Aula 14

Prof: Henrique Augusto Maltauro

Desenvolvendo Algoritmos

Listas e Arrays

Listas e Arrays

Uma lista ou array é uma estrutura de dados que contém uma lista de objetos ordenados, chamados elementos.

Esses elementos são referenciados por um índice.

Os elementos podem ser de qualquer tipo, string, caractere, numérico, data, etc.

Listas e Arrays

- C#: Array

Na linguagem C#, os arrays possuem o índice com base zero, ou seja, o primeiro elemento dele possui o índice zero.

Listas e Arrays

- C#: Array

```
int[] numeros = new int[5];

numeros[0] = 1;
numeros[1] = 3;
numeros[2] = 9;
numeros[3] = 15;
numeros[4] = 22;
```

Listas e Arrays

- C#: Array

No caso do C#, nós não vamos trabalhar muito com arrays, por que dificilmente se é utilizado o array em C#.

O problema do array, é que ele tem um tamanho fixo, ou seja, se ele for criado com 2 elementos, ele vai sempre ter apenas 2 elementos.

Listas e Arrays

- C#: System.Collections.Generic

É um grupo de estruturas de dados, que possibilita utilizar listas de diversas formas.

A vantagem é que eles têm um tamanho dinâmico, ou seja, o que limita a quantidade de elementos que essas listas possuem, é a memória da máquina em que o programa estiver sendo executado.

Listas e Arrays

- C#: System.Collections.Generic

Ao total, são umas 50 estruturas diferentes que o C# disponibiliza para o nosso uso.

Mas a gente vai se concentrar na principal, que é o List.

Listas e Arrays

- C#: List

A estrutura list define uma lista fortemente tipada, ou seja, é preciso especificar qual o tipo de dado que aquela lista está armazenando.

Ela fornece diversos métodos para pesquisar, classificar e manipular essas listas.

Listas e Arrays

- C#: List (declaração)

A declaração da estrutura list precisa ter a informação de qual o tipo de dados que ela irá armazenar.

Essa informação é passada entre os símbolos de menor e maior.

Listas e Arrays

- C#: List (declaração)

```
List<string> nomes;
```

Listas e Arrays

- C#: List (inicialização)

Como no C#, tudo é um objeto, a inicialização do list precisa ser feita através de um construtor.

Nós logo vamos aprender melhor o que é um construtor, mas basicamente, ele constrói a estrutura inicial para aquela variável.

Listas e Arrays

- C#: List (inicialização)

```
List<string> nomes = new List<string>();
```

Listas e Arrays

- C#: List (inicialização com valores)

```
List<string> nomes = new List<string>()
{
    "henrique",
    "gustavo",
    "fernando",
    "elizabeth"
};
```

Listas e Arrays

- C#: List (propriedades)

A estrutura list possui algumas propriedades que nos auxiliam no acesso dos elementos que a lista armazena.

Listas e Arrays

- C#: List (propriedades: [ ])

Da mesma maneira como nos arrays, é utilizado os colchetes para acessar os elementos dentro da lista.

Listas e Arrays

- C#: List (propriedades: [ ])

```
List<string> nomes = new List<string>()
{ "henrique", "gustavo", "fernando", "elizabeth" };

Console.WriteLine(nomes[0]); // Saída: henrique
Console.WriteLine(nomes[1]); // Saída: gustavo
Console.WriteLine(nomes[2]); // Saída: fernando
Console.WriteLine(nomes[3]); // Saída: elizabeth
```

Listas e Arrays

- C#: List (propriedades: Count)

O propriedade Count, retorna a quantidade de elementos que a lista possui.

Listas e Arrays

- C#: List (propriedades: Count)

```
List<string> nomes = new List<string>()
{ "henrique", "gustavo", "fernando", "elizabeth" };

Console.WriteLine(nomes.Count); // Saída: 4
```

Listas e Arrays

- C#: List (manipulação)

Como dito anteriormente, a estrutura list fornece uma série de métodos para a manipulação dos seus elementos.

No total, são 31 métodos, mas nós vamos focar apenas em 6 deles, que serão os principais para que possamos manipular normalmente as nossas listas.

Listas e Arrays

- C#: List (manipulação: Add)

O método Add, recebe um elemento, que vai ser adicionado no final da lista.

Listas e Arrays

- C#: List (manipulação: Add)

```
List<string> nomes = new List<string>();

nomes.Add("Henrique");
```

Listas e Arrays

- C#: List (manipulação: Insert)

O método Insert, recebe um índice e um elemento, que vai ser adicionado na lista no índice especificado.

Listas e Arrays

- C#: List (manipulação: Insert)

```
List<string> nomes = new List<string>();

nomes.Insert(2, "Henrique");
```

Listas e Arrays

- C#: List (manipulação: Contains)

O método Contains, recebe um elemento, e verifica se esse elemento existe dentro da lista, e retorna um valor booleano a partir disso.

Listas e Arrays

- C#: List (manipulação: Contains)

```
List<string> nomes = new List<string>()
{ "henrique", "gustavo", "fernando", "elizabeth" };

Console.WriteLine(nomes.Contains("henrique")); // Saída: true
Console.WriteLine(nomes.Contains("giovana"));  // Saída: false
```

Listas e Arrays

- C#: List (manipulação: Remove)

O método Remove, recebe um elemento, que vai ser removido da lista.

Listas e Arrays

- C#: List (manipulação: Remove)

```
List<string> nomes = new List<string>()
{ "henrique", "gustavo", "fernando", "elizabeth" };

nomes.Remove("henrique");
```

Listas e Arrays

- C#: List (manipulação: RemoveAt)

O método Remove, recebe um índice, que vai ser removido da lista.

Listas e Arrays

- C#: List (manipulação: RemoveAt)

```
List<string> nomes = new List<string>()
{ "henrique", "gustavo", "fernando", "elizabeth" };

nomes.RemoveAt(2);
```

Listas e Arrays

- C#: List (manipulação: Clear)

O método Clear, remove todos os elementos da lista.

Listas e Arrays

- C#: List (manipulação: Clear)

```
List<string> nomes = new List<string>()
{ "henrique", "gustavo", "fernando", "elizabeth" };

nomes.Clear();
```

Exercício

Exercício

1 - Receber um número int, e usando um List<int> de 1 a 10, calcular e exibir no console a tabuada do número informado.