

# Estruturas de Repetição e String

Vanessa Braganholo  
vanessa@ic.uff.br

# Aula de hoje...

---

- ▶ Estruturas de repetição

- ▶ *while...do*
- ▶ *do...while*
- ▶ *for*

- ▶ String

- ▶ Manipulação de textos

# Estruturas de Repetição

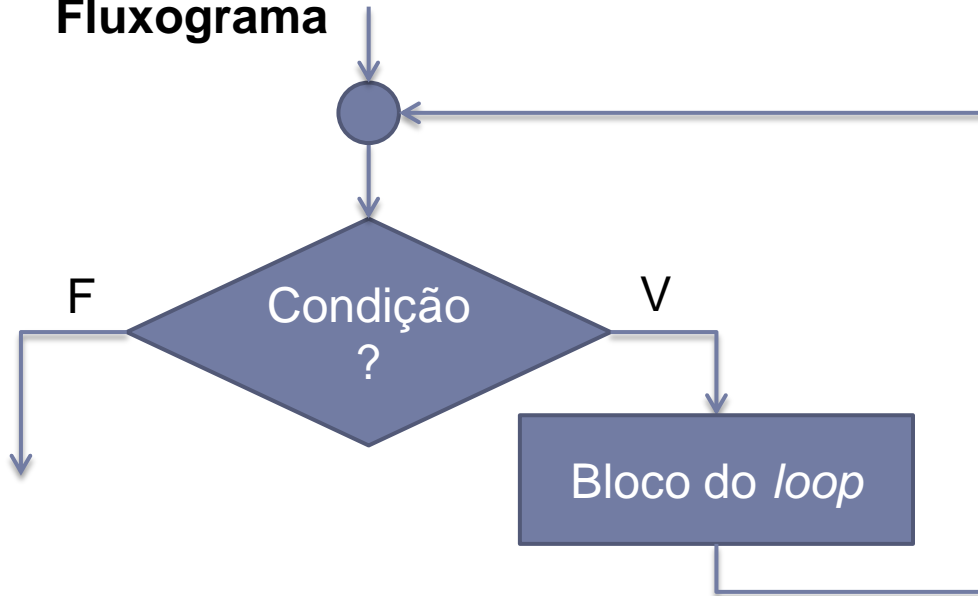
---



- ▶ Permitem que um bloco de comandos seja executado diversas vezes
- ▶ **Repetição condicional:** executa um bloco de código enquanto uma condição lógica for verdadeira
  - ▶ *do...while*
  - ▶ *while...do*
- ▶ **Repetição contável:** executa um bloco de código um número determinado de vezes
  - ▶ *for*

# Repetição condicional do tipo *while...do*

Fluxograma



Pseudocódigo

```
...  
Enquanto CONDIÇÃO faça  
    INSTRUÇÃO 1  
    INSTRUÇÃO 2  
    ...  
    INSTRUÇÃO N  
...
```

# Repetição condicional do tipo *while...do*

---

## Java

```
...  
while (CONDIÇÃO) {  
    INSTRUÇÃO 1;  
    INSTRUÇÃO 2;  
    ...  
    INSTRUÇÃO N;  
}  
...
```

# Repetição condicional do tipo *while...do*

---

- ▶ Executa o bloco de instruções enquanto a condição for verdadeira
- ▶ A condição é uma expressão booleana que pode fazer uso de quaisquer operadores
- ▶ A condição deve sempre estar entre parênteses
- ▶ Pode-se omitir { e } caso execute somente uma instrução

# Repetição condicional do tipo *while...do*

---

- ▶ Executa o bloco de instruções enquanto a condição for verdadeira
- ▶ A condição é uma expressão booleana que pode fazer uso de quaisquer operadores
- ▶ A condição deve sempre estar entre parênteses
- ▶ Pode-se omitir { e } caso execute somente uma instrução

