

ESTDAD - Estrutura de Dados

Aula Introdutória

Engenharia da Computação – 3ª série

Revisão de Java
(E1/E2)

2025

Tipos de Dados

- Caracteres:

- ☐ char (um caractere com 16 bits unicode) ;
- ☐ classe String (cadeia de caracteres);

- Inteiros:

- ☐ byte, short, int e long (8, 16, 32 e 64 bits);

- Ponto flutuante:

- ☐ float e double (32 e 64 bits);

- Lógico:

- ☐ boolean (8bits);

Identificadores

- nomes devem começar com uma letra, um caractere 'sublinha' ou 'underline' (_) ou o símbolo cifrão (\$). Os caracteres subseqüentes podem também incluir números;
- não utilizar caracteres especiais, como acentos, símbolos (? / : @ # etc), ç, entre outros, exceto os acima citados;
- as letras podem ser maiúsculas ou minúsculas;
- não podem ser utilizadas palavras reservadas, como: final, float, for, int, etc;
- Case sensitive;

Declaração de variáveis

- *tipodado nome1, nome2 = valorinicial;*
 - Exemplo: `int x, y, z;`
- *tipodado nome1 = valorinicial;*
 - Exemplo: `int x = 0;`

Operadores

- Atribuição:

- “=”

- Aritméticos:

- “++”, “--”, “+”, “-”, “*”, “/”, “%”

- Lógicos:

- “&&”, “||”, “!”

- Relacionais:

- “<”, “<=”, “>”, “>=”, “==”, “!=”

Estrutura de classes

Formato:

```
class nome {  
    tipodado atributos;  
    public static void main ( String args [ ] ){  
        Instruções;  
    }  
    tiporetorno método(argumentos);  
}
```

Exemplo:

```
class Primeiro {  
    public static void main ( String args [ ] ){  
        System.out.println("Alo Mundo!");  
    }  
}
```

Modificadores

- **public:** pode ser invocado livremente. Indica um método que é visível para qualquer um que enxergue a classe;
- **protected:** pode ser utilizado apenas no mesmo pacote e em subclasses;
- **private:** pode ser invocado apenas na classe;
- **final:** não pode ser sobrescrito. Equivale à declaração de constante;
- **static:** não necessita de objeto. Pode ser invocado a partir do nome da classe. Por exemplo:
`Integer.parseInt(<String>)`

Transformação de tipos

- Para inteiro:
 - *destino* = Integer.parseInt(*origem*);
- Para float:
 - *destino* = Float.parseFloat(*origem*);
- Para Double:
 - *destino* = Double.parseDouble(*origem*);

Saída de dados (console)

```
import java.io.*;
```

```
...
```

```
System.out.print( literal ou variável );
```

```
System.out.println( literal ou variável );
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo

```
import java.io.*;

public class EXTransforma{
    public static void main ( String args [] ) {
        String s1 = "1", s2 = "5";

        int n1 = Integer.parseInt(s1);
        int n2 = Integer.parseInt(s2);

        int tot = n1+n2;

        String saida = "Soma:" + n1 + " + " + n2 + " = " + tot + "\nFim.";
        System.out.print(saida);
    }
}
```

Entrada de dados (console)

```
import java.io.*;
...
BufferedReader nomebuffer;
Nomebuffer = new BufferedReader
                ( new InputStreamReader (System.in));

...
try {
    ...
    nomevar = entrada.readLine();
    ...
}
catch (Exception e) {
    System.out.println("Ocorreu um erro durante a leitura !");
}
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo 1

```
import java.io.*;
class ExlOa {
    public static void main ( String args [ ] ){
        BufferedReader entrada;
        entrada = new BufferedReader(
                        new InputStreamReader (System.in) );
        String nome;
        try {
            System.out.println("Qual o seu nome ? ");
            nome = entrada.readLine();
            System.out.println(nome);
        } catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura !");
        }
    }
}
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo 2

```
import java.io.*;
class ExIOb {
    public static void main ( String args [ ] ){
        BufferedReader entrada;
        entrada = new BufferedReader(new
            InputStreamReader(System.in));

        try{
            System.out.print("Entre com dois valores inteiros:\n");
            int n1 = Integer.parseInt(entrada.readLine());
            int n2 = Integer.parseInt(entrada.readLine());
            int tot = n1+n2;
            String saida = "Resposta: "+tot+"\n";
            System.out.print(saida);
        }
        catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura!");
        }
    }
}
```

Entrada de Dados (alternativa)

```
import java.io.*;  
import java.util.Scanner;  
...  
Scanner NomeScanner = new Scanner(System.in);  
...  
TipoVar NomeVar = NomeScanner.nextTipoVar( );  
...
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo 1

```
import java.io.*;
import java.util.Scanner;

class ExIOalternativo1 {
    public static void main ( String args [ ] ){
        Scanner entrada = new Scanner(System.in);
        String nome;

        try {
            System.out.println("Qual o seu nome ? ");
            nome = entrada.nextLine();
            System.out.println(nome);
        }
        catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura!");
        }
    }
}
```


ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo 2

```
import java.io.*;
import java.util.Scanner;

class ExIOalternativo2
{
    public static void main ( String args [ ] ) {
        Scanner entrada = new Scanner(System.in);
        try {
            System.out.print("Entre com dois valores inteiros:\n");
            int n1 = entrada.nextInt();
            int n2 = entrada.nextInt();
            int tot = n1+n2;
            String saida = "Resposta: "+tot+"\n";
            System.out.print(saida);
        }
        catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura!");
        }
    }
}
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo 3

