Atividade avaliativa 3

Aluno: Vinícius Celestino de Medeiros

Link dos arquivos no GitHub:

https://github.com/vinimedeiros13/DataStructure/tree/main/MyCodes/Exerc%C3%ADcios/Avalia%C3%A7%C3%A3o_3

Questão 1: Optando em utilizar fila e pilha **Estática**

Arquivo **.h** com os protótipos das funções:

```c
#define MAX 5

typedef struct aluno{
 int matricula;
 char nome[30];
 float n1, n2;
}Aluno;

typedef struct fila{
 int inicio, final, quantidade;
 Aluno elementos[MAX];
}Fila;

typedef struct pilha{
 int quantidade, topo;
 struct aluno dados[MAX], elementos[MAX];
}Pilha;

/* Funções para fila */

int cria_fila(Fila *fi); /* Cria a fila */

int fila_cheia(Fila *fi); /* Verifica se a fila está cheia */

int fila_vazia(Fila *fi); /* Verifica se a fila está vazia */

int busca_fila(Fila *fi, Aluno *al); /* Consultar o primeiro elemento
da fila */
```

```c
int enfileirar(Fila *fi, Aluno *al); /* Inserir elemento na fila CESAR */

int desenfileirar(Fila *fi, Aluno *al); /* Remover elemento na fila CESAR */

void imprimir_fila(Fila *fi); /* Exibir conteúdo da fila CESAR */

void libera_fila(Fila *fi); /* Liberando a fila */

/* Funções para pilha */

int cria_pilha(Pilha *pi); /* Cria a pilha */

int pilha_cheia(Pilha *pi); /* Verificar se a pilha está cheia */

int pilha_vazia(Pilha *pi); /* Verifica se a pilha está vazia */

int busca_topo_pilha(Pilha *pi, Aluno *al); /* Consulta o elemento do topo */

int empilhar(Pilha *pi, Aluno *al); /* Inserir elemento no topo da pilha */

int desempilhar(Pilha *pi, Aluno *al); /* Remover elemento no topo da pilha */

void imprimir_pilha(Pilha *pi); /* Exibir elementos */

void libera_pilha(Pilha *pi); /* Libera a pilha */

int inverter_fila(Fila *fi); /* Função para inverter a fila */
```

Arquivo **.c** com a implementação das funções:

```c
#include <stdio.h>
#include <stdlib.h>
#include "FilaPilhaEst_1.h"

/* Implementação das funções */

int cria_fila(Fila *fi){ /* Criando um ponteiro e alocando uma
quantidade de memória pra ele */
 if(fi != NULL){
   fi -> inicio = 0; /* Colocando o valor de 0 para inicio */
   fi -> final = 0; /* Colocando o valor de 0 para final */
   fi -> quantidade = 0; /* Colocando o valor de 0 para quantidade */
 }
 return 1; /* Retornando o ponteiro alocado */
}

int fila_cheia(Fila *fi){ /* Verificar se a fila está cheia*/
 if(fi == NULL) return -1;
 if(fi -> quantidade == MAX) /* Compara a quantidade, se está cheia */
   return 1;
 else
   return 0;
}

int fila_vazia(Fila *fi){
 if(fi == NULL) return -1;
 if(fi -> quantidade == 0)
   return 1;
 else
   return 0;
}

int busca_fila(Fila *fi, Aluno *al){ /* Busca o primeiro elemento da
fila (Somente pode buscar pelo primeiro elemento)*/
 if((fi == NULL) || (fila_vazia(fi))) return 0;

 *al = fi -> elementos[fi -> inicio]; /* Recebe os dados da fila que
está no início */
   return 1;
}
```

```c
int enfileirar(Fila *fi, Aluno *al){
 if (fila_cheia( fi ) ) return 0;
  (fi -> quantidade)++;
 fi -> elementos[fi-> final] = *al;
 fi -> final = (fi->final + 1) % MAX;
 return 1;
}

int desenfileirar (Fila *fi, Aluno *al){
 *al = fi -> elementos[fi->inicio];

 (fi->quantidade)--;
 fi->inicio = (fi->inicio + 1) % MAX;
   return 1;
}

void imprimir_fila(Fila *fi){
 for(int i = 0, aux = fi->inicio; i < fi->quantidade; i++, aux
=(aux+1)%MAX){
   printf("Nome: %s\n", fi->elementos[aux].nome);
   printf("Matricula: %d\n", fi->elementos[aux].matricula);
   printf("Nota 1: %.2lf\n", fi->elementos[aux].n1);
   printf("Nota 2: %.2lf\n\n", fi->elementos[aux].n2);
 }
}

void libera_fila(Fila *fi){ /* Liberando a fila */
 free(fi->elementos);
 free(fi);
}

int cria_pilha(Pilha *pi){ /* Criando um ponteiro e alocando uma
quantidade de memória pra ele */
 if(pi != NULL){
   pi -> quantidade = 0; /* Colocando o valor de 0 para quantidade */
   pi -> topo = 0;
 }
 return 1; /* Retornando o ponteiro alocado */
}

int pilha_cheia(Pilha *pi){
 if(pi == NULL) return -1;
 if(pi -> quantidade == MAX)
   return 1;
 else
   return 0;
```

```
}

int pilha_vazia(Pilha *pi){
 if(pi == NULL) return -1;
 return (pi -> quantidade == 0);
}

int busca_topo_pilha(Pilha *pi, Aluno *al){
 if((pi == NULL) || (pi -> quantidade == 0))
   return 0;
 *al = pi -> elementos[pi -> quantidade - 1];
 /* No conteúdo da variável *al irá copiar o vetor dados na
 posição quantidade -1 (último elemento inserido) */
 return 1;
}

int empilhar(Pilha *pi, Aluno *al){
 if (pilha_cheia(pi)) return 0;
 pi -> elementos[pi -> topo] = *al;
 pi -> topo++;
 pi -> quantidade++;
   return 1;
}

int desempilhar(Pilha *pi, Aluno *al){
 if (pilha_vazia(pi)) return 0;

 *al = pi -> elementos[--pi -> topo];
 pi -> quantidade--;
   return 1;
}

void libera_pilha(Pilha *pi){
 free(pi);
}

void imprimir_pilha(Pilha *pi){
 for ( int i = 0; i < pi->topo ; i++){
     printf("Nome: %s\n", pi->elementos[i].nome);
     printf("Matricula: %d\n", pi->elementos[i].matricula);
     printf("Nota 1: %.2lf\n", pi->elementos[i].n1);
     printf("Nota 2: %.2lf\n\n", pi->elementos[i].n2);
 }
}

int inverter_fila(Fila *fi){
```

```
 Aluno aluno;
Pilha *pi=(Pilha*) malloc(sizeof(Pilha));

cria_pilha(pi);
while(!fila_vazia(fi)){
  desenfileirar(fi, &aluno);
  empilhar(pi, &aluno);
}
while(!pilha_vazia(pi)){
  desempilhar(pi, &aluno);
  enfileirar(fi, &aluno);
}
return 1;
}
```

