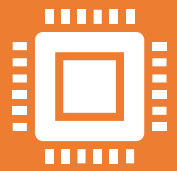
LISTAS

Prof. Hélio Esperidião.

arrays



Se quisermos armazenar muitos dados na memória, podemos fazer o uso de arrays.

Arrays nos possibilitam guardar uma quantidade de elementos e depois acessá-los de forma fácil.

O problema é que manipular um array não é fácil

Problema dos Arrays

Imagine um array com 5 instancias armazenadas.

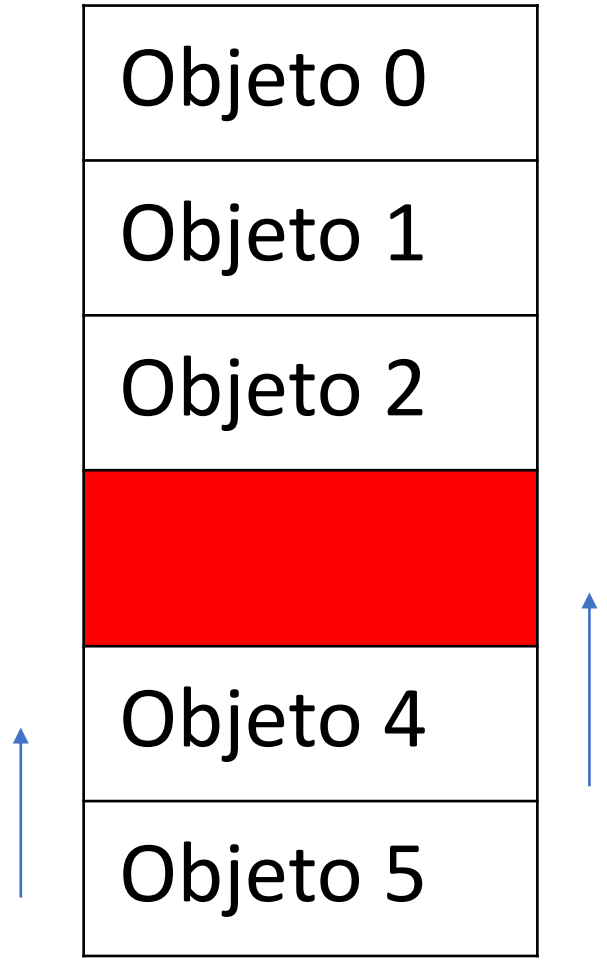
Se quisermos remover a posição 1, como fazemos?

Se apagarmos a posição 1 precisaremos reordenar todo nosso array.

Para inserir um elemento no meio do array?

Precisamos "abrir um buraco" no array, empurrando elementos pra baixo, para aí sim colocar o novo elemento no meio.

Array de objetos



LISTAS

Para resolver os problemas do array, podemos trabalhar com uma classe do C# chamada List .

Para utilizarmos uma lista dentro do código precisamos informar qual é o tipo de elemento que a lista armazenará.

Claro que os arrays possuem funcionalidade impar para resolver alguns problemas

Sintaxe

List<Classe> lista = new List <Classe> ();

Da mesma forma que criamos a lista de objetos de uma determinada Classe.

Também é possível criar uma lista de números inteiros ou de qualquer outro tipo do C#

Trabalhando com listas

```
List<Classe> lista = new List <Classe> ();
Classe c1 = new Classe();
Classe c2 = new Classe();
Classe c3 = new Classe();
lista.Add(c1);
lista.Add(c2);
lista.Add(c3);
```

Exemplo prático

```
List<Classe> lista = new List<Classe>();
Classe c1 = new Classe();
Classe c2 = new Classe();
Classe c3 = new Classe();
```

Recuperar dados e remover dados;

- Podemos acessá-la pela sua posição.
- Classe c20= lista[0];
- lista.RemoveAt(0);

```
Classe c20 = lista[0];
lista.RemoveAt(0);
```

Quantidade de elementos

- Se quisermos saber quantos elementos existem no List , podemos usar a propriedade Count :

```
int tamanho = lista.Count;
```