

Roteiro 5 - Gabriel Souza de Oliveira - 222050042

1.1 -

```
#ifndef FILA_H
#define FILA_H
#include <stdio.h>
#include <stdlib.h>

#define MAX 100
typedef struct{
    int qtd, inicio, fim;
    int dados[MAX];
}Fila;

Fila* criarFila(){
    Fila* fila;
    fila = (Fila*)malloc(sizeof(Fila));
    if (fila != NULL){
        fila->qtd = 0;
        fila->inicio = 0;
        fila->fim = 0;
    }
    return fila;
}

void destroiFila(Fila *fila){
    if (fila != NULL)
        free(fila);
}

int tamanhoFila (Fila *fila){
    if (fila == NULL)
        return -1;
    return fila->qtd;
}

int filaCheia(Fila *fila){
    if (fila == NULL)
        return -1;
    return (fila->qtd == MAX);
}
```

```
int filaVazia(Fila *fila){
    if (fila == NULL)
        return -1;
    return (fila->qtd == 0);
}

int enfileirar (Fila *fila, int elemento){
    if (fila == NULL)
        return 0;
    if (filaCheia(fila))
        return 0;
    fila->dados[fila->fim] = elemento;
    fila->fim = (fila->fim+1)%MAX;
    fila->qtd++;
    return 1;
}

int desenfileirar(Fila *fila){
    if (fila == NULL)
        return 0;
    if (filaVazia(fila))
        return 0;
    fila->inicio = (fila->inicio+1)%MAX;
    fila->qtd--;
    return 1;
}

int verInicio(Fila* fila, int* elemento){
    if (fila == NULL)
        return 0;
    if (filaVazia(fila))
        return 0;
    *elemento = fila->dados[fila->inicio];
    return 1;
}

void imprime(Fila* fila){
    if (fila == NULL)
        return;
    if (filaVazia(fila)){
        printf ("Fila vazia!\n");
        return;
    }
}
```

```

    int i = fila->inicio;
    printf ("Elementos: \n");
    do{
        printf ("%d ", fila->dados[i]);
        i = (i+1)%MAX;
    } while (i != fila->fim);
    printf ("\n");
}

#endif //FILA_H

```

```

#include <stdio.h>
#include <stdlib.h>
#include "fila_sequencial_estatica.h"

int main(){
    int option, elemento;
    Fila* fila = NULL;

    do{
        printf ("1- Criar fila.\n");
        printf ("2- Enfileirar um item.\n");
        printf ("3- Ver o inicio da fila.\n");
        printf ("4- Desenfileirar um item.\n");
        printf ("5- Imprimir a fila.\n");
        printf ("6- Destruir a fila.\n");
        printf ("7- Sair.\n");

        printf ("Opcao: ");
        scanf ("%d", &option);

        switch (option){
            case 1:
                if (fila != NULL){
                    destroiFila(fila);
                }
                fila = criarFila();
                break;
            case 2:
                printf ("Digite o numero que deseja adicionar a fila:
\n");

                scanf ("%d", &elemento);
                enfileirar(fila, elemento);

```

```

        break;
    case 3:
        if (verInicio(fila, &elemento)){
            printf ("Inicio da Fila: %d\n", elemento);
        } else {
            printf ("Nao foi possivel ver o inicio");
        }
        break;
    case 4:
        desenfileirar(fila);
        break;
    case 5:
        imprime(fila);
        break;
    case 6:
        destroiFila(fila);
        break;
    case 7:
        if (fila != NULL){
            destroiFila(fila);
        }
        printf ("Saindo do menu!\n");
        break;
    default:
        printf ("Opcao invalida! Tente de novo.\n");
    }
} while (option != 7);
return 0;
}

