

## Lista 6 de Laboratório de Programação II

Nome : Messias Feres Curi Melo

Matrícula : 2022003764

### Problema 1

#### 1.1

main.c

```
C main.c > main()
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <stdbool.h>
4  #include "deque.h"
5
6  int main()
7  {
8      Deque *D;
9      int elem, escolha;
10     printf("-<X Menu X>-\\n");
11     printf("1 - Criar Deque\\n");
12     printf("2 - Inserir um item no fim\\n");
13     printf("3 - Inserir um item no inicio\\n");
14     printf("4 - Ver o inicio do Deque\\n");
15     printf("5 - Ver o fim do Deque\\n");
16     printf("6 - Remover um item do fim\\n");
17     printf("7 - Remover um item do inicio\\n");
18     printf("8 - Imprimir o Deque\\n");
19     printf("9 - Destruir o Deque\\n");
20     printf("10 - Sair\\n");
21
22     while(true){
23         printf("Escolha: ");
24         scanf("%d", &escolha);
25
26         if(escolha == 1){
27             D = criaDeque();
28             printf("-> Criado com sucesso!\\n");
29         }else if(escolha == 2){
30             printf("-> Elemento a ser inserido no fim: ");
31             scanf("%d",&elem);
32             insereFim(D, elem);
33             printf("-> Inserido com sucesso!\\n");
34         }else if(escolha == 3){
35             printf("-> Elemento a ser inserido no inicio: ");
36             scanf("%d",&elem);
37             insereIni(D, elem);
38             printf("-> Inserido com sucesso!\\n");
39         }else if(escolha == 4){
40             verInicio(D, &elem);
41             printf("-> Inicio: %d\\n", elem);
42         }else if(escolha == 5){
43             verFim(D, &elem);
44             printf("-> Fim: %d\\n", elem);
45         }else if(escolha == 6){
46             removeFim(D);
47             printf("-> Removido item do fim com sucesso!\\n");
48         }else if(escolha == 7){
49             removeIni(D);
50             printf("-> Removido item do inicio com sucesso!\\n");
51         }else if(escolha == 8){
52             imprimeDeque(D);
53         }else if(escolha == 9){
54             destruiDeque(D);
55         }else if(escolha == 10){
56             return 0;
57         }
58     }
59 }
```

```

48     removeIni(D);
49     printf("-> Removido item do inicio com sucesso!\n");
50 }else if(escolha == 8){
51     imprime(D);
52 }else if(escolha == 9){
53     destroiDeque(D);
54     printf("-> Destruido com sucesso!\n");
55 }else if(escolha == 10){
56     printf("-> Finalizando!\n");
57     break;
58 }else{
59     printf("-> Número inválido!\n");
60 }
61 }
62
63 return 0;
64 }
65

```

## deque.c

```

C deque.c > criaDeque()
1  #include "deque.h"
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  Deque* criaDeque(){
6      Deque* dq;
7      dq = (Deque*) malloc (sizeof(Deque));
8      if(dq != NULL){
9          dq->qtd = dq->ini = dq->fim = 0;
10     }
11     return dq;
12 }
13
14 void destroiDeque(Deque *dq){
15     if(dq != NULL)
16         free(dq);
17 }
18
19 int tamanhoDeque(Deque *dq){
20     if(dq == NULL)
21         return -1;
22     return dq->qtd;
23 }
24
25 int estaCheio(Deque *dq){
26     if(dq == NULL)
27         return -1;
28     return (dq->qtd == MAX);
29 }
30
31 int estaVazio(Deque *dq){
32     if(dq == NULL)
33         return -1;
34     return (dq->qtd == 0);
35 }
36
37 int insereIni(Deque* dq, int elem){
38     if(dq == NULL) return 0;
39     if(estaCheio(dq)) return 0;
40     dq->ini = (dq->ini-1 < 0 ? MAX-1 : dq->ini-1);
41     dq->dados[dq->ini] = elem;
42     dq->qtd++;
43     return 1;
44 }
45
46 int insereFim(Deque* dq, int elem){
47     if(dq == NULL) return 0;

