

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RIO GRANDE DO NORTE

PROGRAMAÇÃO ESTRUTURADA E ORIENTADA A OBJETOS

Docente: Éberton da Silva Marinho

e-mail: ebertonsm@gmail.com

eberton.marinho@gmail.com

09/04/2019

SUMÁRIO

- Entrada e Saída de Dados
- Estruturas de Repetição



ENTRADA DE DADOS

- A maior parte dos programas que fazemos necessitam de uma entrada, seja através do usuário ou de sistemas legados, para dar o início a algum procedimento;
- De forma simplificada, um programa tem 3 fases:
 - Entrada: dados utilizados no processamento
 - Processamento: O problema é resolvido com base nas entradas
 - Saída: O resultado da resolução do problema é apresentado



ENTRADA DE DADOS

- Há algumas formas do usuário fornecer entradas a programas Java. Uma delas é usando a classe Scanner da biblioteca java.útil
- A classe Scanner possui vários métodos que permitem a entrada de diferentes tipos, entre eles:
 - next(): retorna uma String, que não tenha espaços em branco;
 - nextDouble(): retorna um número em notação ponto flutuante;
 - nextInt(): retorna um número inteiro;
 - nextLine(): retorna uma String, linha inteira digitada pelo usuário;
 - nextLong() - retorna um número inteiro de 64 bits.



ENTRADA DE DADOS

- Para utilizar a classe Scanner, é necessário importá-la através da declaração:
 - `import java.util.Scanner;`
- Antes de utilizar a classe, é necessário instanciá-la através do comando:
 - `Scanner sc = new Scanner(System.in);`
- A partir daqui, basta utilizar os métodos da classe Scanner para ler dados do usuário.
 - `int num = sc.nextInt();`
 - `double salario = sc.nextDouble();`
 - `String nome = sc.nextLine();`



SAÍDA DE DADOS

- Há algumas formas do usuário fornecer saída de dados em programas Java. Uma delas é usando os métodos da biblioteca System.out
- A biblioteca System.out possui vários métodos que possibilitam a saída de dados.
 - print: Imprime um objeto passada como parâmetro pelo seu método;
 - println: Imprime um objeto passada como parâmetro pelo seu método e adiciona uma quebra de linha ao final;
 - printf: Imprime um objeto de acordo com o formato de impressão passado como parâmetro



SAÍDA DE DADOS

○ System.out.printf

- System.out.printf(formato, objetos a serem impressos)
- Formatos:
 - %s = String
 - %d = Inteiro
 - %f = número com ponto flutuante.
 - \t = tabulação
 - \n = salto de linha
- Exemplo:
 - System.out.printf ("%d \t %f \t %s", 5, 254.336, "Texto");
 - Saída: 5 254,336 Texto



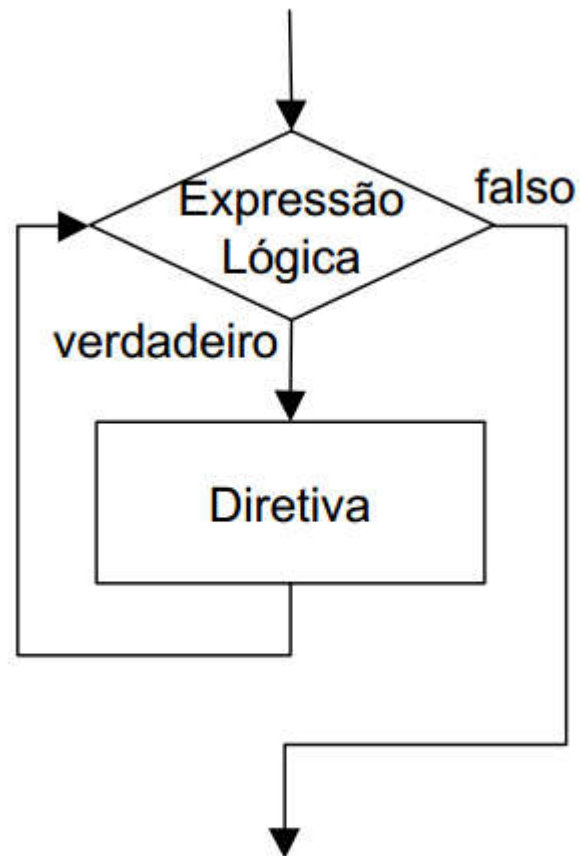
ESTRUTURAS DE REPETIÇÃO

- Deve ser usada sempre que uma rotina tenha que ser executada mais do que uma vez durante o mesmo processamento.
- As repetições podem ser controladas por:
 - **Contador Fixo**
 - **Validação de Condição**



ESTRUTURAS DE REPETIÇÃO

- Fluxo de uma instrução de repetição



ESTRUTURAS DE REPETIÇÃO

- for(; enquanto condição for verdadeira;)
- while(condição for verdadeira)
- do ... while (condição for verdadeira)



ESTRUTURAS DE REPETIÇÃO

- Aspectos a atentar em uma estrutura de repetição
 - Inicialização: Antes de iniciar a repetição, a condição para entrada na estrutura deve ser satisfeita
 - Critério de parada: Antes de cada repetição, o critério de parada da repetição é avaliado. Se a condição não for mais atendida, a repetição é interrompida
 - Próximo passo: No final de cada repetição, um próximo passo deve ser definido para que o código possa um dia parar



