

**MAC2166 – Introdução à Computação**  
**ESCOLA POLITÉCNICA**  
Terceira Prova – 24 de junho de 2014

Nome: \_\_\_\_\_

Assinatura: \_\_\_\_\_

Nº USP: \_\_\_\_\_ Turma: \_\_\_\_\_

Professor: \_\_\_\_\_

**Instruções:**

1. Não destaque as folhas deste caderno.
2. A prova consta de 3 questões. Verifique antes de começar a prova se o seu caderno de questões está completo.
3. A prova pode ser feita a lápis. Cuidado com a legibilidade e, principalmente, com a TABULAÇÃO.
4. Qualquer questão pode ser resolvida em qualquer página. Se a questão não está na página correspondente ao enunciado basta indicar isto na página e escrever QUESTÃO X em letras ENORMES antes da solução.
5. Não é necessário apagar rascunhos no caderno de questões.
6. Não é permitido o uso de folhas avulsas para rascunho.
7. Não é permitido o uso de equipamentos eletrônicos.
8. Não é permitido a consulta a livros, apontamentos ou colegas.
9. Você pode definir funções e usá-las à vontade.

**DURAÇÃO DA PROVA: 2 horas**

Questão	Valor	Nota
1	3,5	
2	2,5	
3	4,0	
Total	10,0	

### QUESTÃO 1 (3,5 pontos)

(a) (1,5 ponto) Escreva uma função com protótipo

`float arctan(float x);` ou `double arctan(double x);`

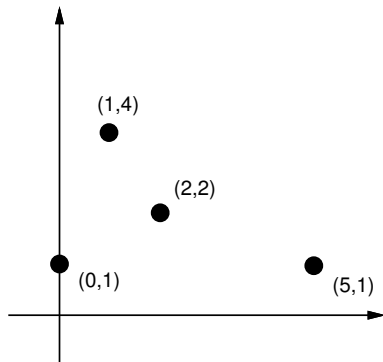
que recebe como parâmetro um número real  $x \in [0, 1]$  e retorna uma aproximação do arco tangente de  $x$  em radianos. A aproximação deve ser calculada usando a série

$$\arctan(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \dots$$

incluindo todos os termos até, inclusive, o primeiro tal que  $\frac{x^k}{k} < 0.0001$ .

**(b) (2,0 pontos)** Escreva um programa em C que lê um inteiro  $n$ ,  $n > 0$ , e as coordenadas reais de  $n$  pontos no primeiro quadrante ( $x \geq 0$  e  $y \geq 0$ , mas distintos de  $(0,0)$ ), e determina um ponto tal que a reta que passa pela origem e por esse ponto forma o menor ângulo com o eixo horizontal. O programa deve imprimir as coordenadas desse ponto e o correspondente ângulo. Se houver mais de um ponto que determina o menor ângulo, o programa pode imprimir qualquer um deles.

**Exemplo:** Observe a figura abaixo e verifique que os ângulos correspondentes aos pontos marcados são aproximadamente os mostrados ao lado



ponto	ângulo (rad.)	ângulo (graus)
(0,1)	$\pi/2 \approx 1.57$	$90^\circ$
(2,2)	$\pi/4 \approx 0.78$	$45^\circ$
(1,4)	$5\pi/12 \approx 1.31$	$75^\circ$
(5,1)	$\pi/16 \approx 0.19$	$11^\circ$

Dentre as retas que passam pela origem e por um desses pontos, o menor ângulo com o eixo horizontal é formado por aquela que passa por  $(5,1)$ .

Para calcular o valor do ângulo  $\alpha$  entre o eixo horizontal e a reta que passa pela origem e pelo ponto de coordenadas  $(x, y)$ , use a seguinte fórmula:

$$\alpha = \begin{cases} \arctan\left(\frac{y}{x}\right), & \text{caso } y < x, \\ \frac{\pi}{2} - \arctan\left(\frac{x}{y}\right), & \text{caso contrário,} \end{cases}$$

na qual  $\arctan$  é uma função que calcula o arco tangente de números entre 0 e 1.

Use 3.14 como valor aproximado de  $\pi$  e use a função do item anterior, mesmo que você não a tenha feito. Não é necessário reescrevê-la aqui.



## QUESTÃO 2 (2,5 pontos)

Lewis Carroll, o autor de *Alice no País das Maravilhas*, inventou o seguinte jogo de palavras, cujo nome em inglês é **word ladder**. Nele, são dadas duas palavras de **mesmo comprimento**  $t$ , e o jogador deve encontrar uma sequência de palavras intermediárias, todas de comprimento  $t$ , em que duas palavras consecutivas diferem em exatamente uma letra.

### Exemplos:

Palavras: HEXA e MOLE

Sequência: HEXA -> MEXA -> MELA -> MOLA -> MOLE

Palavras: GALHO e MISTA

Sequência: GALHO -> GANHO -> GANSO -> MANSO -> MANSA -> MASSA -> MISSA -> MISTA

Escreva um programa em C que, dado um inteiro positivo  $n$  e uma sequência de  $n$  palavras, verifica se a sequência é uma escada de palavras, ou seja, se todas as palavras da sequência têm o mesmo comprimento e se cada palavra da sequência, exceto a primeira, difere da anterior em exatamente uma letra. O seu programa deve imprimir **Sim**, **é uma escada de palavras**, ou **Não**, **não é uma escada de palavras**, conforme o caso.

