Leitura complementar

Laço ou loop e repetição

- ✓ Estruturas de repetição com comando FOR...NEXT (PARA...PRÓXIMO);
- ✓ Estruturas de repetição com comando WHILE (ENQUANTO);
- ✓ Estruturas de repetição com comando DO...WHILE (FAÇA...ENQUANTO).

Introdução

Nesta aula, aprenderemos o conceito de laço e como escrever e utilizar os laços em programação.

Estruturas de laços e repetições são construídas para executar trechos de uma lógica várias vezes. Um laço em programação significa um retorno a linhas de programa já executadas para que sejam executadas novamente. Em inglês, **laço** significa **loop**, no sentido de retornar dando uma volta, fazendo um círculo.

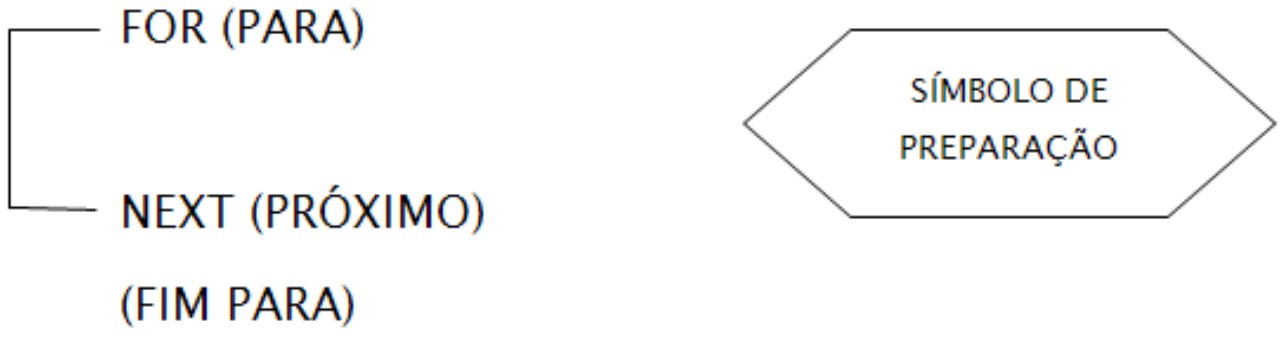
Os laços de repetição se encarregam de repetir determinados comandos enquanto uma determinada condição for satisfeita.

Os comandos descritos a seguir são utilizados para efetuar um laço dentro de uma lógica de programação.

Comando FOR...NEXT (PARA...PROXIMO)

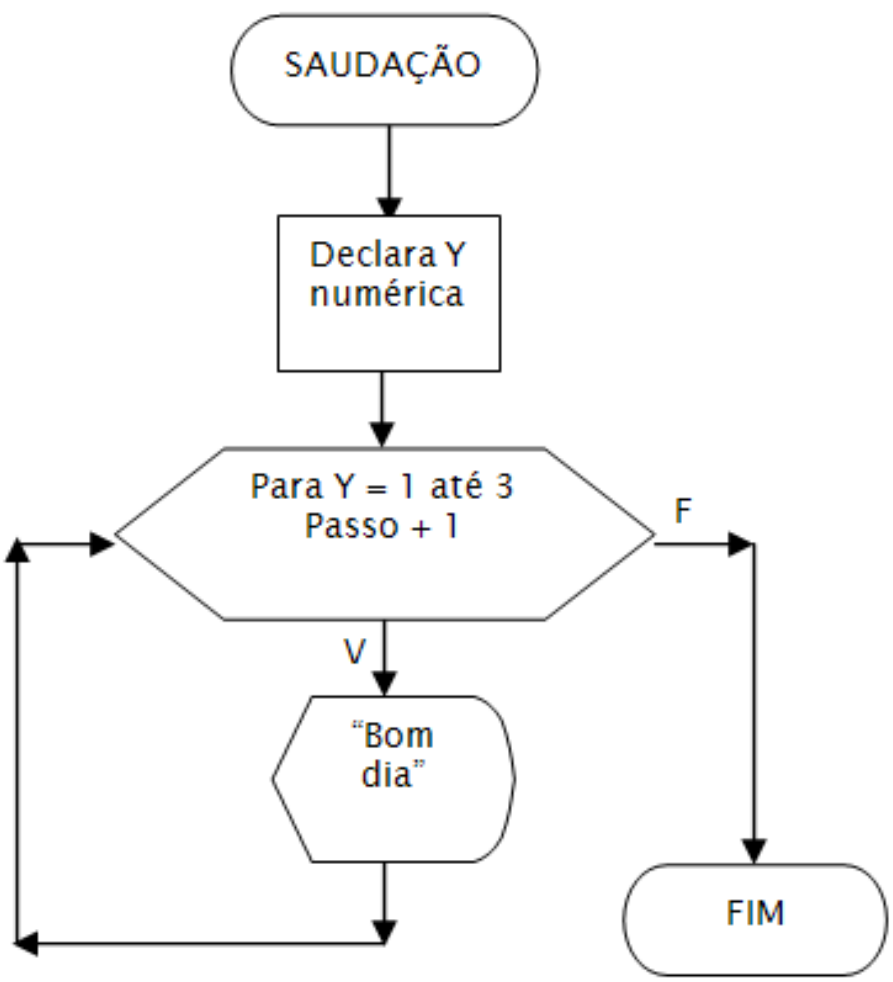
Um loop com o comando **FOR** é realizado quando uma condição com operação matemática é checada para executar um trecho de programa até o passo em que essa condição seja satisfeita (verdadeira).