Arquivo **main** para testar as funções:

```
#include <stdio.h>
#include "FilaPilhaEst_1.h"
#include <string.h>

int main(void){
 Fila fi;

 /* Aluno 1 */
  Aluno AlunoX;
 AlunoX.matricula = 1111;

 strcpy(AlunoX.nome, "Arnaldo Cesar Coelho");

 AlunoX.n1 = 10.0;
 AlunoX.n2 = 9.73;

 /* Aluno 2 */

 Aluno AlunoY;
 AlunoY.matricula = 2222;

 strcpy(AlunoY.nome, "Balotelli Soares Muniz");
```

```
AlunoY.n1 = 8;
AlunoY.n2 = 9.73;

/* Aluno 3 */

Aluno AlunoZ;
AlunoZ.matricula = 3333;

strcpy(AlunoZ.nome, "Cristiano Ronaldo Silva");

AlunoZ.n1 = 7;
AlunoZ.n2 = 5;

/* Aluno 4 */

Aluno AlunoB;
AlunoB.matricula = 4444;

strcpy(AlunoB.nome, "Diego Maradona Juarez");

AlunoB.n1 = 8;
AlunoB.n2 = 9;

/* Aluno 5 */

Aluno AlunoV;
AlunoV.matricula = 5555;

strcpy(AlunoV.nome, "Eduardo Sanchez Souza");

AlunoV.n1 = 5;
AlunoV.n2 = 10;

/* Criando a fila vazia */
cria_fila(&fi);

/* Enfileirando os elementos */
enfileirar(&fi, &AlunoX);
enfileirar(&fi, &AlunoY);
enfileirar(&fi, &AlunoZ);
enfileirar(&fi, &AlunoB);
enfileirar(&fi, &AlunoV);
```

