



# Técnicas de Programação e Algoritmos - TPA

- Estrutura de Decisão Simples e Composta
  - JAVA -



# Objetivos

- Apresentar os tipos de Estruturas de Decisão Simples e Composta utilizados na linguagem Java.
- Conhecer os Operadores Lógicos e Relacionais



# Estruturas de Decisão - Conceito

- ▶ Uma estrutura de decisão permite a escolha de um grupo de ações e estruturas a serem executadas, quando determinadas condições são ou não satisfeitas.
- ▶ A estrutura de decisão testa uma determinada **CONDIÇÃO** através do comando “If (condição)”.
- ▶ Caso a resposta à condição seja VERDADEIRA, o programa executará a linha de instruções, ou o bloco de instruções da parte verdadeira do comando de decisão.

# Estruturas de Decisão Simples

**if** (condição)

comando para condição verdadeira;

## - Sintaxe Java:

**if** (condição)

{

Comando 1 para condição verdadeira;

Comando 2 para condição verdadeira;

Comando n para condição verdadeira;

}

# Estrutura de Decisão – Condição

**Condição** → é uma expressão lógica que quando inspecionada (testada) pode gerar um resultado falso ou verdadeiro. Exemplos:

```
if (10>20) { comandos ... }
```

```
if (a==b) { comandos ... }
```

```
if (b<=100) { comandos ... }
```

**Obs:** Para comparar valores utilizamos o operador “==”, pois o operador “=” é usado para atribuição de dados. Exemplo:

```
String nome = “Maria das Dores”;
```



# Estrutura de Decisão Composta

Neste modelo teremos comandos que serão executados tanto para o resultado verdadeiro quanto para o resultado falso.

**if** (condição)

Comando para condição verdadeira;

**else**

Comando para condição falsa;

# Estrutura de Decisão Composta – Sintaxe Java

**if** (condição)

{

Comando 1 para condição verdadeira;

Comando 2 para condição verdadeira;

Comando n para condição verdadeira;

}

**else**

{

Comando 1 para condição falsa;

Comando 2 para condição falsa;

Comando n para condição falsa;

}

# Operadores Relacionais

Operadores relacionais estabelecem comparações entre dois valores de mesmo tipo primitivo:

== Igual

> Maior

< Menor

<> Diferente

>= Maior igual

<= Menor igual





# Operadores relacionais

- Os operadores relacionais são usados para comparar valores em algoritmos e são fundamentais para a tomada de decisões.
- Eles retornam um valor booleano (verdadeiro ou falso) com base na comparação feita. Aqui estão os principais operadores relacionais:



# Igual a (==):

- Verifica se dois valores são iguais.
- if (a == b):
- # Executa se a for igual a b



# Diferente de (!=):

- Verifica se dois valores são diferentes.

if (a != b):

    # Executa se a for diferente de b



# Maior que (>):

- Verifica se um valor é maior que outro.

if (a > b):

    # Executa se a for maior que b



# Menor que (<):

- Verifica se um valor é menor que outro.

if (a < b):

    # Executa se a for menor que b

# Maior ou igual a ( $\geq$ ):

- Verifica se um valor é maior ou igual a outro.

if ( $a \geq b$ ):

# Executa se a for maior ou igual a b

# Menor ou igual a ( $\leq$ ):

- Verifica se um valor é menor ou igual a outro.

if (a  $\leq$  b):

# Executa se a for menor ou igual a b



# Exemplos em Pseudocódigo

## ■ Operador Igual a (==)

```
se (nota == 10) então  
    escrever "Nota máxima!"  
fim se
```



# Operador Diferente de (!=)

```
se (nota != 10) então  
    escrever "Nota não é máxima."  
fim se
```



# Operador Maior que (>)

se (idade > 18) então

    escrever "Maior de idade"

fim se



# Operador Menor que (<)

se (idade < 18) então

    escrever "Menor de idade"

fim se

# Operador Maior ou igual a ( $\geq$ )

```
se (nota  $\geq$  7) então  
    escrever "Aprovado"  
fim se
```



# Operador Menor ou igual a ( $\leq$ )

```
se (nota  $\leq$  4) então  
    escrever "Reprovado"  
fim se
```

# Operadores Lógicos


Os operadores lógicos permitem complementar e conectar novas formações de comparações:

