LINGUAGEM DE PROGRAMAÇÃO ORIENTADA À OBJETOS – LPOO

Aula 08 – Looping

TÓPICOS DA AULA

- Learn about the **loop structure**
- Create **while** loops
- Create **for** loops
- Create **do…while** loops
- Improve loop performance

Java Programming – Joyce Farrell (Chapter 6)

ESTRUTURAS DE REPETIÇÃO

- Permitem que um determinado trecho de código seja executado várias vezes.

- No Java possuímos três instruções de repetição

while **do-while** **for**

ESTRUTURAS DE REPETIÇÃO - WHILE

- A instrução de repetição **while (enquanto)** recebe como parâmetro uma condição.

- Enquanto essa condição for **verdadeira**, o bloco de comandos do corpo do laço será executado.

- A condição da instrução deve ser um valor do tipo boolean.

- **O teste da condição booleana vem antes de ser executado o laço**

ESTRUTURAS DE REPETIÇÃO - WHILE

Sintaxe:

```
while (condição) {

// Trecho de código

}
```

Exemplo:

```
int cont = 0;
while (cont < 10) {
        System.out.print(cont + " ");
        cont = cont + 1;
}
```

O trecho dentro do bloco código while será executado até o momento em que a condição (cont < 10) for falsa. Isso ocorrerá qdo o valor da variavel cont for igual a 10.

O que acontece ser remover a instrução cont = cont + 1 ???

ESTRUTURAS DE REPETIÇÃO – DO-WHILE

Sintaxe:

```
do {

// Trecho de código

} while (condição);
```

Exemplo:

```
int cont = 0;
do {
        System.out.print(cont + " ");
        cont = cont + 1;
} while (cont < 10) {
```

O trecho dentro do bloco código while será executado, **pelo menos uma vez** e até o momento em que a condição (cont < 10) for falsa. Isso ocorrerá qdo o valor da variavel cont for igual a 10.

Novamente: O que acontece ser remover a instrução cont = cont + 1 ???

ESTRUTURAS DE REPETIÇÃO – FOR

- Outro comando de loop extremamente utilizado é o **FOR**

- A ideia é a mesma do while – fazer um trecho de código ser repetido enquanto uma condição continuar verdadeira

- **Possui também um espaço para inicialização e modificação das variáveis**

- Vantagens da estrutura for é que não precisamos nos preocupar em fazer o controle da parada manualmente. Evitando assim problemas com loop infinito

ESTRUTURAS DE REPETIÇÃO – FOR

Sintaxe:

```
for (inicialização; condição; incremento) {
 // Trecho de código
}
```

Exemplo:

```
for (int i = 0; i < 10; i ++) {
      System.out.printl("Olá Mundo!!!");
}
```

O trecho dentro do bloco for será executado até o momento em que a condição (i < 10) for falsa. Isso ocorrerá qdo o valor da variavel cont for igual a 10.

Apesar de termos condições booleanas nos nossos laços, em algum momento, podemos decidir parar o loop por algum motivo especial sem que o resto do laço seja executado.

ESTRUTURAS DE REPETIÇÃO – BREAK

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

        int i, x = 1, y = 100;

        for (i = x; i <= y; i++) {

            if (i % 19 == 0) {
                System.out.println("\nAchei um numero divisivel por 19 entre 1 e 100... " + i);
                break;
            }
            System.out.print(i + " ");

        }
    }
}
```

O código vai vai percorrer os números de x a y e parar quando encontrar um número divisível por 19, uma vez que foi utilizado a palavra-chave **break**

EXERCICIOS

1) Crie um classe em Java que contem 2 métodos que:

   a) *Percorre os números de 0 a 100 e exibe a quantidade números impares*
   b) *Peça para o usuário digitar números, usar a classe JOptionPane (-1 para finalizar). E exibe a quantidade de números pares que o usuário informou*

Para testar os métodos, crie uma nova classe que contém o método main e faz a chamada para esses métodos.

EXERCICIOS

2) Crei uma classe em Java chamada Looping com o seguinte método:

- **exibeNumerosSequencial** – que recebe 2 parâmetros inteiro que indica o valor inicial e valor final da sequencia a ser exibida.

Exemplo

- *exibeNumerosSequencial (0,100) -> Exibe no console os números de 0 -100*
- *exibeNumerosSequencial (10,40) -> Exibe no console os números de 10 - 40*

Para testar o método, crie uma nova classe chamada LoopingTest que contém o método main.

PRÓXIMA AULA

- *Characters, Strings and the StringBuilder*

Leitura:

- *Java Programming – Joyce Farrell*
- *Chapter - Characters, Strings and the StringBuilder) pag. 353*