```

```

48     if(estaCheio(dq)) return 0;
49     dq->dados[dq->fim] = elem;
50     dq->fim = (dq->fim+1) % MAX;
51     dq->qtd++;
52     return 1;
53 }
54
55 int removeIni(Deque* dq){
56     if(dq == NULL) return 0;
57     if(estaVazio(dq)) return 0;
58     dq->ini = (dq->ini+1) % MAX;
59     dq->qtd--;
60     return 1;
61 }
62
63 int removeFim(Deque* dq){
64     if(dq == NULL) return 0;
65     if(estaVazio(dq)) return 0;
66     dq->fim = (dq->fim-1 < 0 ? MAX-1 : dq->fim-1);
67     dq->qtd--;
68     return 1;
69 }
70
71 int verInicio(Deque* dq, int* p){
72     if(dq == NULL) return 0;
73     if(estaVazio(dq)) return 0;
74     *p = dq->dados[dq->ini];
75     return 1;
76 }
77
78 int verFim(Deque* dq, int* p){
79     if(dq == NULL) return 0;
80     if(estaVazio(dq)) return 0;
81     int i = (dq->fim-1 < 0 ? MAX-1 : dq->fim-1);
82     *p = dq->dados[i];
83     return 1;
84 }
85
86 void imprime(Deque* dq){
87     if(dq == NULL) return;
88     if(estaVazio(dq)){
89         printf("Deque Vazio!\n");
90         return;
91     }
92     int i = dq->ini;
93     printf("Elementos: \n");
94     do{
95         printf("%d ", dq->dados[i]);
96         i = (i + 1) % MAX;
97     }while(i != dq->fim);
98     printf("\n");
99 }

```

## deque.h

```
C deque.h > ...
1  #ifndef DEQUE_H
2  #define DEQUE_H
3
4  #define MAX 100
5
6  typedef struct{
7      int qtd, ini, fim;
8      int dados[MAX];
9  }Deque;
10
11  Deque* criaDeque();
12
13  void destroiDeque(Deque *dq);
14
15  int tamanhoDeque(Deque *dq);
16
17  int estaCheio(Deque *dq);
18
19  int estaVazio(Deque *dq);
20
21  int insereIni(Deque* dq, int elem);
22
23  int insereFim(Deque* dq, int elem);
24
25  int removeIni(Deque* dq);
26
27  int removeFim(Deque* dq);
28
29  int verInicio(Deque* dq, int* p);
30
31  int verFim(Deque* dq, int* p);
32
33  void imprime(Deque* dq);
34  |
35  #endif
```

## Console

```
messiasfcn@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PRG II/Lista6/d1$ gcc main.c deque.h deque.c -o d1
messiasfcn@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PRG II/Lista6/d1$ ./d1
-<X Menu X>-
1 - Criar Deque
2 - Inserir um item no fim
3 - Inserir um item no início
4 - Ver o início do Deque
5 - Ver o fim do Deque
6 - Remover um item do fim
7 - Remover um item do início
8 - Imprimir o Deque
9 - Destruir o Deque
10 - Sair
Escolha: 1
-> Criado com sucesso!
Escolha: 2
-> Elemento a ser inserido no fim: 10
-> Inserido com sucesso!
Escolha: 2
-> Elemento a ser inserido no fim: 34
-> Inserido com sucesso!
Escolha: 3
-> Elemento a ser inserido no início: 45
-> Inserido com sucesso!
Escolha: 3
-> Elemento a ser inserido no início: 4
-> Inserido com sucesso!
Escolha: 8
Elementos:
4 45 10 34
Escolha: 4
-> Início: 4
Escolha: 5
-> Fim: 34
Escolha: 6
-> Removido item do fim com sucesso!
Escolha: 10
-> Finalizando!
messiasfcn@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PRG II/Lista6/d1$
```

