Fundamentos de Programação1

RECURSÃO

“**Recursão X Iteração**””

Slides 13

Prof. SIMÃO

Jean Marcelo SIMÃO

Iteração

Até então, nós aprendemos a construir algoritmos ou funções iterativas cujas repetições são baseados em comandos/estruturas de laços de repetição. Estas estruturas são o *para-passo*, o *enquanto-faça* e o *repita-até* (ou *faça-enquanto*). Um exemplo simples de função iterativa é **fact()** que calcula o fatorial de um inteiro...

```
algoritmo "Cálculo de Fatorial"
var a : inteiro

funcao fact ( n : Inteiro): inteiro
var
  i, resp  : inteiro

inicio
    resp <- 1

    para i de 1 até n passo 1 faça

          resp <- resp * i

    fimpara

   retorne resp

fimfuncao
```

```
// parte-principal do algoritmo
resultado  :  inteiro;

Inicio


  resultado <- fact ( 6 );

  escreval ("O fatorial de 6 é  ", resultado )

fimalgoritmo
```

Obs.: Iteração = Repetição.

Recursão

Em algoritmos, bem como em linguagem C, funções podem chamar a si mesmas. A função é recursiva se um comando no corpo da função a chama. Recursão é o processo de definir algo em termos de si mesmo e é, algumas vezes, chamado de definição circular. Um exemplo simples de função recursiva é **factr()** que calcula o fatorial de um inteiro... (Schildt, 1997)

```
algoritmo "cálculo de fatorial"
var
  resultado : inteiro

funcao factr ( n : inteiro ) : inteiro
var
   resp        : inteiro

inicio

   se ( n = 1 ) entao
      retorne 1
   senao
      // chamada recursiva
      resp <- n * factr ( n-1 )
      retorne resp
   fimse

fimfuncao
```

```
// parte-principal do algoritmo
inicio

  resultado <- factr ( 2 )
  escreval ( "O fatorial de 2 é ", resultado )

fimalgoritmo
```

Entendendo a Recursão A

```
algoritmo "cálculo de fatorial"
var
  resultado : inteiro

funcao factr ( n : inteiro ) : inteiro
var
    resp        : inteiro

inicio

    se ( n = 1 ) entao
        retorne 1
    senao
        // chamada recursiva
        resp <- n * factr ( n-1 )
        retorne resp
    fimse

fimfuncao

//Parte-Principal do Algoritmo
inicio

  resultado <- factr ( 2 )
  escreval ( "O fatorial de 2 é ", resultado )

fimalgoritmo
```

fact (2)

> fact (1)
>
> retorna 1

2 = 2 x 1

retorna 2;

Entendendo a Recursão B

```
algoritmo "cálculo de fatorial"
var
  resultado : inteiro

funcao factr ( n : inteiro ) : inteiro
var
    resp        : inteiro

inicio

    se ( n = 1 ) entao
        retorne 1
    senao
        // Chamada recursiva
        resp <- n * factr ( n-1 )
        retorne resp
    fimse

fimfuncao

//Parte-Principal do Algoritmo
inicio

  resultado <- factr ( 3 )
  escreval ( "O fatorial de 3 é ", resultado )

fimalgoritmo
```

fact (3)

fact (2)

fact (1)

retorna 1

2 = 2 x 1

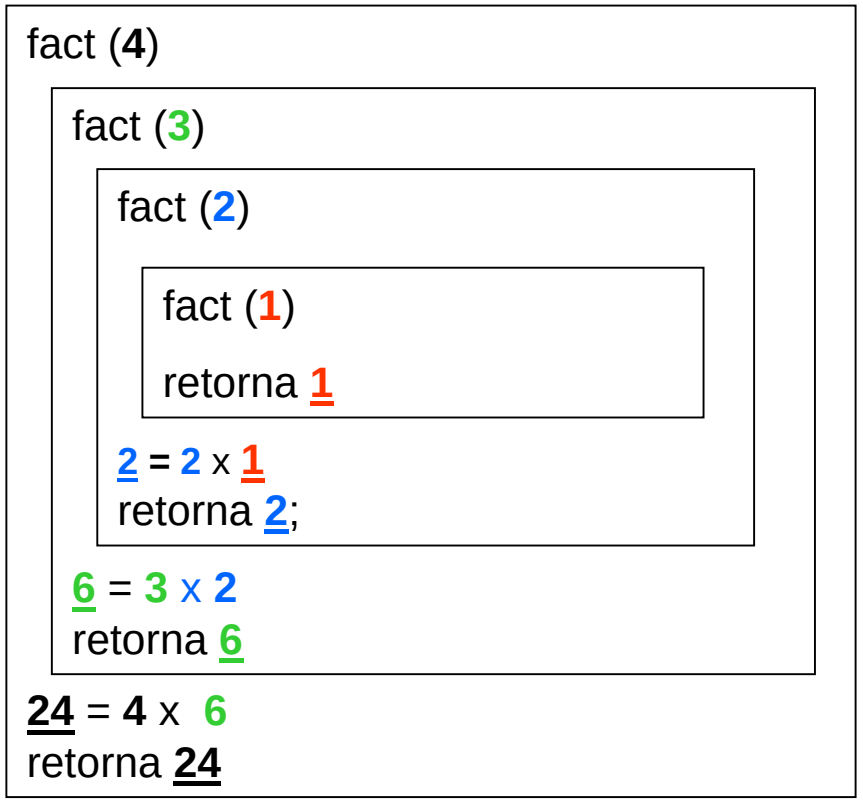
retorna 2;

6 = 3 x 2

retorna 6

Entendendo a Recursão C

```
algoritmo "cálculo de fatorial"
var
  resultado : inteiro

funcao factr ( n : inteiro ) : inteiro
var
    resp        : inteiro

inicio

    se ( n = 1 ) entao
        retorne 1
    senao
        // Chamada recursiva
        resp <- n * factr ( n-1 )
        retorne resp
    fimse

fimfuncao

//Parte-Principal do Algoritmo
inicio

  resultado <- factr ( 4 )
  escreval ( "O fatorial de 4 é ", resultado )

fimalgoritmo
```



Exercícios A

a) Validar as variáveis nos dois exemplos anteriores, i.e. nas funções iterativa e recursiva para o cálculo do fatorial (incluindo o questão de fatorial de zero...).

b) Elaborar e/ou pesquisar outros exemplos de funções recursivas.

c) Refletir e escrever sobre as diferenças de funções iterativas e de funções recursivas.

Exercícios B

- Implementar a função de cálculo de exponenciação de forma recursiva.
  - A função terá como parametros uma base e um expoente.
- Escreva a função para cálculo do n-ésimo termo da série de Fibonacci utilizando recursividade.
  - Pesquisar sobre a série de Fibonacci.