

IAL-003 – Algoritmos e Programação de Computadores

Prof. Me. Anderson Vanin

Estrutura de Repetição

Considere o algoritmo apresentado abaixo:

```
algoritmo "semnome"  
var  
n1,n2,n3,n4,media:real  
inicio  
escreval("Digite as 4 notas: ")  
leia(n1,n2,n3,n4)  
media <- (n1+n2+n3+n4)/4  
se (media>=6) entao  
    escreval("Aluno Aprovado. Media: ",media)  
senao  
    escreval("Aluno Reprovado. Media: ",media)  
fimse  
fimalgoritmo
```

Estrutura de Repetição

Observe que o algoritmo processa a média de um único aluno. E se existirem mais alunos?

- Podemos escrever o algoritmo para cada aluno. Assim sendo teremos de escrever **50 vezes o código se existirem 50 alunos.**
- Solução simples porém **inviável.**

Estrutura de Repetição

Outra solução: depois de executar o comando que escreve a situação de um aluno fazer com que o comando para a leitura de dados fosse executado novamente. **Este procedimento seria repetido mais 49 vezes.** A estes trechos do algoritmo que são repetidos damos o nome de **loop ou laço de repetição.**

Estrutura de Repetição

O **VisuAlg** implementa as três estruturas de repetição usuais nas linguagens de programação: o laço contado **para...ate...faca** (similar ao for...to...do do Pascal), e os laços condicionados **enquanto...faca** (similar ao while...do) e **repita...ate** (similar ao repeat...until).

Para ... faça

Esta estrutura repete uma sequência de comandos um determinado número de vezes.

```
para <variável> de <valor-inicial> ate <valor-limite> [passo <incremento>] faca  
    <sequência-de-comandos>  
fimpara
```

Enquanto ... faça

Esta estrutura repete uma sequência de comandos enquanto uma determinada condição (especificada através de uma expressão lógica) for satisfeita.

```
enquanto <expressão-lógica> faça  
    <sequência-de-comandos>  
fimenquanto
```

Repita ... até

Esta estrutura repete uma sequência de comandos até que uma determinada condição (especificada através de uma expressão lógica) seja satisfeita.

