, y = \operatorname{tg} x$ , если  $x$  меняется от 1 до 5 с шагом 0,25.

2. Дано целое число  $A$ . Это число умножают на 2 и к произведению прибавляют 1. Полученное число опять умножают на 2 и прибавляют 1. Сколько раз необходимо умножить на 2 и прибавить единицу к данному числу, чтобы получилось число больше  $Y$ .

3. Спортсмен, готовясь к соревнованиям по бегу на длинные дистанции, в первый день тренировок пробежал  $A$  км. Каждый следующий день он увеличивал норму пробега на  $B$  %. Какая норма пробега будет у спортсмена в  $k$ -й день тренировок?

4. Даны натуральные числа  $n$  и  $k$ . Проверьте, есть ли в записи числа  $x = nk$  цифра  $m$ .

### Вариант № 10

1. Напечатайте таблицу соответствия между весом в фунтах, килограммах и граммах для значений от 1 до 15 фунтов с шагом 0,5 фунта (1 фунт = 453 г.).

2. Числа Фибоначчи определяются по следующему закону:  $a_1 = 1$ ,  $a_2 = 1$ ,  $a_{n+1} = a_n + a_{n-1}$ . Определите член Фибоначчи, который больше заданного числа  $X$ .

3. Дан натуральный отрезок  $[a; b]$ . Выведите из него все числа, кратные 7.

4. В каких трехзначных числах удвоенная сумма цифр равна их произведению?

### Вариант № 11

1. Около стены наклонно стоит палка длиной  $X$ . Один ее конец находится на расстоянии  $Y$  от стены. Определите значение угла между палкой и полом для  $Y$ , изменяющегося от 2 до 4 с шагом 0,2 м.

2. Числа Фибоначчи определяются по следующему закону:  $a_1 = 1$ ,  $a_2 = 1$ ,  $a_{n+1} = a_n + a_{n-1}$ . Суммируйте члены Фибоначчи, идущие подряд, до тех пор, пока сумма не станет больше заданного числа  $X$ .

3. Дан натуральный отрезок  $[a; b]$ . Выведите из него все числа, меньшие данного числа  $X$ .

4. Найдите все двузначные числа, сумма цифр которых делится на 9.

### Вариант № 12

1. Плотность воздуха убывает с высотой по закону  $\rho = \rho_0 e^{-hz}$ . Считая, что  $\rho_0 = 1,29 \text{ кг/м}^3$ ,  $z = 1,25 \cdot 10^{-4} \text{ 1/м}$ , определите зависимости плотности воздуха от высоты для значений от 0 до 1000 м с шагом 100 м. Результаты представьте в виде таблицы.

2. Числа Фибоначчи определяются по следующему закону:  $a_1 = 1$ ,  $a_2 = 1$ ,  $a_{n+1} = a_n + a_{n-1}$ . Между какими членами Фибоначчи находится данное число  $K$ ?

3. Найдите сумму всех четырехзначных чисел, кратных  $k$ .

4. Найдите все трехзначные числа, третья цифра которых равна сумме первой и второй цифр.

## ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ С ИСПОЛЬЗОВАНИЕМ ВЛОЖЕННЫХ ЦИКЛОВ

*Цель работы:* овладеть практическими навыками разработки программ с использованием вложенных циклов.

### Теоретические сведения

Если тело какого-то цикла содержит другую циклическую структуру, то такие циклы называются *вложенными*. Цикл, который содержит другой цикл, называют *внешним*. Цикл, который содержится в теле другого цикла, называют *внутренним*.

Внешний и внутренний циклы могут быть любыми из трех видов: цикл с постусловием Repeat, цикл с предусловием While и цикл с параметром For.

Правила организации внешнего и внутреннего циклов такие же, как и для простых циклов каждого вида. Но при программировании вложенных циклов необходимо соблюдать следующее дополнительное условие: все операторы внутреннего цикла должны полностью располагаться в теле внешнего цикла. Так же следует помнить, что имена параметров внешнего и внутреннего циклов должны быть разными.

Каждое повторение внешнего цикла означает завершение всех внутренних циклов. При этом всем выражениям, которые управляют внутренними циклами, вновь присваиваются начальные значения.

Глубина вложенности циклов друг в друга не определена, в основном зависит от мощности компьютера.

### Примеры выполнения заданий

**Пример 1.** Составьте блок-схему (рис. 5.1) и напишите программу вывода на экран таблицы умножения.

```
Program Primer_1;  
Var a,I,J:Integer;  
BEGIN
```

```

For I:=1 to 9 do
Begin
  For J:=1 to 9 do
  Begin
    A:=i*J;
    Write(I,'*',J,'=',A:2,' ');
  End;
  Writeln; {Перевод курсора
на новую строку}
End;
END.

```

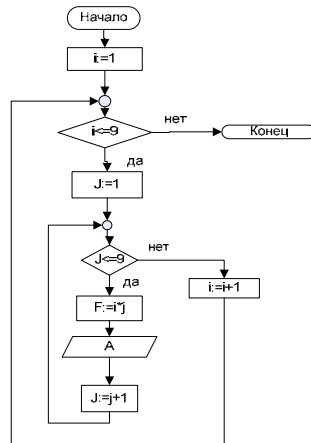


Рис. 5.1

**Пример 2.** Составьте блок-схему (рис. 5.2) и напишите программу решения следующей задачи: из отрезка  $[a, b]$  выведите числа, которые являются степенью числа 6.

```

Program Primer_2;
Var a,b,p,x,k:Integer;
BEGIN
Writeln('Ввести концы отрезка');
Readln(a,b);
If a>b Then begin x:=a; a:=b; b:=x; end; {Если начало отрезка введено больше конца}
k:=0; {Счетчик искомых чисел}
For x:=a to b do
  Begin
    p:=1; {Для нахождения произведения}
    While p<x do p:=p*6; {Пока произведение меньше числа x}
    If p=x {После выполнения цикла: если p=x, то}
      Then Begin
        Writeln (p); {Выводим найденное число}
        k:=k+1; {и считаем, что одно число уже вывели}
      End;
  End;
End;

```

If  $k=0$  {если счетчик искоемых чисел остался равен 0, то нужных чисел нет}

Then Writeln('Степеней числа 6 на данном отрезке нет');

End.

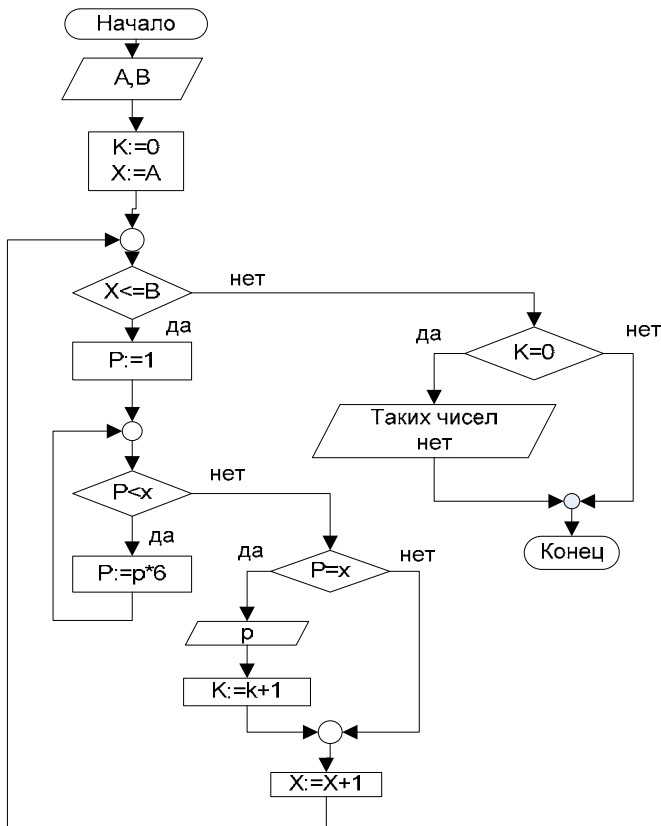


Рис. 5.2

**Пример 3.** Составьте блок-схему (рис. 5.3) и напишите программу подсчета количества чисел палиндромов, лежащих в промежутке от 100 до 1000 (число  $M$  называется палиндромом, если оно равно своему обращенному (то есть число читается слева направо и справа налево одинаково, например, 383).



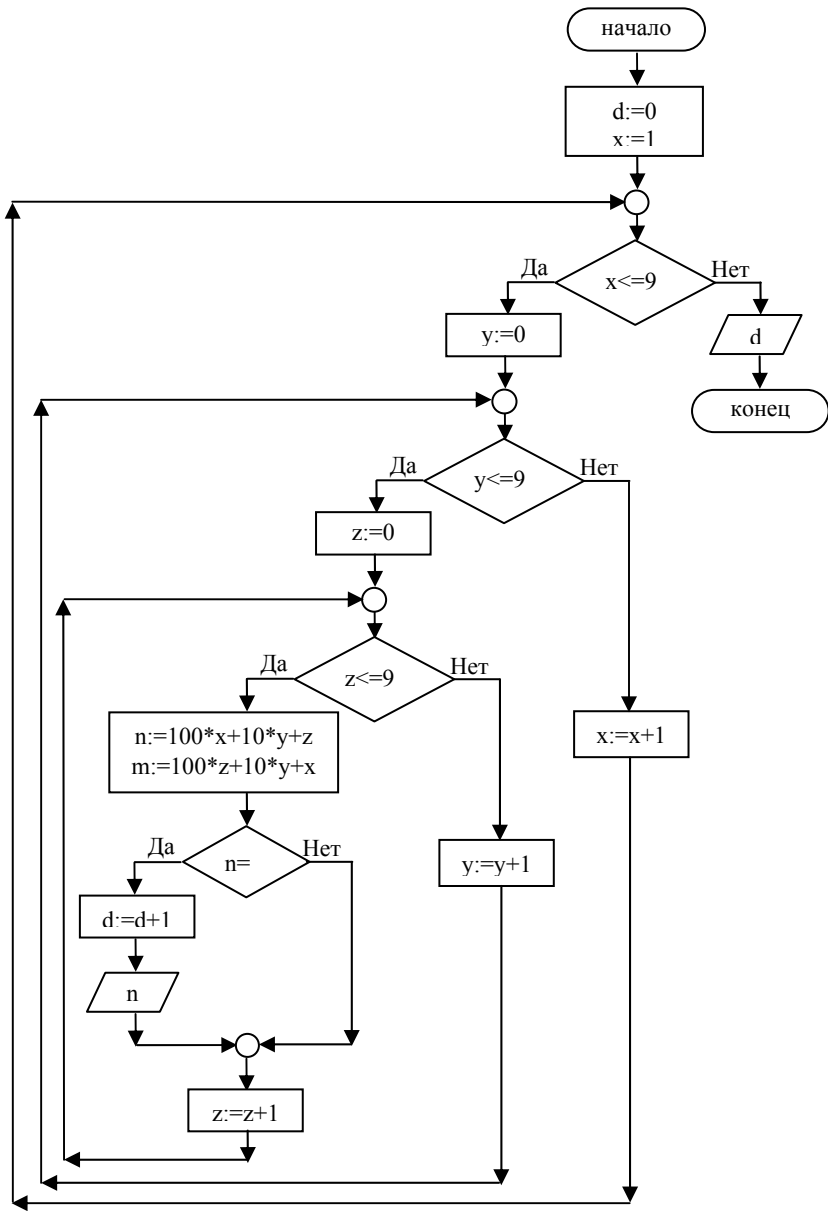


Рис. 5.3

```

PROGRAM Primer_3;
  var d :Integer; { Счетчик палиндромических чисел }
      x,y,z: Integer; { Параметры циклов: это цифры трехзначного
числа }
      n :Integer; { Трехзначное число образованное цифрами x, y, z }
      m :Integer; { Число, образованное теми же цифрами в обрат-
ном порядке }
  BEGIN
    d:=0;
    For x:=1 to 9 do
      For y:=0 to 9 do
        For z:=0 to 9 do
          begin
            n:=100*x+10*y+z; {Трехзначное число, образованное цифрами
x,y,z}
            m:=100*z+10*y+x; {Число, образованное теми же цифрами
в обратном порядке}
            if n=m then begin d:=d+1; Write (n,' ') end;
            end;
            WriteLn; WriteLn ('Результат: ',d)
          end;
        end;
      end;
    END.

```

### **Контрольные вопросы**

1. Какие циклы называются вложенными?
2. Какой цикл является внешним?
3. Какой цикл является внутренним?
4. Какой цикл повторится большее количество раз: внешний или внутренний?
5. Перечислите правила организации вложенных циклов.
6. Какие циклы можно использовать при составлении программ с вложенными циклами?
7. Какая глубина вложенности циклов доступна для ПаскальABC?

## Задания для самостоятельного выполнения

1. В каждой задаче варианта составьте блок-схему и напишите программу решения задач.

2. Оформите отчет по лабораторной работе, который должен состоять из следующих разделов: номер лабораторной работы и ее название, условие каждой задачи, ее блок-схема и программа.

### Вариант № 1

1. Дано целое число  $X$ . Разложите его на простые множители.

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы делителей которых меньше заданного числа  $X$ .

3. В каких двузначных числах удвоенная сумма цифр равна их произведению?

### Вариант № 2

1. На натуральном отрезке чисел  $[A, B]$  найдите все совершенные числа. *Справка.* Число является совершенным, если оно равно сумме своих делителей без самого числа. Например,  $6 = 1 + 2 + 3$ .

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы делителей которых больше заданного числа  $X$ .

3. Найдите двузначное число, равное утроенному произведению его цифр.

### Вариант № 3

1. Шестизначное целое число начинается слева цифрой 1. Если эту цифру переместить с первого места на последнее, то новое шестизначное число будет втрое больше исходного. Найдите исходное число.

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых равной заданному числу  $X$ .

3. Найдите все двузначные числа, сумма квадратов цифр которых делится на 17.

### Вариант № 4

1. У Андрея и Бори было 11 орехов. У Андрея и Вовы – 12. У Вовы и Бори – 13. Сколько всего орехов было у мальчиков?

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых меньше заданного числа  $X$ .

3. Найдите двузначное число, обладающее тем свойством, что куб суммы его цифр равен квадрату самого числа.

### **Вариант № 5**

1. Выведите все четырехзначные числа, которые делятся на 45, а две средние цифры у них 97.

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых больше заданного числа  $X$ .

3. Найдите все трехзначные числа, которые можно представить разностью между квадратом числа, образованного первыми двумя цифрами и квадратом третьей цифры.

### **Вариант № 6**

1. Определите, можно ли упаковать 60 кг конфет в коробки по 3 и 5 кг. Если можно, то каким образом (приведите все варианты)?

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых являются нечетным числом.

3. Найдите все трехзначные числа, средняя цифра которых равна сумме первой и третьей цифр.

### **Вариант № 7**

1. Составьте программу получения всех трехзначных целых чисел, сумма цифр которых равна данному натуральному числу  $X$ .

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, которые являются степенью двойки.

3. Подсчитайте  $k$  – количество цифр в десятичной записи целого неотрицательного числа  $N$ .

### **Вариант № 8**

1. Один долгожитель, которому за 100 лет, заметил, что если к сумме квадратов цифр его возраста добавить дату его рождения (натуральное число, не большее 31), то получится его возраст. Сколько лет долгожителю?

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых являются делителями числа  $X$ .

3. Найдите все трехзначные числа, сумма цифр которых равна данному целому числу.

### Вариант № 9

1. Два двухзначных числа, записанных одно за другим, образуют четырехзначное число, которое делится на их произведение. Найдите эти числа.

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых есть число кратное числу  $X$ .

3. Даны натуральные числа  $n$  и  $k$ . Проверьте, есть ли в записи числа  $x = nk$  цифра  $m$ .

### Вариант № 10

1. Выведите на экран все обыкновенные несократимые дроби, заключенные между 0 и 1, знаменатель которых не превышает 7.

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых являются четным числом.

3. В каких трехзначных числах удвоенная сумма цифр равна их произведению?

### Вариант № 11

1. К числу 15 слева и справа припишите по одной цифре так, чтобы полученное число делилось на 15. Получите все такие числа.

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых делится на 5.

3. Найдите все двузначные числа, сумма цифр которых делится на 9.

### Вариант № 12

1. На натуральном отрезке чисел  $[A, B]$  найдите все простые числа.

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых делятся на 7.

3. Найдите все трехзначные числа, третья цифра которых равна сумме первой и второй цифр.

## Лабораторная работа № 6

### СТРОКОВЫЕ ПЕРЕМЕННЫЕ

*Цель работы:* овладеть практическими навыками работы со строковыми данными, с процедурами и функциями обработки текстов.

#### Теоретические сведения

Для работы с текстами и наборами слов используются данные строкового типа. Формат объявления строковой переменной имеет следующий вид:

*1 способ*

**Type**

Имя типа = **String**;

**Var**

**Имя переменной : имя типа;**

*2 способ*

**Var Имя переменной : String;**

STRING – служебное слово; максимальная длина символьной 255 символов. Строковый тип данных занимает промежуточное положение между простыми и структурированными типами данных.

Для работы со строковыми переменными используются свои операции, процедуры и функции. Текстовой константой, как уже известно, является последовательность символов, заключенная в апострофы. Выражения, операндами которых являются текстовые константы или строковые переменные, называются *строковыми выражениями*.

Для присваивания значения строковой переменной служит оператор присваивания. Значения строковых переменных можно вводить с клавиатуры, как и значения числовых переменных. Строковая переменная, которая не содержит ни одного символа, является пустой строкой, или нуль-строкой, и обозначается двумя апострофами без пробела (например, a:='').

Для объединения нескольких строк в одну служит операция *сцепления*, или операция *конкатенации*, которая обозначается знаком «+». При операции сцепления необходимо учитывать, чтобы длина полученной строки не превышала максимальной длины, иначе все лишние символы отбрасываются.

Операции отношения проводят сравнение строк. Строки считаются равными, если они имеют одинаковую длину, состоят из одних и тех же символов, идущих в одинаковом порядке. Сравнение строк идет слева направо до первого неодинакового символа. Если длины строк одинаковые, то большей считается та строка, у которой код несовпадающего символа больший. Если же строки состоят из одинаковых символов, то большей считается строка с большей длиной.

Для работы со строковыми переменными используются специальные процедуры и функции.

### *Стандартные функции*

**Length** (имя строковой переменной) – определение длины строки. Результат применения этой функции представлен в табл. 6.1.

Таблица 6.1

Значение <i>st</i>	Выражение	Результат
'123456789'	Length ( <i>st</i> )	9
'System 370'	Length ( <i>st</i> )	10

**Pos** (*st*<sub>1</sub>, *st*<sub>2</sub>) – нахождение первого появления в строке *st*<sub>2</sub> строки *st*<sub>1</sub>. Результат применения этой функции представлен в табл. 6.2.

Таблица 6.2

Значение <i>S</i> <sub>2</sub>	Выражение	Результат
'abcdefg'	Pos ('de', <i>S</i> <sub>2</sub> )	4
'abcdefg'	Pos ('r', <i>S</i> <sub>2</sub> )	0

**Copy** (*st*, начальная позиция, длина) – выделение из строки *st* подстроки от начальной позиции и указанной длины. Результат применения этой функции представлен в табл. 6.3.

Таблица 6.3

Значение <i>st</i>	Выражение	Результат
'ABCDEFGF'	Copy ( <i>st</i> ,2,3)	'BCD'
'ABCDEFGF'	Copy ( <i>st</i> ,4,10)	'DEFG'

**Concat ( $S_1, S_2, \dots, S_n$ )** – выполняет сцепление строк  $S_1, S_2, \dots, S_n$  в одну строку в том порядке, в каком они указаны. Сумма символов всех сцепленных строк не должна превышать 255. Результат применения этой функции представлен в табл. 6.4.

Таблица 6.4

Выражение	Результат
Concat ('AA','XX','Y')	'AAXXY'
Concat ('Индекс','220251')	'Индекс220251'

### **Стандартные процедуры**

**Delete (*st*, начальная позиция, длина)** – удаление из строки *st* символов от начальной позиции и указанной длины. Результат применения этой функции представлен в табл. 6.5

Таблица 6.5

Значение <i>st</i>	Выражение	Результат
'абвгде'	Delete (St,4,2)	'абве'
'река Волга'	Delete (St,1,5)	'Волга'

**Insert ( $st_1, st_2$ , начальная позиция)** – вставка строки  $st_1$  в строку  $st_2$  с отмеченной позиции. Результат применения этой процедуры представлен в следующей программе.

```

Var      S1,S2:string [11];
        Begin
          S1:=' EC ';
          S2:='ЭВМ1841';
          Insert (S1,S2,4);
        End.
```



В результате значение строки  $S_2$  станет равным 'ЭВМ ЕС 1841'

**Str** (арифметическое выражение, строковая переменная) – преобразование числового значения арифметического выражения в строковую переменную. Результат применения этой процедуры представлен в табл. 6.6.

Таблица 6.6

Значение $N$	Выражение	Результат
1500	str ( $N:6,St$ )	' 1500'
4,8E+03	str ( $N:10,St$ )	' 4800'
76854	str ( $N:3,St$ )	' 76854'

В этой таблице символ «\_» (подчеркивание) заменяет пробел.

**Val** (строковая переменная, числовая переменная,  $cod$ ) – преобразование строковой переменной в числовую переменную. В строковой переменной не должно быть пробелов. Если во время перевода ошибок не было, то переменной  $cod$  автоматически присвоится значение 0. Если же ошибки были, то переменной  $cod$  присвоится какое-то другое значение. Результат применения этой процедуры представлен в табл. 6.7

Таблица 6.7

Значение $st$	Выражение	Результат
'1450'	Val ( $st,N,Code$ )	Cod = 0
'14.2E+02'	Val ( $st,N,Code$ )	Cod = 0
'14.2A+02'	Val ( $st,N,Code$ )	Cod $\neq$ 0

### Примеры выполнения заданий

**Пример 1.** Составьте блок-схему (рис. 6.1) и напишите программу решения следующей задачи: дан текст. Напечатайте его, удалив первое вхождение «w», если такое есть.

```
PROGRAM primer_1;
var s:string;
    p:byte;
```

```

BEGIN
writeln("введите
текст ");
readln (s);
p:=pos('w',s);
if p<>0
then
begin
delete (s,p,1);
writeln ('Текст
после  удаления-
>',s);
end
else writeln
('Символ w отсут-
ствует');
END.