ESTRUTURAS DE REPETIÇÃO

○ Estrutura for

- for(inicialização; condição de parada; próximo passo)
comandos

- Exemplo:

```
for(int i = 1; i <= 10; i++){  
    System.out.println("Iteração: " + i);  
}
```

- Saída:

Iteração 1

Iteração 2

Iteração 3

Iteração 4

Iteração 5

Iteração 6

Iteração 7

Iteração 8

Iteração 9

Iteração 10



ESTRUTURAS DE REPETIÇÃO

○ Estrutura while

- while (condição de parada)
comandos

- Exemplo:

```
int i = 1;  
while (i <= 10) {  
    System.out.println("Iteração: " + i);  
    i++;  
}
```

- Saída:

Iteração 1

Iteração 2

Iteração 3

Iteração 4

Iteração 5

Iteração 6

Iteração 7

Iteração 8

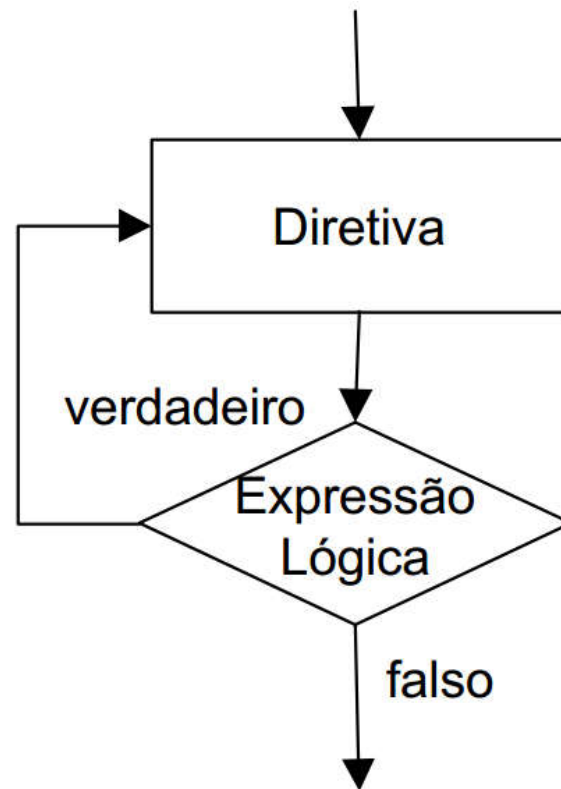
Iteração 9

Iteração 10



ESTRUTURAS DE REPETIÇÃO

- Fluxo de uma instrução de repetição da instrução do while



ESTRUTURAS DE REPETIÇÃO

○ Estrutura do

- do
 comandos
 while(condição de parada)

- Exemplo:

```
int i = 1;  
do {  
    System.out.println("Iteração: " + i);  
    i++;  
}while(i <= 10);
```

- Saída:

Iteração 1
Iteração 2
Iteração 3
Iteração 4
Iteração 5

Iteração 6

Iteração 7

Iteração 8

Iteração 9

Iteração 10



EXEMPLOS DE ESTRUTURAS DE REPETIÇÃO

○ Com contador fixo

- Exemplo 1

```
for(int i = 1; i <= 10; i++){  
    //comandos  
}
```

- Exemplo 2

```
int i = 1;  
while (i <= 10){  
    // comandos  
    i++;  
}
```

- Exemplo 3

```
int i = 1;  
do{  
    // comandos  
    i++;  
}while (i <= 10);
```



EXEMPLOS DE ESTRUTURAS DE REPETIÇÃO

○ Com validação de condição

- Exemplo 1

```
Scanner sc = new Scanner(System.in);  
System.out.println("Digite uma opção:");  
int opcao = sc.nextInt();  
while (opcao != 0) {  
    // comandos  
  
    System.out.println("Digite uma opção:");  
    opcao = sc.nextInt();  
}
```



ESTRUTURAS DE REPETIÇÃO

○ Quando usar:

- for: quando se sabe exatamente quantas vezes o código será executado. Geralmente mais utilizado com um contador.
- while: quando não há como prever quantas vezes o código será executado, pois depende de alguma variável não determinística, como a escolha do usuário, ou no caso de expressões mais genéricas.
- do ... while: tem um propósito muito parecido com o while, porém, executa obrigatoriamente uma vez os comandos a serem repetidos, antes de avaliar a condição de parada.



INTERRUPÇÕES DE REPETIÇÃO


- As três estruturas de repetição não estão condicionadas apenas à avaliação da condição de parada, elas podem ser interrompidas antes disso através dos comandos **continue** e **break**.
- **continue**: faz com que o fluxo da estrutura de repetição volte ao início do bloco de comandos;
- **break**: interrompe a execução da estrutura de repetição.



INTERRUPÇÕES DE REPETIÇÃO

- break e continue

```
while (!terminado) {  
    passePagina();  
    if (alguemChamou == true) {  
        break;           // caia fora deste loop  
    }  
    if (paginaDePropaganda == true) {  
        continue;       // pule esta iteração  
    }  
    leia();  
}  
restoDoPrograma();
```



ESTRUTURAS DE REPETIÇÃO

- Exercícios



DICAS DO DIA

- Certifique-se que a estrutura de repetição será inicialmente executada em algum momento do código.
- Certifique-se que a estrutura de repetição para em algum momento. A não ser que a repetição infinita faça parte do projeto do programa.



DÚVIDAS

- e-mail:

ebertonsm@gmail.com

eberton.marinho@ifrn.edu.br

- Endereço eletrônico da disciplina:

- <http://docente.ifrn.edu.br/ebertonmarinho>