Universidade Federal de São João del Rei
Departamento de Ciência da Computação
Curso de Ciência da Computação

Roteiro 7

Rodrigo José Zonzin
212050002

1

```c
#include <stdio.h>
#include <stdlib.h>

#include "matrizEstatica.h"



// Inicializa uma matriz com linhas e colunas especificadas
Matriz* criaMatriz(int linhas, int colunas){
    Matriz *matriz = (Matriz*) malloc(sizeof(Matriz));

    matriz->linhas = linhas;
    matriz->colunas = colunas;
    for(int i = 0; i < linhas; i++){
        for(int j = 0; j < colunas; j++){
            matriz->data[i][j] = 0; // Pode ser outro valor padr o se preferir
        }
    }

    return matriz;
}

// Define o valor de um elemento da matriz na posi  o (linha, coluna)
void definirElemento(Matriz *matriz, int linha, int coluna, int valor){
    if(linha >= 0 && linha < matriz->linhas && coluna >= 0 && coluna <
        matriz->colunas){
        matriz->data[linha][coluna] = valor;
    } else{
        printf("Posi  o inv lida\n");
    }
}

// Obt m o valor de um elemento da matriz na posi  o (linha, coluna)
int obterElemento(Matriz *matriz, int linha, int coluna){
    if(linha >= 0 && linha < matriz->linhas && coluna >= 0 && coluna <
        matriz->colunas){
        return matriz->data[linha][coluna];
    } else{
        printf("Posi  o inv lida\n");
        return 0; // Pode retornar outro valor padr o se preferir
    }
```

```c
}

// Imprime a matriz
void imprimirMatriz(Matriz *matriz){
    for(int i = 0; i < matriz->linhas; i++){
        for(int j = 0; j < matriz->colunas; j++){
            printf("%d ", matriz->data[i][j]);
        }
        printf("\n");
    }
}

Matriz* transposta(Matriz* original){
    int linhas = original->colunas;
    int colunas = original->linhas;

    Matriz* resultado = criaMatriz(linhas, colunas);

    for(int i = 0; i < original->linhas; i++) {
        for(int j = 0; j < original->colunas; j++) {
            resultado->data[j][i] = original->data[i][j];
        }
    }

    return resultado;
}

int main(){
    Matriz *minhaMatriz = criaMatriz(3, 3);


    definirElemento(minhaMatriz, 0, 1, 1);
    definirElemento(minhaMatriz, 1, 1, 2);
    definirElemento(minhaMatriz, 2, 2, 3);

    imprimirMatriz(minhaMatriz);

    int valor = obterElemento(minhaMatriz, 1, 1);
    printf("Valor na posi  o (1, 1): %d\n", valor);

    Matriz *minhaTransposta = transposta(minhaMatriz);
    imprimirMatriz(minhaTransposta);

    return 0;
}
```

../ex11/matrizEstatica.c

```c
#define _MATRIZESTATICA_H

#define MAX_TAM 100

struct matriz{
    int data[MAX_TAM][MAX_TAM];
    int linhas;
    int colunas;
};

```

```
11 typedef struct matriz Matriz;
```

../ex11/matrizEstatica.h

2

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  struct matriz {
5      int **data;
6      int linhas;
7      int colunas;
8  };
9
10 typedef struct matriz Matriz;
11
12 Matriz *criaMatriz(int linhas, int colunas) {
13     Matriz *mat = (Matriz *)malloc(sizeof(Matriz));
14     if (mat == NULL) {
15         perror("Erro ao alocar a matriz");
16         exit(1);
17     }
18
19     mat->linhas = linhas;
20     mat->colunas = colunas;
21
22     mat->data = (int **)malloc(linhas * sizeof(int *));
23     if (mat->data == NULL) {
24         perror("Erro ao alocar as linhas da matriz");
25         exit(1);
26     }
27
28     for (int i = 0; i < linhas; i++) {
29         mat->data[i] = (int *)malloc(colunas * sizeof(int));
30         if (mat->data[i] == NULL) {
31             perror("Erro ao alocar as colunas da matriz");
32             exit(1);
33         }
34     }
35
36     return mat;
37 }
38
39 void preencherMatriz(Matriz *mat, int *valores) {
40     for (int i = 0; i < mat->linhas; i++) {
41         for (int j = 0; j < mat->colunas; j++) {
42             mat->data[i][j] = valores[i * mat->colunas + j];
43         }
44     }
45 }
46
47 void imprimirMatriz(Matriz *mat) {
48     for (int i = 0; i < mat->linhas; i++) {
49         for (int j = 0; j < mat->colunas; j++) {
50             printf("%d ", mat->data[i][j]);
51         }
52         printf("\n");
53     }
```

```c
}

void liberarMatriz(Matriz *mat) {
    for (int i = 0; i < mat->linhas; i++) {
        free(mat->data[i]);
    }
    free(mat->data);
    free(mat);
}

