Programação Avançada - AULA 06

Matheus Moresco

Análise e Desenvolvimento de Sistemas - 5º Período

2025/01

Últimas aulas

- Revisão de POO
- Encapsulamento
- Herança
- Métodos construtores e Sobrecarga
- Classes Abstratas e Interfaces

Aula de Hoje

- Entender o conceito de **arrays e listas** em Java.
- Diferenciar **arrays estáticos** e **listas dinâmicas**.
- Aprender a manipular arrays e listas com métodos úteis.
- Aplicar arrays e listas em **exemplos práticos**.

Estruturas de Dados em Java

- Java possui duas estruturas principais para armazenar múltiplos valores:
  - **Array ->** int[] numeros = new int[5];
  - **List** -> List<Integer> numeros = new ArrayList<>();

Diferenças entre Array e Listas

- **Array**:
  - Tamanho fixo após a criação.
  - Pode armazenar qualquer tipo de dado primitivo ou objeto.
  - Rápido no acesso direto aos elementos.
  - Não possui métodos embutidos para adicionar ou remover elementos.

- **Lista**:
  - Tamanho dinâmico (pode crescer ou diminuir).
  - Armazena objetos (não suporta tipos primitivos diretamente, mas podemos usar Integer, Double, etc.).
  - Possui métodos embutidos como add(), remove(), get().
  - Mais flexível, mas pode ser um pouco mais lento que arrays em algumas operações.

Quando Usar Arrays ou Listas?

- Use Arrays quando:
  - O número de elementos é conhecido e fixo.
  - Precisa de alta performance para acessar elementos.
  - Está trabalhando com dados primitivos (int, double, etc.).

- Use Listas quando:
  - Precisa adicionar ou remover elementos com frequência.
  - Não sabe o tamanho exato dos dados.
  - Quer métodos prontos para buscar, remover e ordenar elementos.

Arrays - Declaração e Inicialização

- Inicializar um array vazio:
  1. Definir o tipo os valores que serão inseridos no array;
  2. Atribuir um nome da variável;
  3. Usar o new para criar uma nova estrutura;
  4. Definir o tamanho do array.
- Iniciar o array já preenchido:
  1. Definir o tipo dos elementos e atribuir os valores diretamente.

```
// Declarando um array de inteiros com tamanho 5
int[] numeros = new int[5];

// Inicializando um array diretamente
int[] valores = {10, 20, 30, 40, 50};
```

Arrays - Acessando Elementos

- Para acessar os valores armazenados nos elementos de um array, usamos os colchetes “[ ]”, para acessar o valor de um elemento em uma determinada posição.

```
// Inicializando um array diretamente
int[] valores = {10, 20, 30, 40, 50};

System.out.println(valores[2]); // Saída: 30
```

Arrays – Percorrendo o Array

- Usando for tradicional:

```
for (int i = 0; i < valores.length; i++) {
    System.out.println(valores[i]);
}
```

- Usando for-each (mais simples):

```
for (int valor : valores) {
    System.out.println(valor);
}
```

Arrays - Modificando um Array

- A modificação de elementos no array é feita acessando o elemento pela sua posição e atribuindo um novo valor a ele.

```
valores[1] = 25; // Altera o segundo elemento
```

ArrayList - Importação e Declaração

- Para inicializarmos uma lista em Java, precisamos:
  1. Importar os objetos **ArrayList** e **List** da biblioteca ‘java.util’;
  2. Criar um objeto do tipo **List**, passando como parâmetro o tipo do objeto que será salvo na lista.
  3. Atribuir um nome a variável;
  4. Usando o ‘new’, construir um novo objeto **ArrayList**;

```
import java.util.ArrayList;
import java.util.List;

List<String> nomes = new ArrayList<>();
```

ArrayList - Adicionando Elementos

- Para adicionar elementos ao ArrayList, usamos o método de add() presente no objeto.
- Neste método podemos passar novos objetos, desde que sejam do mesmo tipo definido na declaração do ArrayList.

```
nomes.add("Ana");
nomes.add("Bruno");
nomes.add("Carlos");
```

ArrayList - Acessando Elementos

- Para acessar elementos no ArrayList, usamos o método de get() e passamos como parâmetro o índice do elemento que queremos acessar.

```
System.out.println(nomes.get(1)); // Saída: Bruno
```

ArrayList - Percorrendo uma Lista

- Usando for tradicional:

```
for (int i = 0; i < nomes.size(); i++) {
    System.out.println(nomes.get(i));
}
```

- Usando for-each (mais simples):

```
for (String nome : nomes) {
    System.out.println(nome);
}
```

ArrayList - Modificando e Removendo Elementos

- Para modificar elementos em um ArrayList, usamos o método set(), passando como parâmetro a posição do elemento a ser alterado e o novo valor do elemento.
- Para a deleção usamos o método remove() e passamos como parâmetro a posição do elemento a ser deletado.

```
nomes.set(1, "Beatriz");  // Modifica o segundo elemento
nomes.remove(0);          // Remove o primeiro elemento
```

Comparação entre Arrays e Listas

Característica	Arrays (int[])	Listas (ArrayList<Integer>)
Tamanho	Fixo	Dinâmico
Velocidade	Mais rápido para acesso direto	Mais flexível
Adicionar/Remover	Difícil (precisa criar novo array)	Fácil (add(), remove())
Uso recomendado	Quando o tamanho é conhecido e fixo	Quando o tamanho varia

Exemplo prático

Usar a classe Pessoa

1. Criar um atributo de telefones a pessoa, onde ela pode cadastrar até 3 telefones.
2. Criar um atributo de veículos a pessoa, onde ela pode cadastrar quantos veículos ela quiser.

Exercício

1. Crie uma classe Empresa
2. Crie os seguinte atributos na empresa:
    1. Nome;
    2. CNPJ;
    3. E-mails: Crie um array que permite o cadastro de até 5 e-mails;
    4. Funcionários: uma lista com os funcionários da empresa (use a classe Funcionário, criada anteriormente)
    5. Crie os métodos para cadastro, remoção e exibição de funcionários e e-mail