

Язык программирования Паскаль

Лекция 2

План

1. Общие сведения
2. Алфавит языка
3. Структура программы на языке Паскаль
4. Типы данных
5. Выражения
6. Организация ввода/вывода
7. Операторы

Литература



Литература

1. Касторнов А.Ф., Евстратова Г.А. Язык программирования Паскаль : учебное пособие для вузов. - Череповец : ГОУ ВПО ЧГУ, 2010. - 117 с. - Библиогр.: С.114.
2. Электронный учебник по языку программирования Паскаль
[/http://pascal.guti.ru](http://pascal.guti.ru)

Общие сведения

- ▣ **Паскаль** (англ. *Pascal*) — язык программирования общего назначения.
- ▣ Язык назван в честь выдающегося французского математика, физика, литератора и философа Блеза Паскаля, который создал первую в мире механическую машину, складывающую два числа.



Общие сведения

- Язык Паскаль был создан Никлаусом Виртом в 1968-69 годах. Позиционировался как эффективный язык, основанный на технологии структурного программирования. Объектное расширение языка Паскаль было разработано в 1986 году фирмой Apple, язык получил имя Object Pascal.
- Особенности языка являются строгая типизация и наличие средств структурного (процедурного) программирования.
- Наиболее известной реализацией языка Object Pascal, обеспечившая его широкое распространение и развитие, является среда программирования Turbo Pascal фирмы Borland, выросшая затем в среду Delphi, в которой были внедрены значительные расширения языка.

Алфавит языка

Язык программирования Паскаль использует следующий набор символов:

- английские и русские буквы, которые могут быть как строчными, так и прописными
- арабские цифры
- знаки операций и знаки, входящие в составное обозначение операций:
+ - * / = <> < > <= >= :=
- ограничители:
. , ' () [] { } .. : ;
- специальные символы:
\$ # ^



Алфавит языка

Идентификаторы (имена переменных, констант программ, модулей, функций, процедур) записываются с помощью допустимых в Паскале символов и должны удовлетворять следующим правилам:

- имя должно состоять только из английских букв, цифр и символов подчеркивания;
- строчные и прописные буквы не различаются;
- имя может начинаться только с английской буквы или символа подчеркивания;
- длина идентификатора может быть любой, но значимыми являются только первые 63 символа.



Алфавит языка

Слова, употребляемые в большинстве языков программирования, в том числе и в Паскале, делятся на три группы:

- служебные (зарезервированные, ключевые) слова;
- предопределенные имена;
- идентификаторы, определяемые программистом.



Алфавит языка

- *Служебные слова* – это слова языка программирования, которые имеют специальное, раз и навсегда закрепленное за ними значение. В программе нельзя использовать идентификаторы, совпадающие по написанию ключевыми словами (например, нельзя обозначить переменную именем **begin**).
- Служебные слова языка программирования Паскаль:

and	end	nil	set
array	file	not	then
begin	for	of	to
case	function	or	type
const	goto	packed	until
div	if	procedure	var
do	in	program	while
downto	label	record	with
else	mod	repeat	



Алфавит языка

- ▣ *Предопределенные (стандартные) имена* также имеют специальный (заранее заданный) смысл. Однако программист может обходить закрепленные за ними значения и использовать их в качестве имен, определяемых программистом. Если программист не определит явно, для каких целей применяется то или иное стандартное имя в программе, оно будет восприниматься в присущем данному имени предопределенном смысле (например, предопределенными являются имена Integer, Writeln и др.).
- ▣ *Пользовательские имена* определяются программистом и должны быть явно объявлены в программе.

Структура программы на языке Паскаль

- Программа состоит из заголовка, раздела описаний и раздела операторов.
- Каждый из разделов может содержать комментарии, которые представляют собой произвольную последовательность символов, расположенную внутри фигурных скобок {}.



Заголовок программы

- В заголовке указывается **имя программы** и **список параметров**.

- Общий вид заголовка:

program name (input, output, x, y,...);

где name – имя программы; input – файл ввода; output – файл вывода; x, y – внешние файлы, используемые в программе.

- Заголовок может не иметь параметров.



