Estruturas de Repetição

Lista de Exercícios - 04

PITÁGORAS
FACULDADE

Algoritmos e Linguagens de Programação

Professor: Edwar Saliba Júnior

Estruturas de Repetição

O que são e para que servem?

São comandos que são utilizados na programação quando se deseja repetir, determinada parte do código, mais de uma vez. Ou seja, ao invés de escrevermos o mesmo código duas ou mais vezes, utilizamos uma estrutura de repetição. Exemplo:

Queremos imprimir os números inteiros de 1 a 10 no vídeo do computador.

a) *Sem estrutura de repetição*

```
início
    escreva "1"
    escreva "2"
    escreva "3"
    escreva "4"
    escreva "5"
    escreva "6"
    escreva "7"
    escreva "8"
    escreva "9"
    escreva "10"
fim
```

b) *Com estrutura de repetição PARA*

```
início
    declare Cont : inteiro

    para Cont ← 1 até 10 passo 1 faça
        escreva Cont
    fim para
fim
```

c) *Com estrutura de repetição ENQUANTO*

```
início
    declare Cont : inteiro

    Cont ← 1

    enquanto (Cont <= 10) faça
        escreva Cont
        Cont ← Cont + 1
    fim enquanto
fim
```

d) *Com estrutura de repetição REPITA*[1]

```
início
    declare Cont : inteiro

    Cont ← 1

    repita
        escreva Cont
        Cont ← Cont + 1
    enquanto (Cont <= 10)
fim
```

Observação: Para este caso especificamente, não há muito problema em escrevermos o comando "Escreva" seguido de um número, dez vezes em nosso algoritmo. Por outro lado, imagine como ficaria nosso algoritmo, sem estrutura de repetição, se quiséssemos imprimir os números inteiros de 1 a 10.000.

Nota: As estruturas de repetição também são conhecidas por: LUPES ou LAÇOS.

Para, Enquanto e Repita

Vimos acima que existem três estruturas de repetição diferentes, a estrutura PARA, a estrutura ENQUANTO e a estrutura REPITA. Até aí tudo bem, mas, quando utilizarmos cada uma delas?

Vejamos!

[1] A estrutura apresentada é exclusiva para a Linguagem de Programação C e suas derivadas. No caso da Linguagem de Programação Pascal e suas derivadas, a estrutura passa a ser: REPITA ... ATÉ (Condição). Há diferença na forma de teste da condição, nestas estruturas.

1. ***Estrutura PARA***

Deverá ser utilizada quando se sabe previamente o número de repetições que deverão ser executadas. Exemplo:

Imprima todos os números pares no intervalo de 1 a 30.

Para este problema, já foi determinado o número de vezes que o lupe será executado, ou seja, 30 vezes.

Resolução do problema:

```
início
    declare Cont : inteiro

    para Cont ← 1 até 30 passo 1 faça
        se (Cont mod 2 = 0) então   /* Testa se Cont possui valor par. */
            escreva Cont
    fim para
fim
```

2. ***Estrutura REPITA***

Deverá ser utilizada quando o lupe tem que ser executado no mínimo uma vez e, a execução do lupe mais de uma vez estará sujeita à condição imposta no final. Exemplo:

Imprima o somatório de todos os números inteiros no intervalo de 0 (zero) a N. Onde N deve ser um número inteiro maior ou igual a zero e será escolhido pelo usuário.

Para este problema, podemos considerar que, no mínimo uma vez o lupe deverá ser feito, pois, o menor número que o usuário poderá digitar é o 0 (zero).

Resolução do problema:

```
início
    declare N, Total, Cont : inteiro

    Total ← 0

    escreva "Digite um número inteiro maior ou igual a zero: "
    leia N

    Cont ← 0

    repita
        Total ← Total + Cont
        Cont ← Cont + 1
    enquanto (Cont <= N)

    escreva "O Somatório do intervalo de 0 a N é: ", Total
fim
```

3. ***Estrutura ENQUANTO***

Deverá ser utilizada quando, antes de se executar o lupe, for necessário testar uma condição.

Imprima o resultado da operação X^Y (leia-se: X elevado a Y). Onde X é a base e o primeiro número que o usuário digitará, e Y é o expoente ou potência e será o segundo número a ser digitado. Ambos inteiros.

Para este problema deveremos fazer o teste da condição antes de entrarmos no lupe.

Resolução do problema:

```
início
    declare X, Y, Total : inteiro

    escreva "Digite o valor da base X: "
    leia X
    escreva "Digite o valor do expoente Y: "
    leia Y

    Total ← 1

    enquanto (Y > 0) faça
        Total ← Total * X
        Y ← Y - 1
    fim enquanto

    escreva "Total de X elevado a Y é: ", Total
fim
```

Exercícios

1) Elabore um algoritmo que solicite que o usuário entre com 100 números inteiros quaisquer. Imprima a soma dos números digitados.

