



# Processamento da Informação

## Estruturas de repetição: “while”

Prof. Jesús P. Mena-Chalco  
CMCC/UFABC

Q2/2018

# Estrutura de repetição: laço

Usado em situações em que é necessário repetir um determinado trecho de um programa, geralmente, um determinado número de vezes.

## Duas formas:

- Escrever o trecho quantas vezes for necessário, ou
- Utilizar o conceito de **Laços**.



```
1 public class HelloWorld
2 {
3     public static void main(String []args) {
4         int n=1;
5
6         while (n<=10) {
7             System.out.println("Alo mundo!");
8         }
9     }
10 }
11
```

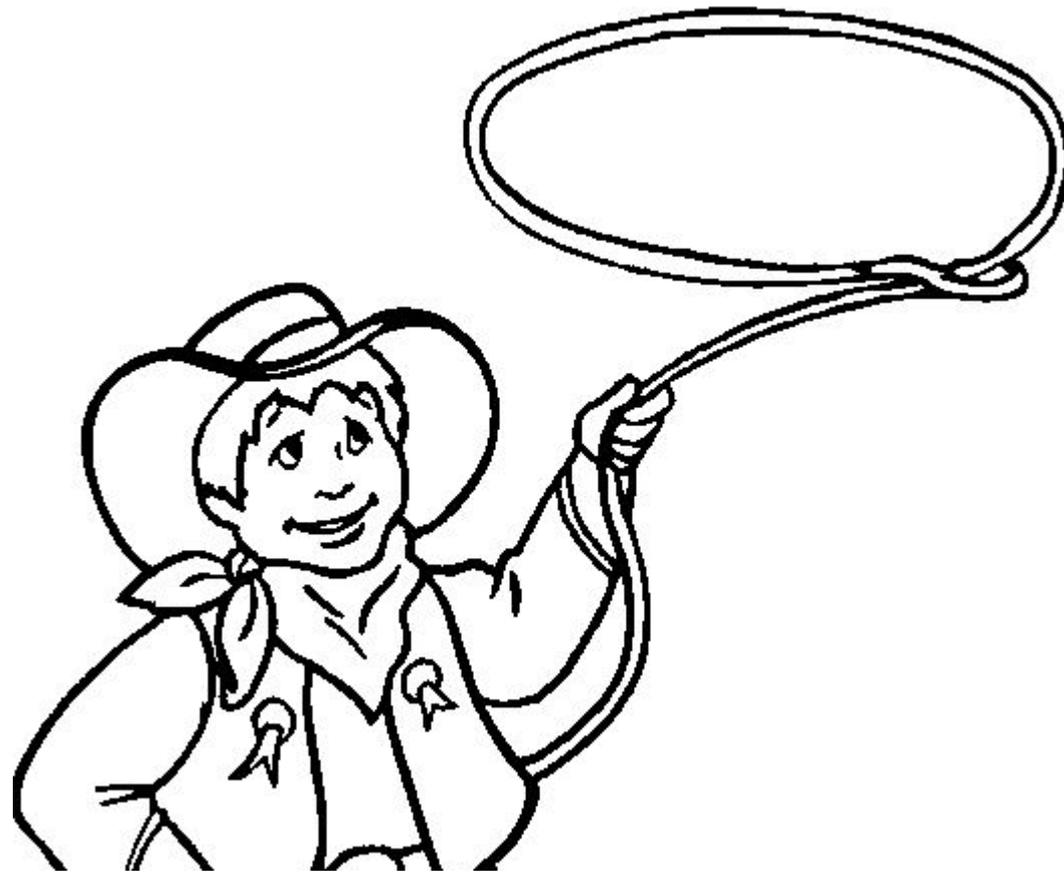


```
1 public class HelloWorld
2 {
3     public static void main(String []args) {
4         int n=1;
5
6         while (n<=10) {
7             System.out.println("Alo mundo!");
8             n = n+1;
9         }
10    }
11 }
```

# Estrutura de repetição: laço while

```
while (...) {  
    ...  
    ...  
    ...  
}
```

# Estrutura de repetição: laço



# Exercício 01: Contagem regressiva

```
1 public class TesteLaco01
2 {
3     public static void main(String []args) {
4
5         int n = 5;
6
7         while (n>0) {
8             System.out.println("Numero "+n);
9             n = n-1;
10        }
11
12        System.out.println("Boom!");
13    }
14 }
```

# Exercício 01: Contagem regressiva

```
Numero 5  
Numero 4  
Numero 3  
Numero 2  
Numero 1  
Boom!
```

# Exercício 02: Somatória de números inteiros

Soma de números inteiros no intervalo [a, b]

```
static int soma(int a, int b) {  
    int soma = 0;  
    int n = a;  
  
    while (n<=b){  
        soma = soma + n;  
        n = n+1;  
    }  
    return soma;  
}
```

# Exercício 02: Somatória de números inteiros

```
1 public class TesteLaco02
2 {
3     static int soma(int a, int b) {
4         int soma = 0;
5         int n = a;
6
7         while (n<=b){
8             soma = soma + n;
9             n = n+1;
10        }
11        return soma;
12    }
13
14    public static void main(String []args) {
15        System.out.println( soma(0,10) );
16        System.out.println( soma(-10,10) );
17        System.out.println( soma(10,0) );
18    }
19 }
```

# Exercício 02: Somatória de números inteiros

```
55  
0  
0
```

# Exercício 03: Somatória de números

Crie um método, em que dado um inteiro  $n > 0$ , permita somar a seguinte sequência:

$$1^2 + 2^2 + 3^2 + \dots + n^2$$

**Assinatura:** `static int somaP(int n)`

# Exercício 03: Somatória de números

```
static int somaP(int n) {  
    int soma = 0;  
    int contador = 1;  
  
    while (contador<=n){  
        soma = soma + contador*contador;  
        contador = contador+1;  
    }  
    return soma;  
}
```