```

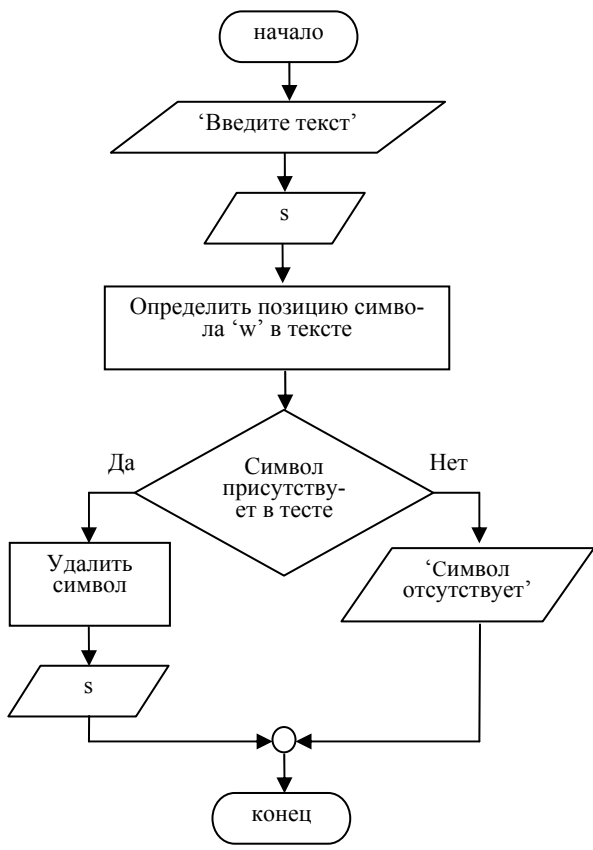


Рис. 6.1

**Пример 2.** Составьте блок-схему (рис. 6.2) и напишите программу решения следующей задачи: дан текст. Напечатайте его, удалив все вхождения «th». Определите начальную и конечную длину строки.

```

PROGRAM primer_2;
var s:string;
p:byte;
BEGIN
writeln('Введите текст');
readln (s);

```

```

writeln ('начальная длина строки=',length (s)) ;
p:=pos('th',s);
while p<>0 do
begin
delete (s,p,2);
p:=pos('th',s);
end;
writeln ('текст после удаления->',s);
writeln ('конечная длина строки->',length (s)) ;
END.

```

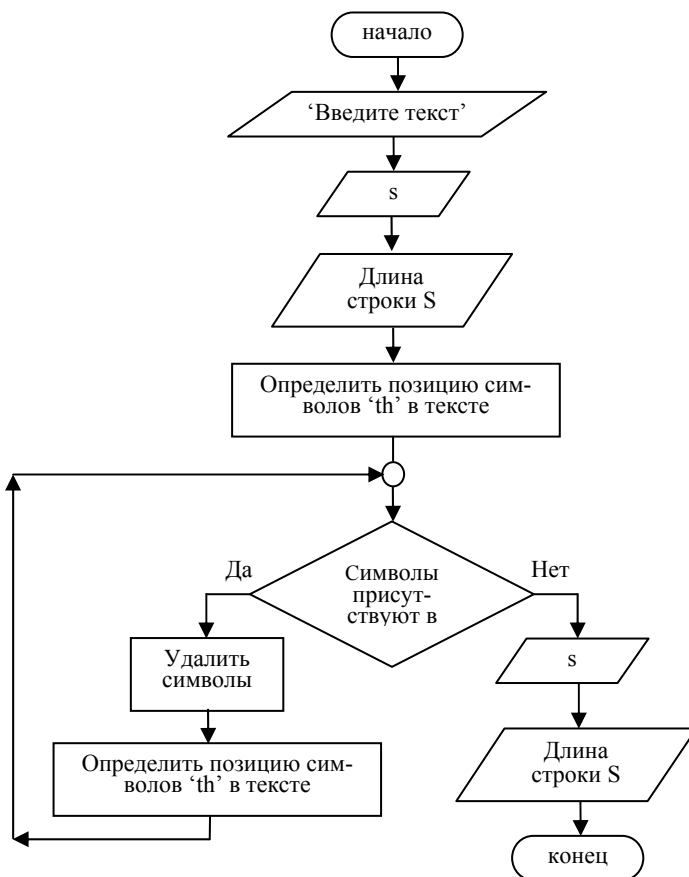


Рис. 6.2

**Пример 3.** Составьте блок-схему (рис. 6.3) и напишите программу решения следующей задачи: дано слово. Подсчитайте, сколько раз в нем встречается данная буква и замените ее на символ «-».

```

PROGRAM Primer_3;
Var s,a,x:String[30];
    L,i,k:Integer;
BEGIN
    Writeln('Ввести слово');
    Readln(s);
    L:=Length(s); {Определе-
ние длины слова}
    Writeln('Ввести букву');
    Readln(a);
    x:=""; {Новое слово пока
пустое}
    k:=0; {Счетчик количества
данной буквы}
    For i:=1 to L do
        If copy(s,i,1)=a {Если те-
кущая буква слова равна дан-
ной букве}
        Then Begin x:=x+'-'; k:=k+1;
        End
        {то к новому слову прибав-
ляем символ «-» и к счетчику
прибавляем 1}
        Else x:=x+copy(s,i,1);
        {иначе к новому слову при-
бавляем текущую букву старо-
го слова}
    Writeln('Данное слово -- ', s);
    Writeln('Буква ',a,' встре-
чается ',k,' раз');
    Writeln('Полученное слово
-- ',x);
END.

```

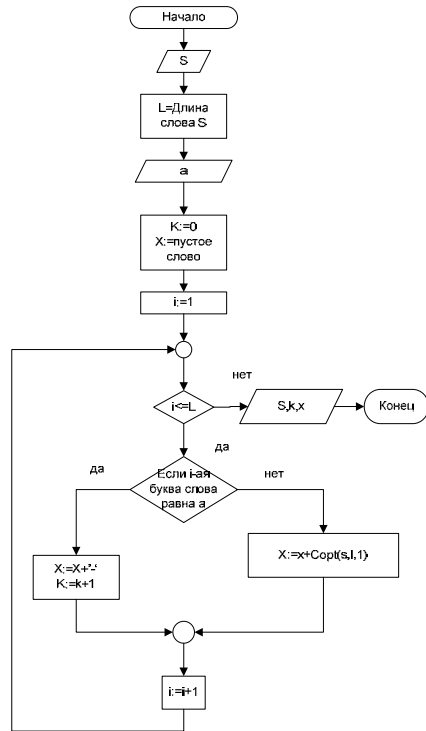


Рис. 6.3

**Пример 4.** Составьте блок-схему (рис. 6.4) и напишите программу решения следующей задачи: дан текст. Слова в тексте отделяются одно от другого одним пробелом. Определите, есть ли в тексте данное слово.

```
PROGRAM Primer_4;
Var
  t,t1,x,y :String;
  D,k,z:Integer;
BEGIN
  Writeln('Ввести текст');
  Readln(t);
  D:=Length(t); {Определение длины текста}
  t1:=t+' ';
```

{копирование исходного текста, так как при работе исходный текст может пострадать и добавление в конце текста пробела, чтобы последнее слово заканчивалось пробелом, как и все остальные слова в тексте}

```
Writeln('Ввести слово X');
Readln(x);
k:=0;
```

{k – это переменная, которая будет являться дополнительной. Сначала этой переменной присвоим значение 0, а затем если в тексте обнаружим искомое слово, то значение этой переменной изменим, например, на 5}

{Алгоритм будет заключаться в следующем. Поскольку слова в тексте отделяются друг от друга одним пробелом, то будем находить позицию пробела в тексте. Сделаем копию от начала текста до этой позиции – это есть слово текста. Сравним полученное слово текста и данное слово. Если они равны, то изменим значение дополнительной переменной. После этого слово из текста вместе с пробелом удалим. В конце получим, что исходный текст станет пустым.}

```
While t1<>'' do {Пока текст t1 не пустой}
  Begin
```

```

Z:=Pos(' ',t1); {Поиск позиции пробела в тексте t1}
Y:=Copy(t1,1,z-1); {Копирование слова из текста t1 от первой
позиции до позиции z-1}
If x=y {если данное слово равно слову из текста}
Then k:=5;
Delete(t1,1,z); {Удаление слова из текста}
End;
If k=0
Then Writeln('В тексте ',t,' нет слова ', x)
Else Writeln('В тексте ',t,' есть слово ',x);
END.

```

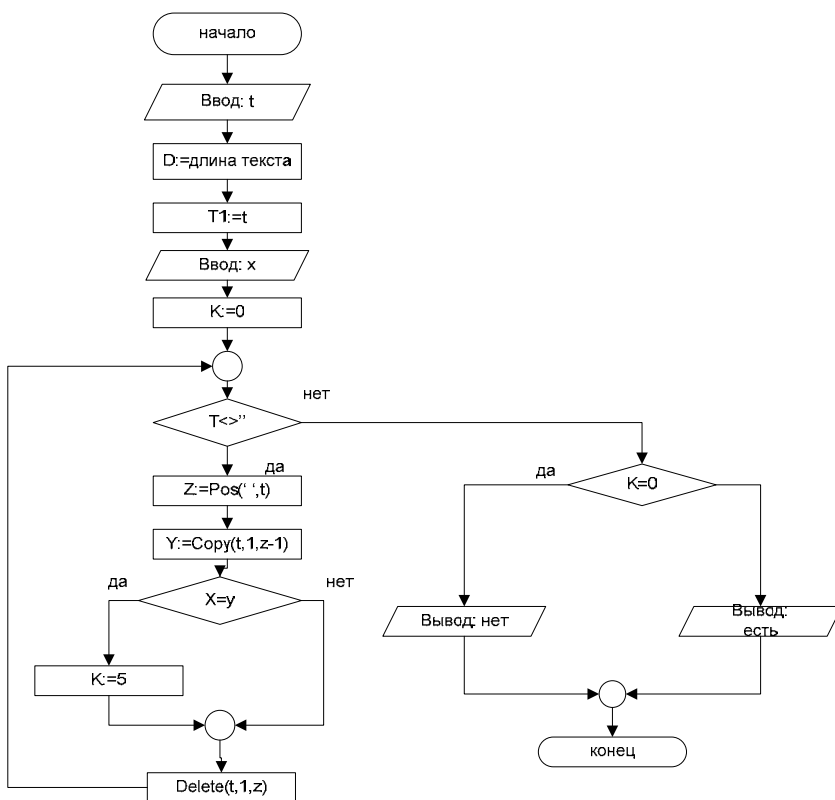


Рис. 6.4

**Пример 5.** Составьте блок-схему (рис. 6.5) и напишите программу решения следующей задачи: дан текст. Слова в тексте отделяются одно от другого одним пробелом. Замените одно данное слово  $X$  другим данным словом  $Y$ .

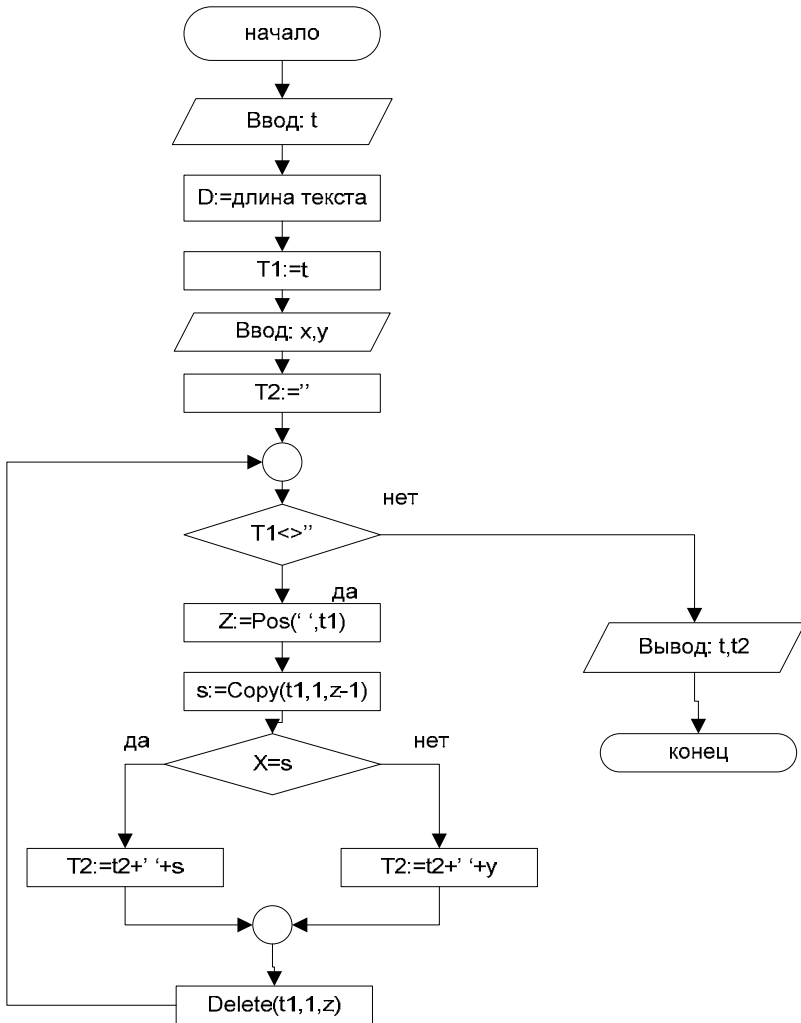


Рис. 6.5

```

PROGRAM Primer_5;
Var
  t,t1,t2,x,y,s :String;
  D,z:Integer;
BEGIN
  Writeln('Ввести текст'); Readln(t);
    D:=Length(t); {Определение длины текста}
  t1:=Copy(t,1,D-1)+' '; {в исходном тексте точку в конце заменим
на пробел, получим новый текст}
  Writeln('Ввести слово X'); Readln(x); {слово из текста, которое
хотим заменить}
  Writeln('Ввести слово Y'); Readln(y); {слово на которое меняем
слово из текста}
  T2:=""; {Новый текст пока пустой}
  {Алгоритм будет заключаться в следующем. Поскольку слова в
тексте отделяются друг от друга одним пробелом, будем находить
позицию пробела в тексте. Сделаем копию от начала текста до этой
позиции – это есть слово текста. Сравним полученное слово текста
и данное слово. Если они равны, то к новому тексту прибавим новое
слово. Если же они не равны, то к новому тексту прибавим слово из
старого текста. Затем из старого текста слово удаляем.}
  While t1<>" do {Пока текст не пустой}
    Begin
      Z:=Pos(' ',t1); {Поиск позиции пробела в тексте}
      s:=Copy(t1,1,z-1); {Копирование слова из текста}
      If s=x
        {Если данное слово x равно слову y из текста }
          Then t2:=t2+' '+ y {К новому тексту прибавляем данное
слово y }
          Else t2:=t2+' '+s; {К новому тексту прибавляем слово из
исходного текста}
        Delete(t1,1,z); {Удаление слова из текста}
      End;
    Writeln('Исходный текст ',t);
    Writeln('Полученный текст ',t2+'.');
  END.

```



**Пример 6.** Составьте блок-схему (рис. 6.6) и напишите программу решения следующей задачи: с клавиатуры вводится длинное целое число, не менее 50 цифр. Определите сумму цифр данного числа.

```

PROGRAM primer_6;
Var
  t,x :String[30];
  {t – исходный текст,  x – один символ из текста}
  l,s,z,I,cod:Integer;
  {l – длина текста,
  S – сумма цифр исходного числа,
  Z – число, в которое переводится один символ текста,
  I – переменная цикла,
  Cod – переменная для процедуры VAL}
BEGIN
  {Введем число как текст, т.е. набор символов-цифр}
  Writeln('Ввести текст');
  Readln(t);
  l:=Length(t); {Определение длины текста}
  {Будем выделять по одному символу из текста, переводить этот
  символ в цифру и прибавлять ее к сумме}
  s:=0;
  For i:=1 to l do
    Begin
      x:=Copy(t,I,1); {Выделение одного символа из текста}
      Val(x, z, cod); {Перевод символа x в цифру z}
      If cod=0 {Если перевод прошел без ошибок, то переменная cod
      принимает значение 0}
        Then s:=s+z {Нахождение суммы цифр}
        Else Writeln('При переводе ‘,I,’ символа обнаружена ошибка’);
    End;
  Writeln('Сумма цифр исходного числа =',s);
END.

```

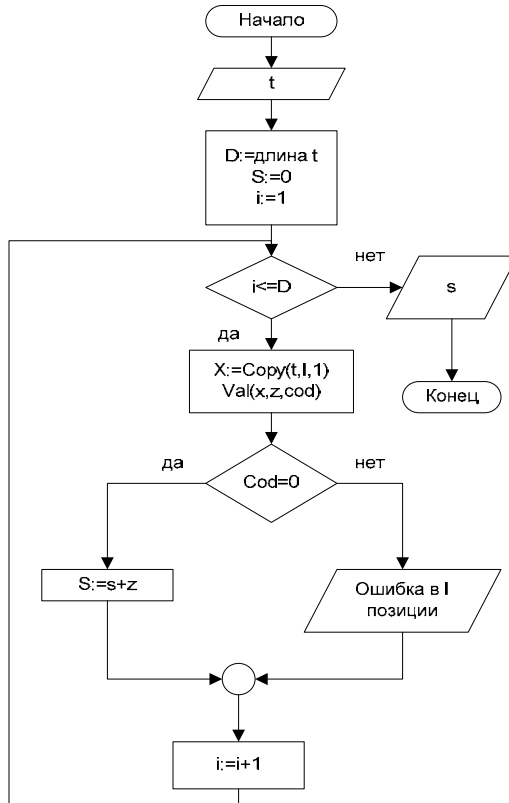


Рис. 6.6

**Пример 7.** Напишите программу решения следующей задачи: введите две строки, состоящие из цифр, если хотя бы одна из строк содержит символы, отличные от цифр, то выдайте соответствующее сообщение, иначе определите сумму введенных чисел и результат сцепления двух строк.

```

PROGRAM primer_7;
  var S1,S2:string;
      X1,X2,code1,code2:integer;
BEGIN
  writeln ('Введите первую строку цифр');

```

```

readln (s1);
writeln ("Введите вторую строку цифр ");
readln (s2);
val (s1,x1, code1);
val (s2,x2, code2);
if code1<>0 then writeln ('Ошибка в первой строке цифр');
if code2<>0 then writeln ('Ошибка во второй строке цифр');
if (code1=0) and (code2=0) then
begin
    writeln('Результат сцепления строк->',Concat(s1,s2));
    writeln ('Сумма введенных чисел->', x1+x2);
end;
END.

```

### **Контрольные вопросы**

1. Строковый тип относится к простому или структурированному типу данных? Ответ аргументируйте.
2. Приведите примеры объявления строковых переменных различной длины?
3. Охарактеризуйте строковый тип данных.
4. Перечислите признаки, благодаря которым строковый тип данных можно отнести к простым.
5. Перечислите процедуры и функции специально предназначенные для работы со строками.
6. «pascal»... «PASCAL» Поставьте знак отношения. Ответ аргументируйте.
7. Назначение и синтаксис процедуры Delete.
8. Назначение и синтаксис процедуры Insert.
9. Назначение и синтаксис процедуры Str.
10. Назначение и синтаксис процедуры Val.
11. Назначение и синтаксис функции Copy.
12. Назначение и синтаксис функции Concat.
13. Назначение и синтаксис функции Length.
14. Назначение и синтаксис функции Pos.

## Задания Для Самостоятельного Выполнения

1. В каждой задаче варианта составьте блок-схему и напишите программу решения задач.

2. Оформите отчет по лабораторной работе, который должен состоять из следующих разделов: номер лабораторной работы и ее название, условие каждой задачи, ее блок-схема и программа.

### Вариант № 1

1. Определите количество букв «а» в слове.

2. Дан текст, в котором слова отделяются пробелами, в конце стоит точка. Определите количество слов и выведите на экран каждое слово, стоящее на четном месте в предложении.

3. С клавиатуры введите длинное целое число, не менее 50 цифр. Определите первую и последнюю цифры числа и поменять их местами.

### Вариант № 2

1. Сколько букв «у» в слове стоит на четных местах.

2. Дан текст, слова в нем разделяются пробелом, в конце стоит точка. Найдите самое короткое слово в массиве и его номер в тексте.

3. С клавиатуры введите длинное целое число, не менее 50 цифр. Определите сумму цифр данного числа.

### Вариант № 3

1. Дана символьная строка. Замените все символы «!» точками, кроме первого встречающегося в строке, и выведите полученную строку.

2. Дан текст, слова в нем разделяются пробелом, в конце стоит точка. Найдите количество слов в тексте и выведите каждое слово, стоящее на четном месте в предложении.

3. С клавиатуры введите длинное целое число, не менее 50 цифр. Найдите сумму цифр, стоящих в середине числа.

### Вариант № 4

1. Является ли перевертышем слово.

2. Дан текст, в котором слова отделяются пробелами, в конце стоит точка. Определите количество слов в тексте и выведите те слова, которые имеют четное количество символов.

3. С клавиатуры введите длинное целое число, которое имеет не менее 50 цифр. Найдите произведение первой, средней и последней цифр.

### **Вариант № 5**

1. Замените в слове все буквы «о» пробелами.

2. Дан текст, в котором слова отделяются пробелами, в конце стоит точка. Определите количество слов и выведите только те слова, количество символов в которых нечетное.

3. С клавиатуры введите длинное целое число, не менее 50 цифр. Определите сумму всех четных цифр данного числа.

### **Вариант № 6**

1. Выясните, какая из букв – первая или последняя – встречается в заданном слове чаще.

2. Дан текст, в котором слова отделяются пробелами, в конце стоит точка. Определите количество слов в тексте и выведите слова, которые по длине меньше заданного числа.

3. С клавиатуры введите длинное целое число, не менее 50 цифр. Ответьте на вопрос, являются ли первая цифра четной, а последняя цифра нечетной.

### **Вариант № 7**

1. Напишите программу, которая вводит строку и выводит ее, сокращая каждый раз на один символ до тех пор, пока в строке не останется один символ.

2. Дан текст, в котором слова отделяются пробелами, в конце стоит точка. Определите количество слов в тексте и выведите те слова, длина которых больше заданного числа.

3. С клавиатуры введите длинное целое число, не менее 50 цифр. Переставьте местами первую и последнюю цифры данного числа и выведите полученное число.

### **Вариант № 8**

1. Задано существительное первого склонения, оканчивающееся на «а». Напечатайте это слово во всех падежах.

2. Дан текст, в котором слова отделяются пробелами, в конце стоит точка. Определите количество слов в тексте и найдите среднюю длину слова.

3. С клавиатуры вводится длинное целое число, которое имеет четное количество цифр не менее 50. Найдите средние цифры числа и переставьте их местами.

### **Вариант № 9**

1. Проверьте баланс скобок {}, {} в строке.
2. Дан текст, в котором слова отделяются пробелами, в конце стоит точка. Исключите из текста слова, длина которых меньше заданного числа.
3. С клавиатуры вводится длинное целое число, которое имеет нечетное количество цифр не менее 50. Найдите среднюю цифру числа, если она четная, то поменять ее на 0; если она нечетная, то поменять ее на 9. Выведите полученное число.

### **Вариант № 10**

1. Выясните, какая из букв – вторая или предпоследняя – встречается в заданном слове чаще.
2. Дан текст, в котором слова отделяются пробелами, в конце стоит точка. Определите количество слов, которые начинаются и заканчиваются одинаковым символом.
3. С клавиатуры вводится длинное целое число, не менее 50 цифр. Определите сумму четных цифр числа.

