Aula 08 – Laços (cont.)

Norton Trevisan Roman

12 de abril de 2013

Do ... While

- Existe ainda um outro tipo de laço

Do ... While

- Existe ainda um outro tipo de laço
  - O do...while

```
do {
    comando_1;
    comando_2;
    ...
    comando_n;
} while (condição);
```

Do ... While

- Existe ainda um outro tipo de laço
  - O do...while
  - O while faz o teste antes de rodar o laço pela primeira vez

```
do {
    comando_1;
    comando_2;
    ...
    comando_n;
} while (condição);
```

Do ... While

- Existe ainda um outro tipo de laço
  - O do...while
  - O while faz o teste antes de rodar o laço pela primeira vez
  - O do...while faz o teste depois de rodar o laço pela primeira vez

```
do {
    comando_1;
    comando_2;
    ...
    comando_n;
} while (condição);
```

Do ... While

- Existe ainda um outro tipo de laço
  - O do...while
  - O while faz o teste antes de rodar o laço pela primeira vez
  - O do...while faz o teste depois de rodar o laço pela primeira vez
    - ★ Rodando a segunda apenas se o teste for positivo

```
do {
    comando_1;
    comando_2;
    ...
    comando_n;
} while (condição);
```

Do ... While

- Existe ainda um outro tipo de laço
  - O do...while
  - O while faz o teste antes de rodar o laço pela primeira vez
  - O do...while faz o teste depois de rodar o laço pela primeira vez
    - ★ Rodando a segunda apenas se o teste for positivo

```
do {
    comando_1;
    comando_2;
    ...
    comando_n;
} while (condição);
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    do {
        System.out.println(area+"\t"+
              valorPiscina(area,tipo));
        area = area + 50;
    } while (area<=200);
}
```

Do ... While

- Existe ainda um outro tipo de laço
  - O do...while
  - O while faz o teste antes de rodar o laço pela primeira vez
  - O do...while faz o teste depois de rodar o laço pela primeira vez
    - ★ Rodando a segunda apenas se o teste for positivo
- E quando usar um ou o outro?

```
do {
    comando_1;
    comando_2;
    ...
    comando_n;
} while (condição);
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    do {
        System.out.println(area+"\t"+
              valorPiscina(area,tipo));
        area = area + 50;
    } while (area<=200);
}
```

Do ... While

- Existe ainda um outro tipo de laço
  - O do...while
  - O while faz o teste antes de rodar o laço pela primeira vez
  - O do...while faz o teste depois de rodar o laço pela primeira vez
    - ★ Rodando a segunda apenas se o teste for positivo

```
do {
    comando_1;
    comando_2;
    ...
    comando_n;
} while (condição);
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    do {
        System.out.println(area+"\t"+
              valorPiscina(area,tipo));
        area = area + 50;
    } while (area<=200);
}
```

- E quando usar um ou o outro?
  - Depende de quando o teste deve ser feito

Do ... While

- Existe ainda um outro tipo de laço
  - O do...while
  - O while faz o teste antes de rodar o laço pela primeira vez
  - O do...while faz o teste depois de rodar o laço pela primeira vez
    - ★ Rodando a segunda apenas se o teste for positivo

```
do {
    comando_1;
    comando_2;
    ...
    comando_n;
} while (condição);
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    do {
        System.out.println(area+"\t"+
              valorPiscina(area,tipo));
        area = area + 50;
    } while (area<=200);
}
```

- E quando usar um ou o outro?
  - Depende de quando o teste deve ser feito
  - Se antes, ou depois do corpo do laço rodar uma vez

Do ... While

- E qual dentre o while e do...while é melhor?

Do ... While

- E qual dentre o while e do...while é melhor?
  - São totalmente equivalentes

Do ... While

- E qual dentre o while e do...while é melhor?
  - São totalmente equivalentes
  - Todo while, pode ser escrito como do... while:

```
while (condição) {
    comando;
}
```

```
if (condição) {
    do {
        comando;
    } while (condição);
}
```

Do ... While

- E qual dentre o while e do...while é melhor?
  - São totalmente equivalentes
  - Todo while, pode ser escrito como do... while:

```
while (condição) {
    comando;
}
```

```
if (condição) {
    do {
        comando;
    } while (condição);
}
```

  - E vice versa:

```
do {
    comando;
} while (condição);
```

```
condição = true;
while (condição) {
    comando;
    recalcula_condição;
}
```

Do ... While

- E qual dentre o while e do...while é melhor?
  - São totalmente equivalentes
  - Todo while, pode ser escrito como do... while:

```
while (condição) {
    comando;
}
```

```
if (condição) {
    do {
        comando;
    } while (condição);
}
```

  - E vice versa:

```
do {
    comando;
} while (condição);
```

```
condição = true;
while (condição) {
    comando;
    recalcula_condição;
}
```

- O que ditará qual deles será usado será a conveniência para o programador

Do ... While

- E qual dentre o while e do...while é melhor?
  - São totalmente equivalentes
  - Todo while, pode ser escrito como do... while:

```
while (condição) {
    comando;
}
```

```
if (condição) {
    do {
        comando;
    } while (condição);
}
```

  - E vice versa:

```
do {
    comando;
} while (condição);
```

```
condição = true;
while (condição) {
    comando;
    recalcula_condição;
}
```

- O que ditará qual deles será usado será a conveniência para o programador
  - Será escolhido, naturalmente, aquele que exigir a escrita de menos código

Do ... While

- Em nosso código, os seguintes trechos são equivalentes:

