

IOS – Instituto de
Oportunidade Social

Aula Java 03 - Desvios condicionais – if/else



- > Estrutura de um programa em Java
- > Comando de entrada
- > Estruturas condicionais
 - > Desvio condicional simples (if)
 - > Desvio condicional composto (if/else)

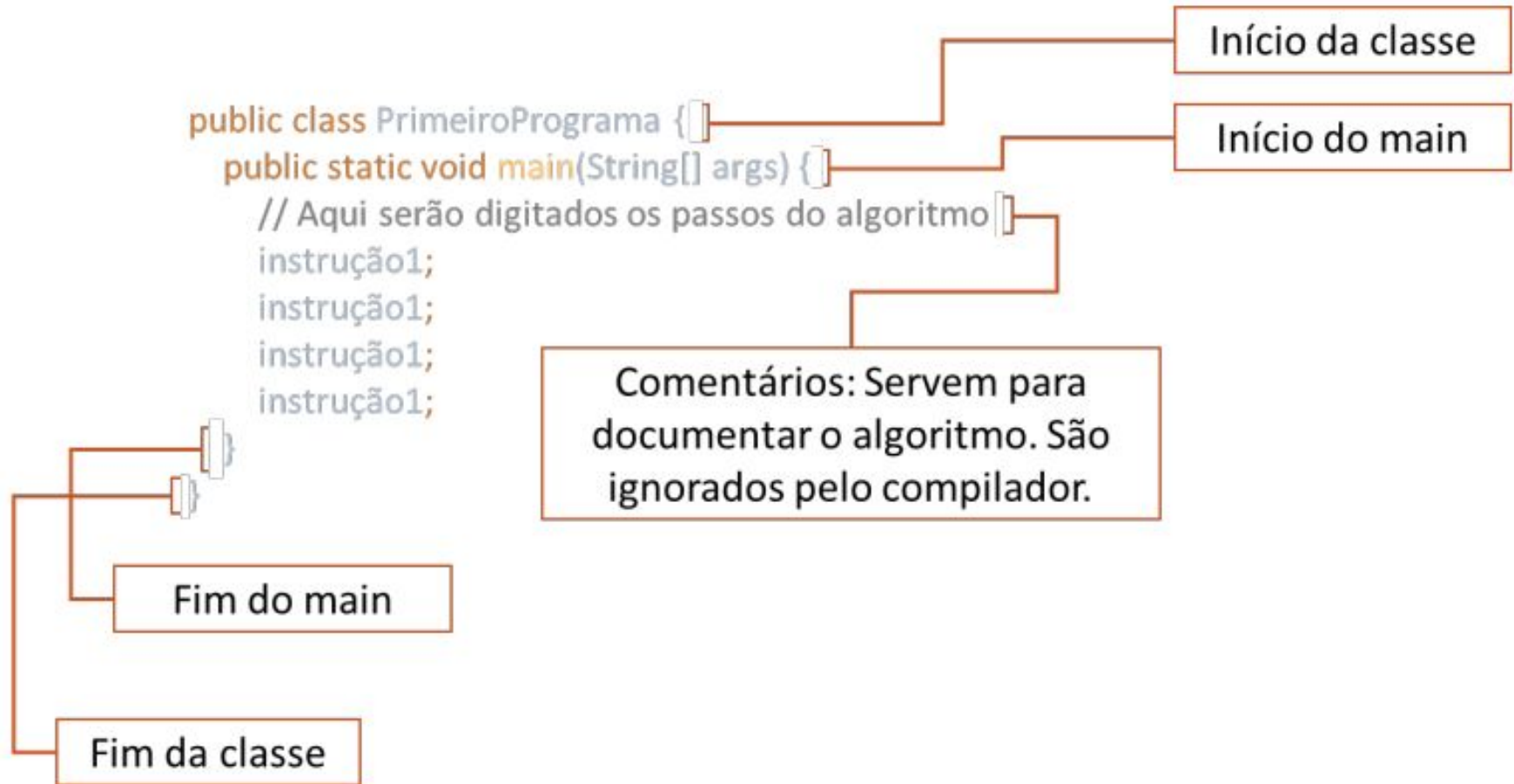
IOS – Instituto de
Oportunidade Social

Estrutura de um programa em Java



Estrutura de um programa em Java

> Comando de entrada



> Comando de entrada

No início do aprendizado de uma linguagem de programação, geralmente, o resultado do processamento de um algoritmo pode ser armazenado em uma variável e/ou pode ser exibido diretamente em um dispositivo de saída. O monitor (console) é o dispositivo de saída padrão de um projeto em Java. Como foi visto o comando de saída utilizado para escrever dados e resultados na saída padrão (monitor/console) é:

```
System.out.println("");
```