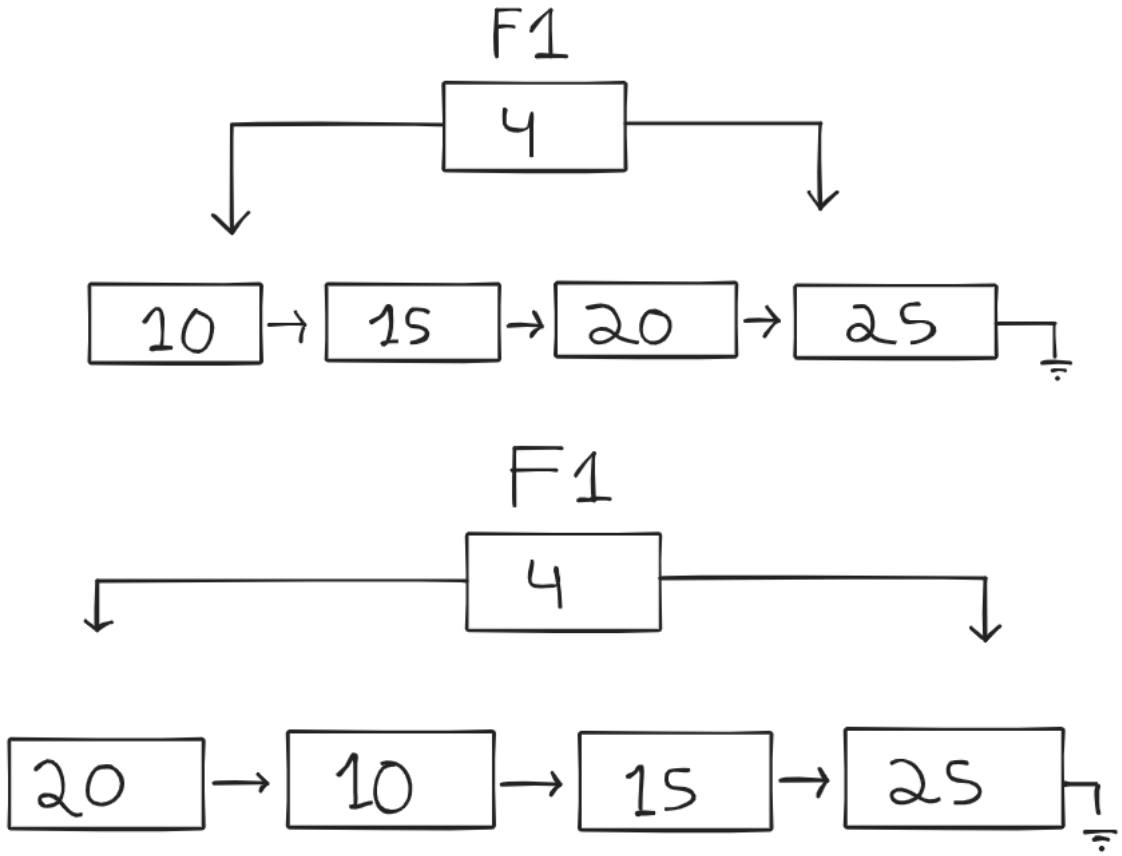
```c
/* Exibindo a fila */
 imprimir_fila(&fi);

 printf("INVERTENDO A ORDEM DOS ELEMENTOS:\n\n");

 /* Invertendo a fila com a função */
 inverter_fila(&fi);
  /* Exibindo a fila com os elementos invertidos */
 imprimir_fila(&fi);

 return 0;
}
```

Saída no terminal com o antes e depois da inversão da fila:

```
Nome: Arnaldo Cesar Coelho
Matricula: 1111
Nota 1: 10.00
Nota 2: 9.73

Nome: Balotelli Soares Muniz
Matricula: 2222
Nota 1: 8.00
Nota 2: 9.73

Nome: Cristiano Ronaldo Silva
Matricula: 3333
Nota 1: 7.00
Nota 2: 5.00

Nome: Diego Maradona Juarez
Matricula: 4444
Nota 1: 8.00
Nota 2: 9.00

Nome: Eduardo Sanchez Souza
Matricula: 5555
Nota 1: 5.00
Nota 2: 10.00
```

```
INVERTENDO A ORDEM DOS ELEMENTOS:

Nome: Eduardo Sanchez Souza
Matricula: 5555
Nota 1: 5.00
Nota 2: 10.00

Nome: Diego Maradona Juarez
Matricula: 4444
Nota 1: 8.00
Nota 2: 9.00

Nome: Cristiano Ronaldo Silva
Matricula: 3333
Nota 1: 7.00
Nota 2: 5.00

Nome: Balotelli Soares Muniz
Matricula: 2222
Nota 1: 8.00
Nota 2: 9.73

Nome: Arnaldo Cesar Coelho
Matricula: 1111
Nota 1: 10.00
Nota 2: 9.73
```

Questão 2:




```c
#include <stdio.h>

int main(){

 typedef struct stnofila{
   int dado;
   struct stnofila *prox;
 }no;

 typedef struct tfila{
   no* inicio;
   no* final;
   int tamanho;
 }fila;

 no *aux;
```

```c
 /*

 ↓   <-   4    ->   ↓
 10 -> 15 -> 20 -> 25

 */

 /* Criando a fila */
 fila filaz;
 /* Ponteiro inicio recebe o inicio da fila */
 no *fila_inicio = filaz.inicio;

 /* Percorrendo a fila */
 while(fila_inicio != NULL){
   /* Se o inicio da fila for 20 ele deve parar*/
   if(fila_inicio -> dado == 20)
     break;
   /* auxiliar recebe o atual */
   aux = fila_inicio;
   /* o elemento atual recebe o proximo elemento */
   fila_inicio = fila_inicio -> prox;
 }
 /* o elemento anterior ao atual recebe o elemento posterior ao atual
*/
 aux -> prox = fila_inicio -> prox;
 /* o no com valor 20 passa a apontar para o inicio */
 fila_inicio -> prox = filaz.inicio;
 /* agora o 20 será o inicio */
 filaz.inicio = fila_inicio;

 /*

 ↓   <-   4    ->   ↓
 20 -> 10 -> 15 -> 25

 */

 return 0;
}
```