

Lab 0: Representação de Matriz

Prof. Waldemar Celes
Departamento de Informática, PUC-Rio

11 de Agosto de 2021

A correção dos trabalhos será feita automaticamente por programas de teste. Sigam as especificações nos seus detalhes!

O objetivo deste laboratório é criar funções para representação e manipulação de vetores e matrizes dinâmicas. A matriz será representada por um vetor de ponteiros, onde cada elemento aponta para o vetor linha. Pede-se:

1. Implemente uma função que crie dinamicamente um vetor de dimensão n , onde n representa o número de elementos do vetor:

```
double* criavet (int n);
```

2. Implemente uma função que libere a memória de um vetor previamente criado.

```
void liberavet (double* v);
```

3. Implemente uma função que calcule e retorne o valor do produto escalar entre dois vetores de dimensão n .

$$v \cdot w = \sum_0^{n-1} v_i w_i$$

```
double escalar (int n, const double* v, const double* w);
```

4. Implemente uma função que calcule a norma-2 de um vetor de dimensão n . Sabe-se que a norma-2 de um vetor é dado por:

$$\|v\|_2 = \sqrt{\sum_0^{n-1} v_i^2}$$

```
double norma2 (int n, const double* v);
```

5. Implemente uma função que calcule a multiplicação de um vetor v por um valor escalar s , preenchendo o vetor w com o resultado:

$$w_i = s v_i$$

```
void multvs (int n, const double* v, double s, double* w);
```

6. Implemente uma função que crie dinamicamente uma matriz de dimensão $m \times n$, onde m representa o número de linhas e n representa o número de colunas:

```
double** criamat (int m, int n);
```

7. Implemente uma função que crie dinamicamente uma matriz triangular inferior de dimensão $n \times n$. Nessa matriz, a primeira linha tem apenas 1 elemento, a segunda tem 2, a terceira tem 3, e assim por diante.

```
double** criamatri (int n);
```

8. Implemente uma função que libere a memória de uma matriz previamente criada. A função deve funcionar tanto para matriz cheia como para matriz triangular. A função recebe o número de linhas m da matriz:

```
void liberamat (int m, double** A);
```

9. Implemente uma função que preencha a transposta de uma dada matriz. A função recebe as dimensões $m \times n$ da matriz original, a matriz original A e a matriz transposta a ser preenchida T (com dimensão $n \times m$):

$$T_{ji} = A_{ij}$$

```
void transposta (int m, int n, const double** A, double** T);
```

10. Implemente uma função que receba uma matriz e um vetor, e preencha um outro vetor com o resultado da multiplicação da matriz pelo vetor. A função recebe a dimensão $m \times n$ da matriz e assume que o primeiro vetor v tem dimensão n e o vetor resultado w tem dimensão m :

$$w_i = \sum_{j=0}^{n-1} A_{ij} v_j$$

```
void multmv (int m, int n, const double** A, const double* v, double* w);
```

11. Implemente uma função que calcule a multiplicação entre duas matrizes: $C = AB$. A função recebe as dimensões m , n e q , e as matrizes $A_{m \times n}$, $B_{n \times q}$ e $C_{m \times q}$, preenchendo C :

$$C_{ik} = \sum_{j=0}^{n-1} A_{ij} B_{jk}$$

```
void multmm (int m, int n, int q, const double** A, const double** B, double** C);
```

Agrupe os protótipos das funções em um módulo “matriz.h” e as implementações em um módulo “matriz.c”. Escreva um outro módulo “main.c” para testar sua implementação.

Entrega: O código fonte deste trabalho (isto é, os arquivos “matriz.h”, “matriz.c” e “main.c”) devem ser enviados via página da disciplina no EAD.