Estruturas de Repetição

Aula 04

Gustavo Willam Pereira

Comando `while`

- A estrutura de repetição ***while*** é utilizada para executar instruções repetidas vezes, enquanto um expressão booleana for verdadeira. A sintaxe da instrução while é:
  - `while (expressaoBooleana)`
  - `    ... comando`

- A instrução "***expressaoBooleana***" é avaliada, se for verdadeira o comando é executado. A estrutura ***while*** é repetida executando os comandos até que a condição se torne falsa.

Estruturas de repetição - While

```
// Programa para calcular a media da turma
// utilizando laço  controlado por contador

import javax.swing.JOptionPane;

public class Average
{

   public static void main(String args[])
   {
     int total;  //armazena o total de notas
     int media;  //armazena a media das notas
     int nota;   //armazena a nota digitada
     int counter; //quantas notas serao digitadas
     int i = 1;      //variavel auxiliar

     String strcounter;
     String strnota;

     total = 0; // inicializa a variável total

strcounter = JOptionPane.showInputDialog("Digite a quantidade
                                          de notas a serem processadas");

  //converte de string para inteiro
   counter = Integer.parseInt(strcounter);

   while ( i <= counter )
   {
     strnota = JOptionPane.showInputDialog("Digite a nota " + i);

     nota = Integer.parseInt(strnota);
     total = total + nota;

     i = i + 1;
   }

   media = total / counter;

   JOptionPane.showMessageDialog(null, "A media das notas
                                        digitadas sao " + media, "Classe media",
                                        JOptionPane.INFORMATION_MESSAGE);

   System.exit (0);

  } //fim do método principal

}  //fim da classe Average
```

Estruturas de repetição - While

```
// Programa para calcular a media da turma utilizando laço
// controlado por sentinela

import javax.swing.JOptionPane;
import java.text.DecimalFormat;  //classe para formatação de números

public class Average2
{

   public static void main(String args[])
   {
     int total;  //armazena o total de notas
     double media;  //armazena a media das notas
     int nota;   //armazena a nota digitada
     int counter = 0;      //variável auxiliar para controle do laço

     String strcounter;
     String strnota;

     total = 0; // inicializa a variável total

     strnota = JOptionPane.showInputDialog("Digite a nota ou -1 para sair");

     nota = Integer.parseInt(strnota);
```

Estruturas de repetição - While

```
while ( nota != -1 )
  {
     nota = Integer.parseInt(strnota);  //converte a nota digitada de String para Integer
     total = total + nota;  //adiciona a nota ao total de notas
     strnota = JOptionPane.showInputDialog("Digite a nota ou - 1 para sair");
     nota = Integer.parseInt(strnota);
     counter = counter + 1; //incrementa o contador
  }

  DecimalFormat twoDigits = new DecimalFormat("0.00");   //declara um objeto twoDigits do tipo DecimalFormat

  if ( counter != 0 )
  {
    media = (double) total / counter;   //conversão explicita da variável total.

  JOptionPane.showMessageDialog(null, "A media das notas digitadas sao " +
                          twoDigits.format(media), "Classe media", JOptionPane.INFORMATION_MESSAGE);
  }
  else
    JOptionPane.showMessageDialog(null, "Nenhuma nota digitada", "Classe media",
                          JOptionPane.INFORMATION_MESSAGE);

  System.exit (0);

}
```

Estruturas de repetição - While

- Exemplo calculando horas trabalhadas

```
Scanner leitura = new Scanner(System.in);
int emp_cont = 0; // nenhum empregado processado ainda
float pag = 0;
float horas = 0;
float salario_hora = 0;
while (emp_cont < 7) // testa o contador de empregados
{
    System.out.println("Horas trabalhadas: ");
    horas = leitura.nextInt();
    System.out.println("Salario por hora: R$");
    salario_hora = leitura.nextFloat();
    pag = horas * salario_hora;
    System.out.println("Pagamento semanal é R$: " + pag);
    emp_cont = emp_cont + 1; // incrementa o contador de
empregados
} // Final do laço while
System.out.println("Todos os empregados processados.");
```

Operadores de Atribuição

- Qualquer comando da forma
  - *variável* = *variável operador expressão;*
- Onde operador é um dos operadores binários +, -, *, /, ou % pode ser escrito da forma
  - *variável operador* = *expressão;*