Suponha que são dadas a seguinte definição e função:

```
#define MAX 30
int leia_string(char palavra[MAX]);
```

Essa função lê do teclado uma palavra, a armazena no vetor **palavra**, e retorna o comprimento da palavra lida. Suponha também que a palavra lida tem menos que **MAX** caracteres.



### QUESTÃO 3 (4,0 pontos)

Dada uma sequência de  $n$  ( $n > 0$ ) números inteiros, definimos a **mediana inteira** da seguinte forma:

- se  $n$  é ímpar, a mediana inteira é o valor que ocupa a posição central na sequência quando ordenada,
- se  $n$  é par, a mediana inteira é a parte inteira da média entre os valores que ocupam as posições mais centrais na sequência quando ordenada.

#### Exemplos:

Sequência: 7    -3    7    1    3

Sequência ordenada: -3    1    3    7    7

Mediana inteira: 3

Sequência: 11    10    6    3    1    0

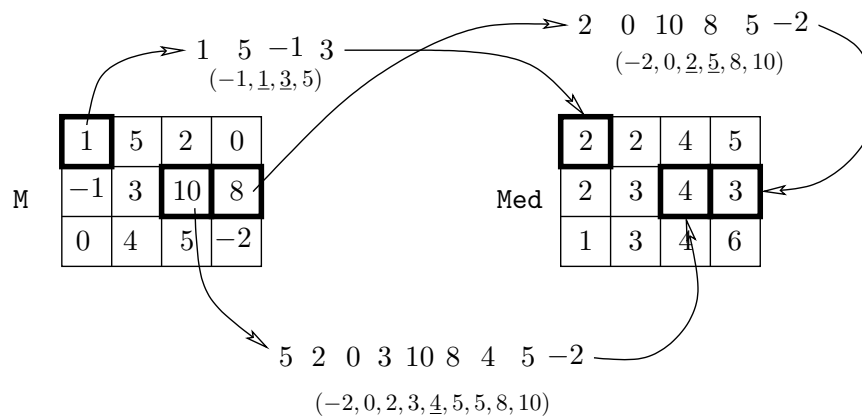
Sequência ordenada: 0    1    3    6    10    11

Mediana inteira: 4 (pois os dois elementos mais centrais são 3 e 6, resultando em média 4.5, cuja parte inteira é 4).

Neste exercício, os vizinhos de uma posição qualquer de uma matriz são aqueles definidos pela vizinhança do tipo rei (isto é, as 8 posições adjacentes), como no EP4.

Seja  $M$  uma matriz de inteiros. Desejamos calcular uma matriz  $Med$ , também inteira, de mesma dimensão da matriz  $M$ , e na qual o elemento  $Med[i][j]$  é exatamente igual à **mediana inteira** dos valores de  $M$  na vizinhança de  $[i][j]$  mais o valor em  $M[i][j]$ .

**Exemplo:** Na figura abaixo é mostrada uma matriz  $M$  e a respectiva matriz  $Med$ . Em destaque, a sequência de pontos (e respectiva sequência ordenada) considerada para o cálculo da mediana inteira em três coordenadas distintas da matriz.



Suponha que são dadas as seguintes definições e funções:

```
#define MAX 100
```

a) `void leia_matriz(int M[MAX][MAX], int *nlin, int *ncol);`

Lê uma matriz  $M$  de inteiros e devolve em  $*nlin$  e  $*ncol$  o número de linhas e de colunas de  $M$ , respectivamente. Suponha que a matriz lida tem no máximo  $MAX$  linhas e no máximo  $MAX$  colunas.

b) `void imprima_matriz(int M[MAX][MAX], int nlin, int ncol);`

Recebe uma matriz  $M$  de inteiros com  $nlin$  linhas e  $ncol$  colunas, e imprime a matriz.

c) `void ordene(int V[MAX], int k);`

Recebe um vetor  $V$  com  $k$  inteiros, e rearranja os elementos em  $V$  de forma que os inteiros fiquem ordenados em  $V$ .

**a) (2,0 pontos)** Escreva uma função

```
int vizinhos(int M[MAX][MAX], int nlin, int ncol, int i, int j, int V[MAX]);
```

que recebe uma matriz **M** de inteiros com **nlin** linhas e **ncol** colunas, os índices **i** (linha) e **j** (coluna) de uma posição da matriz, e devolve em **V** os valores de **M** em **[i][j]** e na vizinhança de **[i][j]**. A função deve retornar a quantidade de elementos armazenados no vetor **V** (que é igual ao número de vizinhos mais um).



**b) (2,0 pontos)** Escreva um programa em C que lê uma matriz **M** de inteiros, usando a função dada, e imprime a matriz mediana inteira calculada a partir de **M**. Suponha que o número de linhas e de colunas da matriz é no máximo **MAX**.

Para fazer este item, use as funções dadas e a do item (a), mesmo que você não a tenha feito.  
Não é necessário reescrever nem o protótipo, nem as funções aqui.