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                 valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    if (area <= 200) {
        do {
            System.out.println(area+"\t"+
                  valorPiscina(area,tipo));
            area = area + 50;
        } while (area<=200);
    }
}
```

Do ... While

- Em nosso código, os seguintes trechos são equivalentes:

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                 valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    if (area <= 200) {
        do {
            System.out.println(area+"\t"+
                  valorPiscina(area,tipo));
            area = area + 50;
        } while (area<=200);
    }
}
```

- Mas como o if sempre resultará verdadeiro, nesse código, então:

Do ... While

- Em nosso código, os seguintes trechos são equivalentes:

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    if (area <= 200) {
        do {
            System.out.println(area+"\t"+
                    valorPiscina(area,tipo));
            area = area + 50;
        } while (area<=200);
    }
}
```

- Mas como o if sempre resultará verdadeiro, nesse código, então:

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    do {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
        area = area + 50;
    } while (area<=200);
}
```

Do ... While

- Da mesma forma, os seguintes trechos são equivalentes:

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                        valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    if (tipo <= PLASTICO) {
        do {
         System.out.println(tipo+"\t\t"+
                    valorPiscina(area,tipo));
         tipo = tipo + 1;
        } while (tipo <= PLASTICO);
    }
}
```

Do ... While

- Da mesma forma, os seguintes trechos são equivalentes:

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                        valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    if (tipo <= PLASTICO) {
        do {
         System.out.println(tipo+"\t\t"+
                    valorPiscina(area,tipo));
         tipo = tipo + 1;
        } while (tipo <= PLASTICO);
    }
}
```

- E:

Do ... While

- Da mesma forma, os seguintes trechos são equivalentes:

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                    valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    if (tipo <= PLASTICO) {
        do {
         System.out.println(tipo+"\t\t"+
                    valorPiscina(area,tipo));
         tipo = tipo + 1;
        } while (tipo <= PLASTICO);
    }
}
```

- E:

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                    valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    do {
     System.out.println(tipo+"\t\t"+
                    valorPiscina(area,tipo));
     tipo = tipo + 1;
    } while (tipo <= PLASTICO);
}
```

Do ... While

- E, finalmente...

Do ... While

- E, finalmente... eles também podem ser aninhados:

Do ... While

- E, finalmente... eles também podem ser aninhados:

```
public static void main(String[] args) {
    double area = 50;
    int tipo;
    System.out.println("Área\tMaterial\tValor");
    while (area <= 200) {
        tipo = ALVENARIA;
        while (tipo <= PLASTICO) {
            System.out.println(area+"\t"+tipo+"\t\t"+
                               valorPiscina(area,tipo));
            tipo = tipo+1;
        }
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo;
    System.out.println("Área\tMaterial\tValor");
    do {
        tipo = ALVENARIA;
        do {
            System.out.println(area+"\t"+tipo+"\t\t"+
                               valorPiscina(area,tipo));
            tipo = tipo+1;
        } while (tipo <= PLASTICO);
        area = area+50;
    } while (area <= 200);
}
```

Resumindo

While

```
while (condição) {
    comando;
}
```

Do...While

```
do {
    comando;
} while (condição);
```

Resumindo

While

```
while (condição) {
    comando;
}
```

Do...While

```
do {
    comando;
} while (condição);
```

- Primeiro testa a condição

Resumindo

While

```
while (condição) {
    comando;
}
```

Do...While

```
do {
    comando;
} while (condição);
```

- Primeiro testa a condição
- Se ela for verdadeira, executa o laço (corpo do while)

Resumindo

While

```
while (condição) {
    comando;
}
```

Do...While

```
do {
    comando;
} while (condição);
```

- Primeiro testa a condição
- Se ela for verdadeira, executa o laço (corpo do while)
  - Ao final do corpo, volta ao início, testando novamente a condição

Resumindo

While

```
while (condição) {
    comando;
}
```

Do...While

```
do {
    comando;
} while (condição);
```

- Primeiro testa a condição
- Se ela for verdadeira, executa o laço (corpo do while)
  - Ao final do corpo, volta ao início, testando novamente a condição
- Se ela for falsa, passa à próxima instrução após o while

Resumindo

While

```
while (condição) {
    comando;
}
```

- Primeiro testa a condição
- Se ela for verdadeira, executa o laço (corpo do while)
  - Ao final do corpo, volta ao início, testando novamente a condição
- Se ela for falsa, passa à próxima instrução após o while

Do...While

```
do {
    comando;
} while (condição);
```

- Primeiro executa o laço (corpo do do...while)

Resumindo

While

```
while (condição) {
    comando;
}
```

- Primeiro testa a condição
- Se ela for verdadeira, executa o laço (corpo do while)
  - Ao final do corpo, volta ao início, testando novamente a condição
- Se ela for falsa, passa à próxima instrução após o while

Do...While

```
do {
    comando;
} while (condição);
```

- Primeiro executa o laço (corpo do do...while)
- Ao final do corpo, testa então a condição

Resumindo

While

```
while (condição) {
    comando;
}
```

- Primeiro testa a condição
- Se ela for verdadeira, executa o laço (corpo do while)
  - Ao final do corpo, volta ao início, testando novamente a condição
- Se ela for falsa, passa à próxima instrução após o while

Do...While

```
do {
    comando;
} while (condição);
```

- Primeiro executa o laço (corpo do do...while)
- Ao final do corpo, testa então a condição
- Se ela for verdadeira, executa o laço novamente

Resumindo

While

```
while (condição) {
    comando;
}
```

- Primeiro testa a condição
- Se ela for verdadeira, executa o laço (corpo do while)
  - Ao final do corpo, volta ao início, testando novamente a condição
- Se ela for falsa, passa à próxima instrução após o while

Do...While

```
do {
    comando;
} while (condição);
```

- Primeiro executa o laço (corpo do do...while)
- Ao final do corpo, testa então a condição
- Se ela for verdadeira, executa o laço novamente
- Se ela for falsa, passa à próxima instrução após o do...while

Revendo o While...

