

# Mecanismos para o tratamento de exceções

Prof. André Takeshi Endo

# Exceção

- Exceção (*Exception Handling*)
- Representa uma condição **inesperada/anormal**
  - Lançadas pelas classes (métodos)
  - Lançadas pelo ambiente de execução Java
- Não é um conceito de OO
- Mas está presente na maioria das linguagens de programação OO

# Exceção

- Analogia

## Sequência padrão (sem tratar exceção)

```
mao.cortar(cebola); //exceção ?  
panela.adicionar(cebola);  
mao.cortar(tomate); //exceção ?  
panela.adicionar(tomate);  
panela.adicionar(oleo.medida(colher));  
panela.gerarcomida();  
comer();
```

 Fluxo normal

 Fluxo com exceção no 2º  
mao.cortar

## Sequência com tratamento de exceção

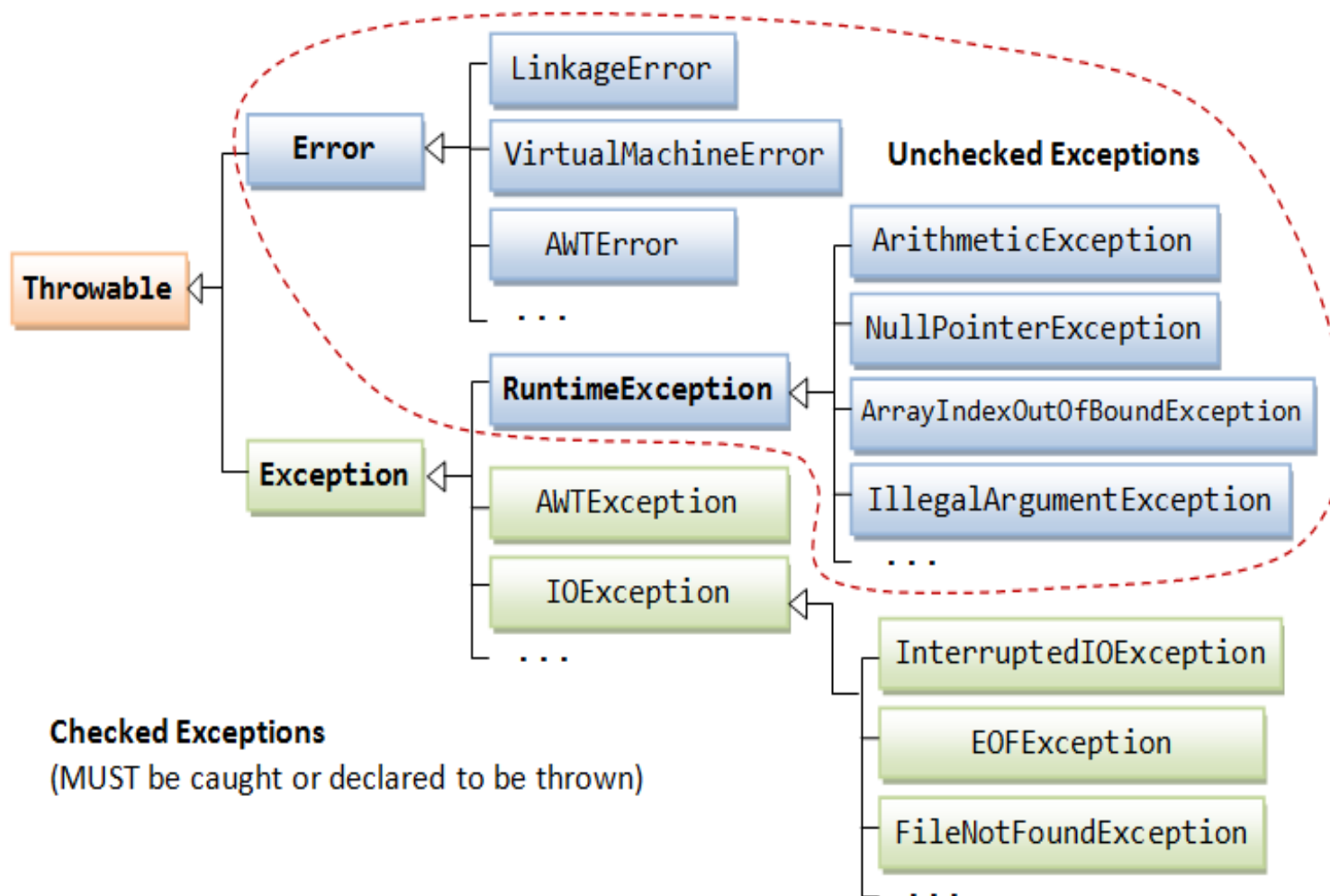
```
tente  
{  
    mao.cortar(cebola); //exceção ?  
    panela.adicionar(cebola);  
    mao.cortar(tomate); //exceção ?  
    panela.adicionar(tomate);  
    panela.adicionar(oleo.medida(colher));  
    panela.gerarcomida();  
}  
imprevisto(CortarDedo e)  
{  
    socorro.aplicarCurativo(mao);  
}  
comer();
```