Na sintaxe do comando **FOR**, declaramos sua inicialização, sua condição e seu incremento. A cada vez que o loop é executado, a variável do comando que é utilizada como contador aumenta ou diminui de acordo com o incremento, até que a condição do comando não seja mais satisfeita. Esse tipo de loop é uma repetição contável.

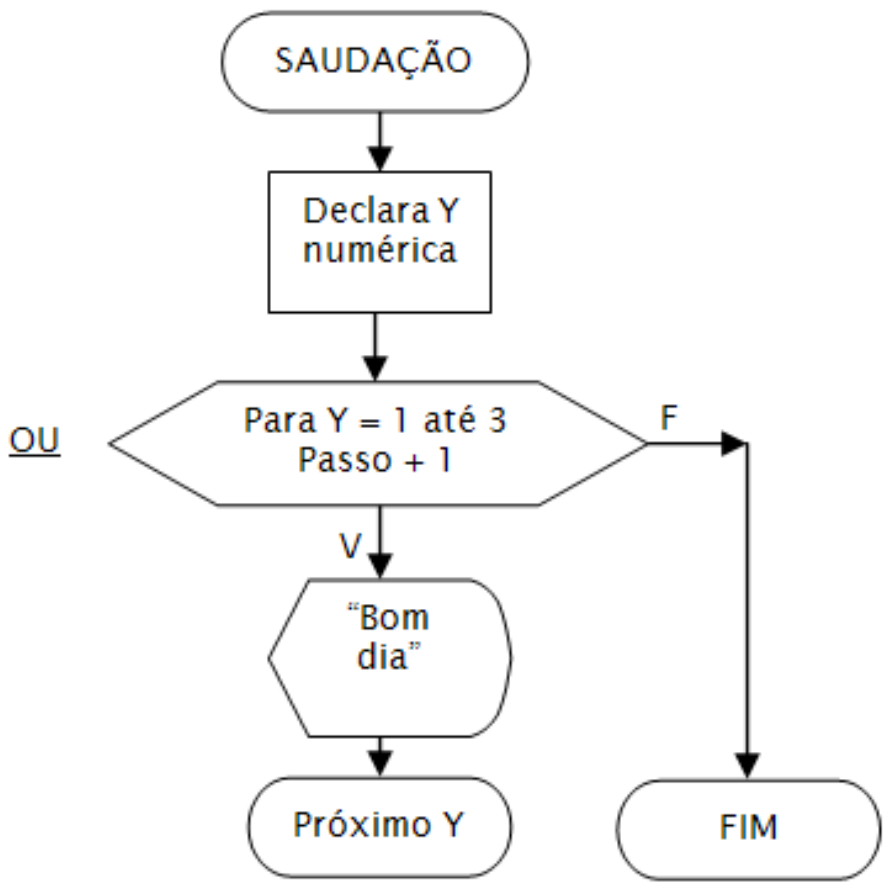


A seguir, temos um exemplo de algoritmo e fluxograma de um programa com o comando **PARA** que exibe três vezes a mensagem "**Bom dia**".

```
SAUDAÇÃO
  Declara Y numérica
  Para Y = 1 até 3 passo + 1
    Exibir "Bom dia"
  Próximo Y
FIM
```



