

Qualificação Profissional de Assistente de
Desenvolvimento de Sistemas

LÓGICA DE PROGRAMAÇÃO

**GEEaD - Grupo de Estudos de
Educação a Distância
Centro de Educação Tecnológica
Paula Souza
São Paulo – SP, 2019**

Expediente

PROGRAMA NOVOTEC VIRTUAL
GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
QUALIFICAÇÃO PROFISSIONAL DE ASSISTENTE DE DESENVOLVIMENTO DE SISTEMAS
LÓGICA DE PROGRAMAÇÃO

PÚBLICO ALVO: ALUNOS DA 3ª SÉRIE DO ENSINO MÉDIO
TEMPO DE INTEGRALIZAÇÃO: 34 SEMANAS

Autores:

*Eliana Cristina Nogueira Barion
Marcelo Fernando Iguchi
Paulo Henrique Mendes Carvalho
Rute Akie Utida*

Revisão Técnica:

Sandra Maria Leandro

Revisão Gramatical:

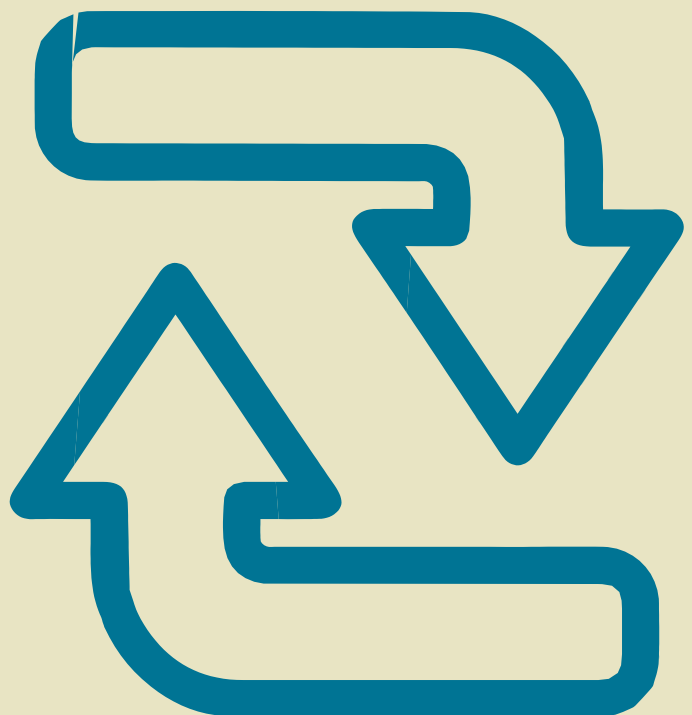
Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:

Flávio Biazim

AGENDA 6

ESTRUTURAS DE REPETIÇÃO I





MERGULHANDO NO TEMA...



Na agenda anterior, você estudou as Estruturas de Decisão, também conhecidas por desvios condicionais.

Além dos desvios condicionais, é possível criar desvios em loop ou repetição, ou seja, repetir trechos do algoritmo sob determinada condição e controlar a forma com que serão executados. Nesta unidade você conhecerá a primeira Estrutura de Repetição, o comando para...fim-para. Na próxima agenda você conhecerá outros dois laços. Preparado?

Além dos desvios sequenciais, é possível criar desvios em loop ou repetição, ou seja, repetir trechos do algoritmo sobre determinada condição e controlar a forma com que serão executados.

Para iniciar seus estudos, assista à videoaula do professor Sandro Valérius, que apresenta esse Laço de Repetição.



