ATENÇÃO:	**a)** Todos os vetores utilizados têm que ser declarados na função ***main***. **b)** Não é permitido utilizar variáveis globais. O #**define** pode ser utilizado.

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). Crie e 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 utilizar um vetor auxiliar**. Crie e utilize três **procedimentos**: um para preencher o vetor, outro para invertê-lo e o terceiro para imprimi-lo após a inversão.

Obs.: O objetivo deste 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. Crie e 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: O exemplo da Figura 1 preenche e imprime um vetor com 10 números inteiros positivos de 1 a 10 gerados aleatoriamente.

```c
#include <stdio.h>

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;

}
```

Figura 1: Exemplo de uso das funções rand e srand.

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 não-decrescente. Crie e 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 não-decrescente) 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. Crie e 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. Crie e 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 deste vetor e a posição em que ele se encontra. Crie e 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 deste vetor e a posição em que ele se encontra. Crie e 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 primeiro 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. Crie e 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 $C_i = 2 \times A_i + B_i$, onde $0 \leq i \leq 14$. Crie e 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. Crie e utilize três **procedimentos**: um para preencher o vetor A, outro para preencher os vetores 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. Crie e utilize 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. Crie e utilize 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. Crie e utilize 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 algoritmo deve parar de ler as palavras quando o usuário digitar **Fim**. O seu algoritmo também deve informar a porcentagem de cada uma (**Sim** e **Nao**) em relação ao total de palavras digitadas. Crie e utilize um **procedimento** para realizar o que se pede.

16) Faça um algoritmo para ler (pelo teclado) os 30 elementos de um vetor do tipo inteiro e verificar se o mesmo está em ordem não-crescente. Crie e 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 não-crescente) deve ser na função *main*.

17) Faça um algoritmo para ler duas palavras (com no máximo 10 caracteres cada uma) e imprimir a menor delas ou se elas têm o mesmo tamanho. Crie e utilize dois **procedimentos**: um para ler as palavras e outro para imprimir o que se pede.

Obs.: Considere que o usuário digitará somente letras minúsculas.

18) Faça um algoritmo que leia duas frases (de no máximo 20 caracteres cada uma) e imprima se as frases possuem o mesmo comprimento, bem como se são iguais ou diferentes no conteúdo. Exemplo:

Frase 1: Brasil Hexa 2010

Frase 2: Brasil! Hexa 2010!

Resultado: As duas frases são de tamanhos diferentes. As duas frases possuem conteúdo distintos.

Obs.: Considere que as frases não iniciam e nem terminam com espaço, bem como só existe um único espaço entre as palavras das frases.

19) Faça um algoritmo para ler uma palavra pelo teclado (com no máximo 15 caracteres) e verificar se ela é **Palíndrome** ou não, **sem utilizar qualquer estrutura de dados auxiliar**. Crie e utilize um **procedimento** para ler a palavra e uma **função** para fazer a verificação. A impressão da mensagem (se a palavra é ou não Palíndrome) deve ser na função *main*.

Obs. 1: Uma palavra Palíndrome é aquela que lida de frente para trás e de trás para frente tem a mesma sequência de caracteres. Exemplos: arara, ovo, asa, radar, matam, etc;

Obs. 2: Lembre-se que o usuário pode digitar letras maiúsculas e minúsculas.

20) Faça um algoritmo que leia pelo teclado os 20 números do vetor A e construa um vetor B com os mesmos números de A, sendo que estes deverão estar invertidos. Ou seja, o primeiro número de A passa a ser o último

de B, o segundo de A passa a ser o penúltimo de B e assim por diante. Crie e utilize dois **procedimentos**: um para ler o vetor A e outro para preencher o vetor B.

21) Faça um algoritmo que preencha o vetor A com 20 números inteiros gerados aleatoriamente de 0 até 10. Em seguida, preencha o vetor B onde cada elemento B_i receba o fatorial do elemento A_i, com $0 \leq i \leq 19$. Crie e utilize dois **procedimentos**: um para preencher o vetor A e outro para preencher o vetor B. Crie e utilize também uma **função** para calcular o fatorial de um número.

22) Considere um vetor (lido pelo teclado) com 20 números ordenados de maneira não-decrescente. Faça um algoritmo para ler cinco números (um de cada vez) e, **a cada leitura**, inserir o número no vetor, **mantendo o mesmo ordenado**. Crie e utilize três **procedimentos**: um para preencher o vetor com os 20 números; outro para ler os cinco números, inseri-los no vetor e organizar o mesmo; e o último para imprimir o vetor após a organização.

Obs. 1: O procedimento para preencher o vetor pelo teclado só pode permitir inserir o número no vetor se o mesmo continuar em ordem não-decrescente. Ou seja, enquanto o número digitado não mantiver o vetor ordenado, o usuário deve digitar um novo número;

Obs. 2: Não é permitido utilizar qualquer estrutura de dados auxiliar.

Obs. 3: Não é permitido utilizar qualquer algoritmo de ordenação.

