Exercícios – Lista 1

ATENÇÃO:	a) Todos os vetores utilizados têm que ser declarados na função ***main***. b) Não é permitido declarar/utilizar variáveis globais, exceto o ***#define***.

1) Faça um algoritmo para verificar se um número real lido pelo teclado encontra-se ou não em um vetor com 30 números reais (também lido pelo teclado). Utilize um **procedimento** para preencher o vetor e uma **função** para verificar se o número pertence ou não ao vetor. A impressão desta informação (se o número pertence ou não ao vetor) deve ser na função *main*.

2) Faça um algoritmo para ler (pelo teclado) um vetor com 15 elementos inteiros e depois inverter este mesmo vetor, **sem usar um vetor auxiliar**. Utilize três **procedimentos**: um para preencher o vetor, outro para invertê-lo e o terceiro para imprimi-lo após a inversão.

Obs.: O objetivo desse exercício não é imprimir o vetor em ordem inversa, mas sim colocar os elementos dentro do vetor em ordem inversa.

3) Considere um vetor com 40 números inteiros positivos gerados aleatoriamente de 1 a 100. Faça um algoritmo para verificar o número de vezes que um número inteiro positivo *n* lido pelo teclado aparece neste vetor. O programa também deve informar em quais posições (índices) do vetor o número aparece, caso ele pertença ao vetor. Utilize dois **procedimentos**: um para preencher o vetor e outro para realizar a verificação.

Obs. 1: O seu programa deve verificar **primeiro** quantas vezes o número *n* aparece no vetor. **Depois**, se ele aparecer alguma vez no vetor, imprimir as posições que ele aparece. Se ele não pertencer ao vetor, seu programa deve imprimir: "Número não pertence ao vetor".

Obs. 2: Veja o código abaixo que preenche um vetor com 10 números inteiros positivos gerados aleatoriamente de 1 a 10.

```
int main( ){

    int i, vetor[10];

    srand(time(NULL));  /* Gera uma semente aleatória */

    for(i=0;  i < 10;  i++){
        vetor[i] = rand( ) % 10 + 1; /* Gera números aleatórios de 1 a 10 */
        printf("%d ", vetor[i]);
    }
    return 0;
}
```

4) Faça um algoritmo para ler (pelo teclado) os 25 elementos de um vetor do tipo inteiro e verificar se o mesmo está em **ordem crescente**. Utilize um **procedimento** para preencher o vetor e uma **função** para a verificação. A impressão da informação (se o vetor está ou não em ordem crescente) deve ser na função *main*.

5) Faça um algoritmo para ler (pelo teclado) os 10 elementos de um vetor do tipo inteiro e verificar se os mesmos formam uma progressão aritmética. Utilize um **procedimento** para preencher o vetor e uma **função** para a verificação. A impressão da informação (se os elementos do vetor formam ou não uma progressão aritmética) deve ser na função *main*.

6) Faça um algoritmo para preencher um vetor (de tamanho 10) com elementos gerados aleatoriamente de 1 a 20, de maneira que não sejam inseridos números iguais no vetor, ou seja, todos os números contidos no vetor têm que ser distintos. Utilize dois **procedimentos**: um para preencher o vetor e outro para imprimi-lo.

7) Faça um algoritmo que leia pelo teclado os 15 números de um vetor do tipo inteiro e imprima na tela o **maior** elemento desse vetor e a posição em que ele se encontra. Utilize dois **procedimentos**: um para preencher o vetor e outro para imprimir as informações.

Obs.: Caso o maior elemento apareça mais de uma vez no vetor, a posição a ser impressa é a do **último** maior elemento.

8) Faça um algoritmo que leia pelo teclado os 15 números de um vetor do tipo inteiro e imprima na tela o **menor** elemento desse vetor e a posição em que ele se encontra. Utilize dois **procedimentos**: um para preencher o vetor e outro para imprimir as informações.

Obs.: Caso o menor elemento apareça mais de uma vez no vetor, a posição a ser impressa é a do **último** menor elemento.

9) Considere um vetor com 20 números inteiros positivos gerados aleatoriamente de 0 a 49. Faça um algoritmo que imprima na tela a quantidade de números pares e de números ímpares presentes no vetor. Utilize dois **procedimentos**: um para preencher o vetor e outro para imprimir as informações.

10) Faça um algoritmo para ler pelo teclado dois vetores A e B, cada um contendo 15 números inteiros, e em seguida preencher um vetor C, sendo que $C[i] = 2*A[i] + B[i]$, onde $0 \leq i \leq 14$. Utilize três **procedimentos**: um para ler os elementos dos vetores A e B, outro para preencher o vetor C e um terceiro para imprimir o vetor C após o preenchimento.

11) Considere um vetor A com 50 números inteiros positivos gerados aleatoriamente de 1 a 100. Faça um algoritmo para preencher outros dois vetores B e C, onde o vetor B deve conter apenas os números **pares** do vetor A e o vetor C deve conter apenas os números **ímpares** do vetor A. Utilize três procedimentos: um para preencher o vetor A, outro para preencher o vetor B e C, e um terceiro para imprimir os vetores B e C após o preenchimento.

12) Faça um algoritmo que leia uma frase (considerando os espaços) com no máximo 50 caracteres e imprima esta mesma frase sem os espaços. Faça dois **procedimentos**: um para ler a frase e outro para a impressão da mesma sem os espaços.

13) Faça um algoritmo que leia uma frase (considerando os espaços) com no máximo 100 caracteres e calcule o número de vogais e consoantes existentes na frase. Faça dois **procedimentos**: um para ler a frase e outro para imprimir as informações.

Obs.: Lembre-se que o usuário pode digitar letras maiúsculas e minúsculas.

14) Faça um algoritmo que leia uma frase (considerando os espaços) com no máximo 30 caracteres e verifique se uma letra (lida pelo teclado na função *main*) existe na frase. Faça um **procedimento** para ler a frase e uma **função** para a verificação. A impressão da informação tem que ser feita na função *main*.

Obs.: Lembre-se que o usuário pode digitar letras maiúsculas e minúsculas.

15) Faça um algoritmo que leia algumas palavras (com no máximo 10 caracteres) e calcule quantas palavras **Sim** e **Nao** foram digitadas. O programa deve parar de ler as palavras quando o usuário digitar **Fim**. O seu programa também deve informar a porcentagem de cada uma (**Sim** e **Nao**) em relação ao total de palavras digitadas.

IMPORTANTE:

A função para transformar uma letra para maiúscula é a função **toupper**( ) e para minúscula é **tolower**( ), ambas da biblioteca **ctype.h**.

Um exemplo da utilização dessas funções pode ser visto a seguir:

```c
#include <stdio.h>
#include <string.h>
#include <ctype.h>

void lerFrase(char *frase);
void imprimeFrase(char *frase);

int main(){

    char frase[31], letra;

    lerFrase(frase);

    imprimeFrase(frase);

    printf("\n\n");
    return 0;
}

void lerFrase(char *frase){

     printf("\nDigite a frase: ");
     scanf("%[^\n]s", frase);
}

void imprimeFrase(char *frase){

    int i;

    char letra;

    printf("\nFrase em maiusculo: ");
    for(i=0; i < strlen(frase); i++){
        letra = toupper(frase[i]);
        printf("%c", letra);
    }

    printf("\n\nFrase em minusculo: ");
    for(i=0; i < strlen(frase); i++){
        letra = tolower(frase[i]);
        printf("%c", letra);
    }

}
```