Aula 07 – Condicionais e Operadores Lógicos

Norton Trevisan Roman

9 de abril de 2018

Condicionais Aninhados

- Vamos testar também os parâmetros de `areaPiscina`

```
static double areaPiscina(
                    double raio) {
  if (raio >= 0) return Math.PI
                * Math.pow(raio,2);
  else return(-1);
}
```

Condicionais Aninhados

- Vamos testar também os parâmetros de `areaPiscina`

```
static double areaPiscina(
                    double raio) {
  if (raio >= 0) return Math.PI
              * Math.pow(raio,2);
  else return(-1);
}
```

- E em `areaCasa`, como fazemos?
- Existem 2 parâmetros a serem testados

```
static void areaCasa(float
      lateral, float cquarto) {
  float areaq;
  float areas;
  float areat;
  ...
}
```

Condicionais Aninhados

- Primeiro teste um

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                          lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                        lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Condicionais Aninhados

- Primeiro teste um
- Se der problema, acuse o erro

```
static void areaCasa(float lateral,
                                    float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                              lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                            lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Condicionais Aninhados

- Primeiro teste um
- Se der problema, acuse o erro
- Senão, teste o segundo

```
static void areaCasa(float lateral,
                                    float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                            lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                          lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Condicionais Aninhados

- Primeiro teste um
- Se der problema, acuse o erro
- Senão, teste o segundo
- Se der problema, acuse o erro

```
static void areaCasa(float lateral,
                                    float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                            lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                          lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Condicionais Aninhados

- Primeiro teste um
- Se der problema, acuse o erro
- Senão, teste o segundo
- Se der problema, acuse o erro
- Senão, continue com o cálculo

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                          lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                        lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Condicionais Aninhados

- Diz-se que os IFs estão aninhados ou encaixados

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                          lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                        lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Condicionais Aninhados

- Dentro de um condicional podemos ter qualquer tipo de comando

```
if (<condicao 1>) <comando 1>;
else
  if (<condicao 2>) <comando 2>;
  else
    if (<condicao 3>) <comando 3>;
    else <comando 4>;
```

Condicionais Aninhados

- Dentro de um condicional podemos ter qualquer tipo de comando

```
if (<condicao 1>) <comando 1>;
else
  if (<condicao 2>) <comando 2>;
  else
    if (<condicao 3>) <comando 3>;
    else <comando 4>;
```

- Inclusive outro condicional

Condicionais Aninhados

- Dentro de um condicional podemos ter qualquer tipo de comando

```
if (<condicao 1>) <comando 1>;
else
  if (<condicao 2>) <comando 2>;
  else
    if (<condicao 3>) <comando 3>;
    else <comando 4>;
```

- Inclusive outro condicional
- Note a ausência do $\{\} \rightarrow$ usamos quando há mais de um comando (um bloco)

Condicionais Aninhados

- Dentro de um condicional podemos ter qualquer tipo de comando

```
if (<condicao 1>) <comando 1>;
else
  if (<condicao 2>) <comando 2>;
  else
    if (<condicao 3>) <comando 3>;
    else <comando 4>;
```

- Inclusive outro condicional
- Note a ausência do $\{\} \rightarrow$ usamos quando há mais de um comando (um bloco)
- O `if...else` conta como um único comando

Condicionais Aninhados

- Identação é fundamental!
- Assim podemos ver qual `else` corresponde a qual `if`

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- Quando comando 1 será executado?

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- Quando comando 1 será executado?
- Quando condição 1, condição 2, condição 3 e condição 4 forem verdadeiras

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- E quando comando 2 será executado?

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- E quando comando 2 será executado?
- Quando `condição 1`, `condição 2` e `condição 3` forem verdadeiras e `condição 4` for falsa

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- E comando 3?

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- E comando 3?
- Quando `condição 1` e `condição 2` forem verdadeiras e `condição 3` for falsa

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- E comando 4?

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- E comando 4?
- Quando condição 1 for verdadeira e condição 2 falsa

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                    <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- E comando 5?

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- E comando 5?
- Quando condição 1 for falsa

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;
            else <comando 2>;
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- E se removermos uma linha, quando comando 3 será executado?

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;

        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- E se removermos uma linha, quando comando 3 será executado?

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                    <comando 1>;

        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

- Quando `condição 1`, `condição 2` e `condição 3` forem verdadeiras e `condição 4` for falsa

Condicionais Aninhados

- O compilador achará que o `else` é do `if` mais próximo

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;

        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- O compilador achará que o `else` é do `if` mais próximo
- Que fazer?

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>)
            if (<condicao 4>)
                <comando 1>;

        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- O compilador achará que o `else` é do `if` mais próximo
- Que fazer?
- Mudar a forma de entendimento com {}

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>){
            if (<condicao 4>)
                <comando 1>;
        }
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Condicionais Aninhados

- O compilador achará que o `else` é do `if` mais próximo
- Que fazer?
- Mudar a forma de entendimento com {}
- Agora sim, o else está alinhado ao if correto