```
import java.io.*;
import java.util.Scanner;

class ExIOalternativo3{
    public static void main ( String args [ ] ) {
        Scanner entrada = new Scanner(System.in);
        try {
            System.out.println("Digite um valor inteiro:");
            while(!entrada.hasNextInt()){
                entrada.nextLine();
                System.out.println("O valor não é Inteiro, tente de novo:");
            }
            int n = entrada.nextInt();
            String saida = "Ok, "+n+" é um valor inteiro!";
            System.out.println(saida);
        }
        catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura!");
        }
    }
}
```

Entrada e Saída (GUI)

```
import javax.swing.JOptionPane;  
  
...  
nomevar = JOptionPane.showInputDialog(mensagem);  
  
...  
JOptionPane.showMessageDialog  
    (null, mensagem, JOptionPane.PLAIN_MESSAGE);
```

↑
Indica a posição onde
será apresentado o
painel

↑
Indica que não será
apresentado ícone de
informação

ESTDAD - Estrutura de Dados

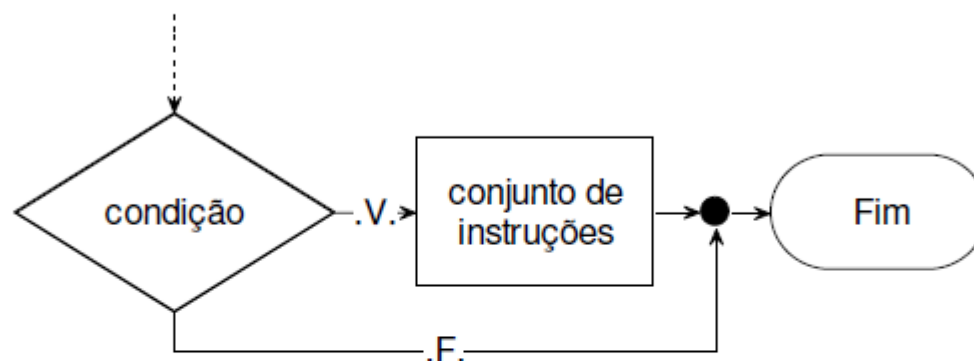
Revisão de Java

Exemplo

```
import javax.swing.JOptionPane;

class ExLOGUI {
    public static void main ( String args [ ] ){
        try {
            String n1=JOptionPane.showInputDialog("Digite um número:");
            int num1 = Integer.parseInt(n1);
            String n2=JOptionPane.showInputDialog("Digite outro:");
            int num2 = Integer.parseInt(n2);
            int tot=num1+num2;
            JOptionPane.showMessageDialog(null,"A soma é:      " + tot,
                "Output",JOptionPane.PLAIN_MESSAGE);
        }
        catch (Exception e) {
            JOptionPane.showMessageDialog(null,"Houve um erro na leitura!",
                "Output",JOptionPane.PLAIN_MESSAGE);
        }
    }
}
```

Estruturas de Decisão Simples

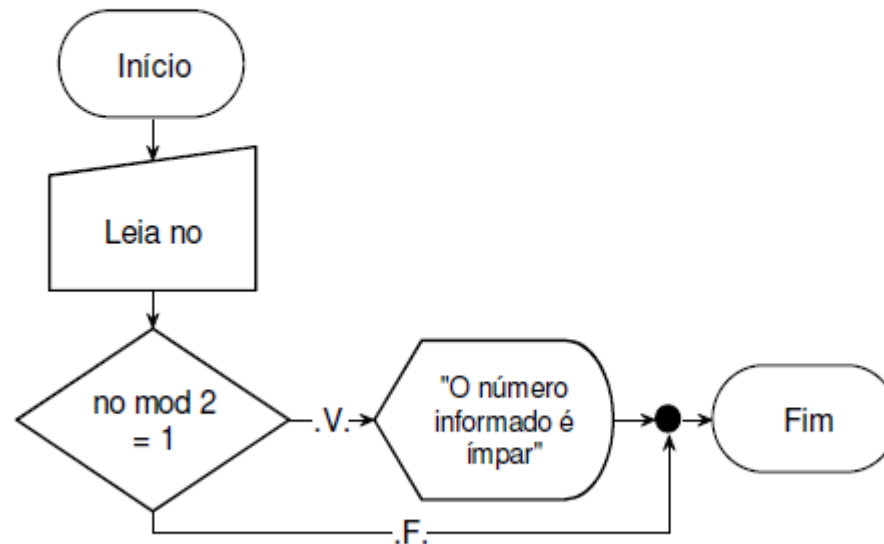


```
if (condição) {  
    instruções;  
}
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo



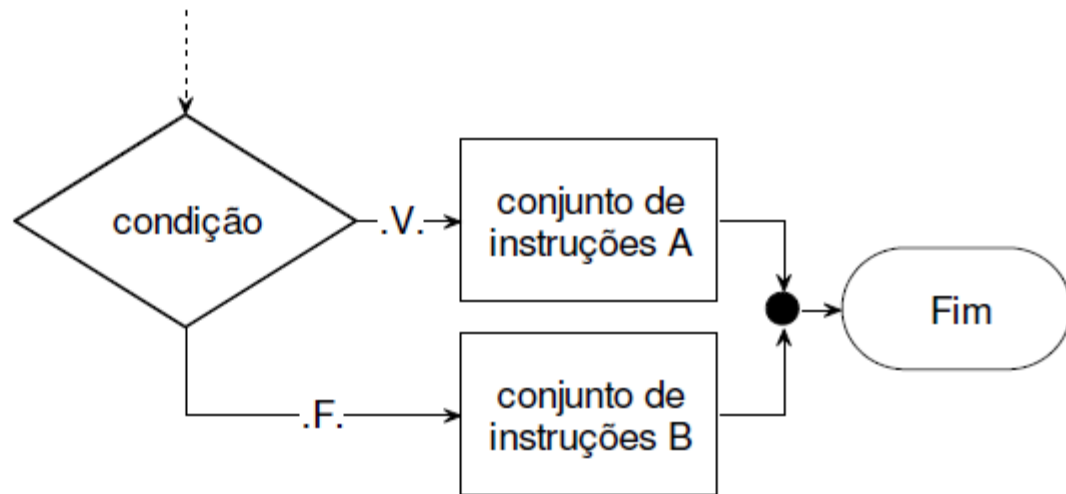
ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo

```
import java.io.*;
class NumImpar {
    public static void main ( String args [ ] ){
        BufferedReader entrada;
        entrada = new BufferedReader( new InputStreamReader (System.in) );
        int numero;
        try {
            System.out.println("Qual o número ? ");
            numero = Integer.parseInt( entrada.readLine() );
            if ( numero % 2 == 1) {
                System.out.println ( " O número informado é ímpar " ) ;
            }
        } catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura !");
        }
    }
}
```


Estruturas de Decisão Composta

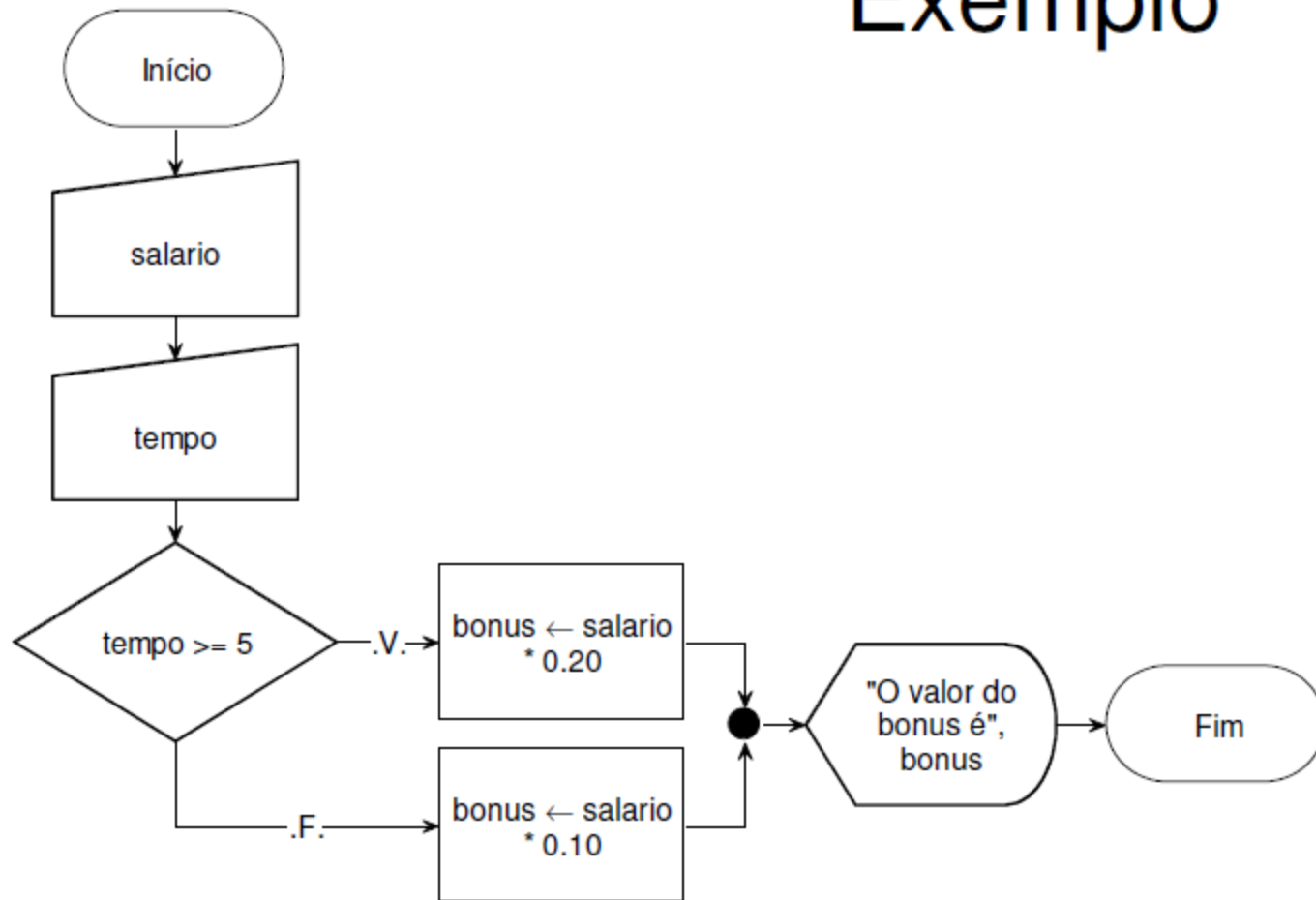


```
if (condição) {  
    instruçõesA;  
} else {  
    instruçõesB;  
}
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo



ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo

```
import java.io.*;
class Premio {
    public static void main ( String args [ ] ){
        BufferedReader entrada;
        entrada = new BufferedReader( new InputStreamReader (System.in) );
        float salario, bonus;
        int tempo;
        try {
            System.out.println("Qual o salário ? ");
            salario = Float.parseFloat( entrada.readLine() );
            System.out.println("Quanto tempo esta na empresa ? ");
            tempo = Integer.parseInt( entrada.readLine() );
            if ( tempo >= 5) {
                bonus = salario * 0.20f;
            } else {
                bonus = salario * 0.10f;
            }
            System.out.println ("O valor do bônus e : " + bonus) ;
        } catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura !");
        }
    }
}
```

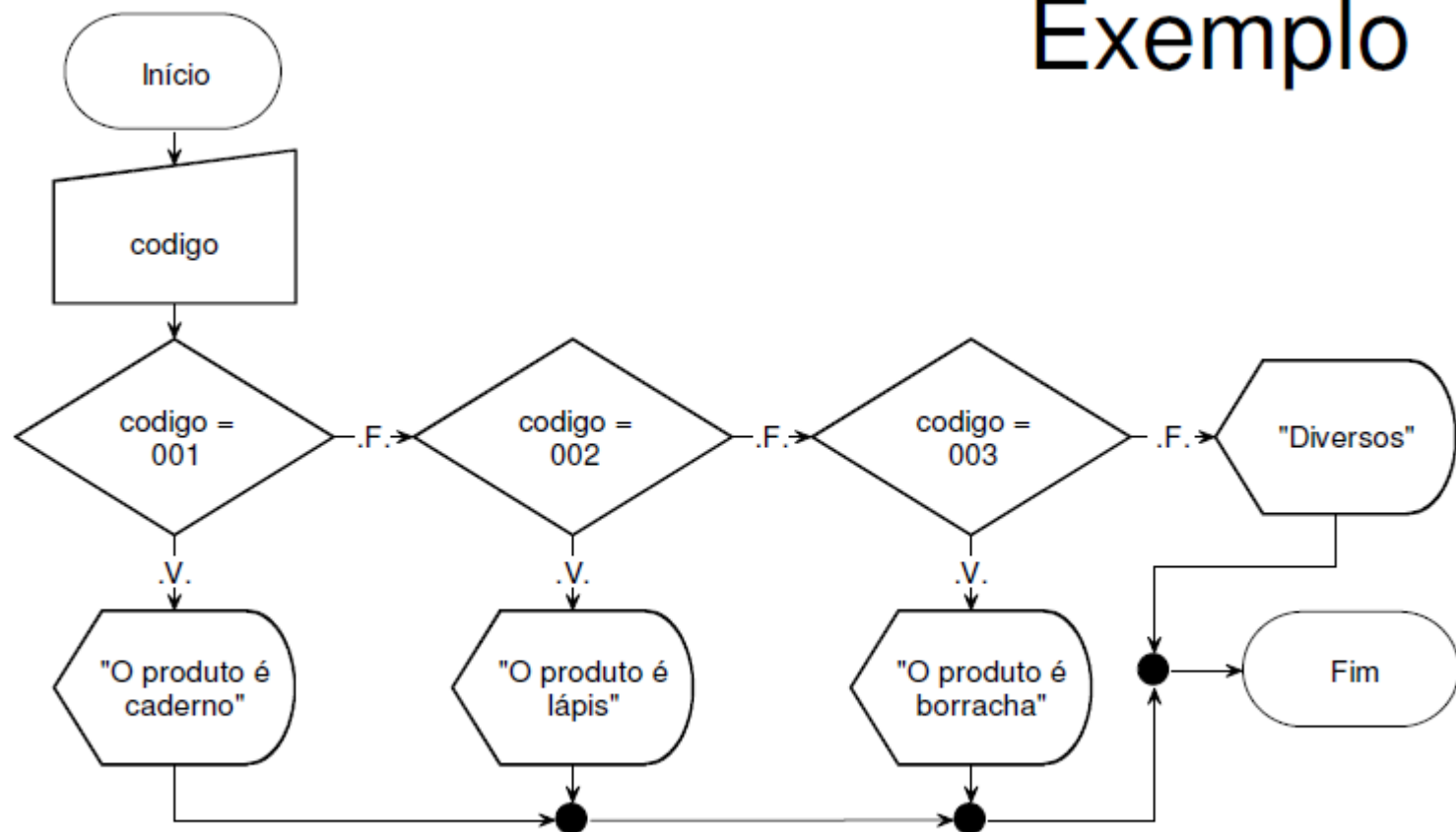
Estruturas de Decisão Múltipla

```
switch (variável){  
    case valor1 : instruçãoA;  
        break;  
    case valor2 : instruçãoB;  
        break;  
    case valor1 : instruçãoC;  
        break;  
    default : instruçãoD  
}
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo



ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo

```
import java.io.*;
class Produto {
    public static void main ( String args [ ] ){
        BufferedReader entrada;
        entrada = new BufferedReader( new InputStreamReader (System.in) );
        int codigo;
        try {
            System.out.println("Digite o código : ");
            codigo = Integer.parseInt( entrada.readLine() );
            switch (codigo) {
                case 001 : System.out.println ("Caderno") ;
                           break;
                case 002 : System.out.println ("Lápis") ;
                           break;
                case 003 : System.out.println ("Borracha") ;
                           break;
                default : System.out.println ("Diversos") ;
            }
        } catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura !");
        }
    }
}
```

Estruturas de Repetição (while)

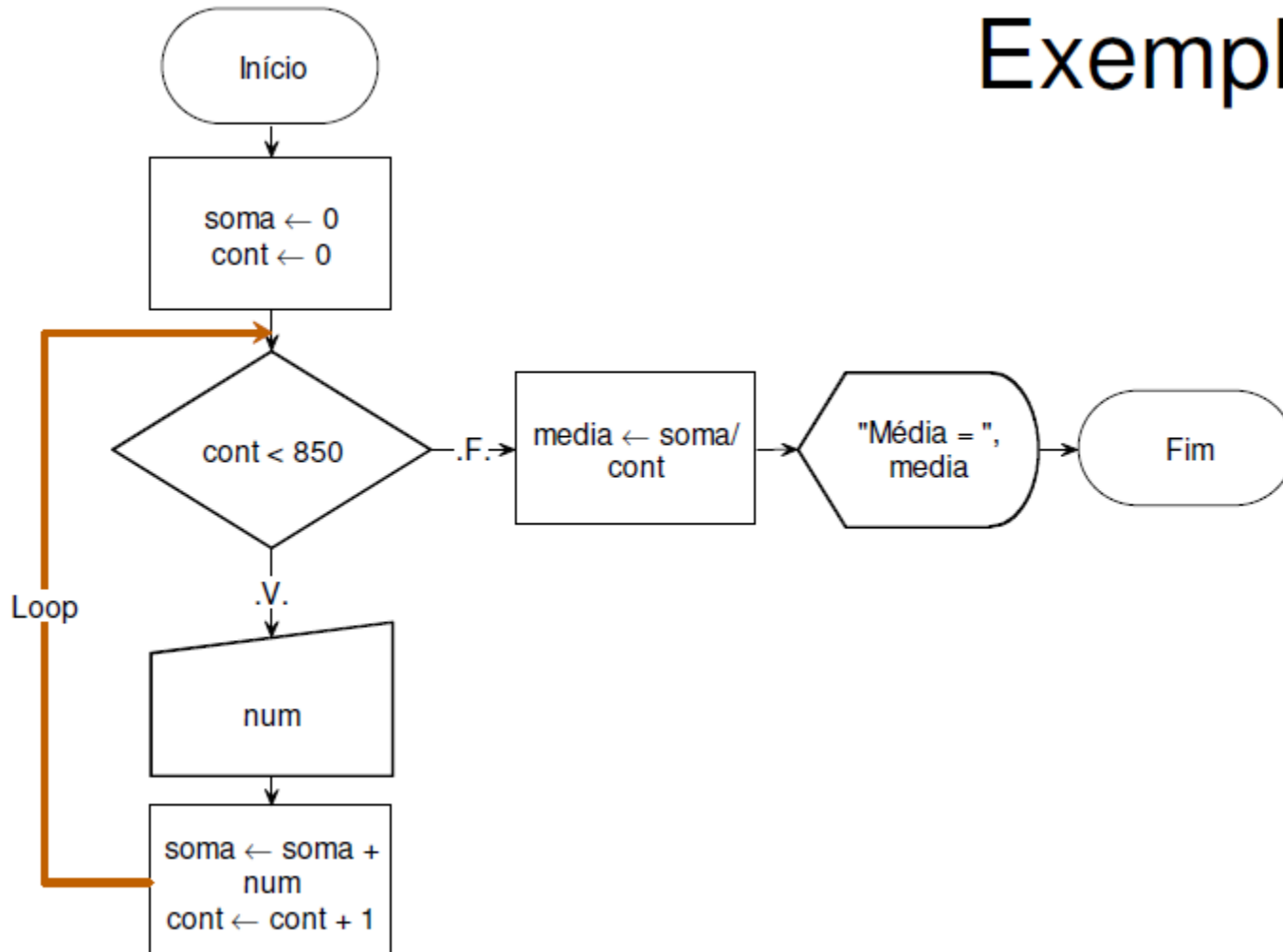
- Estrutura Enquanto: a condição de repetição é verificada antes de entrar no laço.

```
while (condição) {  
    instruções  
}
```


ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo



ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo

```
import java.io.*;
class ExEnquanto {
    public static void main ( String args [ ] ){
        BufferedReader entrada;
        entrada = new BufferedReader( new InputStreamReader (System.in) );
        float numero, media, soma;
        int cont;
        cont = 0;
        soma = 0f;
        try {
            while (cont < 850) {
                System.out.println("Digite o número : ");
                numero = Float.parseFloat( entrada.readLine() );
                soma = soma + numero;
                cont = cont +1;
            }
            media = soma / cont;
            System.out.println("Média = " + media);
        } catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura !");
        }
    }
}
```