2) Elabore um algoritmo que leia um número qualquer digitado pelo usuário e calcule seu Fatorial. (Exemplo: 5! = 5 x 4 x 3 x 2 x 1)

3) Elabore um algoritmo em que o usuário entre com um número inteiro qualquer, e o software imprima os 20 números subsequentes ao que foi digitado pelo usuário.

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

5) Elabore um algoritmo que solicite que o usuário entre com 300 números quaisquer. Ao final apresente separadamente:

    a. A soma dos 100 primeiros números digitados;
    b. A soma do 101º número até o 200º;
    c. A soma do 201º número até o 300º.

6) Elabore um algoritmo 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)

7) Elabore um algoritmo que solicite que o usuário entre com 100 números quaisquer. Ao final apresente separadamente:

    a. A soma dos números pares que existirem entre o 1º número digitado até 50º;
    b. A soma dos números ímpares que existirem entre o 51º número digitado até o 100º.

8) Escreva um algoritmo que solicite que o usuário entre com valores inteiros quaisquer. Ao final imprima a quantidade de números digitados, o somatório dos valores digitados, e a média aritmética do somatório.

9) Elabore um algoritmo para fazer cálculo de potenciação. Ou seja, x^y. (Exemplo: 3^4 = 3 x 3 x 3 x 3). Seu algoritmo deverá solicitar que o usuário entre com o valor da base (x) e do expoente (y) e apresentar o resultado do cálculo sem utilizar os operadores ** ou ^. Para resolver o problema utilize estrutura de repetição.

10) Escreva um algoritmo que calcule a média da seguinte seqüência numérica a seguir: 1/2 + 1/3 + 1/4 + 1/5 + 1/6 + ... + 1/50. Feito isto, o algoritmo deverá apresentar uma lista contendo todos os números da seqüencia que estão acima da média calculada.

11) Elabore um algoritmo que apresente todos os números primos no intervalo de 1 a 50. Um número é considerado Primo quando ele puder ser dividido exclusivamente por 1 e por ele próprio.

Mais informações e exercícios poderão ser encontrados em:

ASCENCIO, Ana F. G.; CAMPOS, Edilene A. V. de. **Fundamentos da Programação de Computadores** : Algoritmos, Pascal e C/C++, São Paulo: Pearson, 2002.

- Páginas: 79 a 124.

ASCENCIO, Ana F. G.; CAMPOS, Edilene A. V. de. **Fundamentos da Programação de Computadores** : Algoritmos, Pascal, C/C++ e Java, 2ª. Ed., São Paulo: Pearson, 2007.

- Páginas: 93 a 144.

Exercícios de Depuração Usando Estruturas de Repetição

1) Apresente o que será impresso na tela do computador pelos algoritmos a seguir:

a)
```
início
    declare J, I, X : inteiro
    J ← 100
    X ← 3
    J ← J + 40
    I ← 5 ^ X * 4
    enquanto (X >= 5) então
        J ← J – 15
        X ← X + 1
        I ← I + X – J
    fim enquanto
    escreva J, I, X
fim
```

b)
```
início
    declare J, I, X : inteiro
    J ← 100
    X ← 3
    J ← J + 40
    I ← 5 ^ X * 4
    repita
        J ← J – 15
        X ← X + 1
        I ← I + X – J
    enquanto (X >= 5)
    escreva J, I, X
fim
```

c)
```
início
    declare J, I, X : inteiro
    J ← 100
    X ← 3
    J ← J + 40
    I ← 5 ^ X * 4
    enquanto (X <= 5) faça
        J ← J – 15
        X ← X + 1
        I ← I + X – J
    fim enquanto
    escreva J, I, X
fim
```

d)
```
início
    declare M, N, Y : inteiro
    M ← 10
    Y ← 1
    para N ← 1 até 3 passo 1 faça
        M ← M – 8
        Y ← Y * 3
    fim para
    escreva M, Y, N
fim
```

e)
```
início
    declare P, Q : inteiro
    declare VALOR : real
    P ← 5
    Q ← P - 8
    VALOR ← 18
    repita
        VALOR ← VALOR + (VALOR * P + Q)
        P ← P + 2
        Q ← Q + 1
    enquanto (Q < 0)
    escreva VALOR
fim
```

f)
```
início
    declare CONT : inteiro
    declare VALOR : real
    declare RESP : caracter
    CONT ← 0
    VALOR ← 0
    RESP ← 's'
    enquanto (RESP = 's') faça
        VALOR ← VALOR + 139
        CONT ← CONT + 1
        se (CONT > 3) então
            RESP ← 'n'
        fim se
    fim enquanto
    escreva VALOR
fim
```

g)
```
início
    declare N : inteiro
    declare SOMA : real
    SOMA ← 0
    para N ← 1 até 5 passo 1 faça
        SOMA ← SOMA + 1 / N
    fim para
    escreva SOMA
fim
```

h)
```
início
    declare N : inteiro
    N ← 0
    enquanto (N < 5) faça
        se (N = 0) então
            escreva "Esse número não existe: 1/0"
        senão
            escreva 1 / N
        fim se
        N ← N + 1
    fim enquanto
fim
```