```
if (<condicao 1>)
    if (<condicao 2>)
        if (<condicao 3>){
            if (<condicao 4>)
                    <comando 1>;
        }
        else <comando 3>;
    else <comando 4>;
else <comando 5>;
```

Print

- Voltemos ao main
- Podemos incrementar um pouco a resposta ao usuário
- `print` faz o mesmo que `println`, exceto que não dá nova linha

```
public static void main(String[]
                                    args) {
  double preco;
  boolean valorOK = false;

  preco = valor(-20);
  valorOK = preco >= 0;

  System.out.print("O valor da
                          construção ");
  if (valorOK)
    System.out.println("é: "+preco);
  else
    System.out.println("não foi
            obtido: área negativa");
}
```

Print

- Permite, assim, que possamos mudar parte da mensagem, conforme o resultado de algum condicional

```
public static void main(String[]
                                    args) {
  double preco;
  boolean valorOK = false;

  preco = valor(-20);
  valorOK = preco >= 0;

  System.out.print("O valor da
                          construção ");
  if (valorOK)
    System.out.println("é: "+preco);
  else
    System.out.println("não foi
            obtido: área negativa");
}
```

Operadores Lógicos

- Considere o método para cálculo da área da casa

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                           lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                         lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Operadores Lógicos

- Considere o método para cálculo da área da casa
- São necessários mesmo 2 IFs para isso?
- O que eles significam?

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                           lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                         lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Operadores Lógicos

- Que o método acusará erro quando *lateral* < 0 ou *cquarto* < 0

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                           lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                         lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Operadores Lógicos

- Que o método acusará erro quando $lateral < 0$ ou $cquarto < 0$
- Precisamos de um meio de representar isso

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                           lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                         lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Operadores Lógicos

- Que o método acusará erro quando $lateral < 0$ ou $cquarto < 0$
- Precisamos de um meio de representar isso
  - Operador lógico or

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                           lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                         lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Operadores Lógicos

- Em java, ou é representado por ||

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;

  if (lateral<0 || cquarto<0)
    System.out.println("Erro: parâmetro<0");
  else {
    System.out.println("...");
    areas = lateral*lateral;
    System.out.println("..."+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("..."+areaq);
    System.out.println("..."+areaq);
    areat = areas + 2*areaq;
    System.out.println("..."+areat);
  }
}
```

Operadores Lógicos

- Em java, ou é representado por ||
- E como o compilador sabe que deve fazer o < antes?

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;

  if (lateral<0 || cquarto<0)
    System.out.println("Erro: parâmetro<0");
  else {
    System.out.println("...");
    areas = lateral*lateral;
    System.out.println("..."+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("..."+areaq);
    System.out.println("..."+areaq);
    areat = areas + 2*areaq;
    System.out.println("..."+areat);
  }
}
```

Operadores Lógicos

- Em java, ou é representado por ||
- E como o compilador sabe que deve fazer o < antes?
- Precedência: operadores relacionais têm precedência sobre operadores lógicos

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;

  if (lateral<0 || cquarto<0)
    System.out.println("Erro: parâmetro<0");
  else {
    System.out.println("...");
    areas = lateral*lateral;
    System.out.println("..."+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("..."+areaq);
    System.out.println("..."+areaq);
    areat = areas + 2*areaq;
    System.out.println("..."+areat);
  }
}
```

Operadores Lógicos

- Voltando ao código anterior...
- Haveria alternativa a dizer que o método acusará erro quando *lateral* < 0 ou *cquarto* < 0?

