Computação – Informática

Estrutura de Repetição
“**repita até**”
ou
“**faça enquanto**”

Slide 10

Prof. SIMÃO

Jean Marcelo SIMÃO

Estrutura de Repetição
“repita até”

---

repita

conjunto de comandos

até (condição ser Verdadeira)

Obs. : Formato padrão utilizado em algoritmos e suportado pelo Visualg.

Exemplo de Algoritmo – 1A

```
algoritmo " Imprimir os números de 1 a 1000 "
var
    num   : inteiro

inicio

    num ← 1

    repita

        escreval ( "Número: ", num )

        num ← num + 1

    ate  ( num > 1000 )

fimalgoritmo
```

Obs. O conjunto de comandos sempre executa pelo menos uma vez... diferentemente do *enquanto-faça*

Estrutura de Repetição “enquanto - faça”

faça

conjunto de comandos

enquanto (condição for verdadeira)

Obs. : Formato **NÃO** utilizado em algoritmos e **NÃO** suportado pelo Visualg. Entretanto, é o formato utilizado pela linguagem C...

Exemplo de Algoritmo – 1B

```
algoritmo " Imprimir os números de 1 a 1000"
var
   num    : inteiro

inicio

   num <- 1

   faca

        escreval ( "Número: ", num )

        num <- num + 1

    enquanto ( num <= 1000 )

fimalgoritmo
```

repita até X faça enquanto

```
algoritmo "Imprimir os números de 1 a 1000"
var
    num     : inteiro

inicio

    num <- 1

    repita

        escreval ( "Número: ", num )

        num <- num + 1

    ate  ( num > 1000 )

fimalgoritmo
```

```
algoritmo "Imprimir os números de 1 a 1000"
var
    num     : inteiro

inicio

    num <- 1

    faca

        escreval ( "Número: ", num )

        num <- num + 1

    enquanto ( num <= 1000 )

fimalgoritmo
```

As condições são diferentes porque a semântica (ou significado) do *repita-até* é diferente da semântica do *enquanto-faça*.

Exemplo de Algoritmo - 2

```
algoritmo "Somar todo os números de 1 a 1000 "
var
    soma, num      : inteiro

inicio
    soma <- 0
    num <- 1

    repita

            soma <-  soma + num
            num <- num + 1

    ate ( num > 1000 )

    escreval ( "O somatório dos números entre 1 e 1000 é:", soma )

fimalgoritmo
```

Explicando o Exemplo

```
algoritmo " Somar todo os números de 1 a 1000 "
var
    soma, num       : inteiro

inicio

    soma <- 0
    num <- 1

    repita

            soma <-  soma + num
            num <- num + 1

    ate ( num > 1000 )

    escreval ( "O somatório dos números entre 1 e 1000 é:", soma )

fimalgoritmo
```

Passo	*Num*	*Soma*
1	2	1
2	3	3
3	4	6
4	5	10
5	6	15
6	7	21
7	8	28
8	9	36
...	...	...

Exercício 1

- **Algoritmo para somar todos os números de uma seqüência que começa por um e finaliza em um número dado pelo usuário.**

- **Obs.: Usar a estrutura *repita-até*.**

Solução Exercício 1

```
algoritmo " Somar todo os números de 1 a n "
var
    soma, num, numdado  : inteiro

inicio
    soma <- 0
    num   <- 1

    escreval ( "Soma de números de zero até um número dado." )
    escreval ( "Informe um número" )
    leia ( numdado )

    repita

            soma <-  soma + num
            num <-    num + 1

    ate ( num > numdado )

    escreval ( "O somatório dos números entre 1 e ",  numdado, " é ", soma )

fimalgoritmo
```

Exercício 2

- **Algoritmo para somar os números impares entre 5 e 500 (inclusive).**

- **Obs. Utilizar a estrutura *repita ate*.**

Solução Ex.2 - V.1

```
algoritmo "Somar os impares entre 5 e 500"
var
    soma, num, resto    : inteiro
inicio
    soma ← 0
    num ← 5

    escreval ( "Soma de números impares entre 5 e 500." )

    repita

        resto <- num % 2

        se ( resto = 1 ) entao
            soma <-  soma + num
        fimse

        num <- num + 1
    ate ( num > 500 )

    escreval ( "O soma dos números impares entre 5 e 500 é:", soma )

fimalgoritmo
```

Solução Ex. 2 - V.2

```
algoritmo "Somar os impares entre 5 e 500 V2".
var
    soma, num   : inteiro

inicio

    soma <- 0
    num <- 5

    escreval ( "Soma de números impares entre 5 e 500." )

    repita

        soma ← soma + num

        num ← num + 2

    ate ( num > 500 )

    escreval ( "O soma dos números impares entre 5 e 500 é:", soma )

fimalgoritmo
```

Passo	soma	num
0	0	5
1	5	7
2	12	9
3	21	11
4	32	13
5	45	15
6	60	17

Exercício 3

- **Elaborar um algoritmo para o cálculo da soma, subtração, multiplicação ou divisão de dois números reais fornecidos pelo usuário, segundo sua opção.**

- **O usuário poderá realizar quantas operações desejar enquanto não optar por sair do programa.**

Obs. Utilizar a estrutura *repita-até*.