repita

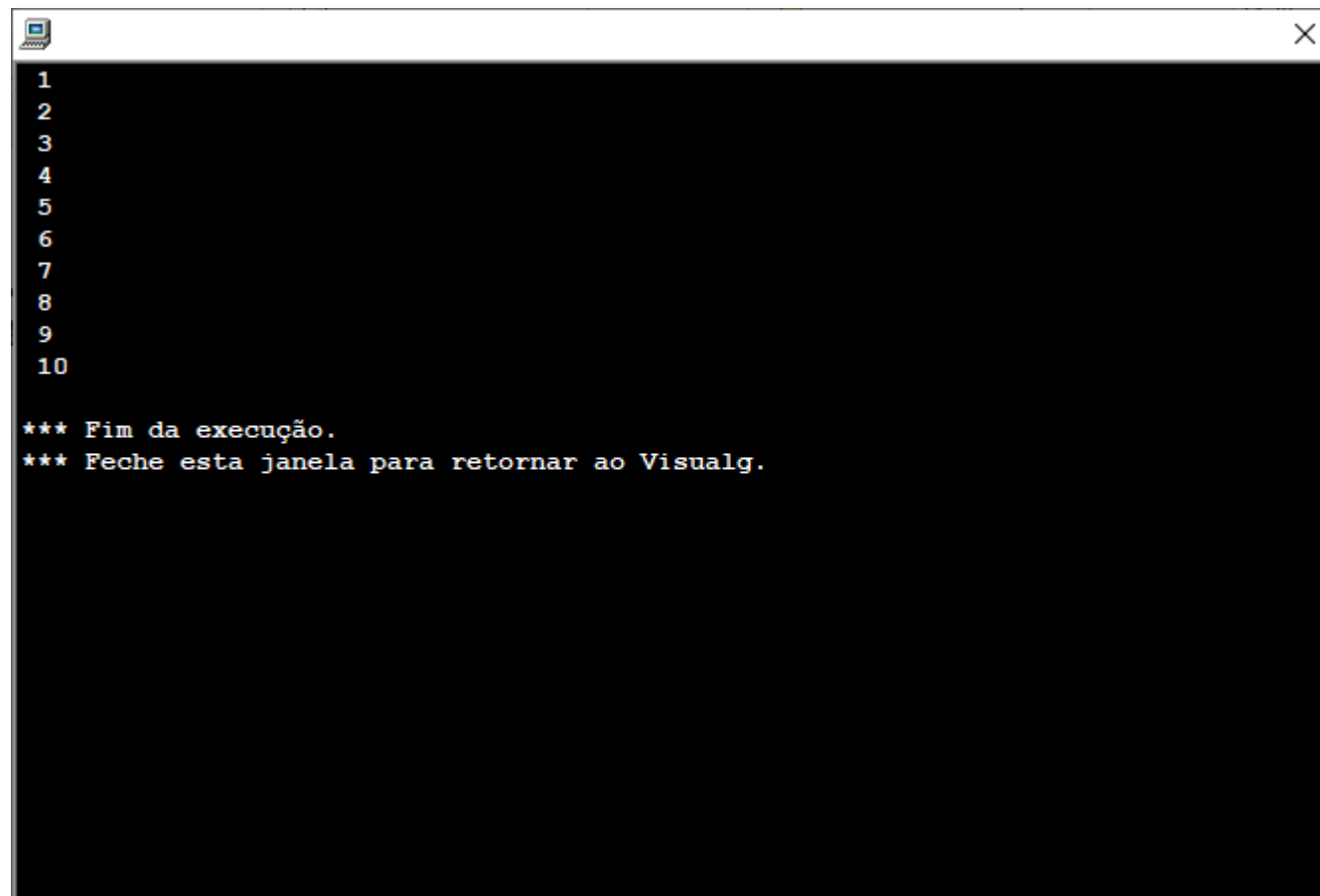
 <sequência-de-comandos>

ate <expressão-lógica>

Exemplo: Para ... faça

No exemplo a seguir, os números de 1 a 10 são exibidos em ordem crescente

```
algoritmo "semnome"  
var  
i:inteiro  
inicio  
para i de 1 ate 10 faca  
    escreval(i)  
fimpara  
  
fimalgoritmo
```

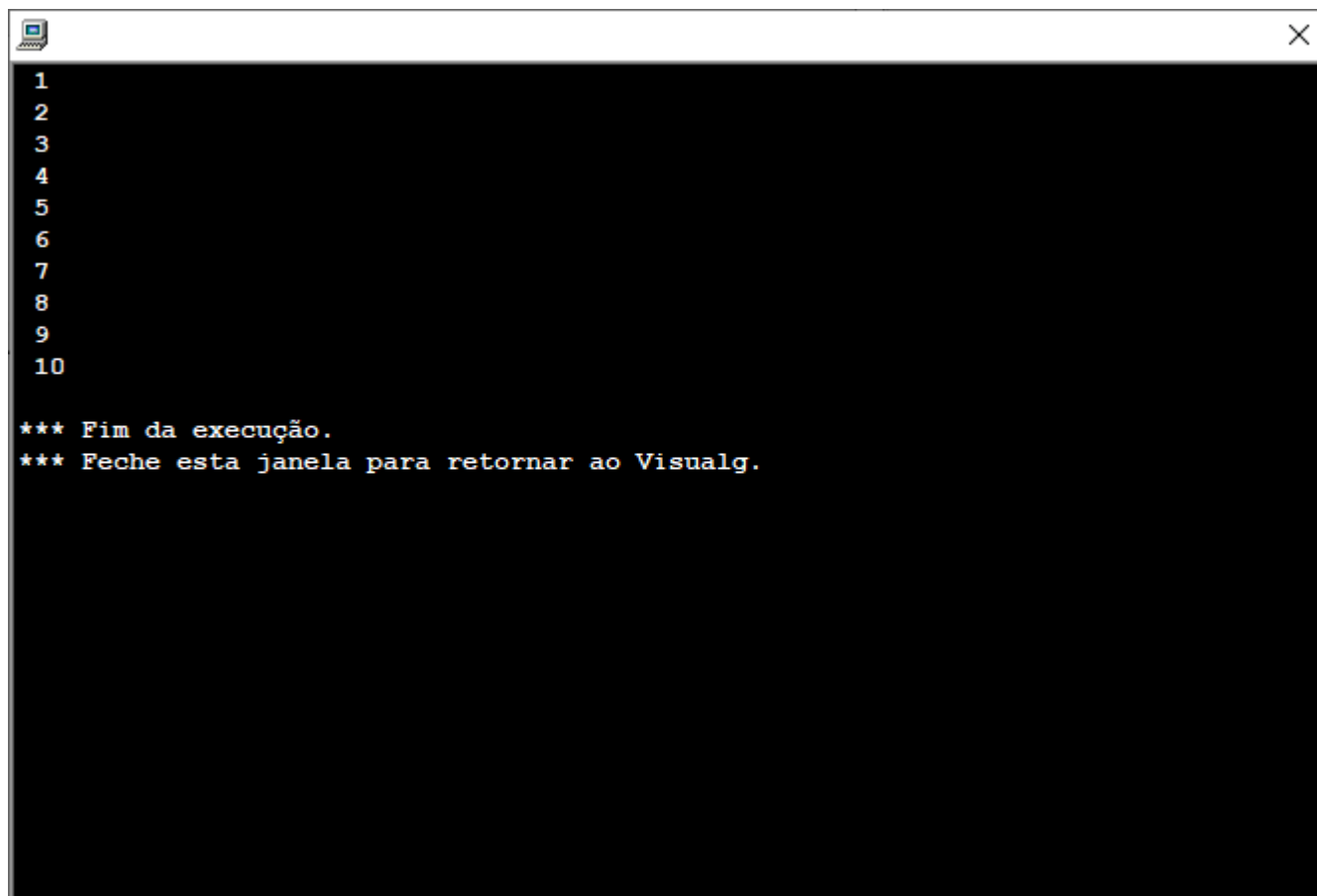


```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
  
*** Fim da execução.  
*** Feche esta janela para retornar ao Visualg.
```

