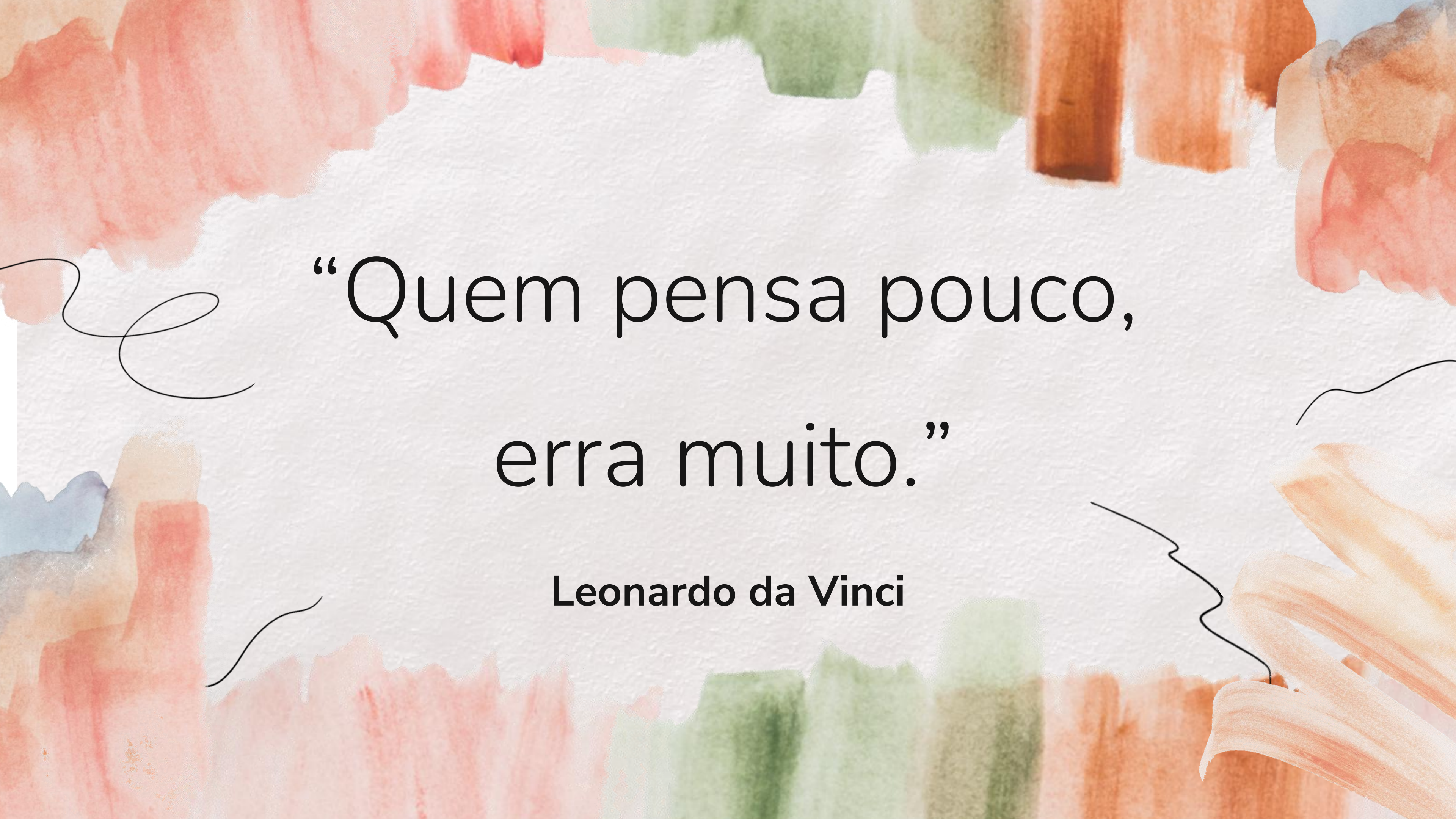




TRATAMENTO DE EXCEÇÕES

Prof^a Lucília Ribeiro



“Quem pensa pouco,
erra muito.”

Leonardo da Vinci

Exceções

- Erros que acontecem durante a **execução** de um programa (exceção != interrupção)
- Falhas de memória, impossibilidade de acesso a um arquivo, atribuição de valor impossível a um objeto etc
- Problema piora quando a aplicação usa bibliotecas de classes criadas por outros usuários. Como estabelecer uma comunicação confiável?

