Programação Estruturada e Orientada a Objetos

Strings

2013

O que veremos hoje?

- Introdução
- Revisão de String
- Exercícios

Transparências baseadas no material do Prof. Gilbert Azevedo

Strings

- string
  - É um tipo de dados que representa uma coleção de caracteres
  - Valores do tipo *string* são amplamente utilizados no desenvolvimento de aplicativos por representar, em geral, todos os textos que aparecem em um programa
  - Em C#, o tipo *string* é uma classe, o que facilita a sua utilização em relação ao tipo *string* de outras linguagens de programação

Declaração de String

- A declaração de variáveis *string* é feita de forma análoga à declaração de variáveis de outros tipos primitivos (int, double, ...)
  - Ex: Declaração de *strings* sem atribuição inicial
    - string s1, s2;
  - Ex: Declaração de *strings* com atribuição inicial
    - string s3 = "C++", s4 = "Algoritmos";

Operações Básicas com Strings

- Atribuição
  - O operador de atribuição "=" é utilizado para atribuir um valor a uma *string*
    - string s;
    - s = "Algoritmos";
    - s = "";
- Indexação
  - O operador de indexação "[ ]" é usado para acessar (ler) cada caractere da *string* individualmente.
  - O índice do primeiro caractere é zero.
    - string s = "Algoritmos";
    - Console.WriteLine(s[0]); // Escreve A

Operações Básicas com Strings

- Entrada de dados
  - O método ReadLine da classe Console é utilizado para ler uma string do teclado
    - string s;
    - s = Console.ReadLine();
- Saída de dados
  - O método WriteLine é usado para mostrar o conteúdo da variável string
    - string s = "Algoritmos";
    - Console.WriteLine(s);

Comparação entre Strings

- A igualdade entre *strings* é realizada através dos operadores relacionais (== e !=)
- A comparação é realizado pelo método CompareTo
  - string s1 = "G", s2 = "g";
  - s1 == s2; // Falso
  - s1 != s2; // Verdadeiro
  - s1.CompareTo(s2); // 1
  - "g".CompareTo(s2); // 0
  - "A".CompareTo(s1); // -1
  - "a".CompareTo(s2); // -1
  - "Z".CompareTo(s1); // 1
  - "z".CompareTo(s2);// 1

Concatenação de Strings

- Concatenação
  - É o termo normalmente utilizado para indicar a união entre duas ou mais *strings*
  - O operador "+" é utilizado para realizar a concatenação
    - string s = "Bota";
    - s = s + "fogo"; // Atribui "Botafogo"
    - s = s + 's'; // Atribui "Botafogos"
    - s = 'a' + 'b'; // Erro: Resultado é int

Concatenação de Strings

- Concatenação duas strings

```
string s1 = "Olá ";
string s2 = "Coleguinha!";
Console.WriteLine(">>>> '{0}'!", string.Concat(s1,s2));
```

- Concatenação array de strings

```
string[] s = { "Tudo ", "funciona ", "coleguinha "};
Console.WriteLine(string.Concat(s));
```

Contém

- Verifica se uma string contém outra

```
string s1 = "Os coleguinhas são estudiosos!";
    string s2 = "são";
    bool b;
    b = s1.Contains(s2);
    Console.WriteLine("existe s2 dentro de s1?: {0}",
b);
```

Tamanho da String

- int Length;
  - Retorna o número de caracteres da *string*
  - string s = "Algoritmos";
  - int i = s.Length; // Atribui 10 a i
  - Console.WriteLine(s.Length); // Escreve 10

Métodos da Classe string

- string Remove(int startIndex);
- string Remove(int startIndex, int count);
  - Remove *count* caracteres da string, iniciando na posição *startIndex*. Se *count* é omitido, remove os caracteres a partir de *startIndex*.
  - Obs: A string original não é alterada
  - string s = "Algoritmos";
  - Console.WriteLine(s.Remove(4)); //"Algo";
  - Console.WriteLine(s.Remove(4,3)); //"Algomos"

Métodos da Classe string

- string Substring(int startIndex);
- string Substring (int startIndex, int length);
  - Retorna *length* caracteres da string, iniciando na posição *startIndex*. Se *length* é omitido, retorna os caracteres a partir de *startIndex*.
  - Obs: A string original não é alterada
  - string s = "Algoritmos";
  - Console.WriteLine(s.Substring(4)); //"ritmos";
  - Console.WriteLine(s.Substring(4,3)); //"rit"

Métodos da Classe string

- int IndexOf(char value);
- int IndexOf(string value);
  - Retorna a posição inicial do caractere ou string *value* dentro da string
  - Retorna -1 se *value* não for encontrado
  - string s = "Algoritmos";
  - Console.WriteLine(s.IndexOf('o')); // 3
  - Console.WriteLine(s.IndexOf("go")); // 2
  - Console.WriteLine(s.IndexOf("x")); // -1

Métodos da Classe string

- Split(Char[]) – retorna um array de string delimitados através do array de caracteres passado como parâmetro
- ToLower() – retorna uma string com todos os caracteres em caixa baixa
- ToUpper() – retorna uma string com todos os caracteres em caixa alta
- Trim() – remove os espaços no inicio e fim da string
- StartsWith(String) – determina se a string combina com string passada como parâmetro

Métodos da Classe string

- PadLeft(Int32) – retorna uma string com n caracteres alinhado a esquerda através de espaços em branco
- Join – concatena elementos de array de string com um separador

```
string[] sArr = {"Os", "coleguinhas", "são", "estudiosos!"};
Console.WriteLine(String.Join(", ", sArr));
```

Exercícios

1. Ler uma string e contar quantas palavras tem nela.

   Ex.: "Programar não é moleza" $\rightarrow$ 4 palavras
2. Ler uma string e mostrar de trás para frente.

   Ex.: "Programar não é moleza" $\rightarrow$ "azelom é oãn ramargorP"
3. Ler uma string e mostrar as iniciais de cada palavra.

   Ex.: "Programar não é moleza" $\rightarrow$ "Pném"
4. Ler uma string com vários espaços entre as palavras e mostrar com somente um espaço entre as palavras.

   Ex.: " Programar não é moleza " $\rightarrow$ "Programar não é moleza"
5. Ler uma string e escrever cada palavra desta de trás para frente.

   Ex.: "Programar não é moleza" $\rightarrow$ "ramargorP oãn é azelom"

Dúvidas