## 1.2

### main.c

```
C main.c > main()
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <stdbool.h>
4  #include "deque.h"
5
6  int main(){
7      Deque *D;
8      int elem, escolha;
9      printf("-<X Menu X>-\\n");
10     printf("1 - Criar Deque\\n");
11     printf("2 - Inserir um item no fim\\n");
12     printf("3 - Inserir um item no inicio\\n");
13     printf("4 - Ver o inicio do Deque\\n");
14     printf("5 - Ver o fim do Deque\\n");
15     printf("6 - Remover um item do fim\\n");
16     printf("7 - Remover um item do inicio\\n");
17     printf("8 - Imprimir o Deque\\n");
18     printf("9 - Destruir o Deque\\n");
19     printf("10 - Sair\\n");
20
21     while(true){
22         printf("Escolha: ");
23         scanf("%d", &escolha);
24
25         if(escolha == 1){
26             D = criaDeque();
27             printf("-> Criado com sucesso!\\n");
28         }else if(escolha == 2){
29             printf("-> Elemento a ser inserido no fim: ");
30             scanf("%d", &elem);
31             insereFim(D, elem);
32             printf("-> Inserido com sucesso!\\n");
33         }else if(escolha == 3){
34             printf("-> Elemento a ser inserido no inicio: ");
35             scanf("%d", &elem);
36             insereIni(D, elem);
37             printf("-> Inserido com sucesso!\\n");
38         }else if(escolha == 4){
39             verInicio(D, &elem);
40             printf("-> Inicio: %d\\n", elem);
41         }else if(escolha == 5){
42             verFim(D, &elem);
43             printf("-> Fim: %d\\n", elem);
44         }else if(escolha == 6){
45             removeFim(D);
46             printf("-> Removido item do fim com sucesso!\\n");
47         }else if(escolha == 7){
48             removeIni(D);
49             printf("-> Removido item do inicio com sucesso!\\n");
50         }else if(escolha == 8){
51             imprime(D);
52         }else if(escolha == 9){
53             destroiDeque(D);
54             printf("-> Destruído com sucesso!\\n");
55         }else if(escolha == 10){
56             printf("-> Finalizando!\\n");
57             break;
58         }else{
59             printf("-> Número inválido!\\n");
60         }
61     }
62
63     return 0;
64 }
65
```

## deque.c

```
C deque.c > ...
1  #include "deque.h"
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  NO* alocarNO(){
6      return (NO*) malloc (sizeof(NO));
7  }
8
9  void liberarNO(NO* q){
10     free(q);
11 }
12
13 Deque* criaDeque(){
14     Deque* dq;
15     dq = (Deque*) malloc (sizeof(Deque));
16     if(dq != NULL){
17         dq->qtd = 0;
18         dq->ini = NULL;
19         dq->fim = NULL;
20     }
21     return dq;
22 }
23
24 void destroiDeque(Deque *dq){
25     if(dq != NULL){
26         NO* aux;
27         while(dq->ini != NULL){
28             aux = dq->ini;
29             dq->ini = dq->ini->prox;
30             liberarNO(aux);
31         }
32         free(dq);
33     }
34 }
35
36 int tamanhoDeque(Deque *dq){
37     if(dq == NULL)
38         return -1;
39     return dq->qtd;
40 }
41
42 int estaVazio(Deque *dq){
43     if(dq == NULL)
44         return -1;
45     return (dq->qtd == 0);
46 }
```

```

47
48  ▾ int insereIni(Deque* dq, int elem){
49      if(dq == NULL) return 0;
50      NO* novo = alocarNO();
51      if(novo == NULL) return 0;
52      novo->info = elem;
53      novo->ant = NULL;
54      ▾ if(estaVazio(dq)){
55          novo->prox = NULL;
56          dq->fim = novo;
57      ▾ }else{
58          dq->ini->ant = novo;
59          novo->prox = dq->ini;
60      }
61      dq->ini = novo;
62      dq->qtd++;
63      return 1;
64  }
65
66  ▾ int insereFim(Deque* dq, int elem){
67      if(dq == NULL) return 0;
68      NO* novo = alocarNO();
69      if(novo == NULL) return 0;
70      novo->info = elem;
71      novo->prox = NULL;
72      ▾ if(estaVazio(dq)){
73          novo->ant = NULL;
74          dq->ini = novo;
75      ▾ }else{
76          dq->fim->prox = novo;
77          novo->ant = dq->fim;
78      }
79      dq->fim = novo;
80      dq->qtd++;
81      return 1;
82  }
83
84  ▾ int removeIni(Deque* dq){
85      if(dq == NULL) return 0;
86      if(estaVazio(dq)) return 0;
87      NO* aux = dq->ini;
88      ▾ if(dq->ini == dq->fim){
89          dq->ini = dq->fim = NULL;
90      ▾ }else{
91          dq->ini = dq->ini->prox;
92          dq->ini->ant = NULL;