# Exercício 03: Somatória de números

```
1 public class TesteLaco03
2 {
3     static int somaP(int n) {
4         int soma = 0;
5         int contador = 1;
6
7         while (contador<=n){
8             soma = soma + contador*contador;
9             contador = contador+1;
10        }
11        return soma;
12    }
13
14    public static void main(String []args) {
15        System.out.println( somaP(4) );
16    }
17 }
```

# Exercício 03: Somatória de números

Modifique o método **somaP** em que, além do número **n**, seja utilizado um outro número **k**, de tal forma que o calculo da seguinte somatória seja realizada:

$$1^k + 2^k + 3^k + \dots + n^k$$

**Assinatura:** `static int somaP(int n, int k)`

# Exercício 03: Somatória de números

```
1 import java.math.*;
2
3 public class TesteLaco03
4 {
5     static int somaP(int n, int k) {
6         int soma = 0;
7         int contador = 1;
8
9         while (contador<=n){
10             soma = soma + (int)Math.pow(contador, k);
11             contador = contador+1;
12         }
13         return soma;
14     }
15
16     public static void main(String []args) {
17         System.out.println( somaP(4,5) );
18     }
19 }
```

# Exercício 03: Somatória de números

Modifique o método **somaP** em que, além do número **n**, seja utilizado um outro número **k**, de tal forma que o calculo da seguinte somatória seja realizada:

$$k^1 + k^2 + k^3 + \dots + k^n$$

**Assinatura:** `static int somaP(int n, int k)`

# Exercício 03: Somatória de números

```
1 import java.math.*;
2
3 public class TesteLaco03
4 {
5     static int somaP(int n, int k) {
6         int soma = 0;
7         int contador = 1;
8
9         while (contador<=n){
10            soma = soma + (int)Math.pow(k, contador);
11            contador = contador+1;
12        }
13        return soma;
14    }
15
16    public static void main(String []args) {
17        System.out.println( somaP(4,5) );
18    }
19 }
```

# Exercício 04: Número triangular

Dizemos que um número natural é triangular se ele é produto de três número naturais consecutivos.

Dado um inteiro não negativo  $n$ , crie um método que permita verificar se é triangular.

## Exemplo:

- 120 é triangular, pois  $4*5*6 = 120$ .
- 2730 é triangular, pois  $13*14*15 = 2730$ .

**Assinatura:** `static boolean numeroTriangular(int n)`

Devolve “true” se o número for triangular, caso contrário “false”.

# Exercício 04: Número triangular

```
static boolean numeroTriangular(int n){
    int i=1;

    while ( i*(i+1)*(i+2)<n ) {
        i = i+1;
    }

    if ( i*(i+1)*(i+2)==n )
        return true;
    else
        return false;
}
```

# Exercício 04: Número triangular

```
1 public class TesteLaco05
2 {
3     static boolean numeroTriangular(int n){
4         int i=1;
5
6         while ( i*(i+1)*(i+2)<n ) {
7             i = i+1;
8         }
9
10        if ( i*(i+1)*(i+2)==n )
11            return true;
12        else
13            return false;
14    }
15
16    public static void main(String []args)
17    {
18        System.out.println( numeroTriangular(120) );
19        System.out.println( numeroTriangular(2730) );
20        System.out.println( numeroTriangular(27) );
21    }
22 }
```

# Exercício 04: Número triangular

```
true  
true  
false
```



# **Atividade em aula**

# Questão 1

```
static int m1(int a) {  
    while (a>0) {  
        a = a-1;  
    }  
    return a;  
}
```

```
System.out.println( m1(10) );  
System.out.println( m1(0) );
```

```
0  
0
```

# Questão 2

```
static int m2(int x, int y) {  
    int t, r=0;  
  
    if (x>y) {  
        t = x;  
        x = y;  
        y = t;  
    }  
  
    while (x<=y) {  
        r = r+x;  
        x = x+1;  
    }  
  
    return r;  
}
```

```
System.out.println( m2(10,14) );  
System.out.println( m2(4,-3) );
```

Somatória dos inteiros  
no intervalo  $[x, y]$

# Questão 3

```
static int m3(int v) {  
    while ( (v>0 && v<10) || (v>10 && v<20) ) {  
        v = v+5;  
    }  
    return v;  
}
```

```
System.out.println( m3(10) );  
System.out.println( m3(1) );
```

10  
21

# Questão 4

```
static int m4(int x) {  
    int n=1;  
    while ( x>1 ) {  
        n = n*x;  
        x = x-1;  
    }  
    return n;  
}
```

```
System.out.println( m4(3) );  
System.out.println( m4(6) );
```

6  
720

Fatorial de x

# Questão 5

```
static int operacao(int x) {  
    int i, j, soma=0;  
  
    i=1;  
    while ( i<=x ) {  
        j=1;  
        while (j<=x) {  
            soma = soma + x;  
            j = j+1;  
        }  
        i = i+1;  
    }  
  
    return soma;  
}
```

```
System.out.println( operacao(2) );  
System.out.println( operacao(3) );
```

$x^3$

# Desafio

```
static int somatoria(int n) {  
    int i, soma=0;  
  
    while (n>=1) {  
        i=n;  
        while (i>=1) {  
            soma = soma + n;  
            i = i-1;  
        }  
        n = n-1;  
    }  
    return soma;  
}
```

O método somatoria, dado um inteiro  $n$  positivo, calcula:

$$1^2 + 2^2 + 3^2 + \dots + n^2$$