# Exceção em Java

- Bloco try-catch-finally
- throws → na definição do método para indicar que tal método pode lançar uma exceção
- throw → instrução que lança um **objeto de classe exceção**

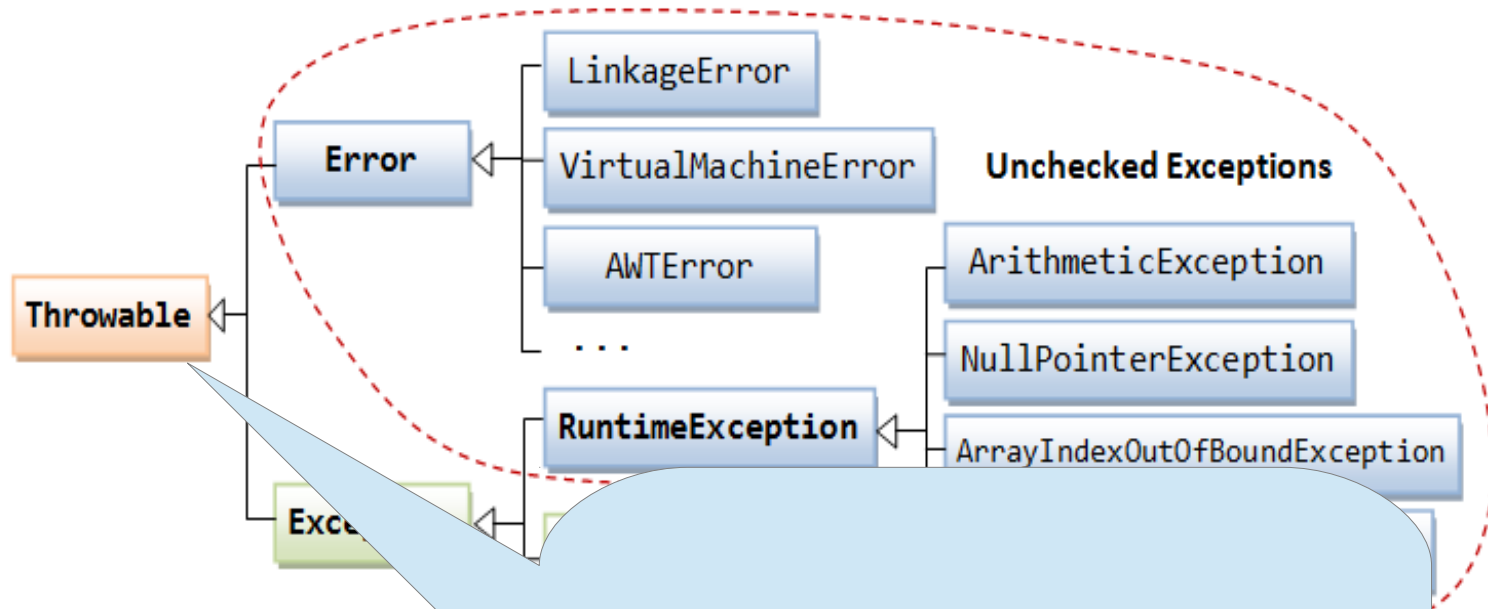
# Exceção em Java

- Toda classe de exceção herda da Throwable



# Exceção em Java

- Toda classe de exceção herda da Throwable



## Checked Exceptions

(MUST be caught or declared to

Nas instruções, catch, throws e throw, devem ser obrigatoriamente objetos (classes) que herdam desta classe.