Solução exercício.

```
algoritmo      " Operações elementares sobre dois
               números cf. opção do usuário "
// Parte Principal
var
   result, prim_num, seg_num   : real
   opcao                       : inteiro

inicio

   repita

      escreval ( " Operações sobre 2 números reais." )
      escreval ( " Digite 1 para soma," )
      escreval ( "        2 para subtração, " )
      escreval ( "        3 para multiplicação ou" )
      escreval ( "        4 para divisão." )
      escreval ( "        5 para sair do programa." )
      leia ( opcao )

      se ( ( opcao > 0 )  e ( opcao < 5 ) ) entao

         escreval ( "Digite o primeiro número: " )
         leia ( prim_num )

         escreval ( "Digite o segundo número: " )
         leia ( seg_num )

         escolha ( opcao )

            caso 1
               result <- prim_num + seg_num
               escreval ( "O resultado da soma é:  ", result )

            caso 2
               result <-  prim_num - seg_num
               escreval ( "O resultado da subtração é:  ", result )

            caso 3
               result ←  prim_num * seg_num
               escreval ( "O resultado da multiplic. é: ", result )

            caso 4

               se ( seg_num <> 0 ) entao

                  result ←  prim_num / seg_num
                  escreval ( "O resultado da divisão é: ", result )

               senao

                  escreval ( "Divisão por zero impossível" )

               fimse

         fimescolha

      senao

         se ( opcao <> 5 ) entao
               escreval ( "Opção inválida" )
         fimse

      fimse

   ate  ( opcao = 5 )

fimalgoritmo
```

Exercício 4

4.1 - Elaborar um algoritmo para receber as notas de 150 alunos e calcular/apresentar a média das notas.

4.2 - Elaborar um algoritmo para receber as 4 notas de cada um dos 150 alunos, calculando/apresentando a média de cada um, bem como a média geral da turma.

Obs.: Em ambos, utilizar *repita-até*.

Solução 4.1

```
algoritmo “Média de notas de 150 alunos”
var
    soma, media, nota : real
    cont : nteiro
inicio
    soma <- 0
    cont   <- 1
    escreval ( “Média de notas de 150 alunos.” )

    repita
        escreval ( “Digite ”, cont, “ª. nota:” )

        leia ( nota )
        se ( ( nota <= 10 ) e ( nota >= 0 ) ) entao
            soma <-  soma + nota
            cont <- cont + 1
        senao
            escreval ( “Nota inválida.” )
        fimse
    ate ( cont >150 )

    media <- soma / 150
    escreval ( “A média das notas é: ”, media )
fimalgoritmo
```

```
algoritmo "Média de notas de 150 alunos"
var
    soma, media, nota : real
    cont : inteiro

inicio
    soma <- 0
    cont   <- 1
    escreval ( "Média de notas de 150 alunos." )

    repita
        escreval ( "Digite ", cont, "ª. nota:" )

        repita

            leia ( nota )
            se ( ( nota > 10 ) ou ( nota < 0 ) ) entao
                escreval ( "Nota inválida." )
            fimse
        ate ( ( nota <= 10 ) e ( nota >= 0 ) )

        soma <-  soma + nota
        cont <- cont + 1

    ate ( cont >150 )

    media <- soma / 150
    escreval ( "A média das notas é: ", media )

fimalgoritmo
```

Exercícios 5.

- **Algoritmo para permitir ao usuário escolher entre o cálculo do cubo, do quadrado ou da raiz quadrada de um número dado por ele. O usuário também pode escolher como opção 'sair do programa'.**
  - **Obs. 1: Uma primeira versão usando apenas estrutura de decisão *se fim-se* para tratar a opção do usuário.**
  - **Obs. 2: Uma segunda versão usando apenas estrutura de decisão *se senão* para tratar a opção do usuário.**
  - **Obs. 3: Uma terceira versão usando apenas estrutura de decisão *escolha caso* para tratar a opção do usuário.**
  - **Obs. 4: Para cada uma das versões anteriores usar a estrutura *repita-até*, comparando com as soluções realizadas anteriomente com a estrutura *equanto-faça*.**

Exercícios 6

- **6.1 Algoritmo para permitir o cálculo da área ou do perímetro de uma circuferência.**
  - **Obs.1: Raio fornecido pelo usuário.**
  - **Obs.2: Depois de cada cálculo o algoritmo deve permitir ao usuário escolher a mesma ou outra opção.**
  - **Obs.3: Utilizar a estrutura escolha-caso.**
  - **Obs.4. Utilizar uma variável tipo *caracter* para tratar a opção do usuário.**
  - **Obs.5: O algoritmo só terminará quando o usuário escolher uma opção de término.**
  - **Obs.6: Utilizar a estrura *repita-até*.**

