INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS
Cid. Universitária Zeferino Vaz, Barão Geraldo
13083-852 Campinas SP

MC 102 - Algoritmos e programação de computadores
Segunda lista de exercícios

1. O programa abaixo lê o tamanho dos catetos de um triângulo retângulo e imprime o tamanho da hipotenusa e a área do triângulo. Complete as lacunas ou marque com um traço as que não devem ser preenchidas. Note que os argumentos podem ser passados tanto por referência como por valor.

```
#include <stdio.h>
#include <math.h>

 ..... pitagoras(float  ....cat1, float  ....cat2, float  ....hip)
{
   ....hip = sqrt(....cat1 * ....cat1 + ....cat2 * ....cat2);
   return  (...cat1 * ...cat2/2.0);
}

int  main()
{
   float  c1, c2, h, a;
   printf("Insira o tamanho dos catetos do triangulo retangulo\n");
   scanf("%f %f", &c1, &c2); /* Lê o tamanho dos catetos */

   ....a = pitagoras(....c1, ....c2, ....h);
   printf("A hipotenusa do triangulo retangulo eh: %.2f\n", ....h);
   printf("A area do triangulo retangulo eh: %.2f\n", ....a);
   return  0;
}
```

2. Faça uma função que calcule a aproximação para a integral:

$$\int_0^x e^{-u^2}\,du = x - \frac{x^3}{3*1!} + \frac{x^5}{5*2!} - \frac{x^7}{7*3!} + \cdots$$

Sua função deverá ter o seguinte protótipo:

double aprox_int(**float** x, **float** cte);

Sendo que x é o mesmo x da fórmula e *cte* é um valor utilizado como critério de parada da aproximação, isto é, a sua função deverá interromper o cálculo assim que o valor absoluto do i-ésimo termo for menor que *cte*.

3. O programa a seguir contém uma função principal *main* à esquerda e a implementação de quatro funções à direita. Execute o programa no papel, mantendo uma tabela com todos os valores que as variáveis dos programas assumem. Depois cheque sua solução, copiando o código fonte, o compilando e executando em um computador.

```
#include <stdio.h>

/* Pragma das funções */
void P(int i, int j, int *pk);
void Q(int h, int *pj);
void R(int *pi);
void S(int x, int y, int *pz);

int main() {
    int a, b, c;

    a = 0; b = 1; c = 2;

    Q(0, &c);
    Q(1, &a);
    Q(2, &b);

    a = 5; b = 8; c = 3;

    S(a, b, &c);
    S(7, a+b, &c);
    S(a*b, a/b, &c);

    return 0;
}
```

```
void P(int i, int j, int *pk) {
    *pk = *pk + 1;
    printf("%d\t%d\t%d\n", i, j, *pk);
}

void Q(int h, int *pj) {
    int i = *pj;
    if (h==0)
      P(i, h, pj);
    else if (h==1)
      P(h, *pj, &i);
    else
      R(&i);

    printf ("%d\t%d\t%d\n",i, *pj, h);
}

void R(int *pi) {
    *pi = *pi+1;
}

void S(int x, int y, int *pz) {
    *pz = x + y + *pz;
    printf("%d\t%d\t%d\n", x, y, *pz);
}
```

4. Simule a execução da função recursiva `ff` abaixo com os argumentos $n = 7$ e $ind = 0$. Depois implemente-a em um computador e cheque sua resposta.

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

5. Dados dois inteiros positivos m e n, escreva duas funções recursivas `quociente(m, n)` e `resto(m, n)`, que retornem, respectivamente, o quociente e o resto da divisão de m por n. Suas funções não podem usar multiplicações ou divisões.

6. Para determinar a representação binária de um número deve-se realizar sucessivas divisões deste número por 2 e quando seu resto for menor que 2, imprimir, do último para o primeiro, todos os restos das divisões. Por exemplo, 6 em representação binária é `1100` pois:
   6 / 2 = 3, resto 0
   3 / 2 = 1, resto 1
   1 / 2 = 0, resto 1
   Desenvolva uma função recursiva que imprime um inteiro positivo n qualquer em binário.