Раздел описаний

- Блок программы состоит из шести разделов, следующих в строго определенном порядке:

- 1) раздел подключаемых модулей (библиотек) (uses);
- 2) раздел меток (label);
- 3) раздел констант (const);
- 4) раздел типов (type);
- 5) раздел переменных (var);
- 6) раздел процедур и функций.

- Каждый из первых пяти разделов начинается с соответствующего ключевого слова (uses, label, const, type, var), которое записывается один раз в начале раздела и отделяется от последующей информации пробелом, либо концом строки, либо комментарием.
-



Раздел подключаемых модулей (библиотек) (uses)

□ Данный раздел содержит имена подключаемых модулей (библиотек). Их имена отделяются друг от друга запятыми, в конце описания ставится знак «;»

□ Общий вид:

uses unit1, unit2, unit3...;

где unit1, unit2, unit3 – имена модулей (библиотек).

Например:

uses CRT, Graph;



Раздел меток (label)

- Любой оператор в программе может быть помечен меткой. Имя метки задается по правилам образования идентификаторов. В качестве метки также могут использоваться произвольные целые числа в диапазоне от 0 до 9999. Все метки, используемые в программе, должны быть перечислены в разделе описания меток.



Раздел меток (label)

- Общий вид:

label 11, 12, 13...;

где 11, 12, 13 – метки.

Пример:

label 5, 10, 100;

- Метка отделяется от оператора двоеточием.

Пример:

Пусть оператор $a := b$ имеет метку 20, тогда этот оператор выглядит так:

20: a := b;



Раздел констант (const)

▣ **Константа** – это идентификатор, обозначающий некоторую неизменную величину определенного типа. Все константы, используемые в программе, описываются в разделе `const`, в разделе операторов могут использоваться только имена констант. Это делает программу более наглядной и удобной при отладке и внесении изменений.

▣ **Общий вид:**

```
const a1 = c1; a2 = c2; ...
```

где `a1` – имя константы, `c1` – значение константы.



Раздел констант (const)

В Паскале используются следующие стандартные типы констант:

- ▣ **Целочисленные** константы. Могут быть определены посредством чисел, записанных в десятичном или шестнадцатиричном формате данных. Это число не должно содержать десятичной точки.
- ▣ **Вещественные** константы. Могут быть определены числами, записанными в десятичном формате данных с использованием десятичной точки.
- ▣ **Символьные** константы. Могут быть определены посредством некоторого символа (заключенного в апострофы).
- ▣ **Строковые** константы. Могут быть определены последовательностью произвольных символов (заключенных в апострофы).
- ▣ **Типизированные** константы. Представляют собой инициализированные переменные, которые могут использоваться в программе наравне с обычными переменными. Каждой типизированной константе ставится в соответствие имя, тип и начальное значение.



Раздел констант (const)

Пример:

const

pi = 3,1415926535; e = 2.718281828459; {вещественные константы}

symb = '?'; {символьная константа}

year: integer = 2001; {типизированная константа}



Раздел типов (type)

- Если в программе вводится тип, отличный от стандартного, то этот тип описывается в разделе type.
- Общий вид:

```
type t1 = описание_типа; t2 = описание_типа; ...
```

где t1 и t2 – идентификаторы вводимых типов.

Пример:

```
type color = (red, yellow, green, blue);
```

Здесь описан тип color, задаваемый перечислением значений.



Раздел переменных (var)

- Переменной называют элемент программы, который предназначен для хранения, коррекции и передачи данных внутри программы. Пусть в программе встречаются переменные $v11, v12, \dots$; все они должны быть описаны следующим образом:

var $v11, v12, \dots$: type1; $v21, v22, \dots$: type2; ...

где $v11, v12, \dots$ - имена переменных;

type1 – тип переменных $v11, v12, \dots$;

type2 – тип переменных $v21, v22, \dots$.

Пример:

var k, i, j: integer; a, b: real;

- Каждая переменная должна быть описана до ее использования в программе и отнесена к одному и только одному типу.



Раздел процедур и функций

- В этом разделе приводится описание пользовательских процедур и функций.



Раздел операторов

- Эта часть программы начинается с ключевого слова **begin** и заканчивается словом **end**, после которого должна стоять точка. Раздел операторов – это непосредственно выполняемая часть программы, состоящая из операторов.

