Основы алгоритмизации и программирования.

Строки

Педагогическое образование, 3 семестр

Mayer Svetlana Fyodorovna

Символы (тип Char)

Символ хранится в 2-байтовой ячейке памяти в виде символа Юникода (UTF-16):
: 2^{16} = 65536 всего различных символов

Соответствие между символами и их кодами:

```
var c := 'Ю';
var n := Ord(c); // n := c.Code
Print(n); // 1070
c := Chr(n);
```

Другие функции:

```
Pred(c) – Предыдущий символ в таблице кодов
Succ(c) – Следующий символ в таблице кодов
```

Символьные константы: **#код** – символ с определенным кодом.

```
#10 – новая строка (Linux)
#13#10 – новая строка (Windows)
#9 – tab
Print('a'#10'b');
```

```
a
b
```

Методы класса типа char

```
•   char.IsLetter(c); // буква ли, boolean
•   char.IsDigit(c); // десятичное ли число, boolean
•   char.IsLower(c); // буква нижнего регистра, boolean
•   char.IsUpper(c); // буква верхнего регистра, boolean
•   char.IsPunctuation(c); // знак препинания, boolean

c := char.ToLower(c)
c := char.ToUpper(c)
```

Операции для типа char:

```
c1 < c2;// сравнение по коду
c1 > c2;
c in ['a','e','i','o','u',’y’]; // c принадлежит гласным
c in ['a'..'z','A'..'Z’]; // c принадлежит английским буквам
```

Некоторые выражения

с принадлежит диапазону:

```
c.InRange('a'..'z')
```

Трансформация символа в цифру:

```
var n := Ord(c) – Ord('0');
```

Увеличение кода символа на n:

```
c := Chr(Ord(c) + n);
  // или
Inc(c,2);
```

Цикл for по символам

счетчик цикла может быть типом char:

```
begin
  var vowels := ['a','e','i','o','u','y'];
  for var c := 'a' to 'z' do
    if c in vowels then
      Print(char.ToUpper(c));
  for var c := #0 to #255 do
    if char.IsPunctuation(c) then
      Print($'{c}{c.Code}  ');
end.
```

```
A E I O U Y
!33   "34   #35   %37   &38   '39   (40   )41   *42   ,44   -45   .46   /47   :58
;59   ?63   @64   [91   \92   ]93   _95   {123   }125   ¡161   «171   173   ·183
»187   ¿191
```

Строки

Строки имеют тип **string**. Строки состоят из символов.

s[i] – доступ к i-му символу строки s (индексирование с 1 – Delphi стандарт)
s.Length – длина строки

```
begin
  var s := 'Informatics';
  for var i:=1 to s.Length do
    Print(s[i]);
end.
```

```
I n f o r m a t i c s
```

Цикл foreach:

```
begin
  var s := 'Informatics';
  foreach var c in s do
    Print(c);
end.
```

```
I n f o r m a t i c s
```

Операции со строками

```
• s1 + s2  // конкатенация s1 и s2
• s1 += s2
• s1 < s2  // лексикографическое сравнение
•   'abcd' < 'aad'
• s * n    // конкатенация n копий строки s
• s[a:b]       // срез (индексация с 1)
• s[a:b:step]  // срез с шагом
• s?[3:5]   ;   // безопасный срез (не вызывает ошибок)
• s?[a:b:step] // безопасный срез с шагом
• s1 in s      //
```

Неэффективное изменение символов строк

Пример. Увеличьте все коды символов в s на 1

Неэффективное решение.

```
begin
  var s := 'abcdefghijjklmnopqrstuvwxy';
  for var i:=1 to s.Length do
    s[i] := Succ(s[i]);
  Print(s);
end.
```

```
bcdefghijkklmnopqrstuvwxyz
```

В памяти:

```
begin
  var s := 'abcdefghijjklmnopqrstuvwxy';
  s[3] := 'd';
  // In memory: s := s[1:3] + 'd' + s[4:];
end.
```

Как изменять строки эффективно

Тип **StringBuilder** для эффективного **изменения строк**.

Для StringBuilder индексация начинается с 0

```
begin
  var s := 'abcdefghijjklmnopqrstuvwxy';
  var sb := new StringBuilder(s);
  for var i:=0 to sb.Length-1 do
    sb[i] := Succ(sb[i]); // эффективно!!
  s := sb.ToString();
  Print(s);
end.
```

```
bcdefghijkklmnopqrstuvwxyz
```

Как создавать строки эффективно

Использовать тип **StringBuilder**!

