Структура консольного приложения

Структура консольного приложения имеет вид:

заголовок программы;

раздел подключения модулей;

раздел описаний;

тело программы.

Заголовок программы состоит из служебного слова program и имени программы, например:

```
program my_prog001 ;
```

В разделе подключения модулей используется служебное слово uses. В модулях находятся функции и процедуры языка.

Раздел описаний включает следующие подразделы:

- раздел описания констант;
- раздел описания типов;
- раздел описания переменных;
- раздел описания процедур и функций.

```
program имя_программы ;
uses modul1 , modul2 , . . . , moduln ;
const описания_констант ;
var описания_переменных ;
begin
операторы_языка ;
end .
```

пример текста программы на Free Pascal:

```
program one ;
const
a =7;
var
b , c : real ;
begin
c :=a +2; b:=c−a * sin ( a );
end .
```

Запуск программы

- После того как текст программы набран, его следует перевести в машинный код. Для этого необходимо вызвать транслятор с помощью команды Compile — Compile (комбинация клавиш Alt+F9).

- Для запуска транслированной программы необходимо выполнить команду Run — Run (комбинация клавиш Ctrl+F9), после чего на экране появляется окно командной строки, в котором пользователь и осуществляет диалог с программой. После завершения работы программы вновь появляется экран среды Free Pascal.

Комментарии

{Комментарий может выглядеть так!}

(* Или так . *)

//А если вы используете такой способ,

//то каждая строка должна начинаться

//с двух символов «косая черта».

Правильный идентификатор

- Идентификатор – это уникальное имя любого объекта (программы, переменной, константы, функции и т.д.). Имя формируется по правилу правильного идентификатора: имя не должно начинаться с цифры.
- Правильно:

```
A: integer;
a1:real;
_1a:char;
```

- Не правильно:

```
1a:integer;
```

Данные в языке Free Pascal

Перед использованием любая переменная должна быть описана. Описание переменной на языке Free Pascal осуществляется с помощью служебного слова var:

var имя_переменной : тип_переменной ;

Если объявляется несколько переменных одного типа, то описание выглядит следующим образом:

var переменная_1, переменная_2,..., переменная_N: тип_переменных ;

Например:

```
var
ha : integer ; //Объявлена
целочисленная //переменная.
hb , c : real ; //Объявлены две
//вещественные переменные.
```

Константа — это величина, которая не изменяет своего значения в процессе выполнения программы. Описание константы имеет вид:

```
const имя_константы = значение ;
```

Например:

```
const
  h=3; //Целочисленная константа.
  bk= −7.521; //Вещественная константа.
  c= ’ abcde ’ ; //Символьная константа.
```

Типы данных

Символьный тип данных

Описывают символьный тип с помощью служебного слова char. Например:

```
var c : char ;
```

В тексте программы значения переменных и константы символьного типа должны быть заключены в апострофы: ’a’, ’b’, ’+’.

Целочисленный тип данных

Тип	Диапазон	Размер, байт
Byte	$0 \ldots 255$	1
Word	$0 \ldots 65535$	2
LongWord	$0 \ldots 4294967295$	4
ShortInt	$-128 \ldots 127$	1
Integer	$-2147483648 \ldots 2147483647$	4
LongInt	$-2147483648 \ldots 2147483647$	4
Smallint	$-32768 \ldots 32767$	2
Int64	$-2^{63} \ldots 2^{63}$	8
Cardinal	$0 \ldots 4294967295$	4

Описание целочисленных переменных в программе может быть таким:

```
var
b : byte ;
i , j : integer ;
W: word ;
L_1 , L_2 : longint ;
```

Вещественный тип данных

Тип	Диапазон	Кол-во знач-х цифр	Размер, байт
Single	$1.5E45 \ldots 3.4E+38$	7—8	4
Real	$2.9E-39 \ldots 1.7E+38$	15—16	8
Double	$5.0E-324 \ldots 1.7E+308$	15—16	8
Extended	$3.4E-4932 \ldots 3.4E+4932$	19—20	10
Comp	$-2^{63} \ldots 2^{63}$	19—20	8
Currency	$-922337203685477.5808 \ldots 922337203685477.5807$	19—20	8

http://habrahabr.ru/post/112953/

Примеры описания вещественных переменных:

```
var
r1 , r2 : real ;
D: double ;
```

Логический тип данных

Данные логического типа могут принимать только два значения: истина (true) или ложь (false).

Тип	Размер, байт
Boolean	1
ByteBool	1
WordBool	2
LongBool	4

Пример объявления логической переменной:

```
var FL : boolean ;
```

Операции и выражения

Операция	Действие	Тип операндов	Тип результата
+	сложение	целый/вещественный	целый/вещественный
+	сцепление строк	строковый	строковый
-	вычитание	целый/вещественный	целый/вещественный
*	умножение	целый/вещественный	целый/вещественный
/	деление	целый/вещественный	вещественный
div	целочисленное деление	целый	целый
mod	остаток от деления	целый	целый
not	арифметическое/логическое отрицание	целый/логический	целый/логический
and	арифметическое/логическое И	целый/логический	целый/логический
or	арифметическое/логическое ИЛИ	целый/логический	целый/логический
xor	арифметическое/логическое исключающее ИЛИ	целый/логический	целый/логический

shl	сдвиг влево	целый	целый
shr	сдвиг вправо	целый	целый
in	вхождение в множество	множество	логический
<	меньше	не структурированный	логический
>	больше	не структурированный	логический
<=	меньше или равно	не структурированный	логический
>=	больше или равно	не структурированный	логический
=	равно	не структурированный	логический
<>	не равно	не структурированный	логический

- div — целочисленное деление (возвращает целую часть частного, дробная часть отбрасывается), например, 17 div 10 = 1;
- mod — остаток от деления, например, 17 mod 3 = 2.

В сложных выражениях порядок, в котором выполняются операции, соответствует приоритету операций. В языке Free Pascal приняты следующие приоритеты:

1) not.

2) *, /, div, mod, and, shl, shr.

3) +, –, or, xor.

4) =, <>, >, <, >=, <=.

Использование скобок в выражениях позволяет менять порядок вычислений.