- **6.2 Algoritmo para cálculo da área de um quadrado, de um triângulo retângulo ou de uma circuferência dependendo da escolha do usuário.**
  - **Obs.1: Parâmetros (e.g. lado ou raio) fornecidos pelo usuário.**
  - **Obs.2: Utilizar a estrutura escolha-caso.**
  - **Obs.3. Utilizar uma variável tipo caracter para tratar a opção do usuário.**
  - **Obs.4: Depois de cada cálculo o algoritmo deve permitir ao usuário escolher a mesma ou outra opção.**
  - **Obs.5: O algoritmo só terminará quando o usuário escolher uma opção de término.**
  - **Obs.6: Utilizar a estrutura *repita-até*.**

Exercícios - 7

- **7.1 - Algoritmo para o cálculo do quadrado e da raiz da raiz quadrada de um número dado pelo usuário.**

- **7.2 Algoritmo para o cálculo do fatorial de um número dado pelo usuário.**
  - **Obs.: Usar a estrutura de repetição *repita - até*.**

Algoritmo Fatorial

```
algoritmo "Fatorial"
var
   numero, resultado : Inteiro
inicio
   escreval ( "Informe um número decimal inteiro: " )
   leia ( numero )
   se ( numero > 1 ) entao
        resultado <- numero
        repita
             resultado <- resultado * (numero – 1)
             numero <- numero - 1
        ate ( numero = 1 )
        escreval ( "O fatorial é:", resultado )
   senao
        se ( (numero = 0) ou ( numero = 1) ) entao
             escreval ( "O fatorial é: 1." )
        senão
             escreval ( "Número invalido." )
        fimse
   fimse
fimalgoritmo
```

Numero = 5

	resultado	numero
0	5	5
1	5*4=20	4
2	20*3=60	3
3	60*2=120	2
4	120*1=120	1

Obs. O algoritmo da aula passada (usando *equanto-faça*) está certo?
E este, está certo?

Algoritmo Fatorial

```
algoritmo "Fatorial"
var
   numero, resultado : inteiro
inicio

   escreval ( "Informe um número decimal inteiro: " )
   leia ( numero )
   se ( numero > 1 ) entao
        resultado <- numero
        repita
             numero <- numero - 1
             resultado <- resultado * (numero)
        ate ( numero = 1 )
        escreval ( "O fatorial é:", resultado )
   senao
        se ( (numero = 0) ou ( numero = 1) ) entao
             escreval ( "O fatorial é: 1." )
        senao
             escreval ( "Número invalido." )
        fimse
   fimse

fimalgoritmo
```

```
algoritmo "Fatorial"
var
   numero, resultado : inteiro
inicio
   escreval ( "Informe um número decimal inteiro: " )
   leia ( numero )
   se ( numero > 1 ) entao
        resultado <- numero
        repita
             numero <- numero - 1
             resultado <- resultado * (numero)
        ate ( numero = 1 )
        escreval ( "O fatorial é:", resultado )
   senao
        se ( ( numero = 0) ou ( numero = 1 ) ) entao

             escreval ( "O fatorial é: 1." )
        senão
             escreval ( "Número invalido." )
        fimse
   fimse
fimalgoritmo
```

```
algoritmo "Fatorial"
var
    numero, resultado  : inteiro
inicio

   escreval ( "Informe um número decimal inteiro: " )
   leia ( numero )

   se ( numero > 0 ) entao
        resultado <- numero
        enquanto ( numero <> 1 ) faca
             numero <- numero - 1
             resultado <- resultado * numero
        fimenquanto
        escreval ( "O fatorial é:", resultado )
   senao
        se ( numero = 0 ) entao
             escreval ( "O fatorial é: 1." )
        senao
             escreval ( "Número invalido." )
   fimse
fimalgoritmo
```

Algoritmo Fatorial V.2

```
algoritmo "Fatorial"
var
   numero, resultado : inteiro
inicio

   escreval ( "Informe um número decimal inteiro: " )
   leia ( numero )

   se ( numero > 0 ) entao
        resultado <- 1
        repita
             resultado <- resultado * numero
             numero <- numero - 1
        ate ( numero = 1 )
        escreval ( "O fatorial é:", resultado )
   senao
        se ( numero = 0 ) entao
             escreval ( "O fatorial é: 1." )
        senao
             escreval ( "Número invalido." )
        fimse
   fimse
fimalgoritmo
```

Numero = 5
Resultado = 5

	resultado	numero
0	1	5
1	1*5=5	4
2	5*4=20	3
3	20*3=60	2
4	60*2=120	1

Exercício 8

- Refazer todos os exercícios anteriores, desta aula, usando a estrutura faça-enquanto.