

The slide features abstract green geometric shapes. On the left, a solid green triangle points downwards. On the right, a complex arrangement of overlapping translucent green triangles in various shades of green and yellow-green creates a dynamic, layered effect.

Estruturas de controle

Prof.: Nelson Bellincanta Filho

Estruturas de controle

- ▶ Um programa de computador é um conjunto de instruções organizadas para produzir a solução de um problema determinado;
- ▶ Naturalmente as instruções de um programa são executadas em sequência, o que se denomina fluxo sequencial de execução;
- ▶ No entanto, em inúmeras circunstâncias, é necessário executá-las em uma ordem diferente da estritamente sequencial;
- ▶ Essas situações são caracterizadas pela necessidade da repetição de instruções e também pelo desvio do fluxo de execução, tarefas que podem ser realizadas pelas estruturas de controle da linguagem.

Diretivas e blocos



Diretivas e blocos

- ▶ As instruções de um programa são chamadas diretivas (statements), as quais são tradicionalmente escritas uma após a outra e separadas, por exemplo, com uma quebra de linha ou um caractere de pontuação;
- ▶ Em Java, assim como na linguagem C/C++/C#, as diretivas são separadas umas das outras pelo símbolo de pontuação ; (ponto e vírgula), sendo possível existir várias diretivas numa mesma linha desde que separadas por ;.

```
diretiva1; diretiva2;  
diretiva3;  
:  
diretivaN;
```

Diretivas e blocos

- ▶ As diretivas podem ser tratadas individualmente ou como um conjunto denominado bloco;
- ▶ Um bloco em Java é um grupo de diretivas delimitadas por chaves ({ e }), que define um escopo próprio e que recebe um tratamento equivalente ao de uma única diretiva;

// bloco que equivale a uma única diretiva

```
{  
    diretiva1;  
    diretiva2;  
    :  
    diretivaN;  
}
```

Diretivas e blocos

- ▶ As estruturas de controle operam sobre uma diretiva individual ou um bloco de diretivas;
- ▶ Outra consequência da criação de blocos é que as estruturas de controle podem ser aninhadas de forma recursiva;

Tipos de estruturas de controle



Tipos de estruturas de controle

As linguagens de programação possuem estruturas diferentes para controlar o fluxo de execução, ou seja, há estruturas de controle destinadas à repetição e outras que permitem desviar o fluxo de execução.

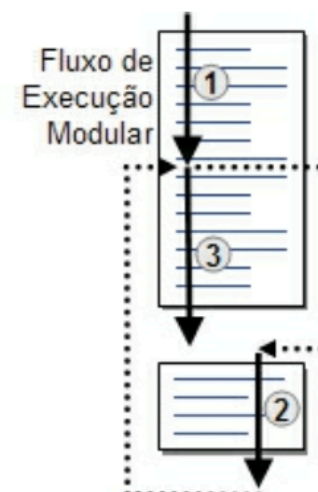
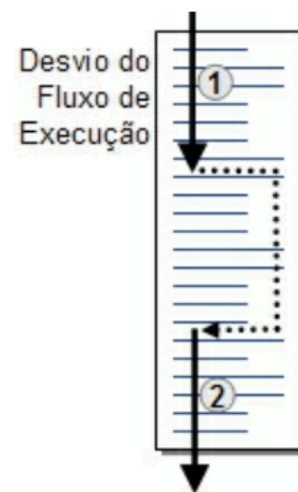


Tipos de estruturas de controle

- ▶ Essas estruturas são divididas em:
 - ▶ Repetição simples - Efetuam a repetição de diretivas ou blocos, criando os chamados laços. O número de repetições pode ser predefinido ou determinado pelo programa durante a execução. Corresponde à diretiva *for* no Java.
 - ▶ Repetição condicionais - São como as estruturas de repetição simples, mas cuja repetição está associada à avaliação de uma condição. Geralmente são usadas quando não se conhece de antemão o número necessário de repetições. No Java são representadas pelas diretivas *while* e *do while*;

