GERMINARE
Escola de Negócios

GerminaTECH

Lógica de Programação com Java – LPR

Operadores Relacionais

Prof. Carlos Eduardo Santi

Operadores relacionais

- Estes operadores são usados quando é necessário comparar valores explícitos ou armazenados em variáveis.
- Por exemplo, se tivermos que verificar se a idade de alguém é maior que 18, ou se o saldo na conta é maior que zero.
- Podemos dizer que os operadores relacionais são usados para comparar valores.

Operadores relacionais

- Crie um novo projeto chamado **Relacionais**, crie duas variáveis inteiras A e B e atribua os valores 10 e 25 a elas, respectivamente.
- Mostre o resultado de **A > B**, **A < B**, **A == B**, **A != B**.

```
public class Main {
    public static void main(String[] args) {
        int A = 10, B = 25;
        System.out.println(A < B);
        System.out.println(A > B);
        System.out.println(A == B);
        System.out.println(A != B);
    }
}
```

```
true
false
false
true
```

Operadores relacionais

- Como já era previsto, os dois possíveis resultados para uma operação envolvendo operadores relacionais são ‘true’ ou ‘false’.
- O resultado das comparações de A e B, levaram, no caso de A > B, a ‘false’. Como A = 10 e B = 25, a comparação 10 > 25 resulta em FALSO!
- No outro caso, A < B levou ao resultado ‘true’, pois 10 é menor que 25.

GERMINARE
Escola de Negócios

Operadores relacionais

Lista de operadores relacionais.

OPERADOR	DESCRIÇÃO	EXEMPLO
>	Maior que	A > B
<	Menor que	A < B
>=	Maior ou igual a	A >= B
<=	Menor ou igual a	A <= B
==	Igual	A == B
!=	Não Igual (Diferente)	A != B

Veja que o operador de igualdade tem dois sinais de igual e o de diferença tem um sinal de exclamação e um igual, verifica se é “não igual”.

Operadores lógicos

- Os operadores lógicos trabalham com os valores lógicos ‘true’ e ‘false’, explicitamente, ou com o resultado de operações e expressões relacionais.
- Suas operações são representadas em tabelas-verdade:

Negação	
A	!A
false	true
true	false

E / AND		
A	B	A && B
false	false	false
false	true	false
true	false	false
true	true	true

OU / OR		
A	B	A || B
false	false	false
false	true	true
true	false	true
true	true	true

Atribuição de valores lógicos

- Uma variável pode receber um valor lógico explicitamente:

```
public class Main {
    public static void main(String[] args) {
        boolean A = true, B = false;

        System.out.println(A && B);
        System.out.println(A || B);
        System.out.println(A && !B);
        System.out.println(!A || B);
    }
}
```

```
false
true
true
false
```

Aqui, as variáveis A e B recebem valores **true** e **false**, respectivamente.

Atribuição de valores lógicos

- Uma variável pode receber um valor como resultado de uma expressão relacional:

```
public class Main {
    public static void main(String[] args) {
        boolean adulto;
        int idade = 30;
        adulto = (idade > 18);

        System.out.println(adulto);
    }
}
```

true

Aqui, a variável adulto recebe o resultado da comparação de **idade** com o valor **18**

Expressões lógicas

No exemplo anterior, atribuímos o valor **true** para a variável adulto, pois o valor de idade era maior que 18.

O que será apresentado ao executar o código abaixo?

```
public class Main {
    public static void main(String[] args) {
        double altura = 1.6;
        int idade = 12;
        boolean permite;
        permite = (idade >= 10) && (altura > 1.5);
                          true                 true
        System.out.println(permite);
    }
}
```

Precedência dos operadores

- O operadores lógicos também seguem uma regra de precedência, respeitando a ordem em que aparecem na tabela abaixo.
- Parênteses sempre têm precedência!

OPERADOR	DESCRIÇÃO	EXEMPLO
!	NÃO/NOT lógico (operador unário)	!A
&&	E/AND lógico	A && B
||	OU/OR lógico	A || B

Precedência dos operadores

- Avalie a expressão abaixo:

```
public class Main {
    public static void main(String[] args) {
        boolean saldo = false, especial = false, senha = false, podeSacar;
        podeSacar = !senha && (!saldo || especial);

        System.out.println(podeSacar);
    }
}
```

Resolução

1. Parênteses --- !saldoDisponivel, depois a operação **||**
2. !senhaCorreta, depois a operação **&&**