**Aula 04 - Estruturas de Repetição**

**1. Lógica**

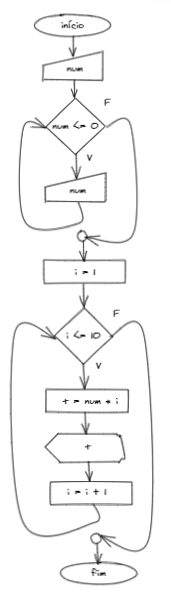
Em determinados programas, precisamos de entrada de dados, processamento e saída de dados. Entretanto, em alguns casos, precisamos também ter decisões e até mesmo **repetições** de determinados trechos de código. Neste caso, podemos utilizar as estruturas de repetição While, Do..While ou For no Java.

**2. While**

**Exemplo:**

Fazer um programa para receber um número e validar se esse número é positivo. Após isso, exibir a tabuada de 1 a 10 desse número:

Fluxograma:



Programação Java:

package estrutura\_repeticao;

import java.util.Scanner;

public class Exemplo01 {

public static *void* main(*String*[] *args*) {

*Scanner* ler = **new** Scanner(System.in);

*int* num, t, i;

System.out.printf("Digite um número positivo: ");

num = ler.nextInt();

while(num <= 0) {

System.out.printf("Erro, número negativo! Digite um número positivo: ");

num = ler.nextInt();

}

i = 1;

while (i <= 10) {

t = num \* i;

System.out.printf("\n%d X %d = %d", num, i, t);

i++;

}

}

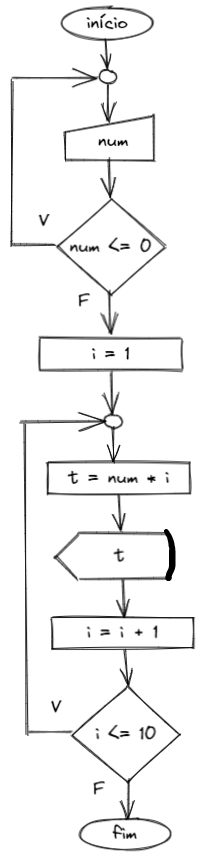
}

**3. Do..While**

**Exemplo:**

Fazer um programa para receber um número e validar se esse número é positivo. Após isso, exibir a tabuada de 1 a 10 desse número:

Fluxograma:



Programação Java:

package estrutura\_repeticao;

import java.util.Scanner;

public class Exemplo03 {

public static *void* main(*String*[] *args*) {

*Scanner* ler = **new** Scanner(System.in);

*int* num, t, i;

do {

System.out.printf("Digite um número positivo: ");

num = ler.nextInt();

}

while(num <= 0);

i = 1;

do {

t = num \* i;

System.out.printf("\n%d X %d = %d", num, i, t);

i++;

}

while (i <= 10);

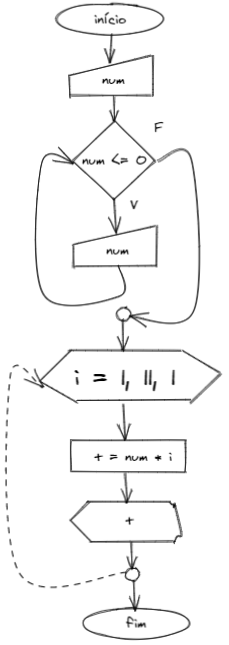
}

}

**4. For**

Exemplo:

Fazer um programa para receber um número e validar se esse número é positivo. Após isso, exibir a tabuada de 1 a 10 desse número:



Programação Java:

package estrutura\_repeticao;

import java.util.Scanner;

public class Exemplo02 {

public static *void* main(*String*[] *args*) {

*Scanner* ler = **new** Scanner(System.in);

*int* num, t, i;

System.out.printf("Digite um número positivo: ");

num = ler.nextInt();

while(num <= 0) {

System.out.printf("Erro, número negativo! Digite um número positivo: ");

num = ler.nextInt();

