07

Desafio opcional: Trabalhando com string

Nessa aula, implementamos o algoritmo de busca binária, outro algoritmo de busca, mais eficiente que a busca linear. O desafio agora é aplicar o mesmo algoritmo para buscar *strings*, usando o *array* de nomes, que usamos nos últimos desafios, para testar o algoritmo.

Opinião do instrutor

Para buscar *strings*, podemos usar o mesmo método, fazendo apenas algumas alterações. Na assinatura do método, vamos alterar o tipo dos parâmetros `notas e busca`.

```
private static int busca(String[] nomes, int de, int ate,
  // Algoritmo de busca binária
}
```

COPIAR CÓDIGO

O parâmetro `notas`, do tipo `Nota[]`, foi substituído pelo parâmetro `nomes`, de tipo `String[]`, e alteramos o tipo do parâmetro `busca`, de `double` para `String`. Da mesma forma que alteramos o tipo do *array* recebido, vamos alterar o tipo da variável `nota` para `String`, e naturalmente seu nome também será alterado.

```
String nome = nomes[meio];
```

COPIAR CÓDIGO

Por fim, vamos alterar a condição dos *ifs*, usando o método `compareTo`, da classe `String`, para comparar os textos.

```
if(busca.compareTo(nome) == 0) {
  return meio;
}
```

COPIAR CÓDIGO

Se o método `compareTo` retornar `0`, as *strings* são iguais e podemos retornar a posição do elemento.

```
if(busca.compareTo(nome) < 0) {
  return busca(nomes, de, meio - 1, busca);
}
```

COPIAR CÓDIGO

Mas, se retornar um valor negativo, a `busca` é "menor" que o elemento do meio e chamamos o método `busca` novamente. O resto do método continua igual. Ao final das alterações teremos o seguinte código...

```
private static int busca(String[] nomes, int de, int ate,

  if(de > ate) {
    return -1;
  }
  int meio = (de + ate) / 2;
  String nome = nomes[meio];
  if(busca.compareTo(nome) == 0) {
    return meio;
  }
  if(busca.compareTo(nome) < 0) {
    return busca(nomes, de, meio - 1, busca);
  }
```

```
    return busca(nomes, meio + 1, ate, busca);
  }
```

COPIAR CÓDIGO

Ainda podemos criar um outro método `busca`, que chamaremos quando quisermos procurar no *array* inteiro.

```
  private static int busca(String[] nomes, String busca) {
    return busca(nomes, 0, nomes.length - 1, busca);
  }
```

COPIAR CÓDIGO

Implementamos o seguinte código no método `main`, para testar o algoritmo.

```
  String busca = "Paulo";
  int resultado = busca(nomes, busca);
  if(resultado < 0) {
    System.out.println("Não encontrei " + busca + " no arra
  } else {
    System.out.println("Encontrei " + busca + " no índice "
  }
```

COPIAR CÓDIGO

Obtivemos o seguinte resultado.

```
  Encontrei Paulo no índice 9
```

COPIAR CÓDIGO

Mas, se alterarmos a busca para um nome que não está na lista...

```
  String busca = "Felipe";
```

COPIAR CÓDIGO

Temos o seguinte resultado...

```
Não encontrei Felipe no array
```

COPIAR CÓDIGO

Logo, podemos concluir que o método funciona.