Пример. Создать строку `'abcdefghijjklmnopqrstuvwxyz'`

Неэффективное решение.

```
begin
  var s := '';
  for var c:='a' to 'z' do
    s += c;
  Print(s);
end.
```

Эффективное решение.

```
begin
  var sb := new StringBuilder;
  for var c:='a' to 'z' do
    sb += c;
  var s := sb.ToString();
  Print(s);
end.
```

Методы строк

Методы:

- Предположим, что строки индексируются с 0
- не изменяет строки, но возвращает новые строки

```
s.ToLower
s.ToUpper
s.Contains(s1)
s.CompareTo(s1)
s.StartsWith(s1)
s.EndsWith(s1)
s.IndexOf(s1,from=0),
s.IndexOfAny(array of char)
s1 := s.Insert(from,s1)
s1 := s.Remove(from,len)
s3 := s.Replace(s1,s2)
s1 := s.SubString(from,len)
s1 := s.Trim
var a: array of string := s.ToWords // s.Split(' ')
string.Join(' ',ss)
```

Примеры

Пример 1. Вычислите сумму всех цифр в s.

```
begin
  var s := 'abc1d2ef3g44h5';
  var sum := 0;
  foreach var c in s do
    if char.IsDigit(c) then
      sum += Ord(c) - Ord('0');
  Print(sum);
end.
```

Пример 2. Использование методов последовательности

```
begin
  var s := 'abc1d2ef3g44h5';
  var sum := s.Where(c -> char.IsDigit(c)).Sum(c -> Ord(c) - Ord('0'));
  Print(sum);
end.
```

Пример 3. Количество английских букв в строке

```
var count := s.Count(c -> c.InRange('a','z') or c.InRange('A','Z'));
```

Примеры (2)

Пример 4. Отсортируйте все слова в строке, удалите лишние пробелы

```
begin
  var s := '  cc  bb  aa gg ee dd ff ';
  s := s.ToWords.Order.JoinIntoString;
  Print(s);
end.
```

```
aa bb cc dd ee ff gg
```

Пример 5. Сколько раз подстрока'мехмат' встречается в s?

```
begin
  var s := махмат abc махматмахматдемахмат ';
  var count := 0;
  var ind := s.IndexOf('махмат');
  while ind<>-1 do
  begin
    count += 1;
    ind := s.IndexOf('махмат',ind+1);
  end;
  Print(count);
end.
```

Примеры (3)

Пример 6. Создайте строку `'abcdefghjklmnopqrstuvwxyz'`

```
var s := Range('a','z').JoinIntoString;
```

Пример 7. Инвертировать все символы, кроме первого и последнего

```
var s :='hello';
var n := s.Length;
s := s[:2] + s[n-1:1:-1] + s[n:];
print(s) // hlleo
```

Пример 8. Выведите все слова, которые начинаются и заканчиваются на одной и той же букве

```
begin
  var s := 'ara kran kok lom byte shine';
  s.ToWords.Where(s1 -> s1.First = s1.Last).Print;
end.
```

```
ara kok
```

Примеры (4)

Пример 9. Выведите все слова со вторым символом = 'p'.

```
begin
  var s := 'a appa epda a c upsa';
  s.ToWords.Where(w -> w?[2:3]='p').Print;
end.
```

```
appa epda upsa
```

Пример 10. Количество слов с каждой буквы

```
begin
  var s := 'ara kran kok lom byte alla bit brr neck kodu ka';
  s.ToWords.GroupBy(ss -> ss[1]).Select(g -> (g.Key,g.Count))
    .OrderByDescending(t -> t[1]).PrintLines;
end.
```

```
(k,4)
(b,3)
(a,2)
(l,1)
(n,1)
```

Трансформация число ↔ строка

Номер преобразования в строку :

```
var r := 3.14;
var sr := r.ToString;
var i := 345;
var si := i.ToString;
```

Преобразование строки в число (возможно, с возникновением исключения):

```
begin
  var s := '345';
  var i := integer.Parse(s);
  s := '3.14';
  var r := real.Parse(s);
  s := 'abc';
  r := real.Parse(s); // исключение
end.
```

Трансформация число ↔ строка(2)

пример 11. Найти сумму всех реальных чисел в строке

```
begin
  var s := '12.5 ag 3.4 d 4.7  dfgh 5.11 asg';

  var r: real;
  var sum := 0.0;
  foreach var w in s.ToWords do
    if real.TryParse(w,r) then
      sum += r;
  Print(sum);
end.
```

Файл как последовательность строк

Функция ReadLines(FName) открывает файл FName, считывает строки из него как последовательность строк, затем закрывает файл.

Функция WriteLines(FName,ss) открывает файл FName, записывает последовательность строк ss в файл, затем закрывает его

Пример. Запишите в файл b.txt все строки из файла a.txt кроме 1-й строки:

```
begin
  var s: sequence of string := ReadLines('a.txt');

  WriteLines('b.txt',s.Skip(1));
end.
```

Трансформация текстовых файлов

Пример. Удалите начальные и конечные пробелы в каждой строке файла.

```
begin
  var ss: sequence of string := ReadLines('a.txt');
  ss := ss.Select(s -> s.Trim);
  WriteLines('b.txt',ss);
end.
```

Q & A