### **Вариант № 11**

1. Дан текст. Установите пробелы вместо символов, номера позиций которых при делении на 4 дают в остатке 3.
2. Дан текст, в котором слова отделяются пробелами, в конце стоит точка. Выведите те слова, которые оканчиваются на сочетание «ова».
3. С клавиатуры введите длинное целое число, не менее 50 цифр. Определите сумму нечетных цифр числа.

### **Вариант № 12**

1. Дана символьная строка. Замените все символы «?» точками, кроме последнего встречающегося в строке, и выведите полученную строку.
2. Дан текст, в котором слова отделяются пробелами, в конце стоит точка. Найдите слово максимальной длины.
3. С клавиатуры введите длинное целое число, не менее 50 цифр. Выведите, на каких местах в данном числе расположена цифра «5».

## ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ПОЛЬЗОВАТЕЛЬСКИХ ПРОЦЕДУР И ФУНКЦИЙ

**Цель работы:** получить практические навыки в написании подпрограмм в виде процедур и функций, изучить механизм передачи параметров в процедурах и функциях.

### Теоретические сведения

В практике программирования часто встречаются случаи, когда по ходу выполнения программы приходится выполнять одни и те же действия но при различных исходных данных. Чтобы исключить повторение одинаковых записей и сделать тем самым программу проще и понятнее, можно выделить эти повторяющиеся действия в самостоятельную часть программы – подпрограмму, которая может быть использована многократно по мере необходимости.

*Подпрограммой* называют обособленную, оформленную в виде отдельной синтаксической конструкции и снабженную именем часть программы. Использование подпрограмм позволяет, сосредоточив в них подробное описание некоторых операций, в остальной программе только указывать имена подпрограммы, чтобы выполнить эти операции. Возможны неоднократные вызовы подпрограмм из разных участков программы, причем при вызове подпрограмме можно передать некоторую информацию (различную для разных вызовов), чтобы одна и та же подпрограмма выполняла решение подзадачи для разных случаев.

За наличие подпрограмм как средства структурирования программ язык программирования Pascal называется *процедурно-ориентированным*. Подпрограммы в ПаскальABC реализованы посредством процедур и функций. Процедуры и функции различаются назначением и способом их использования.

*Процедура* – это независимая именованная часть программы, которую можно вызвать по имени для выполнения определенных действий. Структура процедуры повторяет структуру программы. Процедура не может выступать в качестве операнда в выражении.

Упоминание имени процедуры в тексте программы приводит к активизации процедуры и называется ее *вызовом*. Например, *Read(F)* читает с клавиатуры некоторое значение и присваивает его переменной *F*, *Delay(5)* вызывает задержку выполнения программы на 5 мс.

*Функция* аналогична процедуре, но имеются два отличия: функция передает в точку вызова скалярное значение; имя функции может входить в выражение в качестве операнда. Например, функция *Chr(65)* возвращает в точку вызова символ *A* (код ASCII – 65), *Sqr(X)* – возводит в квадрат значения целого или вещественного *X* и возвращает в точку вызова вычисленное значение квадрата числа *X*.

Процедуры и функции в Паскале (как и во многих других языках программирования) являются удобным средством для однократного автономного описания тех фрагментов алгоритма, которые повторяются в разных его частях. Такие описания образуют раздел, который должен быть последним из числа разделов описаний программы. Для выполнения описанных в этом разделе фрагментов алгоритма в разделе операторов программы должны содержаться обращения к соответствующим процедурам и функциям.

Все процедуры и функции языка ПаскальABC делятся на две группы: *встроенные* (стандартные) и *определенные пользователем*. Встроенные входят в состав языка и вызываются для выполнения по строго фиксированному имени (например, *Writeln('s=',s)*; *Readln(a,b,c)*;). Определенные пользователем процедуры разрабатываются и именуется самим пользователем.

Все стандартные средства располагаются в специализированных библиотечных модулях. Для использования стандартной процедуры или функции к программе подключается тот или иной специализированный библиотечный модуль, в который входит данная стандартная процедура или функция, для чего имя специализированного библиотечного модуля указывается в разделе *uses*. Затем в программе осуществляется вызов процедуры или функции, для чего записывается ее имя и указываются фактические параметры, например, *Sin(X)*, *Chr(125)*, *Inc(X,5)*. До сих пор рассматривались стандартные процедуры и функции, которые входят в модуль *System*. При выполнении программы к каждой программе этот модуль подключается автоматически, поэтому раньше не записывалась строка *Uses*. Так как после выполнения функции ее значение присваивается имени, то имя функции используется в выражении.



Разделы описаний процедуры, подобно основной программе содержит разделы *label*, *const*, *type*, *var* и, в свою очередь, раздел процедур и функций. Раздел операторов заключается в операторные скобки *begin* – *end*, причем после *end*, в отличие от основной программы, ставится символ «;». Процедуры помещаются в главной программе в разделе объявлений основной программы.

Отличительная особенность функции заключается в том, что она имеет только один результат выполнения, который обозначается именем функции и возвращается (передается) в основную программу. Таким образом, алгоритм можно оформить в виде функции в том случае, если в качестве результата получается одно единственное значение. Для вызова функции достаточно указать ее имя (с фактическими параметрами) в любом выражении. Отметим, что имя функции можно использовать в арифметических выражениях и других командах.

### ***Процедуры пользователя***

Заголовок процедуры имеет следующий вид:

**Procedure < имя > (< список формальных параметров >);**

Список формальных параметров необязателен и может отсутствовать. Если же он есть, то в нем должны быть перечислены имена формальных параметров и их тип. Несколько однотипных параметров можно объединять в подписки, например:

Procedure SB (a : Real; b, d : Integer; c : Char);

Процедура пользователя представляет собой именованную группу операторов, реализующую определенную часть общей задачи и вызываемую при необходимости для выполнения по имени из любой позиции раздела операторов.

Обращение к подпрограммам осуществляется указанием имени подпрограммы и *списка фактических параметров*, в качестве которых могут выступать константы, переменные и выражения, отделяемые друг от друга запятой. Количество и типы фактических параметров при обращении к подпрограмме должны точно соответствовать количеству и типам ее формальных параметров.

Для обращения к процедуре используется специальный оператор, состоящий из имени процедуры и списка фактических параметров.

Соответствие между фактическими и формальными параметрами должно быть следующим:

- количество фактических параметров должно быть равно количеству формальных параметров;
- соответствующие фактические и формальные параметры должны совпадать по порядку следования и типу. Соответствующие параметры желательно обозначать не одинаково.

Параметры процедур могут быть двух видов: параметры-значения, параметры-переменные.

*Параметры-значения* используются для передачи исходных данных в процедуру, в списке формальных параметров они указываются через запятую с определением типов.

*Параметры-переменные* необходимы для сохранения в них результатов выполнения процедуры, а значит значения этих параметров передаются из процедуры в основную программу, перед ними необходимо ставить зарезервированное слово *Var*, например:

### **Procedure FF ( Var a : Real);**

После завершения выполнения процедуры управление передается в основную программу оператору, который первым следует за вызовом процедуры.

### ***Функции пользователя***

Заголовок функции пользователя имеет следующий синтаксис:

**Function < имя > (< список формальных параметров >):  
<тип>;**

Для обращения к функции ее имя со списком фактических параметров должно войти как операнд в некоторое выражение. Отсюда вытекает несколько отличий описания функции от описания процедуры:

- в виде функции описывается алгоритм, результатом выполнения которого является скалярная величина;
- в блоке функции должен быть хотя бы один оператор присваивания с именем функции в левой части;
- в заголовке функции должен быть указан тип функции (тип возвращаемого через имя функции результата).

Все переменные, используемые в основной программе и подпрограммах (процедурах), могут быть разделены на глобальные и локальные переменные. Если строка `Var` основной программы идет перед описанием процедуры или функции, то переменные, описанные в этой строке, являются *глобальными* и могут действовать как в основной программе, так и в процедуре или в функции. Если же строка `Var` основной программы записана после объявления процедур или функций, то эти переменные являются локальными и действуют только в пределах основной программы. В процедуре или функции также имеются локальные переменные – это переменные, которые используются только внутри процедуры или функции, никак не связанные с основной программой и описываемые в разделе `Var` процедуры или функции.

Пользовательские процедуры и функции будем размещать в головной (основной) программе одним из способов:

- в разделе описаний основной программы;
- оформлять в виде отдельных файлов и подключать в раздел описаний основной программы с помощью директивы компилятора `{I имя файла}` в процессе компиляции (где *I* – сокращение английского слова *include* – включить).

При отладке программ, которые используют процедуры или функции пользователя, для выполнения этих процедур или функций по шагам применяется клавиша `F7`.

При составление блок-схем блок-схемы процедуры и основной программы составляются отдельно. Блоки (рис. 7.1), в которых идет обращение к процедуре или функции, изображаются прямоугольниками с двойными стенками. Внутри прямоугольника записывается оператор обращения к процедуре или функции.

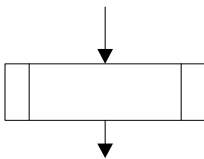


Рис. 7.1

## Примеры выполнения заданий

**Пример 1.** Составьте блок-схему (рис. 7.2) и напишите программу решения следующей задачи: демонстрация использования процедуры без параметров, которая рисует строку из \*.

```

PROGRAM primer_1;
var a,b: Integer;
{-----}
PROCEDURE Stars;
var i: Integer;
BEGIN
For i:=1 to 40 do Write (*);
WriteLn
END;
{-----}
BEGIN
WriteLn;
Stars;
Write ('Введите значение a... ');
ReadLn (a);
Write ('Введите значение b... ');
ReadLn (b);
Stars; WriteLn ('a+b=',a+b,
a*b=' ,a*b, ' ; a-b=' ,a-b); Stars
END.
    
```

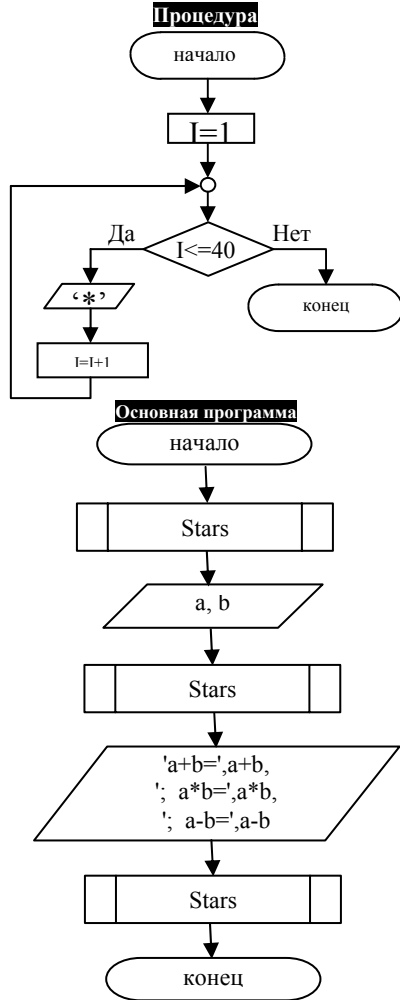


Рис. 7.2. Блок-схема процедуры и основной программы примера 1

**Пример 2.** Составьте блок-схему (рис. 7.3) и напишите программу решения следующей задачи: демонстрация использования процедуры с параметрами.

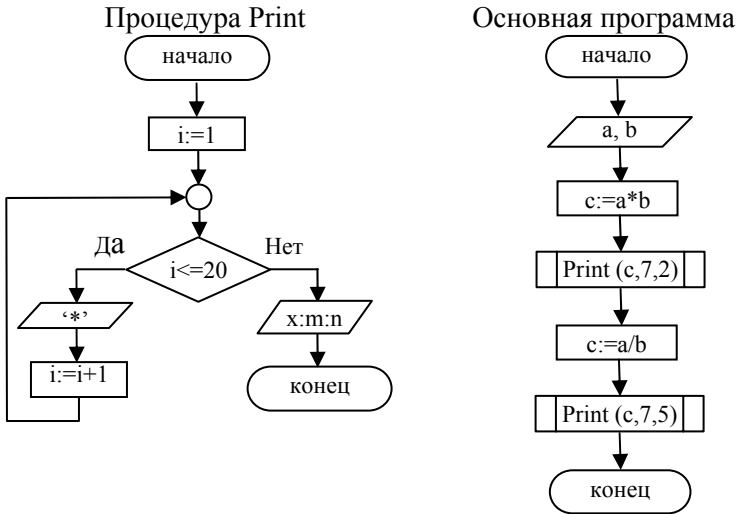


Рис. 7.3. Блок-схема процедуры и основной программы примера 2

```

PROGRAM Primer_2;
var a,b,c: Real;
{-----}
PROCEDURE Print (x: Real; m,n: Integer);
{Вывод на экран значения переменной x с n цифрами после запятой}
{x,n - параметры, передаваемые по значению}
var i: Integer;
BEGIN
For i:=1 to 20 do Write ('*');
WriteLn; WriteLn (x:m:n)
END;
{-----}
BEGIN
Write ('Введите значение a... '); ReadLn (a);
Write ('Введите значение b... '); ReadLn (b);
c := a*b; Print (c,7,2); c := a/b; Print (c,7,5)
END.
  
```

**Пример 3.** Напишите программу решения следующей задачи: задается два целых числа 5 и 7, эти числа передаются процедуре *Inc2*, в которой они удваиваются. Один из параметров передается как параметр-переменная, другой – как параметр-значение. Значения параметров до и после вызова процедуры, а так же результат их удвоения выводятся на экран.

```
PROGRAM primer_3;
Var a,b: Integer;
Procedure Inc2(Var c: Integer; b: Integer);
BEGIN
c := c + c;
b := b + b;
Writeln('Удвоенные :', c:5, b:5)
END; {Inc2}
BEGIN
a:=5; b:=7;
Writeln('Исходные :', a:5, b:5);
Inc2(a, b);
Writeln('Результат :', a:5, b:5)
END.
В результате прогона программы будет выведено:
Исходные : 5 7
Удвоенные : 10 14
Результат : 10 7
```

Таким образом, видим, что переменная *c* стоит после *Var*, то есть объявлена как параметр-переменная и ее новое значение было передано в основную программу, а переменная *b* объявлена в заголовке процедуры как параметр-значение и ее новое значение в основную программу не было передано.

**Пример 4.** Напишите программу, в которой необходимо описать функцию, вычисляющую факториал, и использовать ее для вычисления  $f(m, n) = \frac{n!m!}{(n+m)!}$ , где  $m, n$  – неотрицательные целые числа.

```

PROGRAM primer_4;
Var n,m: Integer; f: Real;
FUNCTION fact(n: Integer): Integer;
Var i,p : Integer;
BEGIN p:=1;
For i:=2 To n do p := p* i;
fact:= p
END;
BEGIN
Write(' m = ');
ReadLn(m);
Write(' n = ');
ReadLn(n);
f:= fact(n) * fact(m) / fact(n+m);
Writeln(f)
END.

```

**Пример 5.** Напишите программу вычисления степени числа  $X^n$ , где  $n$  – целое число,  $X \neq 0$ . Степень числа с целым показателем вычислите по формуле

$$X^n = \begin{cases} 1, & \text{если } n = 0; \\ X^n, & \text{если } n > 0; \\ \frac{1}{X^{-n}}, & \text{если } n < 0. \end{cases}$$

При решении задачи использовать функцию пользователя.

```

PROGRAM Primer_5;
Var {Глобальные переменные: n, x, konz}
n: integer; {Степень числа}
x: real; {Основание числа}
konz: real; {Результат вычислений}
{Функция Xst}
FUNCTION Xst (k:integer; y: real):real; {Список формальных па-
раметров k, y}

```

```

    {Входные параметры k- показатель степени, y – основание сте-
пени; выходной параметр Xst – значение  $y^k$ }
    Var j: integer; {Локальные переменные j, rez}
    rez: real;
    BEGIN
    rez:=1;
    j:=1;
    While j <= k do
    begin
    rez:=rez*y; {Вычисление степени}
    inc(j); {Следующее значение j}
    end;
    xst:=rez;
    END;
    BEGIN
    Write('Введите показатель степени числа = ');
    ReadLn(n);
    Write('Введите основание степени = ');
    ReadLn(x);
    if n=0 then konz:=1
    else if n>0 then konz:=Xst(n,x) {Вызов функции Xst из
выражения}
    {Фактические параметры n, x}
    {Входные параметры n, x; выходной параметр Xst}
    else konz:=Xst(-n,1/x); {Вызов функции Xst из выражения}
    {Фактические параметры -n, 1/x}
    {Входные параметры -n, 1/x; выходной параметр Xst}
    WriteLn(x:8:2,' в степени ', n,' = ',konz:8:3);
    END.

```

### **Контрольные вопросы**

1. Что такое подпрограмма?
2. В каком разделе описываются пользовательские процедуры и функции?
3. Как называется язык программирования ПаскальABC за наличие подпрограмм?
4. Посредством чего реализованы подпрограммы в ПаскальABC?



5. Что такое процедура?
6. Чем функция отличается от процедуры?
7. Как вызываются процедуры?
8. На какие группы делятся все процедуры и функции в ПаскальABC?
9. Приведите примеры встроенных процедур. Каково их назначение?
10. Приведите примеры встроенных функций. Каково их назначение?
11. Где располагаются стандартные процедуры и функции?
12. Как осуществляется вызов стандартной процедуры или функции?
13. В каком случае организуется пользовательские подпрограммы?
14. Какова структура пользовательской процедуры?
15. Какова структура пользовательской функции?
16. Что такое параметр?
17. Установите соответствующие обозначения в заголовке процедуры.

```
PROCEDURE <...> (...);
Const ... ;Type ... ;Var ... ;Begin операторы End;
```

18. Установите соответствующие обозначения в заголовке функции.

```
FUNCTION <...> (...): <...>;
Const ... ;Type ... ;Var ... ;Begin операторы End.
```

19. Какие параметры являются формальными? Почему они так называются?
20. Какие параметры являются фактическими?
21. Что может включать в себя список формальных параметров?
22. В чем заключается соответствие между формальными и фактическими параметрами?
23. *Procedure MyProcedure (var B, D: Real; A: Real; c: String);* Перечислите параметры-переменные, параметры-значения.
24. Охарактеризуйте механизм передачи данных с помощью параметров-переменных.

25. Охарактеризуйте механизм передачи данных с помощью параметров-значений.

26. Охарактеризуйте механизм передачи данных с помощью параметров-констант.

27. Какие объекты называются локальными?

28. Какие объекты называются глобальными?

29. Каковы способы размещения пользовательских подпрограмм в основной программе?

30. Укажите имена локальных, глобальных переменных, формальных и фактических параметров. В каком месте программы осуществляется вызов процедуры?

```
PROGRAM Primer;  
var x,y: Real;  
PROCEDURE I_N_P_U_T (var a,b: Real);  
BEGIN  
...  
END;  
BEGIN  
I_N_P_U_T (x,y); ...  
END.
```

31. Почему использование параметров-переменных нежелательно в функции?

32. Укажите имена локальных и глобальных переменных, формальных и фактических параметров. В каком месте программы осуществляется вызов процедуры?

```
program ekz;  
var x,y,z:integer;  
procedure nechet (a: integer; var b:integer);  
begin  
...  
end;  
begin  
writeln ('vvedite chislo');  
readln (x);  
nechet (x,z);  
...  
end.
```

33. Укажите имена локальных и глобальных переменных, формальных и фактических параметров. В каком месте программы осуществляется вызов функции?

```
uses crt;
var a:real;
function tg(x:real):real;
begin
  tg:=sin(x)/cos(x);
end;
begin
  clrscr;
  writeln ('vvedite chislo');
  readln (a);
  writeln ('catangens chisla ',a:5:2,' = ',1/tg(a):5:2) ;
end.
```

### Задания для самостоятельного выполнения

1. В каждой задаче составить блок-схему и написать программу решения задач.

2. Оформите отчет по лабораторной работе, который должен состоять из следующих разделов: номера лабораторной работы и ее названия, условия каждой задачи, ее блок-схемы и программы.

#### Вариант № 1

1. Даны значения  $c, d, l, k$ . Напишите функцию пользователя  $\min$  для вычисления значения следующего выражения:

$$\frac{\min(c + d, l, k) + \min(k, c, l)}{3 + \min(k, d - l, c + l)}.$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы делителей которых меньше заданного числа  $X$ . Проверку, является ли сумма делителей данного числа меньше заданного числа  $X$ , сделайте с помощью процедуры пользователя.

3. Даны три числа  $A, B, C$ . Найдите наибольший общий делитель этих чисел. Определение наибольшего общего делителя двух чисел оформите в виде пользовательской процедуры или функции.

4. Вычислите значение выражения  $y = x^n + z^n$ . Вычисление значения степени оформите в виде пользовательской процедуры или функции.

### Вариант № 2

1. Даны значения  $x$  и  $y$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения:

$$z = \frac{\min(0, x) - \min(0, y)}{\max^2(y, x)}.$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы делителей которых больше заданного числа  $X$ . Проверку, является ли сумма делителей данного числа больше заданного числа  $X$ , сделайте с помощью процедуры пользователя.

3. Даны две простые дроби числитель первой  $A$ , знаменатель  $B$ , числитель второй дроби  $C$ , знаменатель  $D$ . Найдите их сумму и, если можно, сократите полученный результат. Определение наибольшего общего делителя двух чисел оформите в виде пользовательской процедуры или функции.

4. Вычислите значение выражения  $y = x^n - z^k$ . Вычисление значения степени оформите в виде пользовательской процедуры или функции.

### Вариант № 3

1. Даны значения  $x$  и  $y$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения:

$$v = \max(\min(x - y, y - x), 0).$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых равной заданному числу  $X$ . Проверку, является ли сумма цифр данного числа равной заданному числу  $X$ , сделайте с помощью процедуры пользователя.

3. Два отрезка на плоскости заданы координатами своих концов. Сравните длины этих отрезков. Нахождение длины отрезка оформите в виде пользовательской процедуры или функции.

4. Вычислите значение выражения  $y = ax^n$ . Вычисление значения степени оформите в виде пользовательской процедуры или функции.

#### Вариант № 4

1. Даны значения  $x$  и  $y$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения:

$$w = \max^2(\max(x * y, x + y), 0).$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых меньше заданного числа  $X$ . Проверку, является ли сумма цифр данного числа меньше заданного числа  $X$ , сделайте с помощью процедуры пользователя.

3. Даны координаты вершин некоторого четырехугольника. Найдите его площадь. Вычисление площади одного треугольника по формуле Герона оформите в виде пользовательской процедуры или функции.

4. Вычислите значение выражения  $y = a : b + z^n$ . Вычисление значения степени оформите в виде пользовательской процедуры или функции.

#### Вариант № 5

1. Даны значения  $x$  и  $y$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения

$$t = \frac{\max(\min(x, 5), \max(y, 0))}{5}$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых больше заданного числа  $X$ . Проверку, является ли сумма цифр данного числа больше заданного числа  $X$ , сделайте с помощью процедуры пользователя.

3. Два отрезка выходят из начала координат. Даны координаты концов этих отрезков. Какой из отрезков длиннее? Определение длины одного отрезка оформите в виде пользовательской процедуры или функции.

4. Найдите сумму  $N$  чисел Фибоначчи. Вычисление каждого члена Фибоначчи оформите в виде пользовательской процедуры или функции.

