

ПРАКТИЧЕСКАЯ РАБОТА №1

ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЛИНЕЙНОЙ СТРУКТУРЫ

Цель и задача работы: научиться использовать внутренние функции, освоить понятие “арифметический оператор”, написать программу, вводящую данные с клавиатуры, обрабатывающую их с помощью внутренних функций и выводящую результат на экран дисплея.

1. Программирование на Турбо-Паскале

Язык программирования Паскаль был разработан Николасом Виртом в начале 70-х годов для обучения программированию. Поэтому он наиболее удобен как первый изучаемый язык программирования.

Семь основных элементов программирования

Большинство программ создаются для решения какой-либо задачи. Решение задачи достигается благодаря обработке информации или данных. Поэтому программист должен знать:

- как ввести информацию в программу (ввод);
- как хранить информацию в программе (данные);
- как указать правильные команды для обработки данных (операции);
- как передать обратно данные из программы пользователю (вывод).

Можно упорядочить команды таким образом, чтобы:

- некоторые из них выполнялись только, если выполняется некоторое условие или ряд условий (условное выполнение);
- другие выполнялись повторно некоторое число раз (циклы);
- третьи выделялись в отдельные части, которые могут быть выполнены в разных местах программы (подпрограммы).

Таким образом, перечислены все семь основных элементов программирования: ввод, данные, операции, вывод, условное выполнение, циклы и подпрограммы. Этот список не является исчерпывающим, однако, он содержит те элементы, которые обычно присущи всем программам (и языкам программирования).

Ниже дается краткое описание каждого элемента.

Ввод – считывание значений, поступающих с клавиатуры, с диска или из порта ввода/вывода.

Данные – это константы, переменные и структуры, содержащие числа (целые и вещественные), текст (символы и строки) или адреса (переменных и структур).

Операции осуществляют присваивание значений, их комбинирование (сложение, деление и так далее) и сравнение значений (равные, не равные и так далее).

Вывод – запись информации на экран, на диск или в порт ввода/вывода.

Условное выполнение предполагает выполнение набора команд в случае, если выполняется (является истинным) некоторое условие (если это условие не выполняется, то эти команды пропускаются или же выполняется другой набор команд) или если некоторый элемент данных имеет некоторое специальное значение или значение из некоторого спектра.

Благодаря *циклам* некоторый набор команд выполняется повторно или фиксированное число раз, или пока является истинным некоторое условие, или пока некоторое условие не стало истинным.

Подпрограммы представляют собой наборы инструкций, которые имеют самостоятельное имя и которые могут быть выполнены из любого места программы при обращении к ним по имени.

2. Стандарт Турбо Паскаля

Основные символы языка

Турбо Паскаль использует следующие подмножества набора символов кода ASCII:

- Буквы английского алфавита от A до Z и от a до z.
- Цифры – арабские цифры от 0 до 9.
- Шестнадцатичные цифры – арабские цифры от 0 до 9, буквы от A до F и буквы от a до f.
- Пробелы – символ пробела (ASCII 32) и все управляющие символы кода ASCII (ASCII 0-31), включая символ конца строки или символ возврата каретки (ASCII 13).

Специальные символы и зарезервированные слова представляют собой символы, имеющие одно или несколько фиксированных значений. Специальными символами являются следующие одиночные символы:

+ - * / = < > [] . , () : ; ^ @ { } \$ #

Следующие пары символов также представляют собой специальные символы:

`<= >= := .. (* *) (. .)`

Список слов, зарезервированных в Турбо Паскале можно найти в справке программы.

Идентификаторы

Идентификаторами обозначаются имена констант, типов, переменных, процедур, модулей, программ и полей в записях. Идентификатор может иметь любую длину, однако только первые его 63 символа являются значимыми. Идентификатор должен начинаться с буквы и не может содержать пробелов. После первого символа идентификатора можно использовать буквы, цифры и символы подчеркивания. В идентификаторах можно использовать как строчные, так и прописные буквы (компилятор их не различает).

Приведем несколько примеров идентификаторов:

`Writeln`

`Exit`

`Real2String`

`System.MemAvail`

`Dos.Exec`

`WinCrt.WhereX`

Числа

Для чисел, представляющих собой константы целого и вещественного типа, используется обычная десятичная запись. Целая константа в шестнадцатичном формате имеет в качестве префикса знак доллара (\$). Научное обозначение (E или e с показателем степени) в вещественных типах читается, как "на десять в степени". Например, $7E-2$ означает $7 \cdot 10^{-2}$, а $12.25E+6$ или $12.25E6$ оба обозначают $12.25 \cdot 10^6$.