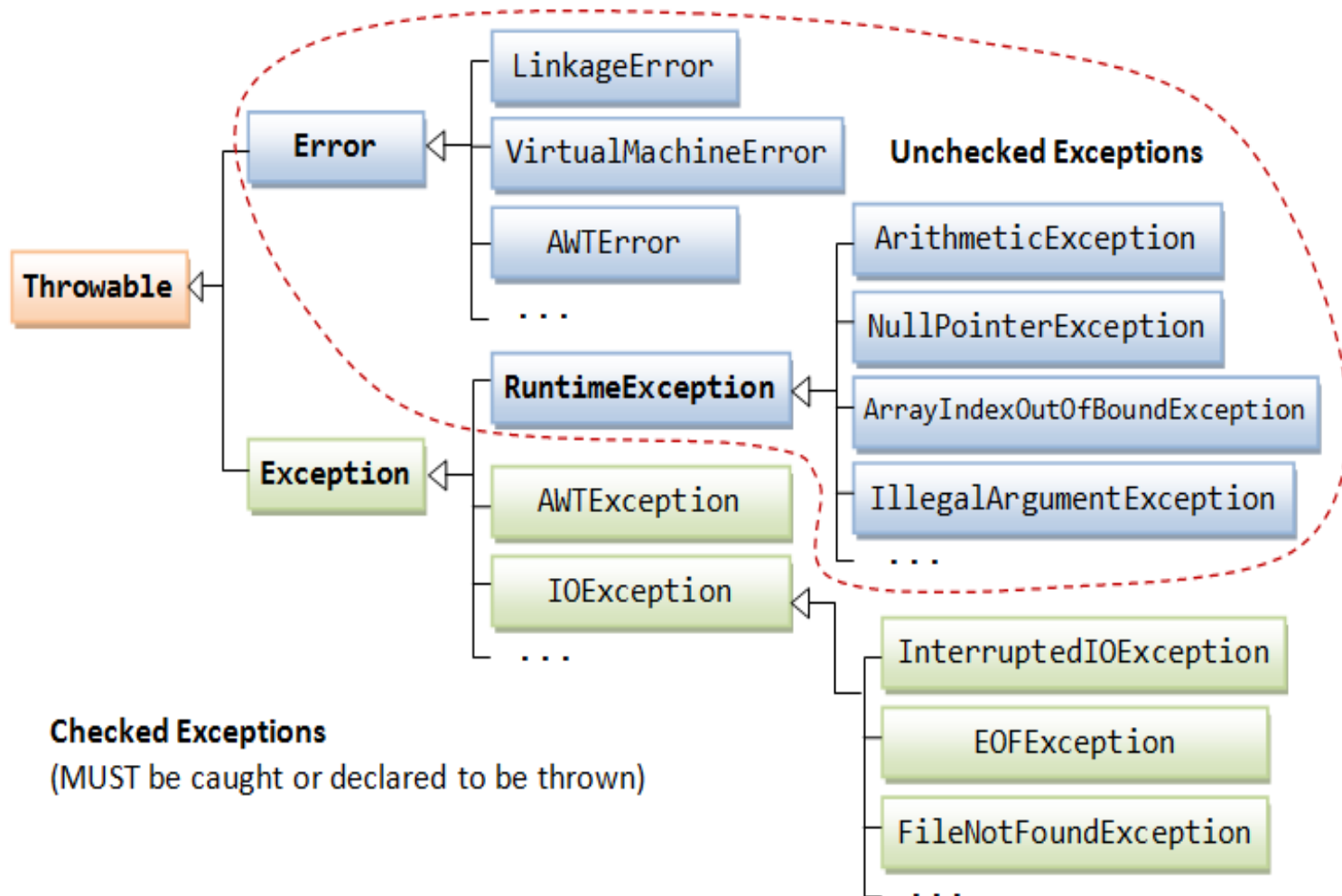
# Exceção em Java

- Algumas exceções não precisam ser checadas (try-catch-finally)
- Em outras palavras, o compilador não vai reclamar que determinada exceção não foi tratada