```

Saída:

```

PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\output> cd 'c:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\outp
PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\output> & .\exercício1-1.exe'
1- Criar fila.
2- Enfileirar um item.
3- Ver o início da fila.
4- Desenfileirar um item.
5- Imprimir a fila.
6- Destruir a fila.
7- Sair.
Opcao: 1
Opcao: 2
Digite o numero que deseja adicionar a fila:
15
Opcao: 2
Digite o numero que deseja adicionar a fila:
10
Opcao: 3
Início da Fila: 15
Opcao: 5
Elementos:
15 10
Opcao: 4
1- Criar fila.
2- Enfileirar um item.
3- Ver o início da fila.
4- Desenfileirar um item.
5- Imprimir a fila.
6- Destruir a fila.
7- Sair.
Opcao: 5
Elementos:
10

```

1.2 -

```

#ifndef FSE_H
#define FSE_H
#include <stdio.h>
#include <stdlib.h>

typedef struct NO{
    int info;
    struct NO* prox;
}NO;

typedef struct{
    int qtd;
    struct NO* inicio;
    struct NO* fim;
}Fila;

Fila* criaFila(){
    Fila* fila;
    fila = (Fila*)malloc(sizeof(Fila));
    if (fila != NULL){
        fila->qtd = 0;
        fila->inicio = NULL;
        fila->fim = NULL;
    }
    return fila;
}

```

```

}

void destroiFila(Fila* fila) {
    if (fila != NULL) {
        NO* auxiliar;
        while (fila->inicio != NULL) {
            auxiliar = fila->inicio;
            fila->inicio = fila->inicio->prox;
            free(auxiliar);
        }
        free(fila);
    }
}

NO* criaNo() {
    NO* novo = (NO*)malloc(sizeof(NO));
    novo->info = 0;
    novo->prox = NULL;
    return novo;
}

void destroiNo(NO* no) {
    free(no);
}

int tamanhoFila (Fila *fila) {
    if (fila == NULL)
        return 0;
    return fila->qtd;
}

int estaVazia(Fila* fila) {
    if (fila == NULL)
        return 0;
    return fila->inicio == NULL;
}

int enfileirar(Fila* fila, int elemento) {
    if (fila == NULL)
        return 0;
    NO* auxiliar = criaNo();
    auxiliar->info = elemento;

```

```

        if (estaVazia(fila)){
            fila->inicio = auxiliar;
            fila->fim = auxiliar;
        } else{
            fila->fim->prox = auxiliar; // Se a fila não estiver vazia,
            // Adicione após o último elemento;
            fila->fim = auxiliar; // Atualize o último elemento
        }
        fila->qtd++;
        return 1;
    }
}

int desenfileirar(Fila* fila){
    if (fila == NULL)
        return 0;
    if (estaVazia(fila))
        return 0;
    NO* auxiliar = fila->inicio;
    fila->inicio = fila->inicio->prox;

    destroiNo(auxiliar);
    fila->qtd--;
    return 1;
}

int verInicio(Fila* fila, int* elemento){
    if (fila == NULL)
        return 0;
    if (estaVazia(fila))
        return 0;
    *elemento = fila->inicio->info;
    return 1;
}

void imprime(Fila* fila){
    if (fila == NULL)
        return;
    if (estaVazia(fila)){
        printf ("Fila Vazia!\n");
        return;
    }

    printf ("Elementos:\n");

```

```

    NO* auxiliar = fila->inicio;
    do {
        printf ("%d ", auxiliar->info);
        auxiliar = auxiliar->prox;
    } while (auxiliar != NULL);
    printf ("\n");
}

#endif //FSE_H

```

```

#include <stdio.h>
#include <stdlib.h>
#include "fila_simplesmente_encadeada.h"

int main(){
    int option, elemento;
    Fila* fila = NULL;
    do{
        printf ("1- Criar fila.\n");
        printf ("2- Enfileirar um item.\n");
        printf ("3- Ver o inicio da fila.\n");
        printf ("4- Desenfileirar um item.\n");
        printf ("5- Imprimir a fila.\n");
        printf ("6- Destruir a fila.\n");
        printf ("7- Sair.\n");

        printf ("Opcao: ");
        scanf ("%d", &option);

        switch (option){
            case 1:
                if (fila != NULL){
                    destroiFila(fila);
                }
                fila = criaFila();
                break;
            case 2:
                printf ("Digite o numero que deseja adicionar a fila:\n");

                scanf ("%d", &elemento);
                enfileirar(fila, elemento);
                break;
            case 3:

