ALGORITMOS II

5ª LISTA DE EXERCÍCIOS – RECURSIVIDADE

Desenvolva todos os exercícios listados em Linguagem C.

1 Dada função `x` a seguir:

```
int x (int n){
    if (n == 0)
       return 0;
    if (n == 1)
       return 1;
    if (n == 2)
       return 2;
    return x(n-1) + x(n-2) + x(n-3);
}
```

a) Qual o valor de `x(6)`?
b) Quantas chamadas serão feitas na avaliação acima?

2 Escreva uma versão não recursiva das seguintes funções:

a)
```
int F (int i){
    if (i > 1)
      return i + F(i-1);
    return 1;
  }
```

b)
```
int F (int i){
    if (i == 0)
      return 0;
    if (i == 1)
      return 1;
    return F(i-1) + F(i-2);
  }
```

3 FATORIAL
a) Escreva uma função iterativa para calcular *N!* (fatorial de *N*).
b) Refaça o item anterior de maneira recursiva.
c) Execute a versão recursiva e não-recursiva da função fatorial e, examine quanto tempo cada uma exige conforme se aumenta o valor de *N*.

4 FIBONACCI. A função de Fibonacci é definida assim:

```
F(0) = 0
F(1) = 1
F(n) = F(n-1) + F(n-2). para n > 1.
```

Descreva a função F em linguagem C. Faça uma versão iterativa e uma recursiva.

5 Seja F a versão recursiva da função de Fibonacci. O cálculo do valor da expressão F(3) provocará a seguinte sequência de invocações da função:

```
F(3)
  F(2)
    F(1)
    F(0)
  F(1)
```

Qual a sequência de invocações da função provocada por F(5)?

6 a) Execute a versão recursiva e não-recursiva da função Fibonacci e, examine quanto tempo cada uma exige conforme se aumenta o valor de *n*.
b) Conte o número de adições necessárias para calcular `Fibo(n)` para 0 <=n<=10, por meio dos métodos iterativo e recursivo. Existe algum tipo de padrão?

7 PRODUTO
a) Escreva uma função iterativa para calcular o produto *a*b* usando adição, onde a e b são inteiros não negativos.
b) Refaça o item anterior de maneira recursiva.

8 Usando os exercícios anteriores, avalie cada um dos itens abaixo, usando ambas as definições iterativa e recursiva:

a) 6! b) 9!
c) 50*3 d) 5*4
e) F(10) f) F(11)

9 EXPONENCIAÇÃO. Escreva uma função recursiva para calcular o valor de *n* elevado ao expoente *x* (n^x), sendo x um número inteiro. Faça um programa, com reprocessamento, que utilize esta função.

10 Construa uma função recursiva para efetuar a análise combinatória. Faça um programa, com reprocessamento, que utilize esta função.

$$C_n^p = C_n^{p-1} + C_{n-1}^{p-1}$$

11 Escreva uma função recursiva capaz de inverter uma seqüência de caracteres. Faça um programa, com reprocessamento, que utilize esta função.

12 Escreva uma função recursiva que determine a soma dos *N* primeiros números naturais. Faça um programa, com reprocessamento, que utilize esta função.

13 Faça uma função recursiva que calcule a soma dos quadrados dos N primeiros números positivos.

14 Qual é a sequência numérica gerada pela seguinte função recursiva?

```
int F (int n){
  if (n == 0)
    return 1;
  if (n == 1)
    return 2;
  return 2*F(n-2)*F(n-1);
}
```

15 A expressão em C, *m % n*, resulta o resto de *m* ao dividir *n*. Defina o máximo divisor comum (MDC) de dois inteiros, *x* e *y*, por:

```
mdc(x,y) = y,                    se (y <= x) e (x % y = 0)
mdc(x,y) = mdc(y,x),             se (x < y)
mdc(x,y) = mdc(y,x % y),caso contrário
```

Escreva uma função recursiva para calcular `mdc(x,y)`. Descubra um método iterativo para calcular essa função

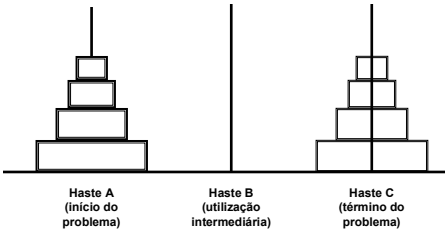
16 A função de Ackerman é definida recursivamente sobre os inteiros não-negativos, como segue:

$a(m,n) = n+1$, se $m=0$
$a(m,n) = a(m-1,1)$, se $m \neq 0$ e $n=0$
$a(m,n) = a(m-1,a(m,n-1))$, se $m \neq 0$ e $n \neq 0$

a) Usando a definição anterior, demonstre que `a(2,2)` é igual a 7.
b) Prove que `a(m,n)` está definido para todos os inteiros não-negativos *m* e *n*.
c) Você consegue descobrir um método iterativo para calcular `a(m,n)`?