# Exceção em Java

- Exceções que herdam de Error e RuntimeException



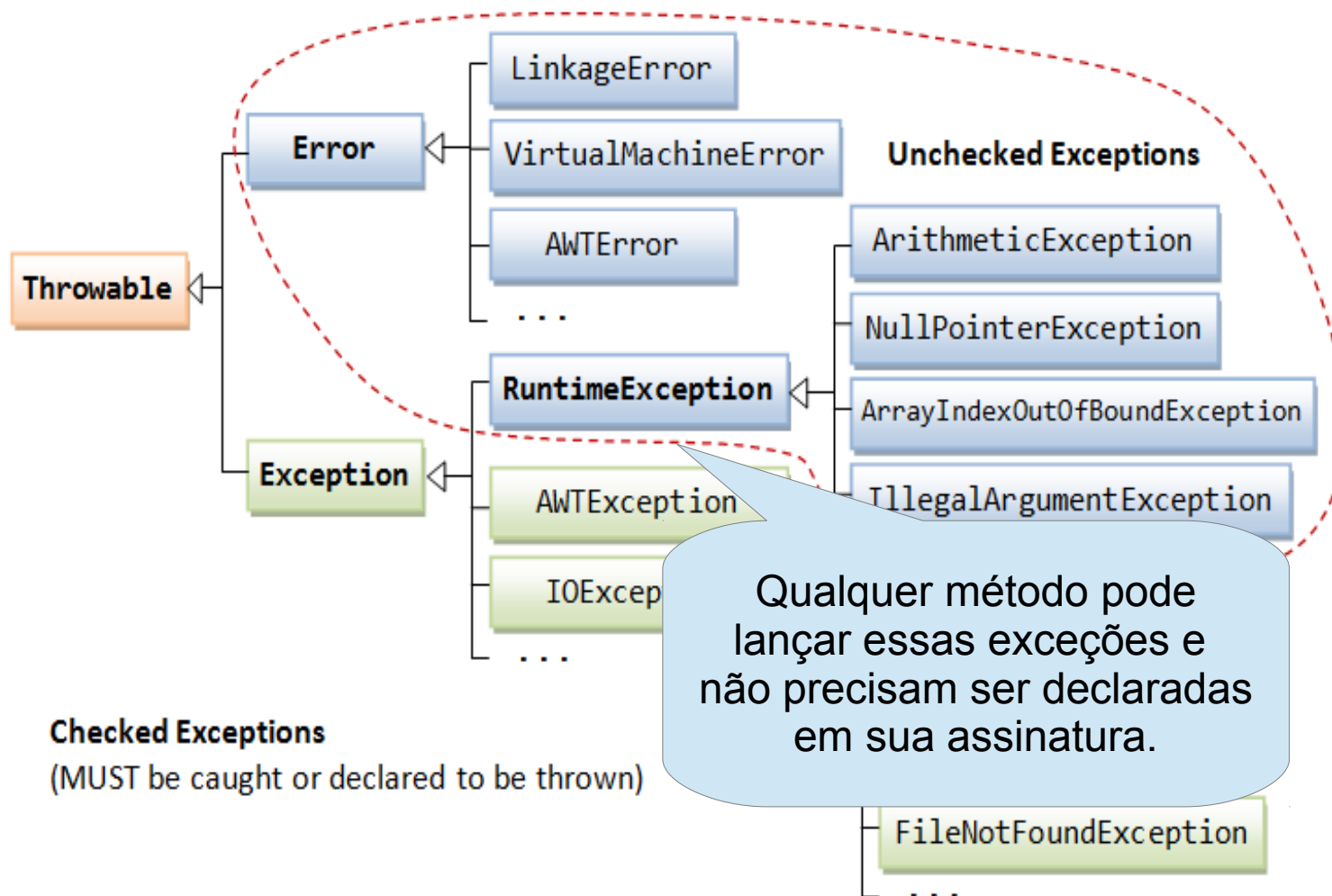
<http://docs.oracle.com/javase/7/docs/api/java/lang/Error.html>