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                          lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                        lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Operadores Lógicos

- O método irá executar o código somente se *lateral* ≥ 0 e *cquarto* ≥ 0

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                           lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                         lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Operadores Lógicos

- O método irá executar o código somente se *lateral* ≥ 0 e *cquarto* ≥ 0
- Operador lógico *and*

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;
  if (lateral<0) System.out.println("Erro:
                           lateral da sala < 0");
  else {
    if (cquarto<0) System.out.println("Erro:
                         lateral do quarto < 0");
    else {
      System.out.println("...");
      areas = lateral*lateral;
      System.out.println("..."+areas);
      areaq = cquarto*(lateral/2);
      System.out.println("..."+areaq);
      System.out.println("..."+areaq);
      areat = areas + 2*areaq;
      System.out.println("..."+areat);
    }
  }
}
```

Operadores Lógicos

- E como representamos um e em java?

Operadores Lógicos

- E como representamos um e em java?
- &&

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;

  if (lateral>=0 && cquarto>=0) {
    System.out.println("...");
    areas = lateral*lateral;
    System.out.println("..."+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("..."+areaq);
    System.out.println("..."+areaq);
    areat = areas + 2*areaq;
    System.out.println("..."+areat);
  }
  else
    System.out.println("Erro: parâmetro<0");
}
```

Operadores Lógicos

- Teríamos ainda outra alternativa?

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;

  if (lateral>=0 && cquarto>=0) {
    System.out.println("...");
    areas = lateral*lateral;
    System.out.println("..."+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("..."+areaq);
    System.out.println("..."+areaq);
    areat = areas + 2*areaq;
    System.out.println("..."+areat);
  }
  else
    System.out.println("Erro: parâmetro<0");
}
```

Operadores Lógicos

- Teríamos ainda outra alternativa?
- Se não for verdade que *lateral* ≥ 0 e *cquarto* ≥ 0, mostre o erro

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;

  if (lateral>=0 && cquarto>=0) {
    System.out.println("...");
    areas = lateral*lateral;
    System.out.println("..."+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("..."+areaq);
    System.out.println("..."+areaq);
    areat = areas + 2*areaq;
    System.out.println("..."+areat);
  }
  else
    System.out.println("Erro: parâmetro<0");
}
```

Operadores Lógicos

- Teríamos ainda outra alternativa?
- Se não for verdade que *lateral* ≥ 0 e *cquarto* ≥ 0, mostre o erro
- Operador lógico *not*

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;

  if (lateral>=0 && cquarto>=0) {
    System.out.println("...");
    areas = lateral*lateral;
    System.out.println("..."+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("..."+areaq);
    System.out.println("..."+areaq);
    areat = areas + 2*areaq;
    System.out.println("..."+areat);
  }
  else
    System.out.println("Erro: parâmetro<0");
}
```

Operadores Lógicos

- E como representamos um não em java?

Operadores Lógicos

- E como representamos um não em java?
- !

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;

  if (!(lateral>=0 && cquarto>=0))
    System.out.println("Erro: parâmetro<0");
  else {
    System.out.println("...");
    areas = lateral*lateral;
    System.out.println("..."+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("..."+areaq);
    System.out.println("..."+areaq);
    areat = areas + 2*areaq;
    System.out.println("..."+areat);
  }
}
```

Operadores Lógicos

- E como representamos um não em java?
- !
- Veja que negamos toda a expressão `lateral>=0 && cquarto>=0`

```
static void areaCasa(float lateral,
                                  float cquarto) {
  float areaq; float areas; float areat;

  if (!(lateral>=0 && cquarto>=0))
    System.out.println("Erro: parâmetro<0");
  else {
    System.out.println("...");
    areas = lateral*lateral;
    System.out.println("..."+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("..."+areaq);
    System.out.println("..."+areaq);
    areat = areas + 2*areaq;
    System.out.println("..."+areat);
  }
}
```

Tabela Verdade

- E: *comando1* será executado?

condição1 \ condição2	V
V	

```
if (condição1)
    if (condição2)
        comando1;
```

Tabela Verdade

- E: *comando1* será executado?

condição1 \ condição2	*V*
V	V

```
if (condição1)
    if (condição2)
        comando1;
```

Tabela Verdade

- E: *comando1* será executado?