```



```

93     }
94     liberarNO(aux);
95     dq->qtd--;
96     return 1;
97 }
98
99 int removeFim(Deque* dq){
100     if(dq == NULL) return 0;
101     if(estaVazio(dq)) return 0;
102     NO* aux = dq->fim;
103     if(dq->ini == dq->fim){
104         dq->ini = dq->fim = NULL;
105     }else{
106         dq->fim = dq->fim->ant;
107         dq->ini->prox = NULL;
108     }
109     liberarNO(aux);
110     dq->qtd--;
111     return 1;
112 }
113
114 int verInicio(Deque* dq, int* p){
115     if(dq == NULL) return 0;
116     if(estaVazio(dq)) return 0;
117     *p = dq->ini->info;
118     return 1;
119 }
120
121 int verFim(Deque* dq, int* p){
122     if(dq == NULL) return 0;
123     if(estaVazio(dq)) return 0;
124     *p = dq->fim->info;
125     return 1;
126 }
127
128 void imprime(Deque* dq){
129     if(dq == NULL) return;
130     if(estaVazio(dq)){
131         printf("Deque Vazio!\n");
132         return;
133     }
134     NO* aux = dq->ini;
135     printf("Elementos:\n");
136     while(aux != NULL){
137         printf("%d ", aux->info);
138         aux = aux->prox;
139     }
140     printf("\n");
141 }
142

```

## deque.h

```
C deque.h > ...
1  #ifndef DDE_H
2  #define DDE_H
3
4  typedef struct NO{
5      int info;
6      struct NO* prox;
7      struct NO* ant;
8  }NO;
9
10 typedef struct{
11     int qtd;
12     struct NO* ini;
13     struct NO* fim;
14 }Deque;
15
16 NO* alocarNO();
17
18 void liberarNO(NO* q);
19
20 Deque* criaDeque();
21
22 void destroiDeque(Deque *dq);
23
24 int tamanhoDeque(Deque *dq);
25
26 int estaVazio(Deque *dq);
27
28 int insereIni(Deque* dq, int elem);
29
30 int insereFim(Deque* dq, int elem);
31
32 int removeIni(Deque* dq);
33
34 int removeFim(Deque* dq);
35
36 int verInicio(Deque* dq, int* p);
37
38 int verFim(Deque* dq, int* p);
39
40 void imprime(Deque* dq);
41
42 #endif
```

## Console

```
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/d2$ gcc main.c deque.h deque.c -o d2
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/d2$ ./d2
-<X Menu X>-
1 - Criar Deque
2 - Inserir um item no fim
3 - Inserir um item no início
4 - Ver o início do Deque
5 - Ver o fim do Deque
6 - Remover um item do fim
7 - Remover um item do início
8 - Imprimir o Deque
9 - Destruir o Deque
10 - Sair
Escolha: 1
-> Criado com sucesso!
Escolha: 2
-> Elemento a ser inserido no fim: 54
-> Inserido com sucesso!
Escolha: 2
-> Elemento a ser inserido no fim: 32
-> Inserido com sucesso!
Escolha: 3
-> Elemento a ser inserido no início: 43
-> Inserido com sucesso!
Escolha: 4
-> Início: 43
Escolha: 5
-> Fim: 32
Escolha: 7
-> Removido item do inicio com sucesso!
Escolha: 8
Elementos:
54 32
Escolha: 9
-> Destruído com sucesso!
Escolha: 10
-> Finalizando!
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/d2$
```

## Problema 2

### 2.1

main.c

```
C main.c > main()
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <stdbool.h>
4  #include "fpse.h"
5
6  int main(){
7      FilaP* F;
8      int elem, escolha, prio;
9      printf("-<X Menu X>-\n");
10     printf("1 - Criar Fila\n");
11     printf("2 - Inserir um item pela prioridade\n");
12     printf("3 - Ver o início da Fila\n");
13     printf("4 - Remover um item\n");
14     printf("5 - Imprimir a Fila\n");
15     printf("6 - Mostrar o tamanho da Fila\n");
16     printf("7 - Destruir a Fila\n");
17     printf("8 - Sair\n");
18
19     while(true){
20         printf("Escolha: ");
21         scanf("%d", &escolha);
22
23         if(escolha == 1){
24             F = criaFila();
25             printf("-> Criado com sucesso!\n");
26         }else if(escolha == 2){
27             printf("-> Elemento a ser inserido: ");
28             scanf("%d",&elem);
29             printf("-> Prioridade do elemento: ");
30             scanf("%d",&prio);
31             inserirPrio(F, elem, prio);
32             printf("-> Inserido com sucesso!\n");
33         }else if(escolha == 3){
34             verIni(F, &elem);
35             printf("-> Início: %d\n", elem);
36         }else if(escolha == 4){
37             removeIni(F);
38             printf("-> Removido com sucesso!\n");
39         }else if(escolha == 5){
40             imprime(F);
41         }else if(escolha == 6){
42             int tamanho = tamanhoFila(F);
43             printf("-> O tamanho da fila é: %d\n", tamanho);
44         }else if(escolha == 7){
45             destroiFila(F);
46             printf("-> Destruído com sucesso!\n");
47         }else if(escolha == 8){
48             break;
49         }else{
50             printf("-> Número inválido!\n");
51         }
52     }
53
54     return 0;
55 }
56
57
```