- Os while vistos tinham características em comum em suas variáveis de controle (a variável da condição):

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                        valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                  valorPiscina(area,tipo));
        area = area+50;
    }
}
```

Revendo o While...

- Os while vistos tinham características em comum em suas variáveis de controle (a variável da condição):
  - Ambos variavam em passos constantes

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                        valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                 valorPiscina(area,tipo));
        area = area+50;
    }
}
```

Revendo o While...

- Os while vistos tinham características em comum em suas variáveis de controle (a variável da condição):
  - Ambos variavam em passos constantes
    - ★ De 1 em 1

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                  valorPiscina(area,tipo));
        area = area+50;
    }
}
```

Revendo o While...

- Os while vistos tinham características em comum em suas variáveis de controle (a variável da condição):
  - Ambos variavam em passos constantes
    - ★ De 1 em 1
    - ★ De 50 em 50;

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                        valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                 valorPiscina(area,tipo));
        area = area+50;
    }
}
```

Revendo o While...

- Os while vistos tinham características em comum em suas variáveis de controle (a variável da condição):
  - Ambos variavam em passos constantes
    - ★ De 1 em 1
    - ★ De 50 em 50;
  - Não haveria um modo de deixar esse código mais enxuto?

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                        valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                 valorPiscina(area,tipo));
        area = area+50;
    }
}
```

Revendo o While...

- Os while vistos tinham características em comum em suas variáveis de controle (a variável da condição):
  - Ambos variavam em passos constantes
    - ★ De 1 em 1
    - ★ De 50 em 50;
  - Não haveria um modo de deixar esse código mais enxuto?
    - ★ Um modo de dizer "para tipo começando em 0, variando de 1 em 1, até 3, faça..."

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                        valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                 valorPiscina(area,tipo));
        area = area+50;
    }
}
```

Revendo o While...

- Os while vistos tinham características em comum em suas variáveis de controle (a variável da condição):
  - Ambos variavam em passos constantes
    - ★ De 1 em 1
    - ★ De 50 em 50;
  - Não haveria um modo de deixar esse código mais enxuto?
    - ★ Um modo de dizer "para tipo começando em 0, variando de 1 em 1, até 3, faça..."
    - ★ Ou "para area começando em 50, variando de 50 em 50, até 200, faça..."

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                        valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                 valorPiscina(area,tipo));
        area = area+50;
    }
}
```

For

- O laço for:

```
for (inicialização; condição; atualização) {
    comandos
}
```

```
inicialização
while (condição) {
    comandos;
    atualização
}
```

For

- O laço for:

```
for (inicialização; condição; atualização) {
    comandos
}
```

```
inicialização
while (condição) {
    comandos;
    atualização
}
```

- Funcionamento:

For

- O laço for:

```
for (inicialização; condição; atualização) {
    comandos
}
```

```
inicialização
while (condição) {
    comandos;
    atualização
}
```

- Funcionamento:
  - Primeiro, há a inicialização das variáveis de controle

For

- O laço for:

```
for (inicialização; condição; atualização) {
    comandos
}
```

```
inicialização
while (condição) {
    comandos;
    atualização
}
```

- Funcionamento:
  - Primeiro, há a inicialização das variáveis de controle
    - ★ Esse passo é executado uma única vez

For

- O laço for:

```
for (inicialização; condição; atualização) {
    comandos
}
```

```
inicialização
while (condição) {
    comandos;
    atualização
}
```

- Funcionamento:
  - Primeiro, há a inicialização das variáveis de controle
    - ★ Esse passo é executado uma única vez
  - Em seguida, a condição é testada

For

- O laço for:

```
for (inicialização; condição; atualização) {
    comandos
}
```

```
inicialização
while (condição) {
    comandos;
    atualização
}
```

- Funcionamento:
  - Primeiro, há a inicialização das variáveis de controle
    - ★ Esse passo é executado uma única vez
  - Em seguida, a condição é testada
  - Se resultar verdadeira, os comandos do corpo do for são executados

For

- O laço for:

```
for (inicialização; condição; atualização) {
    comandos
}
```

```
inicialização
while (condição) {
    comandos;
    atualização
}
```

- Funcionamento:
  - Primeiro, há a inicialização das variáveis de controle
    - ★ Esse passo é executado uma única vez
  - Em seguida, a condição é testada
  - Se resultar verdadeira, os comandos do corpo do for são executados
    - ★ Ao final do corpo, é executada a atualização

For

- O laço for:

```
for (inicialização; condição; atualização) {
    comandos
}
```

```
inicialização
while (condição) {
    comandos;
    atualização
}
```

- Funcionamento:
  - Primeiro, há a inicialização das variáveis de controle
    - ★ Esse passo é executado uma única vez
  - Em seguida, a condição é testada
  - Se resultar verdadeira, os comandos do corpo do for são executados
    - ★ Ao final do corpo, é executada a atualização
    - ★ Inicia-se o laço novamente, voltando ao teste da condição

For

- O laço for:

```
for (inicialização; condição; atualização) {
    comandos
}
```

```
inicialização
while (condição) {
    comandos;
    atualização
}
```

- Funcionamento:
  - Primeiro, há a inicialização das variáveis de controle
    - ★ Esse passo é executado uma única vez
  - Em seguida, a condição é testada
  - Se resultar verdadeira, os comandos do corpo do for são executados
    - ★ Ao final do corpo, é executada a atualização
    - ★ Inicia-se o laço novamente, voltando ao teste da condição
  - Se falsa, o corpo é ignorado

For

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 100;

    System.out.println("Material\tValor");
    for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                  tipo = tipo+1) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
    }
}
```

For

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 100;

    System.out.println("Material\tValor");
    for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                  tipo = tipo+1) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
    }
}
```

- Alguns detalhes...

For

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 100;

    System.out.println("Material\tValor");
    for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                  tipo = tipo+1) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
    }
}
```

- Alguns detalhes...
  - É possível declarar a variável de controle dentro do for

For

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 100;

    System.out.println("Material\tValor");
    for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                  tipo = tipo+1) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
    }
}
```

- Alguns detalhes...
  - É possível declarar a variável de controle dentro do for
    - ★ Nesse caso, seu escopo é o corpo do for

For

```
public static void main(String[] args) {
    double area = 100;
    int tipo = ALVENARIA;

    System.out.println("Material\tValor");
    while (tipo <= PLASTICO) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
        tipo = tipo+1;
    }
}
```

```
public static void main(String[] args) {
    double area = 100;

    System.out.println("Material\tValor");
    for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                  tipo = tipo+1) {
        System.out.println(tipo+"\t\t"+
                           valorPiscina(area,tipo));
    }
}
```

- Alguns detalhes...
  - É possível declarar a variável de controle dentro do for
    - ★ Nesse caso, seu escopo é o corpo do for
    - ★ Nada a impede de ser declarada antes, para ser visível a mais alguém:

```
int tipo;
for(tipo = ALVENARIA; tipo <= PLASTICO; tipo = tipo+1) {
    System.out.println(tipo+"\\t\\t"+valorPiscina(area,tipo));
}
```

For

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 50; area <= 200;
                          area = area+50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

- Incremento de um em um não é só o que o for é capaz de fazer:

For

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 50; area <= 200;
                        area = area+50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

- Incremento de um em um não é só o que o for é capaz de fazer:
  - Qualquer expressão algébrica pode ser usada

For

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 50; area <= 200;
                        area = area+50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

- Incremento de um em um não é só o que o for é capaz de fazer:
  - Qualquer expressão algébrica pode ser usada
    - ★ Até mesmo coisas como `area = 2*area + Math.pow(area,3)`

For

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 50; area <= 200;
                          area = area+50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

- Incremento de um em um não é só o que o for é capaz de fazer:
  - Qualquer expressão algébrica pode ser usada
    - ★ Até mesmo coisas como `area = 2*area + Math.pow(area,3)`
  - E não é apenas o int que pode ser usado como variável de controle

For

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 50; area <= 200;
                          area = area+50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

- Incremento de um em um não é só o que o for é capaz de fazer:
  - Qualquer expressão algébrica pode ser usada
    - ★ Até mesmo coisas como `area = 2*area + Math.pow(area,3)`
  - E não é apenas o int que pode ser usado como variável de controle
    - ★ Podemos também usar outros tipos

For

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 50; area <= 200;
                          area = area+50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

- Incremento de um em um não é só o que o for é capaz de fazer:
  - Qualquer expressão algébrica pode ser usada
    - Até mesmo coisas como `area = 2*area + Math.pow(area,3)`
  - E não é apenas o int que pode ser usado como variável de controle
    - Podemos também usar outros tipos
- Da mesma forma, na condição qualquer expressão lógica ou relacional pode ser usada:

For

```
public static void main(String[] args) {
    double area = 50;
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    while (area <= 200) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 50; area <= 200;
                          area = area+50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

- Incremento de um em um não é só o que o for é capaz de fazer:
  - Qualquer expressão algébrica pode ser usada
    - Até mesmo coisas como `area = 2*area + Math.pow(area,3)`
  - E não é apenas o int que pode ser usado como variável de controle
    - Podemos também usar outros tipos
- Da mesma forma, na condição qualquer expressão lógica ou relacional pode ser usada:
  - Ex: `(true && area <= 200) || (area == 300)`

For

- Já que qualquer expressão pode ser usada na atualização, nada nos impede de fazer um decremento

For

- Já que qualquer expressão pode ser usada na atualização, nada nos impede de fazer um decremento

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 50; area <= 200;
                        area = area+50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 200; area >= 50;
                            area = area-50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

For

- Já que qualquer expressão pode ser usada na atualização, nada nos impede de fazer um decremento

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 50; area <= 200;
                        area = area+50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 200; area >= 50;
                            area = area-50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

- E o resultado seria apenas a inversão da tabela

For

- Já que qualquer expressão pode ser usada na atualização, nada nos impede de fazer um decremento

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 50; area <= 200;
                        area = area+50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

```
public static void main(String[] args) {
    int tipo = ALVENARIA;

    System.out.println("Área\tValor");
    for(double area = 200; area >= 50;
                        area = area-50) {
        System.out.println(area+"\t"+
                valorPiscina(area,tipo));
    }
}
```

- E o resultado seria apenas a inversão da tabela

```
Área Valor
50.0 75000.0
100.0 150000.0
150.0 225000.0
200.0 300000.0
```

```
Área Valor
200.0 300000.0
150.0 225000.0
100.0 150000.0
50.0 75000.0
```