Operador		Símbolo	Função
E	And	&&	Conjunção
Ou	Or		Disjunção
Não	Not	!	Negação

## Exemplos:

```
if(a<b) && (b<c) { comandos ... }
```

```
if(a<b) || (b<c) || (c<d) { comandos ... }
```

- 
- Os operadores lógicos são fundamentais na programação e na construção de algoritmos, pois permitem a combinação de várias condições.
  - Eles são usados para controlar o fluxo de execução dos programas, especialmente em estruturas de decisão (como if, while, for).
  - Aqui estão os principais operadores lógicos utilizados em algoritmos:

## AND (E) - && ou and:

- Este operador retorna true apenas se **ambas** as condições forem verdadeiras.

if (condicao1 and condicao2):

    # Executa se ambas as condições forem verdadeiras




# OR (OU) - || ou or:

- Este operador retorna true se **pelo menos uma** das condições for verdadeira.

if (condicao1 or condicao2):

    # Executa se pelo menos uma das  
condições for verdadeira



# NOT (NÃO) - ! ou not:

- Este operador inverte o valor lógico de uma condição.

if not (condicao):

# Executa se a condição for falsa

# Exemplos em Pseudocódigo

## ■ Operador AND

```
se (idade >= 18) e (temCarteiraDeMotorista) então  
    escrever "Pode dirigir"  
senão  
    escrever "Não pode dirigir"  
fim se
```



# Operador OR

se (idade < 18) ou (temCarteiraDeMotorista  
== falso) então

    escrever "Não pode dirigir"

senão

    escrever "Pode dirigir"

fim se



# Operador NOT

se não (temCarteiraDeMotorista) então

    escrever "Não pode dirigir"

senão

    escrever "Pode dirigir"

fim se



# Tabela Verdade

- A tabela verdade é uma ferramenta usada na lógica para descrever o comportamento de operadores lógicos em todas as possíveis combinações de valores das variáveis envolvidas.
- Ela é particularmente útil na construção e análise de algoritmos que envolvem lógica condicional.
- Vamos ver como construir tabelas verdade para os operadores lógicos básicos: AND, OR e NOT.

# Tabela Verdade para o Operador AND (E)

- O operador AND (&& ou and) retorna verdadeiro somente quando ambas as condições são verdadeiras.

A	B	A AND B
True	True	True
True	False	False
False	True	False
False	False	False

# Tabela Verdade para o Operador OR (OU)

- O operador OR (|| ou or) retorna verdadeiro quando pelo menos uma das condições é verdadeira.

A	B	A OR B
True	True	True
True	False	True
False	True	True
False	False	False



# Tabela Verdade para o Operador NOT (NÃO)

- O operador NOT (! ou not) inverte o valor lógico da condição.

A	NOT A
True	False
False	True



# Estrutura tabela verdade em java

- Para criar um algoritmo em Java que utiliza a estrutura de tabela verdade, podemos implementar um exemplo que demonstra o comportamento dos operadores lógicos AND, OR e NOT. Esse exemplo vai usar uma tabela verdade para todas as combinações possíveis de dois valores booleanos.



```
public class TabelaVerdade {
```

```
    public static void main(String[] args) {  
        boolean[] valores = {true, false};
```

```
        System.out.println("A\tB\tA AND B\tA OR B\tNOT A");
```

```
        for (boolean A : valores) {  
            for (boolean B : valores) {  
                boolean andResult = A && B;  
                boolean orResult = A || B;  
                boolean notResult = !A;
```

```
                System.out.println(A + "\t" + B + "\t" + andResult + "\t" + orResult +  
"\t" + notResult);
```

```
            }
```

```
        }
```

```
    }
```

```
}
```