Lista de Exercício #Especial

Programação

01. Faça um programa que receba um valor inteiro n, crie uma matriz A de tamanho n x n dinamicamente. Preencha a matriz A e valide-a (repita o preenchimento, caso não atenta ao requisito estabelecido a seguir) para que ela possua todos os elementos diferentes. Após validada a matriz A, gere uma matriz identidade B de tamanho n x n. Some a matriz A com a matriz B, gerando a matriz C. Encontre o maior elemento da matriz C. Ao final, imprima as matrizes A, B e C. Você precisará elaborar as seguintes funções:

a. main() <- fazendo tudo que foi solicitado na questão, com o auxílio das funções abaixo;
b. int **matrizIdentidade(int n) <- gera matriz identidade de tamanho n;
c. int validaMatriz(int **A, int n) <- retorna 0 (tem alguém repetido) ou 1 (tudo diferente) de acordo com a matriz A de tamanho n;
d. int **somaMatrizes(int **A, int **B, int n) <- soma matriz A e B (elemento por elemento) gerando a matriz de retorno, todas de tamanho n; (2,0 pt)
e. int maior(int **C, int n) <- retorna o maior elemento da matriz C de tamanho n;
f. void imprimirMatriz(int **A, int n) <- imprime a matriz A de tamanho n.

02. Faça um programa que aloque dinamicamente uma matriz quadrada de tamanho n x n e um vetor de tamanho n. O valor de n deve ser dado pelo usuário. Com a matriz preenchida, peça para o usuário digitar um número. Encontre-o nessa matriz. Caso o número não exista encerre o programa e diga "Número inexistente". Caso exista, continue. Continuando o programa, você terá em mãos a linha e coluna do elemento encontrado. Pegue todos os elementos desta linha e copie-os no vetor. Com o vetor preenchido, encontre o maior elemento do vetor e imprima-o. Você precisará elaborar as seguintes funções:

a. int maior (int *V, int n) <- retorna o maior elemento do vetor V de tamanho n.
b. int pos(int **A, int n, int m, int *lin, int *col) <- retorna 0 se não encontrar o elemento, retorna 1 se encontrar. Em lin e col deve conter as posições do elemento encontrado na matriz A de tamanho n x m.

03. Faça um programa para dado um tamanho específico imprimir um quadrado, uma pirâmide e um sinal de soma (+). Você precisará elaborar as seguintes funções:

a. int **quadrado (int n) <- retorna uma matriz n x n preenchida de zeros e uns de modo que os uns indiquem o formato do quadrado de tamanho n.
   Exemplo com n = 5
   11111
   10001
   10001
   10001
   11111
b. int **piramide (int n) <- retorna uma matriz preenchida de zeros e uns de modo que os uns indiquem o formato de uma pirâmide de tamanho n. Perceba que a matriz não terá o tamanho de n x n, pois na largura será necessário mais espaço.

Exemplo com n = 5
000010000
000111000
001111100
011111110
111111111

c. void imprimir (int **D, int n, int m) <- função imprime o desenho de uma matriz n x m de modo que cada 1 vira # e cada 0 vira um espaço.
Exemplo da matriz da pirâmide resultaria em:

```
    #
   ###
  #####
 #######
#########
```

04. Faça um programa que aloque dinamicamente uma matriz de caracteres de tamanho n x m. Tanto n como m devem ser digitados pelo usuário. Preencha a matriz e use todas as funções abaixo, desmontrando seu uso no main():

a. int * extrairLinha(char **M, int n, int m, int l) <- retorne NULL se a linha não existir na matriz M e retorne um vetor caso exista. Esse vetor retornado deverá conter a mesma informação presente na linha l da matriz M para o vetor E.
b. void converte(char **M, int n, int m, char c) <- encontre todas aparições de c na matriz M e transforme-o em maiusculo se for minusculo e vice-versa.
c. int somenteLetras(char **M, int n, int m) <- verifique se a matriz contem apenas letras, se sim retorne 1, se não retorne 0.

05. Faça um programa que receba um nome (use um scanf mesmo, para facilitar) e inverta-o. Com o novo string, gere uma matriz com a função da novaMatriz(), no qual os parâmetros n é o tamanho do string, m é o tamanho da string desconsiderando caracteres iguais, x e y são os valores inteiros da primeiras e última letra do string, respectivamente. Depois, o usuário deverá escolher realizar um corte na matriz indicando uma coluna e qual lado deverá cortar (esquerdo ou direito). Imprima a matriz antes do recorte e a matriz recortada. Use as funções abaixo para ajudá-lo a criar o programa:

a. void ordenar (char *x)
Entrada: um string x
Processamento: ordena a string x de modo crescente
Saída: não tem retorno
Exemplo: "texto" resultaria em "eottx"
b. int **novaMatriz(int n, int m, int x, int y)
Entrada: quatro inteiros n, m, x e y
Processamento: gere uma matriz dinamicamente de tamanho n (linhas) por m (colunas), preencha-a com os números x e y de modo que na borda tenha o mesmo valor que x e nos espaços centrais fique o mesmo valor que y.
Saída: retorne a matriz criada
Exemplo: 3, 4, 2, 3 resultaria na matriz: [[2, 2, 2, 2], [2, 3, 3, 2], [2, 2, 2, 2]]
c. int **cortar(int **a, int n, int m, int coluna, char lado)
Entrada: uma matriz a; três inteiros n, m, coluna; e um char lado
Processamento: gere uma matriz dinamicamente que deverá conter um

recorte da matriz a (de tamanho n x m) baseado na coluna e lado (esquerdo 'E' ou direito 'D') escolhido.
Saída: nova matriz baseada no recorte da matriz a feito no processamento