17 A Torre de Hanói é um jogo com uma base histórica citada num ritual praticado por sacerdotes brâmanes para predizer o fim do mundo. O jogo inicia-se com uma série de anéis de ouro de tamanhos decrescentes, empilhados numa haste presa numa tábua (os sacerdotes brâmanes usavam 64 anéis). O objetivo é empilhar todos os anéis numa segunda haste em ordem decrescente de tamanho. Antes que isso possa ser feito, o fim do mundo chegará. Uma terceira haste está disponível para uso como armazenamento intermediário. Construa um programa recursivo para o problema da torre de Hanói. O problema consiste em transferir N discos, utilizando três hastes, que estão na haste A para a haste C, utilizando as seguintes regras:

- pode-se mover somente um disco de cada vez;
- nenhum disco deve ser colocado sobre um disco menor;
- qualquer disco pode ser movido de qualquer haste para qualquer haste, respeitando a regra acima.



18 Escreva uma função recursiva que retorne a soma dos elementos de um vetor. Faça um programa, com reprocessamento, que utilize esta função.

19 Faça uma função recursiva para somar os elementos de um vetor por bisseção. Utilize reprocessamento no programa principal.

20 Faça uma função recursiva que pesquise um valor em um vetor por bisseção. Faça um programa, com reprocessamento, que utilize esta função.

21 Considere *v* como um vetor de inteiros. Escreva funções recursivas para calcular:

a) o elemento mínimo do vetor;
b) o elemento máximo do vetor;
c) a soma dos elementos do vetor;
d) o produto dos elementos do vetor;
e) a média dos elementos do vetor.

22 Faça uma função recursiva que pesquise um valor em um vetor usando busca binária. Faça um programa, com reprocessamento, que utilize esta função.

23 Se um vetor contiver n elementos, qual o número máximo de chamadas recursivas feitas pelo algoritmo de busca binária?

24 Determine o que a seguinte função recursiva calcula. Escreva uma função iterativa para atingir o mesmo objetivo.

```
int Func (int n){
  if (n == 0)
    return 0;
  return n + Func(n-1);
}
```

25 Escreva uma função recursiva para calcular o número de seqüências de n dígitos binários que não contém dois 1's seguidos.

26 Crie uma função para gerar a seqüência `T`, onde `T(1) = 1`, `T(n)=T(n-1)+3` para `n>=2`. Faça um programa que utilize esta função.

27 Faça programas para calcular um termo qualquer das seqüências:

a) `S(1)=10`, `S(n)=S(n-1)+10` para `n>=2`;
b) `B(1)=1`, `B(n)=B(n-1)+n^2` para `n>=2`.

28 Dada a função `X` a seguir:

```
int X (int N, int M){
  if (N == M || M == 0)
    return 1;
  return X(N -1,M) + X(N -1,M -1);
}
```

a) Qual o valor de `X(5,3)`?
b) Qual o valor de `X(2,2)`?
c) Qual o valor de `X(3,5)`?
d) Quantas chamadas serão feitas nas avaliações acima?

29 Escreva uma função recursiva que, dado um número natural n, retorne-o em base binária.

30 Recursividade pode ser utilizada para gerar todas as possíveis permutações de um conjunto de símbolos. Por exemplo, existem seis permutações no conjunto de símbolos A, B e C: ABC, ACB, BAC, BCA, CBA e CAB. O conjunto de permutações de N símbolos é gerado tomando-se cada símbolo por vez e prefixando-o a todas as permutações que resultam dos N-1 símbolos restantes. Conseqüentemente, permutações num conjunto de símbolos podem ser especificadas em termos de permutações num conjunto menor de símbolos. Formular uma função recursiva para este problema.

