



PRO_MOV by Reskilling4Employment Software Developer

PCE - Programação de computadores - estruturada

Bruno Santos

bruno.santos@cesae.pt

Arrays

- Arrays ou vetores são variáveis que ao permitem guardar conjuntos de valores do mesmo tipo.

```
int[] idades = new int[]{10, 15, 8, 7};  
  
String[] meses = new String[12];  
meses[0] = "Janeiro";  
meses[1] = "Fevereiro";  
//...  
meses[11] = "Dezembro";
```

Arrays monodimensionais

- Nos arrays monodimensionais é possível guardar dados como se de uma linha de uma tabela, por exemplo, notas de testes de um aluno.
- Exemplo: array com 1 “linha” e 10 “colunas”

```
int[] vetor = new int[10];
```

Arrays multidimensionais

- Nos arrays multidimensionais é possível guardar dados como se o array funcionasse como uma tabela tendo múltiplas linhas e colunas.
- Exemplo: array com 10 “colunas” e cada “coluna” com 10 “linhas”.

```
int[][] matriz = new int[10][10];
```

Listas

- O ArrayList é uma classe da biblioteca Java que faz parte do pacote `java.util`. É uma implementação da interface `List`, que representa uma coleção dinâmica baseada em arrays, capaz de armazenar elementos e redimensionar-se automaticamente quando necessário.
- Diferentemente dos arrays fixos (`int[]`, `String[]`), o ArrayList pode crescer e encolher à medida que elementos são adicionados ou removidos.

Listas

- **Dinâmico:** redimensiona-se automaticamente quando necessário.
- **Ordenado:** mantém a ordem de inserção dos elementos.
- **Aceita duplicados**
- **Não é thread-safe:** se for usado em múltiplas threads, precisa de sincronização externa.
- **Baseado em índices:** permite acesso rápido aos elementos com base no índice.

Listas

```
// ArrayList para String  
ArrayList<String> nomes = new ArrayList<>();  
  
// ArrayList para int  
ArrayList<Integer> numeros = new ArrayList<>();
```

Listas

```
ArrayList<String> nomes = new ArrayList<>();  
nomes.add("Jose");  
nomes.add("Maria");  
nomes.add("Ana");  
nomes.add("Luis");  
  
for(String nome : nomes) {  
    System.out.println(nome);  
}
```

```
/Users/brunosantos/Library/  
Jose  
Maria  
Ana  
Luis
```


Listas

```
ArrayList<String> nomes = new ArrayList<>();  
nomes.add("Jose");  
nomes.add("Maria");  
nomes.add("Ana");  
nomes.add("Luis");  
  
for(String nome : nomes) {  
    System.out.println(nome);  
}
```

```
/Users/brunosantos/Library/  
Jose  
Maria  
Ana  
Luis
```

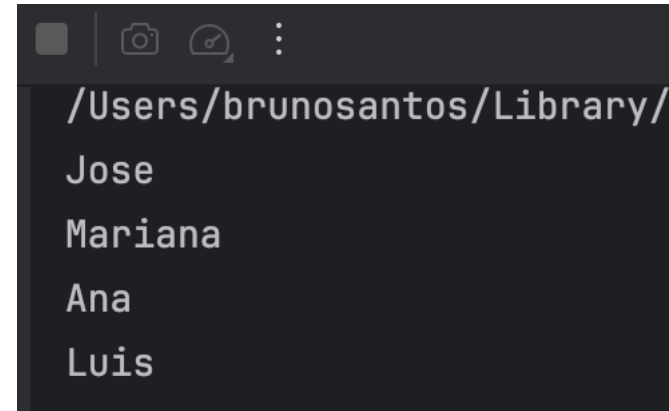
Listas

```
ArrayList<String> nomes = new ArrayList<>();  
nomes.add("Jose");  
nomes.add("Maria");  
nomes.add("Ana");  
nomes.add("Luis");  
  
System.out.println(nomes.get(2));
```

```
/Users/brunosantos/Library/  
Ana  
  
Process finished with exit
```