OU

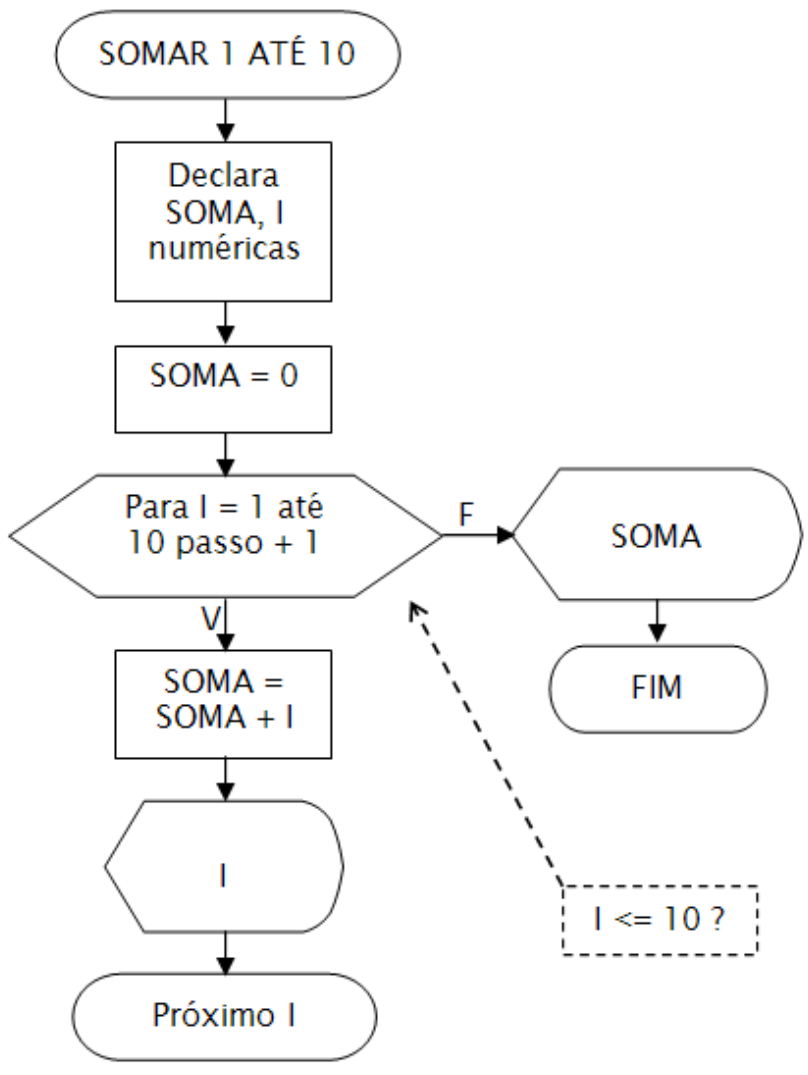


Note que no fluxo da direita está sendo utilizado um símbolo de terminação com o comando **PRÓXIMO Y**. Esse símbolo representa a seta do fluxo que retorna ao símbolo de preparação, conforme mostra o fluxo da esquerda. Depois deste símbolo não há mais símbolos. As duas estruturas são corretas.

A seguir, temos um exemplo de algoritmo com o comando **PARA** que soma os números de 1 (um) até 10 (dez), exibindo cada número somado e o valor total da soma:

```
SOMAR 1 ATÉ 10
   Declara SOMA, I numéricas
   SOMA = 0
   Para I = 1 até 10 passo + 1
    SOMA  = SOMA + I
    Exibir I
   Próximo I
   Exibir SOMA
FIM
```

A seguir, temos um fluxograma com o comando **PARA** que soma os números de 1 (um) até 10 (dez), exibindo cada número somado e o valor total da soma:



Teste de mesa

I =	1	2	3	4	5	6	7	8	9	10
Soma =	0	1	3	6	10	15	21	28	36	45	55

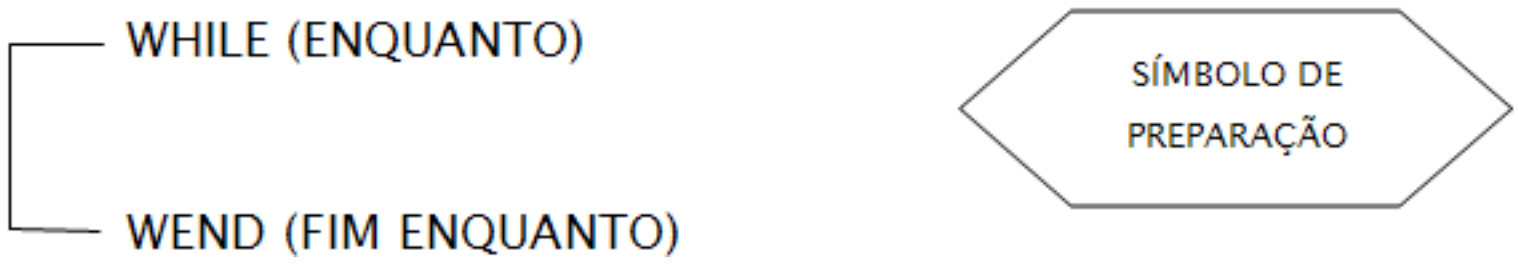
Serão exibidos todos os valores de **I** e somente o último valor de **SOMA**

Comando WHILE (ENQUANTO)

Um loop com o comando **WHILE** é realizado quando uma condição é checada para executar um trecho de programa até o passo em que essa condição seja satisfeita (verdadeira). Trata-se de um loop de repetição contável.

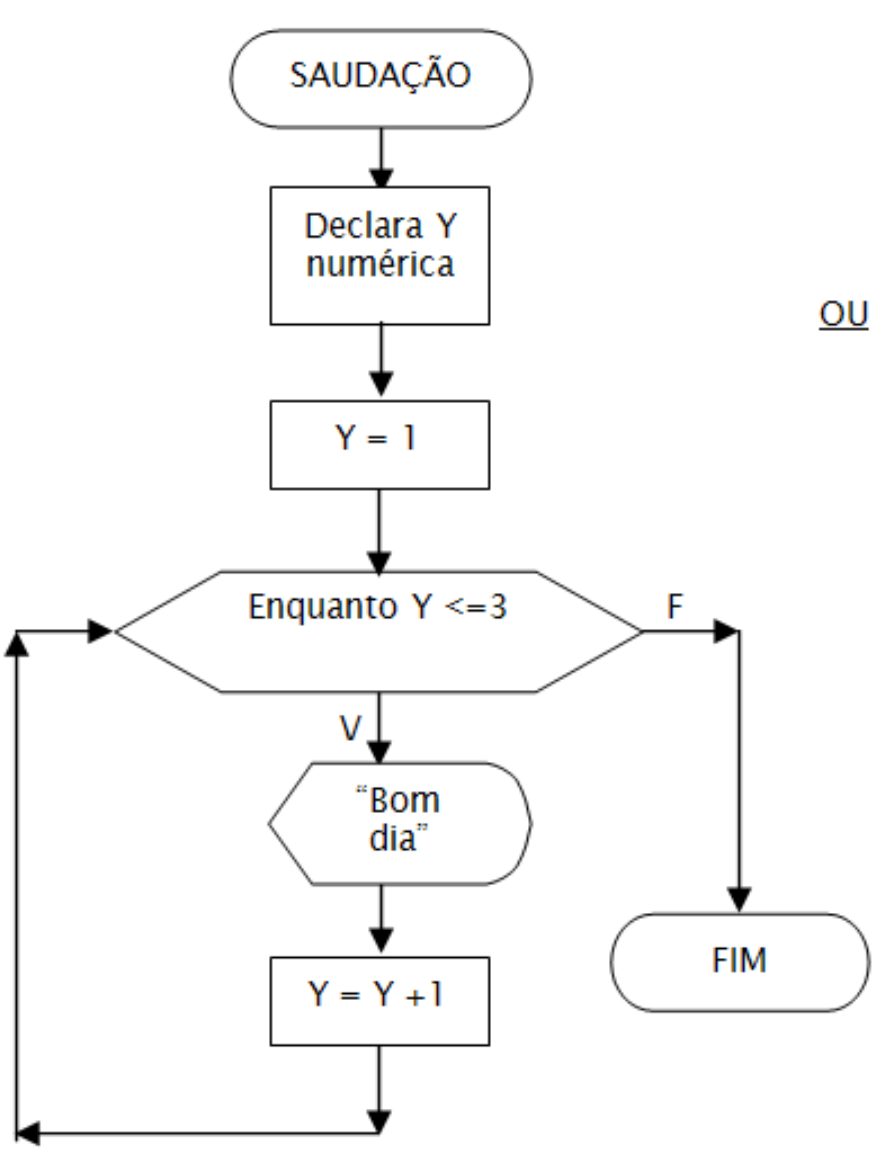
Na sintaxe do comando **WHILE**, declaramos apenas sua condição. Sua inicialização e seu incremento são ações executadas separadamente.