Типы данных

Тип данных определяет:

- возможные значения данных;
- внутреннюю форму представления данных в ЭВМ;
- операции, которые могут выполняться над данными.



Типы данных

В Паскале выделяют следующие типы данных:

▢ *Простые*

- Порядковые
 - ▢ Целые
 - ▢ Логические
 - ▢ Символьные
 - ▢ Перечисляемые
 - ▢ Интервальные
- Вещественные

▢ *Структурированные*

- Массивы
- Строки
- Множества
- Записи
- Файлы

▢ *Указатели*



Типы данных

Порядковые типы, выделяемые из группы простых типов, характеризуются следующими свойствами:

- все возможные значения порядкового типа представляют собой ограниченное упорядоченное множество;
- к любому порядковому типу может быть применена стандартная функция `Ord`, которая в качестве результата возвращает порядковый номер конкретного значения в данном типе;
- к любому порядковому типу могут быть применены стандартные функции `Pred` и `Succ`, которые возвращают предыдущее и последующее значения соответственно;
- к любому порядковому типу могут быть применены стандартные функции `Low` и `High`, которые возвращают наименьшее и наибольшее значения величин данного типа.



Типы данных

Рассмотрим некоторые типы данных:

Идентификатор	Длина (байт)	Диапазон значений	Операции
Целые типы			
<code>integer</code>	2	-32768..32767	+, -, /, *, Div, Mod, >=, <=, =, <>, <, >
<code>byte</code>	1	0..255	+, -, /, *, Div, Mod, >=, <=, =, <>, <, >
<code>word</code>	2	0..65535	+, -, /, *, Div, Mod, >=, <=, =, <>, <, >
<code>shortint</code>	1	-128..127	+, -, /, *, Div, Mod, >=, <=, =, <>, <, >
<code>longint</code>	4	-2147483648..2147483647	+, -, /, *, Div, Mod, >=, <=, =, <>, <, >

Типы данных

Вещественные типы			
<code>real</code>	6	$2,9 \times 10^{-39} - 1,7 \times 10^{38}$	<code>+, -, /, *</code> <code>>=, <=, =, <, ></code>
<code>single</code>	4	$1,5 \times 10^{-45} - 3,4 \times 10^{38}$	<code>+, -, /, *</code> <code>>=, <=, =, <, ></code>
<code>double</code>	8	$5 \times 10^{-324} - 1,7 \times 10^{308}$	<code>+, -, /, *</code> <code>>=, <=, =, <, ></code>
<code>extended</code>	10	$3,4 \times 10^{-4932} - 1,1 \times 10^{4932}$	<code>+, -, /, *</code> <code>>=, <=, =, <, ></code>
Логический тип			
<code>boolean</code>	1	<code>true, false</code>	<code>Not, And, Or, Xor,</code> <code>>=, <=, =, <, ></code>
Символьный тип			
<code>char</code>	1	все символы кода ASCII	<code>+</code> <code>>=, <=, =, <, ></code>

Выражения

- Выражение задает правило вычисления некоторого значения. Выражение состоит из констант, переменных, функций, знаков операций и скобок.



Математические операции

Символ операции	Название операции	Пример
*	умножение	2*3 (результат: 6)
/	деление	30/2 (результат: 1.5E+01)
+	сложение	2+3 (результат: 5)
-	вычитание	5-3 (результат: 2)
div	целочисленное деление	5 div 2 (результат: 2)
mod	остаток от деления	5 mod 2 (результат: 1)



Логические операции

Над логическими аргументами в Паскале определены следующие операции:

- NOT – логическое отрицание («НЕ»)
- AND – логическое умножение («И»)
- OR – логическое сложение («ИЛИ»)
- XOR – логическое «Исключающее ИЛИ»



Логические операции

A	B	not A	A and B	A or B	A xor B
true	true	false	true	true	false
true	false	false	false	true	true
false	true	true	false	true	true
false	false	true	false	false	false



Операции отношения (сравнения)

К операциям отношения в Паскале относятся такие операции, как:

- равно (=);
- не равно ($\neq$);
- больше ($>$);
- меньше ($<$);
- больше или равно ($\geq$);
- меньше или равно ($\leq$).

В операциях отношения могут принимать участие не только числа, но и символы, строки, множества и указатели.