Tipos de estruturas de controle

- ▶ Desvio de fluxo - Destinadas ao desvio da execução do programa para uma outra parte, quebrando o fluxo sequencial. O desvio condicional é aquele associado à avaliação de uma expressão, como uma tomada de decisão; e o desvio incondicional ocorre automaticamente. No Java, existem as diretivas **if else** e **switch** para o desvio condicional simples e múltiplo respectivamente, além de **break** e **continue** para o desvio incondicional.
- ▶ Execução modular - A divisão de um programa em partes menores facilita o entendimento, a correção ou modificação. No Java isso é feito por meio da construção de classes e métodos, além da divisão do código em pacotes e módulos. A chamada de métodos constitui um desvio incondicional.



Tipos de estruturas de controle

Além dessas estruturas, existem ainda as estruturas de controle de erros, que desviam o fluxo de execução para simplificar a inclusão de rotinas de tratamento de erros dentro dos programas.



The background features abstract green geometric shapes. On the left, a solid green triangle points downwards. On the right, a complex arrangement of overlapping, semi-transparent green triangles and polygons creates a layered, architectural effect. A thin, light gray line extends from the bottom left towards the right, passing through the green shapes.

Estruturas de repetição simples

Estruturas de repetição simples

- ▶ A repetição é uma das tarefas mais comuns da programação, utilizada para efetuar contagens, totalizações, obtenção de múltiplos dados, impressão etc;
- ▶ A repetição simples é a execução consecutiva de uma diretiva ou um bloco de um número conhecido e fixo de vezes, o que muitas vezes se denomina laço automático;
- ▶ Cada repetição efetuada é chamada de iteração.

Estruturas de repetição simples

- ▶ A repetição é uma das tarefas mais comuns da programação, utilizada para efetuar contagens, totalizações, obtenção de múltiplos dados, impressão etc;
- ▶ A repetição simples é a execução consecutiva de uma diretiva ou um bloco de um número conhecido e fixo de vezes, o que muitas vezes se denomina laço automático;
- ▶ Cada repetição efetuada é chamada de iteração.

Diretiva for

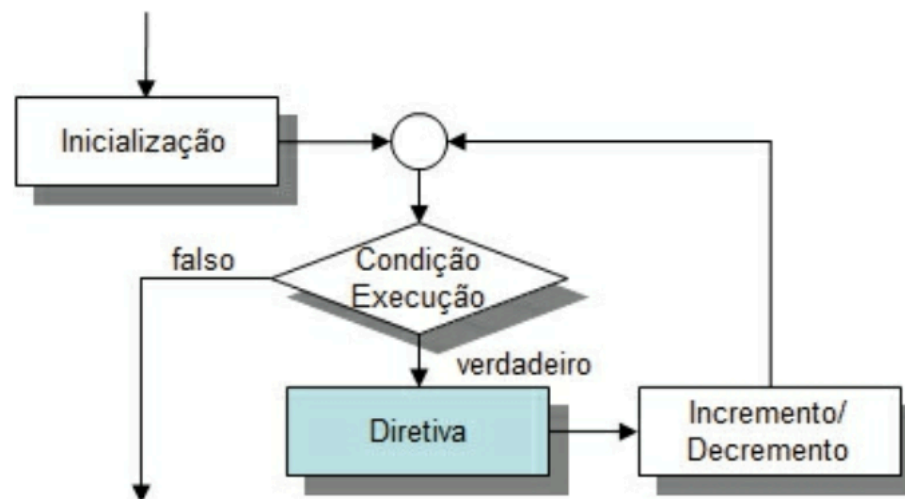


Diretiva for

- Em Java, a repetição simples corresponde à diretiva for, cuja sintaxe é:

```
for ([inicialização]; [condição]; [incremento|decremento])
```

```
< diretiva; | { /* bloco */ } >
```