condição1	condição2 V	condição2 F
V	V	

```
if (condição1)
    if (condição2)
        comando1;
```

Tabela Verdade

- E: *comando1* será executado?

condição1	condição2	
	V	*F*
V	V	F

```
if (condição1)
    if (condição2)
        comando1;
```

Tabela Verdade

- E: *comando1* será executado?

condição1	condição2	
	V	*F*
V	V	F
F		

```
if (condição1)
    if (condição2)
        comando1;
```

Tabela Verdade

- E: *comando1* será executado?

condição1	condição2	
	V	*F*
V	V	F
F	F	

```
if (condição1)
    if (condição2)
        comando1;
```

Tabela Verdade

- E: *comando1* será executado?

condição1		condição2	
		V	*F*
V		V	F
F		F	F

```
if (condição1)
    if (condição2)
        comando1;
```

Tabela Verdade

- E: *comando1* será executado?

condição1		condição2	
		V	*F*
V		V	F
F		F	F

```
if (condição1)
    if (condição2)
        comando1;
```

- Equivale ao operador aritmético $*$, com $V > 0$ e $F = 0$

Tabela Verdade

- OU: *comando1* será executado?

condição1 \ condição2	*V*
V	

```
if (condição1)
    comando1;
else
    if (condição2)
        comando1;
    else
        comando2;
```

Tabela Verdade

- OU: *comando1* será executado?

condição1	condição2
V	*V*
V	V

```
if (condição1)
    comando1;
else
    if (condição2)
        comando1;
    else
        comando2;
```

Tabela Verdade

- OU: *comando1* será executado?

condição1	condição2 V	condição2 F
V	V	

```
if (condição1)
    comando1;
else
    if (condição2)
        comando1;
    else
        comando2;
```

Tabela Verdade

- OU: *comando1* será executado?

condição1	condição2	
	V	*F*
V	V	V

```
if (condição1)
    comando1;
else
    if (condição2)
        comando1;
    else
        comando2;
```

Tabela Verdade

- OU: *comando1* será executado?

condição1	condição2	
	V	*F*
V	V	V
F		

```
if (condição1)
    comando1;
else
    if (condição2)
        comando1;
    else
        comando2;
```

Tabela Verdade

- OU: *comando1* será executado?

condição1	condição2	
	V	*F*
V	V	V
F	V	

```
if (condição1)
    comando1;
else
    if (condição2)
        comando1;
    else
        comando2;
```

Tabela Verdade

- OU: *comando1* será executado?

condição1		condição2	
		V	*F*
V		V	V
F		V	F

```
if (condição1)
    comando1;
else
    if (condição2)
        comando1;
    else
        comando2;
```

Tabela Verdade

- OU: *comando1* será executado?

condição1		condição2	
		V	*F*
V		V	V
F		V	F

```
if (condição1)
    comando1;
else
    if (condição2)
        comando1;
    else
        comando2;
```

- Equivale ao operador aritmético $+$, com $V > 0$ e $F = 0$

Tabela Verdade

- NÃO: *comando1* será executado?

condição1
V

```
if (!condição1)
                comando1;
```

Tabela Verdade

- NÃO: *comando1* será executado?

condição1	V	F

```
if (!condição1)
                comando1;
```

Tabela Verdade

- NÃO: *comando1* será executado?

condição1	F
V	
F	

```
if (!condição1)
                comando1;
```

Tabela Verdade

- NÃO: *comando1* será executado?