Estrutura de um programa em Java

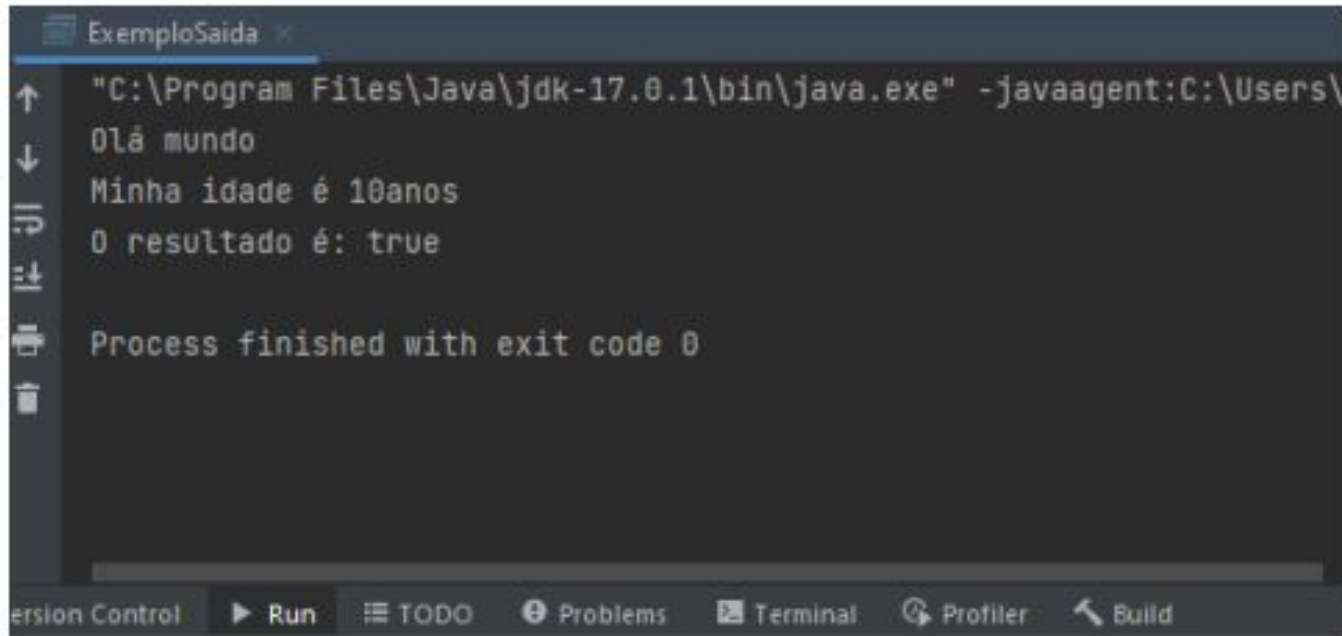
Por exemplo, o código abaixo: bloco

Código deve estar armazenado em um arquivo com o mesmo nome da classe:
ExemploSaida.java

```
public class ExemploSaida {  
  
    public static void main(String[] args) {  
        System.out.println("Olá mundo");  
        System.out.println("Minha idade é " + 10 + " anos");  
        System.out.println("O resultado é: " + (3 + 5 <= 22));  
    }  
}
```