For Aninhado

- Laços for podem também ser aninhados

```
public static void main(String[] args) {
    double area = 50;
    int tipo;
    System.out.println("Área\tMaterial\tValor");
    while (area <= 200) {
        tipo = ALVENARIA;
        while (tipo <= PLASTICO) {
            System.out.println(area+"\t"+tipo+"\t\t"+
                               valorPiscina(area,tipo));
            tipo = tipo+1;
        }
        area = area+50;
    }
}
```

For Aninhado

- Laços for podem também ser aninhados

```
public static void main(String[] args) {
    double area = 50;
    int tipo;
    System.out.println("Área\tMaterial\tValor");
    while (area <= 200) {
        tipo = ALVENARIA;
        while (tipo <= PLASTICO) {
            System.out.println(area+"\t"+tipo+"\t\t"+
                               valorPiscina(area,tipo));
            tipo = tipo+1;
        }
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    System.out.println("Área\tMaterial\tValor");
    for(double area = 50; area <= 200;
                                  area = area+50) {
        for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                   tipo = tipo+1) {
            System.out.println(area+"\t"+tipo+"\t\t"
                               +valorPiscina(area,tipo));
        }
    }
}
```

For Aninhado

- Laços for podem também ser aninhados

```
public static void main(String[] args) {
    double area = 50;
    int tipo;
    System.out.println("Área\tMaterial\tValor");
    while (area <= 200) {
        tipo = ALVENARIA;
        while (tipo <= PLASTICO) {
            System.out.println(area+"\t"+tipo+"\t\t"+
                               valorPiscina(area,tipo));
            tipo = tipo+1;
        }
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    System.out.println("Área\tMaterial\tValor");
    for(double area = 50; area <= 200;
                                    area = area+50) {
        for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                     tipo = tipo+1) {
            System.out.println(area+"\t"+tipo+"\t\t"
                               +valorPiscina(area,tipo));
        }
    }
}
```

For Aninhado

- Laços for podem também ser aninhados

```
public static void main(String[] args) {
    double area = 50;
    int tipo;
    System.out.println("Área\tMaterial\tValor");
    while (area <= 200) {
        tipo = ALVENARIA;
        while (tipo <= PLASTICO) {
            System.out.println(area+"\t"+tipo+"\t\t"+
                               valorPiscina(area,tipo));
            tipo = tipo+1;
        }
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    System.out.println("Área\tMaterial\tValor");
    for(double area = 50; area <= 200;
                                    area = area+50) {
        for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                     tipo = tipo+1) {
            System.out.println(area+"\t"+tipo+"\t\t"
                               +valorPiscina(area,tipo));
        }
    }
}
```

For Aninhado

- Laços for podem também ser aninhados

```
public static void main(String[] args) {
    double area = 50;
    int tipo;
    System.out.println("Área\tMaterial\tValor");
    while (area <= 200) {
        tipo = ALVENARIA;
        while (tipo <= PLASTICO) {
            System.out.println(area+"\t"+tipo+"\t\t"+
                               valorPiscina(area,tipo));
            tipo = tipo+1;
        }
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    System.out.println("Área\tMaterial\tValor");
    for(double area = 50; area <= 200;
                                    area = area+50) {
        for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                     tipo = tipo+1) {
            System.out.println(area+"\t"+tipo+"\t\t"
                               +valorPiscina(area,tipo));
        }
    }
}
```

- Ficam até mais fáceis de serem entendidos

For Aninhado

- Laços for podem também ser aninhados

```
public static void main(String[] args) {
    double area = 50;
    int tipo;
    System.out.println("Área\tMaterial\tValor");
    while (area <= 200) {
        tipo = ALVENARIA;
        while (tipo <= PLASTICO) {
            System.out.println(area+"\t"+tipo+"\t\t"+
                               valorPiscina(area,tipo));
            tipo = tipo+1;
        }
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    System.out.println("Área\tMaterial\tValor");
    for(double area = 50; area <= 200;
                                    area = area+50) {
        for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                     tipo = tipo+1) {
            System.out.println(area+"\t"+tipo+"\t\t"
                               +valorPiscina(area,tipo));
        }
    }
}
```

- Ficam até mais fáceis de serem entendidos
- E qual laço é o melhor?

For Aninhado

- Laços for podem também ser aninhados

```
public static void main(String[] args) {
    double area = 50;
    int tipo;
    System.out.println("Área\tMaterial\tValor");
    while (area <= 200) {
        tipo = ALVENARIA;
        while (tipo <= PLASTICO) {
            System.out.println(area+"\t"+tipo+"\t\t"+
                               valorPiscina(area,tipo));
            tipo = tipo+1;
        }
        area = area+50;
    }
}
```

```
public static void main(String[] args) {
    System.out.println("Área\tMaterial\tValor");
    for(double area = 50; area <= 200;
                                    area = area+50) {
        for(int tipo = ALVENARIA; tipo <= PLASTICO;
                                     tipo = tipo+1) {
            System.out.println(area+"\t"+tipo+"\t\t"
                               +valorPiscina(area,tipo));
        }
    }
}
```

- Ficam até mais fáceis de serem entendidos
- E qual laço é o melhor?
  - São equivalentes... depende da conveniência do programador

For – Múltiplos valores

- Embora a condição tenha que ser <u>única</u>

```
int a;
int b;
for(???; a<b; ???) {
    System.out.println("a = " + a);
    System.out.println("b = " + b);
}
```

For – Múltiplos valores

- Embora a condição tenha que ser <u>única</u>
  - Aceita múltiplas inicializações

```
int a;
int b;
for(a=1, b=4; a<b; ???) {
    System.out.println("a = " + a);
    System.out.println("b = " + b);
}
```