**Nenhuma novidade: igual ao if!!!**

# Exemplo de *while...do*

---

## ► Programa para calcular fatorial de um número:

```
import java.util.Scanner;

public class Fatorial {

    public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);
        System.out.print("Entre com um número inteiro positivo: ");
        int numero = teclado.nextInt();
        long fatorial = 1;
        while (numero > 0) {
            fatorial *= numero--;
        }
        System.out.println("O fatorial desse número é " + fatorial);
    }
}
```



# Exemplo de *while...do*

---

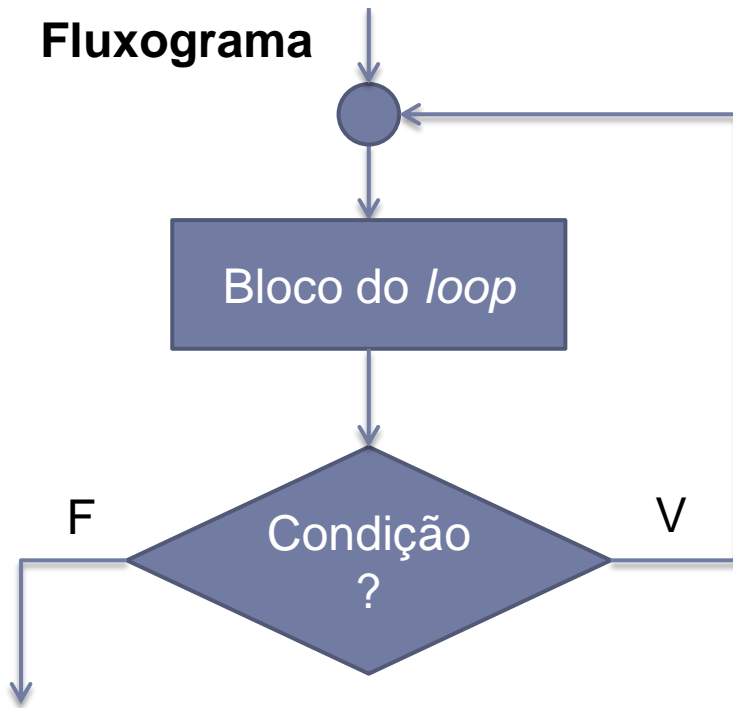
- ▶ Qual a saída do programa abaixo?

```
public class Loop {  
    public static void main(String[] args) {  
        int i = 0;  
        while (true)  
            System.out.println(i++);  
    }  
}
```

- ▶ Evitem forçar loops infinitos sempre que possível!

# Repetição condicional do tipo *do...while*

Fluxograma



Pseudocódigo

```
...  
Faça  
    INSTRUÇÃO 1  
    INSTRUÇÃO 2  
    ...  
    INSTRUÇÃO N  
Enquanto CONDIÇÃO  
...
```

# Repetição condicional do tipo *do...while*

---

## Java

```
...  
do {  
    INSTRUÇÃO 1;  
    INSTRUÇÃO 2;  
    ...  
    INSTRUÇÃO N;  
} while (CONDIÇÃO);  
...
```

# Repetição condicional do tipo *do...while*

---

- ▶ Executa o bloco de instruções enquanto a condição for verdadeira
- ▶ **Garante que ocorrerá ao menos uma execução**
  - ▶ A verificação da condição é feita depois do bloco de instruções
- ▶ Valem as mesmas condições do *while...do*

# Exemplo de *do...while*

---

## ► Programa para calcular fatorial de um número:

```
import java.util.Scanner;

public class Fatorial {

    public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);
        System.out.print("Entre com um número inteiro positivo: ");
        int numero = teclado.nextInt();
        long fatorial = 1;
        do {
            fatorial *= numero--;
        while (numero > 0);
        System.out.println("O fatorial desse número é " + fatorial);
    }
}
```