### Вариант № 6

1. Даны значения  $x, y, z$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения:

$$s = \frac{\min(\max(x, y), \max(y, z))}{\max(y, z)}.$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, суммы цифр которых являются нечетным числом. Проверку, является ли сумма цифр данного числа нечетной, сделайте с помощью процедуры пользователя.

3. Вычислите значение выражения  $C = A! + B!$ . Вычисление значения факториала оформите в виде пользовательской процедуры или функции.

4. Найдите такое число Фибоначчи, которое больше заданного числа  $Z$ . Вычисление каждого числа Фибоначчи оформите в виде пользовательской процедуры или функции.

### Вариант № 7

1. Даны значения  $x, y, z$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения:

$$r = \frac{\max(\min(x, y), z)}{3}.$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, которые являются степенью двойки. Проверку, является ли число степенью двойки, сделайте с помощью процедуры пользователя.

3. Даны три числа  $A, B, C$ . Найдите наибольший общий делитель этих чисел. Определение наибольшего общего делителя двух чисел оформите в виде пользовательской процедуры или функции.

4. На каком месте в ряду будет стоять число Фибоначчи, которое меньше заданного числа  $Z$ . Вычисление каждого члена Фибоначчи оформите в виде пользовательской процедуры или функции.

### Вариант № 8

1. Даны значения  $x, y, z$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения:

$$q = \frac{\max(x + y + z, x * y * z)}{\min(x + y + z, x * y * z)}.$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, которые оканчиваются на 3. Проверку, оканчивается ли число на тройку, сделайте с помощью процедуры пользователя.

3. Даны три числа, могут ли эти числа быть сторонами треугольника. Если треугольник существует, то вычислите длины высот этого треугольника. Вычисление длины высоты треугольника оформите в виде пользовательской процедуры или функции.

4. Вычислите значение выражения  $y = (n! + a!)/p$ . Вычисление факториала оформите в виде пользовательской процедуры или функции.

### Вариант № 9

1. Даны значения  $x, y, z$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения:

$$p = \frac{|\min(x, y) - \max(y, z)|}{2}$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, которые делятся на данное число  $K$ . Проверку, делится ли данное число на  $K$ , сделайте с помощью процедуры пользователя.

3. Даны три числа, могут ли эти числа быть сторонами треугольника. Если треугольник существует, то вычислите длины биссектрис этого треугольника. Вычисление длины биссектрисы треугольника оформите в виде пользовательской процедуры или функции.

4. Вычислите значение выражения  $y = e / (n! + a!)$ . Вычисление факториала оформите в виде пользовательской процедуры или функции.

### Вариант № 10

1. Даны значения  $x, y, z$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения:

$$n = \frac{\min(x + y, y - z)}{\max(x, y)}.$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, которые при делении на данное число  $K$ , дают в остатке число 4. Проверку, равен ли остаток четырем при делении данного числа на  $K$ , сделайте с помощью процедуры пользователя.

3. Даны три числа, могут ли эти числа быть сторонами треугольника. Если треугольник существует, то вычислите длины медиан этого треугольника. Вычисление длины медианы треугольника оформите в виде пользовательской процедуры или функции.

4. Вычислите значение выражения  $y = (z! - a!)/(k - m)$ . Вычисление факториала оформите в виде пользовательской процедуры или функции.

### Вариант № 11

1. Даны значения  $x, y, z$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения:

$$m = \frac{\max(x, y, z)}{\min(x, y)} + 5.$$



2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, которые являются делителями последнего числа  $B$ . Проверку, делится ли число  $B$  на текущее число, сделайте с помощью процедуры пользователя.

3. Даны координаты трех точек на плоскости. Выясните, лежат ли эти точки на одной прямой. Вычисление расстояния между двумя точками оформите в виде пользовательской процедуры или функции.

4. Вычислите значение выражения  $y = a!/(m + n) + b!$  Вычисление факториала оформите в виде пользовательской процедуры или функции.

### Вариант № 12

1. Даны значения  $x, y, z$ . Напишите функции пользователя  $\min$  и  $\max$  для вычисления значения следующего выражения

$$q = \frac{\max(x - y - z, x * y * z)}{\min(x + y + z, x * y * z)}.$$

2. Дан натуральный отрезок чисел  $[A, B]$ . Найдите те числа этого отрезка, которые в своей записи имеют цифру 7. Проверку, имеет ли данное число в своей записи цифру 7, сделайте с помощью процедуры пользователя.

3. Вычислите значение выражения  $C = A!/(K - B)!$  Вычисление значения факториала оформите в виде пользовательской процедуры или функции.

4. Вычислите значение выражения  $y = 1 / x^k + b$ . Вычисление значения степени оформите в виде пользовательской процедуры или функции.

## СТРУКТУРИРОВАННЫЙ ТИП ДАННЫХ. ОДНОМЕРНЫЕ МАССИВЫ

*Цель работы:* овладеть практическими навыками работы с массивами, вводом, выводом их значений, закрепить навыки по организации программ циклической структуры.

### Теоретические сведения

#### *Понятие массива в языке программирования PascalABC*

Массивами пользуются тогда, когда необходимо обработать большой набор однотипных данных. Массив в языке Паскаль – это набор фиксированного числа некоторых значений, которые называются компонентами (элементами). Все компоненты должны быть *одного типа*, который называют типом элементов или базовым типом. Каждый элемент массива имеет свой *порядковый номер* (индекс). Каждому конкретному массиву пользователь в своей программе дает имя, которое называют полной переменной. Каждый элемент массива обозначается полной переменной, и в квадратных скобках указывается номер этого элемента в массиве. Например,  $a[2]$  – элемент массива  $a$ , который стоит на втором месте;  $b[i + k - 1]$  – элемент массива  $b$ , который стоит на  $i + k - 1$  месте. Количество элементов в массиве определяет его размер.

Количество индексов каждого элемента массива определяет размерность массива. Если каждый элемент массива имеет только один индекс, то такие массивы называются одномерными. Если каждый элемент массива имеет два индекса, то такие массивы называют двумерными. Данные типа массив относятся к классу структурных данных.

#### *Объявление одномерного массива в программе*

Одномерный массив объявляется в программе двумя способами:

*1 способ:*

Var имя\_массива : Array [n..m] Of тип\_элементов;

2 способ:

Type имя\_типа=Array [n..m] Of тип\_элементов;

Var имя\_массива : имя\_типа;

Слова *Type* (тип), *Array* (ряд), *Of* (из) – служебные слова. Имя типа и имя массива дает пользователь. В квадратных скобках указывается интервал изменения индексов элементов массива, поэтому  $n$ ,  $m$  обязательно должны быть константами порядкового типа. Тип элементов называется базовым типом массива и может быть любым типом языка ПаскальABC, кроме файлового типа.

### ***Ввод элементов одномерного массива с клавиатуры и вывод на экран***

**Пример 1.** Составьте блок-схему (рис. 8.1) и напишите программу решения следующей задачи: с клавиатуры вводятся  $n \leq 50$  вещественных чисел. Найдите сумму отрицательных элементов данного массива.

```
Program Primer_1;
Type mas = Array[1..50] of Real;
Var a:mas; s:Real; i,n:Integer;
Begin
  Writeln('Ввести размер массива, т.е.количество элементов
массива');
  Readln(n);
  {Ввод самих элементов массива с клавиатуры}
  For i:=1 to n do
    Begin
      Writeln('Ввести a[',i,']');
      Readln(a[i]);
    End;
  {Вывод элементов массива на экран: все элементы в одну
строку}
  Writeln('Массив a:');
  For i:=1 to n do Write(a[i]:6:2);
  Writeln;
  {Вычисление суммы}
```

```

s:=0;
For i:=1 to n do if a[i]<0 Then s:=s+a[i];
{ВЫВОД ОТВЕТА}
Writeln('Сумма=',s:8:2);
End.

```

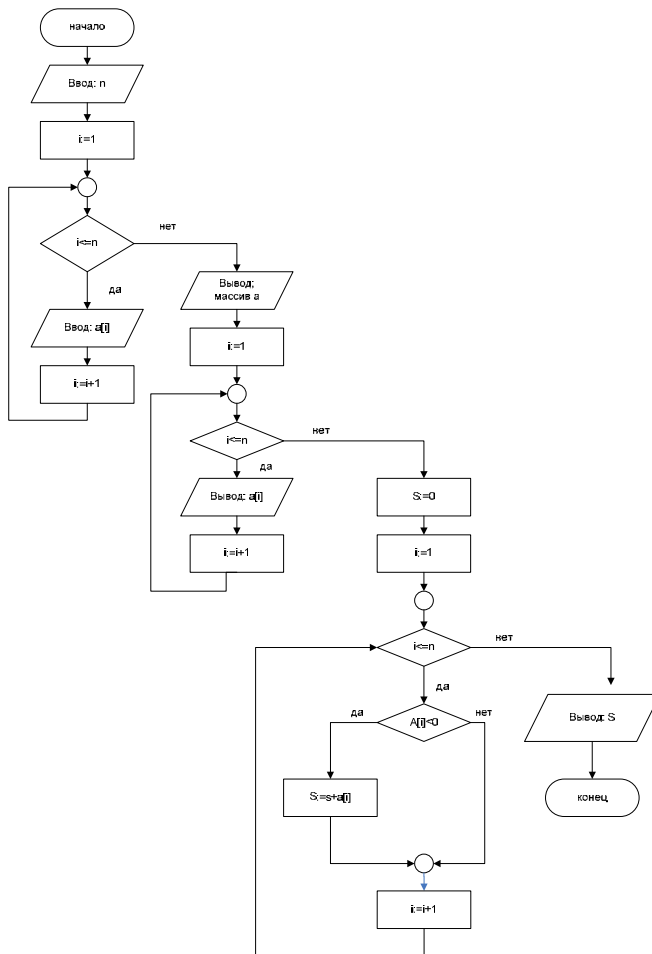


Рис. 8.1. Блок-схема к примеру 1

### *Задание элементов одномерного массива с помощью функции случайных чисел*

В языке ПаскальABC можно получать числа с помощью функции случайных чисел Random. Применяя функцию в разных видах, можно получить число из заданного интервала. В таблице приведены некоторые варианты записи функции случайных чисел.

Изучая варианты, представленные в табл. 8.1, можно сделать вывод, что в общем случае существуют только два выражения  $a + \text{Random}(b - a)$  для получения последовательности целых чисел и  $a + (b - a) * \text{Random}$  для получения случайных последовательностей действительных чисел в диапазоне  $[a, b)$ . Варьируя  $a$  и  $b$ , можно получить все желаемые диапазоны.

Таблица 8.1

Тип	Формат	Промежуток	Выражение, содержащее функцию Random
Целое	Положительное	$[a, b)$	$a + \text{Random}(b - a)$
	Отрицательное	$[a, b)$	$-(a + \text{Random}(b - a))$
	Целое со знаком	$[a, b)$	$(a + \text{Random}(b - a)) - (a + \text{Random}(b - a))$
Действительное	Положительное	$[0, 1)$	Random
	Положительное	$[a, b)$	$a + (b - a) * \text{Random}$
	Действительное со знаком	$[a, b)$	$(a + (b - a) * \text{Random}) - (a + (b - a) * \text{Random})$

Элементы массива можно задавать с помощью функции случайных чисел.

**Пример 2.** Составьте блок-схему (рис. 8.2) и напишите программу решения следующей задачи: одномерный массив, который содержит не более 20 вещественных чисел, задайте с помощью функции случайных чисел. Найдите номер последнего элемента, кратного данному числу  $x$ . Если таких элементов нет, выведите об этом сообщение.

**Решение.** Будем по порядку просматривать все элементы массива. Как только найдется элемент, который делится на  $x$  без остатка, необходимо зафиксировать его номер в массиве. Для этого введем переменную  $m$ , в которой сохраним номер найденного элемента.

Чтобы зафиксировать, нашелся или нет элемент в массиве кратный  $x$ , введем переменную  $k$ . Этой переменной перед

просмотром всех элементов массива дадим значение, например, *True*. Когда найдется нужный элемент, то значение переменной *k* поменяем на *False*. Если же в массиве нужных элементов не найдется, то переменная *k* не изменит свое значение и останется равной *True*. Такие переменные в программе называются флажками, дополнительными переменными, флюгерами.

```

Program Primer_2;
Type mas=Array[1..20] of Integer;
Var a:mas;
    i,n,m,x:Integer;
    k:Boolean;
{k – необходима для вывода сообщения, что искомым чисел нет}
Begin
    Writeln('Ввести число x');
    Readln(x);
    Writeln('Ввести размер массива');
    Readln(n);
    {Ввод элементов массива с помощью функции случайных чисел}
    For i:=1 to n do a[i]:=Random(100);
    {Вывод элементов массива на экран}
    Writeln('Массив a:');
    For i:=1 to n do Write(a[i]:8:2);
    Writeln;
    {Решение задачи}
    k:=True;
m:=0;
    For i := 1 to n do
        If a[i] mod x=0
            Then begin
                m:=i;
                k:= False;
            end;
    {Вывод ответа на экран}
    If k=True
        Then Writeln('элементов кратных', x, ' нет')
        Else Writeln('Последний элемент кратный ',x,'=',a[m]:8:2,' его
номер=',m);
    End.

```

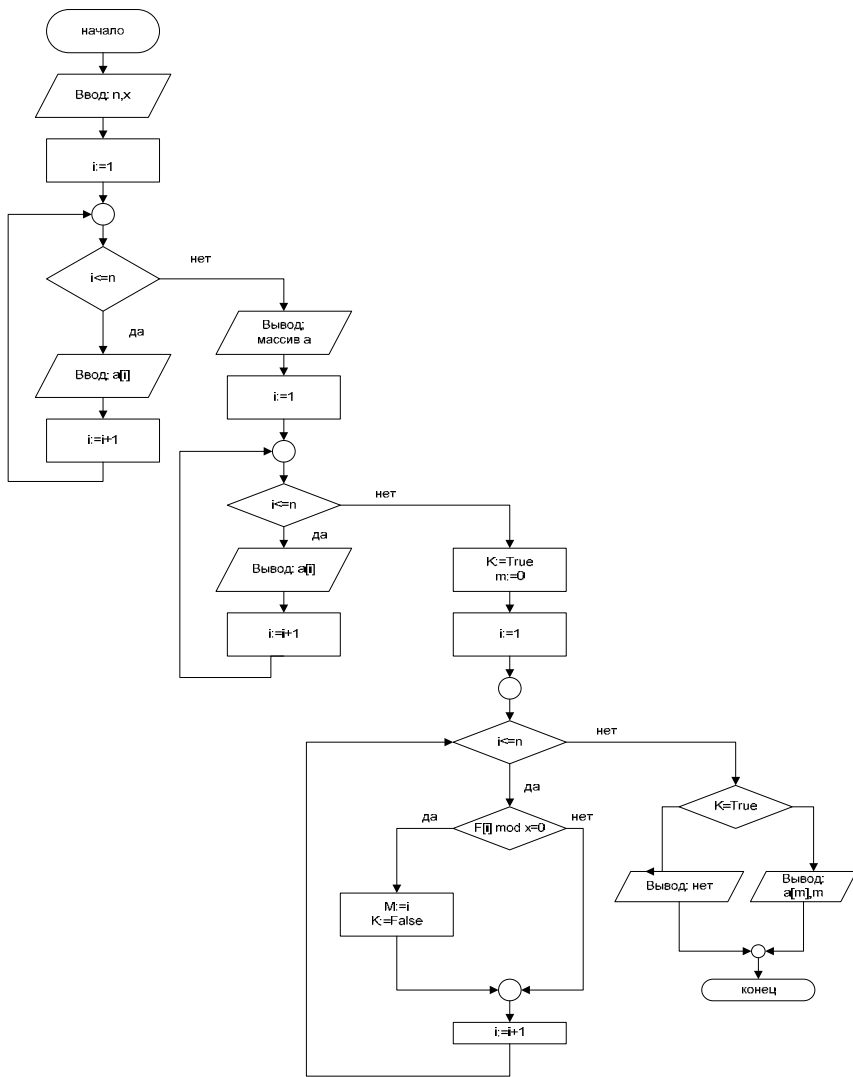


Рис. 8.2. Блок-схема к примеру 2

**Пример 3.** Составьте блок-схему (рис. 8.3) и напишите программу решения следующей задачи: одномерный массив  $x$ , который содержит не более 40 вещественных чисел, задайте с помощью функции случайных чисел. Для каждого элемента массива  $x$  вычислите значение  $y = (x^2 + 5x - 6) / (x^2 + 4)$  и составьте из значений  $y$  другой массив.

```

Program Primer_3;
Var x,y: Array[1..40] of Real; i,n,m:Integer;
Begin Writeln('Ввести размер массива'); Readln(n);
      {Ввод элементов массива с помощью функции случайных чисел}
      For i:=1 to n do x[i]:=100*(Random-Random);
      {Вывод элементов массива x на экран}
      Writeln('Массив x:');
      For i:=1 to n do Write(x[i]:8:2); Writeln;
      {Составление массива y и одновременно вывод этого массива на
      экран}
      Writeln('Массив y:');
      For i:=1 to n do
        Begin y[i]:=(x[i]*x[i]+5*x[i]-6)/(x[i]*x[i]+4); Write(y[i]:8:2);
      End;
      End.

```



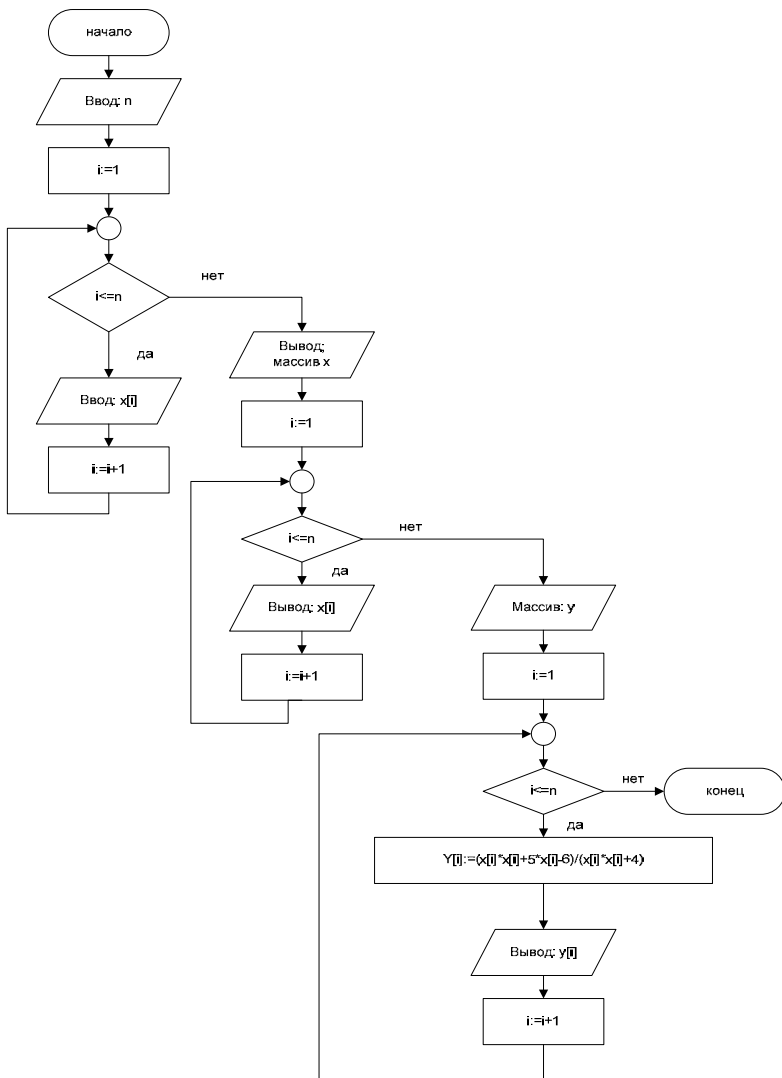


Рис. 8.3. Блок-схема к примеру 3

## *Процедуры ввода и вывода элементов одномерного массива*

Поскольку в задачах с использованием массивов всегда нужно вводить элементы массива и этот массив на экран, создадим процедуры ввода элементов массива и вывода на экран. Этими процедурами будем пользоваться во всех остальных программах.

**Пример 4.** Составьте блок-схему (рис. 8.5) и напишите программу решения следующей задачи: одномерный массив  $x$ , который содержит не более 40 вещественных чисел, задайте с помощью функции случайных чисел. Найдите максимальный по величине элемент и укажите его номер в массиве.

**Решение.** Решение задачи разобьем на несколько частей:

1. Ввод элементов массива.
2. Вывод элементов массива на экран.
3. Нахождение максимального элемента и вывод его на экран.

Каждую из этих частей можно представить как процедуру пользователя. В основной программе нужно будет записать только обращение к этим процедурам в перечисленном порядке.

Максимальный элемент будем находить следующим образом:

1. Сначала за максимум примем первый элемент массива (зафиксируем это так:  $\max := a[\text{первый}]$ ) и зафиксируем его номер в переменной  $\text{nom} := 1$ .

2. Затем каждый следующий элемент массива будем сравнивать с тем, который находится в ячейке  $\max$ . Если элемент массива больше того элемента, который находится в ячейке  $\max$ , то значение ячейки  $\max$  меняем на значение большего элемента и значение ячейки  $\text{nom}$  также меняем на номер этого большего элемента.

3. После просмотра всех элементов массива в ячейке  $\max$  будет находиться самый большой элемент, а в ячейке  $\text{nom}$  – его номер.

Блок-схемы изображаются отдельно для каждой части задачи.