For – Múltiplos valores

- Embora a condição tenha que ser <u>única</u>
  - Aceita múltiplas inicializações
    - ★ Separadas por vírgula

```
int a;
int b;
for(a=1, b=4; a<b; ???) {
    System.out.println("a = " + a);
    System.out.println("b = " + b);
}
```

For – Múltiplos valores

- Embora a condição tenha que ser única
  - Aceita múltiplas inicializações
    - ★ Separadas por vírgula
    - ★ Declaradas fora do for

```
int a;
int b;
for(a=1, b=4; a<b; ???) {
    System.out.println("a = " + a);
    System.out.println("b = " + b);
}
```

For – Múltiplos valores

- Embora a condição tenha que ser única
  - Aceita múltiplas inicializações
    - ★ Separadas por vírgula
    - ★ Declaradas fora do for
  - Aceita múltiplas atualizações

```
int a;
int b;
for(a=1, b=4; a<b; a++, b--) {
    System.out.println("a = " + a);
    System.out.println("b = " + b);
}
```

For – Múltiplos valores

- Embora a condição tenha que ser única
  - Aceita múltiplas inicializações
    - ★ Separadas por vírgula
    - ★ Declaradas fora do for
  - Aceita múltiplas atualizações
    - ★ Também separadas por vírgula

```
int a;
int b;
for(a=1, b=4; a<b; a++, b--) {
    System.out.println("a = " + a);
    System.out.println("b = " + b);
}
```

For – Múltiplos valores

- Embora a condição tenha que ser única
  - Aceita múltiplas inicializações
    - ★ Separadas por vírgula
    - ★ Declaradas fora do for
  - Aceita múltiplas atualizações
    - ★ Também separadas por vírgula
- a++? b--?

```
int a;
int b;
for(a=1, b=4; a<b; a++, b--) {
    System.out.println("a = " + a);
    System.out.println("b = " + b);
}
```

For – Múltiplos valores

- Embora a condição tenha que ser <u>única</u>
  - Aceita múltiplas inicializações
    - ★ Separadas por vírgula
    - ★ Declaradas <u>fora</u> do for
  - Aceita múltiplas atualizações
    - ★ Também separadas por vírgula
- a++? b--?
  - São "atalhos" – Expressões contraídas

```
int a;
int b;
for(a=1, b=4; a<b; a++, b--) {
    System.out.println("a = " + a);
    System.out.println("b = " + b);
}
```

Expressão	*Contraída*
x = x + 5	x += 5
x = x - 5	x -= 5
x = x * 5	x *= 5
x = x / 5	x /= 5

For – Múltiplos valores

- Embora a condição tenha que ser única
  - Aceita múltiplas inicializações
    - ★ Separadas por vírgula
    - ★ Declaradas fora do for
  - Aceita múltiplas atualizações
    - ★ Também separadas por vírgula
- a++? b--?
  - São "atalhos" – Expressões contraídas
- Úteis para realizar operação e armazenar resultado na mesma variável

```
int a;
int b;
for(a=1, b=4; a<b; a++, b--) {
    System.out.println("a = " + a);
    System.out.println("b = " + b);
}
```

Expressão	*Contraída*
x = x + 5	x += 5
x = x - 5	x -= 5
x = x * 5	x *= 5
x = x / 5	x /= 5

Expressões Contraídas

- E o ++?

Expressões Contraídas

- E o ++?
  - Tem duas formas: x++ ou ++x

Expressões Contraídas

- E o ++?
  - Tem duas formas: x++ ou ++x
  - Usados isoladamente, tanto ++x quanto x++ correspondem a x = x+1

Expressões Contraídas

- E o ++?
  - Tem duas formas: x++ ou ++x
  - Usados isoladamente, tanto ++x quanto x++ correspondem a x = x+1

Código

```
int x = 2;
int y = 2;
x++;
++y;
System.out.println("x = "+x+", y = "+y);
```

Saída

```
x = 3, y = 3
```

Expressões Contraídas

- E o ++?
  - Tem duas formas: x++ ou ++x
  - Usados isoladamente, tanto ++x quanto x++ correspondem a x = x+1

Código

```
int x = 2;
int y = 2;
x++;
++y;
System.out.println("x = "+x+", y = "+y);
```

Saída

```
x = 3, y = 3
```

  - Mas coisas acontecem quando usados em conjunto com outros comandos:

Expressões Contraídas

- E o ++?
  - Tem duas formas: x++ ou ++x
  - Usados isoladamente, tanto ++x quanto x++ correspondem a x = x+1

Código

```
int x = 2;
int y = 2;
x++;
++y;
System.out.println("x = "+x+", y = "+y);
```

Saída

```
x = 3, y = 3
```

  - Mas coisas acontecem quando usados em conjunto com outros comandos:

Código

```
int x = 2;
int y = 2;
System.out.println("x = "+ x++ +", y = "+ ++y);
System.out.println("x = "+ x +", y = "+ y);
```

Saída

```
x = 2, y = 3
x = 3, y = 3
```

Expressões Contraídas

- O que houve?

Código

```
int x = 2;
int y = 2;
System.out.println("x = "+ x++ +", y = "+ ++y);
System.out.println("x = "+ x +", y = "+ y);
```

Saída

```
x = 2, y = 3
x = 3, y = 3
```

Expressões Contraídas

- O que houve?

Código

```
int x = 2;
int y = 2;
System.out.println("x = "+ x++ +", y = "+ ++y);
System.out.println("x = "+ x +", y = "+ y);
```

Saída

```
x = 2, y = 3
x = 3, y = 3
```

- x++ é um pós-incremento

Expressões Contraídas

- O que houve?