## fpse.c

```
C fpse.c > tamanhoFila(FilaP *)
1  #include "fpse.h"
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  NO* alocarNO(){
6      return (NO*) malloc (sizeof(NO));
7  }
8
9  void liberarNO(NO* q){
10     free(q);
11 }
12
13 FilaP* criaFila(){
14     FilaP* fp;
15     fp = (FilaP*) malloc (sizeof(FilaP));
16     if(fp != NULL)
17         *fp = NULL;
18     return fp;
19 }
20
21 int estaVazia(FilaP* fp){
22     if(fp == NULL) return -1;
23     return ((*fp) == NULL);
24 }
25
26 int tamanhoFila(FilaP* fp){
27     if (fp == NULL) return 0;
28     int tamanho = 0;
29     NO* aux = *fp;
30     while (aux != NULL) {
31         tamanho++;
32         aux = aux->prox;
33     }
34     return tamanho;
35 }
36
37 int inserirPrio(FilaP* fp, int elem, int pri){
38     if(fp == NULL) return 0;
39     NO* novo = alocarNO();
40     if(novo == NULL) return 0;
41
42     novo->info = elem;
43     novo->prio = pri;
44
45     if(estaVazia(fp)){
46         novo->prox = *fp;
47         *fp = novo;
```

```

48     }else{
49         NO* aux, *ant;
50         ant = NULL;
51         aux = *fp; //Inicio
52         while(aux != NULL && aux->prio >= novo->prio){
53             ant = aux;
54             aux = aux->prox;
55         }
56         if(ant == NULL){
57             novo->prox = *fp;
58             *fp = novo;
59         }else{
60             novo->prox = ant->prox;
61             ant->prox = novo;
62         }
63     }
64     return 1;
65 }
66
67 int removeIni(FilaP* fp){
68     if(fp == NULL) return 0;
69     if(estaVazia(fp)) return 0;
70     NO* aux = *fp;
71     *fp = aux->prox;
72     liberarNO(aux);
73     return 1;
74 }
75
76 int verIni(FilaP* fp, int* p){
77     if(fp == NULL) return 0;
78     if(estaVazia(fp)) return 0;
79     *p = (*fp)->info;
80     return 1;
81 }
82
83
84 void imprime(FilaP* fp){
85     if(fp == NULL) return;
86     if(estaVazia(fp)){
87         printf("Fila Vazia!\n");
88         return;
89     }
90     NO* aux = *fp;
91     while(aux != NULL){
92         printf("[%d, %d] ", aux->prio, aux->info);
93         aux = aux->prox;
94     }
95     printf("\n");
96 }
97
98 void destroiFila(FilaP* fp){
99     if(fp != NULL){
100         NO* aux;
101         while((*fp) != NULL){
102             aux = *fp;
103             *fp = (*fp)->prox;
104             liberarNO(aux);
105         }
106         free(fp);
107     }
108 }
109

```

## fpse.h

```
C fpse.h > tamanhoFila(FilaP *)
1  #ifndef FPSE_H
2  #define FPSE_H
3
4  typedef struct NO{
5      int info, prio;
6      struct NO* prox;
7  }NO;
8
9  typedef struct NO* FilaP;
10
11  NO* alocarNO();
12
13  void liberarNO(NO* q);
14
15  FilaP* criaFila();
16
17  int estaVazia(FilaP* fp);
18
19  int tamanhoFila(FilaP* fp);
20
21  int inserirPrio(FilaP* fp, int elem, int pri);
22
23  int removeIni(FilaP* fp);
24
25  int verIni(FilaP* fp, int* p);
26
27  void imprime(FilaP* fp);
28
29  void destroiFila(FilaP* fp);
30
31  #endif
```