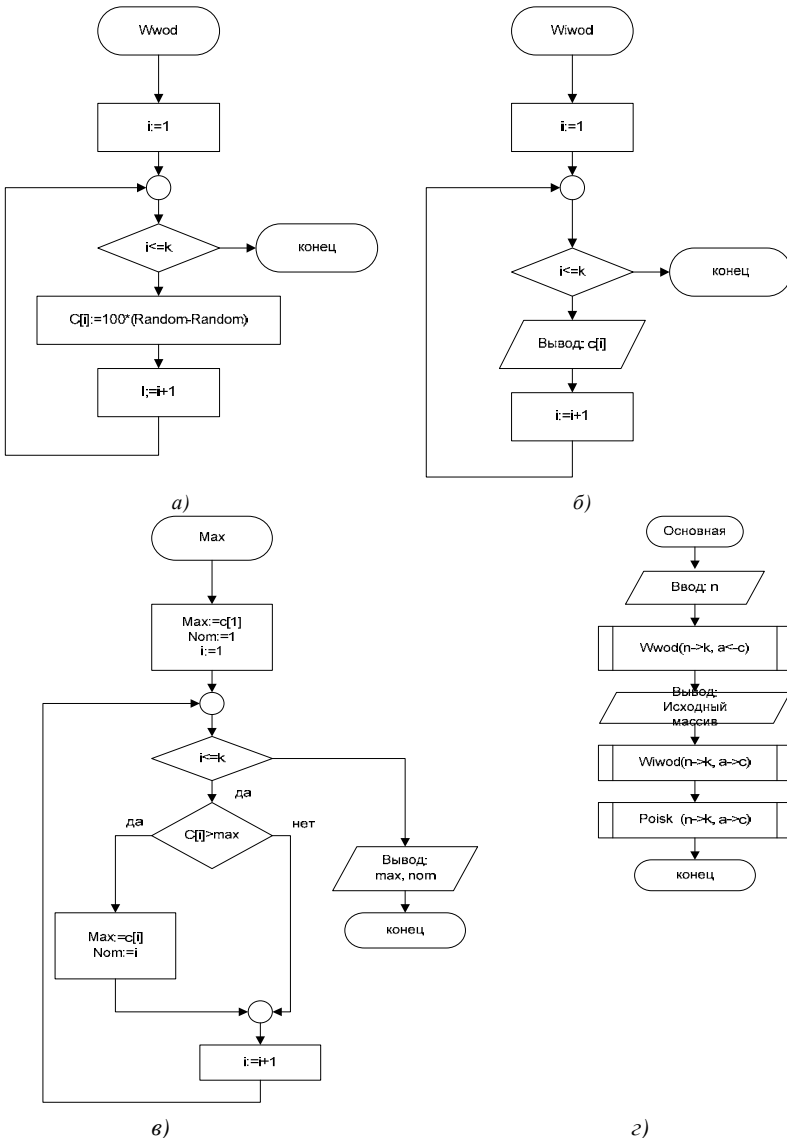


Рис. 8.5. Блок-схема к примеру 4:  
 а – блок-схема задания элементов массива;  
 б – блок-схема вывода элементов массива на экран;  
 в – блок-схема нахождения максимального элемента в массиве;  
 г – блок-схема основной программы

```

Program Primer_4;
{Раздел объявлений}
Type mas=Array[1..40] of Real;
      {Процедура ввода элементов массива}
Procedure Wwod(k:Integer; Var c:mas);
  {k – входной формальный параметр, который обозначает
  количество элементов в массиве, c – выходной формальный
  параметр, который обозначает массив}
  Var i:Integer; {i – локальный параметр}
  Begin
    For i:=1 to k do c[i]:=100*(Random-Random);
  End;
      {Процедура вывода элементов массива на экран}
Procedure Wuwod(k:Integer; c:mas);
  Var i:Integer;
  Begin
    Writeln('Массив :');
    For i:=1 to k do Write(c[i]:6:2);
    Writeln;
  End;
      {Процедура поиска максимального элемента в массиве}
Procedure Poisk(k:Integer; c:mas);
  Var i:Integer; max:Real;
  Begin
    max:=c[1]; nom:=1;
    For i:=1 to k do If max<c[i] Then Begin max:=c[i];nom:=i;End;
    Writeln('Максимальный элемент=',c[nom]:6:2,'его номер=',nom:3);
  End;
      {Основная программа }
  Var a:mas;
      n:Integer; {a, n – фактические параметры}
  Begin
    Writeln('Ввести количество элементов массива');
    Readln( n );
    Wwod(n,a);
    Writeln('Исходный массив');
    Wuwod(n,a);

```

```
Poisk(n,a);  
End.
```

### Примеры выполнения заданий

**Пример 5.** Составьте блок-схему (рис. 8.6) и напишите программу решения следующей задачи: дан одномерный массив, который содержит не более 60 символов. Ответьте на вопрос, какой символ встречается в данном массиве чаще других.

**Решение.** Решение задачи будет состоять из следующих частей:

1. Ввод элементов массива.
2. Вывод элементов массива на экран.
3. Подсчет, сколько раз встречается каждый символ в данном массиве и одновременно поиск максимального числа появлений элемента.

Для этого количество появлений элемента в массиве примем за  $p := 0$ .

Затем берем первый символ массива и подсчитываем, сколько раз он встречается в массиве. Для этого заводим новый счетчик  $t := 0$ . В ячейку  $max$  отправим первый элемент массива и значение ячейки  $max$  будем сравнивать со всеми остальными элементами массива. Если находится одинаковый с ним символ, то счетчик  $t := t + 1$ . После просмотра всех элементов массива сравним  $p$  и  $t$ , если  $t > p$ , то  $p := t$  и  $max := c[1]$ .

Теперь переходим ко второму элементу массива и проделываем с ним те же операции, что и с первым. Перебрав таким образом все элементы массива, получим, что в ячейке  $p$  находится максимальное число встреч того элемента, который находится в  $max$ .

Все части программы оформим в виде процедур пользователя.

```
Program a5;  
Type mas=array[1..60] of Char;  
Procedure Wwod(k:integer; Var c:mas);  
Var i:Integer;  
Begin  
Writeln('Ввести символы массива в виде строки');  
For i:=1 to k do Read(c[i]);
```

```

End;
Procedure Wiwod(k:Integer; c:mas);
Var i:Integer;
Begin
Writeln('Массив: a');
For i:=1 to k do Write(c[i]:3);
End;
Procedure Poisk(k:Integer; c:mas);
Var i,j,p,t:Integer; max:char;
Begin
p:=0;
For i:=1 to k do
  Begin
  t:=0;
  For j:=1 to k do If c[i]=c[j] Then t:=t+1;
  If t>p then begin p:=t; max:=c[i]; end;
  end;
Writeln('Чаще других встречается символ--',max);
  end;
{Основная программа}
Var n:Integer; a:mas;
Begin
Writeln('Ввести количество элементов'); Readln(n);
Wwod(n,a);
Writeln('Исходный массив');
Wiwod(n,a);
Poisk(n,a);
End.

```

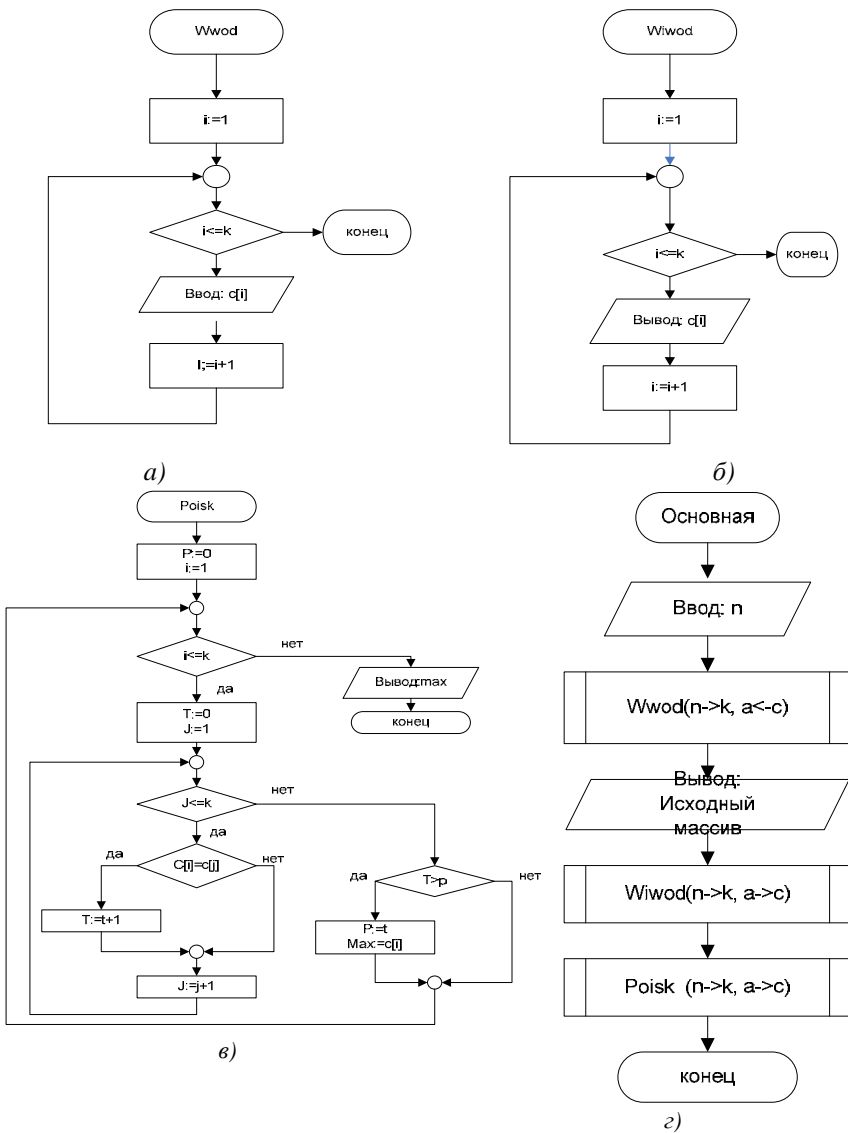


Рис. 8.6. Блок-схема примера 5:  
 а – блок-схема ввода элементов массива;  
 б – блок-схема вывода элементов массива на экран;  
 в – блок-схема поиска максимального элемента;  
 г – блок-схема основной программы

**Пример 6.** Составьте блок-схему (рис. 8.7) и напишите программу решения следующей задачи: дан массив целых чисел. Составьте новый массив из простых чисел старого массива.

**Решение.** Как обычно разобьем решение задачи на следующие подзадачи:

1. Ввод элементов массива с использованием функции случайных чисел.
2. Вывод элементов массива на экран в строку.
3. Создание нового массива.

Для каждой части задачи создадим процедуру пользователя. Ввод и вывод элементов массива встречается в каждой задаче с небольшими изменениями, поэтому в этой задаче блок-схемы для этих частей не приводятся.

Разберем третью часть задачи.

Сначала отметим, что в исходном массиве не все числа будут простыми. Если в исходном массиве  $n$  элементов, то в новом массиве элементов будет  $m \leq n$ . Значит индексы элементов в новом массиве будут изменяться по-другому, не так, как в старом массиве. Чтобы создать новый массив, необходимо просмотреть все элементы старого массива, то есть организовать цикл `For i := 1 to n do Begin ... End;  $a[i]$  – это элемент старого массива стоящий на  $i$ -ом месте. Нужно определить, является ли этот элемент простым числом.`

Число является простым, если оно имеет только два делителя: 1 и само это число. Чтобы определить, является ли число простым, напишем специальную подпрограмму. В цикле, приведенном выше, каждое число  $a[i]$  будем отправлять в подпрограмму и из подпрограммы будем получать ответ, простое это число или нет.

Если число простое, то его нужно отправить в новый массив. Пусть индексы элементов нового массива будут  $j$ . Тогда перед циклом по  $I$  необходимо присвоить  $j := 0$ . Как только из подпрограммы придет сообщение, что число  $a[i]$  простое, индекс  $j := j + 1$  и  $b[j] := a[i]$ . Таким образом составится новый массив, в котором будет  $j$  элементов. Массив необходимо вывести на экран, используя ту же процедуру вывода, что и для старого массива.



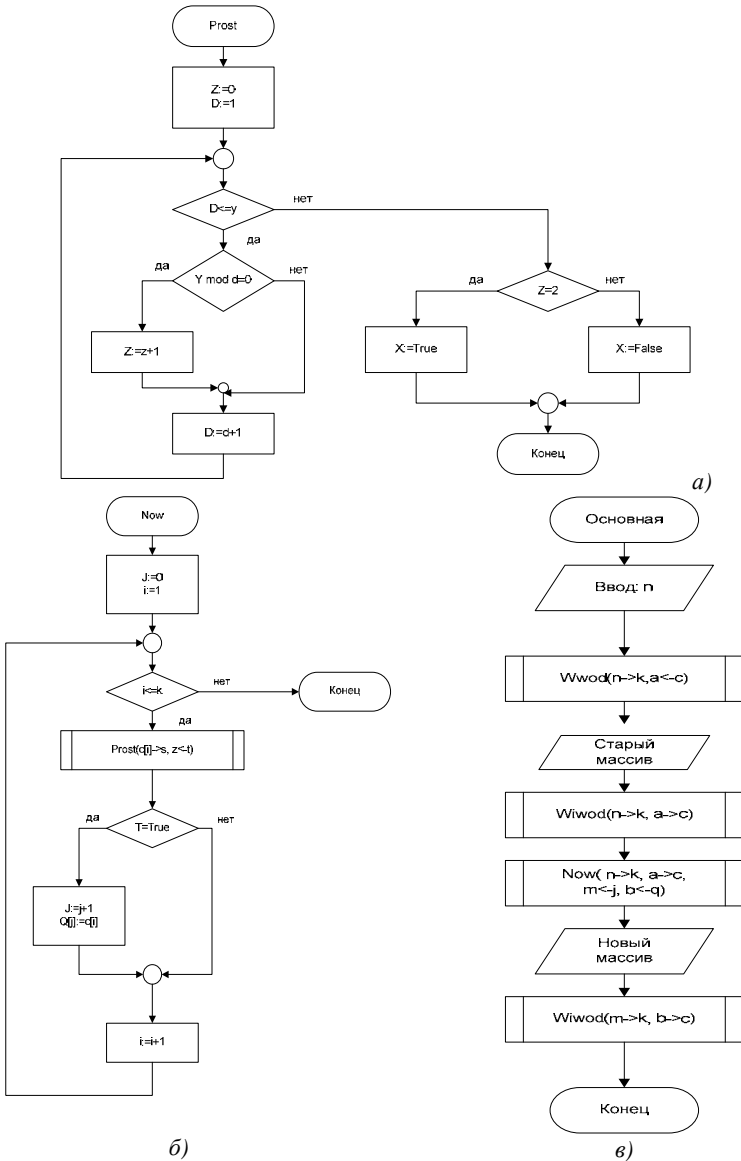


Рис. 8.7. Блок-схема к примеру 6:  
 а – блок-схема определения простого числа;  
 б – блок-схема составления массива из простых чисел;  
 в – блок-схема основной программы

```

Program Primer_6;
Type mas=array[1..50] of Integer;
Procedure wwod(k:Integer; Var c:mas);
Var i:Integer;
Begin
For i:=1 to k do c[i]:=Random(100);
End;
Procedure Wiwod(k:Integer; c:mas);
Var i:Integer;
Begin
For i:=1 to k do Write(c[i]:4);
Writeln;
End;
Procedure Prost(y:Integer; Var x:Boolean);
Var z,d:Integer;
Begin
z:=0;
For d:=1 to y do If y mod d=0 Then z:=z+1;
If z=2 Then x:=True Else x:=False;
End;
Procedure Now(k:Integer; c:mas; Var j:Integer; Var q:mas);
Var i:Integer; t:boolean;
Begin
j:=0;
For i:=1 to k do
  Begin
  Prost(c[i],t);
  If t=True Then begin j:=j+1; q[j]:=c[i]; end;
  End;
End;
{Основная программа}
Var a,b:mas; n,m:Integer;
Begin
Writeln('Ввести количество элементов массива, то есть размер
массива');
Readln(n);
Wwod(n,a);
Writeln('Исходный массив');

```

```
Wiwod(n,a);  
Now(n,a,m,b);  
Writeln('Полученный массив');  
Wiwod(m,b);  
End.
```

### Контрольные вопросы

1. Что такое массив?
2. Что имеет каждый элемент массива?
3. Как называют имя массива?
4. Как обозначается каждый элемент массива?
5. Что такое размерность массива?
6. Как в программе объявляется одномерный массив? Перечислите все способы.
7. Для чего предназначена функция Random?
8. Напишите алгоритм нахождения максимального элемента массива.
9. Type  $T = \text{array } [T1] \text{ of } T2$ , где  $T - \dots$ ;  $T1 - \dots$ ;  $T2 - \dots$   
Вставьте необходимые слова вместо многоточия.
10. Var  $T: \text{array } [T1] \text{ of } T2$ , где  $T - \dots$ ;  $T1 - \dots$ ;  $T2 - \dots$   
Вставьте необходимые слова вместо многоточия.
11. Type  $\dots = \dots [T1] \dots T2$ , где  $T - \text{имя типа массива}$ ;  $T1 - \dots$ ;  
 $T2 - \dots$  Вставьте необходимые слова вместо многоточия.
12.  $T = \text{array } [1..10] \text{ of integer}$ ; В каком подразделе можно описать массив таким образом?
13.  $C: \text{array } [1..10] \text{ of real}$ ; В каком подразделе можно описать массив таким образом?
14.  $\text{mas}: \text{array } [1..10] \text{ of byte} = (1, 3, 5, 50, 7, 240, 13, 143, 196, 87)$ ; В каком подразделе можно описать массив таким образом?
15.  $\dots B = \text{array } [1..20] \text{ of real}$ ;  $\dots A: \text{array } [1..30] \text{ of } B$ ; Вставьте названия подразделов.
16. Охарактеризуйте данный массив:  $B: \text{array } [1..20] \text{ of real}$ ;
17. Охарактеризуйте данный массив:  $F: \text{array } [1..15] \text{ of string}[30]$
18. Охарактеризуйте данный массив:  $G: \text{array } [\text{Red, Yellow, Green}] \text{ of char}$ ;
19. Охарактеризуйте данный массив:  $L: \text{array } [\text{byte}] \text{ of Boolean}$ ;
20. Как осуществляется ввод элементов одномерного массива?
21. Как осуществляется вывод элементов одномерного массива?

## Задания для самостоятельного выполнения

1. В каждой задаче варианта составьте блок-схему и напишите программу решения задач.

2. Оформите отчет по лабораторной работе, который должен состоять из следующих разделов: номера лабораторной работы и ее названия, условия каждой задачи, ее блок-схемы и программы.

3. В каждой задаче используйте процедуры ввода и вывода элементов массива.

### Вариант № 1

1. Дан одномерный массив, который содержит не более 50 вещественных чисел. Найдите сумму элементов, которые расположены на местах с нечетными номерами.

2. Дан одномерный массив, который содержит не более 40 целых чисел. Найдите второй по величине элемент и поменяйте его местами с последним элементом массива.

3. Дан одномерный массив, который содержит не более 60 английских букв, среди которых есть одинаковые. Получите все элементы, которые входят в массив по одному разу.

4. Даны оценки, полученные на экзамене по информатике студентами одной группы, по десятибалльной системе. Определите сколько студентов получили 10 баллов, сколько – 9, сколько – 8, сколько – 7, сколько – 6, сколько – 5, сколько – 4, сколько – 3. При выводе ответа слово «студент» должно стоять в соответствующей форме.

### Вариант № 2

1. Дан одномерный массив, который содержит не более 50 целых чисел. Найдите сумму элементов, которые делятся на данное число  $X$ .

2. Дан одномерный массив, который содержит не более 40 целых чисел. Найдите минимальный элемент и поменяйте его местами с тем элементом, который стоит по середине массива.

3. Дан одномерный массив, который содержит не более 60 вещественных чисел, среди которых есть положительные, отрицательные и нули. Переставьте элементы массива так, чтобы сначала шли нули, потом отрицательные, а затем положительные элементы массива.

4. Дана температура воздуха за месяц и вводится номер месяца. Найдите среднюю температуру этого месяца. По номеру месяца

определите, сколько дней в этом месяце: может быть 31 день, 30 дней, 28 или 29 дней.

### **Вариант № 3**

1. Дан одномерный массив, который содержит не более 50 целых чисел. Найдите сумму элементов, у которых сумма делителей меньше данного числа  $X$ .

2. Дан одномерный массив, который содержит не более 40 целых чисел. Создайте новый массив, элементами которого являются суммы цифр каждого числа старого массива.

3. Дан одномерный массив, который содержит не более 60 английских букв. Получите новый массив, в котором данная буква заменена на символ «\_».

4. Имеется пять магазинов с разными названиями: ЦУМ, ГУМ, Немига, Восточный, Первомайский. В них имеется один и тот же товар, который пользуется большим спросом. Количество товара в каждом магазине задайте с помощью датчика случайных чисел. Определите, в какой магазин в первую очередь необходимо завести новую партию этого товара.

### **Вариант № 4**

1. Дан одномерный массив, который содержит не более 50 целых чисел. Найдите сумму отрицательных и сумму положительных элементов и сравните их по модулю.

2. Дан одномерный массив, который содержит не более 40 целых чисел. Создайте новый массив, элементами которого являются суммы делителей каждого числа старого массива

3. Дан одномерный массив, который содержит не более 60 английских букв. Получите новый массив, который будет содержать те элементы, которые входят в исходный массив более одного раза.

4. Имеется пять бензоколонок по разным адресам: ул. Володарского, ул. Партизанская, ул. Комсомольская, ул. Ленина, ул. Рокоссовского. На этих бензоколонках имеется бензин марки 95 в разных количествах. Количество бензина на бензоколонках задайте с помощью датчика случайных чисел. Необходимо заправить колонну машин, которая потребит 500 л этого бензина. По какому адресу расположена бензоколонка, на которую можно отправлять данную колонну машин?

### **Вариант № 5**

1. Дан одномерный массив, который содержит не более 50 целых чисел. Найдите и выведите те элементы, которые больше предыдущего.
2. Дан одномерный массив, который содержит не более 40 целых чисел. Создайте новый массив, элементами которого являются суммы первой и последней цифры каждого числа старого массива.
3. Дан одномерный массив, который содержит не более 60 английских букв. Получите новый массив, в котором буквы будут расставлены по алфавиту.
4. Есть группа спортсменов из  $n \leq 20$  человек. Приводится рост и вес каждого из них. Вес спортсмена считается нормальным, если от роста отнять 100 и полученное число отличается от веса не более чем на 3. Выведите номера тех спортсменов, чей вес превышает норму.

### **Вариант № 6**

1. Дан одномерный массив, который содержит не более 50 целых чисел. Замените каждый элемент с четным номером цифрой 2, а с нечетным номером – цифрой 5.
2. Дан одномерный массив, который содержит не более 40 целых чисел. Создайте новый массив, элементами которого являются среднее арифметическое двух соседних элементов. Последний элемент в новом массиве есть среднее арифметическое первого и последнего элемента старого массива.
3. Дан одномерный массив, который содержит не более 60 символов. Эти символы образуют некоторые слова, разделенные одним пробелом. Выведите каждое слово отдельно (то есть выведите слова в столбик).
4. Приводятся показатели производства работы  $n \leq 30$  рабочих. Определите номер рабочего, у которого показатель наибольший, и номер рабочего, у которого показатель второй по величине.

### **Вариант № 7**

1. Дан одномерный массив, который содержит не более 50 целых чисел. Замените каждый элемент массива произведением индексов соседних элементов, если нет соседних элементов, оставьте число без изменения.
2. Дан одномерный массив, который содержит не более 40 целых чисел. Создайте новый массив, элементами которого является 0, если число простое, и 7 – если число составное.

3. Дан одномерный массив, который содержит не более 60 символов. Символы образуют слова, которые друг от друга отделяются пробелом. Замените в словах букву «С» на букву «А».

4. Приводится рост учеников одного класса, где количество учеников  $n \leq 30$ . Определите номер ученика самого высокого и номер ученика второго по росту в классе.

### **Вариант № 8**

1. Дан одномерный массив, который содержит не более 50 целых чисел. Циклически сдвиньте элементы массива на  $K$  позиций влево.

2. Дан одномерный массив, который содержит не более 40 целых чисел. Вычислите сумму элементов, которые расположены между минимальным элементом и элементом, стоящим по середине.