condição1	
V	F
F	V

```
if (!condição1)
                comando1;
```

Operadores

- Operadores aritméticos, relacionais e lógicos podem ser misturados

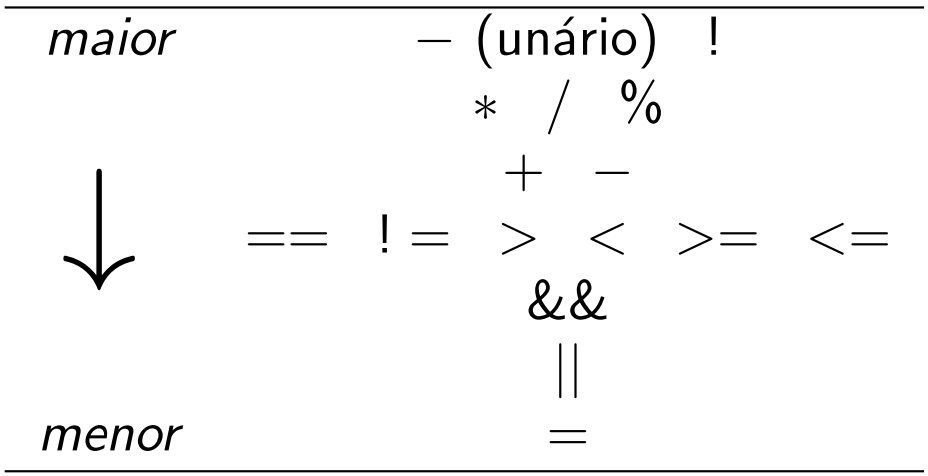
Operadores

- Operadores aritméticos, relacionais e lógicos podem ser misturados
- Como isso será entendido?

```
int x = 3;
if (((20-x)>5) &&
    ((4/x) == 1) ||
    ((16-x)>10))
  System.out.
    println("passou");
```

Operadores

- Precedência:

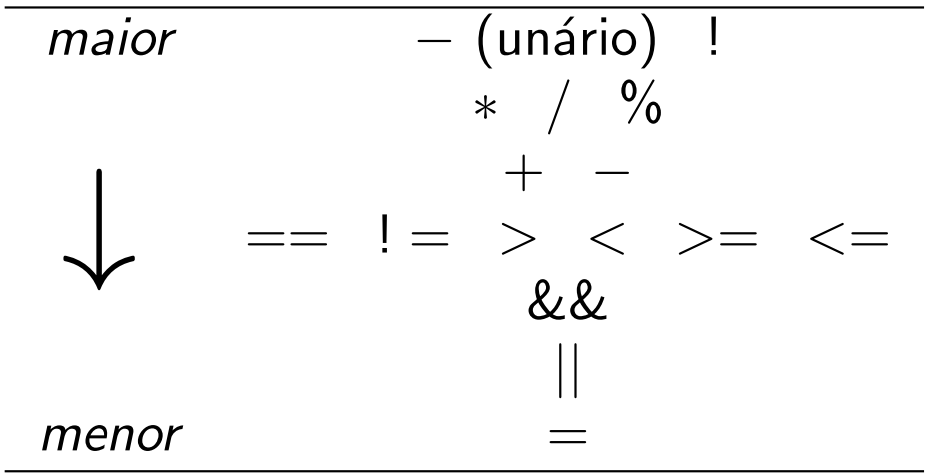



```
int x = 3;
if (((20-x)>5) &&
    ((4/x) == 1) ||
    ((16-x)>10))
  System.out.
    println("passou");
```

- E a resposta é . . .

Operadores

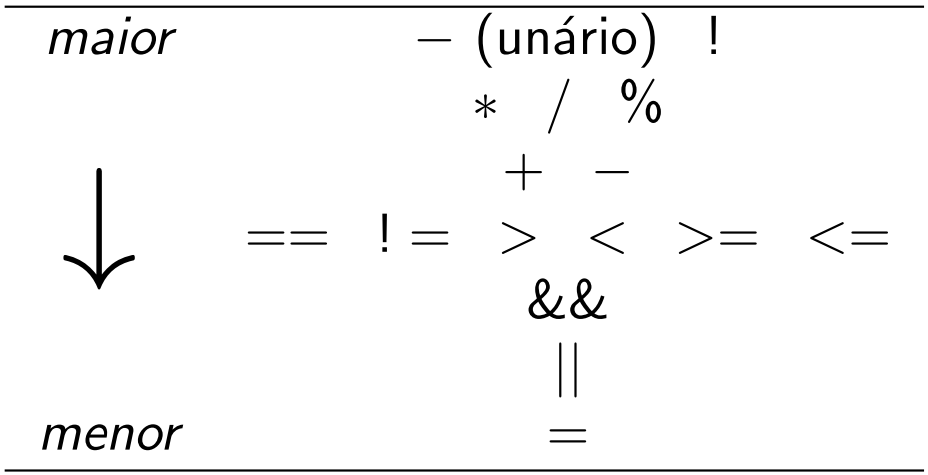
- Precedência:




```
int x = 3;
if (((20-x)>5) &&
    ((4/x) == 1) ||
    ((16-x)>10))
  System.out.
    println("passou");
```

