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

Roteiro 5

Rodrigo José Zonzin
212050002

1 Fila

1.1

```c
#include "fila.h"

Fila* cria_fila(){
    Fila *f = (Fila*)malloc(sizeof(Fila*));
    f->elementos = (int*)malloc(sizeof(int)*50);
    f->n = 0;

    return f;
}

void enfilera(Fila *f, int a){
    f->elementos[f->n] = a;
    f->n = f->n + 1;
}

void desenfilera(Fila *f){
    for(int i = 0; i< (f->n)-1; i++){
        f->elementos[i] = f->elementos[i+1];
    }
    f->n = f->n - 1;
}

void imprime_fila(Fila *f){
    printf("[");
    for(int i = 0; i<(f->n)-1; i++){
        printf("%d ", f->elementos[i]);
    }
    printf("%d]\n", f->elementos[f->n -1]);
}

int inicio_fila(Fila *f){
    return f->elementos[0];
}

void destroiFila(Fila *f){
    free(f->elementos);
    free(f);
}

```

```c
void clearTerminal(){
    #ifdef _WIN32
        system("cls"); // Limpa o terminal no Windows
    #else
        system("clear"); // Limpa o terminal no Unix/Linux
    #endif
}
```

../ex1/fila.c

```c
#define _FILA_H

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


struct fila{
    int *elementos;
    int n;
};
typedef struct fila Fila;

Fila* cria_fila();
void enfilera(Fila*, int);
void desenfilera(Fila*);
void imprime_fila(Fila*);
void destroiFila(Fila*);
```

../ex1/fila.h

```c
#include "fila.h"

int main(){
    int chamada = 10;
    int elemento;
    Fila *f;

    while(chamada != 7){
        printf("\n\n1-Cria Fila\t2-Enfilera\t3-VerInicio\n");
        printf("4-Desenfilera\t5-Imprimir\t6-Destruir\n");
        scanf("%d", &chamada);


        if(chamada == 1){
            f = cria_fila();
            continue;
        }

        if(chamada == 2){
            printf("Qual elemento?\n");
            scanf("%d", &elemento);
            enfilera(f, elemento);
            continue;
        }

        if(chamada == 3){
            printf("%d\n", inicio_fila(f));
            continue;
        }

```

```
31          if(chamada == 4){
32              desenfilera(f);
33              continue;
34          }
35
36          if(chamada==5){
37              imprime_fila(f);
38              continue;
39          }
40
41          if(chamada == 6){
42              destroiFila(f);
43              break;
44          }
45      }
46      printf("E eh isso\n");
47      return 0;
48  }
```

../ex1/main.c

1.2

```
1  #include "fila.h"
2
3  Fila* cria_fila(){
4      Fila*f=(Fila*)malloc(sizeof(Fila));
5      f->qtd=0;
6      f->ini= NULL;
7      f->fim= NULL;
8      return f;
9  }
10
11 void enfilera(Fila*f, int a){
12     NO* novo=(NO*)malloc(sizeof(NO));
13
14     novo->info = a;
15     novo->prox = NULL;
16
17     if(f->qtd == 0){
18         f->ini = novo;
19         f->fim = novo;
20     }
21     else{
22         f->fim->prox=novo;
23         f->fim=novo;
24     }
25
26     f->qtd++;
27 }
28
29 void desenfilera(Fila*f){
30     if(f->qtd==0){
31         return;
32     }
33
34     NO* temp = f->ini;
35     f->ini=f->ini->prox;
36     free(temp);
37     f->qtd--;
```

```c
}

void imprime_fila(Fila*f){
    printf("[");
    NO*atual=f->ini;

    while(atual!=NULL){
        if(atual->prox == NULL){
            printf("%d]\n", atual->info);
            break;
        }
        printf("%d ",atual->info);
        atual=atual->prox;
    }
}

int inicio_fila(Fila*f){
    if(f->qtd==0){
        printf("Lista vazia.\n");
        return-1;
    }
    return f->ini->info;
}

void destroiFila(Fila*f){
    NO*atual=f->ini;
    while(atual!=NULL){
        NO*temp=atual;
        atual=atual->prox;
        free(temp);
    }
    free(f);
}
```

../ex2/fila.c

```c
#define _FILA_H

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

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

typedef struct fila{
    int qtd;
    struct NO* ini;
    struct NO* fim;
}Fila;

Fila* cria_fila();
void enfilera(Fila*, int);
void desenfilera(Fila*);
void imprime_fila(Fila*);
int inicio_fila(Fila*);
void destroiFila(Fila*);
```

../ex2/fila.h

```c
#include "fila.h"

int main(){
    Fila *f = cria_fila();

    enfilera(f, 00);
    enfilera(f, 01);
    enfilera(f, 10);
    enfilera(f, 11);

    imprime_fila(f);

    desenfilera(f);
    imprime_fila(f);



    return 0;
}
```