}

for (i=1; i<=10; i++) {

t = num \* i;

System.out.printf("\n%d X %d = %d", num, i, t);

}

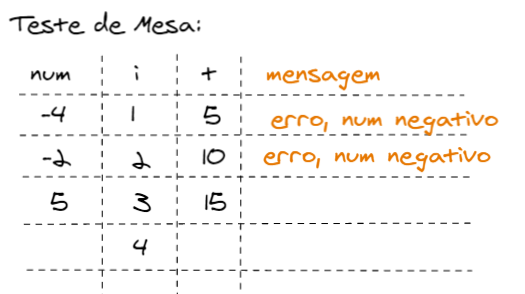
}

}

**5. Teste de Mesa**

Na Lógica de Programação, podemos utilizar um recurso extremamente útil chamado Teste de Mesa. Ele consiste em anotar todos os valores das variáveis conforme o programa vai avançando passo a passo.

Utilizando o exemplo acima, podemos ter o seguinte teste de mesa:



Neste caso, podemos ver que a cada looping do programa os valores estão sendo alterados corretamente, tendo o programa exibindo o resultado da tabuada do número 5 corretamente.

**6. Manipulando Loopings (While, Do While e For)**

**6.1. Break**

Caso você queira interromper um looping, você pode utilizar o comando break para isso, veja:

package estrutura\_repeticao;

import java.util.Scanner;

public class Exemplo01 {

public static *void* main(*String*[] *args*) {

*Scanner* ler = **new** Scanner(System.in);

*int* num, t, i;

System.out.printf("Digite um número positivo: ");

num = ler.nextInt();

while(num <= 0) {

System.out.printf("Erro, número negativo! Digite um número positivo: ");

num = ler.nextInt();

}

i = 1;

while (i <= 10) {

t = num \* i;

System.out.printf("\n%d X %d = %d", num, i, t);

if (i == 8)

break;

i++;

}

}

}

Neste caso, no mesmo exercício da tabuada, quando o i = 8, nós vamos “parar” o looping com o comando **break**.

**6.2. Continue**

Caso você queira pular uma iteração do looping, você pode utilizar o comando “continue” para isso, veja:

package estrutura\_repeticao;

import java.util.Scanner;

public class Exemplo02 {

public static *void* main(*String*[] *args*) {

*Scanner* ler = **new** Scanner(System.in);

*int* num, t, i;

System.out.printf("Digite um número positivo: ");

num = ler.nextInt();

while(num <= 0) {

System.out.printf("Erro, número negativo! Digite um número positivo: ");

num = ler.nextInt();

}

for (i=1; i<=10; i++) {

if (i == 8)

continue;

t = num \* i;

System.out.printf("\n%d X %d = %d", num, i, t);

}

}

}

Neste caso, no mesmo exercício da tabuada, quando o i = 8, nós vamos “pular” aquela iteração do looping com o comando **continue**.

**7. Exercícios**

31. Criar uma rotina de entrada que aceite somente um valor positivo.

32. Entrar com dois valores via teclado, onde o segundo deverá ser maior que o primeiro. Caso contrário solicitar novamente apenas o segundo valor.

33. Entrar via teclado com o sexo de determinado usuário, aceitar somente “F” ou “M” como respostas válidas.

34. Exibir a tabuada do número cinco no intervalo de um a dez.

35. Entrar via teclado com um valor qualquer. Travar a digitação, no sentido de aceitar somente valores positivos. Após a digitação, exibir a tabuada do valor solicitado, no intervalo de um a dez.

36. Entrar via teclado com um valor (X) qualquer. Travar a digitação, no sentido de aceitar somente valores positivos. Solicitar o intervalo que o programa que deverá calcular a tabuada do valor digitado, sendo que o segundo valor (B), deverá ser maior que o primeiro (A), caso contrário, digitar novamente somente o segundo. Após a validação dos dados, exibir a tabuada do valor digitado, no intervalo decrescente, ou seja, a tabuada de X no intervalo de B para A.