O loop com o comando **WHILE** é utilizado quando não conhecemos o número de iterações que temos que realizar. Esta é a diferença do loop **WHILE** com o comando **FOR**.

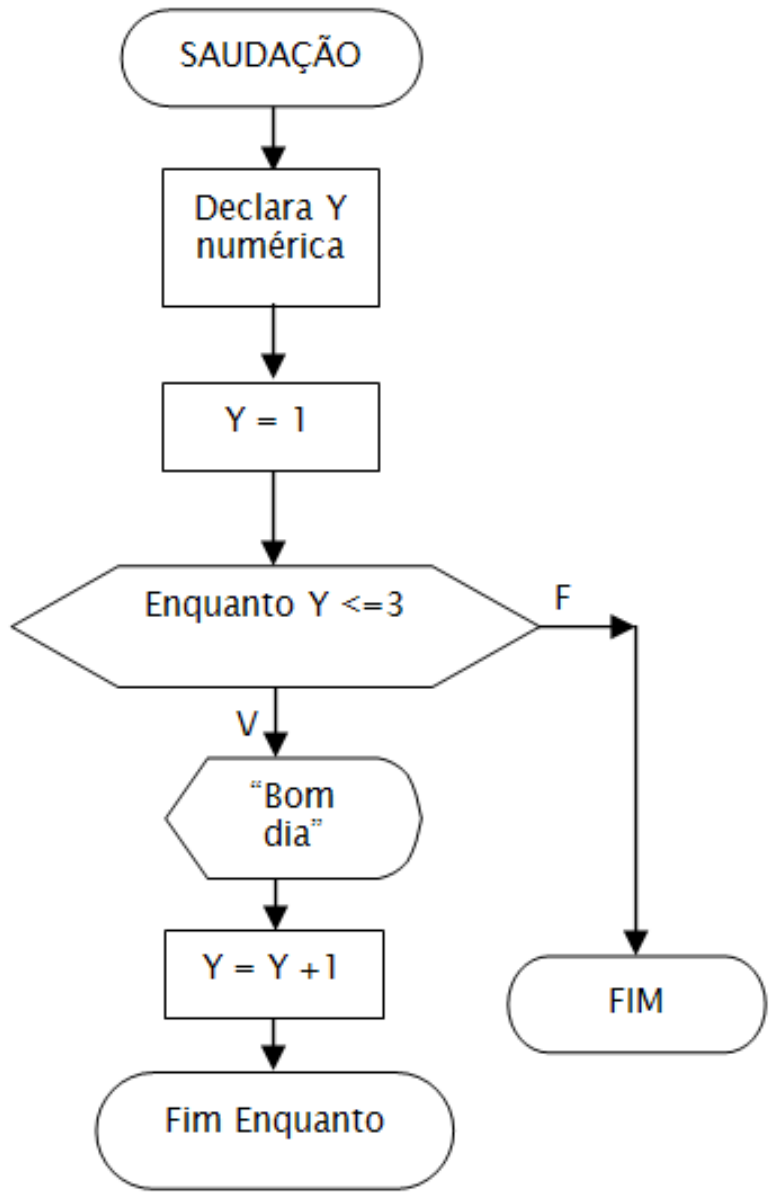


A seguir, temos um exemplo de algoritmo e fluxograma de um programa com o comando **ENQUANTO** que exibe três vezes a mensagem "**Bom dia**":

```
SAUDAÇÃO
   Declara Y numérica
   Y = 1
   Enquanto Y <= 3
    Exibir "Bom dia"
    Y = Y + 1
   Fim Enquanto
FIM
```