Exceções

```
public class Excl1a {  
    public static void main(String arg[ ]) {  
        int n = 9000000000;  
        int[] vetor = new int[n];  
        for(int i = 0; i <= n; i++) {  
            vetor[i] = i;  
            System.out.println(i);  
        }  
    }  
}
```

Exceções

```
public class Exc1 {  
    public static void main(String arg[ ]) {  
        int[] vetor = new int[10];  
        for(int i = 0; i <= 15; i++) {  
            vetor[i] = i;  
            System.out.println(i);  
        }  
    }  
}
```

```
C:\Users\lucil\Desktop\codigos\P00\Excecao>javac Exc1.java
```

```
C:\Users\lucil\Desktop\codigos\P00\Excecao>java Exc1
```

```
0  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 10 out of bounds for length 10  
at Exc1.main(Exc1.java:5)
```

Para piorar...

```
1  class TesteExc {
2      public static void main(String[] args) {
3          System.out.println("Inicio do main");
4          metodo1();
5          System.out.println("Fim do main");
6      }
7      public static void metodo1() {
8          System.out.println("Inicio do metodo1");
9          metodo2();
10         System.out.println("Fim do metodo1");
11     }
12     public static void metodo2() {
13         System.out.println("Inicio do metodo2");
14         int[] vetor = new int[10];
15         for(int i = 0; i <= 15; i++) {
16             vetor[i] = i;
17             System.out.println(i);
18         }
19         System.out.println("Fim do metodo2");
20     }
21 }
```


Rastro da Pilha (stacktrace)

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException  
    at TesteExc.metodo2(TesteExc.java:16)  
    at TesteExc.metodo1(TesteExc.java:9)  
    at TesteExc.main(TesteExc.java:4)
```

- Exceção lançada (**thrown**) → JVM verifica se o método atual toma alguma precaução ao tentar (**try**) executar esse trecho de código
- **metodo2** não está tratando o problema, verifica se **metodo1** está se precavendo de um problema chamado **ArrayIndexOutOfBoundsException**

Exceções

Mecanismo Potente para Detectar Erros
em Tempo de Execução (RunTime)

try

catch

throw


```
public static void metodo2() {  
    System.out.println("Inicio do metodo2");  
    int[] vetor = new int[10];  
    try {  
        for(int i = 0; i <= 15; i++) {  
            vetor[i] = i;  
            System.out.println(i);  
        }  
    } catch (ArrayIndexOutOfBoundsException erro) {  
        System.out.println("Erro: " + erro);  
    }  
    System.out.println("Fim do metodo2");  
}
```

```
Inicio do main  
Inicio do metodo1  
Inicio do metodo2  
0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
Erro: java.lang.ArrayIndexOutOfBoundsException  
Fim do metodo2  
Fim do metodo1  
Fim do main
```


Experimentos

- Em vez de fazer o `try` em torno do `for` inteiro, faça apenas com o bloco de dentro do `for`. Qual a diferença?

```
for(int i = 0; i <= 15; i++) {  
    try {  
        vetor[i] = i;  
        System.out.println(i);  
    } catch (ArrayIndexOutOfBoundsException erro) {  
        System.out.println("Erro: " + erro);  
    }  
}  
System.out.println("Fim do metodo2");
```

- Agora retire o `try/catch` do `método2` e coloque-o em volta da chamada ao `metodo2` no `metodo1`

```
public static void metodo1() {  
    System.out.println("Inicio do metodo1");  
    try {  
        metodo2();  
    } catch (ArrayIndexOutOfBoundsException erro) {  
        System.out.println("Erro: ");  
    }  
    System.out.println("Fim do metodo1");  
}
```

- Faça a mesma coisa, retirando o `try/catch` do `método1` e coloque-o em volta da chamada ao `metodo1` no `main`

```
public static void main(String[] args) {  
    System.out.println("Inicio do main");  
    try {  
        metodo1();  
    } catch (ArrayIndexOutOfBoundsException erro) {  
        System.out.println("Erro: ");  
    }  
    System.out.println("Fim do main");  
}
```


Checked Exceptions

```
class TesteExc3 {  
    public static void metodo() {  
        new java.io.FileReader("arquivo.txt");  
    }  
}
```

```
TesteExc3.java:3: error: unreported exception FileNotFoundException; must be caught or declared to be thrown  
    new java.io.FileReader("arquivo.txt");  
        ^
```

1 error



Soluções

```
class TesteExc3 {  
    public static void metodo() {  
        try {  
            new java.io.FileReader("arquivo.txt");  
        } catch (java.io.FileNotFoundException erro) {  
            System.out.println("Erro ao abrir o arquivo");  
        }  
    }  
}
```

```
class TesteExc3 {  
    public static void metodo() throws java.io.FileNotFoundException {  
        new java.io.FileReader("arquivo.txt");  
    }  
}
```


Exemplos

```
class TestaException2 {  
    public static void main(String[] args) {  
        try {  
            new java.io.FileReader("arquivo.txt");  
        } catch (java.io.FileNotFoundException erro) {  
            //codigo de tratamento de exceção  
            System.out.println("ERRO");  
            //erro.printStackTrace();  
        }  
    }  
}
```

```
class TestaException {  
    public static void main(String[] args) throws java.io.FileNotFoundException {  
        new java.io.FileReader("arquivo.txt");  
    }  
}
```




Obrigada!

professora@lucilia.com.br