Estrutura Repetição (do/while)

- Estrutura Repita: permite que um ou mais comandos sejam executados repetidamente *até* uma condição específica tornar-se verdadeira.

```
do {  
    instruções  
while (condição);
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo

```
import java.io.*;
class ExEnquanto {
    public static void main ( String args [ ] ){
        BufferedReader entrada;
        entrada = new BufferedReader( new InputStreamReader (System.in) );
        float numero, media, soma;
        int cont;
        cont = 0;
        soma = 0f;
        try {
            do {
                System.out.println("Digite o número : ");
                numero = Float.parseFloat( entrada.readLine() );
                soma = soma + numero;
                cont = cont +1;
            } while (cont < 850)
            media = soma / cont;
            System.out.println("Média = " + media);
        } catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura !");
        }
    }
}
```

Estrutura de Repetição (for)

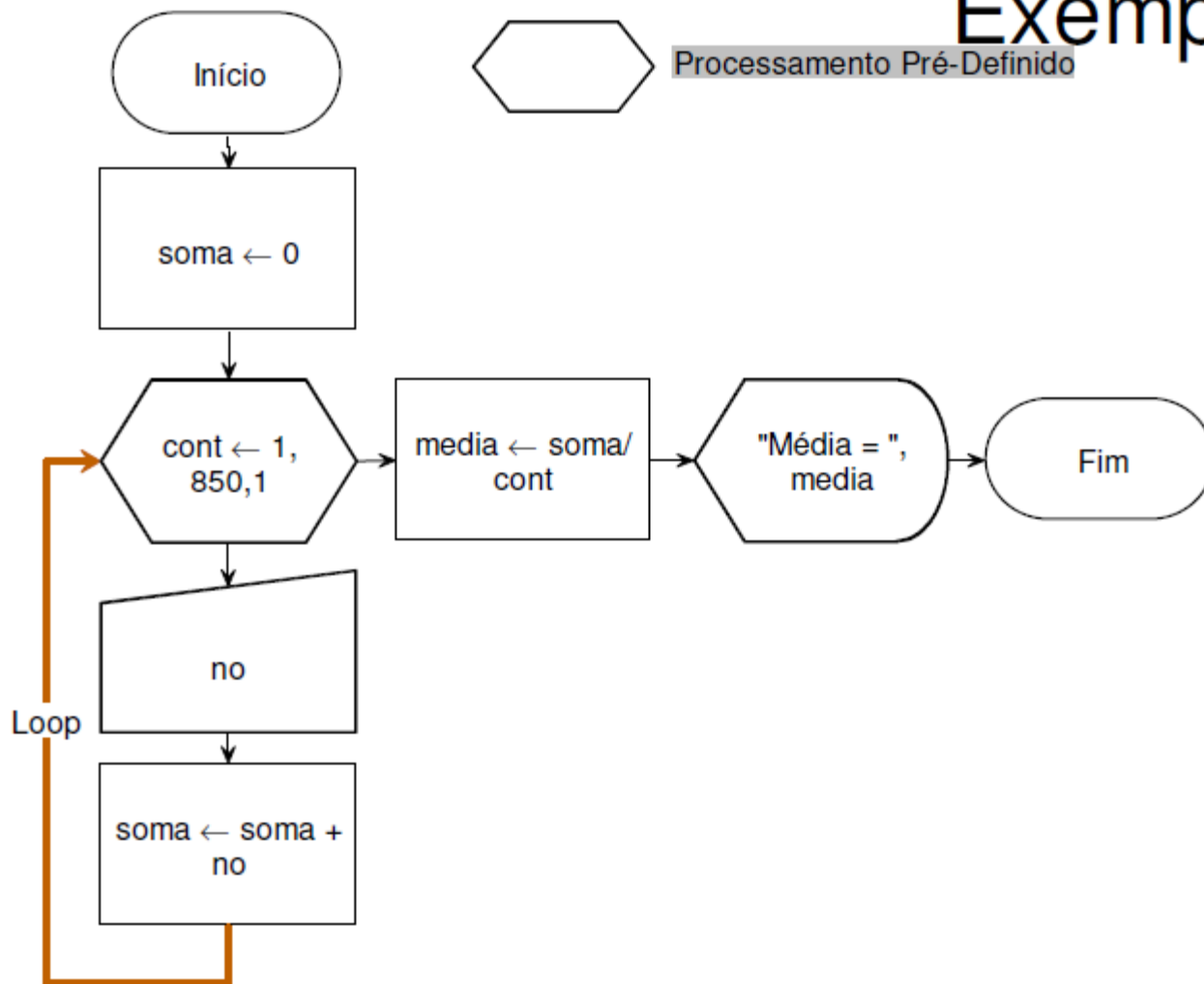
- Estrutura Para: utiliza variáveis de controle que definem exatamente o número de vezes que a seqüência de instruções será executada.