OU

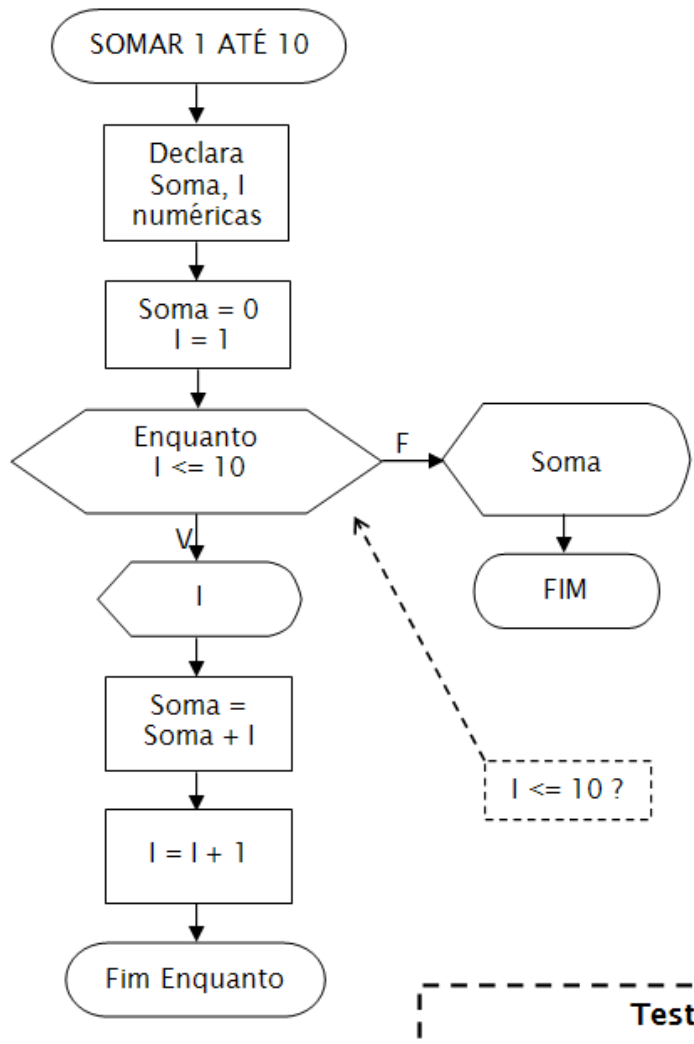


Note que no fluxo da direita está sendo utilizado um símbolo de terminação com o comando **FIM ENQUANTO** no lugar das setas. As duas estruturas são corretas.

A seguir, temos um exemplo de algoritmo com o comando **ENQUANTO** que soma os números de 1 (um) até 10 (dez), exibindo cada número somado e o valor total da soma:

```
SOMAR 1 ATÉ 10
   Declara SOMA, I numéricas
   SOMA = 0
   I = 1
   Enquanto I <= 10
     Exibir I
     SOMA  = SOMA + I
     I = I + 1
   Fim Enquanto
   Exibir SOMA
FIM
```

A seguir, temos um fluxograma com o comando **ENQUANTO** que soma os números de 1 (um) até 10 (dez), exibindo cada número somado e o valor total da soma:



Teste de mesa

I =	1	2	3	4	5	6	7	8	9	10
Soma = 0	1	3	6	10	15	21	28	36	45	55

Serão exibidos todos os valores de **I** e somente o último valor de **SOMA**

5.4 Comando DO...WHILE (FAÇA....ENQUANTO)

Um loop com o comando **DO...WHILE** é utilizado quando é necessário executar um trecho de código ao menos uma vez, antes da condição do loop ser checada para continuar ou não a iteração.

Isso quer dizer que, mesmo quando a condição estabelecida pelo loop é falsa, quando utilizamos **DO...WHILE**, o trecho de código será executado uma vez antes de ter sua iteração negada.

Assim como o comando **WHILE**, o **DO...WHILE** é utilizado quando não conhecemos o número de iterações que iremos realizar.

A seguir, temos um exemplo de algoritmo que utiliza o comando DO...WHILE para somar números até que o usuário digite o número zero (0):

```
SOMAR NÚMEROS
  Declara SOMA, X numéricas
  SOMA = 0
  Faça:
   Ler X
   Se X != 0:
         SOMA  = SOMA + X
  Enquanto X != 0
  Exibir SOMA
FIM
```

Veja que, como já havíamos atribuído o valor zero à variável **SOMA** antes de iniciar o loop, se utilizássemos a estrutura **WHILE**, o trecho de código não seria executado nenhuma vez. Com o uso do comando DO WHILE, é possível forçar a execução do trecho de código e obter resultados muito diferentes.