Estrutura de um programa em Java

Produz o seguinte resultado mostrado na tela do computador.



```
ExemploSaida x
↑ "C:\Program Files\Java\jdk-17.0.1\bin\java.exe" -javaagent:C:\Users\
↓ Olá mundo
: Minha idade é 10anos
: O resultado é: true
:
: Process finished with exit code 0
:

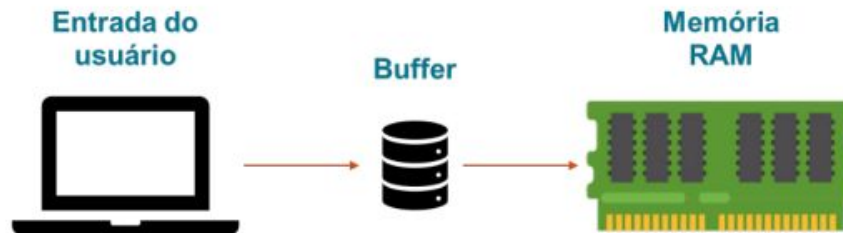
Version Control Run TODO Problems Terminal Profiler Build
```

Além de imprimir os dados e resultado na saída padrão do computador, podemos também ler dados digitado de um teclado e armazená-los na memória do computador por meio de variáveis. Para isso, é necessário criar um buffer para guardar os dados digitados no teclado e depois transferi-los para a memória (variável).



Saiba mais!

Buffer é uma região de memória utilizada para armazenar temporariamente os dados.



Para criar o **buffer** é necessário utilizar a classe **Scanner**:

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



É o buffer do teclado

Após criar o buffer do teclado basta transferir esse dado para a variável. A leitura de um dado é realizada associando o tipo de entrada ao tipo da variável que receberá o dado

Tipo de dado	Usar
String	<code>entrada.nextLine();</code>
Int	<code>entrada.nextInt();</code>
Double	<code>entrada.nextDouble();</code>
Float	<code>entrada.nextFloat();</code>
Char	<code>entrada.next().charAt(0);</code>
Boolean	<code>entrada.nextBoolean();</code>

Para usar a classe **Scanner** é necessário fazer um **import de um pacote antes da definição da classe.**

```
import java.util.Scanner;

public class Exemplo {
    public static void main(String[] args) {
        Scanner exemplo = new Scanner(System.in);
    }
}
```

> Vamos praticar

Vamos fazer um programa em que solicitaremos para o usuário que informe o nome e o valor de duas notas de um aluno. O programa deverá calcular a média dessas notas e depois imprimir a média calculada. Nesse exemplo, precisaremos de três variáveis duas para armazenar as notas digitadas e outra para armazenar a média calculada. Portanto, siga os passos para escrever o nosso programa para calcular a média das notas:

IOS – Instituto de
Oportunidade Social

Estruturas condicionais



Geralmente, as instruções de um programa são executadas sequencialmente uma após a outra, na ordem em que foram escritas. Porém muitas vezes é necessário alterar a sequência da execução de um programa de acordo com o resultado de um teste lógico ou até mesmo repetir um determinado trecho de código por inúmeras vezes. Nesse contexto, as estruturas condicionais permitem decidir o fluxo de execução de programa, selecionando as instruções que serão executadas de acordo com um teste lógico (Desvio condicional).

O Desvio Condicional é usado quando existe a necessidade de verificar condições para executar uma instrução ou um bloco de instruções. Toda condição testada pode retornar dois valores lógicos: true ou false. Vejamos alguns exemplos de condições:

```
(x > y)
```

```
(peso < 50.0)
```

