Fundamentos de Programação1

RECURSÃO

“Recursão X Iteração”

Slides 13 – Linguagem C

Prof. SIMÃO

Jean Marcelo SIMÃO

Iteração

Até então, aprendemos a construir programas com funções iterativas cujas repetições são baseados em comandos/estruturas de laços de repetição. Estas estruturas são o *for ( ) { }*, o *while () { }* e o *do{ }while ();*.

Um exemplo simples de função iterativa é **fatiterativo()** que calcula o fatorial de um inteiro...

```
fatiterativo ( int n )
{

   int resposta;
   resposta = 1;

   for ( int t = 1; t <= n; t++ )
   {
      resposta = resposta * t ;
   }

   return resposta;

}
```

Obs.: Iteração = Repetição.

```
int main()
{
   int num;
   int resp;
   printf ( " Programa de cálculo de fatorial. \n " );

   printf ( " Digite um número: \n " );
   scanf ( "%i", &num );

   if ( num > 0 )
   {
      resp = fatiterativo ( num );
      printf ( " O fatorial calculado com é: %i \n \n ", resp);
   }
   else
   {
      printf ( " Número inapropriado." );
   }

   system ( "Pause" );
   return 0;
}
```

Recursão

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 **fatrecursivo()** que calcula o fatorial de um inteiro... (Schildt, 1997)

```
fatrecursivo ( int n )
{
  int resposta;

  if ( 1 == n )
  {
    return 1;
  }

  // chamada recursiva

  resposta = n * fatrecursivo ( n-1 ) ;

  return resposta;
}
```

```
int main()
{
   int num,
   int resp;
   printf ( " Programa de cálculo de fatorial. \n " );

   printf ( "Digite um número: \n" );
   scanf ("%i", &num );

   if ( num > 0 )
   {
      resp = fatrecursivo ( num );
      printf ( "O fatorial calculado é: %i. \n \n", resp);
   }
   else
   {
      printf ( "Número inapropriado." );
   }

   system ("Pause");
   return 0;
}
```

Entendendo a Recursão A

```
fatrecursivo ( int n )
{
   int resposta;

   if ( 1 == n )
   {
      return 1;
   }

   // chamada recursiva

   resposta = n * fatrecursivo ( n-1 );

   return resposta;
}
```

fatrecursivo (2)

> fatrecursivo (1)
>
> retorna 1

2 = 2 x 1
retorna 2;

Entendendo a Recursão B

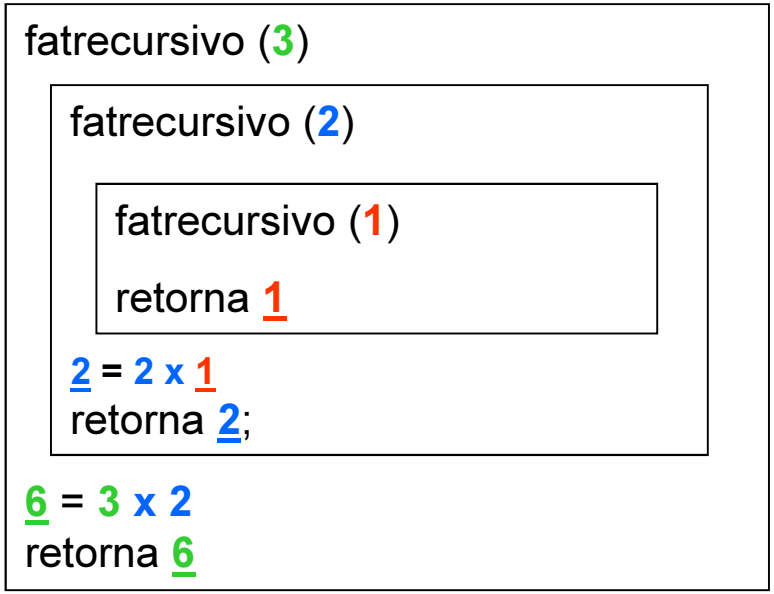
```
fatrecursivo ( int n )
{
   int resposta;

   if ( 1 == n )
   {
      return 1;
   }

   // chamada recursiva

   resposta = n * fatrecursivo ( n-1 );

   return resposta;
}
```



Entendendo a Recursão C

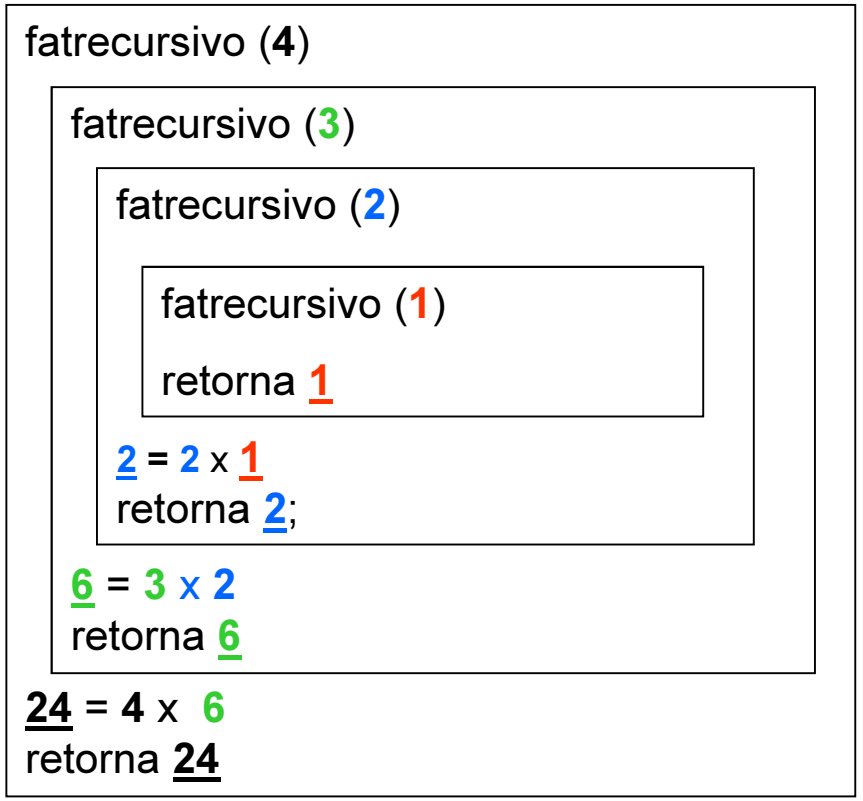
```
fatrecursivo ( int n )
{
   int resposta;

   if ( 1 == n )
   {
      return 1;
   }

   // chamada recursiva

   resposta = n * fatrecursivo ( n-1 );

   return resposta;
}
```



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.