- E a resposta é “passou”

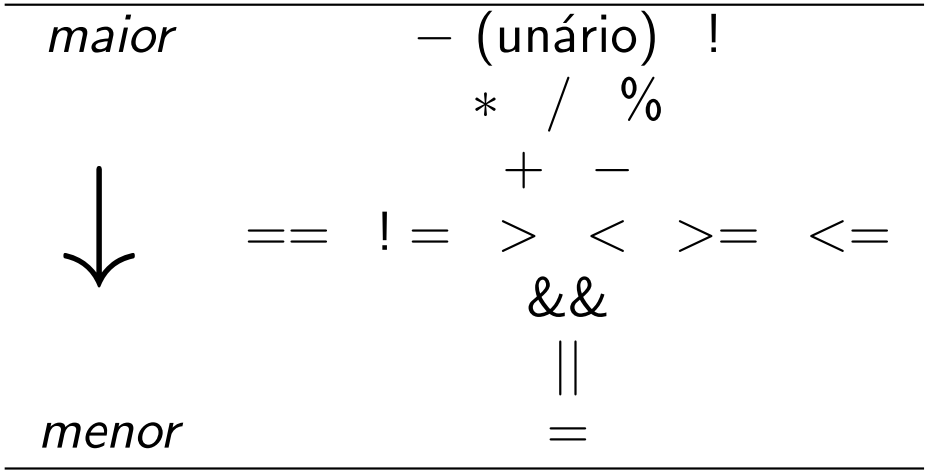
Operadores




```
int x = ?;
if (((20-x)>5) &&
    ((4/x) == 1) ||
    ((16-x)>10))
  System.out.
    println("passou");
```

- $x = 5$

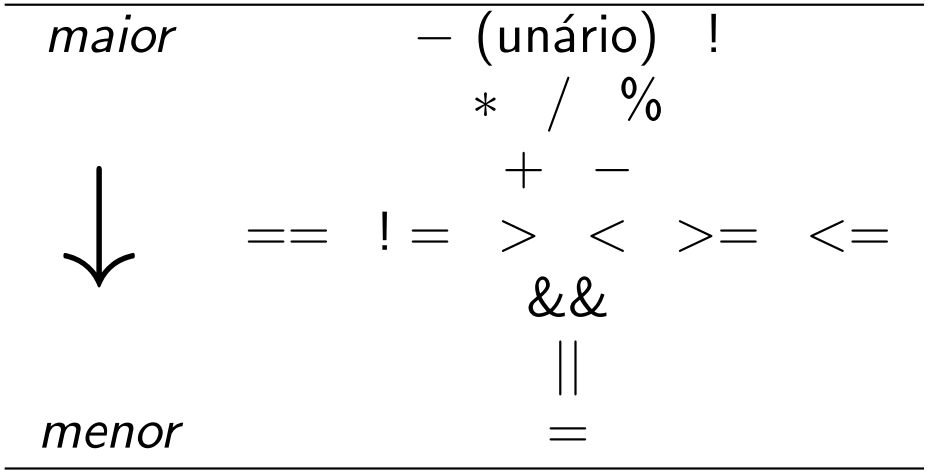
Operadores




```
int x = ?;
if (((20-x)>5) &&
    ((4/x) == 1) ||
    ((16-x)>10))
  System.out.
    println("passou");
```

- $x = 5$
  - V && F || V $\rightarrow$ "passou"