Числа с десятичными точками или показателями степени представляют собой константы вещественного типа. Остальные десятичные числа обозначают константы целого типа. Они должны принимать значения в диапазоне от -2147483648 до 2147483647.

Шестнадцатеричные числа обозначают константы целочисленного типа. Они должны находиться в диапазоне от \$00000000 до \$FFFFFFFF. Окончательный знак значения определяется шестнадцатеричной записью.

Строки символов

Строка символов представляет собой последовательность, содержащую ноль и более символов из расширенного набора символов кода

ASCII, записанную в одной строке программы и заключенную в одиночные кавычки (апострофы). Строка символов, ничего не содержащая между апострофами, называется нулевой строкой. Два последовательных апострофа в строке символов обозначают один символ – апостроф. Атрибут длины строки символов выражается действительным количеством символов между апострофами.

Приведем несколько примеров строк символов:

'TURBO'

'You"ll see'

'''

','

Комментарии

Следующие конструкции представляют собой комментарии и поэтому му игнорируются компилятором:

{ любой текст, не содержащий правую фигурную скобку }

(* любой текст, не содержащий звездочку/правую круглую скобку *)

Комментарий, содержащий знак доллара (\$) сразу после открывающей скобки { или (*, является директивой компилятора. За символом \$ следует мнемоника команды компилятора.

Типы данных

Целочисленный тип

В Турбо Паскале имеется пять предопределенных целочисленных типов: shortint (короткое целое), integer (целое), longint (длинное целое), byte (длиной в байт) и word (длиной в слово). Каждый тип обозначает определенное подмножество целых чисел, как это показано в следующей таблице.

Тип	Диапазон	Формат
короткое целое (Shortint)	-128 .. 127	8 бит со знаком
целое (Integer)	-32768 .. 32767	16 бит со знаком
длинное целое (Longint)	-2147483648 .. 2147483647	32 бита со знаком
длиной в байт (Byte)	0 .. 255	8 бит без знака
длиной в слово (Word)	0 .. 65535	16 бит без знака

Вещественный тип

Имеются следующие виды вещественных типов: вещественное (Real), с одинарной точностью (Single), с двойной точностью (Double), с повышенной точностью (Extended).

Тип	Диапазон	Значение цифры	Размер в байтах
вещественное (real)	от $2.9 \cdot 10^{-39}$ до $1.7 \cdot 10^{38}$	от 11 до 12	6
с одинарной точностью (single)	от $1.5 \cdot 10^{-45}$ до $3.4 \cdot 10^{38}$	от 7 до 8	4
с двойной точностью (double)	от $5.0 \cdot 10^{-324}$ до $1.7 \cdot 10^{308}$	от 15 до 16	8
с повышенной точностью (extended)	от $1.9 \cdot 10^{-4951}$ до $1.1 \cdot 10^{4932}$	от 19 до 20	10

Строковые типы

Значением строкового типа является последовательность символов с динамическим атрибутом длины (в зависимости от действительного числа символов при выполнении программы) и постоянным атрибутом размера в диапазоне от 1 до 255. Текущее значение атрибута длины можно получить с помощью стандартной функции Length.

Операции

После того, как данные получены программой (и присвоены переменным), их нужно каким-то образом обрабатывать, для этого в вашем распоряжении имеются операции.

Большинство операций в Паскале являются бинарными, то есть имеют два операнда; остальные являются унарными, то есть имеют только один операнд. Бинарные операции имеют обычную алгебраическую форму, например, $a + b$. Унарная операция предшествует своему операнду, например, $-b$.

В сложных выражениях порядок выполнения операций устанавливается правилами приоритета

Операции	Приоритет	Категории
@, not	Первый (высший)	унарные операции
*, /, div, mod, and, shl, shr	Второй	мультипликативные операции
+, -, or, xor	Третий	аддитивные операции
=, <, <=, >, >=, in	Четвертый (низший)	операции отношения

Последовательности операций с одинаковым приоритетом вычисляются слева направо. Выражения, заключенные в скобки, вычисляются первыми независимо от того, каковы предшествующие или последующие операции.

Операция присваивания

Базовой операцией является *операция присваивания*, например,

$A := B + C$. {переменной A присвоить результат выражения $B + C$ }

В Паскале знак присваивания представляет собой комбинацию из двоеточия и знака равенства ($:=$). В приведенном примере значение $B + C$ справа от знака присваивания присваивается переменной A , которая стоит слева.

Паскаль поддерживает стандартный набор *бинарных арифметических операций*, которые выполняются над целыми и вещественными значениями:

- умножение ($*$)
- деление целых чисел (div)
- деление вещественных чисел ($/$)
- взятие по модулю (остаток от деления целых чисел) (mod)
- сложение ($+$)
- вычитание ($-$)

Кроме того, Паскаль поддерживает унарные операции: унарный минус и унарный плюс.