3. Дан одномерный массив, который содержит не более 60 символов, которые образуют слова, разделенные одним пробелом. Ответьте на вопрос, есть ли среди этих слов данное слово.

4. Приводится среднесуточная температура воздуха за месяц. Выведите номера тех дней, когда среднемесячная температура была ниже среднесуточной температуры. Количество дней в месяце определите по его названию.

### **Вариант № 9**

1. Дан одномерный массив, который содержит не более 50 целых чисел. Циклически сдвиньте элементы массива на  $M$  позиций вправо.

2. Дан одномерный массив, который содержит не более 40 целых чисел. Найдите сумму элементов, которые расположены между максимальным элементом и элементом, стоящим на 7 месте.

3. Дан одномерный массив, который содержит не более 60 символов, образующих слова, разделенные пробелом. Замените слово «тата» на слово «ded».

4. В области 10 районов. Для каждого известны площади, засеянные пшеницей, и урожай, собранный с этих площадей. Определите, в каком районе средняя урожайность пшеницы была выше.

### **Вариант № 10**

1. Дан одномерный массив, который содержит не более 50 целых чисел. Замените каждый элемент суммой соседних индексов. Если соседних элементов нет, оставьте число без изменения.

2. Дан одномерный массив, который содержит не более 40 целых чисел. Найдите сумму отрицательных элементов, которые расположены между максимальным и минимальным элементом.

3. Дан одномерный массив, который содержит не более 60 символов. Ответьте на вопрос, есть ли среди данных символов сочетание «...».

4. В университете  $n \leq 10$  факультетов. Известен план приема студентов на каждый факультет и число поданных заявлений. Определите конкурс на каждый факультет.

### **Вариант № 11**

1. Дан одномерный массив, который содержит не более 50 целых чисел. Выведите только те элементы, которые меньше предыдущего.

2. Дан одномерный массив, который содержит не более 40 целых чисел. Вычислите количество элементов между максимальным и минимальным.

3. Дан одномерный массив, который содержит не более 60 символов. Сколько раз в данном массиве встречается сочетание символов «xx».

4. Дано длинное целое число, содержащее не более 100 цифр. Рассматривать цифры числа, как массив символов. Определите, есть ли в данном числе цифра 0, если есть, сколько раз она попадает в данном числе.

### **Вариант № 12**

1. Дан одномерный массив, который содержит не более 50 целых чисел. Найдите среднее арифметическое элементов, которые стоят на нечетных местах.

2. Дан одномерный массив, который содержит не более 40 положительных и отрицательных целых чисел. Найдите наименьший положительный элемент и его номер в массиве.

3. Дан одномерный массив, который содержит не более 60 символов. Ответьте на вопрос, есть ли среди этих символов подряд идущие одинаковые символы.

4. Дано длинное целое число, содержащее не более 100 цифр. Рассматривать цифры числа, как массив символов. Определите, есть ли в данном числе четные цифры и сколько их.



## Лабораторная работа № 9

### ДВУМЕРНЫЕ МАССИВЫ

*Цель работы:* Научиться работать с двумерными массивами при решении задач на языке ПаскальABC.

#### Теоретические сведения

Двумерные массивы представляют собой прямоугольные таблицы, которые состоят из строк и столбцов. Например,

$$\begin{matrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{matrix}$$

Двумерный массив отображает математическое понятие – матрица.

Каждый элемент двумерного массива имеет два индекса. Договорились всегда на первом месте писать индекс строки, а на втором – индекс столбца. Элемент двумерного массива, как и одномерного, обозначается полным именем массива, в квадратных скобках через запятую записываются индексы этого элемента. Элемент  $a[2, 3]$  – это элемент, записанный во второй строке и третьем столбце. Поскольку каждый элемент имеет два индекса, то при работе с двумерными массивами, как правило, используются вложенные циклы: один цикл по строкам и вложенный цикл по столбцам или наоборот в зависимости от задачи.

#### *Объявление двумерного массива*

Как и одномерный массив, двумерный можно объявить двумя способами:

*1 способ:*

Type имя типа=array[n..m,k..p] of тип элементов;

Var имя массива: имя типа;

2 способ:

var имя массива: array [n..m,k..p] of тип элементов,

где константы  $n$ ,  $m$  указывают интервал изменения индексов строк; константы  $k$ ,  $p$  – интервал изменения индексов столбцов, имя типа и имя массива задает пользователь, тип элементов – любой тип языка ПаскальABC, кроме файлового типа. Тип элементов массива называется базовым.

### ***Ввод элементов двумерного массива***

Задание элементов можно производить как путем ввода элементов с клавиатуры, так и путем задания с помощью функции случайных чисел.

Пусть объявлен массив:

```
Type tabl=array[1..10,1..10] of Integer;  
Var a:tabl; k, p, l, j : Integer;
```

Запишем процедуру ввода элементов двумерного массива с помощью функции случайных чисел:

```
PROCEDURE Wwod2 (k,p:Integer; Var a:tabl);  
Var i, j : Integer;  
BEGIN  
For i:=1 to k do  
  For j:=1 to p do a[i,j]:=Random(1000); {Целые числа из [0; 1000)}  
  END;
```

Процедура ввода элементов того же двумерного массива, с клавиатуры.

```
PROCEDURE Wwod1(k,p:integer; Var a:tabl);  
Var i, j : Integer;  
BEGIN  
For i:=1 to k do {Цикл по строкам}  
  For j:=1 to p do {Цикл по столбцам}  
    Begin  
      Writeln('Ввести элемент a[', I, ', ', j, ' ');  
      Readln(a[I, j]);  
    End;
```

```
END.
```

Блок-схема (рис. 9.1) ввода элементов двумерного массива

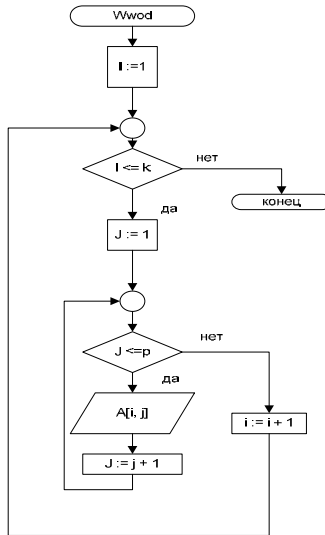


Рис. 9.1

### ***Вывод элементов двумерного массива на экран***

Обычно элементы двумерного массива выводят на экран в виде прямоугольной таблицы, чтобы были видны строки и столбцы. Запишем процедуру вывода массива, созданного в п. 2.

```

PROCEDURE Wiwod (k, p:Integer; a: tabl);
Var i, j : Integer;
BEGIN
For i := 1 to k do
  Begin
  For j := 1 to p do Write (a[ i, j]:6);
  Writeln;
  end;
END;
  
```

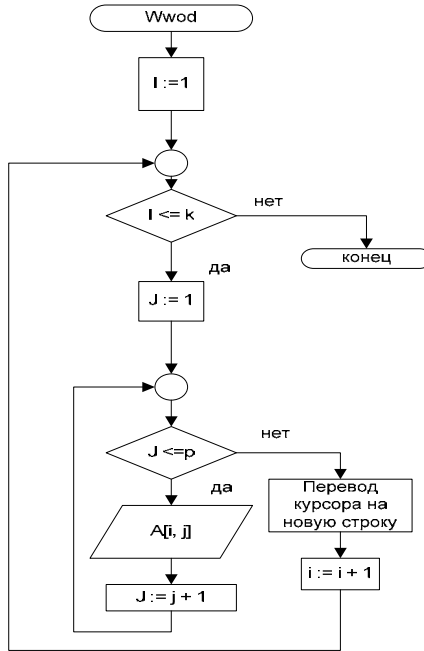


Рис. 9.2. Блок-схема процедуры Wwiod

### ***Свойства диагональных элементов***

Если количество строк равно количеству столбцов и равно  $n$ , то такие массивы называются квадратными. Элементы квадратного двумерного массива, у которых индекс строки равен индексу столбца образуют главную диагональ (если двумерный массив представить как квадрат, то линия с верхнего левого угла до нижнего правого есть главная диагональ). Поэтому можно записать, что элементы, лежащие на главной диагонали, – это элементы с индексами  $a[i, i]$  (если двумерный массив представить как квадрат, то линия с верхнего правого угла до нижнего левого есть побочная диагональ). Элементы  $a[i, j]$ , лежащие на побочной диагонали, имеют следующее свойство  $i + j = n + 1$ , где  $n$  – количество строк в массиве. Поэтому элемент, лежащий на побочной диагонали, можно представить  $a[i, n + 1 - i]$ . Этими свойствами часто пользуются при решении задач.

## Примеры выполнения заданий

**Пример 1.** Составьте блок-схему (рис. 9.3) и напишите программу решения следующей задачи: дан двумерный массив размером  $n \leq 10$  и  $m \leq 10$ . Найдите суммы элементов по строкам.

```
PROGRAM Primer_1;
Type tabl=Array [1..10, 1..10] of Real;
{-----}
PROCEDURE Wwod (k, p:Integer; Var a : tabl);
Var i, j : Integer;
BEGIN
For i := 1 to k do
For j := 1 to p do a[i,j]:=10*Random; {Вещественные числа [0; 10]}
END;
{-----}
PROCEDURE Wiwod ( k, p : Integer; a : tabl);
Var i, j : Integer;
BEGIN
For i := 1 to k do
  Begin
  For j := 1 to p do Write(a[i,j]:8:2);
  Writeln;
  End;
END;
{-----}
{Основная программа}
Var a: tabl; i, j, n, m: Integer; s: real;
BEGIN
Writeln('Ввести количество строк и столбцов');
Readln(n,m);
Wwod(n,m,a);
Writeln('Исходный массив A');
Wiwod(n,m,a);
Writeln('Суммы по строкам:');
For i := 1 to n do
  Begin
  S:=0;
```

```

For j := 1 to m do s:=s+a[I,j];
Writeln('сумма ', i, ' строки=', s:8:2);
End;
END.

```

Поскольку блок-схемы процедур ввода и вывода приведены выше, то здесь укажем только блок-схему (рис. 9.3) основной программы.

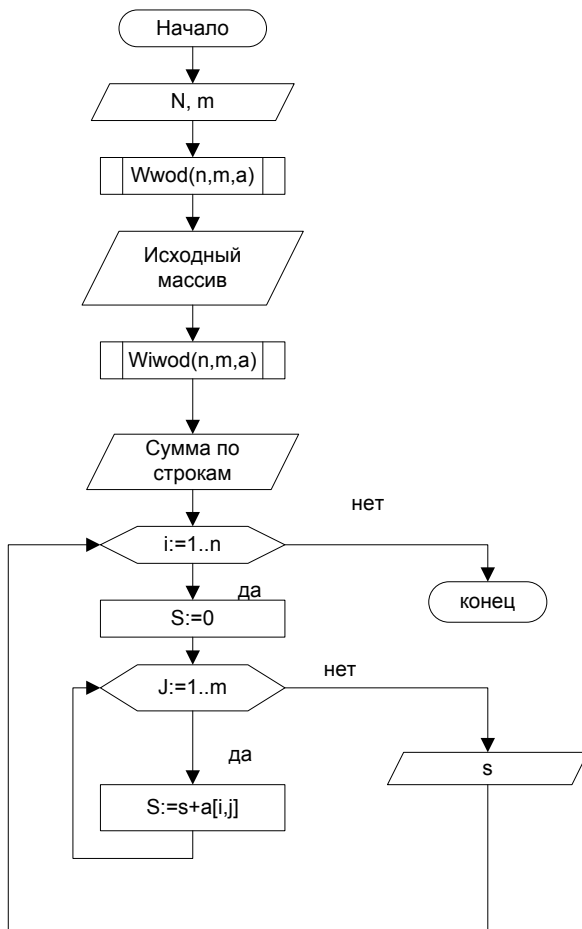


Рис. 9.3. Блок-схема основной программы примера 1

**Пример 2.** Составьте блок-схему (рис. 9.4) и напишите программу решения следующей задачи: дан двумерный массив целых чисел, который содержит не более 10 строк и не более 10 столбцов. Найдите и выведите номера тех столбцов, все элементы которых четные.

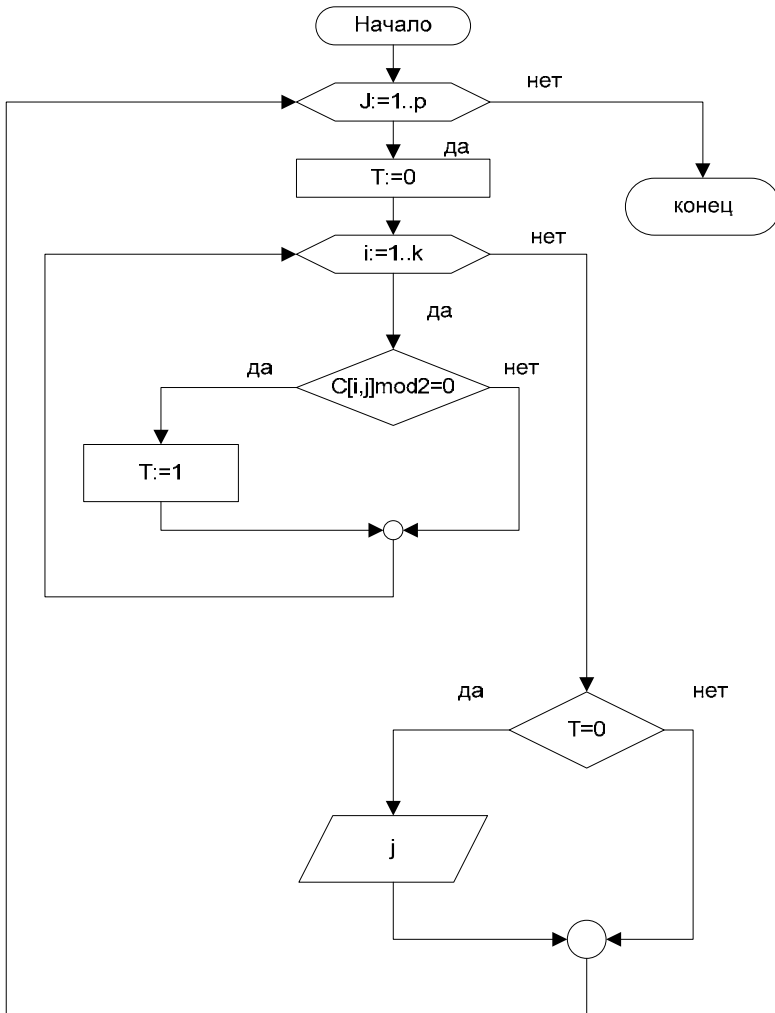


Рис. 9.4. Блок-схема процедуры Chet

```

Program Primer_2;
Type tabl=Array[1..10,1..10] of Integer;
{-----}
{Процедура ввода элементов двумерного массива}
Procedure Wwod2(k,p:Integer; Var x:tabl);
Var i,j : Integer;
Begin
For i := 1 to k do
  For j := 1 to p do
    X[i,j]:=Random(100);
  End;
{-----}
{Процедура вывода элементов двумерного массива на экран в
виде матрицы}
Procedure Vivod2(k,p:Integer; x:tabl);
Var i,j:Integer;
Begin
For i := 1 to k do
  Begin
  For j:=1 to p do Write(x[i,j]:5);
  Writeln;
  End;
End;
{-----}
{Нахождение тех столбцов, все элементы которых четные}
Procedure Chet(k,p:Integer; c:tabl);
Var i,j,t:Integer;
Begin
  For j:=1 to p do
    Begin
    t:=0;
    For i:=1 to k do If c[i,j] mod 2<>0 Then t:=1;
    If t=0 Then Writeln(' ',j:3,' столбце все элементы четные');
    End;
  End;
{-----}
{Основная программа}
Var n,m:Integer; a:tabl;

```



```

Begin
  Writeln('Ввести количество строк и столбцов');
  Readln(n,m);
  Randomize;
  Wwod2(n,m,a);
  Writeln('Исходный массив');
  Vivod2(n,m,a);
  Chet(n,m,a);
End.

```

**Пример 3.** Составьте блок-схему (рис. 9.5) и напишите программу решения следующей задачи: дан квадратный двумерный массив целых чисел размерностью  $N \leq 10$ . Найдите сумму элементов, которые находятся под побочной диагональю.

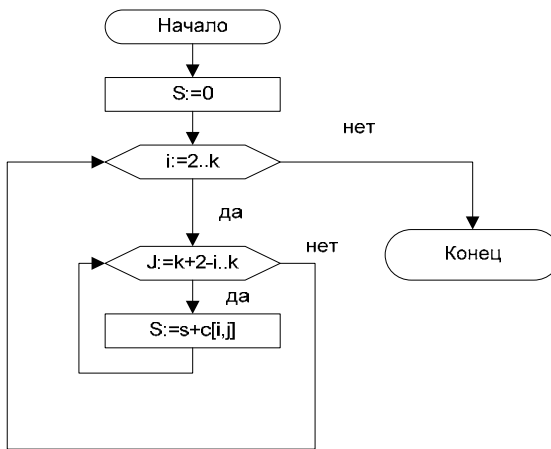


Рис. 9.5. Блок-схема процедуры нахождения суммы

```

Program Primer_3;
Type tabl=Array[1..10,1..10] of Integer;
{Процедура ввода элементов двумерного массива}
Procedure Wwod2(k,p:Integer; Var x:tabl);
Var i,j : Integer;
Begin

```

```

For i := 1 to k do
  For j := 1 to p do
    X[i,j]:=Random(100);
End;
{Процедура вывода элементов двумерного массива на экран в
виде матрицы}
Procedure Vivod2(k,p:Integer; x:tabl);
Var i,j:Integer;
Begin
For i := 1 to k do
  Begin
  For j:=1 to p do Write(x[i,j]:5);
  Writeln;
  End;
End;
{Нахождение суммы элементов, которые расположены под
побочной диагональю}
Procedure Sum(k:Integer; c:tabl);
Var i,j, s:Integer;
Begin
s:=0;
For i:=2 to k do
  For j:=k+2-i to k do s:=s+c[i,j];
Writeln('Сумма элементов под побочной диагональю=',s:6);
End;
{Основная программа}
Var n,m:Integer; a:tabl;
Begin
  Writeln('Ввести количество строк и столбцов');
  Readln(n,m);
  Randomize;
  Wwod2(n,m,a);
  Writeln('Исходный массив');
  Vivod2(n,m,a);
  Sum(n,a);
End.

```

**Пример 4.** Составьте блок-схему (рис. 9.6 а, б) и напишите программу решения следующей задачи: дан двумерный массив целых чисел, который содержит не более 10 строк и не более 10 столбцов. Массив выведите на экран в виде матрицы. Поменяйте  $w$ -ю и  $p$ -ю строки, если  $w$  и  $p$  вводятся с клавиатуры.

```

Program Primer_4;
Type tabl=Array[1..10,1..10] of Integer;
{Процедура ввода элементов двумерного массива}
Procedure Wwod2(k,p:Integer; Var x:tabl);
Var i,j : Integer;
Begin
For i := 1 to k do
For j := 1 to p do
X[i,j]:=Random(100);
End;
{Процедура вывода элементов двумерного массива на экран в виде матрицы}
Procedure Vivod2(k,p:Integer; x:tabl);
Var i,j:Integer;
Begin
For i := 1 to k do
Begin
For j:=1 to p do Write(x[i,j]:5);
Writeln;
End;
End;
{Процедура обмена строк}
Procedure obmen(k,z,w,p:Integer;
Var c:tabl);
Var i,j,x:Integer;
For j:=1 to z do

```

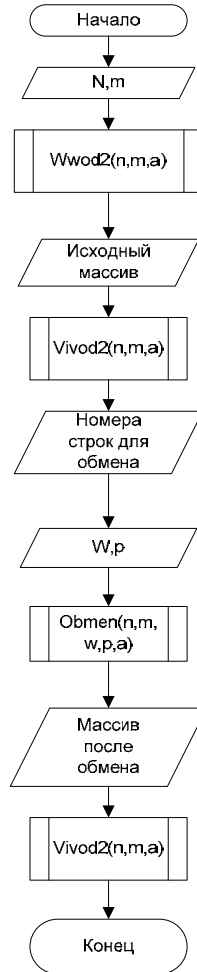


Рис. 9.6. Блок-схема к примеру 4:  
а – блок-схема основной программы

```

    Begin x:=c[w,j]; c[w,j]:=c[p,j];
c[p,j]:=x; End;
End;
{Основная программа}
Var a:tabl; n,m,w,p:Integer;
Begin
    Writeln('Ввести количество строк
и столбцов');
    Readln(n,m);
    Wwod2(n,m,a);
    Writeln('Исходный массив');
    Vivod2(n,m,a);
    Writeln('Ввести номера строк для
обмена');
    Readln(w,p);
    Obmen(n,m,w,p,a);
    Writeln('Массив после обмена');
    Vivod2(n,m,a);
End.

```

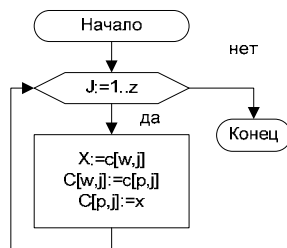


Рис. 9.6 (продолжение).  
Блок-схема к примеру 4:  
б – блок-схема процедуры Obmen

### Контрольные вопросы

1. Какое математическое понятие отображает двумерный массив?
2. Каким образом осуществляется доступ к одному элементу двумерного массива?
3. Что называется базовым типом двумерного массива?
4. Что может являться базовым типом двумерного массива?
5. Определите размер двумерного массива  $3 \times 4$ .
6. Что определяет размерность массива?
7. Что такое мощность массива?
8. Что может выступать в качестве индексов двумерного массива?
9. Как объявить массив из целых чисел с использованием раздела типов?
10. Как объявить массив из вещественных чисел с использованием раздела типов?

11. Как объявить массив из вещественных чисел с использованием раздела переменных?
12. Охарактеризуйте данный массив: `c:array[1..5,1..4] of Integer;`
13. Охарактеризуйте данный массив: `b:array[4..6,3..8] of Real;`
14. Какими способами осуществляется ввод элементов двумерного массива?
15. Запишите процедуру ввода элементов двумерного массива с клавиатуры.
16. Запишите процедуру ввода элементов двумерного массива из целых чисел с помощью функции случайных чисел.
17. Запишите процедуру ввода элементов двумерного массива из вещественных чисел с помощью функции случайных чисел.
18. Запишите процедуру вывода элементов двумерного массива в виде матрицы из целых чисел.
19. Запишите процедуру вывода элементов двумерного массива в виде матрицы из вещественных чисел.

### **Задания для самостоятельного выполнения**

1. В каждой задаче варианта составьте блок-схему и напишите программу решения задач.
2. Оформите отчет по лабораторной работе, который должен состоять из следующих разделов: номера лабораторной работы и ее названия, условия каждой задачи, ее блок-схемы и программы.
3. В каждой задаче варианта напишите программы с использованием процедур пользователя.

#### **Вариант № 1**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Найдите сумму элементов, которые расположены в строках с нечетными номерами.
2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Найдите максимальный по абсолютной величине элемент и поменяйте его местами с последним элементом массива.
3. Даны оценки, полученные на четырех экзаменах во время сессии студентами одной группы, по десятибалльной системе. Определите, сколько студентов не сдали сессию.