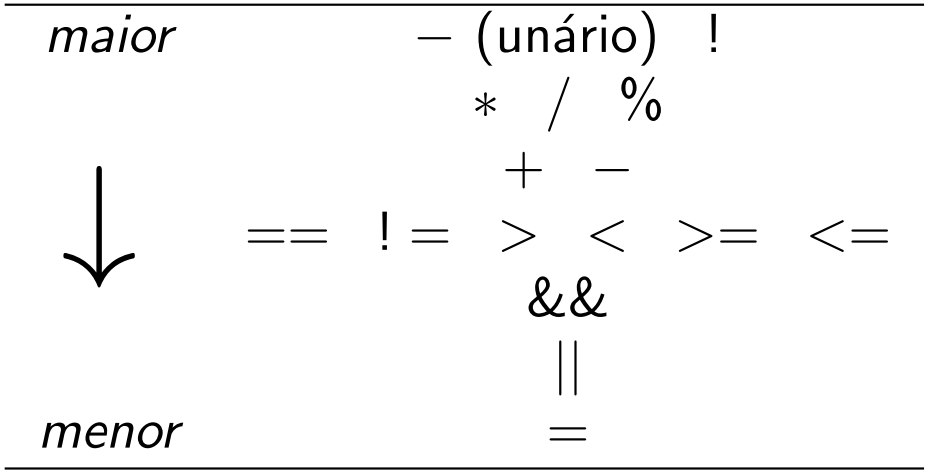
Operadores




```
int x = ?;
if (((20-x)>5) &&
    ((4/x) == 1) ||
    ((16-x)>10))
  System.out.
    println("passou");
```

- $x = 5$
  - V && F || V $\rightarrow$ "passou"
- $x = 6$

Operadores




```
int x = ?;
if (((20-x)>5) &&
    ((4/x) == 1) ||
    ((16-x)>10))
  System.out.
    println("passou");
```

- $x = 5$
  - V && F || V $\rightarrow$ "passou"
- $x = 6$
  - V && F || F $\rightarrow \emptyset$

Atenção!

- Use e abuse de parênteses

Atenção!

- Use e abuse de parênteses
- O que é mais fácil de entender?

Atenção!

- Use e abuse de parênteses
- O que é mais fácil de entender?
  - `20 - x > 5 && 4 / x == 1 || 16 - x > 10`

Atenção!

- Use e abuse de parênteses
- O que é mais fácil de entender?
  - `20 - x > 5 && 4 / x == 1 || 16 - x > 10`
  - `( ( 20 - x ) > 5 ) && ( ( 4 / x ) == 1 ) || ( ( 16 - x ) > 10 )`

Atenção!

- Use e abuse de parênteses
- O que é mais fácil de entender?
  - `20 - x > 5 && 4 / x == 1 || 16 - x > 10`
  - `( ( 20 - x ) > 5 ) && ( ( 4 / x ) == 1 ) || ( ( 16 - x ) > 10 )`
- Ou então, removendo os espaços...
  - `20-x>5&&4/x==1||16-x>10`

Atenção!

- Use e abuse de parênteses
- O que é mais fácil de entender?
  - `20 - x > 5 && 4 / x == 1 || 16 - x > 10`
  - `( ( 20 - x ) > 5 ) && ( ( 4 / x ) == 1 ) || ( ( 16 - x ) > 10 )`
- Ou então, removendo os espaços...
  - `20-x>5&&4/x==1||16-x>10`
  - `((20-x)>5)&&((4/x)==1)||((16-x)>10)`

O operador '?'

- O Java possui um atalho para condicionais:

O operador ‘?’

- O Java possui um atalho para condicionais:
- O operador ‘?’
  - `var = condição ? expressão 1 : expressão 2;`

O operador ‘?’

- O Java possui um atalho para condicionais:
- O operador ‘?’
  - `var = condição ? expressão 1 : expressão 2;`
- Correspondendo a
  - ```
    if (condição) var = expressão 1;
    else var = expressão 2;
    ```
    ```

O operador '?'

- Pode ser usado como substituição a esse tipo de condicional em qualquer parte do código:

```
static double areaPiscina(
                    double raio) {
  double resp;
  if (raio >= 0)
    resp = Math.PI *
                Math.pow(raio,2);
  else resp = -1;
  return(resp);
}
```

```
static double areaPiscina(
                    double raio) {
  double resp;
  resp = (raio >= 0) ?
     Math.PI*Math.pow(raio,2) :
     -1;
  return(resp);
}
```

O operador '?'

- Pode ser usado como substituição a esse tipo de condicional em qualquer parte do código:

```
static double areaPiscina(
                    double raio) {
  if (raio >= 0)
    return Math.PI *
                Math.pow(raio,2);
  else return(-1);
}
```

```
static double areaPiscina(
                      double raio) {
  return((raio >= 0) ?
     Math.PI*Math.pow(raio,2) :
     -1);
}
```

Videoaula

https://www.youtube.com/watch?v=jj35ngukLfg&t=385s