Строки в Pascal

Выполнила ученица 10А класса: Щукина Алина

Строки в Паскале – это данные типа **string**. Они используются для хранения последовательностей символов

- Примеры описания строк:

```
type
      str_type = string[12];
Const
       n = 50;
 var
    s1: string;
    s2, s3: str_type;
    s4: string[n];
    s5, s6, s7: string[7]; …
```

Для обработки строковой информации в Турбо Паскаль введен строковый тип данных

Строкой в Паскале называется последовательность из определенного количества символов

- В Паскале строке соответствует тип данных String.

  var Имя : string [Длина];

  Если длина не указана, выделяется память под строку до 255 символов.

- var s1:string; строка 255 символов
- var s2:string[20]; строка 20 символов

Объявление типизированной константы для типа **string** осуществляется так:

const s: **string** = 'FreePascal'

- **Строковые константы** записываются как последовательности символов, ограниченные апострофами.
- S: = 'Текстовая строка'


Длина строки
Индексы символов строки
0 1 2 3 4 5 6 7 8 9 10 11 12 14 14 15 16 17 18 19 20
Строка Str:
14 М А М А _ М Ы Л А _ Р А М У
Str[4]
Пустые байты

Функции и процедуры строковых величин.

Операции над строками

Строки можно присваивать друг другу. Если максимальная длина переменной слева меньше длины присваиваемой строки, то лишние символы справа отбрасываются

```
...
s1 := 'this is text';
s2 := s1;
...
```

Строки можно объединять с помощью *операции конкатенации*, которая обозначается знаком +.

```
...
s1 := 'John';
s2 := 'Black';
s1 := s1 + ' ' + s2;
...
```

Строки можно сравнивать друг с другом с помощью *операций отношения*. При сравнении строки рассматриваются посимвольно слева направо, при этом сравниваются коды соответствующих пар символов. Строки равны, если они имеют одинаковую длину и посимвольно эквивалентны. В строках разной длины существующий символ всегда больше соответствующего ему отсутствующего символа. Меньшей будет та строка, у которой меньше код первого несовпадающего символа (вне зависимости от максимальных и текущих длин сравниваемых строк).

'abc' > 'ab' (true)

'abc' = 'abc' (true)

'abc' < 'abc ' (false)

Имя строки может использоваться в процедурах ввода-вывода. При вводе в строку считывается из входного потока количество символов, равное длине строки или меньшее, если символ перевода строки (который вводится нажатием клавиши Enter) встретится раньше. При выводе под строку отводится количество позиций, равное ее фактической длине.

```
...
readln (s1);
write (s1);
...
```

К отдельному символу строки можно обращаться как к элементу массива символов, например s1[3]. Символ строки совместим с типом **char**, их можно использовать в выражениях одновременно, например:

```
...
s1[3] := 'h';
writeln (s2[3] + 'r');
...
```

Можно осуществлять коррекцию любого символа строковой переменной, для чего в соответствующем операторе достаточно указать имя переменной типа **string**, вслед за которым в квадратных скобках задается

номер ее элемента

```
str[3]:='j'
```

Элементы строки нумеруются с единицы, т.к. в каждой строковой переменной имеется элемент с номером 0, в котором в виде символа хранится длина текущей строки. Чтобы узнать текущую длину, достаточно применить функцию **ord** к нулевому элементу строки.

```
...
writeln(ord(st[0]))
...
```

Нулевой элемент строковой переменной можно корректировать. При этом будет изменяться текущая длина строки.

Например, выражение str[0]:=#50 устанавливает текущую длину равной 50.

Процедуры и функции для работы со строками

При работе со строками, как правило, возникает необходимость выполнять их копирование, вставку, удаление или поиск. Для эффективной реализации этих действий в Паскале предусмотрены стандартные процедуры и функции. Они кратко описаны ниже.

Функция **Concat (s1, s2, ..., sn)** возвращает строку, являющуюся слиянием строк s1, s2, ..., sn.

Функция **Copy (s, start, len)** возвращает подстроку длиной len, начинающуюся с позиции start строки s.

Процедура **Delete (s, start, len)** удаляет из строки s, начиная с позиции start, подстроку длиной len.

Процедура **Insert (subs, s, start)** вставляет в строку s подстроку subs, начиная с позиции start.

Функция **Length (s)** возвращает фактическую длину строки s, результат имеет тип byte.

Функция **Pos (subs, s)** ищет вхождение подстроки subs в строку s и возвращает номер первого символа subs в s или нуль, если subs не содержится в s.