```
for(contador = valorinicial; condição; incremento){  
    instruções  
}
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo



ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo

```
import java.io.*;
class ExPara {
    public static void main ( String args [] ){
        BufferedReader entrada;
        entrada = new BufferedReader( new InputStreamReader (System.in) );
        float numero, media, soma;
        int cont;
        soma = 0f;
        try {
            for (cont = 0; cont < 3; cont++) {
                System.out.print("Digite o número : ");
                numero = Float.parseFloat( entrada.readLine() );
                soma = soma + numero;
            }
            media = soma / cont;
            System.out.println("Média = " + media);
        } catch (Exception e) {
            System.out.println("Ocorreu um erro durante a leitura !");
        }
    }
}
```


Métodos

```
modificador tiporetorno nome (argumentos){  
    tiporetorno nomevarretorno;  
    instruções;  
    return nomevarretorno; }
```

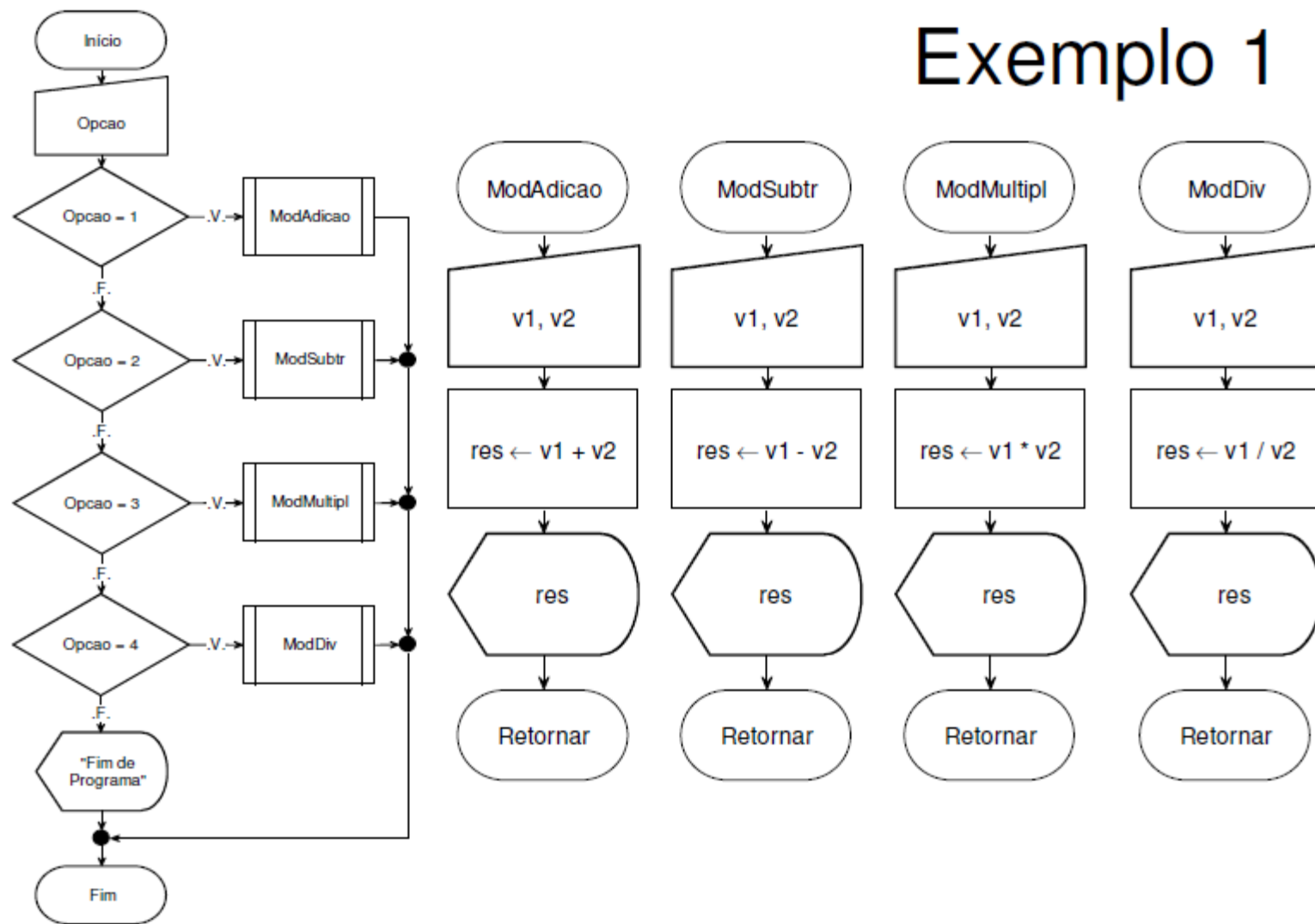
- **nome:** é um Identificador válido da linguagem;
- **argumentos:** lista de argumentos que serão passados como parâmetros para o método. A sintaxe dos argumentos é a de declaração de variáveis;
- **tiporetorno:** indica o tipo do valor retornado pelo método;
- **return:** palavra reservada que indica o valor que será devolvido para o programa;
- **modificadores:** elementos que caracterizam o método quanto à visibilidade (escopo) e qualidade:

Modificadores (relembrando)

- **public:** pode ser invocado livremente. Indica um método que é visível para qualquer um que enxergue a classe;
- **protected:** pode ser utilizado apenas no mesmo pacote e em subclasses;
- **private:** pode ser invocado apenas na classe;
- **final:** não pode ser sobrescrito. Equivale à declaração de constante;
- **static:** não necessita de objeto. Pode ser invocado a partir do nome da classe. Por exemplo:
`Integer.parseInt(<String>)`

