

Atividade prática introdutória sobre recursividade

- 1) [Adaptação da questão 31 do Poscomp 2005] Considere o programa:

```
#include <stdio.h>
int m, n;

int FUN(int n)
{   int x;

    if (n < 1) return 1;
    else
    {   x = n * FUN(n - 1);
        m = m - 1;
        return m + x;
    }
}

int main(void)
{   scanf("%d %d", &m, &n);
    printf("%d %d %d\n", m, n, FUN(n));

    return 0;
}
```

Este programa, para os valores $m = 5$ e $n = 4$, tem como resultado:

- a) 5, 4, 120 b) 5, 4, 14400 c) 1, 4, 165 d) 5, 4, 5 e) 5, 4, 165

- 2) O somatório dos n primeiros números inteiros pode ser expresso de forma recursiva ou não recursiva, como mostram os dois algoritmos a seguir. Implemente e teste cada um deles. Considere que o somatório pode ser expresso como
- $$\text{sigma}(n) = 1 + 2 + 3 + \dots + (n-1) + n$$

Versão não-recursiva da função sigma

```
Função Sigma : int ( n int )_
Var soma, ind : int ;
início
    soma ← 0 ; ind ← 1 ;
    enquanto ind ≤ n faça
    início
        soma ← soma + ind ;
        ind ← ind + 1 ;
    fim
    retorne soma ;
fim
```

Versão recursiva da função sigma

```
Função Sigma : int ( n int )_
início
    se  $n \geq 0$  então
        retorne  $n + \text{sigma}(n - 1)$  ;
    senão
        retorne 0 ;
    fim se ;
fim
```

Atividade prática introdutória sobre recursividade

- 3) A sequência de Fibonacci é uma progressão numérica inteira estritamente positiva que é definida por:

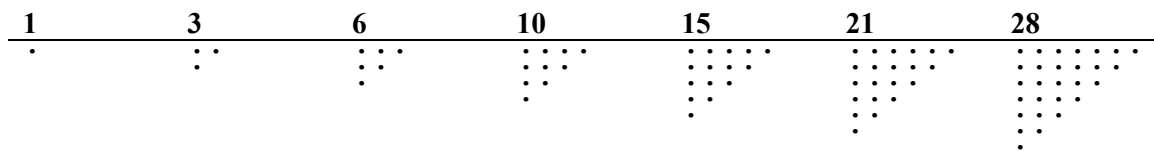
$$\text{Fib}(1) = 1$$

$$\text{Fib}(2) = 1$$

$$\text{Fib}(N) = \text{Fib}(N-1) + \text{Fib}(N-2), \text{ para } N > 2$$

Escreva um programa que solicite do usuário um número inteiro I estritamente positivo (ou seja, maior que zero) e determina o I -ésimo número da sequência de Fibonacci. Utilize uma função recursiva para gerar esse valor.

- 4) Um número triangular é aquele que pode ser representado na forma de um triângulo equilátero. Os primeiros números triangulares são 1, 3, 6, 10, 15, 21 e 28, conforme ilustra a figura a seguir.



Uma definição recursiva da geração do n -ésimo número triangular é:

- Primeira linha tem n elementos
- Soma-se a próxima linha com $n-1$ elementos até que reste apenas um elemento.

Faça um programa que receba um número natural N e determine, por meio de uma rotina recursiva, qual é o N -ésimo número triangular. O programa também deverá imprimir na tela, por meio de outra rotina recursiva, o triângulo equivalente a esse N -ésimo número triangular.

- 5) Crie um programa que implemente e teste a versão não recursiva da função descrita a seguir. Depois, descreva o que faz essa rotina.

```
Função teste : int ( m, n: int ) _  
início  
    se (m mod n) = 0 então  
        retorne n ;  
    senão  
        retorne teste( n, m mod n ) ;  
    fim se ;  
fim
```

- 6) [Questão 21 do ENADE 2005, para Bacharel em Sistemas de Informação] No modo recursivo de representação, a descrição de um conceito faz referência ao próprio conceito. Julgue os itens abaixo, com relação à recursividade como paradigma de programação.

- I São elementos fundamentais de uma definição recursiva: o caso-base (base da recursão) e a reaplicação da definição.
- II O uso da recursão não é possível em linguagens com estruturas para orientação a objetos.
- III As linguagens de programação funcionais têm, na recursão, seu principal elemento de repetição.

Atividade prática introdutória sobre recursividade

IV No que diz respeito ao poder computacional, as estruturas iterativas e recursivas são equivalentes.

V Estruturas iterativas e recursivas não podem ser misturadas em um mesmo programa.

Estão certos apenas os itens

- a) I e IV.
- b) II e III.
- c) I, III e IV.
- d) I, III e V.
- e) II, IV e V.

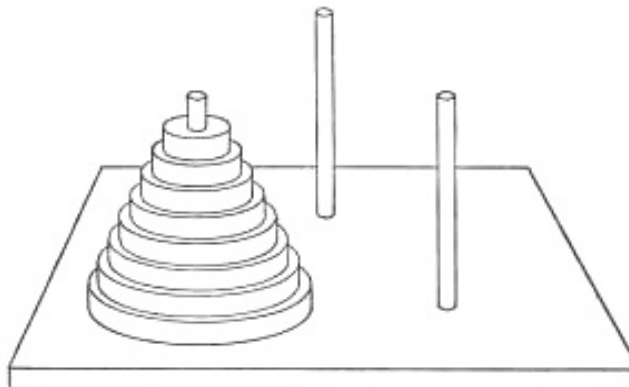
7) [Adaptação da questão 48 do Poscomp 2011] Observe a função recursiva a seguir.

```
int Prova(int N)
{
    if (N == 0)
        return 0;
    else
        return N * 2 - 1 + Prova(N - 1);
}
```

Considerando-se que essa função sempre será chamada com variável N contendo inteiros positivos, o seu valor de retorno será:

- a) O fatorial do valor armazenado em N .
- b) O valor armazenado em N elevado ao quadrado.
- c) O somatório dos N primeiros números inteiros positivos.
- d) O somatório dos N primeiros números pares positivos.
- e) 2 elevado ao valor armazenado em N .