Exemplo: Enquanto ... faça

No exemplo a seguir, os números de 1 a 10 são exibidos em ordem crescente

```
algoritmo "semnome"  
var  
i:inteiro  
inicio  
i ← 1  
enquanto i ≤ 10 faça  
    escreval(i)  
    i ← i + 1  
fimenquanto
```

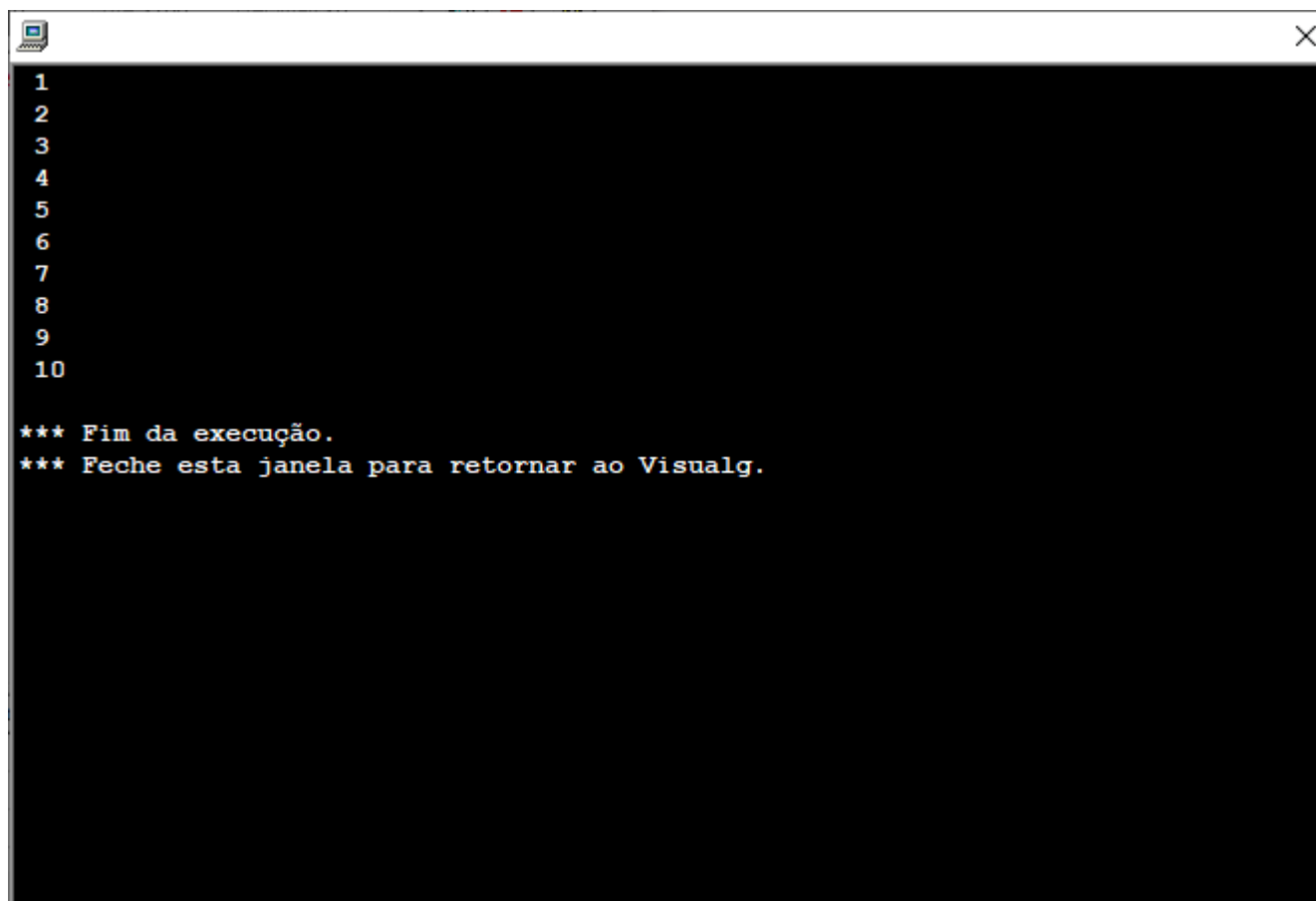


```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
  
*** Fim da execução.  
*** Feche esta janela para retornar ao Visualg.
```