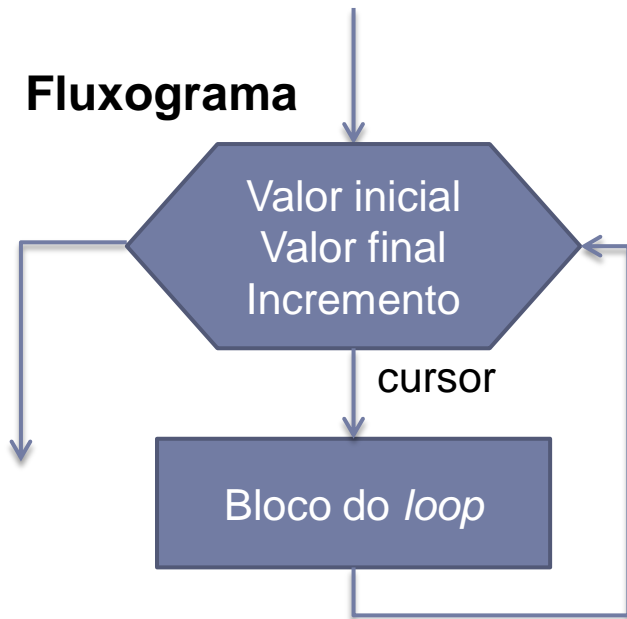
# Mas então... dá no mesmo?

---

- ▶ Naaaaaaaaaaaaaaão!!!
- ▶ Reparem que pedimos para o usuário **"Entre com um número inteiro positivo: "**
  - ▶ Para esse cenário, ambas as estruturas funcionaram
- ▶ O que acontece se pedirmos para o usuário **"Entre com um número inteiro não negativo: "**
  - ▶ Qual das duas estruturas resolve o problema corretamente se o usuário entrar com zero?
  - ▶ Qual o resultado provido pela outra?
  - ▶ Lembrem: fatorial de zero é 1!

# Repetição contável do tipo *for*

## Fluxograma



## Pseudocódigo

```
...  
Para CURSOR variando de VALOR INICIAL  
a VALOR FINAL com passo INCREMENTO  
    INSTRUÇÃO 1  
    INSTRUÇÃO 2  
    ...  
    INSTRUÇÃO N  
...
```

# Repetição contável do tipo *for*

---

## Java

```
...  
for (INICIALIZAÇÃO; TERMINAÇÃO; INCREMENTO) {  
    INSTRUÇÃO 1;  
    INSTRUÇÃO 2;  
    ...  
    INSTRUÇÃO N;  
}  
...
```



# Repetição contável do tipo *for*

---

- ▶ Executa o bloco de instruções por um número predeterminado de vezes
- ▶ **Expressão de inicialização**
  - ▶ Utilizada para iniciar a variável de controle do *loop* (cursor)
  - ▶ Executada uma única vez, antes do primeiro *loop*
- ▶ **Expressão de terminação**
  - ▶ Termina a execução do *loop* quando tiver o valor *false*
  - ▶ Verificada antes de cada *loop*
- ▶ **Expressão de incremento**
  - ▶ Pode incrementar ou decrementar a variável de controle (cursor)
  - ▶ Executada no final de cada *loop*
- ▶ As expressões devem sempre estar entre parênteses e separadas por ponto-e-vírgula
- ▶ Pode-se omitir { e } caso execute somente uma instrução

# Exemplo de *for*

---

## ► Programa para calcular fatorial de um número:

```
import java.util.Scanner;

public class Fatorial {

    public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);
        System.out.print("Entre com um número inteiro positivo: ");
        int numero = teclado.nextInt();
        long fatorial = 1;
        for (int i = 1; i <= numero; i++) {
            fatorial *= i;
        }
        System.out.println("O fatorial desse número é " + fatorial);
    }
}
```

# Exemplo de *for*

---

## ► Qual a diferença de

```
for (int i = 1; i <= numero; i++) {  
    fatorial *= i;  
}
```

## ► Para

```
for (int i = numero; i >= 1; i--) {  
    fatorial *= i;  
}
```

## ► ?

# String

---

- ▶ Classe em Java para representar variáveis textuais
- ▶ Possui uma variedade de métodos para manipulação de texto
- ▶ Métodos podem ser chamados a partir de uma variável ou do texto em si
  - ▶ `System.out.println(texto.charAt(2));`
  - ▶ `System.out.println("Texto".charAt(2));`
- ▶ Para manipulações mais eficientes com strings, veja a classe **StringBuffer**

