Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Algoritmos e Programação de Computadores

Atividades a serem desenvolvidas nas sessões de Laboratório

Sessão 10:

Objetivos:
Exercitar a elaboração de programas que utilizam **estruturas de dados do tipo matriz.**

Atividades:

1. Criar o programa abaixo

Exemplo 1:

```
#include <stdio.h>

int main () {
    int i, j;
    int MAT[3][3];

    for (i=0; i<3; i++) {     /* leitura da matriz */
        for (j=0; j<3; j++) {
             scanf("%d",&MAT[i][j]);
        }
    }
    for (i=0; i<3; i++) {    /* impressão da matriz */
        for (j=0; j<3; j++) {
            printf("%d\t",MAT[i][j]);
        }
        printf("\n");
    }
    getchar();
    getchar();
    return 0;
}
```

a) Corrija todos os erros sintáticos (se houver);
b) Compile, execute e verifique o resultado do programa;
c) Retire o comando `printf("\n");` e execute o programa. Observe a saída do programa.

2. Fazer um programa que lê dois valores inteiros m e n, onde m e n indicam, respectivamente, o número de linhas e colunas de uma matriz A do tipo real. Fazer um programa com duas funções a serem chamadas na main(): uma para ler a matriz A e outra para verificar se existem elementos repetidos em A e indicar quantas vezes aquele elemento aparece em A. Essa informação sobre a repetição do elemento deve ser mostrada na main(). A função deve retornar

(por referência) o elemento que se repete e assim como o número de vezes. Assuma que apenas um elemento pode se repetir na matriz.

3. Faça um programa com duas funções que serão chamadas na main(). Uma para ler uma matriz MAT 4 x 4 de elementos inteiros, e outras para verificar o número de linhas e o número de colunas nulas da matriz. Uma linha ou coluna é nula quando todos os seus elementos são iguais a zero. A informação deve ser mostrada na main(). A função deve retornar por referência as duas informações, ou seja, o número de linhas e colunas nulas.
Exemplo:

```
1 0 2 3
4 0 5 6
0 0 0 0
0 0 0 0
```

A matriz informada tem 2 linhas nulas e 1 coluna nula

4. Fazer um programa que chame três funções na main(): (1) uma para ler uma matriz M[1..6,1..8]; (2) outra que verifica, para cada linha da matriz lida a quantidade de elementos negativos em cada linha. A função deve armazenar em cada posição de um vetor C o número de elementos negativos na linha correspondente de M, mesmo que seja zero. O vetor deve ser retornado por referência para a main(); (3) a terceira função deve ser para mostrar o vetor C na tela.

5. Fazer um programa que chame na main 3 funções: (1) para ler uma matriz M[1..5,1..5]: (2) outra para ler um vetor V[1..5]; (3) uma terceira função que recebe a matriz M e o vetor V e verifica se o vetor é igual a uma das linhas da matriz. A função deve retornar para a main o número da linha que o vetor for igual ou então -1 no caso de não encontrar nenhuma linha igual. Na main(), deve ser mostrada uma das seguintes mensagens conforme o caso: *V É IGUAL À LINHA* K (some 1 ao k na hora de mostrar ao usuário) ou NÃO HÁ LINHA IGUAL AO VETOR NA MATRIZ. Suponha que no máximo uma linha da matriz pode ser igual ao vetor.

6. Você já jogou "Campo minado"? Faça um programa com três funções que serão chamadas na main(): (1) uma para ler uma matriz 10 x 10 de caracteres do teclado. Cada caractere pode ser: * (asterisco) representa uma bomba na coordenada lida; - (traço) representa um local sem bomba. (2) uma função que recebe a matriz lida e cria uma **nova matriz de inteiros** 10 x 10 que contenha para cada posição [i,j] o número de bombas na vizinhança referente a cada célula da matriz de caracteres lida. Essa nova matriz de inteiros será retornada por referência para a main(). (3) Em seguida, uma função para mostrar a nova matriz de inteiros gerada pela função. Obs: Cada posição tem no máximo 8 vizinhos (as diagonais contam!!!). Veja o exemplo para uma matriz 5 x 5:

Entrada:	Saída:
* - - - -	0 1 0 0 0
- - - - -	3 4 3 3 2
* * * * *	1 2 2 3 2
- - - - *	3 4 3 5 3
* - - - *	0 1 0 2 1