23) Faça um algoritmo para ler duas palavras (com no máximo 10 caracteres cada uma) e imprimir as mesmas em ordem alfabética. Crie e utilize dois **procedimentos**: um para ler as palavras e outro para imprimir o que se pede.

Obs.: Considere que o usuário digitará somente letras minúsculas.

24) Faça um algoritmo que leia uma palavra (com no máximo 15 caracteres) e depois embaralhe os caracteres da mesma, fazendo a sua impressão ao final. Por exemplo: recebendo a palavra **python**, pode ser retornado **npthyo**, **ophtyn** ou qualquer outra combinação possível, de forma aleatória. Crie e utilize dois **procedimentos**: um para ler a palavra e outro para embaralhar e imprimir a mesma ao final.

Obs. 1: Não é permitido utilizar qualquer estrutura de dados auxiliar.

Obs. 2: Considere que o usuário digitará somente letras minúsculas.

25) Considere um vetor com 100 números lidos pelo teclado. Faça um algoritmo que imprima todos os números que aparecem somente uma vez no vetor. Crie e utilize dois **procedimentos**: uma para preencher o vetor e outro para imprimir o que se pede. Exemplo com um vetor de 10 números:

Vetor: {3, 1, 9, 8, 3, 10, 1, 5, 9, 1}

Números que aparecem somente uma vez: 8, 10, 5

Obs.: Não é permitido utilizar qualquer estrutura de dados auxiliar.

26) Considere um vetor com 100 números lidos pelo teclado. Faça um algoritmo para ler pelo teclado (na função *main*) um número inteiro n ($1 \leq n \leq 100$) e imprimir todos os números que aparecem no vetor exatamente n vezes. Caso nenhum número apareça exatamente n vezes no vetor, esta informação deve ser impressa. Crie e utilize dois **procedimentos**: uma para preencher o vetor e outro para imprimir o que se pede. Exemplo com um vetor de 10 números e $n = 2$:

Vetor: {3, 1, 9, 8, 3, 10, 1, 5, 9, 1}

Números que aparecem exatamente 2 vezes: 3, 9

27) Considere um vetor com 50 números inteiros lidos pelo teclado. Faça um algoritmo que imprima o segundo maior número presente no vetor. Crie e utilize dois **procedimentos**: uma para preencher o vetor e outro para imprimir o segundo maior número.

Obs. 1: Não é permitido utilizar qualquer estrutura de dados auxiliar.

Obs. 2: Não é permitido ordenar o vetor.

28) Considere o vetor V com 50 números inteiros gerados aleatoriamente de 1 a 100. Faça um algoritmo que modifique V de modo que:

$$V_i = \sum_{\substack{j=0,\\ i \neq j}}^{49} V_j, \quad 0 \leq i \leq 49$$

Crie e utilize três **procedimentos**: um para preencher o vetor, outro para modificá-lo e o terceiro para realizar a impressão do vetor antes e depois da modificação.

Obs.: Não é permitido utilizar qualquer estrutura de dados auxiliar.

29) Considere o vetor V com n números inteiros lidos pelo teclado, sendo $n = 30$. Faça um algoritmo para ler pelo teclado na função *main* o número inteiro k ($1 \leq k \leq 4$) e imprimir o que se pede na expressão abaixo:

$$\underset{0 \leq i \leq n-k}{Max} \sum_{j=i}^{i+(k-1)} V_j$$

Crie e utilize um **procedimento** para ler o vetor e uma **função** para retornar o que se pede. A impressão desta informação deve ser realizada na função *main*.

Obs.: Não é permitido utilizar qualquer estrutura de dados auxiliar.

30) Considere os vetores V^1 e V^2, ambos com 40 números inteiros lidos pelo teclado, e dois números inteiros x ($x \neq 0$) e y ($y \neq 0$) lidos pelo teclado na função *main*. Faça um algoritmo para verificar se a condição abaixo é verdadeira:

$$\sum_{\substack{i=0, \\ i\ \acute{i}mpar}}^{39} x \times |V_i^1| \quad \leq \quad \prod_{\substack{j=0, \\ j\ par}}^{39} y \times V_j^2$$

Crie e utilize um **procedimento** para ler os dois vetores e uma **função** para retornar 1, se a condição é verdadeira, ou retornar 0 caso contrário. A impressão da informação (se a condição é verdadeira ou não) deve ser realizada na função *main*.

Obs.: Não é permitido utilizar qualquer estrutura de dados auxiliar.

IMPORTANTE

A função para transformar uma letra para maiúscula é a **toupper** e para minúscula é a **tolower**, ambas da biblioteca **ctype.h**. Um exemplo da utilização destas funções pode ser visto na Figura 2.

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

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

int main(){

    char frase[31];

    lerFrase(frase);

    imprimeFrase(frase);

    printf("\n\n");

    return 0;
}

void lerFrase(char *frase){

    printf("\nDigite a frase: ");
    scanf("%[^\n]s", frase); //Leitura da frase podendo conter espaços
}

void imprimeFrase(char *frase){

    int i;

    char letra;

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

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

}
```

Figura 2: Exemplo de uso das funções toupper e tolower.