# Alguns métodos de String

---

- ▶ **equals(Object)**
  - ▶ Informa se duas Strings são iguais
  - ▶ Ex.: "Flamengo".equals("flamengo") → false
  - ▶ Ex.: "Flamengo".equals("Flamengo") → true
- ▶ **length()**
  - ▶ Retorna o tamanho da String
  - ▶ Ex.: "Flamengo".length() → 8
- ▶ **concat(String)**
  - ▶ Concatena duas strings, de forma equivalente ao operador +
  - ▶ Ex.: "Fla".concat("mengo") → "Flamengo"
- ▶ **charAt(int)**
  - ▶ Retorna o caractere na posição informada
  - ▶ A primeira posição é zero
  - ▶ Ex.: "Flamengo".charAt(2) → 'a'

# Alguns métodos de String

---

## ▶ `compareTo(String)`

- ▶ Retorna 0 se as strings forem iguais, <0 se a string for lexicamente menor e >0 se for lexicamente maior que o parâmetro
- ▶ `"Fla".compareTo("Flu") → -20`

## ▶ `compareToIgnoreCase(String)`

- ▶ Idem ao anterior, sem considerar diferenças entre maiúsculas e minúsculas
- ▶ `"Fla".compareToIgnoreCase("fla") → 0`

## ▶ `indexOf(String, int)`

- ▶ Busca pela primeira ocorrência de uma substring ou caractere a partir de uma posição informada
- ▶ Ex.: `"Fla x Flu".indexOf("Fl", 0)) → 0`
- ▶ Ex.: `"Fla x Flu".indexOf("Fl", 1)) → 6`

# Alguns métodos de String

---

- ▶ **substring(int, int)**
  - ▶ Retorna a substring que vai da posição inicial (inclusive) até a posição final (exclusive), ambas informadas
  - ▶ Ex.: "Flamengo".substring(3,6)) → "men"
- ▶ **toLowerCase()**
  - ▶ Retorna a string em minúsculas
  - ▶ Ex.: "Flamengo".toLowerCase() → "flamengo"
- ▶ **toUpperCase()**
  - ▶ Retorna a string em maiúsculas
  - ▶ Ex.: "Flamengo".toUpperCase() → "FLAMENGO"
- ▶ **trim()**
  - ▶ Remove espaços antes e depois da string
  - ▶ Ex.: " Flamengo ".trim() → "Flamengo"

# Alguns métodos de String

---

- ▶ Veja os demais métodos em
  - ▶ <http://docs.oracle.com/javase/7/docs/api/java/lang/String.html>
- ▶ Na verdade, todas as classes de apoio do Java podem ser consultadas em
  - ▶ <http://docs.oracle.com/javase/7/docs/api/>



# Exemplo

---

- ▶ Programa para gerar a citação a partir de um nome
  - ▶ Ex.: Leonardo Gresta Paulino Murta → MURTA, L. G. P.

```
import java.util.Scanner;
```

```
public class Citacao {
```

```
    public static void main(String[] args) {
```

```
        Scanner teclado = new Scanner(System.in);
```

```
        String iniciais = "";
```

```
        String sobrenome = "";
```


```
        System.out.print("Entre com um nome completo: ");
```

```
        String nome = teclado.nextLine().trim();
```



# Exemplo

---



```
int inicio = 0;
int fim = nome.indexOf(" ", inicio);
while (fim != -1) {
    iniciais += nome.substring(inicio, inicio + 1) + ". ";
    inicio = fim + 1;
    fim = nome.indexOf(" ", inicio);
}
sobrenome = nome.substring(inicio).toUpperCase();

System.out.print(sobrenome + ", ");
System.out.println(iniciais.toUpperCase().trim());
}
}
```

