Fundamentos de Programação 1

Estrutura de Repetição

“**para - passo**””

Slides 11

Prof. SIMÃO

Jean Marcelo SIMÃO

Estrutura de Repetição
“para passo”

---

para *Variável* **de** *ValorIni* **ate** *ValorFin* **passo** *P* **faca**

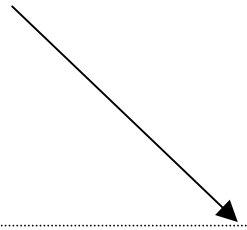
conjunto de comandos

fimpara

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

    para num de 1 ate 1000 passo 1 faca

        escreval ( "Número : ", num )

    fimpara

fimalgoritmo
```



Obs. A variável num é inicializada de forma "automática" (implicitamente) e o incremento dela também é "automático".

Para-passo X outros

```
algoritmo "Imprimir de 1 a 1000"
var
   num    : inteiro
inicio
   num <- 1
   repita
      escreval ( "Número : ", num )
     num <- num + 1
  ate  ( num > 1000 )
fimalgoritmo
```

```
algoritmo "Imprimir de 1 a 1000"
var
   num : inteiro
inicio
   num <- 1
   enquanto ( num <= 1000 ) faca
      escreval ( "Número : ", num )
      num <- num + 1
   fimenquanto
fimalgoritmo
```

```
algoritmo " Imprimir de 1 a 1000 "
var
    num   : inteiro
inicio
   para num de 1 ate 1000 passo 1 faca
      escreval ( "Número : ", num )
   fimpara
fimalgoritmo
```

Exemplo de Algoritmo - 2

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

inicio

    soma <- 0

    para num de 1 ate 1000 passo 1 faca
            soma <-  soma + num
    fimpara

    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

    para num de 1 ate 1000 passo 1 faca
            soma <-  soma + num
    fimpara

    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 *para-passo*.**

Solução Exercício 1

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

inicio

    soma <- 0

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

    para num de 1 ate numdado passo 1 faca
            soma <-  soma + num
    fimpara

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

fimalgoritmo
```

Exercício 2

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

- **Obs. Utilizar a estrutura *para-passo*.**

Solução Ex.2 - V.1

```
algoritmo " Somar os ímpares entre 5 e 500 "
var
    soma, num, resto     : inteiro

inicio
    soma <- 0
    escreval ( "Soma de números ímpares entre 5 e 500." )

    para num de 5 ate 499 passo 1 faca

        resto <- num % 2;
        // Em vez de %, outra opção seria  usar mod
        // resto <- num mod 2;

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

   fimpara

   escreval ( "A soma dos números ímpares entre 5 e 500 é:", soma )
fimalgoritmo
```

Solução Ex. 2 - V.2

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

inicio

    soma <- 0

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

    para num de 5 ate 499 passo 2 faca

            soma ← soma + num

    fimpara

    escreval ( "O soma dos números ímpares 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 *para-passo*.

Solução exercício.

```
algoritmo       " Operações elementares sobre dois
                números cf. opção do usuário "

var

   result, prim_num, seg_num : real
   opcao, cont     : inteiro

inicio

   para cont  de 1 ate 1  passo 0  faca

      // Pode ser que alguns interpretadores de algoritmos
      // não aceite passo 0. Neste caso, como resolver
      // usando ainda para-passo?

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

      leia ( opcao )

      se ( ( opcao > 0 )  e ( opcao < 6 ) ) entao

         se ( opcao <> 5 ) entao

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

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

         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

            caso 5
                  cont <- cont + 1
            fimescolha

         senao

            escreval ( " Opção inválida " )

         fimse
      fimpara
fimalgoritmo
```

Solução exercício.

```
algoritmo      " Operações elementares sobre dois
               números cf. opção do usuário "

var

   result, prim_num, seg_num : real
   opcao, cont     : inteiro

inicio

   para cont  de 1 ate 1 passo 0  faca

      // Pode ser que alguns interpretadores de algoritmos
      // não aceite passo 0. Neste caso, como resolver
      // usando ainda para-passo?

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

      leia ( opcao )

      se ( ( opcao > 0 )  e ( opcao < 6 ) ) entao

         se ( opcao <> 5 ) entao

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

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

Não suportado pelo Visual G

```
         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

            caso 5
                  cont <- cont + 1
            fimescolha

         senao

            escreval ( " Opção inválida " )

         fimse
      fimpara

fimalgoritmo
```

Solução exercício V2.