../ex2/main.c

2 Pilha

2.1

```c
/*
    Criar pilha;
    Empilhar um item;
    Ver o topo da pilha.
    Desempilhar um item;
    Imprimir a pilha;
    Destruir a pilha;
    Sair;
*/
#include "pilha.h"

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

Pilha* cria_pilha(int tamanho){
    Pilha* p = (Pilha*)malloc(sizeof(Pilha));
    p->elementos = (int*)malloc(sizeof(int) * tamanho);
    p->n = 0;
    return p;
}

void empilhar(Pilha* p, int valor){
    p->elementos[p->n] = valor;
    p->n = p->n +1;
}

int ver_topo(Pilha* p){
    if (p->n > 0){
        return p->elementos[p->n - 1];
    }
    else{
        printf("A pilha esta vazia.\n");
        return -1;
```

```
34      }
35  }
36
37  void desempilhar(Pilha* p){
38      if (p->n > 0) {
39          p->n--;
40      } else {
41          printf("A pilha esta vazia.\n");
42      }
43  }
44
45  void imprimir_pilha(Pilha* p) {
46      printf("[");
47      for (int i = 0; i < p->n - 1; i++) {
48          printf("%d, ", p->elementos[i]);
49      }
50
51      if (p->n > 0) {
52          printf("%d", p->elementos[p->n - 1]);
53      }
54
55      printf("]\n");
56  }
57
58  void destruir_pilha(Pilha* p) {
59      free(p->elementos);
60      free(p);
61  }
```

../ex3/pilha.c

```
1  #define _PILHA_H
2
3  struct pilha{
4      int *elementos;
5      int n;
6  };
7  typedef struct pilha Pilha;
```

../ex3/pilha.h

```
1  #include "pilha.h"
2
3  int main(){
4      int tamanho = 50;
5      Pilha* pilha = cria_pilha(tamanho);
6
7      empilhar(pilha, 10);
8      empilhar(pilha, 20);
9      empilhar(pilha, 30);
10     imprimir_pilha(pilha);
11
12     printf("Topo: %d\n", ver_topo(pilha));
13
14     desempilhar(pilha);
15     desempilhar(pilha);
16
17     imprimir_pilha(pilha);
18
19     destruir_pilha(pilha);
```

```
20
21      return 0;
22  }
```

../ex3/main.c

2.2

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #include "pilha.h"
5
6  Pilha* cria_pilha(){
7      Pilha* p = (Pilha*)malloc(sizeof(Pilha));
8      p->qtd = 0;
9      p->topo = NULL;
10     return p;
11 }
12
13 void empilhar(Pilha* p, int valor){
14     NO* novo = (NO*)malloc(sizeof(NO));
15     novo->info = valor;
16     novo->prox = p->topo;
17     p->topo = novo;
18     p->qtd++;
19 }
20
21 int ver_topo(Pilha* p){
22     if(p->qtd > 0){
23         return p->topo->info;
24     }
25     else{
26         printf("A pilha esta vazia.\n");
27         return -1;
28     }
29 }
30
31 void desempilhar(Pilha* p){
32     if (p->qtd > 0){
33         NO* temp = p->topo;
34         p->topo = p->topo->prox;
35         free(temp);
36         p->qtd--;
37     }
38     else{
39         printf("A pilha esta vazia.\n");
40     }
41 }
42
43 void imprimir_pilha(Pilha* p){
44     printf("Pilha: [");
45     NO* atual = p->topo;
46
47     while(atual != NULL){
48         printf("%d", atual->info);
49         if(atual->prox != NULL) {
50             printf(", ");
51         }
52         atual = atual->prox;
```

```
53      }
54      printf("]\n");
55  }
56
57  void destruir_pilha(Pilha* p){
58      while(p->topo != NULL){
59          NO* temp = p->topo;
60          p->topo = p->topo->prox;
61          free(temp);
62      }
63      free(p);
64  }
```

../ex4/pilha.c

```
1  #define _PILHA_H
2
3
4  typedef struct NO{
5      int info ;
6      struct NO* prox ;
7  }NO;
8
9  typedef struct{
10     int qtd;
11     struct NO* topo ;
12 }Pilha ;
```

../ex4/pilha.h

```
1  #include "pilha.h"
2
3  int main(){
4      Pilha* pilha = cria_pilha();
5
6      empilhar(pilha, 10);
7      empilhar(pilha, 20);
8      empilhar(pilha, 30);
9
10     printf("Topo da pilha: %d\n", ver_topo(pilha));
11
12     desempilhar(pilha);
13
14     imprimir_pilha(pilha);
15     destruir_pilha(pilha);
16
17     return 0;
18 }
```

../ex4/main.c

3 Código

https://github.com/RodrigoZonzin/labaeds2