### **Вариант № 2**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Найдите сумму элементов, которые делятся на данное число  $X$ .

2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Найдите минимальный элемент и поменяйте его местами с тем элементом, который стоит в конце массива.

3. Даны оценки, полученные на четырех экзаменах во время сессии студентами одной группы, по десятибалльной системе. Определите, сколько студентов сдали сессию на 10 и 9 баллов.

### **Вариант № 3**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Найдите сумму элементов, у которых сумма делителей меньше данного числа  $X$ .

2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Создайте новый массив, элементами которого являются суммы цифр каждого числа старого массива.

3. Даны оценки, полученные на четырех экзаменах во время сессии студентами одной группы, по десятибалльной системе. Определите сколько студентов сдали сессию только на 7.

### **Вариант № 4**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Найдите сумму отрицательных и сумму положительных элементов и сравните их по модулю.

2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Создайте новый массив, элементами которого являются суммы делителей каждого числа старого массива.

3. Даны оценки, полученные на четырех экзаменах во время сессии студентами одной группы, по десятибалльной системе. Определите сколько студентов сдали сессию на балл не ниже 6.

### **Вариант № 5**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Найдите и выведите те элементы, которые больше предыдущего, стоящего с ним в одной строке.

2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Создайте новый массив, элементами которого являются суммы первой и последней цифры каждого числа старого массива.

3. Есть группа спортсменов из семи человек. Для каждого из них приводится рост и вес. Вес спортсмена считается нормальным, если от роста отнять 100 и полученное число отличается от веса не более чем на 3. Выведите номера тех спортсменов, чей вес превышает норму.

### **Вариант № 6**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Замените в массиве каждый четный элемент цифрой 2, а нечетный – цифрой 5.

2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Найдите сумму тех элементов, которые больше элемента, стоящего на главной диагонали в данной строке.

3. Приводятся показатели производства работы  $n \leq 30$  рабочих за семь дней. Определите номер рабочего, у которого показатель наибольший, и номер рабочего, у которого показатель второй по величине.

### **Вариант № 7**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Замените каждый элемент массива произведением его индексов.

2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Создайте новый массив, элементами которого является 0, если число делится на 5, и 7 – если число не делится на 5.

3. Приводится рост учеников трех параллельных классов, где их количество равно 25 в каждом классе. Определите номер самого высокого ученика в каждом классе.

### **Вариант № 8**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Найдите сумму тех элементов, у которых сумма их индексов – число четное.

2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Вычислите сумму элементов, которые расположены за минимальным элементом.

3. Приводится среднесуточная температура воздуха за несколько недель. Выведите номера тех недель, когда средняя температура всех дней была ниже 0.

### **Вариант № 9**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Найдите сумму элементов, которые стоят перед максимальным элементом

2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Найдите сумму элементов, которые расположены за максимальным элементом.

3. В области 10 районов. Для каждого из них известны площади, засеянные пшеницей, и урожай, собранный с этих площадей. Определите, в каком районе средняя урожайность пшеницы была выше.

### **Вариант № № 10**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Замените каждый элемент массива суммой его индексов.

2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Найдите суммы отрицательных элементов каждого столбца.

3. В университете  $n \leq 10$  факультетов. Известен план приема студентов на каждый факультет и число поданных заявлений. Определите конкурс на каждый факультет.

### **Вариант № 11**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Выведите только те элементы, которые меньше элемента, стоящего на главной диагонали в данной строке.



2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Найдите и выведите номера только тех строк, все элементы которых четные.

3. В области 10 районов. Для каждого из них известны площади, засеянные пшеницей, и урожай, собранный с этих площадей. Определите, в каком районе средняя урожайность пшеницы была самой низкой.

### **Вариант № 12**

1. Дана прямоугольная таблица, которая содержит не более 10 строк и не более 10 столбцов. Вычислите суммы элементов каждого столбца

2. Дан двумерный массив, который содержит не более 10 строк и не более 10 столбцов. Найдите наименьший элемент и его индексы в массиве.

3. В области  $n \leq 10$  районов. Для каждого района известны площади, засеянные пшеницей, и урожай, собранный с этих площадей. Определите среднюю урожайность пшеницы.

## КОМБИНИРОВАННЫЙ ТИП ДАННЫХ: ЗАПИСИ

*Цель работы:* познакомиться с новым типом данных (записи) и научиться работать с отдельными записями и массивами из записей.

### Теоретические сведения

#### *Понятие записи и поля записи*

Наиболее гибкий и удобный механизм построения структур данных самой разной длины заложен в комбинированном типе. Его значения предназначены, главным образом, для представления объектов при создании информационных систем. Например, в информационной системе об успеваемости студентов группы одна строка будет содержать сведения об одном студенте, например, его ФИО, номер группы, оценки по разным предметам, даты сдачи экзаменов и зачетов. Как видим, значения комбинированного типа состоят из нескольких компонент, которые имеют разные типы. Эти компоненты называются *полями*. Другими словами, поле – это переменная любого типа. Каждому полю записи дается свое имя и задается тип элементов.

#### *Объявление типа и переменных этого типа*

Формат объявления комбинированного типа следующий:

*1 способ:*

```
Туре Имя комбинированного типа = record
    Имя поля 1 : тип компонентов поля 1;
    .
    .
    .
    Имя поля N : тип компонентов поляN;
End;
Var Имя записи : Имя комбинированного типа;
```

*2 способ:*

```
Var Имя записи : record
    Имя поля 1 : тип компонентов поля 1;
    .
    .
    .
    Имя поля N : тип компонентов поляN;
End;
```

*Record* – служебное слово переводится запись, имя комбинированного типа, имя записи и имена полей дает пользователь, типом компонент каждого поля может быть любой тип данных

**Пример.** Рассмотрим запись, которая состоит из следующих компонент: фамилии студента, дня, месяца и года рождения, домашнего адреса и телефона. Такую запись можно объявить следующим образом:

```
Type
  Student = Record
    Fio, Adres, Telefon : String[40];
    Den, Mes, God : Integer;
  End;
Var a: Student;
```

Отсюда видно, что если в записи имеется несколько полей одинакового типа, то их можно перечислить через запятую и тип указать один раз. Формат объявления переменной типа запись обозначает только структуру этой переменной. Чтобы записи присвоить какое-то значение, необходимо присвоить значения всем полям этой записи. Обращение к элементу записи (то есть полю записи) происходит по составному имени, которое имеет следующий синтаксис (имя записи, точка, имя поля):

Имя записи. Имя поля

Такую комбинацию называют сложным именем. Поскольку обращение к каждому полю происходит по его названию, при объявлении записи, порядок полей не имеет никакого значения.

### ***Массивы записей***

Из записей можно составлять массивы. Когда составляются массивы из записей, сначала объявляется тип записи, а уже потом объявляется тип массив, у которого базовым типом является тип запись. Например,

```
Type student=Record
  Fio:String;
```

```

o1,o2,o3,o4:Integer;
    End;
    Mas=array [1..30] of student;
Var a:mas;

```

Здесь объявлен массив, который содержит не более 30 записей со следующими полями fio (фамилия студента), o1, o2, o3, o4 – это поля, которые содержат оценки каждого студента по четырем экзаменам во время сессии.

### ***Оператор над записями***

Обращение к полям записи имеет громоздкий вид. Это не всегда удобно, особенно, если имя записи длинное слово. Чтобы сократить такую запись, введен специальный оператор, который называется *оператор над записями* или *оператор присоединения*. Синтаксис такого оператора следующий

**With** Имя записи **do** **Begin** операторы **End**;

Один раз указав в этом операторе имя записи, далее с названиями полей этой записи можно работать как с простыми переменными.

### **Примеры выполнения заданий**

**Пример 1.** Напишите программу решения следующей задачи: даны две простые дроби  $a/b$  и  $c/d$ . Представьте их в виде записи и найдите сумму. Если можно, сократите полученную простую дробь.

**Решение.** Простую дробь можно представить в виде записи, у которой будет два поля: одно – это числитель простой дроби, второе поле – это знаменатель простой дроби. Оба поля будут иметь целый тип.

```

Program Primer_1;
Type dr = Record
    ch,zn: Integer;
    End;
Var x,y,s:dr; n:Integer;

```

{Процедура нахождения наибольшего общего делителя двух чисел. Такая процедура нам понадобится для сокращения полученной дроби}

```
Procedure NOD(a,b:Integer; Var d:Integer);
```

```
Begin
```

```
While a <> b do If a > b Then a:=a-b Else b:=b-a;
```

```
d:=a;
```

```
End;
```

```
Begin
```

```
{Ввод исходных данных}
```

```
Writeln('Ввести числитель и знаменатель первой дроби x');
```

```
Readln(x.ch, x.zn);
```

```
Writeln('Ввести числитель и знаменатель второй дроби y');
```

```
Readln(y.ch, y.zn);
```

```
{Нахождение суммы двух дробей  $s=x+y$  – это тоже дробь со своим числителем и знаменателем}
```

```
s.ch:=x.ch*y.zn+y.ch*x.zn;
```

```
s.zn:=x.zn*y.zn;
```

```
{Вывод полученной дроби без сокращения}
```

```
Writeln(x.ch,'/', x.zn, '+', y.ch, '/', y.zn, '= ', s.ch, '/', s.zn, '=');
```

```
Nod(s.ch,s.zn,n); {Обращение к процедуре}
```

```
{Делим отдельно числитель и знаменатель полученной дроби на наибольший общий делитель}
```

```
s.ch:=s.ch div n;
```

```
s.zn:=s.zn div n;
```

```
{Вывод сокращенной дроби с той же строке}
```

```
Write(s.ch, '/', s.zn);
```

```
Writeln;
```

```
End.
```

**Пример 2.** Напишите программу решения следующей задачи: дан список городов и количество жителей в каждом городе. Выведите названия тех городов, количество жителей которых более 100 тысяч человек.

**Решение.** Один город можно представить в виде записи с полями: название и количество жителей. Поскольку городов несколько, то они образуют массив.

Program Primer\_2;

Type {В разделе объявлений сначала необходимо объявить тип записи, а затем тип массива, так как он состоит из типа записей}

```
gor = Record nazw: String[20];
```

```
kol: Integer; End;
```

```
Mas=array[1..100] of gor; Var g:mas;
```

```
N,i:Integer; Begin {Ввод исходных данных} Writeln('Ввести количество городов');
```

```
Readln(n);
```

```
{Вводится значение каждого поля для каждой записи}
```

```
For i:=1 to n do
```

```
Begin
```

```
Writeln('Ввести название',I,'города');
```

```
Readln(g[i].nazw);
```

```
Writeln('Ввести количество жителей ', I, 'города');
```

```
Readln(g[i].kol);
```

```
End;
```

{Сначала выведем заголовок таблицы, в которой будут находиться названия городов и количество жителей в каждом городе, то есть вывод всех исходных данных}

```
Writeln('название города количество жителей');
```

```
{Вывод городов и количество их жителей}
```

```
For i:=1 to n do Writeln(g[i].nazw:20,g[i].kol:10);
```

{Вывод названия тех городов, у которых количество жителей более 100 тысяч}

```
Writeln('Ответ');
```

```
For i:=1 to n do
```

```
If g[i].kol>100 Then writeln(g[i].nazw);
```

```
End.
```

**Пример 3.** Имеется таблица.

Материал	Плотность, кг/м <sup>3</sup>	Температура плавления
Алюминий	2700	657
Вольфрам	19300	390
Медь	8900	1083
Свинец	11300	327
Олово	7300	232
Цинк	7100	419

Напишите программу нахождения и вывода названия материалов с наибольшей плотностью и наименьшей температурой плавления.

**Решение.** Каждую строку этой таблицы представим в виде записи, у которой три поля: название, плотность и температура, а из записей составим массив.

```
Program Primer_3;
Type {Объявление типа запись}
  mat = Record      nazw: String[20];
    pl, temp: Integer;      End;
{Объявление типа массив из записей}
  Mas=array[1..100] of mat;
{Объявление простых переменных} Var m:mas;
  n,i, max, min, nom1, nom2:Integer;
Begin {Ввод исходных данных}      Writeln('Ввести количество
материалов');
  Readln(n);
  For i:=1 to n do
    With m[i] do
      Begin
        Writeln('Ввести название',I,'материала');
        Readln(nazw);
        Writeln('Ввести плотность ',I,'материала');
        Readln(pl);
        Writeln('Ввести температуру плавления',I,'материала');
        Readln(temp);
      End;
    {Вывод на экран заголовка таблицы, в которой будет помещен
исходный массив}
    Writeln('Название материала плотность температура плавления');
    {Вывод на экран введенного массива}
    For i:=1 to n do
      With m[i] do Begin
        Writeln(nazw:20,pl:10,temp:10);
      End;
    {Нахождение материала с наибольшей плотностью }
    Max:=m[1].pl; Nom1:=1;
```

```

For i:=2 to n do
  With m[i] do {Использования оператора над записями}
    Begin
      If pl>max Then Begin max:=pl; nom1:=I;End;
      End;
  Writeln('Материал с наибольшей плотностью ',m[nom1].nazw);
  {Нахождение материала с наименьшей температурой плавления}
  Min:=m[1].temp; Nom2:=1;
  For i:=2 to n do
    With m[i] do
      Begin
        If temp<min
          Then Begin min:=temp; nom2:=I;End;
          End;
    Writeln('Материал с наименьшей температурой плавления ',
m[nom2].nazw);
  End.

```

**Пример 4.** Напишите программу решения следующей задачи: даны сведения об учениках одного класса по следующей схеме: фамилия, рост, вес, год рождения, домашний адрес. Выведите средний вес и рост учеников класса.

```

Program Primer_4;
Type uch=Record
  fio, adres:String;
  rost, ves: real;
  god : Integer;
  End;
  mas = Array [1..35] of uch;
Var uchenik:mas;
  i, n: Integer;
  v, r, sr, sv : real;
Begin
  Writeln('Ввести количество учеников в классе');
  Readln(n);
  {Введем исходные данные для каждого ученика}
  For i := 1 to n do

```



```

With uchenik[i] do
  Begin
    Writeln('Ввести фамилию ученика');
    Readln(fio);
    Writeln('Ввести домашний адрес ученика');
    Readln(adres);
    Writeln('Ввести год рождения ученика');
    Readln(god);
    Writeln('Ввести рост ученика');
    Readln(rost);
    Writeln('Ввести вес ученика');
    Readln(ves);
  End;
  {Выведем на экран все сведения об учениках}
  Writeln('Ученики класса');
  For i :=1 to n do
    With uchenik[i] do
      Begin
        Writeln(fio:15,adres:20,god:6,rost:6:2,ves:6:2);
      End;
    {Выведем фамилии самого высокого исамого маленького
ученика}
    v:=0;
    R:=0;
    For i:=1 to n do
      With uchenik[i] do
        Begin
          v:=v+ves;
          r:=r+rost;
        End;
      sv:=v/n;
      sr:=r/n;
    Writeln(sv, ' - средний вес учеников класса');
    Writeln(sr, ' - средний рост учеников класса');
  End.

```

## Контрольные вопросы

1. Что такое запись?
2. Как называются элементы записи?
3. Какого типа могут быть поля записи?
4. Могут ли различные поля записи иметь различные типы?
5. Могут ли различные поля записи иметь одинаковый тип?
6. Как объявить запись?
7. Какие служебные слова используются при объявлении записи?
8. Как происходит обращение к полю записи?
9. Что такое составное имя?
10. Что обозначает первая часть составного имени?
11. Что обозначает вторая часть составного имени?
12. Могут ли быть одинаковые имена полей в различных записях?
13. Каково назначение оператора With?
14. Вставьте необходимые обозначения вместо многоточий:  
With ... do ...;
15. Как объявляются массивы записей?

## Задания для самостоятельного выполнения

1. В каждой задаче варианта напишите программу решения задач.
2. Оформите отчет по лабораторной работе, который должен состоять из следующих разделов: номера лабораторной работы и ее названия, условия каждой задачи и ее программы.

### Вариант № 1

1. Даны оценки группы студентов на экзаменах по четырем предметам: алгебре, геометрии, философии, информатике. Найдите, по какому предмету студенты получили наибольшее количество двоек.
2. Даны три простые дроби, опишите их как записи. Найдите сумму этих дробей и сократите.

### Вариант № 2

1. Даны оценки группы студентов на экзаменах по четырем предметам: алгебре, геометрии, философии, информатике. Найдите и выведите фамилии тех студентов, которые все экзамены сдали на 9.

2. Даны три простые дроби, опишите их как записи. Найдите разность этих дробей и сократите.

### **Вариант № 3**

1. Даны оценки группы студентов на экзаменах по четырем предметам: алгебре, геометрии, философии, информатике. Найдите и выведите фамилии тех студентов, которые на экзаменах получили не ниже 7 баллов.

2. Даны три простые дроби, опишите их как записи. Найдите их произведение и сократите.

### **Вариант № 4**

1. Даны оценки группы студентов на экзаменах по четырем предметам: алгебре, геометрии, философии, информатике. Найдите, по какому предмету студенты получили наибольшее количество 10 и 9.

2. Даны три простые дроби, опишите их как записи. Найдите их частное от деления первой дроби на вторую и частное от деления полученного результата на третью дробь и сократите.

### **Вариант № 5**

1. Даны оценки группы студентов на экзаменах по четырем предметам: алгебре, геометрии, философии, информатике. Найдите, по какому предмету студенты получили наибольшее количество пятерок.

2. Даны координаты двух точек в пространстве, опишите их как записи. Найдите расстояние между этими точками.

### **Вариант № 6**

1. Даны сведения по областям: название области, количество засеянной земли рожью, собранный урожай. Выведите название области, у которой наибольшая урожайность.

2. Даны координаты двух точек на плоскости, опишите их как записи, которые являются концами диагонали прямоугольника со сторонами параллельными осям координат. Найдите площадь этого прямоугольника.

### Вариант № 7

1. Даны фамилии и псевдонимы писателей, их год рождения и год смерти в следующей таблице.

Найдите и выведите фамилии и псевдонимы того писателя, который прожил больше остальных

Фамилия	Псевдоним	Год рождения	Год смерти
Луцевич	Янка Купала	1882	1942
Федоров	Янка Мавр	1883	1971
Левицкая	Адам Шкиляр	1895	1968
Лыньков	Василек (Михась)	1899	1975
Макаенок	Малоказдатчик	1920	1982

2. Даны координаты трех точек в пространстве, опишите их как записи. Определите, какая из точек лежит ближе к началу координат.

### Вариант № 8.

1. В магазине продан товар разных наименований и разной стоимости в таблице.

Подсчитайте общую сумму от продаж удочек.

Тип товара	Наименование	Стоимость одной единицы	Количество проданных единиц
Удочки	Бамбуковые	10,7	100
Палатки	Складные	3400	6
Удочки	Зимние	8	30
Мячики	Футбольные	97,2	32
Удочки	Пластиковые	18,1	62
Мячики	Детские	150	47

2. Даны три точки на плоскости, опишите их как записи. Определите, лежат ли эти точки на одной прямой.

**Вариант № 9.**

1. Создайте массив записей, представленных в таблице.  
Найдите и выведите общую площадь перечисленных стран.

Страна	Территория, км <sup>2</sup>	Население, млн. человек	Столица
Латвия	67,7	2,6	Рига
Литва	65,2	3,4	Вильнюс
Польша	313	34,8	Варшава
Украина	603,7	50	Киев
Россия	17075,4	139	Москва

2. Даны три точки на плоскости, опишите их как записи. Определите, могут ли они быть вершинами некоторого треугольника.

**Вариант № 10.**

1. Создайте массив записей, представленных в таблице.  
Найдите и выведите общее количество населения перечисленных стран.

Страна	Территория в км <sup>2</sup>	Население млн. человек	Столица
Латвия	67,7	2,6	Рига
Литва	65,2	3,4	Вильнюс
Польша	313	34,8	Варшава
Украина	603,7	50	Киев
Россия	17075,4	139	Москва

2. Даны три простые дроби, опишите их как записи. Определите, равны ли эти дроби.

### Вариант № 11

1. Создайте массив записей, представленных в таблице.  
Найдите и выведите для каждой страны плотность населения.

Страна	Территория в км <sup>2</sup>	Население млн. человек	Столица
Латвия	67,7	2,6	Рига
Литва	65,2	3,4	Вильнюс
Польша	313	34,8	Варшава
Украина	603,7	50	Киев
Россия	17075,4	139	Москва

2. Даны координаты трех вершин треугольника, опишите их как записи. Найдите площадь этого треугольника.

### Вариант № 12

1. Создайте массив записей, представленных в таблице.  
Найдите и выведите страну с наибольшей плотностью населения.

Страна	Территория в км <sup>2</sup>	Население млн. человек	Столица
Латвия	67,7	2,6	Рига
Литва	65,2	3,4	Вильнюс
Польша	313	34,8	Варшава
Украина	603,7	50	Киев
Россия	17075,4	139	Москва

2. Даны координаты трех вершин треугольника, опишите их как записи. Найдите длины сторон этого треугольника.

## ТИП ДАННЫХ: ФАЙЛЫ ДАННЫХ

*Цель работы:* ознакомиться с типом данных – текстовыми и типизированными файлами, научиться создавать файлы и работать с файлами данных.

### Теоретические сведения

#### *Понятие физического и логического файла*

До сих пор речь шла о физических файлах. *Физический файл* – это информация, сохраненная на твердом носителе под уникальным именем.

Но в языке ПаскальABC имеется специальная структура данных, называемая файлами данных. В отличие от физического файла, такие структуры данных называются логическими файлами. Таким образом, *логический файл* – это структура данных в языке программирования ПаскальABC.

Компьютер используется для обработки информации, и практика программирования привела к очень простой идее: не данные должны привязываться к программе, а наоборот – одни и те же данные могут быть использованы в разных программах. Поэтому эти данные должны сохраняться в готовом виде где-то так, чтобы к ним можно было обратиться в любой момент. Для этого в языках программирования и было введено понятие логического файла. Другими словами логические файлы – это файлы данных.

#### *Виды файлов данных*

Файлы в ПаскальABC классифицируются по двум признакам:

- по типу (логической структуре);
- по методу доступа к элементам файла.

Классификацию файлов можно представить в виде рис. 11.1.