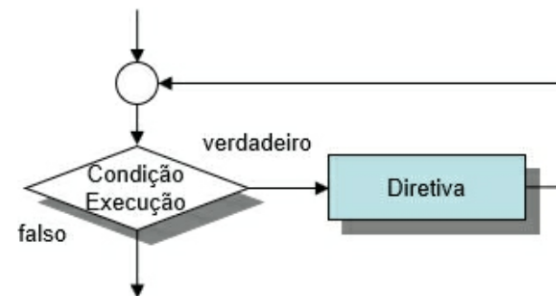
Diretiva while



Diretiva while

- ▶ Na diretiva while, um conjunto de instruções é repetido enquanto o resultado da condição (uma expressão lógica) é avaliado como verdadeiro;
- ▶ Sua sintaxe é:

```
while ( <condição> )  
< diretiva; | { /* bloco */ } >
```



- ▶ O while avalia o resultado da condição antes de executar a diretiva ou bloco associado, assim é possível não ocorrer sua execução caso a condição seja inicialmente falsa.

```
1  /*
2  Classe exemplo diretiva while: DiretivaWhile.java
3  IFPR – Campus Cascavel
4  Disciplina: Programação Orientada à Objetos
5  Professor: Nelson Bellincanta
6  Data da criação: 31/03/2023
7  */
8
9  import java.util.Scanner; //Importa a classe Scanner, que será usada para receber entrada do usuário.
10
11 public class DiretivaWhile {
12     public static void main (String args[]) {
13         Scanner s = new Scanner(System.in); //Cria uma nova instância da classe Scanner
14         System.out.print("Informe um valor inteiro inicial? "); //Exibe a mensagem "Valor inteiro inicial? " na saída padrão (console).
15         int j = s.nextInt(); //Lê um número inteiro digitado pelo usuário usando o método nextInt() da classe Scanner e atribui o valor a variável j.
16         while (j >= 0) { //Inicia um laço de repetição "while", que continuará executando enquanto o valor de j for maior ou igual a zero.
17             System.out.println(j); //Exibe o valor atual de j na saída padrão, seguido de uma quebra de linha.
18             j--; //Decrementa o valor de j em 1 a cada iteração do laço.
19         }
20     }
21 }
```

Diretiva while



```
Aula06 — -bash — 69x15
Aula06 $javac DiretivaWhile.java
Aula06 $java DiretivaWhile
Informe um valor inteiro inicial? 10
10
9
8
7
6
5
4
3
2
1
0
Aula06 $
```

The image shows a terminal window titled 'Aula06 — -bash — 69x15'. The user enters the command 'javac DiretivaWhile.java' to compile the program. Then, they enter 'java DiretivaWhile' to run it. The program prompts the user with 'Informe um valor inteiro inicial? 10'. The user enters '10'. The program then prints a sequence of numbers from 10 down to 0, one per line. Finally, the prompt 'Aula06 \$' is shown.