Matriz* transposta(Matriz* original){
    int linhas = original->colunas;
    int colunas = original->linhas;

    Matriz* resultado = criaMatriz(linhas, colunas);

    for(int i = 0; i < original->linhas; i++) {
        for(int j = 0; j < original->colunas; j++) {
            resultado->data[j][i] = original->data[i][j];
        }
    }

    return resultado;
}

int main() {
    int valores[] = {1, 2, 3, 4, 5, 6};
    int linhas = 2;
    int colunas = 3;

    Matriz *minhaMatriz = criaMatriz(linhas, colunas);
    preencherMatriz(minhaMatriz, valores);

    printf("Matriz:\n");
    imprimirMatriz(minhaMatriz);

    Matriz *minhaTransposta = transposta(minhaMatriz);
    imprimirMatriz(minhaTransposta);

    liberarMatriz(minhaMatriz);
    liberarMatriz(minhaTransposta);

    return 0;
}
```

../ex12/matriz.c

3

```c
#include <stdio.h>
#include <stdlib.h>

// Defini  o da estrutura da Matriz de Faixa (Tridiagonal)
typedef struct {
    int n;          // Dimens o da matriz
    int* diag;      // Elementos da diagonal principal
    int* sup;       // Elementos da superdiagonal (acima da diagonal principal)
    int* sub;       // Elementos da subdiagonal (abaixo da diagonal principal)
} TridiagonalMatrix;
```

```c

// Fun   o para inicializar uma Matriz de Faixa
TridiagonalMatrix* createTridiagonalMatrix(int n) {
    TridiagonalMatrix* mat = (TridiagonalMatrix*)malloc(sizeof(TridiagonalMatrix));
    if (mat == NULL) {
        perror("Erro ao alocar mem ria para a matriz");
        exit(1);
    }

    mat->n = n;
    mat->diag = (int*)malloc(n * sizeof(int));
    mat->sup = (int*)malloc((n - 1) * sizeof(int));
    mat->sub = (int*)malloc((n - 1) * sizeof(int));

    if (mat->diag == NULL || mat->sup == NULL || mat->sub == NULL) {
        perror("Erro ao alocar mem ria para os elementos da matriz");
        exit(1);
    }

    return mat;
}

// Fun   o para definir um elemento na matriz
void setElement(TridiagonalMatrix* mat, int i, int j, int value) {
    if (i < 0 || i >= mat->n || j < 0 || j >= mat->n) {
        fprintf(stderr, " ndices  fora dos limites da matriz\n");
        exit(1);
    }

    if (i == j) {
        mat->diag[i] = value;
    } else if (i == j - 1) {
        mat->sup[i] = value;
    } else if (i == j + 1) {
        mat->sub[j] = value;
    } else {
        fprintf(stderr, "Elemento fora da banda tridiagonal\n");
        exit(1);
    }
}

// Fun   o para obter um elemento da matriz
int getElement(const TridiagonalMatrix* mat, int i, int j) {
    if (i < 0 || i >= mat->n || j < 0 || j >= mat->n) {
        fprintf(stderr, " ndices  fora dos limites da matriz\n");
        exit(1);
    }

    if (i == j) {
        return mat->diag[i];
    } else if (i == j - 1) {
        return mat->sup[i];
    } else if (i == j + 1) {
        return mat->sub[j];
    } else {
        return 0;  // Elementos fora da banda tridiagonal s o zeros
    }
}

```

```c
void imprimeVetoresMatrix(TridiagonalMatrix* mat){
    printf("Diagonal: ");
    for (int i = 0; i < mat->n; i++) {
        printf("%d ", mat->diag[i]);
    }
    printf("\n");

    printf("Superdiagonal: ");
    for (int i = 0; i < mat->n - 1; i++) {
        printf("%d ", mat->sup[i]);
    }
    printf("\n");

    printf("Subdiagonal: ");
    for (int i = 0; i < mat->n - 1; i++) {
        printf("%d ", mat->sub[i]);
    }
    printf("\n");
}

// Fun   o para liberar a mem ria da matriz
void freeTridiagonalMatrix(TridiagonalMatrix* mat) {
    free(mat->diag);
    free(mat->sup);
    free(mat->sub);
    free(mat);
}

int main() {
    int n = 5;
    TridiagonalMatrix* mat = createTridiagonalMatrix(n);

    // Preencher a matriz com alguns valores de exemplo
    for (int i = 0; i < n; i++) {
        setElement(mat, i, i, 2);
    }
    for (int i = 0; i < n - 1; i++) {
        setElement(mat, i, i + 1, 1);
        setElement(mat, i + 1, i, 1);
    }

    imprimeVetoresMatrix(mat);

    // Liberar a mem ria da matriz
    freeTridiagonalMatrix(mat);

    return 0;
}
```

../ex13/matrizFaixa.c

4

Em branco.