7. Desconsidere o conhecimento da função `sqrt` na linguagem C. Uma forma de se obter a raiz quadrada de um número qualquer x seria através de busca binária. Assuma que a raiz quadrada de x está entre 0 e x (se o número for negativo, retorne 0). Para sabermos se um palpite y é a raiz quadrada de x, basta testar se $y*y$ é próximo o suficiente de x ou, em outras palavras, se o módulo da diferença entre eles está dentro de uma tolerância definida. Caso contrário, podemos restringir a busca entre 0 e y ou entre y e x. Escreva uma implementação recursiva para função abaixo que implemente este algoritmo, para um valor de x qualquer, considerando 10^{-6} como tolerância para o cálculo do resultado.
   double raiz(**double** x);

8. Escreva um programa que leia um número inteiro n e um vetor de números reais com n valores. O programa deve conter a seguinte função:

```
int estat(float vet[], int n, float *media, float *mediana, float *desv);
```

   A função recebe um vetor com n valores reais e deve calcular a *media*, a *mediana* e o *desvio padrão* dentre os elementos do vetor. Note que, como estes valores devem ser atualizados pela função, eles estão sendo passados por referência. Além disso, a função deverá retornar a quantidade de elementos do vetor no intervalo $[media - desvio, media + desvio]$.

9. Faça um programa que contêm uma função recursiva que calcula o valor de m a partir de um vetor de números reais V (V_t é o t-ésimo número) com tamanho N e um valor $0 < \lambda < 1$.

$$m = \sum_{t=0}^{N-1} \lambda^{N-t} V_t$$

10. Escreva uma função que receba um vetor de inteiros como parâmetro e seu tamanho n e que retorne: (i) 1, se o vetor estiver ordenado em ordem não decrescente; (ii) -1, se o vetor estiver ordenado em ordem não crescente; e (iii) 0 se o vetor não estiver ordenado.

11. Faça uma função que receba como parâmetro um vetor de inteiros e seu tamanho n e *inverta* os seus elementos. A sua função principal deve imprimir o vetor antes e depois de chamar a função, quando o vetor já deve estar invertido. A sua função *não deve* empregar outros vetores, ou seja, deve alterar o vetor passado por referência.

12. Em uma eleição presidencial existem quatro candidatos. Os votos são informados através dos seguintes códigos:

    1, 2, 3, 4 ⇒ votos para os respectivos candidatos;
    5 ⇒ voto nulo;
    6 ⇒ voto em branco.

    Suponha que os votos tenham sido armazenados em um vetor. Faça um programa que simule uma eleição (carregando o vetor de votos) e posteriormente calcule as seguintes informações:

    (a) Total de votos para cada candidato, de votos brancos e nulos.

    (b) Porcentagem de votos para cada candidato sobre o total de votos válidos (sem considerar brancos e nulos.

    (c) Porcentagem de votos nulos sobre o total de votos;

    (d) Porcentagem de votos brancos sobre o total de votos.

    Seu programa deve ter as seguintes funções para calcular as informações solicitadas:

    float totaliza(**int** votos[], **int** codigo) e
    float porcentagem(**int** total1, **int** total2).

    A função `totaliza` recebe o vetor de votos e um código e retorna o número de votos com aquele código. A função `porcentagem` recebe o número de ocorrências de um conjunto de dados e o total sobre o qual a porcentagem deve ser calculada.

13. Escreva um programa que leia uma sequência de n dígitos e determine quantas sequências isoladas de números iguais existem. Por exemplo, para a sequência a seguir, com $n = 13$:

    3 4 4 1 2 5 5 5 2 2 6 2 2

    há 4 grupos de 1 elemento (3, 1, 2 e 6), 3 grupos de 2 elementos (4, 2 e 2) e 1 grupo de 3 elementos (5). O valor n deve ser um dado de entrada.

14. Faça uma função que receba como parâmetros um vetor de inteiros e o seu tamanho n e que modifique este vetor de maneira que todos os números divisíveis por 2 ocorram antes dos números não divisíveis por 2, sem utilizar outro vetor que não o original.

15. Dadas duas sequências de n e m elementos binários (0 ou 1), onde $n \leq m$, fazer um programa que verifique quantas vezes a primeira sequência ocorre na segunda segunda. Exemplo:

```
primeira sequência: 101
segunda sequência:  1101010011010
                     ---     ---
                       ---
resultado: 3
```

16. Implemente uma função recursiva que implementa a busca binária em um vetor de inteiros de tamanho n, ordenados entre si. O procedimento é similar ao da questão 7.