<http://docs.oracle.com/javase/7/docs/api/java/lang/RuntimeException.html>



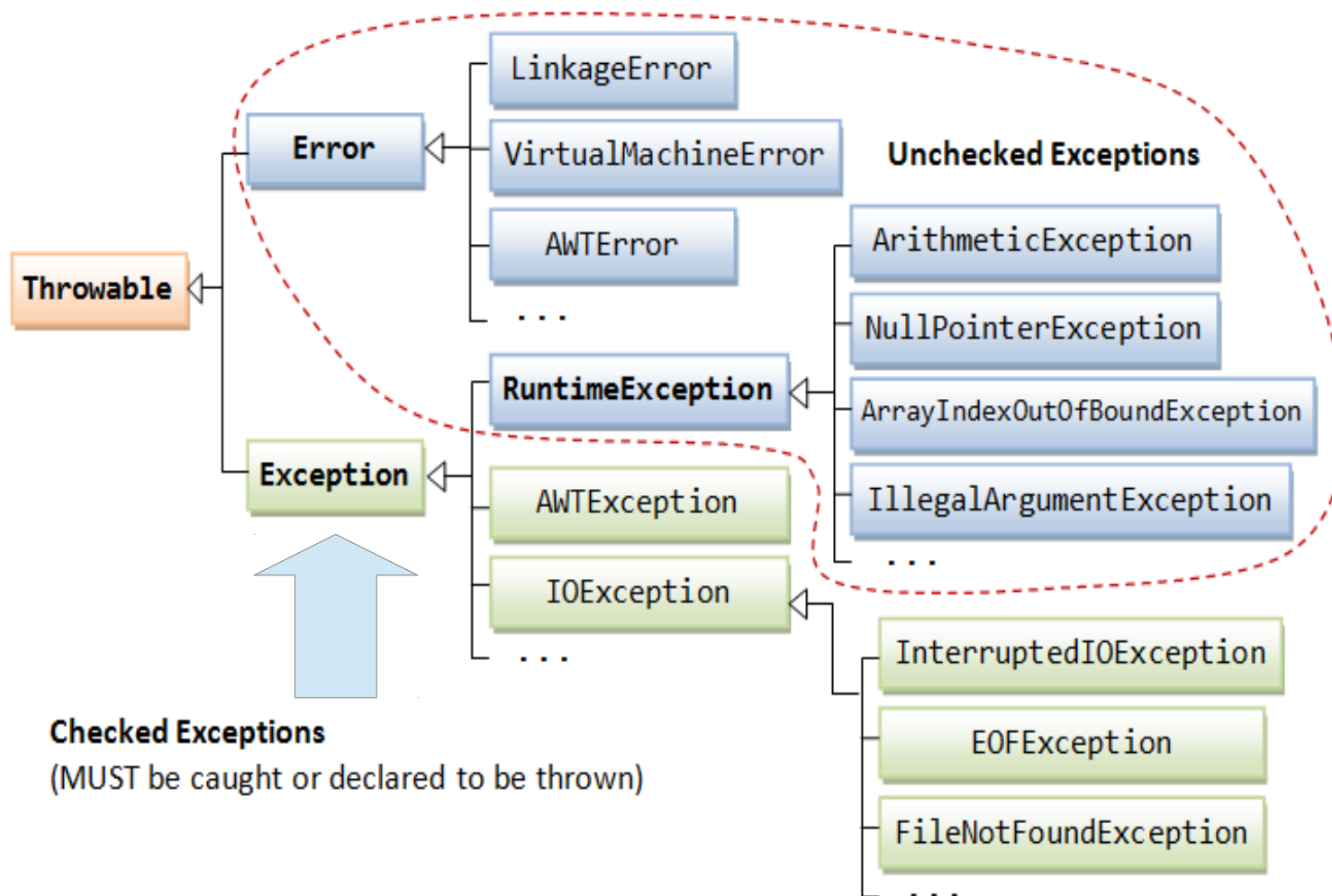
# Exceção em Java

- Exceções que herdam de Error e RuntimeException



# Exceção em Java

- Classe Exception → o programador deve tratar



# Exceção em Java

- Classe Exception → o programador deve tratar
- Um método que pode lançar uma Exception (ou filhos), quando chamá-lo:
  - Tratar com um bloco try-catch
  - Propagá-lo pelo método que está chamando