31 Faça uma função recursiva para imprimir todas as permutações dos números de *1* a *n*, uma permutação por linha.

32 Verificar se uma palavra é palíndromo, utilizando uma função booleana recursiva.

33 Seja $v=(v_1,...,v_i,...,v_f,...,v_n)$ um vetor com *n* dígitos entre 0 e 9. Faça uma função recursiva para verificar se os elementos $(v_i,...,v_f)$ formam um número palíndromo, $1 \leq i \leq f \leq n$.

Observação: Pode considerar que os números podem começar com alguns dígitos 0's, assim, o número `0012100` é palíndromo.

34 **CÁLCULO DE DETERMINANTES POR CO-FATORES.** Seja *A* uma matriz quadrada de ordem *n*. O Menor Complementar M_{ij}, de um elemento a_{ij} da matriz *A* é definido como o determinante da matriz quadrada de ordem *(n-1)* obtida a partir da matriz *A*, excluindo os elementos da linha *i* e da coluna *j*. Assim, o determinante de uma matriz quadrada *A* de ordem *n* pode ser calculado da seguinte maneira:

$$Det(A) = a_{11}.M_{11} - a_{12}.M_{12} + a_{13}.M_{13} - ...$$

Faça uma função recursiva para calcular o determinante de uma matriz de ordem *n* usando o método descrito acima.

35 O cálculo da raiz quadrada de um número pode ser feito através da seguinte forma:

`A`, se `|A2 - N| < E`
`RAIZ(N,A,E) = RAIZ(N,(A2+N)/2A,E)`, caso contrário

em que `N`: número do qual se quer calcular a raiz quadrada;
`A`: uma aproximação deste valor;
`E`: erro admissível.

Defina esta função e teste para alguns valores. Liste a aproximação inicial dada para a raiz e todas as intermediárias.

36 Faça um programa que, utilizando uma função recursiva, gere todas as combinações possíveis para um cartão de apostas da loteria esportiva. A saída do problema deve seguir o formato apresentado abaixo. Para cada combinação obtida deverá ser impresso o cartão com a respectiva marcação dos jogos nas colunas 1, do meio (M) e 2. É necessário, ainda, que seja informado o número da combinação.

Jogo	01	02	03	04	05	06	07	08	09	10	11	12	13
1	X	X	X					X	X				
M				X	X								X
2						X	X			X	X	X	

Combinação Nro: 167435.00
Pressione uma tecla para continuar.

37 Considere o conjunto de todas as expressões aritméticas infixas válidas, completamente entre parênteses, que consiste em nomes de variáveis contendo apenas uma letra, um dígito e os quatro operadores +, -, * e /. A seguinte definição recursiva especifica o conjunto de expressões válidas:

- qualquer variável com uma única letra (A - Z) ou um dígito é uma expressão infixa válida;
- se a e b são expressões infixas válidas, então (a +b), (a - b), (a * b) e (a / b) são expressões infixas válidas;
- as únicas expressões infixas válidas são aquelas definidas pelos dois itens anteriores.

Formular uma função recursiva que receba na entrada uma cadeia de símbolos e produza, como saída, a mensagem EXPRESSAO VALIDA para uma expressão infixa válida e a mensagem EXPRESSAO INVALIDA para uma expressão inválida.

38 Sendo `x^n = x * x^(n-1)`, onde "^" significa elevar um número a uma potência, faça uma função recursiva para o cálculo da potência. Assuma potência inteira positiva, e em uma segunda versão considere também potências negativas.

39 Escreva uma função recursiva para imprimir os n primeiros números pares.

40 Desenvolva uma função recursiva que some dois números naturais, através de incrementos sucessivos.

41 Escreva uma função recursiva que liste todos os subconjuntos de duas letras para um conjunto dado de letras.
[A, C, E, K] → [A, C], [A, E], [A, K], [C, E], [C, K], [E, K]

42 Escreva uma função recursiva que gere todas as possíveis combinações para um jogo da MegaSena com 6 dezenas.

43 A partir de um vetor de números inteiros, escreva uma função que calcule a soma e o produto dos elementos do vetor.

44 **Triângulo de Sierpinski** - O triângulo de Sierpinsky é um fractal que foi descoberto pelo matemático polonês Waclav Sierpinsky (1882-1969) e que se constrói de forma iterativa, isto é, dividindo um triângulo equilátero em quatro triângulos semelhantes (note-se que um dos 4 triângulos resultantes está invertido relativamente ao original). O triângulo invertido é retirado do triângulo original, sobrando apenas os outros três. Aplica-se o mesmo procedimento a cada um dos três triângulos com a orientação original. De modo sucessivo vai-se aplicando o procedimento descrito acima.

