



Linguagem de Programação II

Análise e Desenvolvimento de Sistemas

2º semestre de 2019

Prof. Me. Renato Carioca Duarte



Coleções

- Para muitos aplicativos, você desejará criar e gerenciar grupos de objetos relacionados. Há duas maneiras de agrupar objetos:
 - criando matrizes (array) de objetos
 - criando coleções de objetos
- As matrizes (array) são mais úteis ao criar e trabalhar com um número fixo de objetos fortemente tipados.
- As coleções fornecem uma maneira mais flexível de trabalhar com grupos de objetos. Ao contrário das matrizes, o grupo de objetos com o qual você trabalha pode crescer e reduzir dinamicamente conforme as necessidades do aplicativo são alteradas.
- Para algumas coleções, você pode atribuir uma chave para qualquer objeto que coloque na coleção para que você possa recuperar rapidamente o objeto usando a chave.
- Uma coleção é uma classe, portanto você deve declarar uma instância da classe antes de adicionar elementos a essa coleção.



Array

- Coleção estática.
- Pode referenciar o objeto nulo.
- São indexadas por zero: um array com elementos n é indexado de 0 para n-1.

```
using System;
```

```
class MainClass {  
    public static void Main (string[] args) {  
        int[] a;  
        a = new int[3];  
        a[1] = 35;  
        a[2] = -8;  
        Console.WriteLine(a[0]);  
        Console.WriteLine(a[1]);  
        Console.WriteLine(a[2]);  
        Console.ReadKey();  
    }  
}
```



Array de String

```
class MainClass {  
    public static void Main (string[] args) {  
        string[] a = new string[3];  
        a[0] = "AAA";  
        string s = "BBB";  
        a[1] = s;  
        a[2] = a[0] + " " + s;  
        Console.WriteLine(a[0]);  
        Console.WriteLine(a[1]);  
        Console.WriteLine(a[2]);  
        Console.ReadKey();  
    }  
}
```



Array de Objetos

Repl.it - EarlyWorthlessRoutine x +

repl.it/@RenatoCarioca/EarlyWorthlessRoutine

@RenatoCarioca/EarlyWorthlessRoutine No description

invite 2+ run share + new repl

Files

- main.cs
- Aluno.cs
- main.exe

main.cs saved

```
1 using System;
2
3 class MainClass {
4     public static void Main (string[] args) {
5         Aluno[] a = new Aluno[3];
6         a[0] = new Aluno("Bruno", 567);
7         a[1] = new Aluno("Carla", 123);
8         a[2] = new Aluno("Pedro", 864);
9         Console.WriteLine(a[0].getNome());
10        Console.WriteLine(a[1].getNome());
11        Console.WriteLine(a[2].getNome());
12    }
13 }
14
15
```

Aluno.cs(6,15): warning CS0414: The private field 'Aluno.m
atric'
is assigned but its value is never used
Compilation succeeded - 1 warning(s)
mono main.exe
Bruno
Carla
Pedro
[]



Array de Objetos

No rept podemos colocar todas as classes no mesmo arquivo .cs

The screenshot shows a Repl.it IDE window with the following code in `main.cs`:

```
1 using System;
2
3 class MainClass {
4     public static void Main (string[] args) {
5         Aluno[] a = new Aluno[3];
6         a[0] = new Aluno("Bruno", 567);
7         a[1] = new Aluno("Carla", 123);
8         a[2] = new Aluno("Toledo", 864);
9         Console.WriteLine(a[0].getNome());
10        Console.WriteLine(a[1].getNome());
11        Console.WriteLine(a[2].getNome());
12    }
13 }
14
15 class Aluno
16 {
17     private String nome;
18     private int matric;
19
20     public Aluno (String nome, int matric)
21     {
22         this.nome = nome;
23         this.matric = matric;
24     }
25
26     public String getNome ()
27     {
```

The output window shows the following text:

```
mcs -out:main.exe main.cs
main.cs(18,15): warning CS0414: The private field 'Aluno.m
atric' is assigned but its value is never used
Compilation succeeded - 1 warning(s)
mono main.exe
Bruno
Carla
Toledo
```



Array de Objetos

No rept podemos separar as classes em diferentes arquivos .cs

The screenshot shows a Repl.it IDE interface for a C# project named "EarlyWorthlessRoutine". The left sidebar displays a file explorer with "main.cs", "Aluno.cs", and "main.exe". The main editor area shows the code in "Aluno.cs":

```
1 using System;
2
3 class Aluno
4 {
5     private String nome;
6     private int matric;
7
8     public Aluno (String nome, int matric)
9     {
10         this.nome = nome;
11         this.matric = matric;
12     }
13
14     public String getNome ()
15     {
16         return this.nome;
17     }
18 }
```

The right sidebar shows the console output, which includes the compilation command and the program's execution results:

```
> mcs -out:main.exe Aluno.cs main.cs
Aluno.cs(6,15): warning CS0414: The private field 'Aluno.m
atric' is assigned but its value is never used
Compilation succeeded - 1 warning(s)
> mono main.exe
Bruno
Carla
Toledo
> []
```



Coleção Simples: Classe List

- Representa uma lista fortemente tipada de objetos que podem ser acessados por índice. Fornece métodos para pesquisar, classificar e manipular listas.
- Lista dinâmica com capacidade de armazenar qualquer tipo de objeto.
- Capacidade ilimitada.
- Remoções não deixam “buracos” (tamanho da lista é reajustado automaticamente).