Listas

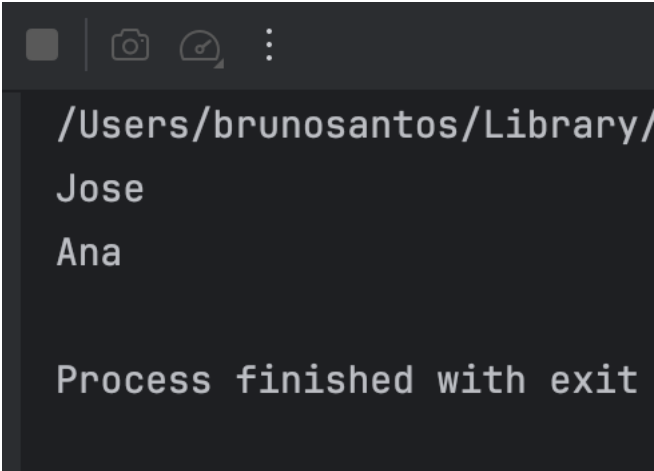
```
ArrayList<String> nomes = new ArrayList<>();  
nomes.add("Jose");  
nomes.add("Maria");  
nomes.add("Ana");  
nomes.add("Luis");  
  
nomes.set(1, "Mariana");  
  
for (String nome : nomes) {  
    System.out.println(nome);  
}
```



```
/Users/brunosantos/Library/  
Jose  
Mariana  
Ana  
Luis
```

Listas

```
ArrayList<String> nomes = new ArrayList<>();  
nomes.add("Jose");  
nomes.add("Maria");  
nomes.add("Ana");  
nomes.add("Luis");  
  
nomes.remove(index: 1);  
nomes.remove(o: "Luis");  
  
for (String nome : nomes) {  
    System.out.println(nome);  
}
```



```
/Users/brunosantos/Library/  
Jose  
Ana  
  
Process finished with exit
```

Listas

```
ArrayList<String> nomes = new ArrayList<>();  
nomes.add("Jose");  
nomes.add("Maria");  
nomes.add("Ana");  
nomes.add("Luis");  
  
System.out.println("Posição da Maria: " + nomes.indexOf("Maria"));
```

```
/Users/brunosantos/Library/Java/Ja  
Posição da Maria: 1
```

```
Process finished with exit code 0
```

Listas

```
ArrayList<String> nomes = new ArrayList<>();  
nomes.add("Jose");  
nomes.add("Maria");  
nomes.add("Ana");  
nomes.add("Luis");  
  
System.out.println("Nº elementos da lista: " + nomes.size());
```

```
/Users/brunosantos/Library/Jav  
Nº elementos da lista: 4  
  
Process finished with exit cod
```

Listas

```
ArrayList<String> nomes = new ArrayList<>();  
nomes.add("Jose");  
nomes.add("Maria");  
nomes.add("Ana");  
nomes.add("Luis");  
  
System.out.println("Ana existe na lista? " + nomes.contains("Ana"));
```

```
/Users/brunosantos/Library/Java/Java  
Ana existe na lista? true  
  
Process finished with exit code 0
```

Listas

```
ArrayList<String> nomes = new ArrayList<>();  
nomes.add("Jose");  
nomes.add("Maria");  
nomes.add("Ana");  
nomes.add("Luis");  
  
System.out.println("Nº elementos na lista: " + nomes.size());  
  
nomes.clear();  
  
System.out.println("Nº elementos na lista: " + nomes.size());
```

```
/Users/brunosantos/Library/Java/Java  
Nº elementos na lista: 4  
Nº elementos na lista: 0  
  
Process finished with exit code 0
```


Tratamento de exceções

- Uma exceção é um evento anormal que interrompe o fluxo normal de execução de um programa.
- Todas as exceções em Java derivam da classe ***Throwable***. Existem dois tipos principais:
 - **Checked Exceptions:** devem ser tratadas ou declaradas no método usando throws. Exemplo: IOException, SQLException.
 - **Unchecked Exceptions:** são subclasses de RuntimeException e geralmente resultam de erros do programador. Exemplo: NullPointerException, ArrayIndexOutOfBoundsException.

Tratamento de exceções

```
import java.util.InputMismatchException;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);

        try {
            System.out.print("Insira um número inteiro: ");
            int number = in.nextInt();
            System.out.println("Inseriu o número: " + number);
        } catch (InputMismatchException e) {
            System.out.println("Erro: Valor inválido. Por favor, insira um número inteiro.");
        } finally {
            in.close();
            System.out.println("Programa encerrado.");
        }
    }
}
```

Tratamento de exceções

```
Insira um número inteiro: aaa  
Erro: Valor inválido. Por favor, insira um número inteiro.  
Programa encerrado.  
  
Process finished with exit code 0
```

Tratamento de exceções

```
public class Main {  
    public static void main(String[] args) {  
  
        int[] numeros = {1, 2, 3, 4, 5};  
  
        try {  
            System.out.println("Valor na posição 10: " + numeros[10]);  
        } catch (ArrayIndexOutOfBoundsException e) {  
            System.out.println("Erro: posição inválida do array.");  
        } finally {  
            System.out.println("Execução do bloco finally.");  
        }  
    }  
}
```

Tratamento de exceções

```
Erro: posição inválida do array.  
Execução do bloco finally.
```

```
Process finished with exit code 0
```

Tratamento de exceções

- **Usar exceções apenas para erros excepcionais:** Evitar tratar condições normais como exceções.
- **Informação clara:** As mensagens devem ajudar o programador ou utilizador a entender o erro.
- **Evitar capturar exceções genéricas (Exception) desnecessariamente:** Sempre que possível, capturar exceções específicas.
- **Bloco finally:** Usar finally para libertar recursos como streams ou ligações a bases de dados.