Escreva uma função que gere o triângulo de Sierpinsky.

45 Faça uma função recursiva que calcule o valor da série S descrita a seguir para um valor n>0 a ser fornecido como parâmetro para a mesma:

$$S = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots + \frac{1}{n!}$$

46 Escreva uma função recursiva, `imprimeSerie(int i, int j, int k)`, que imprime na tela a série de valores do intervalo [i,j], com incremento k.

47 Escreva uma função recursiva, `int somaSerie(int i, int j, int)`, que retorna a soma da série de valores do intervalo [i,j], com incremento k.

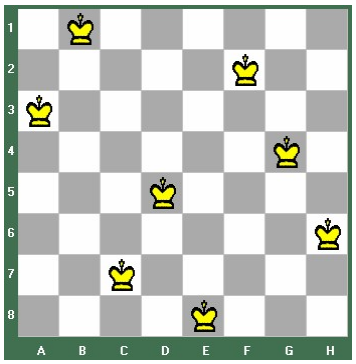
48 Faça uma função recursiva que calcule o valor da série S descrita a seguir para um valor n>0 a ser fornecido como parâmetro para a mesma.

$$S = 2 + \frac{5}{2} + \frac{10}{3} + ... + \frac{1+n^2}{n}$$

49 Escreva uma função recursiva que faça o seguinte: leia um número; se o número for negativo, a função pára; caso contrário, a função imprime o número e faz uma chamada recursiva a si mesma.

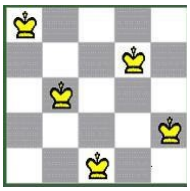
50 Escreva um programa que leia quatro valores inteiros positivos n_A, n_B, t_A e t_B, representando respectivamente as populações atuais de dois países A e B e as taxas de crescimento anual dessas populações, e determine se o país menos populoso poderá ultrapassar a população do outro país, supondo que as taxas de crescimento dessas populações não variam. Em caso afirmativo, o programa deverá determinar também o número de anos necessários para que isto aconteça. Utilize **funções recursivas** para resolver o problema.

51 **RAINHAS**. Escrever um programa que posicione 8 rainhas em um tabuleiro de xadrez de tal forma que cada uma delas não esteja ameaçada por qualquer uma das outras.



Uma solução

Inicialmente resolva o problema com 5 rainhas em um tabuleiro 5 x 5.


Solução- tabuleiro 5 x 5

52 Faça uma função recursiva que retorno o resultado da seguinte série:

$$S = \frac{1}{N} + \frac{2}{N-1} + \frac{3}{N-2} + \frac{4}{N-3} + ...$$

53 **Problema: F91 - SPOJ Problem Set (seletivas)**

McCarthy é um teórico famoso de ciência da computação. No seu trabalho, ele definiu uma função recursiva, chamada f91, que recebe como entrada um inteiro N e retorna um inteiro positivo definido como a seguir:

Se $N \leq 100$, *então* `f91(N) = f91(f91(N + 11));`
Se $N \geq 101$, *então* `f91(N) = N - 10.`

Escreva um programa que computa a função `f91` de McCarthy.

54 Qual o resultado da execução do programa abaixo?

```
int ff (int n) {
  if (n == 1)
    return 1;
  if (n % 2 == 0)
    return ff (n/2);
  return ff ((n-1)/2) + ff ((n+1)/2);
}
int main () {
  printf ("%d", ff(7));
}
```

55 Execute fusc (7,0).

```
int fusc (int n, int profund) {
  int i;
  for (i = 0; i < profund; ++i)
    printf ("  ");
  printf ("fusc(%d,%d)\n", n, profund);
  if (n == 1)
    return 1;
  if (n % 2 == 0)
    return fusc (n/2, profund+1);
  return fusc ((n-1)/2, profund+1) + fusc ((n+1)/2, profund+1);
}
```

56 Qual é o valor de f (1,10)? Escreva uma função equivalente que seja mais simples.

```
double f (double x, double y) {
  if (x >= y)
    return (x + y)/2;
  else return f (f (x+2, y-1), f (x+1, y-2));
}
```