Exemplo: Repita ... até

No exemplo a seguir, os números de 1 a 10 são exibidos em ordem crescente

```
algoritmo "semnome"  
var  
i:inteiro  
inicio  
i <- 1  
repita  
    escreval(i)  
    i <- i + 1  
ate i > 10  
fimalgoritmo
```



```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
  
*** Fim da execução.  
*** Feche esta janela para retornar ao Visualg.
```

Estrutura de Repetição

Todos os exemplos mostrados anteriormente executam a mesma tarefa: Realizar uma contagem de 1 a 10 em ordem crescente. Porém tenha em mente que cada laço de repetição tem uma finalidade específica. Para cada problema a ser resolvido deve-se avaliar o uso de um dos tipos apresentados anteriormente.

Exemplo

Escrever um algoritmo que lê 5 valores, um de cada vez, e conta quantos destes valores são negativos, escrevendo esta informação.

```
algoritmo "semnome"  
var  
i,num,qtde:inteiro  
inicio  
  
para i de 1 ate 5 faca  
    escreval("Digite um número inteiro: ")  
    leia(num)  
    se (num<0) entao  
        qtde <- qtde + 1  
    fimse  
fimpara  
escreval("O total de números negativos digitados é: ",qtde)  
  
fimalgoritmo
```

Exemplo

Escrever um algoritmo que lê 5 valores, um de cada vez, e conta quantos destes valores pares e quantos são ímpares, escrevendo esta informação.

```
algoritmo "semnome"  
var  
i,num,qtde_pares,qtde_impares:inteiro  
inicio  
  
para i de 1 ate 5 faca  
    escreval("Digite um número inteiro: ")  
    leia(num)  
    se (num mod 2 = 0) entao  
        qtde_pares <- qtde_pares + 1  
    senao  
        qtde_impares <- qtde_impares + 1  
    fimse  
fimpara  
escreval("O total de números pares digitados é: ",qtde_pares)  
escreval("O total de números ímpares digitados é: ",qtde_impares)  
  
fimalgoritmo
```

Exemplo

Escrever um algoritmo solicite números inteiros para o usuário, e realize a soma destes números. Programa deve ser encerrado caso o usuário digite o número 0.

algoritmo "semnome"

var

i,num,soma:inteiro

inicio

//enquanto num <> 0 faca

// escreval("Digite um número: ")

// leia(num)

// soma <- soma + num

//fimenquanto

//escreval("A soma dos números digitados é: ",soma)

repita

 escreval("Digite um número: ")

 leia(num)

 soma <- soma + num

ate num = 0

escreval("A soma dos números digitados é: ",soma)

fimalgoritmo

Este trecho comentado não irá funcionar, pois a variável num já é iniciada com 0