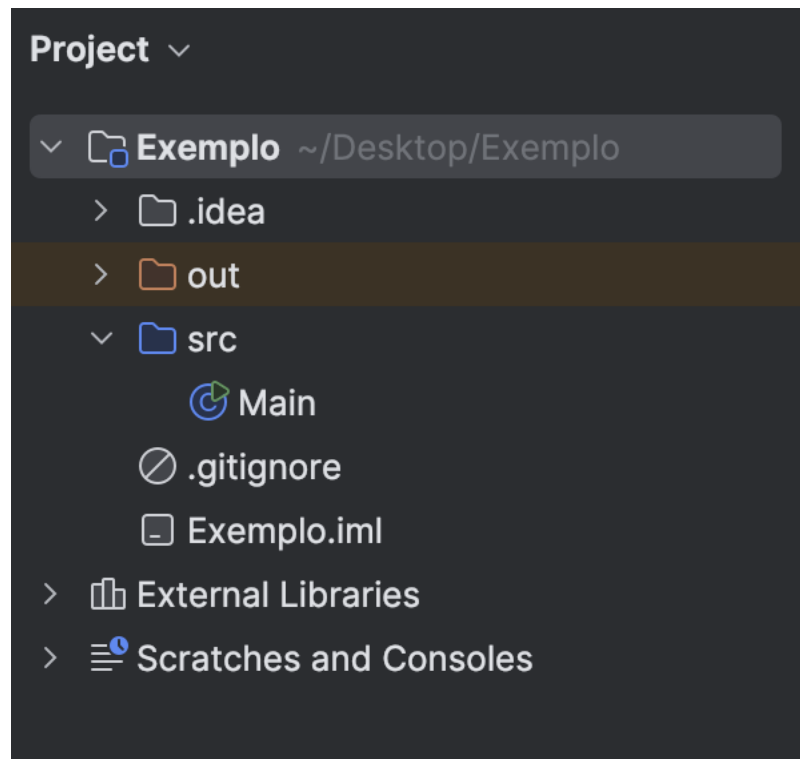
Ficheiros

- A leitura e escrita de ficheiros em Java é feita principalmente através de classes disponíveis no pacote `java.io`

Ficheiros – Leitura

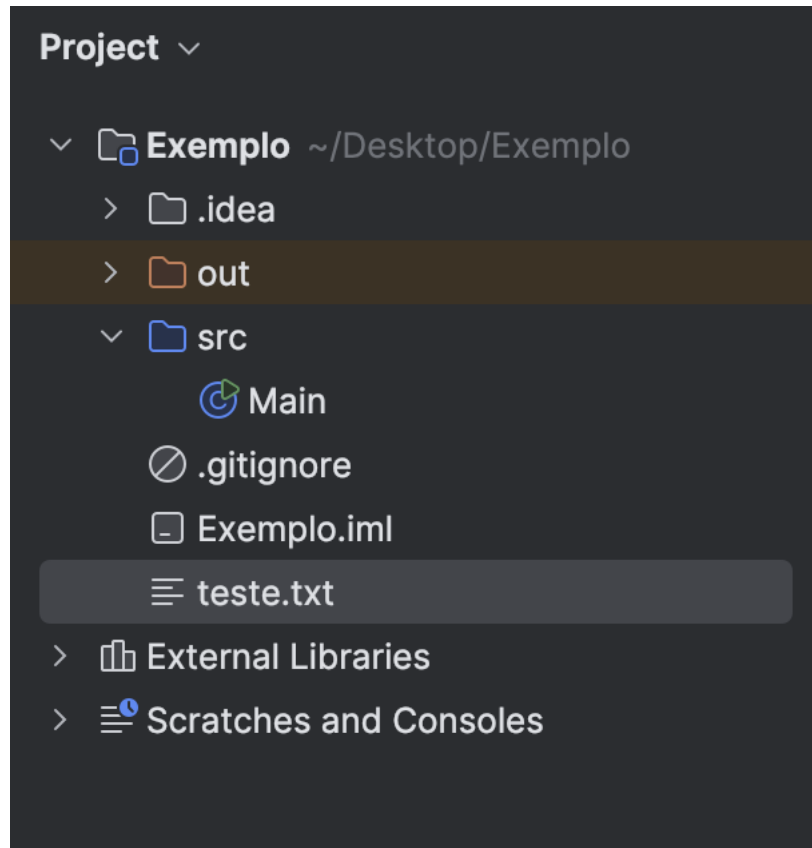
```
public class Main {  
    public static void main(String[] args) {  
  
        try (BufferedReader reader = new BufferedReader(new FileReader(fileName: "teste.txt"))) {  
            String line;  
            while ((line = reader.readLine()) != null) {  
                System.out.println(line);  
            }  
        } catch (IOException e) {  
            System.out.println("Erro ao ler o ficheiro: " + e.getMessage());  
        }  
    }  
}
```