Операции отношений

Операции отношения позволяют сравнивать два значения, возвращая в результате булевское значение True или False.

Ниже приводятся операции отношения, реализуемые в Паскале:

>	больше
>=	больше или равно
<	меньше
<=	меньше или равно
=	равно
<>	не равно
in	есть элемент

Логические операции

Имеется четыре логические операции – and, xor, or и not, которые хотя и аналогичны поразрядным операциям, но не тождественны им. Логиче-

ческие операции выполняются над логическими значениями (True и False), позволяя комбинировать выражения отношения, булевы переменные и булевские выражения.

Операции над строками

Единственной операцией над строками является операция +, которая используется для объединения двух строк.

Вывод

Может показаться забавным, что о выводе речь идет раньше, чем о вводе, однако программа без вывода информации не представляет никакой пользы. Вывод обычно принимает форму, зависящую от того, куда выводится информация: на экран (слова и изображения), или на запоминающее устройство (гибкие и жесткие диски), или в порт ввода/вывода (последовательный порт или порт печатающего устройства).

Процедура Writeln

Назначение Writeln – выводить информацию на экран. Ее формат прост и гибок в использовании:

Writeln(элемент, элемент,...);

где каждый элемент – это то, что нужно напечатать на экране. Элементом может быть целое или вещественное число (3,42, -1732.3), строка ('Hello, world') или булево значение. Кроме того, им может быть именованная константа, переменная или обращение к функции, если она возвращает значение, которое имеет целый, вещественный, символьный, строковый или булев тип. Все элементы печатаются в одной строке в заданном порядке. После этого курсор устанавливается в начало следующей строки. Если вы хотите оставить курсор на той же строке после последнего элемента, то используйте оператор:

Write(элемент, элемент,...);

Когда распечатываются элементы по оператору Writeln, между ними автоматически не вставляются пробелы; если Вы хотите разделить элементы пробелами, то Вы должны указать их сами, например:

Writeln(элемент, ' ', элемент, ' ',...);

Поэтому по следующим операторам будет получен указанный вывод:

A := 1; B := 2; C := 3;	
Name := 'Frank';	
Writeln(A,B,C);	123
Writeln(A, ' ', B, ' ', C);	1 2 3
Writeln('Hi', Name);	HiFrank
Writeln('Hi, ', Name, '.');	Hi, Frank.

Кроме того, можно использовать спецификаторы для определения ширины поля для данного элемента. В этом случае оператор имеет формат:

Writeln(элемент:ширина,...)

где ширина – целое выражение, определяющее общую длину поля, в котором должен быть записан элемент. Например, рассмотрим следующую программу и полученный в результате вывод:

A := 10; B := 2; C := 100;	
Writeln(A,B,C);	102100
Writeln(a:2,B:2,C:2);	10 2100
Writeln(A:3,B:3,C:3);	10 2100
Writeln(A,B:2,C:4);	10 2 100

Отметим, что элемент дополняется начальными пробелами слева с тем, чтобы соответствовать указанной длине поля. Само значение выравнивается справа.

Что если ширина поля меньше, чем необходимо? Во втором операторе Writeln из приведенного выше примера для C, имеющего значение 100, ширина поля указана 2, хотя необходима ширина 3. Как Вы видите в выводе, Паскаль увеличил ширину до минимального необходимого размера.

Этот способ выполняется для всех допустимых элементов: целых чисел, вещественных чисел, символов, строк и булевых выражений. Однако, при указании спецификатора ширины поля вещественные числа распечатываются в экспоненциальной форме:

X := 421.53;	
Writeln(X);	4.2153000000E+02
Writeln(X:8);	4.2E+02

Поэтому Паскаль позволяет добавить второй спецификатор ширины поля:

элемент:ширина:цифры.

Это второе значение указывает распечатать вещественное число в формате с фиксированной точкой и определяет, сколько цифр поместить после десятичной точки:

X := 421.53;	
Writeln(X:6:2);	421.53
Writeln(X:8:2);	421.53
Writeln(X:8:4);	421.5300

Ввод

Стандартный Паскаль имеет две основные функции ввода Read и Readln, которые используются для чтения данных, вводимых с клавиатуры. Общий формат этих операторов следующий:

Read(элемент, элемент, ...); или Readln(элемент, элемент, ...);

где каждый элемент представляет собой переменную целого, вещественного, символьного или строкового типа. Числа должны отделяться от других значений пробелами или нажатием клавиши "Enter".

