Coleções

Orientação a Objetos – DCC025

Gleiph Ghiotto Lima de Menezes
gleiph.ghiotto@ufjf.br

Aula de hoje

- Estudaremos algumas das coleções disponíveis no Java
  - Lista
  - Conjunto
  - Dicionário

Exercício Motivacional

- Até agora vimos *arrays* como variáveis compostas homogêneas (mesmo tipo)
- Os *arrays* têm tamanho fixo, o que dificulta o tratamento de situações onde o número de elementos muda com o passar do tempo
- Como vocês fariam para criar uma classe ArrayDinamico que tivesse os seguintes métodos:
  - void add(int index, Object element)
  - void remove(int index)
  - int size()
  - Object get(int index)
  - ...

Coleções

- O **pacote java.util.*** define diversas estruturas de dados
- As estruturas implementam interfaces padrões:
  - Lista: List
  - Conjunto: Set
  - Dicionário: Map
- Cada interface tem uma implementação padrão (usualmente utilizada pelos programadores)
  - Lista: ArrayList
  - Conjunto: HashSet
  - Dicionário: HashMap

List

- A interface List (e a sua implementação padrão ArrayList) permite a criação de arrays dinâmicos
  - A lista pode conter qualquer tipo de objeto Java, em qualquer quantidade
  - Os elementos podem ser acessados em qualquer ordem
- Declarando um List e instanciando um ArrayList:

```
List<Pessoa> pessoas = new ArrayList<>()
```

Tipo que será guardado na lista

List

- Principais métodos:
  - **add(elemento)**: adiciona elemento no final da lista
  - **add(posição, elemento)**: adiciona elemento em uma posição da lista
  - **remove(elemento)**: remove um elemento da lista
  - **remove(posição)**: remove o elemento que está em uma posição da lista
  - **clear()**: remove todos os elementos da lista
  - **get(posição)**: retorna o elemento em uma posição da lista
  - **indexOf(elemento)**: retorna a posição de um elemento da lista
  - **isEmpty()**: informa se a lista está vazia
  - **size()**: informa o número de elementos da lista
- Ver demais métodos em http://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html

Exemplo

```
List<Pessoa> pessoas = new ArrayList<>();

pessoas.add(new Pessoa("João", 34));
pessoas.add(new Pessoa("Pedro", 14));
pessoas.add(new Pessoa("Paulo", 54));

for (Pessoa pessoa : pessoas) {
    System.out.print(pessoa.getNome() + " tem " +
                     pessoa.getIdade() + " anos.");
}

pessoas.clear();
```

Exemplo

```
for (Pessoa pessoa : pessoas) {
    System.out.print(pessoa.getNome() + " tem " +
                     pessoa.getIdade() + " anos.");
}
```

é o mesmo que

```
for (int i = 0; i < pessoas.size(); i++) {
    Pessoa pessoa = pessoas.get(i);
    System.out.print(pessoa.getNome() + " tem " +
                     pessoa.getIdade() + " anos.");
}
```

Exercício

- Faça um programa que escreva a frase invertida (da última palavra para a primeira)
  - Use List para fazer a inversão

Set

- A interface Set (e a sua implementação padrão HashSet) permite a criação de conjuntos dinâmicos
  - Equivalente a lista, porém não impõe ordem aos elementos e não permite duplicata
- Declarando um Set e instanciando um HashSet:

```
Set<String> palavras = new HashSet<>()
```

Tipo que será guardado no conjunto

Set

- Principais métodos:
  - **add(elemento)**: adiciona elemento no conjunto
  - **remove(elemento)**: remove um elemento do conjunto
  - **clear()**: remove todos os elementos da lista
  - **contains(elemento)**: informa se o elemento está no conjunto
  - **isEmpty()**: informa se o conjunto está vazio
  - **size()**: informa o número de elementos do conjunto
- Ver demais métodos em http://docs.oracle.com/javase/8/docs/api/java/util/Set.html

Exemplo

```
Set<String> palavras = new HashSet<>();

palavras.add("Flamengo");
palavras.add("Fluminense");
palavras.add("Botafogo");
palavras.add("Botafogo");

System.out.println(palavras.size());

for (String palavra : palavras) {
    System.out.println(palavra);
}
```

O que é mostrado aqui?

E aqui? Em qual ordem?

Exercício

- Faça um programa que leia uma frase e informe o número de palavras não repetidas da frase
  - Use Set para fazer essa verificação

Map

- A interface Map (e a sua implementação padrão HashMap) permite a criação de dicionários dinâmicos
  - Um dicionário associa um objeto chave a um objeto valor (key → value)
- Declarando um Map e instanciando um HashMap:

```
Map<String,String> dddPorMunicipio = new HashMap<>()
```

Tipo que será valor do dicionário

Tipo que será chave do dicionário

Map

- Principais métodos:
  - **put(chave, valor)**: adiciona uma chave indexando um valor no dicionário
  - **get(chave):** retorna o valor indexado pela chave
  - **getOrDefault(chave, valor):** retorna o valor indexado pela chave ou o valor default informado
  - **keySet()**: retorna um conjunto com todas as chaves do dicionário
  - **remove(chave)**: remove o valor indexado pela chave no dicionário
  - **clear()**: remove todas as entradas do dicionário
  - **isEmpty()**: informa se o dicionário está vazio
  - **size()**: informa o número de entradas do dicionário
- Ver demais métodos em http://docs.oracle.com/javase/8/docs/api/java/util/Map.html

Exemplo

```
Map<String,String> dddPorMunicipio = new HashMap<>();

dddPorMunicipio.put("São Paulo", "11");
dddPorMunicipio.put("Rio de Janeiro", "21");
dddPorMunicipio.put("Belo Horizonte", "31");

for (String municipio : dddPorMunicipio.keySet()) {
    System.out.println("O DDD de " + municipio + " é " +
                       dddPorMunicipio.get(municipio));
}
```

Exercício

- Faça um programa que leia uma frase e informe o número de ocorrências de cada palavra da frase
  - Use Map para fazer essa contagem
- Dica: Java tem uma Classe para cada tipo primitivo, e faz a tradução automática entre ambos
  - Classe Integer para tipo int
  - Classe Double para tipo double
  - Classe Boolean para tipo boolean
  - Classe Character para tipo char
  - Etc.

Coleções

Orientação a Objetos – DCC025

Gleiph Ghiotto Lima de Menezes
gleiph@ice.ufjf.br

Departamento de Ciência da Computação
UFJF