Ficheiros – Leitura



```
/Users/brunosantos/Library/Java/JavaVirtualMachines/openjdk-22.  
Erro ao ler o ficheiro: teste.txt (No such file or directory)  
  
Process finished with exit code 0
```

Ficheiros – Leitura

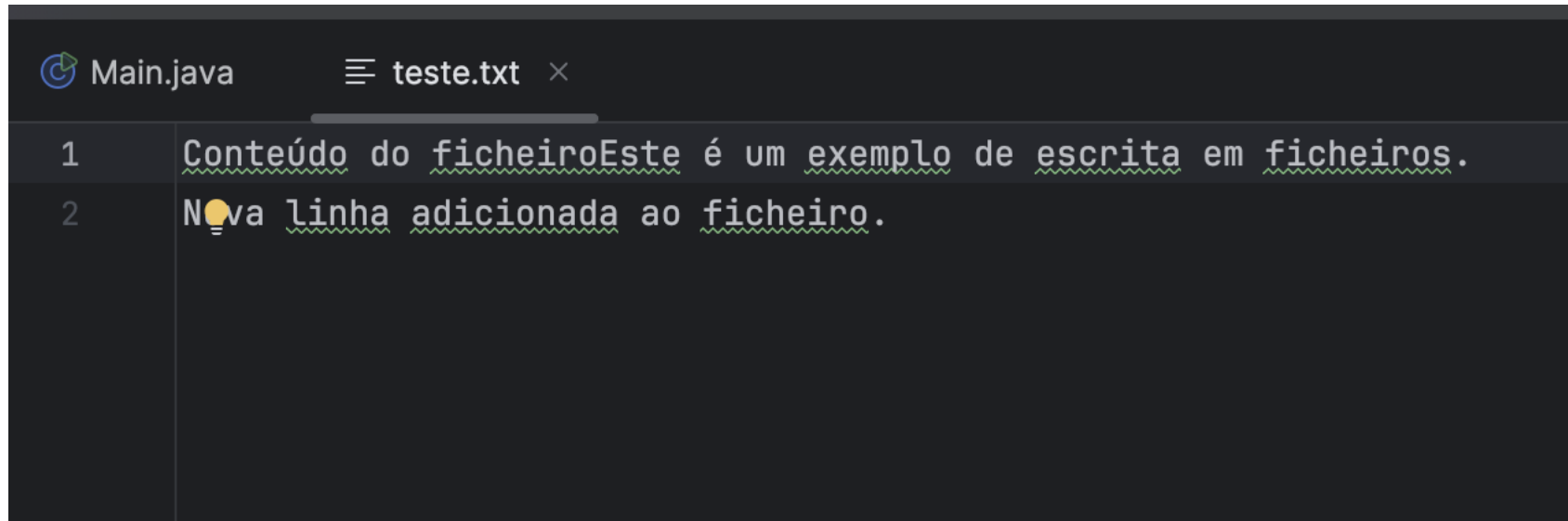


```
/Users/brunosantos/Library/Java/JavaVirtualMach  
Conteúdo do ficheiro  
  
Process finished with exit code 0
```

Ficheiros – Escrita

```
public class Main {  
    public static void main(String[] args) {  
  
        try (BufferedWriter writer = new BufferedWriter(new FileWriter( fileName: "teste.txt", append: true))) {  
            writer.write(str: "Este é um exemplo de escrita em ficheiros.");  
            writer.newLine();  
            writer.write(str: "Nova linha adicionada ao ficheiro.");  
        } catch (IOException e) {  
            System.out.println("Erro ao escrever no ficheiro: " + e.getMessage());  
        }  
    }  
}
```

Ficheiros – Escrita



```

Main.java  teste.txt x
1  Conteúdo do ficheiroEste é um exemplo de escrita em ficheiros.
2  Nova linha adicionada ao ficheiro.

```

The image shows a code editor window with two tabs: 'Main.java' and 'teste.txt'. The 'teste.txt' tab is active. It contains two lines of text in Portuguese. The first line is 'Conteúdo do ficheiroEste é um exemplo de escrita em ficheiros.' and the second line is 'Nova linha adicionada ao ficheiro.'.