Universidade Federal de Goiás
Instituto de Informática
Introdução à Programação
Prova 3 - (Vetores, Matrizes e Strings)

Prof. Msc. Elias Batista Ferreira
Prof. Msc. Gilmar Ferreira Arantes
Prof. Dr. Gustavo Teodoro Laureano
Profa. Dra. Luciana Berretta
Prof. Dr. Thierson Rosa Couto

10/11/2020

Instruções para a Resolução dos Problemas

Esta avaliação é composta por 3 questões, variando do nível de dificuldade 1 a 3. O conteúdo abordado envolve, vetores, matrizes e strings, sendo uma questão para cada um destes tópicos.

Você deve resolvê-la seguindo as seguintes orientações:

1. A entrega desta prova é através do sistema Sharif.
2. Você deve implementar cada uma das questões no seu próprio computador e submeter ao Sharif, na área do problema correspondente à questão da prova.
3. A data e horário para entrega é 10/11/2020 - 23h59min.
4. A detecção de plágio (código duplicado) define nota 0 (zero) para os alunos envolvidos.

Sumário

1 Vetores - Frequência da média(++)

 (++)

Faça um programa que leia um vetor com *N* números inteiros (máximo de 1000 números). Calcule a média artimetica de **todos os números do vetor**, em seguida verifique qual das duas metades desse vetor possui maior quantidade de números acima da média.

Observações

- Para calcular a média, utilize todos os números do vetor.
- Se o vetor possuir quantidade impares de elementos, desconsiderar o elemento do meio, por exemplo, para um vetor de 9 elementos deve-se ignorar o quinto número durante a verificação de números maiores que a média.

Entrada

O programa deve ler um número inteiro *N* maior que 5. *N* indica a quantidade de números que o vetor deve armazenar.

Em seguida, leia e armazene os *N* números em um vetor de inteiros.

Saída

O programa deve apresentar em uma linha a média (com duas casas decimais), a quantidade de números maiores que a média na primeira metade do vetor, a quantidade de números maiores que a média na segunda metade do vetor, e um dos seguintes textos "PRIMEIRA METADE", "EMPATE", "SEGUNDA METADE".

Caso o número lido (*N*) não atenda as especificações da entrada, o programa deve apresentar a mensagem: "QUANTIDADE DE ELEMENTOS INVALIDOS!".

Exemplos

Entrada	Saída
9 15 30 45 90 60 75 36 50 80	53.44 1 2 SEGUNDA METADE

Entrada	Saída
16 100 200 2 90 65 800 20 96 45 63 85 96 150 30 25 150	126.06 2 2 EMPATE

Entrada	Saída
12 800 750 600 650 500 30 150 360 200 70 90 65	355.42 5 1 PRIMEIRA METADE

Entrada	Saída
3 45 800 750	QUANTIDADE DE ELEMENTOS INVALIDOS!

2 Matrizes - Matriz de permutação (+++)

 (+++)

Dizemos que uma matriz inteira $\mathbf{A}_{n\times n}$ é uma matriz de permutação se em cada linha e em cada coluna houver $n-1$ elementos nulos e um único elemento igual a 1.

A matriz A abaixo é de permutação:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

A matriz B abaixo não é de permutação.

$$B = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Dada uma matriz inteira $\mathbf{A}_{n\times n}$, verificar se A é de permutação.

Você deve implementar uma função que recebe a matriz e retorne 0 ou 1, sendo que o 0 (zero) indica que a matriz não é de permutação:

```
/**
 * Função que verifica se a matriz é de permutação
 * @param matriz Indica a matriz a ser verificada
 * @param n indica a dimensão da matriz
 * @param *soma parâmetro de saída, que armazenará a soma de todos os 'n' elementos
   da matriz.
 * @return int
 */
int ehPermutacao( int matriz[500][500], int n, int *soma );
```

Entrada

Na primeira linha ha um inteiro n, 1<n≤500, representando a ordem da matriz quadrada. A seguir haverá n linhas com n inteiros em cada linha separados por um espaço em branco cada, representando os elementos da matriz quadrada.

Saída

Deverá imprimir 3 (três) linhas:

- A dimensão da matriz (n).
- A mensagem "PERMUTACAO"ou "NAO EH PERMUTACAO", que representa se esta **é** ou **não** uma matriz de permutação.
- Soma de todos os elementos da matriz.

Exemplo

Entrada	Saída
4 0 1 0 0 0 0 1 0 1 0 0 0 0 0 0 1	4 PERMUTACAO 4

Entrada	Saída
3 2 -1 0 -1 2 0 0 0 1	3 NAO EH PERMUTACAO 3

3 Strings - Remove vogais (+++)

 (+++)

Escreva a função `remove_vogais` que remove todas a vogais de um texto e calcula a quantidade de vogais removidas. A função `remove_vogais` recebe como parâmetro uma *string* `str`, e um vetor de inteiros com 5 posições, corresndendo às vogais 'a', 'e', 'i', 'o' e 'u'. A função deve modificar a *string* passada como parâmetro e atualizar o vetor de ocorrências de vogais. Considere o tamanho máximo de 256 caracteres para a *string* de entrada.

```

/**
 * Função que remove vogais e calcula a quantidade de vogais removidas
 * @param str string de entrada
 * @param vogais vetor de 5 posições que contabiliza a quantidade
 *        de vogais removidas
 * @return A função atualiza os vetores str e vogais.
 */
void remove_vogais( char * str, int * vogais );
```

Entrada

Seu programa deve ler uma *string*.

Saída

Uma linha contendo a *string* modificada e outras 5 linhas contendo a quantidade das vogais 'a', 'e', 'i', 'o' e 'u' que foram removidas.

Exemplo

Entrada	Saída

`Fulano de Tal da Silva`	`Fln d Tl d Slv`
	`a: 4`
	`e: 1`
	`i: 1`
	`o: 1`
	`u: 1`

Entrada	Saída

`Ciencia DA COMPUTACAO`	`Cnc D CMPTC`
	`a: 4`
	`e: 1`
	`i: 2`
	`o: 2`
	`u: 1`