```



```

        if (verInicio(fila, &elemento)){
            printf ("Inicio da Fila: %d\n", elemento);
        } else {
            printf ("Nao foi possivel ver o inicio");
        }
        break;
    case 4:
        desenfileirar(fila);
        break;
    case 5:
        imprime(fila);
        break;
    case 6:
        destroiFila(fila);
        break;
    case 7:
        if (fila != NULL){
            printf ("Destruindo fila primeiro!\n");
            destroiFila(fila);
            printf ("Saindo do menu!\n");
            break;
        }
        printf ("Saindo do menu!\n");
        break;
    default:
        printf ("Opcao invalida! Tente de novo.\n");
    }
} while (option != 7);
return 0;
}

```

Saída:

```

PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\output> cd 'c:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\output'
PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\output> .\'exercicio1-2.exe'
1- Criar fila.
2- Enfileirar um item.
3- Ver o início da fila.
4- Desenfileirar um item.
5- Imprimir a fila.
6- Destruir a fila.
7- Sair.
Opcao: 1
15
Opcao: 2
Digite o numero que deseja adicionar a fila:
10
Opcao: 3
Início da Fila: 15
Opcao: 5
Elementos:
15 10
Opcao: 4
1- Criar fila.
2- Enfileirar um item.
3- Ver o início da fila.
4- Desenfileirar um item.
5- Imprimir a fila.
6- Destruir a fila.
7- Sair.
Opcao: 5
Elementos:
10

```

2.1 -

```

#ifndef PILHA_H
#define PILHA_H

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

#define MAX 100
typedef struct{
    int topo;
    int dados[MAX];
}Pilha;

Pilha* criaPilha(){
    Pilha* pilha;
    pilha = (Pilha*)malloc(sizeof(Pilha));
    if (pilha != NULL)
        pilha->topo = 0;
    return pilha;
}

void destroiPilha (Pilha *pilha){
    if(pilha != NULL )
        free (pilha);
}

```

```
int tamanhoPilha ( Pilha *pilha){
    if(pilha == NULL)
        return -1;
    return pilha -> topo ;
}

int estaCheia ( Pilha *pilha){
    if(pilha == NULL )
        return -1;
    return (pilha -> topo == MAX );
}

int estaVazia ( Pilha *pilha){
    if(pilha == NULL )
        return -1;
    return (pilha -> topo == 0);
}

int empilhar(Pilha* pilha, int elemento){
    if (pilha == NULL)
        return 0;
    if (estaCheia(pilha))
        return 0;
    pilha->dados[pilha->topo] = elemento;
    pilha->topo++;
    return 1;
}

int desempilhar (Pilha* pilha){
    if (pilha == NULL)
        return 0;
    if (estaVazia(pilha))
        return 0;
    pilha->topo--;
    return 1;
}

int verTopo (Pilha* pilha, int* elemento){
    if (pilha == NULL)
        return 0;
    if (estaVazia(pilha))
        return 0;
    *elemento = pilha->dados[pilha->topo-1];
}
```

```

        return 1;
    }

void imprime(Pilha *pilha){
    if (pilha == NULL)
        return;
    if (estaVazia(pilha)){
        printf ("Pilha Vazia!\n");
        return;
    }
    printf ("Elementos: \n");
    int i;
    for (i = pilha->topo-1; i >= 0; i--){
        printf ("%d ", pilha->dados[i]);
    }
    printf ("\n");
}
#endif //PILHA_H