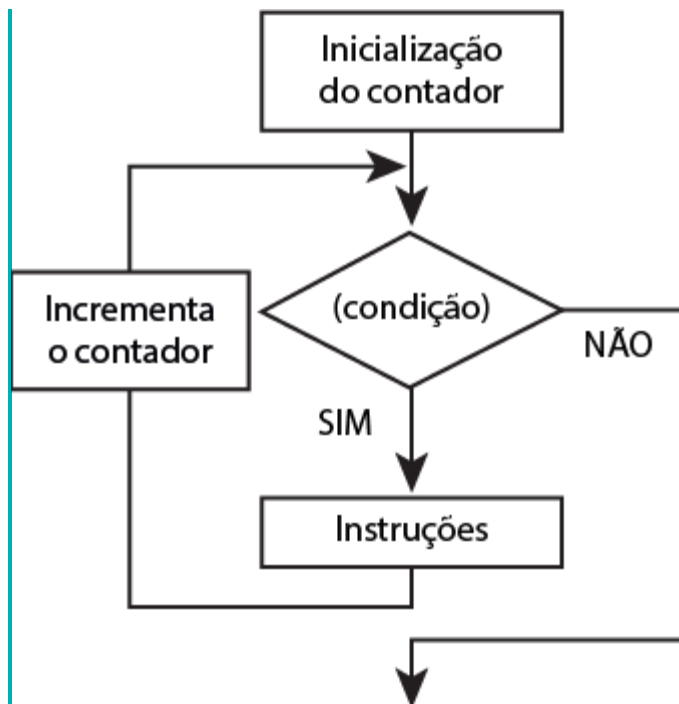
Você viu que o comando “para...fim-para” permite que uma **variável realize a contagem do número de repetição a executar**, conforme a indicação inicial e final dessa contagem e, também, indique o formato em que essa tarefa será realizada. Observe o quadro a seguir e veja as sintaxes em pseudocódigo e em Java, perceba como são parecidas:

PSEUDOCÓDIGO

FLUXOGRAMA

JAVA

Para {variável} = <valor inicial>
até <valor final>
passo <argumento>
faça {comando(s)}
fim-para



```

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



Observe que no Fluxograma aparece a palavra **contador** e tanto no Pseudocódigo quanto no Java não aparecem. Calma! Ela aparece, porém de outra forma. Veja que no Pseudocódigo tem **{variável}** e no Java aparece **i**. Ambos são os contadores que aparecem no Fluxograma.

Conheça um exemplo:

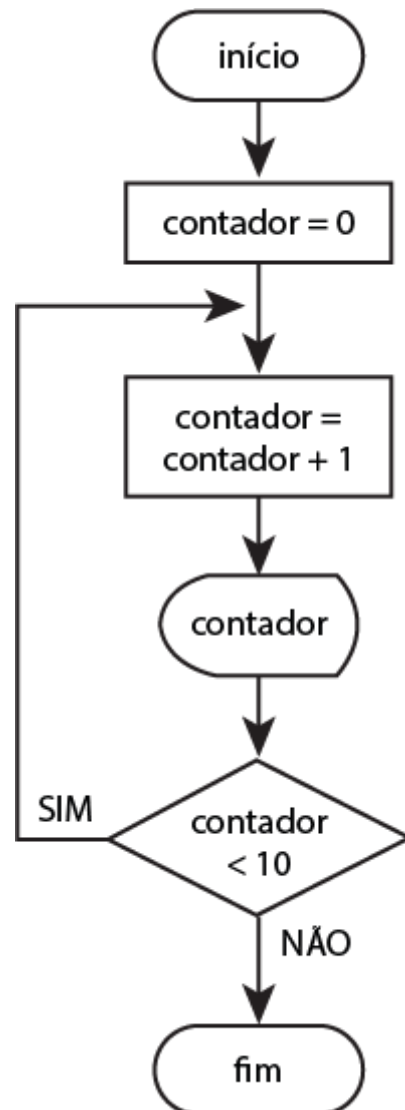
A execução é muito fácil. Para facilitar a compreensão, veja um exemplo de um programa que mostra ao usuário os números de 0 a 9 na tela:



PSEUDOCÓDIGO

Programa mostrar números de 0
ate 9
Declare
 contador como inteiro
Início
 contador \leftarrow 0;
 Para contador de 0 **ate** 9
 passo 1 faça
 Escreva (contador)
 Fim para
Fim.

FLUXOGRAMA



Apenas para você entender melhor, o pseudocódigo inicia-se com a declaração da variável contador com tipo de dados inteiro, ou seja, esta variável irá receber apenas números inteiros. Após a leitura desta variável, iniciada com o valor 0, ocorre o incremento de mais 1 por meio da instrução passo até o valor 10. Esse tipo de estrutura de repetição pode ser usado toda vez que houver a necessidade de repetir trechos finitos, em que se conhece os valores inicial e final.

Dessa forma serão impressos os números de 0 a 9, isso porque a condição é que o contador seja menor que 10 (contador < 10).

No exemplo anterior, tanto no Pseudocódigo quanto no Fluxograma, não fazemos a contagem até o número 10. Por que? Você estudou em unidades anteriores os operadores (aritméticos, comparação e lógicos), então é momento de retomar conceitos.

Observe que a condição apresentada no exemplo é:

- Pseudocódigo: Para contador \leftarrow 0 até 09.
- Fluxograma: contador < 10.

Isso porque de 0 até 9, temos 10 contagens:

Quando o contador está em 0, temos a 1ª contagem (o programa realiza o trecho do algoritmo pela 1ª vez)

Quando o contador está em 1, temos a 2ª contagem.

Quando o contador está em 2, temos a 3ª contagem.

E assim sucessivamente, até que, quando o contador chega em 9, o programa estará contando pela 10ª vez.

Se estivéssemos usando o operador <= (igual) em lugar de < (menor), então o programa totalizaria **11 contagens**.

Veja agora o código em Java:

```
package lacos;

import javax.swing.JOptionPane;

public class lacofor {

    public static void main(String[] args){
        //declaração das variáveis.
        int i; //variável que será utilizada para o laço de repetição for.

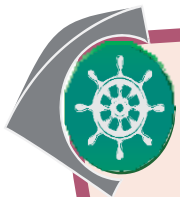
        for (i=0; i<10; i++){
            JOptionPane.showMessageDialog(null, i); // saída de dados.
        } // fecha o laço de repetição for.

    } // fecha o método.

} // fecha a classe.
```

Correspondendo ao Pseudocódigo informado anteriormente, no Java, declaramos as variáveis que iremos utilizar. Neste caso, apenas o *i* como inteiro. Em seguida, iniciaremos o laço de repetição **for** com a variável inicial 0 (zero), variável final 9 (nove) e o incremento com 1. Ou seja, o resultado irá mostrar as sequências dos números 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9.

Para entender melhor, assista à videoaula do Prof. Sandro Valérius



VOCÊ NO COMANDO

Pense em como poderíamos elaborar um programa utilizando a estrutura para...fim-para em que o usuário mostre os números de 01 até 10 na tela. Reflita como podemos resolver essa questão antes de prosseguir a leitura.

Agora, dê uma olhada em um **pseudocódigo** que resolve essa questão:

Programa mostrar números

Programa mostrarnumeros

declare

num como inteiro

Início

Para num <= 1 ate 10 passo 1 faça

escreva(num)

Fim-para

Fim.



Veja, então, o Java:


```
public class ex02 {

    public static void main(String[] args){

        System.out.println("Os números de 0 a 10 são:");