Coleção Simples: Classe List

Repl.it - EarlyWorthlessRoutine x +

repl.it/@RenatoCarioca/EarlyWorthlessRoutine

@RenatoCarioca/EarlyWorthlessRoutine No description

+ new repl

Files

- main.cs
- Aluno.cs
- main.exe

main.cs saved

```
1 using System;
2 using System.Collections.Generic;
3
4 class MainClass {
5     public static void Main (string[] args) {
6         List<Aluno> a = new List <Aluno> ();
7         a.Add(new Aluno("Bruno", 567));
8         a.Add(new Aluno("Carla", 123));
9         a.Add(new Aluno("Toledo", 864));
10        Console.WriteLine(a[0].getNome());
11        Console.WriteLine(a[1].getNome());
12        Console.WriteLine(a[2].getNome());
13        Console.ReadKey();
14    }
15 }
16
17
```

https://EarlyWorthlessRoutine.renatocarioca.r

```
private field `Aluno.m
atic' is assigned but its value is n
ever used
Compilation succeeded - 1 warning(s)
❗ mono main.exe
Bruno
Carla
Toledo
exited, terminated
❗ []
```



Coleção Simples: Classe List

Principais membros:

- `int Count`: propriedade que indica a quantidade de elementos armazenados.
- `void Add(E e)`: insere o objeto `e` no final da lista.
- `void Insert(int index, E e)`: insere o objeto `e` na posição indicada por `index`.
- `void Clear()`: remove todos os elementos da lista.
- `bool Contains(E o)`: indica se a lista contém o objeto `o`.
- `int IndexOf(E o)`: retorna a posição da primeira ocorrência do objeto `o` dentro da lista. Retorna -1 caso o objeto não seja encontrado.
- `void RemoveAt(int index)`: remove o elemento que se encontra na posição `index`.
- `bool Remove(Object o)`: remove a primeira ocorrência do objeto `o` dentro da lista.



Classes System.Collections.Generic

Você pode criar uma coleção genérica usando uma das classes no namespace [System.Collections.Generic](#). Uma coleção genérica é útil quando cada item na coleção tem o mesmo tipo de dados. Uma coleção genérica impõe tipagem forte, permitindo que apenas o tipo de dados desejado seja adicionado.

A tabela a seguir lista algumas das classes frequentemente usadas do namespace [System.Collections.Generic](#):

Classe	Descrição
Dictionary<TKey,TValue>	Representa uma coleção de pares chave-valor organizados com base na chave.
List<T>	Representa uma lista de objetos que podem ser acessados por índice. Fornece métodos para pesquisar, classificar e modificar listas.
Queue<T>	Representa uma coleção de objetos PEPS (primeiro a entrar, primeiro a sair).
SortedList<TKey,TValue>	Representa uma coleção de pares chave/valor que são classificados por chave com base na implementação de IComparer<T> associada.
Stack<T>	Representa uma coleção de objetos UEPS (último a entrar, primeiro a sair).



Classes `System.Collections.Concurrent`

Classes `System.Collections`

As classes no namespace [System.Collections](#) não armazenam elementos como objetos especificamente tipados, mas como objetos do tipo `Object`.

Sempre que possível, você deve usar as coleções genéricas no namespace [System.Collections.Generic](#) ou no [System.Collections.Concurrent](#) em vez dos tipos herdados no namespace `System.Collections`.

A tabela a seguir lista algumas das classes frequentemente usadas no namespace `System.Collections`:

Classe	Descrição
ArrayList	Representa uma matriz de objetos cujo tamanho é aumentado dinamicamente conforme necessário.
Hashtable	Representa uma coleção de pares chave-valor organizados com base no código hash da chave.
Queue	Representa uma coleção de objetos PEPS (primeiro a entrar, primeiro a sair).
Stack	Representa uma coleção de objetos UEPS (último a entrar, primeiro a sair).



Coleção: Classe ArrayList

Repl.it - GroundedSphericalDevel x +

repl.it/@RenatoCarioca/GroundedSphericalDevelopments

@RenatoCarioca/GroundedSphericalDevelopments No description

+ new repl

Files

- main.cs
- Aluno.cs

main.cs saved

```
1 using System;
2 using System.Collections;
3
4 class MainClass {
5     public static void Main (string[] args) {
6         ArrayList a = new ArrayList();
7         a.Add(new Aluno("Bruno", 567));
8         a.Add(new Aluno("Carla", 123));
9         a.Add(new Aluno("Toledo", 864));
10
11         foreach (Aluno aluno in a)
12         {
13             Console.WriteLine(aluno.getNome());
14         }
15     }
16 }
```

https://GroundedSphericalDevelopments.renat

```
mcs -out:main.exe Aluno.cs main.cs
Aluno.cs(7,21): warning CS0414: The private field 'Aluno.matric' is assigned but its value is never used
Compilation succeeded - 1 warning(s)
mono main.exe
Bruno
Carla
Toledo
[]
```