```

```

#include <stdio.h>
#include "pilha_sequencial_estatica.h"

int main(){
    int option, elemento;
    Pilha* pilha = NULL;

    do{
        printf ("1- Criar pilha.\n");
        printf ("2- Empilhar um item.\n");
        printf ("3- Ver o topo da pilha.\n");
        printf ("4- Desempilhar um item.\n");
        printf ("5- Imprimir a pilha.\n");
        printf ("6- Destruir a pilha.\n");
        printf ("7- Sair.\n");

        printf ("Opcao: ");
        scanf ("%d", &option);

        switch (option){
            case 1:
                if (pilha != NULL){
                    destroiPilha(pilha);
                    printf ("Pilha resetada!\n");
                }
            case 2:
                elemento = 0;
                while (elemento != -1){
                    elemento = getint();
                    if (elemento != -1)
                        empilha(pilha, elemento);
                }
            case 3:
                if (estaVazia(pilha))
                    printf ("Pilha vazia!\n");
                else
                    printf ("Topo da pilha: %d\n", pilha->dados[pilha->topo]);
            case 4:
                if (estaVazia(pilha))
                    printf ("Pilha vazia!\n");
                else
                    desempilha(pilha);
            case 5:
                imprime(pilha);
            case 6:
                destroiPilha(pilha);
            case 7:
                break;
        }
    } while (option != 7);
}

```

```

        }
        pilha = criaPilha();
        break;
    case 2:
        printf ("Digite o numero que deseja adicionar a pilha:
\n");

        scanf ("%d", &elemento);
        if (empilhar(pilha, elemento)){
            printf ("Empilhou (%d)\n", elemento);
        } else{
            printf ("Nao foi possivel empilhar!\n");
        }
        break;
    case 3:
        if (verTopo(pilha, &elemento)){
            printf ("Topo da Pilha = %d\n", elemento);
        } else{
            printf ("Nao foi possivel ver o topo.\n");
        }
        break;
    case 4:
        desempilhar(pilha);
        break;
    case 5:
        imprime(pilha);
        break;
    case 6:
        if (pilha != NULL)
            destroiPilha(pilha);
        break;
    case 7:
        if (pilha != NULL)
            destroiPilha(pilha);
        printf ("Saindo do menu!\n");
        break;
    default:
        printf ("Opcao invalida! Tente de novo.\n");
    }
} while (option != 7);
return 0;
}

```

Saída:

```

PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5> cd 'c:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\output'
PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\output> & .\exercicio2-1.exe
1- Criar pilha.
2- Empilhar um item.
3- Ver o topo da pilha.
4- Desempilhar um item.
5- Imprimir a pilha.
6- Destruir a pilha.
7- Sair.
Opcao: 1
Opcao: 2
Digite o numero que deseja adicionar a pilha:
15
Empilhou (15)
Opcao: 2
Digite o numero que deseja adicionar a pilha:
10
Empilhou (10)
Opcao: 5
Elementos:
10 15
Opcao: 3
Topo da Pilha = 10
Opcao: 4
1- Criar pilha.
2- Empilhar um item.
3- Ver o topo da pilha.
4- Desempilhar um item.
5- Imprimir a pilha.
6- Destruir a pilha.
7- Sair.
Opcao: 5
Elementos:
15
Opcao: 5
Elementos:
15

```

2.2 -

```

#ifndef PSE_H
#define PSE_H
#include <stdio.h>
#include <stdlib.h>

typedef struct NO{
    int info;
    struct NO* prox;
}NO;

typedef struct{
    int qtd;
    struct NO* topo;
}Pilha;

Pilha* criaPilha(){
    Pilha* pilha = (Pilha*)malloc(sizeof(Pilha));
    if (pilha != NULL){
        pilha->qtd = 0;
        pilha->topo = NULL;
    }
    return pilha;
}

```

```

void destroiPilha(Pilha** pilha){
    if (*pilha != NULL){
        free (*pilha);
        *pilha = NULL;
    }
}

int tamanhoPilha(Pilha* pilha){
    if (pilha == NULL)
        return 0;
    return pilha->qtd;
}

int estaVazia(Pilha* pilha){
    if (pilha == NULL)
        return 0;
    return pilha->topo == NULL;
}

NO* criaNo(){
    NO* novo = (NO*)malloc(sizeof(NO));
    novo->info = 0;
    novo->prox = NULL;
    return novo;
}

int empilhar(Pilha* pilha, int elemento){
    if (pilha == NULL)
        return 0;
    NO* auxiliar = criaNo();
    auxiliar->info = elemento;
    auxiliar->prox = pilha->topo;
    pilha->topo = auxiliar;
    pilha->qtd++;
    return 1;
}

void destroiNO(NO* no){
    free(no);
}

int desempilhar(Pilha* pilha){

