ESTRUTURA E RECUPERAÇÃO DE DADOS A

PROJETO 1

1. Descrição do Projeto

Imagine que você foi contratado para desenvolver um programa para manipulação de matrizes de números reais que permita ao usuário:

1. **Declarar uma matriz:** fornecendo seu nome e dimensões desejadas.
2. **Destruir uma matriz:** fornecendo seu nome.
3. **Imprimir uma matriz:** fornecendo seu nome.
4. **Atribuir um elemento a uma matriz:** fornecendo o nome da matriz, as coordenadas e o valor.
5. **Atribuir uma linha a uma matriz:** fornecendo o nome da matriz, o índice da linha e a sequência de elementos.
6. **Atribuir uma coluna a uma matriz:** fornecendo o nome da matriz, o índice da coluna e a sequência de elementos.
7. **Transpor uma matriz:** fornecendo seu nome e o nome da matriz resultante (não precisa estar declarada).
8. **Somar duas matrizes:** fornecendo seus nomes e o nome de uma terceira matriz que deverá conter o resultado (não deve estar declarada). As 2 matrizes devem conter as mesmas dimensões.
9. **Dividir uma matriz por outra (elemento a elemento):** fornecendo seus nomes e o nome de uma terceira matriz que deverá conter o resultado (não deve estar declarada). As 2 matrizes devem conter as mesmas dimensões e a matriz que será o denominador da divisão não deve conter nenhum elemento igual a 0.
10. **Multiplicar uma matriz por outra:** fornecendo seus nomes e o nome de uma terceira matriz que deverá conter o resultado (não deve estar declarada). Sendo A_{m1xn1} e B_{m2xn2} as matrizes de entrada e AxB a operação desejada, *n1* deve ser igual a *m2*.
11. **Multiplicar duas matrizes (elemento a elemento):** fornecendo seus nomes e o nome de uma terceira matriz que deverá conter o resultado (não deve estar declarada). As 2 matrizes devem conter as mesmas dimensões.

Para exercer essas funções, o sistema deve manter uma lista ligada de matrizes onde cada nó deve conter o nome da matriz, um ponteiro para a matriz alocada dinamicamente (com as dimensões especificadas pelo usuário)

e um ponteiro para o próximo nó. Os nós desta lista ligada devem ser alocados conforme a necessidade, ou seja, quando o usuário quiser criar uma matriz ou chamar uma operação que resulte em uma matriz. A Figura 1 ilustra esta abordagem:

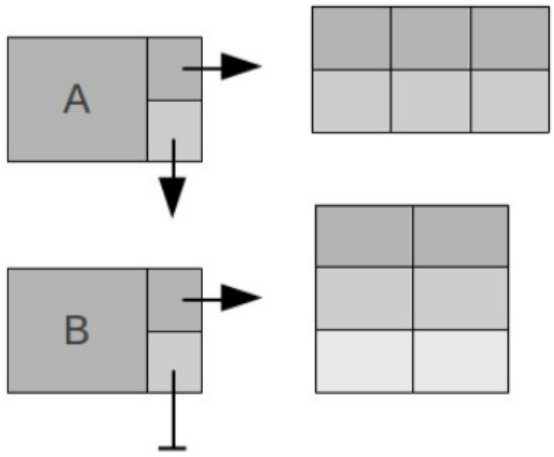

Figura 1: Lista ligada contendo as matrizes A_{2x3} e B_{3x2}

Quando o usuário optar por descartar uma matriz, esta deve ser liberada da memória, assim como seu nó deve ser liberado da lista ligada. Dois TAD's (Tipos Abstratos de Dados) devem ser implementados, um para a lista ligada e um para as matrizes.

2. Comandos e saídas

As funcionalidades do ambiente devem ser requisitadas por meio de comandos seguidos dos dados necessários. Os comandos são representados por duas letras maiúsculas. Os comandos que o sistema deve reconhecer são especificados a seguir. O símbolo // denota um comentário e portanto não faz parte da saída ou dos comandos.

Criar matriz

Comando	CM <nome sem espaços> <número de linhas> <número de colunas>
Saída	OK //Em caso de sucesso ERRO //Caso a matriz já exista ou alguma das dimensões não esteja entre 1 e 50

Destruir matriz

Comando	DM <nome>
Saída	OK //Em caso de sucesso ERRO //Caso a matriz não exista

Imprimir matriz

Comando	IM <nome>
Saída	$a_{1,1}$ ... $a_{1,m}$ ... //Separados por espaço (com formato %4.2f) $a_{n,1}$ ... $a_{n,m}$ ERRO //Caso a matriz não exista

Atribuir um elemento à matriz

Comando	AE <nome> <linha> <coluna> <valor>
Saída	OK //Em caso de sucesso ERRO //Caso a matriz não exista ou as coordenadas sejam inválidas

Atribuir uma linha à matriz

Comando	AL <nome> <linha> <val1> ... <val$_n$> #
Saída	OK //Em caso de sucesso ERRO //Caso a matriz não exista, a linha seja inválida ou a quantidade de valores não seja condizente com as dimensões da matriz

Atribuir uma coluna à matriz

Comando	AC <nome> <coluna> <val1> ... <val$_m$> #
Saída	OK //Em caso de sucesso ERRO //Caso a matriz não exista, a coluna seja inválida ou a quantidade de valores não seja condizente com as dimensões da matriz

Transpor uma matriz

Comando	TM <nome> <nome-resultado sem espaços>
Saída	//Imprime a matriz resultante (mesmo formato do comando IM) ERRO //Caso a matriz não exista ou a matriz resultante já exista

Somar duas matrizes

Comando	SM <nome1> <nome2> <nome-resultado sem espaços >
Saída	//Imprime matriz resultante (mesmo formato do comando IM) ERRO //Caso alguma das matrizes não exista, suas dimensões não sejam as mesmas ou a matriz resultante já exista

Dividir uma matriz por outra (elemento a elemento)

Comando	DV <nome1> <nome2> <nome-resultado sem espaços >
Saída	//Imprime matriz resultante (mesmo formato do comando IM) ERRO //Caso alguma das matrizes não exista, suas dimensões não sejam as mesmas, a segunda matriz contenha algum elemento igual a 0 ou a matriz resultante já exista

Multiplicar uma matriz por outra

Comando	MM <nome1> <nome2> <nome-resultado sem espaços >
Saída	//Imprime matriz resultante (mesmo formato do comando IM) ERRO //Caso alguma das matrizes não exista, suas dimensões estejam incorretas ou a matriz resultante já exista

Multiplicar uma matriz por outra (elemento a elemento)

Comando	ME <nome1> <nome2> <nome-resultado sem espaços >
Saída	//Imprime matriz resultante (mesmo formato do comando IM) ERRO //Caso alguma das matrizes não exista, suas dimensões não sejam as mesmas ou a matriz resultante já exista

Finaliza a execução

Comando	FE
Saída	//Finaliza a execução do programa

3. Observações importantes

- Os trabalhos deverão ser feitos em **grupos de no máximo 4 alunos**;
- O programa deverá respeitar exatamente os padrões de entrada e saída, pois a correção será automática;
- O projeto deve ser implementado em linguagem C e em ambiente Linux;
- Deve ser criado um Makefile para a compilação do seu programa;
- Cada membro do grupo deverá postar em seu escaninho no AVA os arquivos:
  - matriz.c e matriz.h;
  - lista.c e lista.h;
  - main.c e Makefile.
- Qualquer tentativa de plágio será punida com a nota **-Nmax** para todos os integrantes do grupo.