17. Dadas duas sequências, uma de n e a outra de m números inteiros entre 0 e 9, escrever um programa que calcule, empregando funções, a seqüência de números inteiros entre 0 e 9 que represente a sua soma. Por exemplo, para $n = 7$ e $m = 5$:

```
primeira sequência: 3  4  5  1  8  0  5
segunda sequência:        7  3  1  1  8
resultado:          3  5  2  4  9  2  3
```

18. Escreva um procedimento que *intercala* os valores de dois vetores inteiros ordenados de tamanhos i e j e gera um terceiro vetor, de tamanho $i + j$, também ordenado em ordem crescente. Por exemplo, para entrada $i = 7$, $V_1 = \{1, 3, 5, 5, 7, 9, 10\}$ e $j = 6$ $V_2 = \{2, 4, 6, 8, 8, 10\}$ a saída esperada é o vetor $V_{12} = \{1, 2, 3, 4, 5, 5, 6, 7, 8, 8, 9, 10, 10\}$ de tamanho 13.

19. O *Merge Sort* é um algoritmo de ordenação que recebe um vetor de tamanho n, o quebra em duas porções (de tamanhos $\lfloor n/2 \rfloor$ e $\lceil n/2 \rceil$), ordena cada porção e emprega a função *intercala* descrita no exercício 18 para gerar um vetor de tamanho n ordenado. Para ordenar as duas porções geradas, aplica-se o algoritmo recursivamente até que o tamanho das porções seja $i \leq 1$. Empregue a função *intercala* desenvolvida para implementar este algoritmo.

20. Codifique o algoritmo quicksort.

21. As moléculas de DNA podem ser representadas por cadeias de caracteres que usam um alfabeto de 4 letras: A (Adenina), C (Citosina), T (Timina) e G (Guanina). Crie uma função e um programa que utilize essa função, que procure ocorrências de uma subcadeia de DNA dentro de uma outra cadeia de DNA. Por exemplo, a subcadeia `CATA` ocorre na cadeia principal `TCATATGCAAATAGCTGCATACCGA` nas posições 2 e 18.

22. Crie um programa que leia uma sequência de caracteres, modifique a *string* lida de forma que as seus caracteres fiquem em ordem crescente, e imprima a mesma cadeia, já ordenada. Empregue o algoritmo de ordenação por seleção para ordenar a *string*.

23. A função `scanf` da biblioteca *stdio.h*, quando aplicada a valores de ponto flutuante, considera o "." como separador decimal, enquanto que no Brasil nós usamos a ",". Sua tarefa é criar uma versão brasileira, a `scanfBrasil`, que deve ler somente valores escritos na forma "XXXX,YYY" (com um número qualquer de dígitos antes ou depois da vírgula) e armazená-los em variáveis do tipo `double`. Além disso, a função deve retornar 0 se tudo ocorreu bem, e 1 quando o valor lido não é nem um valor de ponto decimal, nem um valor inteiro. No caso de erro, a função não deve armazenar nada na variável a ela passada. Utilize a seguinte `main`:

```
int main() {
    double v;
    if (scanfBrasil(&v) == 0)
      printf("Numero = %lf\n", v);
    else
      printf("Erro!\n");
    return 0;
}
```

24. Faça um programa que leia n cadeias de caracteres, determine seus tamanhos e ordene-as por seu tamanho, empregando o algoritmo de ordenação por inserção. Após ordenar as *strings* lidas, imprima-as ordenadamente, ou seja, imprima *strings* mais curtas antes das mais longas.

25. Escreva um programa que leia um inteiro positivo n e imprima um triângulo de n linhas, constituído por números com o seguinte formato:

```
(Para n=4)
4  3  2  1
3  2  1
2  1
1
```

26. Seja S um texto formado por letras maiúsculas, vírgulas, pontos e brancos, terminado pelo caracter '#' (que somente ocorre no fim do texto). Escreva um programa que leia os caracteres de S um por vez e imprima o número de palavras com comprimento menor ou igual a 5.

27. Escreva um programa que leia duas palavras do teclado e determina se a segunda é um *anagrama* da primeira. Uma palavra é um anagrama de outra se todas as letras de uma ocorrem na outra, *em mesmo número, independente da posição.* Exemplos: `ROMA, MORA, ORAM, AMOR, RAMO` são anagramas entre si.