

Лекция №10

- Программирование на языке Pascal. Строки.

Строки

- Тип данных **STRING** (строка) относится к структурированным типам данных.
- Тип данных **STRING** предназначен для обработки строк (**цепочек символов**).
- Переменная типа **STRING** состоит из цепочки символов.

Описание строк

- В разделе `var` строки описываются следующим образом:

```
var <имя_строки>: string [<длина>];
```

- **[<длина>]** – количество символов в строке.

Строки

- Если <длина> не указана, то считается, что в строке 255 СИМВОЛОВ.
- Компоненты строки нумеруются начиная с 0.
- **Нулевой байт хранит длину строки.**

Строки

Примеры описаний:

- **var**

**s1: string[10]; (*строка длиной 10
СИМВОЛОВ*)**

**s2: string; (*строка длиной 255
СИМВОЛОВ*)**

Строки

- Необходимо отметить, что один символ и строка длиной в один символ

```
var c: char;    s: string[1];
```

не эквивалентны друг другу, так как строка относится к структурированным типам данных, а не к базовым порядковым типам.

Неименованные константы Pascal

- В языке Pascal последовательность любых символов, **заклученная в апострофы**, воспринимается как **символ или строка**.

Например:

- `c:='z'; {c: char}`
- `s:='abc'; {s: string}`

Неименованные символьные константы Pascal

- Константе автоматически присваивается "минимальный" тип данных, достаточный для ее представления: `char` или `string[k]`. Поэтому попытка написать
- `c:='zzz'; {c: char}` вызовет сообщение об ошибке.

Неименованные символьные константы Pascal

- Если константа длиннее той переменной-строки, куда ваша программа пытается ее записать, то в момент присваивания произойдет **усечение** ее до нужной длины.
- **Пустая строка** задается двумя последовательными апострофами:
- `st:= '';`

Операции со строками

- В TP существуют два пути обработки переменных типа `STRING`:
 - обработку всей строки как единого целого, например:
`string_1 := 'Это - строка !';`

Операции со строками

- строка рассматривается как составной объект, состоящий из отдельных символов и доступ к отдельным символам строки осуществляется **по номеру их позиции**, т. е. как к элементу одномерного массива:

```
string_l[1] := 'A';  
writeln ( c [1], c [3]).
```

Операции над строками

- В языке Pascal определена операция сложения строк (конкатенация).

Пример:

```
VAR S1, S2, S3: String;  
BEGIN  
  S1:='Моя';  
  S2:=' программа';S3:=S1+S2; WriteLn (S3);  
END.
```

Операции над строками

- **Функции преобразования типа:**
StrToInt() – строка в целый тип;
StrToFloat () – строка в вещественный тип;
IntToStr() –целый тип в строковый;
FloatToStr() – вещественный тип в строковый.

Стандартные процедуры и функции для строк

- **Функция Length (длина)**
- позволяет определить **фактическую длину** текстовой строки, хранящейся в указанной переменной:

```
VAR Words: String;
```

```
Begin Write ('Введите, пожалуйста, слово: ');  
ReadLn (Words); WriteLn; WriteLn ('Это слово  
состоит из ', Length (Words):3, ' букв!');END.
```

Записи (records)

- **Запись** – структура данных, состоящая из конечного числа компонентов, называемых **полями**.
- **Поля** записи могут быть различных типов. Каждое поле имеет имя. Запись, как единое целое, занимает непрерывную область памяти.

Записи (records). Объявление.

```
type имя_типа_записи = record  
    имя_поля1 : тип_поля;  
    имя_поля2 : тип_поля;  
    ...  
    имя_поляN : тип_поля;  
end;
```


Записи (records). Объявление.

```
type    cars = record
        name : string [25];
        price : real;
        number : integer;
end;
```

Записи (records). Объявление.

```
var  g1, g2 : cars;  
      tabl : array [1 .. 100] of cars;  
      student : record  
name : string [30];  
group : byte;  
marks : array [1 .. 5] of byte;  
end;
```

Обращение к полям записи

<Имя_переменной>.<имя поля> := <выражение>

Stud.name:='Олег';

Stud.gend:='М';

Stud.kurs:=3; ... и т.д.

Обращение к полям записи

Или

```
With Stud do Begin
    name:='Олег';
    gend:='М';
    kurs:=3;
End;
```

Инициализация записей.

```
With g1 do begin
    price := 200;
    number := 12;
end;
```