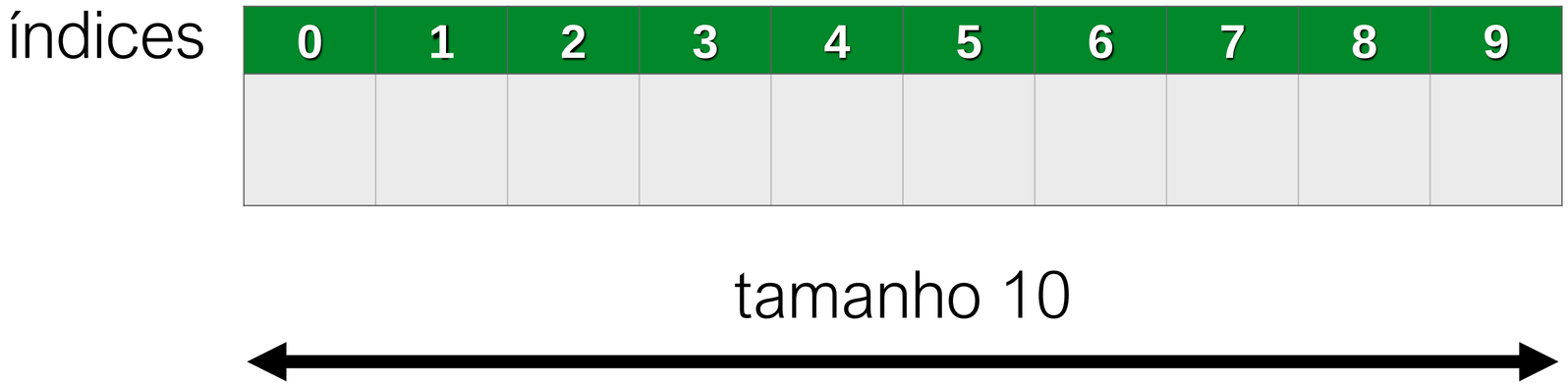
ArrayList

Programação Orientada a Objetos - Aula 07

Prof. Hamilton Machiti da Costa

ArrayList

- O ArrayList é uma classe do pacote java.util que representa uma coleção homogênea de elementos com acesso indexado. Ele não tem limitação de tamanho, crescendo conforme elementos são adicionados.



Construtores

- construtor para instanciar um ArrayList de Objects:

```
ArrayList lista = new ArrayList();
```

- construtor para instanciar um ArrayList de Strings:

```
ArrayList<String> lista = new ArrayList<String>();
```

- construtor para instanciar um ArrayList de double:

```
ArrayList<Double> lista = new ArrayList<Double>();
```

- notem que um ArrayList não pode ser criado com tipos primitivos, somente objetos;

- porém, em um ArrayList de Double você pode adicionar doubles no formato literal - 10.0 - ou variáveis; eles são automaticamente convertidos para objetos Double pela wrapper class Double.

- wrapper classes (classes empacotadoras): existe uma para cada tipo primitivo: Double, Integer, Character, Boolean, as mais usadas, mas também tem Long, Float, Byte e Short. Os nomes são autoexplicativos.

Tamanho do ArrayList

- quando um ArrayList é instanciado seu tamanho é zero; para retornar o tamanho use o método

```
int size();
```

- quando você adiciona um elemento, o tamanho é incrementado; quando remove, é decrementado.

- se o tamanho do ArrayList é zero ele passa a ser 1 depois que é adicionado um elemento;

- se o tamanho do ArrayList é 1 ele passa a ser zero depois que é removido um elemento;

Adição de elementos

- quando um elemento é adicionado, ele é colocado no final do ArrayList e o tamanho é acrescido de uma unidade; o método para se adicionar elementos é o

```
void add(elemento);
```

- o parâmetro elemento deve ser do mesmo tipo do ArrayList; se for um ArrayList<String> você só pode adicionar String, por exemplo; se o tipo não for definido, o ArrayList é de Object e qualquer objeto pode ser adicionado (neste caso não são aceitos tipos primitivos);

Remoção de Elementos

- para se remover um elemento usa-se o método

```
remove(int posicao);
```

- a posição deve ser >=0 e < size(), caso contrário é gerada a exceção ArrayIndexOutOfBoundsException; quando você remove um elemento o tamanho do ArrayList é reduzido de um; além disso, se você remove um elemento que está antes de outros elementos, o índice dos demais é recalculado; por exemplo, se você remove o elemento de índice 2 de um ArrayList com 5 elementos, o de índice 3 passa a ser 2 e o de índice 4 passa a ser 3; neste caso, o tamanho passa de 5 para 4.

Obtenção de um elemento

- para se obter um elemento de uma determinada posição do ArrayList usa-se o método

```
tipo get(int posicao)
```

- onde tipo é o tipo do ArrayList; se for <String> tipo é String; se for Object, o tipo é Object.

- note que este método não remove o elemento, mas apenas pega uma referência a ele.

Percorrendo um ArrayList com um for

```
ArrayList<String> lista = new ArrayList<String>();
lista.add("palavra 1");
lista.add("palavra 2");
String s;
for(int i = 0; i < lista.size(); i++){
    s = lista.get(i);
    System.out.println(s);
}
```

Percorrendo com um for-each

```
ArrayList<String> lista = new ArrayList<String>();
lista.add("palavra 1");
lista.add("palavra 2");
for(String s:lista){
    System.out.println(s);
}
```

For-each

- O resultado de ambos os loops é imprimir

palavra 1

palavra 2

- O for-each é um loop simplificado para coleções. Ele percorre a coleção do início ao fim. A única forma de interrompê-lo antecipadamente é usando um **break**.

- Nota: uma coleção não pode ser aumentada ou reduzida durante um for-each. Isto causa uma exception.

O método abaixo conta quando elementos do arraylist 1 também estão no arraylist2. Ele assume que não há elementos repetidos. Por isso, quando encontra um, já encerra o loop interior com um **break**.

```
3     public int contaIguais(ArrayList<Integer> lista1,
4         ArrayList<Integer> lista2){
5
6         int count = 0;
7         for(int elemento1: lista1){
8             for(int elemento2: lista2){
9                 if(elemento1 == elemento2){
10                    count++;
11                    break;
12                }
13            }
14        }
15        return count;
16    }
```