Código

```
int x = 2;
int y = 2;
System.out.println("x = "+ x++ +", y = "+ ++y);
System.out.println("x = "+ x +", y = "+ y);
```

Saída

```
x = 2, y = 3
x = 3, y = 3
```

- x++ é um pós-incremento
- Diz que o compilador deve usar o valor que está em x e só então incrementá-lo

Expressões Contraídas

- O que houve?

Código

```
int x = 2;
int y = 2;
System.out.println("x = "+ x++ +", y = "+ ++y);
System.out.println("x = "+ x +", y = "+ y);
```

Saída

```
x = 2, y = 3
x = 3, y = 3
```

- x++ é um pós-incremento
- Diz que o compilador deve usar o valor que está em x e só então incrementá-lo
- ++x é um pré-incremento

Expressões Contraídas

- O que houve?

Código

```
int x = 2;
int y = 2;
System.out.println("x = "+ x++ +", y = "+ ++y);
System.out.println("x = "+ x +", y = "+ y);
```

Saída

```
x = 2, y = 3
x = 3, y = 3
```

- x++ é um pós-incremento
- Diz que o compilador deve usar o valor que está em x e só então incrementá-lo

- ++x é um pré-incremento
- Diz que o compilador deve primeiro incrementar o valor de x, e só então usá-lo

Expressões Contraídas

- O que houve?

Código

```
int x = 2;
int y = 2;
System.out.println("x = "+ x++ +", y = "+ ++y);
System.out.println("x = "+ x +", y = "+ y);
```

Saída

```
x = 2, y = 3
x = 3, y = 3
```

- x++ é um pós-incremento
- Diz que o compilador deve usar o valor que está em x e só então incrementá-lo

- ++x é um pré-incremento
- Diz que o compilador deve primeiro incrementar o valor de x, e só então usá-lo

- De forma semelhante ao ++, o -- decrementa, em vez de incrementar

Expressões Contraídas

- O que houve?

Código

```
int x = 2;
int y = 2;
System.out.println("x = "+ x++ +", y = "+ ++y);
System.out.println("x = "+ x +", y = "+ y);
```

Saída

```
x = 2, y = 3
x = 3, y = 3
```

- x++ é um pós-incremento
- Diz que o compilador deve usar o valor que está em x e só então incrementá-lo

- ++x é um pré-incremento
- Diz que o compilador deve primeiro incrementar o valor de x, e só então usá-lo

- De forma semelhante ao ++, o -- decrementa, em vez de incrementar
  - Também em suas duas formas: x-- e --x

Expressões Contraídas

- Mais exemplos:

Código

```
int x = 2;
int y = x++;
System.out.println("x = "+ x +", y = "+ y);
int z = ++x;
System.out.println("x = "+ x +", z = "+ z);
```

Expressões Contraídas

- Mais exemplos:

Código

```
int x = 2;
int y = x++;
System.out.println("x = "+ x +", y = "+ y);
int z = ++x;
System.out.println("x = "+ x +", z = "+ z);
```

Saída

```
x = 3, y = 2
x = 4, z = 4
```

Expressões Contraídas

- Mais exemplos:

Código

```
int x = 2;
int y = x++;
System.out.println("x = "+ x +", y = "+ y);
int z = ++x;
System.out.println("x = "+ x +", z = "+ z);
```

Saída

```
x = 3, y = 2
x = 4, z = 4
```

- y = x++ fará y conter 2, se x contiver 2 antes do ++

Expressões Contraídas

- Mais exemplos:

Código

```
int x = 2;
int y = x++;
System.out.println("x = "+ x +", y = "+ y);
int z = ++x;
System.out.println("x = "+ x +", z = "+ z);
```

Saída

```
x = 3, y = 2
x = 4, z = 4
```

- y = x++ fará y conter 2, se x contiver 2 antes do ++
- z = ++x fará z conter 4, se x contiver 3 antes do ++

Expressões Contraídas

- Mais exemplos:

Código

```
int x = 2;
int y = x++;
System.out.println("x = "+ x +", y = "+ y);
int z = ++x;
System.out.println("x = "+ x +", z = "+ z);
```

Saída

```
x = 3, y = 2
x = 4, z = 4
```

- y = x++ fará y conter 2, se x contiver 2 antes do ++
- z = ++x fará z conter 4, se x contiver 3 antes do ++

Código

```
int x = 1;
int y = x++ + 4;
System.out.println("x = "+ x +", y = "+ y);
int z = ++x + 4;
System.out.println("x = "+ x +", z = "+ z);
```

Expressões Contraídas

- Mais exemplos:

Código

```
int x = 2;
int y = x++;
System.out.println("x = "+ x +", y = "+ y);
int z = ++x;
System.out.println("x = "+ x +", z = "+ z);
```

Saída

```
x = 3, y = 2
x = 4, z = 4
```

- y = x++ fará y conter 2, se x contiver 2 antes do ++
- z = ++x fará z conter 4, se x contiver 3 antes do ++

Código

```
int x = 1;
int y = x++ + 4;
System.out.println("x = "+ x +", y = "+ y);
int z = ++x + 4;
System.out.println("x = "+ x +", z = "+ z);
```

Saída

```
x = 2, y = 5
x = 3, z = 7
```

Laços

- Considere o código ao lado:

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5; x++) {
        System.out.print(x+" ");
    }
}
```

Laços

- Considere o código ao lado:
- O que será impresso?

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5; x++) {
        System.out.print(x+" ");
    }
}
```

Laços

- Considere o código ao lado:
- O que será impresso?
  - 1 2 3 4

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5; x++) {
        System.out.print(x+" ");
    }
}
```

Laços

- Considere o código ao lado:
- O que será impresso?
  - 1 2 3 4
- A inicialização em um laço for é opcional.

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5; x++) {
        System.out.print(x+" ");
    }
}
```

Laços

- Considere o código ao lado:
- O que será impresso?
  - 1 2 3 4
- A inicialização em um laço for é opcional.