```
algoritmo      " Operações elementares sobre dois
               números cf. opção do usuário "

var

   result, prim_num, seg_num : real
   opcao, cont     : inteiro

inicio

   para cont  de 1 ate 1 passo 1 faca


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

      leia ( opcao )

      se ( ( opcao > 0 )  e ( opcao < 6 ) ) entao

         se ( opcao <> 5 ) entao

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

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

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

               caso 2
                     result <-  prim_num - seg_num
                     escreval ( " O resultado da subtração é:  ", result )
                    cont <- cont - 1
               caso 3
                     result <-  prim_num * seg_num
                     escreval ( " O resultado da multiplic. é: ", result  )
                    cont <- cont - 1
                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
                   cont <- cont - 1
             fimescolha

         senao

            escreval ( " Opção inválida " )

         fimse
      fimpara
fimalgoritmo
```

Solução exercício V2.

```
algoritmo      " Operações elementares sobre dois
               números cf. opção do usuário "

var

   result, prim_num, seg_num : real
   opcao, cont     : inteiro

inicio

   para cont  de 1 ate 1 passo 1 faca


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

      leia ( opcao )

      se ( ( opcao > 0 )  e ( opcao < 6 ) ) entao

         se ( opcao <> 5 ) entao

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

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

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

            caso 2
                  result <-  prim_num - seg_num
                  escreval ( " O resultado da subtração é:  ", result )
                  cont <- cont - 1
            caso 3
                  result <-  prim_num * seg_num
                  escreval ( " O resultado da multiplic. é: ", result  )
                  cont <-  cont - 1
            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
                  cont <- cont - 1
            fimescolha

      senao

         escreval ( " Opção inválida " )

      fimse
   fimpara
fimalgoritmo
```

Não suportado pelo Visual G, por erro deste!

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 *para-passo*.

Solução 4.1

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

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

    para cont de 1 ate 150 passo 1 faca
        escreval ( " Digite ", cont, "a. nota: " )

        leia ( nota )
        se ( ( nota <= 10 ) e ( nota >=0 ) ) entao
            soma <-  soma + nota
        senao
            escreval ( " Nota inválida. " )
            cont <- cont - 1
        fimse
    fimpara

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

fimalgoritmo
```

Solução 4.2

```
algoritmo “solucao 4.2"
var
   ma, mt, nota, somaal : real
   aluno, cont : inteiro
inicio
  escreval ( "Algoritmo para cálcular a media de 150 alunos e a média da turma.")
  ma <- 0
  mt <- 0
  aluno <- 1
  para aluno de 1 ate 150 passo 1 faca
     somaal <- 0
     para cont de 1 ate 4 passo 1 faca
       escreval ( " Digite a ", cont , ".a nota " )
       repita
         leia ( nota )
        // se ...
       ate ( ( nota >= 0 ) e ( nota <= 10 ) )
       somaal <- somaal + nota;
     fimpara
     ma <- somaal / 4;
     escreval ( " A média do aluno é: " , ma )
     mt <- mt + ma
  fimpara
  escreval ( aluno )
  escreval ( " A média da turma é: " , mt / ( aluno-1 ) )
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 *para-passo*, comparando com as soluções realizadas anteriomente com a estrutura *equanto-faça* e *repita-até*.**

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 *parra-passo*.**

- **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 *para-passo*.**

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 *para-passo*.**

Algoritmo Fatorial

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

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

   se ( numero > 1 ) entao
        resultado <- numero
        para numero de numero ate 2 passo -1 faca
             resultado <- resultado * ( numero – 1 )
        fimpara
        escreval ( "O fatorial é:", resultado )

   senao
        se ( (numero = 0) ou ( numero = 1) ) entao
             escreval ( "O fatorial é: 1." )
        senão
             escreval ( "Número inválido." )
        fimse
   fimse

fimalgoritmo
```

Numero = 5

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

Algoritmo Fatorial V.2

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

inicio

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

   se ( numero >= 1 ) entao
        resultado <- 1
        para numero de numero ate 1 passo -1 faca
             resultado ← resultado * numero
        fimpara
        escreval ( " O fatorial é: ", resultado )
   senao
        se ( numero = 0 ) entao
             escreval ( " O fatorial é: 1. " )
        senao
             escreval ( " Número inválido. " )
        fimse
   fimse

fimalgoritmo
```

Numero = 5

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

Estrutura de Repetição
“para passo”

formato próximo da linguagem C

para (Var ← ValorInic**;** Var ***OC*** Valor Inic**;** Var ← Var + *Incremento* **)**

conjunto de comandos;

fim-para

Obs1.: OC significa Operador de Comparação (=, ≠, <, >, ...).

Obs2.: 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 à 1000’.
Início

Inteiro num**;**

para (num ← 1**;** num <= 1000**;** num ← num + 1**)**

Imprime(“Número %i : ”, num);

fim-para

Fim.

Obs.: Vale a observação que **não** deveriamos utilizar este formato próximo a linguagem C em algoritmos. Deveriamos utilizar o formato padrão ensinado nesta aula...