Структура программы

Структура программы в Турбо Паскале имеют следующий формат:

program <Имя-программы>;

{раздел описаний}

begin

{раздел операторов}

end.

Раздел описания состоит из четырех секций описания: label (метки), const (константы), type (типы), var (переменные). Не все секции раздела описания должны присутствовать в каждой программе. Турбо-Паскаль позволяет иметь сколь угодно много секций описания, следующих в любом порядке друг за другом.

Пример программы на Паскале

program First;

var

a, b, c: real;

begin

writeln('Введите длины катетов a и b');

readln(a, b);

c := sqrt(a*a+b*b);

writeln('Длина гипотенузы c= ', c);

end.

Внутренние (встроенные) функции языка

Для работы с данными могут использоваться встроенные математические функции Турбо Паскаля:

ABS(X) – абсолютная величина, если аргумент целого типа, то результат целого типа, если аргумент вещественного типа, то результат вещественного типа.

SQR(X) – квадрат выражения X, аргумент и результат функции всегда целого типа.

TRUNC(X) – возвращается целая часть числа, аргумент вещественного типа, результат целого типа.

ROUND(X) – округление чисел, аргумент вещественного типа, результат целого типа.

Функции, возвращающие вещественный результат при вещественном или целом аргументе.

SQRT(X) – квадратный корень из X

SIN(X) – синус, аргумент в радианах

COS(X) – косинус, аргумент в радианах

ARCTAN(X) – арктангенс, результат в радианах

LN(X) – натуральный логарифм (по основанию E)

EXP(X) – экспонента (E в степени X)

Функции, работающие с символьными переменными.

CHR(X) – Возвращает символ, соответствующий числу (коду) X, X : (0..255) (Если число не входит в указанный диапазон, то генерируется ошибка)

ORD(X) – возвращает целое число (код) символа X

Логические функции.

ODD(X) – X – значение целого типа, возвращает TRUE, если X – нечетно, иначе FALSE

Задания для самостоятельной работы

Разработать программу на Турбо Паскале, которая обеспечивает решение поставленной задачи в соответствии с предложенным вариантом.

№ варианта	Номера задач				
1	1	6	8	11	14
2	2	7	9	12	15
3	3	5	10	11	16
4	4	6	9	13	17
5	1	7	10	11	15
6	2	5	8	12	16
7	3	6	10	12	17
8	4	7	8	13	14
9	1	5	9	11	16
10	2	6	8	13	17
11	3	7	9	12	14
12	4	5	10	13	15

Задачи для самостоятельного решения

1. Для заданного x вычислить значение функции $y = \operatorname{ctg}(x)$, $x \in (0; 2\pi)$
2. Вычислить значение функции в заданной точке x . $f(x) = \operatorname{tg}(x) \cdot \ln(x)$, где $x > 0$
3. Вычислить значение функции в заданной точке x . $f(x) = x^2 + 3 \cdot x^{1/2}$, где $x > 0$
4. Вычислить значение функции в заданной точке x .
 $f(x) = 3 \cdot x^3 + 2 \cdot x^2 - 2 \cdot x + \sin(x)$.
5. Найти произведение и сумму цифр заданного двухзначного целого числа
6. Вычислить произведение цифр четырехзначного целого числа.
7. Вычислить сумму цифр трехзначного целого числа.
8. По длине двух сторон треугольника и углу между ними (в градусах) найти длину третьей стороны и площадь этого треугольника
9. По координатам трех вершин треугольника найти его площадь и периметр.
10. По длине стороны и двум прилежащим углам найти периметр и площадь треугольника.
11. По заданным коэффициентам и правым частям уравнений системы
 $A1 \cdot X + B1 \cdot Y = C1$,
 $A2 \cdot X + B2 \cdot Y = C2$;
найти ее решение для линейно независимых уравнений.

12. Вычислить длину окружности, площадь круга и объем шара одного и того же заданного радиуса R .
13. Вычислить периметр, площадь и гипотенузу прямоугольного треугольника по заданным длине одного из катетов и противолежащему углу (в градусах)
14. Задан вектор с координатами (X_1, Y_1) и (X_2, Y_2) . Определить угол наклона вектора к оси OX .
15. Вычислить расстояние между двумя точками с заданными координатами (X_1, Y_1) и (X_2, Y_2) . Определить угол наклона прямой, проходящей через эти точки, к оси OY .
16. Заданы 2 вектора с координатами (X_1, Y_1, Z_1) и (X_2, Y_2, Z_2) . Определить угол между векторами.
17. Задан вектор с координатами (X, Y, Z) . Найти угол наклона этого вектора к оси OY .