# Bloco try-catch-finally

**Ordem de captura  
das exceções.  
Quando a primeira  
na ordem consegue  
capturar, as  
demais não são  
consideradas**

Pode-se  
especializar as  
Exceções  
geradas a partir  
das subclasses  
de Exception

```
try {  
    // Código que pode gerar uma exceção  
}
```

```
catch (MinhaExcecao1 e) {  
    // Código para processar a exceção  
}
```

```
catch (MinhaExcecao2 e) {  
    // Código para processar a exceção  
}
```

```
finally {  
    // Código que sempre será executado  
}
```

# Bloco try-catch-finally

**Ordem de captura das exceções.  
Quando a primeira na ordem consegue capturar, as demais não são consideradas**

Pode-se especializar as Exceções geradas a partir das subclasses de Exception

```
try {  
    // Código que pode gerar uma exceção  
}
```

```
catch (MinhaExcecao1 e) {  
    // Código para processar a exceção  
}
```

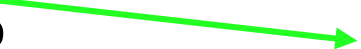
```
catch (MinhaExcecao2 e) {  
    // Código para processar a exceção  
}
```

```
finally {  
    // Código que sempre será executado  
}
```

opcional


# Bloco try-catch-finally

A execução começa  
no início do bloco



O bloco **finally** é executado após  
a execução normal do bloco **try**.  
OBS: Havendo um **return** no **try**, o  
bloco **finally** sempre será  
executado antes da  
saída do método.

Se não houver **return** no bloco  
**try** nem no bloco **finally**,  
a execução continua com  
o código após o bloco **finally**



```
try {  
    // Código que pode gerar uma exceção  
}
```

```
catch (MinhaExcecao1 e) {  
    // Código para processar a exceção  
}
```

```
catch (MinhaExcecao2 e) {  
    // Código para processar a exceção  
}
```