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5; x++) {
        System.out.print(x+" ");
    }
}
```

- Considere agora esse código:

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5;) {
        System.out.print(x+" ");
    }
}
```

Laços

- Considere o código ao lado:
- O que será impresso?
  - 1 2 3 4
- A inicialização em um laço for é opcional.

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5; x++) {
        System.out.print(x+" ");
    }
}
```

- Considere agora esse código:
- O que será impresso?

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5;) {
        System.out.print(x+" ");
    }
}
```

Laços

- Considere o código ao lado:
- O que será impresso?
  - `1 2 3 4`
- A inicialização em um laço for é opcional.

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5; x++) {
        System.out.print(x+" ");
    }
}
```

- Considere agora esse código:
- O que será impresso?
  - `1 1 1 1 1 1 1 1 1...`

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5;) {
        System.out.print(x+" ");
    }
}
```

Laços

- Considere o código ao lado:
- O que será impresso?
  - `1 2 3 4`
- A inicialização em um laço for é opcional.

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5; x++) {
        System.out.print(x+" ");
    }
}
```

- Considere agora esse código:
- O que será impresso?
  - `1 1 1 1 1 1 1 1 1...`
  - Laço infinito! ...

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5;) {
        System.out.print(x+" ");
    }
}
```

Laços

- Considere o código ao lado:
- O que será impresso?
  - `1 2 3 4`
- A inicialização em um laço for é opcional.

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5; x++) {
        System.out.print(x+" ");
    }
}
```

- Considere agora esse código:
- O que será impresso?
  - `1 1 1 1 1 1 1 1 1...`
  - Laço infinito! ... A condição de parada nunca é satisfeita

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5;) {
        System.out.print(x+" ");
    }
}
```

Laços

- Considere o código ao lado:
- O que será impresso?
  - `1 2 3 4`
- A inicialização em um laço for é opcional.

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5; x++) {
        System.out.print(x+" ");
    }
}
```

- Considere agora esse código:
- O que será impresso?
  - `1 1 1 1 1 1 1 1 1...`
  - Laço infinito! ... A condição de parada nunca é satisfeita
- Também a atualização da variável de controle é opcional.

```
public static void main(String[] args) {
    int x = 1;
    for (; x<5;) {
        System.out.print(x+" ");
    }
}
```

Laços

- E esse código?

```
public static void main(String[] args) {
    for (int x=1;;x++) {
        System.out.print(x+" ");
    }
}
```

Laços

- E esse código?
  - 1 2 3 4 5 6 7 8...

```
public static void main(String[] args) {
    for (int x=1;;x++) {
        System.out.print(x+" ");
    }
}
```

Laços

- E esse código?
  - 1 2 3 4 5 6 7 8...
  - De novo! ...

```
public static void main(String[] args) {
    for (int x=1;;x++) {
        System.out.print(x+" ");
    }
}
```

Laços

- E esse código?
  - `1 2 3 4 5 6 7 8...`
  - De novo! ... Ninguém disse ao laço o que testar para parar

```
public static void main(String[] args) {
    for (int x=1;;x++) {
        System.out.print(x+" ");
    }
}
```

Laços

- E esse código?
  - `1 2 3 4 5 6 7 8...`
  - De novo! ... Ninguém disse ao laço o que testar para parar

```
public static void main(String[] args) {
    for (int x=1;;x++) {
        System.out.print(x+" ");
    }
}
```

- A condição de parada em um laço for também é opcional.

Laços

- E esse código?
  - `1 2 3 4 5 6 7 8...`
  - De novo! ... Ninguém disse ao laço o que testar para parar

```
public static void main(String[] args) {
    for (int x=1;;x++) {
        System.out.print(x+" ");
    }
}
```

- A condição de parada em um laço for também é opcional.
- Em suma:

Laços

- E esse código?
  - `1 2 3 4 5 6 7 8...`
  - De novo! ... Ninguém disse ao laço o que testar para parar

```
public static void main(String[] args) {
    for (int x=1;;x++) {
        System.out.print(x+" ");
    }
}
```

- A condição de parada em um laço for também é opcional.
- Em suma:
  - Inicialização, condição e atualização são opcionais

Laços

- E esse código?
  - `1 2 3 4 5 6 7 8...`
  - De novo! ... Ninguém disse ao laço o que testar para parar

```
public static void main(String[] args) {
    for (int x=1;;x++) {
        System.out.print(x+" ");
    }
}
```

- A condição de parada em um laço for também é opcional.
- Em suma:
  - Inicialização, condição e atualização são opcionais
  - A condição aceita qualquer coisa que resulte em verdadeiro ou falso (expressões lógicas e relacionais)

Laços

- E esse código?
  - `1 2 3 4 5 6 7 8...`
  - De novo! ... Ninguém disse ao laço o que testar para parar

```
public static void main(String[] args) {
    for (int x=1;;x++) {
        System.out.print(x+" ");
    }
}
```

- A condição de parada em um laço for também é opcional.
- Em suma:
  - Inicialização, condição e atualização são opcionais
  - A condição aceita qualquer coisa que resulte em verdadeiro ou falso (expressões lógicas e relacionais)
  - Inicialização e atualização são apenas códigos rodados, respectivamente no inicio e fim do laço