Приоритет операций

- Порядок вычисления выражения определяется приоритетом содержащихся в нем операций. В языке Паскаль принят следующий приоритет операций:
 - унарная операция not, унарная операция -
 - * / div mod and
 - + - or xor
 - = <> < > <= >=
- Переопределить порядок выполнения операций можно с помощью скобок.



Некоторые стандартные функции

Функция	Запись	Тип аргумента	Тип результата
Абсолютное значение x	<code>abs(x)</code>	integer, real	integer, real
Синус x	<code>sin(x)</code>	integer, real	real
Косинус x	<code>cos(x)</code>	integer, real	real
Арктангенс x	<code>arctan(x)</code>	integer, real	real
Квадратный корень из x	<code>sqrt(x)</code>	integer, real	real
Значение x^2	<code>sqr(x)</code>	integer, real	integer, real
Значение e^x ($e = 2.718281\dots$)	<code>exp(x)</code>	integer, real	real

Организация ввода

- Для ввода исходных данных чаще всего используется процедура ReadLn:

ReadLn(A1,A2,...,AK);

- Процедура производит чтение значений исходных данных и присваивает эти значения переменным A1, A2, ..., AK.
- Переменные, образующие список ввода, могут принадлежать к целому, вещественному, символьному, строковому типам. Чтение исходных данных логического типа в языке Паскаль недопустимо. При вводе данных происходит преобразование из внешней формы представления во внутреннюю, определяемую типом переменных.



Организация ввода

- Значения вводимых данных могут отделяться друг от друга пробелами и нажатием клавиш табуляции [Tab] и [Enter]. Не допускается разделение вводимых значений запятыми.
- Если процедура ReadLn указана без параметров, она ожидает нажатия пользователем клавиши [Enter].



Организация вывода

- Для вывода результатов работы программы на экран используются процедуры:

```
Write(A1,A2,...,AK);
```

```
WriteLn(A1,A2,...,AK);
```

где $A_1, A_2, \dots, A_K$ – величины или выражения целого, вещественного, логического или символьного типов, а также строки.

- Write производит вывод значений величин $A_1, A_2, \dots, A_K$ в одну строку. WriteLn производит строковый вывод.
- Для величин вещественного типа элемент списка вывода может иметь вид $A:K:M$, где A – величина или выражение вещественного типа, K – ширина поля вывода, M – число цифр в дробной части. K и M – константы целого типа.

Операторы

- Под оператором в языке Паскаль подразумевают описание действия.
- Операторы отделяются друг от друга только точкой с запятой. Если оператор стоит перед `until` или `else`, то в этом случае точка с запятой не ставится.



Оператор присваивания

□ Общий вид:

$v := a;$

где v — переменная, a — выражение, $:=$ операция присваивания. Выражение может содержать константы, переменные, названия функций, знаки операций и скобки.

Пример:

$f := 3 * C + 2 * \sin(x);$

□ В операции присваивания переменная v и выражение a должны иметь один и тот же тип, а для интервального типа — одно и то же подмножество значений.

Замечание. Разрешается присваивать переменной типа `real` выражение типа `integer`. Нельзя присваивать переменной типа `integer` выражение типа `real`.



Составной оператор

- Если при некотором условии надо выполнить определенную последовательность операторов, то их объединяют в один составной оператор.
- Составной оператор начинается ключевым словом **begin** и заканчивается словом **end**. Между этими словами помещаются составляющие операторы, которые выполняются в порядке их следования. После **end** ставится точка с запятой, а после **begin** — только пробел (либо комментарий). Слова **begin** и **end** играют роль операторных скобок.

Пример:

```
begin  i:=2;   k:=i/5  end;
```



Пример

Определить расстояние на плоскости между двумя точками с заданными координатами $M_1(x_1, y_1)$ и $M_2(x_2, y_2)$.