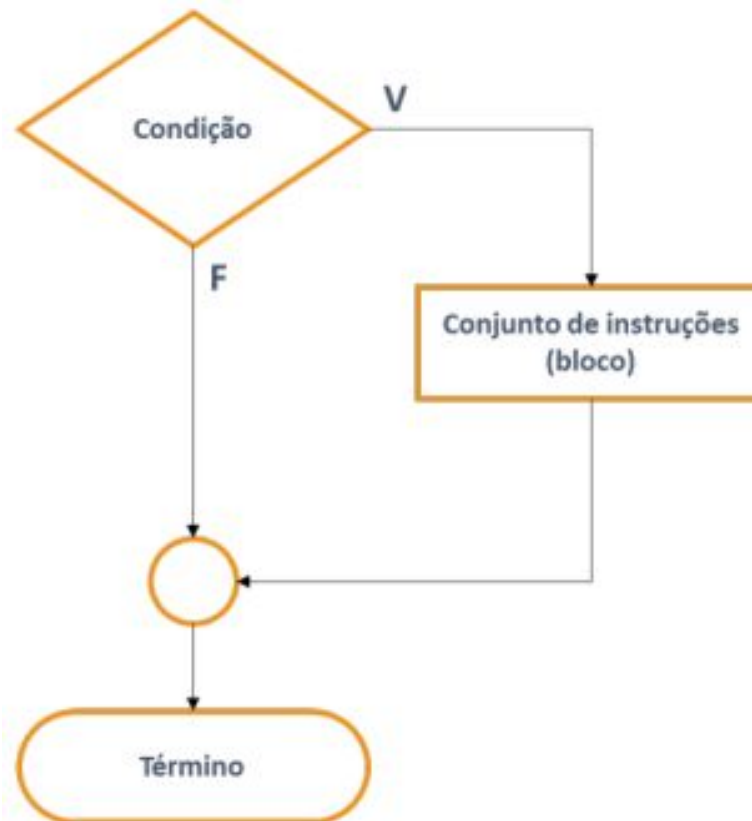
```
((x > 0) && (x < 8))
```

```
((x == 5 && y == 2) || y == 3)
```

> Desvio condicional simples (if)

O desvio condicional simples é usado para verificar se uma dada condição é atendida. Caso a condição seja atendida um determinado conjunto de instruções deverá ser executado. Caso contrário, a condição não for atendida, o fluxo de execução do algoritmo seguirá após o fim do bloco de decisão

> Desvio condicional simples (if)



> Desvio condicional simples (if)

A sintaxe do comando if é:

```
if (condicao){  
    instrucao1;  
    instrucao2;  
    instrucao3;  
}  
proximaInstrucao
```



Início do bloco if

Fim do bloco if

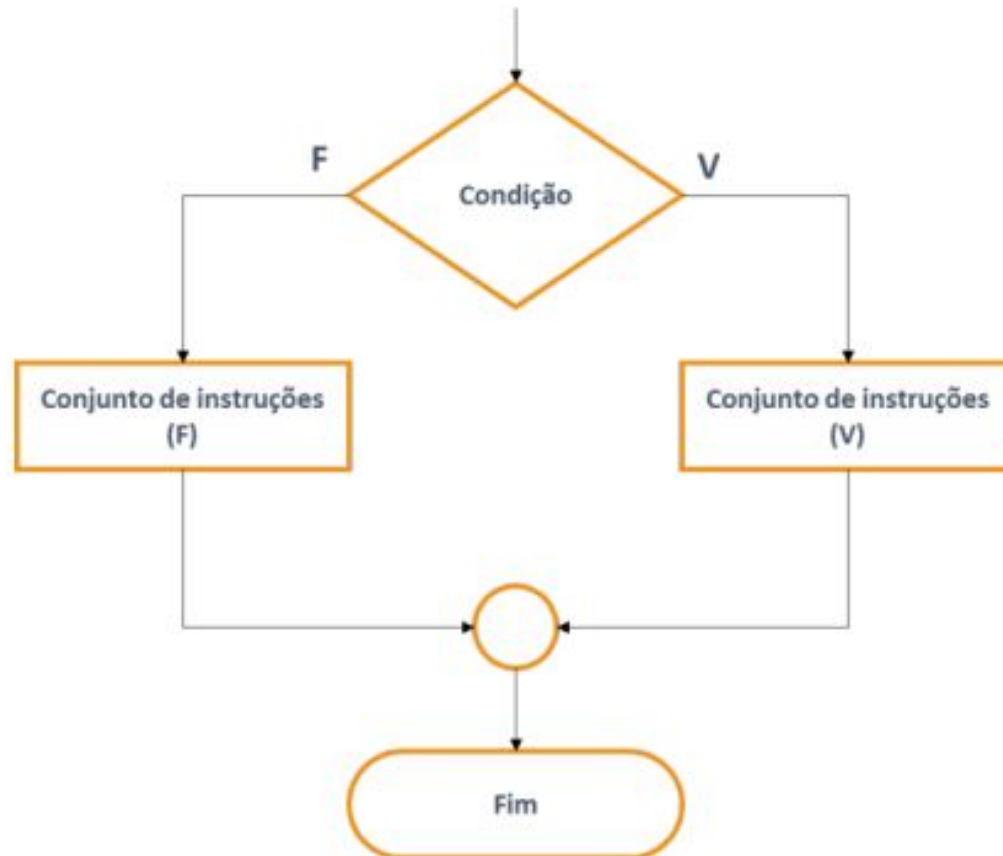
> Vamos praticar!