37. Exibir a tabuada dos valores de um a vinte, no intervalo de um a dez. Entre as tabuadas, solicitar que o usuário pressione uma tecla.

38. Exibir a soma dos números inteiros positivos do intervalo de um a cem.

39. Exibir os trinta primeiros valores da série de Fibonacci. A série: 1, 1, 2, 3, 5, 8, ...

40. Exibir os vinte primeiros valores da série de Bergamaschi. A série: 1, 1, 1, 3, 5, 9, 17, ...

41. Calcular e exibir a soma dos “N” primeiros valores da seqüência abaixo. O valor “N” será digitado, deverá ser positivo, mas menor que cem. Caso o valor não satisfaça a restrição, enviar mensagem de erro e solicitar o valor novamente.

A seqüência: 2, 5, 10, 17, 26, ....

42. Calcular e exibir a soma dos “N” primeiros valores da seqüência abaixo. O valor “N” será digitado, deverá ser positivo, mas menor que cinqüenta. Caso o valor não satisfaça a restrição, enviar mensagem de erro e solicitar o valor novamente.



43. Calcular e exibir a soma dos “N” primeiros valores da seqüência abaixo. O valor “N” será digitado, deverá ser positivo, mas menor que cinqüenta. Caso o valor não satisfaça a restrição, enviar mensagem de erro e solicitar o valor novamente.



44. Entrar via teclado com dez valores positivos. Consistir a digitação e enviar mensagem de erro, se necessário. Após a digitação, exibir:

a) O maior valor;

b) A soma dos valores;

c) A média aritmética dos valores;

45. Entrar via teclado com “N” valores quaisquer. O valor “N” (que representa a quantidade de números) será digitado, deverá ser positivo, mas menor que vinte. Caso a quantidade não satisfaça a restrição, enviar mensagem de erro e solicitar o valor novamente. Após a digitação dos “N” valores, exibir:

a) O maior valor;

b) O menor valor;

c) A soma dos valores;

d) A média aritmética dos valores;

e) A porcentagem de valores que são positivos;

f) A porcentagem de valores negativos;

46. Entrar via teclado com “N” valores quaisquer. O valor “N” (que representa a quantidade de números) será digitado, deverá ser positivo, mas menor que vinte. Caso a quantidade não satisfaça a restrição, enviar mensagem de erro e solicitar o valor novamente. Após a digitação dos “N” valores, exibir:

a) O maior valor;

b) O menor valor;

c) A soma dos valores;

d) A média aritmética dos valores;

e) A porcentagem de valores que são positivos;

f) A porcentagem de valores negativos;

Após exibir os dados, perguntar ao usuário se deseja ou não uma nova execução do programa. Consistir a resposta no sentido de aceitar somente “S” ou “N” e encerrar o programa em função dessa resposta.

47. Calcular o fatorial de um valor que será digitado. Este valor não poderá ser negativo. Enviar mensagem de erro e solicitar o valor novamente, se necessário. Perguntar se o usuário deseja ou não fazer um novo cálculo, consistir a resposta em “S” ou “N”.

N! = N x N-1 x N-2 x N-3 x ....... x (N - (N-1))

Ex: 5! = 5 x 4 x 3 x 2 x 1 = 120

48. Crie um programa em que o usuário entre com um número inteiro qualquer, e o programa imprima os 20 números subsequentes ao que foi digitado pelo usuário

49. Crie um programa que solicite que o usuário entre com dois números (inicial e final). Ao final o programa deverá apresentar o valor total da soma de todos os números do intervalo digitado pelo usuário

50. Elabore um programa que apresente os números pares maiores que 10 no intervalo fechado [A, B]. Sendo que A e B serão números inteiros escolhidos pelo usuário. Um número é par quando este satisfaz a seguinte condição: (NÚMERO mod 2 = 0)