```

```

        if (pilha == NULL)
            return 0;
        if (estaVazia(pilha))
            return 0;
        NO* auxiliar = pilha->topo;
        pilha->topo = pilha->topo->prox;
        destroiNO(auxiliar);
        pilha->qtd--;
        return 1;
    }

int verTopo(Pilha* pilha, int* elemento){
    if (pilha == NULL)
        return 0;
    if (estaVazia(pilha))
        return 0;
    *elemento = pilha->topo->info;
    return 1;
}

void imprime(Pilha* pilha){
    if (pilha == NULL)
        return;
    if (estaVazia(pilha)){
        printf ("Pilha Vazia!\n");
        return;
    }
    printf ("Elementos da Pilha:\n");
    NO* auxiliar = pilha->topo;
    do {
        printf ("%d ", auxiliar->info);
        auxiliar = auxiliar->prox;
    } while (auxiliar != NULL);
    printf ("\n");
}

#endif //PSE_H

#include <stdio.h>
#include "pilha_simplesmente_encadeada.h"

int main(){
    int option, elemento;
    Pilha* pilha = NULL;

```



```
do{
    printf ("1- Criar pilha.\n");
    printf ("2- Empilhar um item.\n");
    printf ("3- Ver o topo da pilha.\n");
    printf ("4- Desempilhar um item.\n");
    printf ("5- Imprimir a pilha.\n");
    printf ("6- Destruir a pilha.\n");
    printf ("7- Sair.\n");

    printf ("Opcao: ");
    scanf ("%d", &option);

    switch (option){
        case 1:
            if (pilha != NULL){
                destroiPilha(&pilha);
                printf ("Pilha resetada!\n");
            }
            pilha = criaPilha();
            break;
        case 2:
            printf ("Digite o numero que deseja adicionar a pilha:
\n");

            scanf ("%d", &elemento);
            if (empilhar(pilha, elemento)){
                printf ("Empilhou (%d)\n", elemento);
            } else{
                printf ("Nao foi possivel empilhar!\n");
            }
            break;
        case 3:
            if (verTopo(pilha, &elemento)){
                printf ("Topo da Pilha = %d\n", elemento);
            } else{
                printf ("Nao foi possivel ver o topo.\n");
            }
            break;
        case 4:
            desempilhar(pilha);
            break;
        case 5:
            imprime(pilha);
            break;
```

```

        case 6:
            if (pilha != NULL)
                destroiPilha(&pilha);
            break;
        case 7:
            if (pilha != NULL)
                destroiPilha(&pilha);
            printf ("Saindo do menu!\n");
            break;
        default:
            printf ("Opcao invalida! Tente de novo.\n");
    }
} while (option != 7);
return 0;
}

```

Saída:

```

PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\output> cd 'c:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\output'
PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 5\output> & .\exercicio2-2.exe
1- Criar pilha.
2- Empilhar um item.
3- Ver o topo da pilha.
4- Desempilhar um item.
5- Imprimir a pilha.
6- Destruir a pilha.
7- Sair.
Opcao: 1
Opcao: 2
Digite o numero que deseja adicionar a pilha:
15
Empilhou (15)
Opcao: 2
Digite o numero que deseja adicionar a pilha:
10
Empilhou (10)
Opcao: 5
Elementos:
10 15
Opcao: 3
Topo da Pilha = 10
Opcao: 4
1- Criar pilha.
2- Empilhar um item.
3- Ver o topo da pilha.
4- Desempilhar um item.
5- Imprimir a pilha.
6- Destruir a pilha.
7- Sair.
Opcao: 5
Elementos:
15
Opcao: 5
Elementos:
15

```