06. Faça um programa que receba um nome e valide até que o usuário digite um palindromo (use um scanf mesmo, para facilitar). Com o palindromo, gere uma matriz com a função da matrizXadrez(), no qual os parâmetros n e m são iguais ao tamanho do palindromo; x e y são os valores inteiros das duas primeiras letras do palindromo; imprima a matriz. Depois, o usuário deverá escolher uma linha ou coluna para extrair e qual linha ou coluna deseja extrair, use a função extrair() para isso; imprima a linha extraída. Use as funções abaixo para ajudá-lo a criar o programa:

a. int palindromo (char *x)
   Entrada: um string x
   Saída: retorna 1 se x for um palindromo ou 0 se não for
b. int **matrizXadrez(int n, int m, int x, int y)
   Entrada: quatro inteiros n, m, x e y
   Processamento: gere uma matriz dinamicamente de tamanho n por m, preencha-a com os números x e y em organização xadrez
   Saída: retorne a matriz criada
   Exemplo: 2, 2, 2, 3 resultaria na matriz: [[2, 3], [3, 2]]
c. int *extrair(int **a, int n, int o)
   Entrada: uma matriz a e dois inteiros n e o
   Processamento: gere um vetor dinamicamente que deverá conter uma linha (o == 0) ou coluna (o == 1) da matriz. A linha ou coluna será indicada pelo n.
   Saída: vetor preenchido com os dados da matriz de acordo com o processamento

07. Faça um programa que receba n, gere uma matriz A de tamanho n x n com a função alocaMatriz. Depois receba três inteiros e preencha a matriz A com a função preenchaMatriz. Depois, receba um inteiro e altere a matriz A com a função matrizSecundaria. Por fim, coloque em um vetor o resultado da função extrairPrincipal e imprima esse vetor. Use as funções abaixo para ajudá-lo a criar o programa:

a. int **alocaMatriz(int n) (1,0 pt) <- gera dinamicamente uma matriz de tamanho n x n;
b. void preenchaMatriz(int **m, int n, int a, int b, int c) (3,0pts) <- preenche a matriz m de tamanho n x n usando os valores a, b e c como no exemplo a seguir. Suponha a = 1, b = 2 e c = 3 e uma matriz 3x3, a função geraria a seguinte matriz:
   1 2 2
   3 1 2
   3 3 1
c. void matrizSecundaria(int **m, int n, int x) <- preenche os espaços da diagonal secundaria de uma matriz n x n com o valor de x. Suponha x = 5 usado na matriz da função anterior, teriamos a seguinte matriz:
   1 2 5
   3 5 2
   5 3 1
d. int *extrairPrincipal(int **m, int n) <- extrai a diagonal principal de uma matriz colocando seus valores em um vetor de tamanho n. Supondo o exemplo anterior, o vetor retornaria com os valores [1, 5, 1].

08. Faça um programa que gere duas matrizes A e B com função alocaMatriz, o tamanho das matrizes pode ser diferente e será definido pelo usuário. Depois preencha as duas matrizes. Encontre a media de ambas as matrizes. Se a media da matriz A for maior que media da matriz B calcule A x B, caso contrário calcule B x A. Se a multiplicação ocorreu, multiplique a matriz resultante por um inteiro dito pelo usuário. Exiba a matriz resultante final. Use as funções abaixo para ajudá-lo a criar o programa:

a. int **alocaMatriz(int n, int m) (1,0 pt) <- gera dinamicamente uma matriz de tamanho n x m;
b. float mediaMatriz(int **m, int n, int m) (2,0pts) <- calcula a média de uma matriz n x m
c. int **multiplicaMatriz(int **m, int n, int m, int l, int c) (3,0pts) <- dado duas matrizes, uma de tamanho n x m e outra de tamanho l x c, gere uma terceira matriz com o resultado da multiplicação. Para multipliar uma matriz, é necessário que a quantidade de colunas da primeira seja igual a quantidade de linhas da segunda. Se não for possível multiplicar a matriz, retorne NULL.
d. void multiplicaInteiro(int **m, int n, int m, int x) (2,0 pts) <- multiplique um inteiro x pela matriz de tamanho n x m.

09. Faça um programa que receba um texto S dado pelo usuário. Use a função contaVogal com o texto S, depois verifique qual a vogal mais presente nesse texto (função maior) e substitua por um caracter dado pelo usuário. Com o novo texto, use a função gerarVetor para gerar um vetor V. Imprima o vetor V. Use as funções abaixo para ajudá-lo a criar o programa:

a. void contarVogal (char *texto, int *a, int *e, int *i, int *o, int *u) <- receba um texto e conte quantas aparições de cada vogal há nesse texto, de modo que cada vogal tenha um contator separado (variáveis a, e, i, o, u). Isto é, dado o texto "roberto", os valores das variáveis devem ficar: a = 0, e = 1, i = 0, o = 2, u = 0.
b. char maior (int a, int e, int i, int o, int u) <- dado cinco inteiros, retorne o respectivo caracter que possuir o maior valor. Em caso de empate, sempre dê preferência a ordem alfabética. Isto é, se a = 5, e = 10, i = 3, o = 10, u = 1, retorne o caracter 'e'.
c. void substituir (char *texto, char vogal, char novo) <- substitua em um texto todo caracter igual a vogal pelo caracter novo. Isto é, dado o texto "roberto" com a vogal "o" e o novo "x" altere o texto para "rxbertx".
d. int *gerarVetor (char *texto) <- gera um novo vetor de inteiros V de mesmo tamanho do texto - 1, no qual cada posição desse novo vetor conterá a distância entre os caracteres consecutivos do texto. A distância entre 'a' e 'f' é 5.