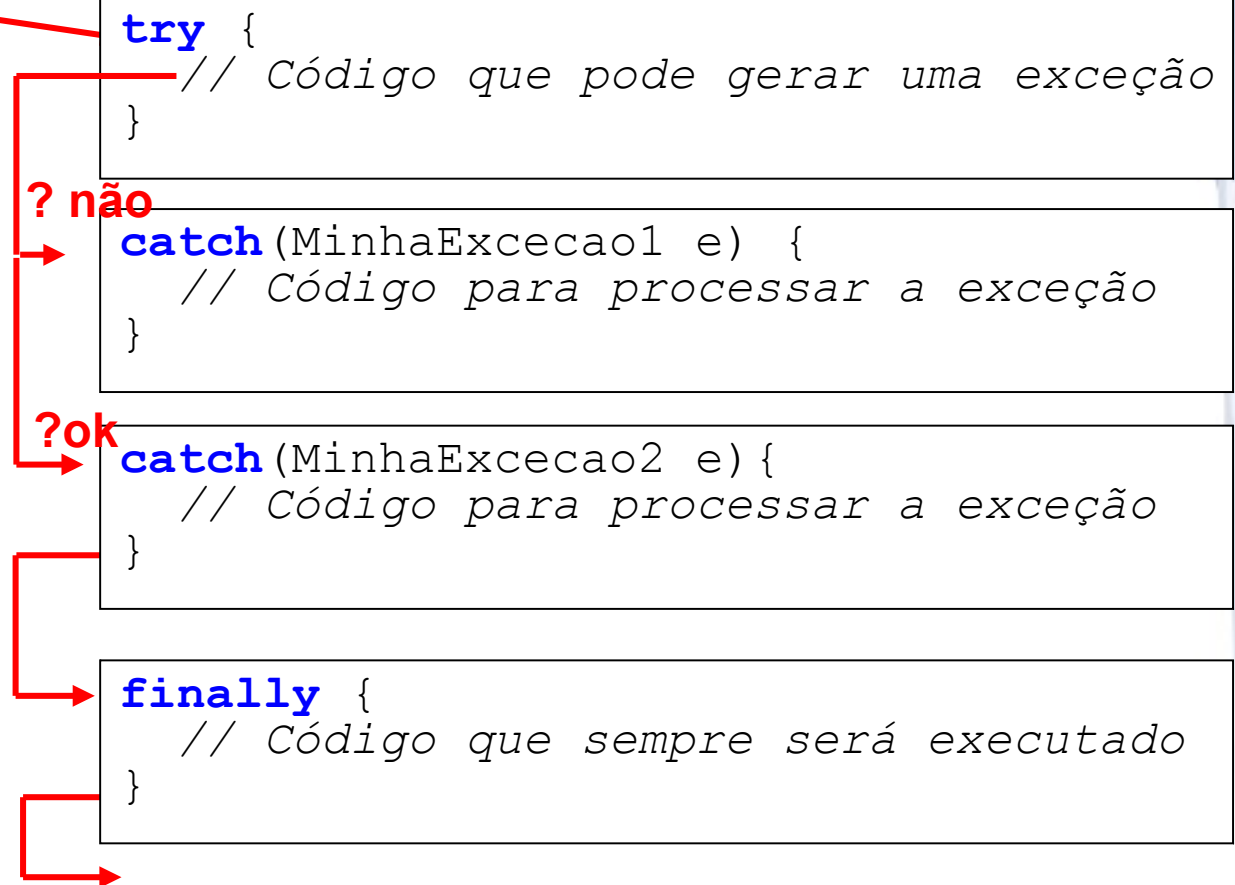
```
finally {  
    // Código que sempre será executado  
}
```

# Bloco try-catch-finally

A execução começa  
no início do bloco

A execução é interrompida  
no ponto onde foi gerado  
a exceção. O controle passa  
para o **primeiro bloco**  
**catch** que conseguir capturá-la.  
(Deixar o **catch** mais genérico  
para o final)

O bloco **finally** será  
executado logo  
após o bloco **catch**.  
A execução continua com  
o código após o bloco **finally**  
caso não exista **return**.





# Exercícios

- Gere o lançamento de um `NullPointerException`
  - Faça seu tratamento
- Gere o lançamento de `ArrayIndexOutOfBoundsException`
  - Faça seu tratamento
- Divisão por zero lança exceção? Qual?

# Exercícios

- Observe a saída para esse código (Caelum)

```
class TesteErro {  
    public static void main(String[] args) {  
        System.out.println("inicio do main");  
        metodo1();  
        System.out.println("fim do main");  
    }  
    static void metodo1() {  
        System.out.println("inicio do metodo1");  
        metodo2();  
        System.out.println("fim do metodo1");  
    }  
    static void metodo2() {  
        System.out.println("inicio do metodo2");  
        int[] array = new int[10];  
        for (int i = 0; i <= 15; i++) {  
            array[i] = i;  
            System.out.println(i);  
        }  
        System.out.println("fim do metodo2");  
    }  
}
```

# Exercícios

- Criar uma classe bancária com um nome do cliente, saldo e número da conta
- Método efetuarSaque(double value)
  - Lançar a exceção IllegalArgumentException quando value for menor ou igual a zero
  - Criar uma exceção SaldoInsuficienteException
    - Lançá-la quando não existir saldo insuficiente
- Testes em JUnit para verificar essa classe

# Referências

- GOODRICH, Michael T.; TAMASSIA, Roberto. Estruturas de dados e algoritmos em Java. 5. ed. Porto Alegre: Bookman, 2013. xxii, 713 p. ISBN 9788582600184.
- Documentação oficial em:  
<http://docs.oracle.com/javase/tutorial/>
- <http://www.caelum.com.br/apostila-java-orientacao-objetos/excecoes-e-controle-de-erros/>
- Material didático sobre POO do prof. Fernando Barreto