ESTDAD - Estrutura de Dados

Revisão de Java



ESTDAD - Estrutura de Dados

Revisão de Java

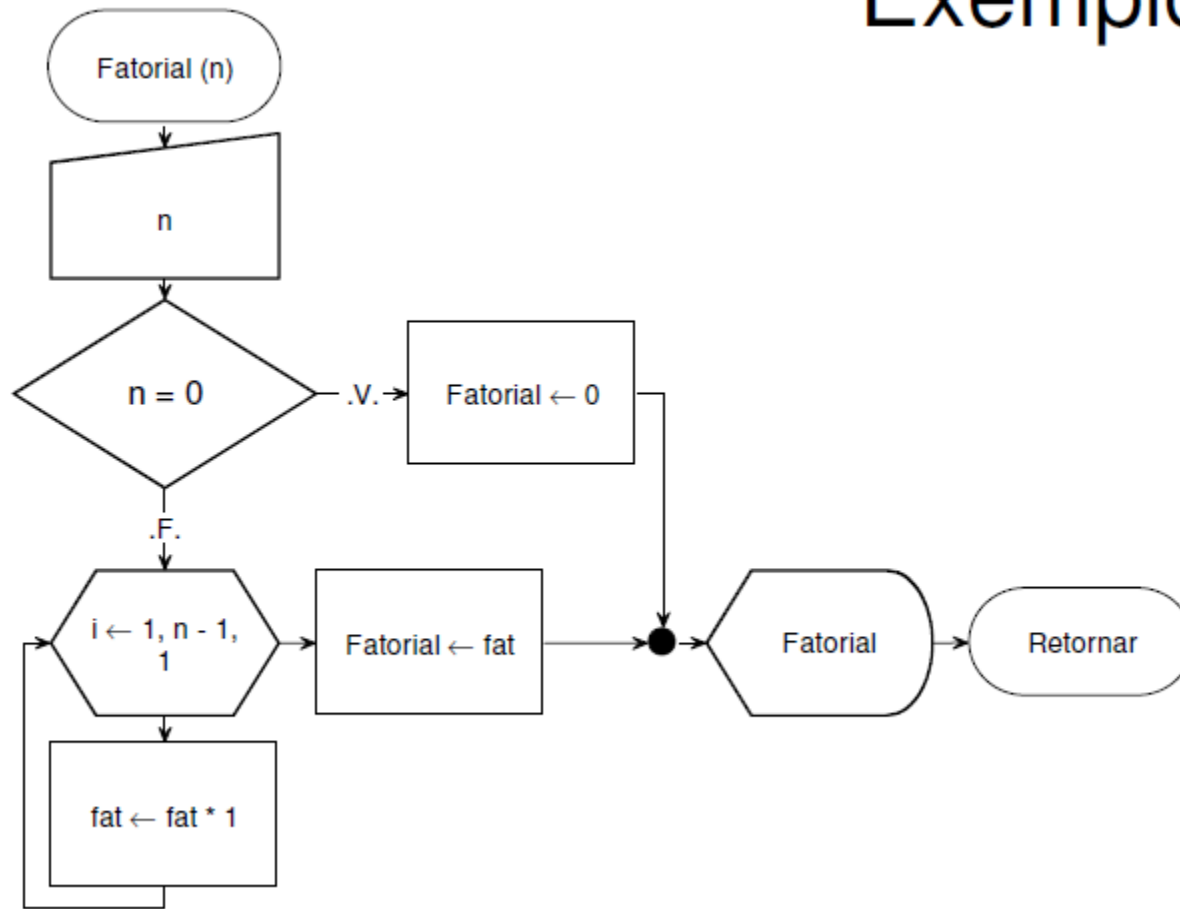
Exemplo 1

```
import java.io.*;
class Exemplo1 {
    public static void main (String args [ ]){
        BufferedReader entrada;
        Entrada = new BufferedReader(new InputStreamReader(System.in));
        Try {
            System.out.println("1 : Adicao ?");
            System.out.println("2 : Subtracao ?");
            System.out.println("3 : Multiplicacao ?");
            System.out.println("4 : Divisao ?");
            System.out.println("Qual a Opcao Desejada?");
            int opcao = Integer.parseInt(entrada.readLine());
            switch (opcao){
                case 1 : modAdicao(); break;
                case 2 : modSubtracao(); break;
                case 3 : modMultiplicacao(); break;
                case 4 : modDivisao();break;
                default : System.out.println("Fim do Programa");
            }
        } catch (Exception erro) {
            System.out.println("Ocorreu um erro de leitura !");
        }
    }
}
```

ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo 2



ESTDAD - Estrutura de Dados

Revisão de Java

Exemplo 2

```
import java.io.*;
class Exemplo2{
    public static void main (String args []){
        BufferedReader entrada;
        entrada = new BufferedReader (new InputStreamReader (System.in));
        try{
            System.out.println ("Qual Numero ?");
            int numero = Integer.parseInt (entrada.readLine());
            int fat = fatorial (numero);
            System.out.println ("fatorial = " + fat);
        }catch (Exception erro){
            System.out.println ("Ocorreu um erro de leitura !");
        }
    }
    static int fatorial (int num){
        int fat = 1;
        for (int i = 1; i <= num; i++){
            fat = fat * i;
        }
        return fat;
    }
}
```


ESTDAD - Estrutura de Dados

Aula Introdutória

FIM