Решение задачи

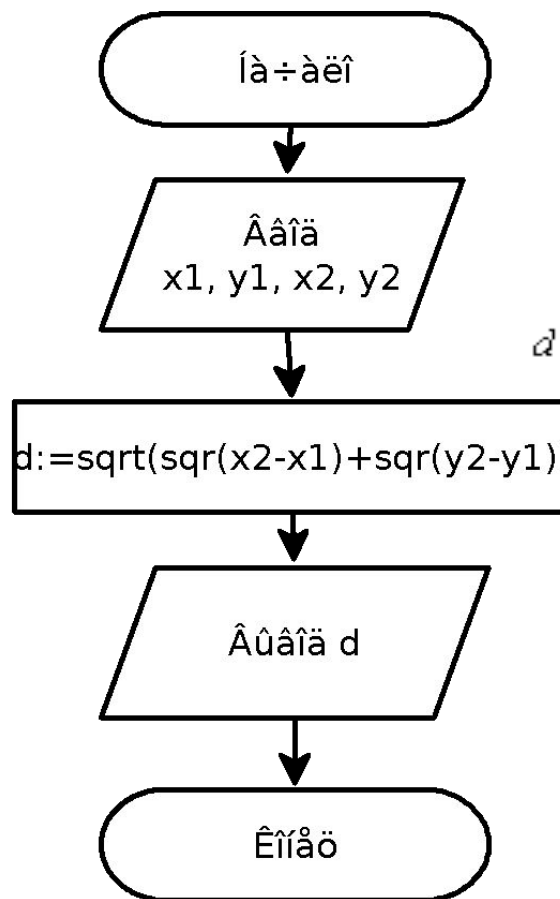
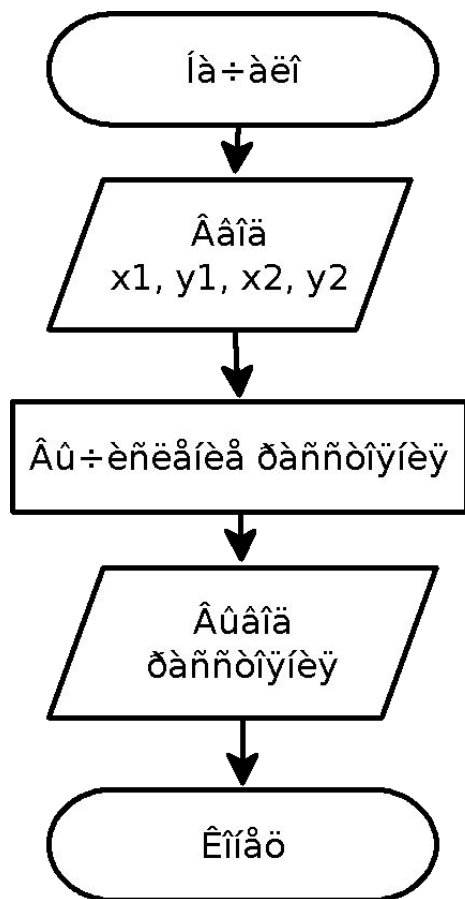
- Математическая модель: расстояние на плоскости между двумя точками $M_1(x_1, y_1)$ и $M_2(x_2, y_2)$ высчитывается по формуле:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$



Пример

- Составим схему алгоритма, а затем уточним содержимое блоков «Вычисление расстояния» и «Вывод расстояния».



$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Пример

Переводим блок-схему на язык Паскаль:

```
program example1;
```

```
var x1, x2, y1, y2: Integer; d: Real;
```

```
begin
```

```
    Writeln('Эта программа вычисляет расстояние между двумя  
    точками на плоскости');
```

```
    Writeln('Введите координаты двух точек:');
```

```
    Write('x1= '); Readln(x1);
```

```
    Write('y1= '); Readln(y1);
```

```
    Write('x2= '); Readln(x2);
```

```
    Write('y2= '); Readln(y2);
```

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

```
    d:=sqrt(sqr(x2-x1)+sqr(y2-y1));
```

```
    Writeln('d= ',d:6:2);
```

```
    Writeln('нажмите [Enter] для завершения работы программы');
```

```
    Readln;
```

```
end.
```



Пример

Проверим работу программы на нескольких тестах:

Набор данных				Результаты вычислений	
x1	y1	x2	y2	Ручных	Машинных
0	0	-4	3	5	5.00
0	0	0	0	0	0.00
10	3	7	8	5,83095	5.83
-23	5	-4	1	19,41649	19.42

С учетом округления при выводе, результаты ручных и машинных вычислений совпадают на выбранных наборах данных, что позволяет предположить правильность составления программы по выбранному алгоритму.