Atividade 8 - Filas e Matrizes Lineares (T2)

23 de outubro de 2025

Problema A

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

typedef struct No {
    struct No *anterior;
    struct No *proximo;
    int valor;
} No;

typedef struct Fila {
    struct No *inicio;
    struct No *fim;
    int tamanho;
} Fila;

Fila* criaFila(void){
    Fila *fila = (Fila*) malloc (sizeof(Fila));

    if(!fila){
        printf("Erro! Nao foi possivel alocar memoria para a fila.\n");
        return NULL;
    }

    fila->inicio = NULL;
    fila->fim = NULL;
    fila->tamanho = 0;

    return fila;
}

int filaExiste(Fila *fila){
    return fila != NULL;
}

int filaVazia(Fila *fila){
    if(!filaExiste(fila)){
        printf("Erro! A fila nao existe.\n");
        return 1;
    }
    return fila->tamanho == 0;
}

```

```c
void insereFila(Fila *fila, int valor){
    if(!filaExiste(fila)){
        printf("Erro! A fila nao existe.\n");
        return;
    }

    No *no = (No*) malloc (sizeof(No));

    if(!no){
        printf("Erro! Nao foi possivel alocar o no na memoria.\n");
        return;
    }

    no->valor = valor;
    no->proximo = NULL;

    if(filaVazia(fila)){
        fila->inicio = no;
        no->anterior = NULL;
    } else if (fila->tamanho > 1){
        no->anterior = fila->fim;
        fila->fim->proximo = no;
    } else {
        no->anterior = fila->inicio;
        fila->inicio->proximo = no;
    }

    fila->fim = no;
    fila->tamanho += 1;
}

int removeFila(Fila* fila){
    if(filaVazia(fila)){
        printf("Erro! A fila esta vazia.\n");
        return -1;
    }

    int valor = fila->inicio->valor;

    if(fila->tamanho > 1){
        fila->inicio = fila->inicio->proximo;
        free(fila->inicio->anterior);
        fila->inicio->anterior = NULL;
    } else {
        free(fila->inicio);
        fila->inicio = NULL;
        fila->fim = NULL;
    }

    fila->tamanho -= 1;
    return valor;
}

void destroiFila(Fila *fila){
    if(!filaExiste(fila)) return;

    while(fila->tamanho){
        removeFila(fila);
```

```
101     }
102
103     free(fila);
104 }
105
106 int main(){
107     int n;
108
109     Fila *fila = criaFila();
110
111     while(1){
112         scanf("%d", &n);
113         if(n == 0) break;
114
115         for(int i = 1; i <= n; ++i){
116             insereFila(fila, i);
117         }
118
119         int rounds = n - 1;
120         int* discarded_cards = (int*) malloc (sizeof(int) * rounds);
121         int j = 0;
122         while(rounds--){
123             discarded_cards[j++] = removeFila(fila);
124             int mover = removeFila(fila);
125             insereFila(fila, mover);
126         }
127
128         int remaining_card = removeFila(fila);
129
130         printf("Discarded cards:");
131         for(int i = 0; i < j; ++i){
132             if(i == 0){
133                 printf(" %d", discarded_cards[i]);
134             } else {
135                 printf(", %d", discarded_cards[i]);
136             }
137         }
138         printf("\n");
139
140         printf("Remaining card: %d\n", remaining_card);
141
142         free(discarded_cards);
143     }
144
145     destroiFila(fila);
146
147     return 0;
148 }
```

Problema B

```c
// Funcao para atualizar um elemento na posicao (linha, coluna)
void atualizaElemento(Matriz *m, int linha, int coluna, int valor) {
    m->matriz[m->num_colunas*linha + coluna] = valor;
}

// Funcao para recuperar um elemento na posicao (linha, coluna)
int recuperaElemento(Matriz *m, int linha, int coluna) {
    return m->matriz[m->num_colunas*linha + coluna]
}
```

Problema C

```c
// Funcao para atualizar um elemento na posicao (linha, coluna)
void atualizaElemento(Matriz *m, int linha, int coluna, int valor) {
    // Logica para atualizar o elemento da matriz.
  // Lembre-se que a matriz esta armazenada como um vetor de
  // acordo com a figura do enunciado.
  // O endereco inicial da matriz e dado por m->matriz!
    int idx = (m->dimensao * (m->dimensao + 1) / 2) - ((m->dimensao -
       linha) * (m->dimensao - linha + 1) / 2) + (coluna - linha);
    m->matriz[idx] = valor;
}

// Funcao para recuperar um elemento na posicao (linha, coluna)
int recuperaElemento(Matriz *m, int linha, int coluna) {
    if (linha > coluna) {
        return 0;
    }
  // Logica para atualizar o elemento da matriz.
  // Lembre-se que a matriz esta armazenada como um vetor de
  // acordo com a figura do enunciado.
  // O endereco inicial da matriz e dado por m->matriz!
    int idx = (m->dimensao * (m->dimensao + 1) / 2) - ((m->dimensao -
       linha) * (m->dimensao - linha + 1) / 2) + (coluna - linha);
    return m->matriz[idx];
}
```