Execução da classe DiretivaWhile.java

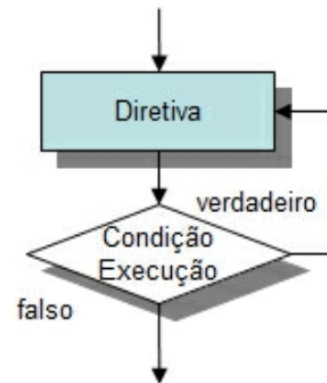
Diretiva do while



Diretiva do while

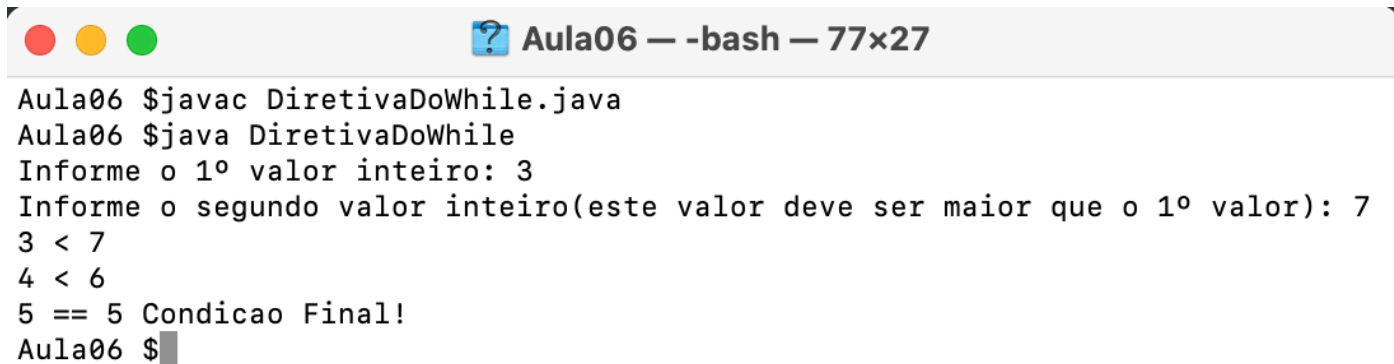
- ▶ O do while é outro laço condicional que repete uma diretiva ou um bloco enquanto a condição é avaliada como verdadeira, mas, diferentemente do while, a instrução instrução associada é executada uma vez antes da avaliação da expressão lógica usada como condição e eventual continuação da repetição;
- ▶ Sua sintaxe é:

```
do {  
    diretiva;  
}  
while ( <condição> );
```




```
1  /*
2  Classe exemplo diretiva do while: DiretivaDoWhile.java
3  IFPR – Campus Cascavel
4  Disciplina: Programação Orientada à Objetos
5  Professor: Nelson Bellincanta
6  Data da criação: 31/03/2023
7  */
8
9
10
11 import java.util.Scanner; //Importa a classe Scanner, que será usada para receber entrada do usuário
12
13 public class DiretivaDoWhile {
14     public static void main (String args[]) {
15         Scanner s = new Scanner(System.in); //Cria uma nova instância da classe Scanner
16         System.out.print("Informe o 1º valor inteiro: "); //Exibe a mensagem ao usuário
17         int min = s.nextInt(); //Lê um número inteiro digitado pelo usuário usando o método nextInt() da classe Scanner
18         System.out.print("Informe o segundo valor inteiro(este valor deve ser maior que o 1º valor): "); //Exibe a mensagem ao usuário
19         int max = s.nextInt(); //Lê um número inteiro digitado pelo usuário usando o método nextInt() da classe Scanner
20         do { //Inicia um laço de repetição "do-while"
21             System.out.println(min + " < " + max); //Imprime uma string formatada contendo os valores de min e max
22             min++; max--; //Incrementa o valor de min em 1 e decrementa o valor de max em 1
23         } while (min < max); //Verifica a condição de continuação do laço
24         System.out.println(min + " == " + max + " Condicao Final!"); //Exibe uma string formatada contendo os valores finais de min e max
25     }
26 }
```

Diretiva do while



```
Aula06 $javac DiretivaDoWhile.java
Aula06 $java DiretivaDoWhile
Informe o 1º valor inteiro: 3
Informe o segundo valor inteiro(esteste valor deve ser maior que o 1º valor): 7
3 < 7
4 < 6
5 == 5 Condicao Final!
Aula06 $
```

Execução da classe DiretivaDoWhile.java

Referências

- ▶ DEITEL, Harvey M.; DEITEL, Paul J. **Java: como programar**. 10ª ed. Editora Pearson Education do Brasil, 2017.
- ▶ FURGERI, Sérgio. **Java 8: Ensino Didático**. 1ª ed., São Paulo: Érica, 2015.
- ▶ JUNIOR, Peter Jandl. **Java Guia do Programador**. 4ª Edição: Atualizado para Java 16. Novatec Editora, 2021.