8) [Questão 51 do ENADE 2005, para Ciência da Computação e Engenharia da Computação]



No famoso jogo da Torre de Hanoi, é dada uma torre com discos de raios diferentes, empilhados por tamanho decrescente em um dos três pinos dados, como ilustra a figura acima. O objetivo do jogo é transportar-se toda a torre para um dos outros pinos, de acordo com as seguintes regras: apenas um disco pode ser deslocado por vez, e, em

Atividade prática introdutória sobre recursividade

todo instante, todos os discos precisam estar em um dos três pinos; além disso, em nenhum momento, um disco pode ser colocado sobre um disco de raio menor que o dele; é claro que o terceiro pino pode ser usado como local temporário para os discos. Imaginando que se tenha uma situação em que a torre inicial tenha um conjunto de 5 discos, qual o número mínimo de movimentações de discos que deverão ser realizadas para se atingir o objetivo do jogo?

- a) 25
- b) 28
- c) 31
- d) 34
- e) 38

9) [Questão 23 do Poscomp 2004] O programa abaixo, quando executado para $A(1, 2)$, faz quantas chamadas recursivas (excluindo a primeira chamada da função)?

```
int A(int m, int n)
{
    if (m == 0)
        return n + 1;
    else
        if (n == 0)
            return A(m - 1, 1);
        else
            return A(m - 1, A(m, n - 1));
}
```

- a) 6
- b) 5
- c) 4
- d) 3
- e) 2

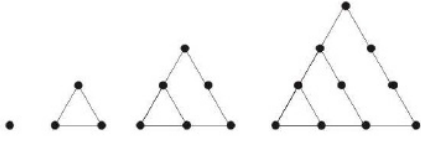
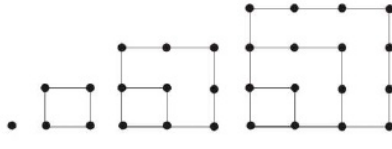
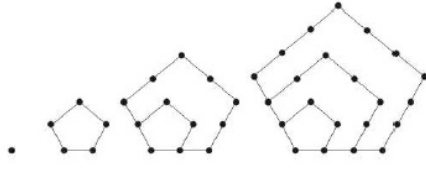
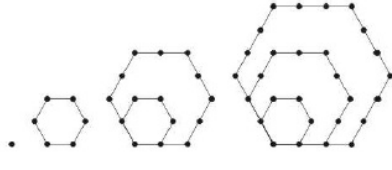
10) Resolva os problemas 2251, 1029 e 2132 do URI.

11) Resolva o problema do SPOJ-BR:

<http://br.spoj.com/problems/F91/>

Atividade prática introdutória sobre recursividade

12) [Questão 17 do Poscomp 2012] A tabela, a seguir mostra as figuras geométricas e suas respectivas relações recursivas.

	Figuras Geométricas	Relações Recursivas
F_3		$\begin{cases} T(1) = 1 \\ T(n) = T(n-1) + n, n > 1 \end{cases}$
F_4		$\begin{cases} Q(1) = 1 \\ Q(n) = Q(n-1) + 2n - 1, n > 1 \end{cases}$
F_5		$\begin{cases} P(1) = 1 \\ P(n) = P(n-1) + 3n - 2, n > 1 \end{cases}$
F_6		$\begin{cases} H(1) = 1 \\ H(n) = H(n-1) + 4n - 3, n > 1 \end{cases}$

Nesta tabela podem ser observadas as seguintes relações:

$$T(1) = 1 \text{ para } F_3; Q(2) = 4 \text{ para } F_4; P(3) = 12 \text{ para } F_5; H(4) = 28 \text{ para } F_6.$$

Com base na tabela e nas relações, assinale a alternativa que apresenta, corretamente, o número de pontos de F_{10} quando $n = 5$.

- a) 55
- b) 65
- c) 75
- d) 85
- e) 95

13) Crie um programa que implemente e teste a versão não recursiva da função descrita a seguir. Depois, descreva o que faz essa rotina. Para testar, utilize 0,001 como valor para o parâmetro *tol*.

```

Função teste : real ( x, r, tol: real )_
início
    se  $|x - r^2| \leq tol$  então
        retorne r ;
    senão
        retorne teste( x, (x/r + r)/2, tol ) ;
    fim se ;
fim
    
```

Atividade prática introdutória sobre recursividade

- 14) Escreva um programa que recebe 10 números inteiros e os armazena em um vetor. Em seguida, em um looping, receba do usuário um valor a ser pesquisado no vetor e faça a busca por meio da rotina recursiva de busca descrita a seguir. Saia do looping quando o usuário informar o valor -999.

```
Função Pesquisa: int (i: int, j: int, r: int)_  
início  
    se i = j então  
        se v[i] = r então  
            retorne 1  
        senão  
            retorne 0  
    fim se  
    senão  
        retorne Pesquisa(i, (i+j) / 2, r) ou Pesquisa((i+j)/2 + 1, j, r)  
    fim se  
fim
```

- 15) Escreva um programa que recebe 10 números inteiros e os armazena em um vetor. Depois faça a ordenação desses valores por meio de algum método de ordenação apresentado em aula. Por fim, em um looping, receba do usuário um valor a ser pesquisado no vetor e faça a busca por meio da rotina recursiva de busca binária descrita na página 170 do livro. Saia do looping quando o usuário informar o valor -999.