Exemplo:

Forma normal : c = c + 3;

Forma abreviada: c += 3;

Operadores de Atribuição

Operador de Atribuição	Expressão Exemplo	Explicação
+=	C += 7	C = C + 7
-=	D -= 4	D = D – 4
*=	E *= 5	E = E * 5
/=	F /= 3	F = F / 3
%=	G %= 9	G = G % 9

Operadores de incremento e decremento

- Operador unário de incremento: ++
- Operador unário de decremento: --
- Operador de incremento
  - Pré-incremento : ++c;
  - Pós-incremento : c++;
- Operador de decremento
  - Pré-decremento : --b;
  - Pós-decremento : b--;

Operadores de incremento e decremento

```
// programa de pre-incremento e pos-incremento

public class Increment
{
  public static void main(String args[] )
  {
     int c;
     c = 5;
     System.out.println( c ); //imprime 5
     System.out.println( c++ ) ; // imprime 5 entao pos-incrementa
     System.out.println( c ) ; // imprime 6

     System.out.println (); // pula uma linha

     c = 5;
     System.out.println ( c ); // imprime 5
     System.out.println( ++c ) ; // pre-incrementa e imprime 6
     System.out.println( c ) ; // imprime 6
  }
}
```

Estruturas de repetição - For

- A instrução ***for*** permite escrever uma forma mais formal de estrutura de repetição.
- Diferente da instrução ***while,*** existe a combinação da inicialização, a expressão booleana e a atualização do loop.

```
for (expressaoInicializacao; expressaoBooleana;
                              variavelControle)
    ... comando
```

Estruturas de repetição - For

- Exemplo que lista os números de 0 a 9.

```
for (int i = 0; i < 10; i++)
{
    System.out.println("Valor de i: " + i);
    i++;
}
```

- A inicialização ocorre uma única vez no início do loop, instrução "***expressaoInicializacao"*** , normalmente um valor inteiro, que é responsável pelo controle do laço.

Estruturas de repetição - For

- A instrução "***expressaoBooleana***" deve conter uma expressão booleana, geralmente envolvendo uma variável que é responsável pelo controle do laço.
- A instrução "***variavelControle***" é atualizada para controlar o laço. Enquanto a expressão booleana for verdadeira a variável de controle é atualizada e os comandos são executados.

Comparações entre while e for

- Instrução ***while***

```
int i = 0;
while (i < 10)
{
    System.out.println("Valor de i: " + i);
    i++;
}
```

- Instrução ***for***

```
for (int i = 0; i < 10; i++)
{
    System.out.println("Valor de i: " + i);

}
```

Estruturas de repetição – do..while

- As estruturas ***while*** e ***for*** testam suas expressões booleanas no início do ***loop***. Se a expressão for falsa no início da interação, os comandos do ***loop*** não é executado.
- A estrutura ***do*** avalia a expressão booleana após a execução dos comandos, portanto, o corpo da instrução é executado ao menos uma vez.

Estrutura de repetição do..while

- Sintaxe da instrução do..while

```
do
{
  comandos
}
while(expressaoBoolean);
```

- A instrução do..while deve sempre terminar com ponto e virgula - “;”

Estrutura de repetição do..while

- Exemplo que lista os números de 0 a 9.

```
int i = 0;

do{

   System.out.println("Valor de i:" + i);
   i++;

}while(i < 10);
```

Comandos ***break*** e ***continue***

- A instrução ***break*** quando utilizada em conjunto com os comandos ***while, for, do***, causam uma saída imediata da execução dos comandos, ou seja termina com a execução do laço de repetição.

Comandos ***break*** e ***continue***

- Exemplo de utilização do comando break

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

        for ( int x = 1; x <= 10; x++ ) {

            if ( x == 5 )
              break; // sai do laço somente se x é 5

            System.out.println("Valor: " + x);

         }
    }

}
```

Comandos ***break*** e ***continue***

- O comando ***continue*** faz com que o programa execute imediatamente a próxima iteração do loop, o restante do corpo da instrução não e executado.

Comandos ***break*** e ***continue***

- Exemplo de utilização do comando continue

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

        for ( int x = 1; x <= 10; x++ ) {

         if ( x == 5 )
          continue;//Se x é igual a 5 passa para próximo laço

          System.out.println("Valor: " + x);

         }

    }
}
```