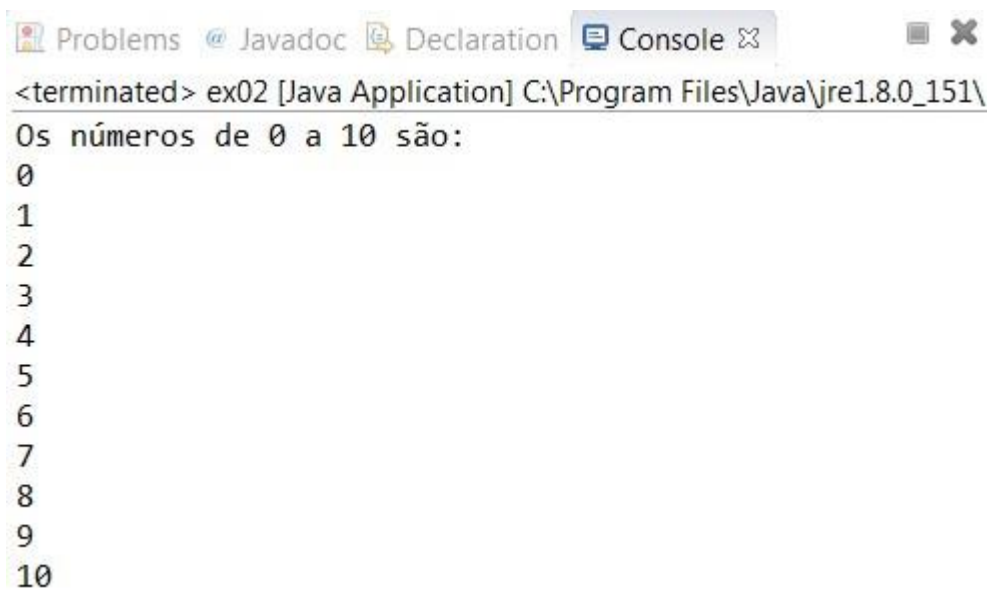
        for (int i=0; i<=10; i++){ //laço de repetição for.

            System.out.println(i); // saída de dados
        }

    }

}
```

E este é o resultado:



The screenshot shows a Java IDE window with the 'Console' tab selected. The title bar indicates the application is 'ex02 [Java Application]' running at 'C:\Program Files\Java\jre1.8.0_151\'. The console output is as follows:

```
<terminated> ex02 [Java Application] C:\Program Files\Java\jre1.8.0_151\
Os números de 0 a 10 são:
0
1
2
3
4
5
6
7
8
9
10
```

Imagem 05— Prompt de comando do Java exibindo os números de 0 a 10 (um por linha).

Observe que apenas usamos a sintaxe **System.out.println** nas linhas 6 e 8 do código para que a informação apareça no Console. Além disso, a estrutura do FOR iniciou com valor 0 e foi até 10 com passos 1 (linha 7 do código), ou seja, ele andou de um em um, como mostra a tela do resultado.

O Laço de Repetição “para...fim-para” é muito simples e fácil de usar. Porém se você quiser incrementar o seu uso conseguimos utilizá-lo junto com a estrutura “se...senão...fim_se”.



VOCÊ NO COMANDO

Agora imagine uma situação em que você precise calcular a média de 30 alunos e informar a situação de cada um deles, ou seja, informar a média como também se cada um dos 30 alunos estão “Aprovados” ou “Reprovados”. Reflita sobre este novo conceito antes de prosseguir com a leitura.

Agora, analise o pseudocódigo e o fluxograma a seguir para conferir se você acertou.

PSEUDOCÓDIGO

Programa mostrar media e situacao

Declare

n1, n2, media como real
contador como inteiro

Início

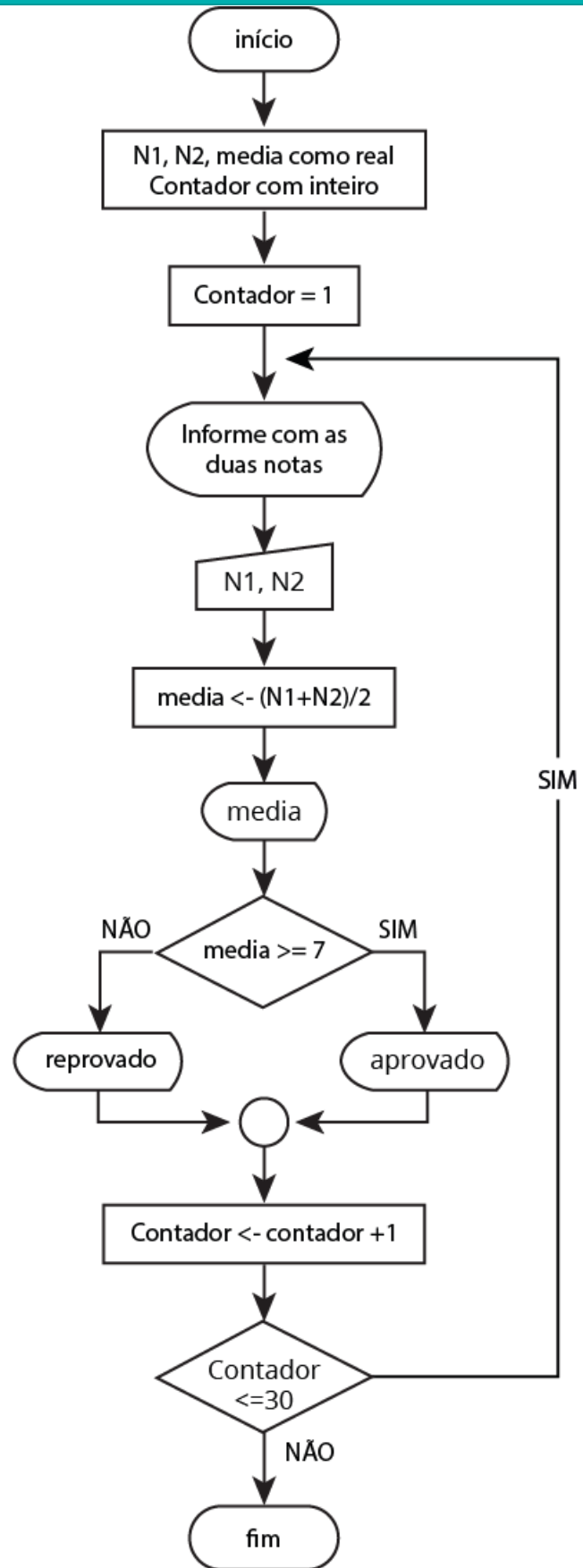
Para contador = 1 até 30 **passo 1 faça**
 escreva ("Informe sua primeira nota:")
 leia (nota1)
 escreva ("Informe sua primeira nota:")
 leia (nota2)
 $media \leftarrow (n1 + n2)/2$
 escreva ("A sua média é: ", media)
 se (media $\geq 7,0$) **então**
 escreva ("Aprovado")
 senão
 escreva ("Reprovado")

Fim-se

Fim-para

Fim.

FLUXOGRAMA



Confira o programa em Java:

```

import java.util.Scanner;
public class ex06 {

    public static void main(String[] args){

        double n1, n2, media;
        n1 = 0;
        n2 = 0;
        // utilizando Scanner como visto nas agendas anteriores.
        Scanner ler = new Scanner(System.in);
        for (int i=0; i<=30; i++){ //laço de repetição para os 30 alunos)
            System.out.println("Digite a primeira nota: " + n1);
            n1 = ler.nextDouble();
            //usuário irá digitar a nota e este comando irá ler e armazenar
            na variável.
            System.out.println("Digite a segunda nota: " + n2);
            n2 = ler.nextDouble();
            media = (n1 + n2)/2; // cálculo da média.
            if (media >= 7.0){
                System.out.println("Sua situação é: Aprovada " + media);
            }
            else {
                System.out.println("Sua situação é: Reprovada " + media);
            }
        }
    }
}

```

Observe que utilizamos o Laço de Repetição **para** (linha 11) e o laço de decisão (linha 18) para resolver o problema informado anteriormente. Além disso, utilizamos o Scanner (linhas 1, 10, 13 e 16) para que o usuário pudesse digitar as notas e o programa realizar todo cálculo.

Para entender melhor, assista à videoaula do Prof. Sandro Valérius:



Repare que estamos utilizando diversos conceitos aprendidos nas agendas anteriores, ou seja, tudo que você está estudando é acumulativo e poderá ser usado tudo de uma vez para que realize um melhor programa para solução do problema.

Teste de Mesa

Para que possamos nos certificar de que o algoritmo realizado está correto, antes de passar para a linguagem de programação (no nosso caso, Java) podemos testá-lo simulando valores e verificar se o resultado é o esperado. Esta simulação não é realizada no computador utilizando nenhum software. É realizada no papel.

Calma! O Teste de Mesa é muito simples de realizar. Basta montar uma pequena tabela e começar a simular os valores utilizando o seu pseudocódigo.

Para exemplificar, vamos voltar no algoritmo anterior (Mostrar os números de 01 até 09 com passo 01) e aplicar o Teste de Mesa.

contador	resultado
0	0
0+1	1
1+1	2
2+1	3
3+1	4
4+1	5
5+1	6
6+1	7
7+1	8
8+1	9

Veja que você iniciou com zero e como o exercício pede o incremento, ou seja, precisa somar 1 ao resultado anterior.

E como foi dito anteriormente, como o valor final é $i < 10$, então iremos até o valor 9.

Imagem 07 – Tabela indicando o avanço do Contador de 0 a 9.

Para exemplificar, vamos voltar no algoritmo anterior (Mostrar os números de 01 até 09 com passo 01) e aplicar o Teste de Mesa.

VOCÊ NO COMANDO

Elabore o Algoritmo, o Fluxograma e um Programa em Java que some todos os números no intervalo de 0 até 100.

Agora confira se você conseguiu resolver o desafio:

1.

Pseudocódigo	Fluxograma
<p>Programa ex1</p> <p>Declare num, soma como inteiro</p> <p>Início soma = 0 Escreva ("Entre com um número") Para num de 1 até 100 passo 1 faça soma = soma + num</p> <p> fim-para escreva(soma)</p> <p>fim.</p>	<pre> graph TD Inicio([Início]) --> Decl[Num, soma como inteiro] Decl --> Cond1{Num <= 1 Soma <= 0} Cond1 -- SIM --> SomaAdd[Soma = soma + num] SomaAdd --> NumInc[Num = num + 1] NumInc --> Cond2{Num <= 100} Cond2 -- SIM --> SomaAdd Cond2 -- Não --> Soma([soma]) Soma --> Fim([fim]) </pre>

Imagem 08 – Resolução do problema em Pseudocódigo e Fluxograma.

Veja no Java:

```

public class ex01 {

    public static void main(String[] args){

        int soma = 0; //declarando a variável como inteira e
                     // iniciando com valor zero.

        System.out.print("A soma dos 100 primeiros números é: ");
        //A frase que está aparecendo na saída de dados.

        for (int i = 1; i<=100; i++){ // Laço de repetição "for"
            soma += i; // soma = soma + i;
        }
        System.out.println(soma); //saída de resultados.

    }

}

```

Resultado:

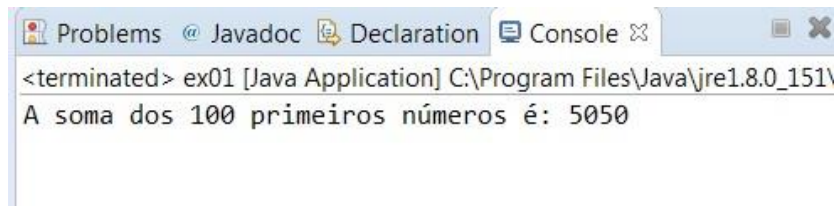


Imagem 09 – Prompt de Comando do Java exibindo a frase: A soma dos 100 primeiros números é: 5050.