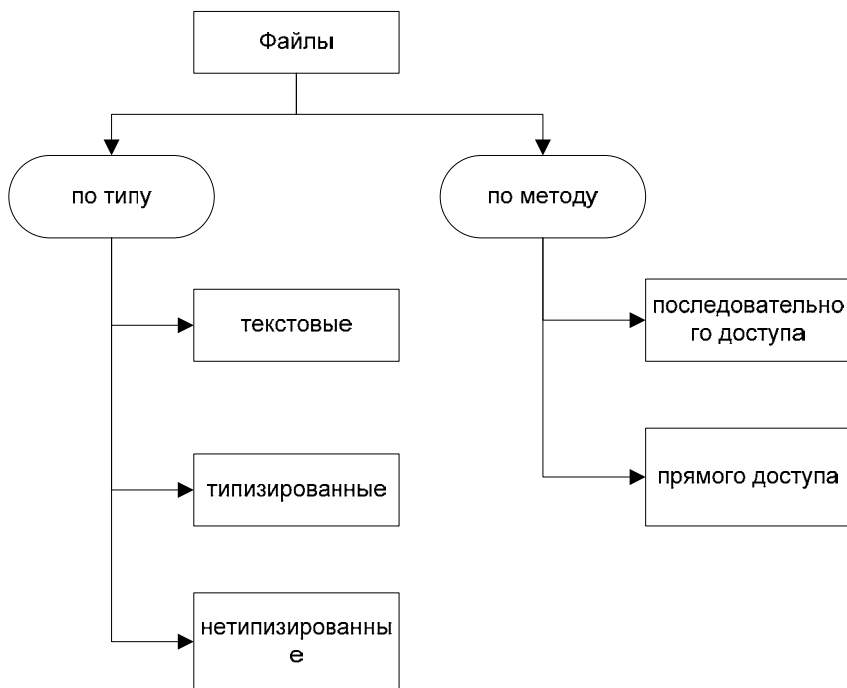


Рис. 11.1. Классификация файлов

Итак, логический файл – это файл данных. Как располагаются данные в логическом файле? Условно его можно представить как бесконечную ленту, разделенную на отдельные ячейки, в каждой из которых можно хранить только один компонент. Имеется специальный указатель, с помощью которого идет доступ к этим ячейкам.

В файлах последовательного доступа указатель всегда устанавливается в начало файла, поэтому поиск можно начинать только с начала файла и проверять по очереди каждый компонент данного файла, пока не будут найдены нужные компоненты.

В файлах прямого доступа указатель можно устанавливать в нужное место и продолжать поиск выборочным методом.

Вообще, логическая структура файла очень похожа на структуру массива. Различия между массивами и файлом данных заключается в следующем:



1. У массива количество элементов фиксируется в момент его объявления, то есть в момент распределения памяти. Нумерация элементов массива выполняется соответственно нижней и верхней границам, указанным при его объявлении. У файла количество элементов в процессе работы программы может изменяться. Нумерация элементов файла выполняется слева направо, начиная от нуля (кроме текстовых файлов). Количество элементов файла в каждый момент времени *не известно*. Зато известно, что в конце файла располагается специальный символ конца файла **EOF**, в качестве которого используется управляющий символ с кодом 26 (это клавиши Ctrl + Z).

2. Элементы массива целиком располагаются в *оперативной памяти* компьютера, поэтому при каждом новом запуске отдельной программы массив создается заново. Элементы файла располагаются на *внешних* носителях информации и там сохраняются. Поэтому в любой программе можно воспользоваться данными, записанными в файл данных.

3. Работу с файлами проводят с помощью стандартных процедур и функций, предназначенных *только для работы с файлами*.

## Текстовые файлы

### *Понятие текстового файла данных*

Файл данных, который состоит из символьных строк, является текстовым. Строки в нем могут быть разной длины. Поэтому в текстовом файле в конце каждой строки помещается специальный символ – конец строки. Для обозначения конца строки имеется специальная функция

**EOLN** (имя файловой переменной).

Это, как правило, пара символов CR (перевод строки – код 13) и LF (возврат каретки – код 10).

Весь текстовый файл, как и любой другой, заканчивается маркером Ctrl + Z, который определяет функция

**EOF**(имя файловой переменной).

Поскольку строки могут быть разной длины, текстовые файлы могут обрабатываться *только последовательно*.

### ***Объявление текстового файла***

Текстовые файлы можно объявить следующим образом:

**Var** имя файловой переменной : Text;

где Text – это стандартный идентификатор, такой же, как Integer и другие.

### ***Создание текстового файла данных***

Чтобы создать текстовый файл данных необходимо выполнить следующие действия:

1. Описать файловый тип с помощью следующих строк:

**Var** имя файловой переменной : Text;

2. Установить связь между файловой переменной в программе и файлом данных на диске. Это делает процедура

**Assign** (имя файловой переменной, ‘имя диска: путь к имени файла данных на диске’);

3. Открыть файл для записи с помощью процедуры

**Rewrite**(имя файловой переменной);

Эта процедура стирает то, что было записано ранее в данный файл и устанавливает указатель в начало файла, перед его первым элементом. Если набора данных не было, то будет создан новый набор данных.

4. Записать необходимую информацию в файл с помощью оператора

**WriteLn**(имя файловой переменной, список записываемых переменных);

5. Закрыть файл с помощью процедуры

**Close**(имя файловой переменной).

### *Дополнение текстового файла новыми данными*

Чтобы в конец текстового файла добавить данные, необходимо выполнить следующие пункты:

1. Описать файловый тип

**Var** имя файловой переменной : Text;

2. Установить связь между файловой переменной в программе и файлом данных на диске.

**Assign** (имя файловой переменной, 'имя диска: путь к имени файла данных на диске')

3. Открыть уже созданный текстовый файл для записи с помощью процедуры

**Append**(имя файловой переменной)

Эта процедура устанавливает указатель в конец файла, после последней строки файла, и позволяет заносить новые данные.

4. Записать необходимую информацию в файл с помощью оператора

**WriteLn**(имя файловой переменной, список записываемых переменных)

5. Закрыть файл с помощью процедуры

**Close**(имя файловой переменной);

### *Использование данных из файла*

Чтобы прочитать информацию из текстового файла необходимо выполнить следующие действия:

1. Описать файловый тип

**Var** имя файловой переменной : Text;

2. Установить связь между файловой переменной в программе и файлом данных на диске.

**Assign** (имя файловой переменной, 'имя диска: путь к имени файла данных на диске');

3. Открыть уже созданный текстовый файл для чтения с помощью процедуры

**Reset**(имя файловой переменной)

Эта процедура устанавливает указатель на первый элемент ранее созданного файла данных и ничего не стирает.

4. Записать необходимую информацию в файл с помощью оператора

**WriteLn**(имя файловой переменной, список записываемых переменных)

5. Закрыть файл с помощью процедуры

**Close**(имя файловой переменной).

### Типизированные файлы

Все элементы типизированного файла должны быть одного типа, следовательно, и одного размера. Элементами таких файлов могут быть числа, массивы, записи, но только не файлы. В типизированных файлах вследствие одинакового размера элементов возможен *произвольный* доступ к ним, то есть доступ как последовательный, начиная с первого и далее, так и прямой, то есть по номеру элемента. Элементы типизированного файла нумеруются целыми числами, начиная с нуля.

## *Объявление типизированного файла*

В общем виде типизированный файл данных можно описать следующим образом:

**Type** имя файлового типа = **File of** тип элементов файла;

**Var** имя файловой переменной : имя файлового типа;

Описание очень похоже на описание массивов, только не указывается количество элементов.

Программирование на уровне файлов сводится в основном к следующим алгоритмам: создание файла; использование данных файла; дополнение файла новыми данными.

## *Создание типизированного файла данных*

Чтобы создать файл данных необходимо выполнить следующие действия:

1. Описать файловый тип с помощью следующих строк:

**Type** имя файлового типа = **File of** тип элементов файла;

**Var** имя файловой переменной : имя файлового типа;

2. Установить связь между файловой переменной в программе и файлом данных на диске.

**Assign** (имя файловой переменной, 'имя диска: путь к имени файла данных на диске');

При записи процедуры **Assign** следует быть внимательным и следить, чтобы был правильно написан путь к нужному файлу.

3. Открыть файл для записи с помощью процедуры

**Rewrite**(имя файловой переменной);

Эта процедура стирает то, что было записано ранее в данный файл и устанавливает указатель в начало файла, перед его первым элементом. Если набора данных не было, то будет создан новый набор данных.

4. Записать необходимую информацию в файл с помощью оператора

**Write**(имя файловой переменной, список записываемых переменных);

5. Закрыть файл с помощью процедуры

**Close**(имя файловой переменной);

Если файл не закрыть, данные файла будут потеряны.

### *Дополнение типизированного файла новыми данными*

Для этого необходимо выполнить следующие действия:

1. Описать файловый тип.

**Type** имя файлового типа = **File of** тип элементов файла;

**Var** имя файловой переменной : имя файлового типа;

2. Установить связь между файловой переменной и файлом данных.

**Assign** (имя файловой переменной, 'имя диска: путь к имени файла данных на диске');

Открыть уже созданный файл для записи процедурой

**Reset**(имя файловой переменной);

Эта процедура устанавливает указатель на первый элемент ранее созданного файла данных и ничего не стирает.

3. Установить указатель файла за нужным элементом файла процедурой

**Seek**(имя файловой переменной, номер элемента);

Процедура **Seek** может иметь следующие формы:

– а) **Seek**(имя файловой переменной, **FileSize**(имя файловой переменной)); – установка указателя непосредственно за последним элементом файла;

– б) **Seek**(имя файловой переменной, **FilePose**(имя файловой переменной)+1); – пропуск одного текущего элемента файла;

– с) **Seek** (имя файловой переменной, 0); – установка указателя на начало файла.

4. Записать новые данные в файл процедурой:

**Write**(имя файловой переменной, список записываемых переменных);

5. Закрывать файл процедурой:

**Close**(имя файловой переменной);

### *Использование данных из типизированного файла*

Чтобы прочитать информацию из файла данных, необходимо выполнить следующие пункты:

1. Описать файловую переменную

**Type** имя файлового типа = **File of** тип элементов файла;

**Var** имя файловой переменной : имя файлового типа;

2. Установить связь между файловой переменной и файлом данных

**Assign** (имя файловой переменной, ‘ имя диска: путь к имени файла данных на диске ‘);

3. Открыть файл для чтения

**Reset**(имя файловой переменной);

4. Прочитать файл с помощью процедуры

**Read**(имя файловой переменной, имя читаемой переменной);

По этой процедуре читается очередной элемент файла и его значение присваивается имени читаемой переменной. После этого указатель перемещается на следующую позицию файла. Если в процессе исполнения указатель будет установлен на позицию, в которой уже нет элементов файла, то возникает операция «конец файла» и чтение файла завершается.

5. Закрывать файл.

**Close**(имя файловой переменной).

### Примеры выполнения заданий

**Пример 1.** Создайте файл под названием *masc.dat*, в котором будет находиться массив целых чисел. Пусть файл *masc.dat* будет находиться на диске *c* в папке *Other*.

При составлении программы воспользуйтесь алгоритмом для создания типизированного файла.

```
Program Primer_1;
Type celchisla=File of Integer;
Var f:celchisla;
{Имя файлового типа celchisla, имя файловой переменной f.
Пункт 1}
  I,n,m:Integer; {Простые переменные}
  Begin
    Assign(f,'c:\Other\masc.dat'); { Пункт 2}
    Rewrite(f); { Пункт 3}
    Write(' Введите количество элементов файла');
    Read(n);
    Write(f,n); { Пункт 4. Запишем в файл f количество элемен-
тов массива n. Этот элемент в файле будет записан на 0 место.}
    For i:=1 to n do
      Begin
        {Вводим поочередно числа массива}
        Write('Введите ',i,' -ое число ');
        Read(m);
        {Записываем каждое число в файл f}
        Write(f,m);          {Пункт 4}
```



```
End;  
Close(f); { Пункт 5}  
End.
```

Запишите эту программу в каталог *Other* под именем *Stud1.pas*. После выполнения этой программы в каталоге *Other* будет два файла: один хранит текст программы с именем *Stud1.pas*, и второй, массив введенных чисел с именем *masc.dat*.

**Пример 2.** Ранее был создан файл данных в каталоге *Other* с именем *masc.dat*. Найдите сумму элементов этого файла.

```
Program Primer_2;  
Type celchisla=file of Integer;  
Var F:celchisla;  
I,n,m,s:Integer;  
Begin  
Assign(f, 'c:\Other\masc.dat');  
Reset(f); { Открываем файл для чтения}  
Read(f,n); { Читаем первый элемент файла – это количество  
чисел в массиве и записываем это количество в переменную n}  
S:=0;  
For i:=1 to n do  
Begin  
Read(f,m); { Читаем очередной элемент файла и записываем  
его значение в переменную m}  
S:=s+m;  
End;  
Close(f);  
Writeln(' Сумма элементов файла =',s);  
End.
```

Файл, из которого берутся данные, называется входным файлом, а в который они записываются – выходным файлом.

Необходимо отметить, что нельзя одновременно записать информацию в файл данных и прочитать ее. Сначала полностью создайте файл данных и только после этого данными можно пользоваться.

**Пример 3.** Постройте типизированный файл данных, в котором представлены сведения об оценках, полученных студентами группы на сессии. Это такой файл, который состоит из записей со следующими полями: фамилия студента, математика, информатика, философия. Выведите все фамилии студентов, которые имеют задолженности, и их оценки.

```
Program Primer_3_1;  
{Сначала объявляем тип запись, только после этого объявим тип  
массив из записей}  
Type sesia=Record  
    fio:String[15];  
    matem, infor, filos:Integer;  
end;  
mas=File of sesia;  
Var f:mas; i,n,m:Integer; z:sesia;  
Begin  
{создадим файл данных}  
Assign(f,'c:\masc2.dat');  
Rewrite(f);  
Writeln('Ввести количество студентов'); Readln(n);  
For i:=1 to n do  
    Begin  
        Writeln('Ввести фамилию ',I,' студента');  
        Readln(z.fio);  
        Writeln('Ввести оценку по математике ',I,' студента');  
        Readln(z.matem);  
        Writeln('Ввести оценку по информатике ',I,' студента');  
        Readln(z.infor);  
        Writeln('Ввести оценку по философии ',I,' студента');  
        Readln(z.filos);  
        Write(f,z); {Записали всю запись в файл f}  
    End;  
Close(f);  
End.
```

{Записанные данные будут храниться в файле и их можно использовать сколько угодно раз. Для этого необходимо составить

новую программу. Используем данные из файла и выведем то, что требуется по условию задачи}

```
Program Primer_3_2;  
Type sesia=Record  
    fio:String[15];  
    matem, infor, filos:Integer;  
end;
```

```
mas=File of sesia;
```

```
Var f:mas; i,n,m:Integer; z:sesia;
```

```
Begin
```

```
Assign(f,'c:\masc2.dat');
```

```
Reset(f);
```

{Пока не конец файла, будем читать из файла запись, проверять, сдал ли студент все экзамены, и, если нужно, выводить сообщения о нем на экран}

```
While not(eof(f)) do {пока не конец файла f}
```

```
Begin
```

```
Read(f,z); {читаем запись из файла и отправляем ее в переменную z}
```

```
If (z.matem<4) or (z.infor<4) or (z.filos<4)
```

```
Then Writeln(z.fio:15, z.matem:5, z.infor:5, z.filos:5);
```

```
End;
```

```
Close(f);
```

```
End.
```

## Контрольные вопросы

1. Что такое логический файл?
2. Какого типа бывают файлы данных?
3. Чем отличается файл данных последовательного доступа от файла данных прямого доступа?
4. Что такое текстовый файл данных?
5. Как объявить в программе переменную текстового файла данных?
6. К файлам какого доступа относятся текстовые файлы?
7. Какая особенность у текстовых файлов?
8. Как объявить тип текстовой файловой переменной в программе?
9. Как объявить текстовую файловую переменную в программе?
10. Что делает процедура Assign?

11. Что делает процедура Rewrite?
12. Что делает процедура Reset?
13. Что делает процедура Writeln?
14. Что делает процедура Readln?
15. Что делает процедура Close?
16. Что обозначает процедура Append?
17. Что обозначает функция Eof?
18. Какая особенность у типизированных файлов?
19. Как объявить тип файловой переменной в программе?
20. Как объявить файловую переменную в программе?
21. Что делает процедура Write?
22. Что делает процедура Read?
23. Что обозначает процедура Seek?

### **Задания для самостоятельного выполнения**

1. В каждой задаче варианта напишите программу решения задач.
2. Оформите отчет по лабораторной работе, который должен состоять из следующих разделов: номера лабораторной работы и ее названия, условия каждой задачи и ее программы.

#### **Вариант № 1**

1. Постройте файл данных, компонентами которого являются вещественные числа. Найдите сумму квадратов компонентов файла.
2. Постройте типизированный файл данных, который состоит из записей с такими полями: фамилия владельца автомобиля, марка автомобиля, номер автомобиля. Дана фамилия владельца автомобиля, необходимо в файле данных найти марку и номер его автомобиля. Подсчитайте, сколько всего автомобилей такой же марки в данном файле.

#### **Вариант № 2**

1. Постройте файл данных, компонентами которого являются целые числа. Выведите на экран все четные числа данного файла.
2. Постройте типизированный файл данных, который состоит из записей с такими полями: фамилия автора книги, название книги, год издания. Выведите все названия книг, которые изданы после 2000 г. и ответьте на вопрос, есть ли среди этих книг с названием «Информатика».

### **Вариант № 3**

1. Постройте файл данных, компонентами которого являются вещественные числа. Выведите на экран все числа этого файла, которые больше данного числа  $X$  и меньше данного числа  $Y$ .

2. Постройте типизированный файл данных, который состоит из записей с такими полями: название игрушки, ее цена, возраст, для которого она предназначена. Найдите игрушки, цена которых больше или равна 40 000 руб. Выведите название тех игрушек, которые подходят для возраста от 3 до 5 лет.

### **Вариант № 4**

1. Постройте файл данных, компонентами которого являются вещественные числа. Найдите разность между первой и последней компонентой файла.

2. Постройте типизированный файл данных, который состоит из записей с такими полями: фамилия служащего, его имя, номер его телефона. Даны две фамилии служащих, выведите номера их телефонов.

### **Вариант № 5**

1. Постройте файл данных, компонентами которого являются вещественные числа. Поменяйте местами первую и последнюю компоненту данного файла.

2. Постройте типизированный файл данных, который состоит из записей с такими полями: размер кубика, его цвет, материал, из которого он сделан. Найдите количество металлических и деревянных кубиков.

### **Вариант № 6**

1. Постройте файл данных, компонентами которого являются целые числа. Замените в этом файле все четные числа на 0.

2. Постройте типизированный файл данных, который состоит из записей с такими полями: фамилия ученика, его имя, номер класса, буква класса. Выведите на экран фамилии учеников с одинаковыми именами.

### **Вариант № 7**

1. Постройте файл данных, компонентами которого являются целые числа. Найдите сумму нечетных чисел этого файла.

2. Постройте типизированный файл данных, который состоит из записей с такими полями: фамилия пассажира, количество его багажа, вес багажа. Найдите количество пассажиров, которые имеют багаж из двух и более вещей. Определите, есть ли хотя бы два пассажира, у которых одинаковый вес багажа.

### **Вариант № 8**

1. Постройте файл данных, компонентами которого символы латинского алфавита (не в алфавитном порядке и не обязательно все). Поменяйте местами первую и пятую компоненты.

2. Постройте типизированный файл данных, который состоит из записей с такими полями: число месяца, название месяца, год. Найдите год с наибольшим номером и выведите все его зимние даты.

### **Вариант № 9**

1. Постройте файл данных, компонентами которого являются символы латинского алфавита (не в алфавитном порядке и не обязательно все). Поменяйте местами последнюю и пятую компоненты местами.

2. Постройте типизированный файл данных, который состоит из записей с такими полями: фамилия владельца автомобиля, марка автомобиля, его максимальная скорость. Найдите марку автомобиля с наибольшей скоростью и выведите фамилию его владельца.

### **Вариант № 10**

1. Постройте файл данных, компонентами которого являются натуральные числа. Определит количество компонент, которые делятся на 3 без остатка.

2. Постройте типизированный файл данных, который состоит из записей с такими полями: фамилия автора, название книги, область, к которой относится данная книга. Выведите все книги, которые относятся к области «Медицина».

### **Вариант № 11**

1. Постройте файл данных, компонентами которого являются натуральные числа. Подсчитайте количество значений, которые делятся на 5 и не делятся на 10.

2. Постройте типизированный файл данных, который состоит из записей с такими полями: название мероприятия, место проведения, день проведения, месяц проведения. Выведите мероприятия, которые будут проходить в Минске в марте.

### **Вариант № 12**

1. Постройте файл данных, компонентами которого являются натуральные числа. Подсчитайте сумму тех компонент, которые делятся на 5 и не делятся на 10.

2. Постройте типизированный файл данных, который состоит из записей с такими полями: название продукта, количество углеводов, количество белков, количество жиров. Выведите названия тех продуктов, которые соответствуют следующим условиям: белки меньше 20, жиры больше 10, углеводы больше 30.

## БИБЛИОРГАФИЧЕСКИЙ СПИСОК

1. Павловский, А. И. Теоретические основы алгоритмизации / А. И. Павловский, В. В. Пенкрат. – Минск: БГПУ, 2007. – 58 с.
2. Пенкрат, В. В. Методы алгоритмизации / В. В. Пенкрат. – Минск: БГПУ, 2012. – 107 с.
3. Бурдо, Е. Н. Основы программирования / Е. Н. Бурдо, Н. И. Витушко, А. Ю. Зуенок. – Минск: БНТУ, 2007. – 173 с.
4. Нарэйка, Н. М. Праграмаванне на мове Pascal : лабараторны практыкум : ч. 1 / Н. М. Нарэйка, А. Я. Пупцаў, В. У. Пянкрат. – Минск: БДПУ, 1999. – 153 с.
5. Нарэйка, Н. М. Праграмаванне на мове Pascal : лабараторны практыкум : ч. 2 / Н. М. Нарэйка, А. Я. Пупцаў, В. У. Пянкрат. – Минск: БДПУ, 1999. – 69 с.
6. Марченко, А. И. Программирование в среде Borland Pascal 7.0 / А. И. Марченко, Л. А. Марченко. – Киев: ЮНИОР, 1998. – 478 с.
7. Турбо Паскаль 7.0 – Киев: Торгово-издательское бюро ВНУ, 1997. – 444 с.
8. Павловский, А. И. Компьютер и занимательные задачи / А. И. Павловский, С. А. Апанасевич. – Минск: Белорусская ассоциация «Конкурс», 2014. – 126 с.



Учебное издание

**АСТАПЧИК** Наталья Ивановна  
**ПЕНКРАТ** Валентина Владимировна

## **ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ**

Учебно-методическое пособие для студентов  
специальности 1-08 01 01 «Профессиональное обучение  
(по направлениям)», направления специальности 1-08 01 01-01  
«Профессиональное обучение (машиностроение)», 1-08 01 01-05  
«Профессиональное обучение (строительство)», 1-08 01 01-07  
«Профессиональное обучение (информатика)»

Редактор *Е. С. Кочерго*  
Компьютерная верстка *Е. А. Беспанской*

Подписано в печать 08.06.2018. Формат 60×84 <sup>1</sup>/<sub>16</sub>. Бумага офсетная. Ризография.  
Усл. печ. л. 10.70. Уч.-изд. л. 8,36. Тираж 100. Заказ 1133.

Издатель и полиграфическое исполнение: Белорусский национальный технический университет.  
Свидетельство о государственной регистрации издателя, изготовителя, распространителя  
печатных изданий № 1/173 от 12.02.2014. Пр. Независимости, 65. 220013, г. Минск.