Vamos criar um programa para verificar o número de pontos de um jogador. Caso o número de pontos for maior ou igual a 1000, o programa deverá imprimir uma mensagem que o jogador passou para a segunda fase.

> Desvio condicional composto (if/else)

O desvio condicional composto prevê dois conjuntos de instruções para serem executados de acordo com a avaliação da condição. Nesse caso, esse um conjunto de instruções para serem executados se a condição for verdadeira e um outro conjunto de instruções para serem executados se a condição for falsa.

> Desvio condicional composto (if/else)



> Desvio condicional composto (if/else)

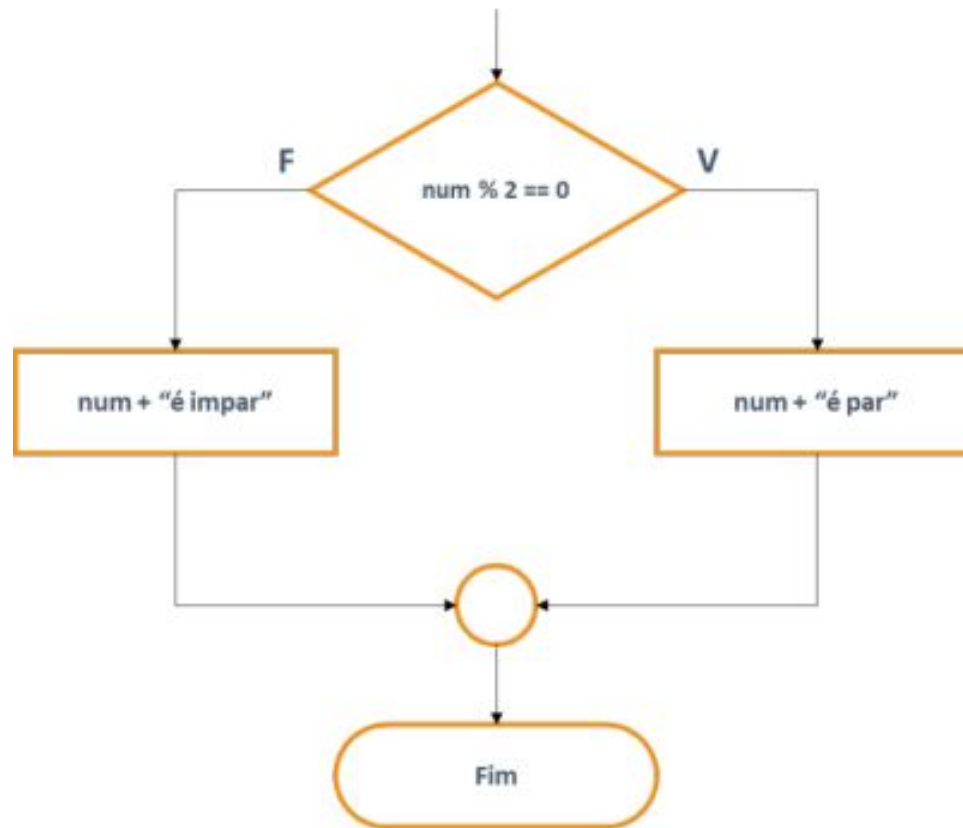
O comando **if/else** possui a seguinte sintaxe:



> Vamos praticar

Vamos criar um programa para verificar se um número digitado pelo teclado é par ou ímpar. Caso o número seja par, o programa deverá imprimir a mensagem que ele é par e, caso o número seja ímpar, o programa deverá imprimir a mensagem que ele é ímpar.

> Desvio condicional composto (if/else)



IOS – Instituto de
Oportunidade Social

Exercícios



Exercício