# Exercício

---

- ▶ Faça um programa para montar a tabela de multiplicação de números de 1 a 10 (ex.:  $1 \times 1 = 1$ ,  $1 \times 2 = 2$ , etc.)
- ▶ Faça um programa para determinar o número de dígitos de um número informado (sem usar String)

# Exercício

---

- ▶ Faça um programa para calcular a série de Fibonacci para um número informado pelo usuário, sendo  $F(0) = 0$ ,  $F(1) = 1$  e  $F(n) = F(n-1) + F(n-2)$
- ▶ Por exemplo, caso o usuário informe o número 9, o resultado seria: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34

# Exercício

---

- ▶ Faça um programa para listar todos os divisores de um número ou dizer que o número é primo caso não existam divisores
  - ▶ Ao final, verifique se o usuário deseja analisar outro número

# Exercício

---

- ▶ Faça um programa que calcule o retorno de um investimento financeiro fazendo as contas mês a mês, sem usar a fórmula de juros compostos
  - ▶ O usuário deve informar quanto será investido por mês e qual será a taxa de juros mensal
  - ▶ O programa deve informar o saldo do investimento após um ano (soma das aplicações mês a mês considerando os juros compostos), e perguntar ao usuário se ele deseja que seja calculado o ano seguinte, sucessivamente
  - ▶ Por exemplo, caso o usuário deseje investir R\$ 100,00 por mês, e tenha uma taxa de juros de 1% ao mês, o programa forneceria a seguinte saída:

Saldo do investimento após 1 ano:

1280.9328043328942

Deseja processar mais um ano? (S/N)

# Exercício

---

- ▶ Escreva um programa em Java que imprime na tela os  $n$  primeiros números perfeitos. Um número perfeito é aquele que é igual à soma dos seus divisores. Por exemplo,  $6 = 1 + 2 + 3$ .

# Exercício

---

- ▶ Um número inteiro é considerado triangular se este for o produto de 3 números inteiros consecutivos, como, por exemplo,  $120 = 4 \times 5 \times 6$
- ▶ Elabore um programa que, após ler um número  $n$  do teclado, verifique se  $n$  é triangular



# Exercício

---

- ▶ Elabore um programa em Java que leia  $n$  valores e mostre a soma de seus quadrados.

# Exercício

---

- ▶ Faça um programa que lê dois valores  $x$  e  $y$ , e calcula o valor de  $x$  dividido por  $y$ , além do resto da divisão. Não é permitido usar as operações de divisão e resto de divisão do Java (use apenas soma e subtração).

# Exercício

---

- ▶ Escreva um programa em Java que permita a entrada de uma String S e então exiba na tela todas as possíveis rotações à esquerda de S. Por exemplo, se o usuário digitar “Banana”, o programa deve exibir:
  - ▶ “Banana”
  - ▶ “ananaB”
  - ▶ “nanaBa”
  - ▶ “anaBan”
  - ▶ “naBana”
  - ▶ “aBanan”
  - ▶ “Banana”

# Exercício

---

- ▶ Faça um programa em Java que lê uma String e verifica se essa String contém um número real válido. Considere que o número real pode ser positivo ou negativo, e que o separador decimal será o símbolo “.”, quando houver
- ▶ Exemplo:
  - ▶ -1.22 → é um número real
  - ▶ 10 → é um número real
  - ▶ 12.A → não é um número real

# Exercício

---

- ▶ Faça um programa para justificar um texto com um número de colunas informado pelo usuário
- ▶ Por exemplo, para o texto “Este é um exemplo de texto que vamos justificar usando o nosso programa” quando justificado em 18 colunas, teríamos:

Este é um exemplo  
de texto que vamos  
justificar usando  
o nosso programa

- ▶ Dica: o método `lastIndexOf(String, int)` pode ser útil

# Referências

---

- ▶ Slides de Leonardo Murta

# Estruturas de Repetição e String

Vanessa Braganholo  
vanessa@ic.uff.br