

Урок 78. Строковый тип. Операции со строками

Строки имеют тип `string`, состоят из набора последовательно расположенных символов `char` и используются для представления текста.

Тип `string` в PascalABC.NET является классом и реализуется типом `System.String` платформы .NET. Символьный тип `char` занимает 2 байта и хранит Unicode-символ. Символы реализуются типом `System.Char` платформы .NET.

1. Length Длина строки

Строки могут иметь произвольную длину. Свойство `Length: integer` возвращает длину строки.

```
var s: string := 'Word';
writeln(s.Length); // 4
```

2. [i] Операция индексирования (индексное свойство)

К символам в строке можно обращаться, используя индекс: `s[i]` обозначает *i*-тый символ в строке, нумерация начинается с единицы. Если индекс *i* выходит за пределы длины строки, то генерируется исключение.

s[1]	s[2]	s[3]	s[4]	
'W'	'o'	'r'	'd'	0

\$0057, \$006F, \$0072, \$0064, \$0000

```
for var i := 1 to s.Length do
    writeln(s[i]);
```

3. <, >, <=, >=, =, <> Операции сравнения

Над строками определены операции сравнения: `<`, `>`, `<=`, `>=`, `=`, `<>`. Сравнение строк на неравенство осуществляется **лексикографически**: `s1 < s2` если для первого несовпадающего символа с номером *i* `s1[i] < s2[i]` или все символы строк совпадают, но `s1` короче `s2`.

```
writeln(s <> 'Word'); // False
writeln('abc' <= 'bcd'); // True
writeln('Word1' > s); // True
```

4. + Конкатенация (слияние строк)

К строкам и символам применима операция конкатенации (слияния) `+`, ее результат имеет строковый тип. Например, `'a'+'b'='ab'`.

```
writeln(s + 'book'); // Wordbook
```

К строкам также применима операция `+=`:

```
s += s1; // s := s + s1;
```

Строка может складываться с числом, при этом число предварительно преобразуется к строковому представлению:

```
s := 'Ширина: ' + 15; // s = 'Ширина: 15'
s := 20.5 + ''; // s = '20.5'
s += 1; // s = '20.51'
```

5. * Повторение (умножение) строки

Над строками и целыми определена операция *: $s*n$ и $n*s$ означает строку, образованную из строки s , повторенной n раз:

```
s := 'Word';  
writeln(s * 3);           // WordWordWord  
s := '*' * 10;            // s = '*****'  
s := 5 * 'ab'             // s = 'ababababab'  
s := 'd'; s *= 3;         // s = 'ddd'
```

6. in Операция проверки вложения подстроки в строку

Над строками определена операция `in` для проверки вложения подстроки в строку:

```
'bc' in 'abcde'    // True  
'c' in 'abd'      // False
```