## Console

```
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/f1$ gcc main.c fpse.h fpse.c -o f1
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/f1$ ./f1
- <X Menu X> -
1 - Criar Fila
2 - Inserir um item pela prioridade
3 - Ver o início da Fila
4 - Remover um item
5 - Imprimir a Fila
6 - Mostrar o tamanho da Fila
7 - Destruir a Fila
8 - Sair
Escolha: 1
-> Criado com sucesso!
Escolha: 2
-> Elemento a ser inserido: 10
-> Prioridade do elemento: 3
-> Inserido com sucesso!
Escolha: 2
-> Elemento a ser inserido: 37
-> Prioridade do elemento: 1
-> Inserido com sucesso!
Escolha: 2
-> Elemento a ser inserido: 86
-> Prioridade do elemento: 2
-> Inserido com sucesso!
Escolha: 3
-> Início: 10
Escolha: 5
[3, 10] [2, 86] [1, 37]
Escolha: 6
-> O tamanho da fila é: 3
Escolha: 7
-> Destruído com sucesso!
Escolha: 8
-> Finalizando!
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/f1$
```

## 2.2

### main.c

```
C main.c > main()
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <stdbool.h>
4  #include "fpheap.h"
5
6  int main(){
7      FilaP* F;
8      int elem, escolha, prio;
9      printf("-<X Menu X>-\\n");
10     printf("1 - Criar Fila\\n");
11     printf("2 - Inserir um item pela prioridade\\n");
12     printf("3 - Ver o início da Fila\\n");
13     printf("4 - Remover um item\\n");
14     printf("5 - Imprimir a Fila\\n");
15     printf("6 - Mostrar o tamanho da Fila\\n");
16     printf("7 - Destruir a Fila\\n");
17     printf("8 - Sair\\n");
18
19     while(true){
20         printf("Escolha: ");
21         scanf("%d", &escolha);
22
23         if(escolha == 1){
24             F = criaFila();
25             printf("-> Criado com sucesso!\\n");
26         }else if(escolha == 2){
27             printf("-> Elemento a ser inserido: ");
28             scanf("%d",&elem);
29             printf("-> Prioridade do elemento: ");
30             scanf("%d",&prio);
31             inserirPrio(F, elem, prio);
32             printf("-> Inserido com sucesso!\\n");
33         }else if(escolha == 3){
34             verIni(F, &elem, &prio);
35             printf("-> Valor inicial: %d e prioridade: %d\\n", elem, prio);
36         }else if(escolha == 4){
37             removeIni(F);
38             printf("-> Removido com sucesso!\\n");
39         }else if(escolha == 5){
40             imprime(F);
41         }else if(escolha == 6){
42             int tamanho = tamanhoFila(F);
43             printf("-> O tamanho da fila é: %d\\n", tamanho);
44         }else if(escolha == 7){
45             destroiFila(F);
46             printf("-> Destruído com sucesso!\\n");
47         }else if(escolha == 8){
48             printf("-> Finalizando!\\n");
49             break;
50         }else{
51             printf("-> Número inválido!\\n");
52         }
53     }
54
55     return 0;
56 }
57
```



## fpheap.c

```
C fpheap.c > ...
1  #include "fpheap.h"
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  Filap* criaFila(){
6      Filap* fp;
7      fp = (Filap*) malloc (sizeof(Filap));
8      if(fp != NULL)
9          fp->qtd = 0;
10     return fp;
11 }
12
13 void destroiFila(Filap* fp){
14     if(fp != NULL)
15         free(fp);
16 }
17
18 int tamanhoFila(Filap *fp){
19     if(fp == NULL) return -1;
20     return fp->qtd;
21 }
22
23 int estaCheia(Filap *fp){
24     if(fp == NULL) return -1;
25     return (fp->qtd == MAX);
26 }
27
28 int estaVazia(Filap *fp){
29     if(fp == NULL) return -1;
30     return (fp->qtd == 0);
31 }
32
33 void trocaNO(NO* a, NO* b){
34     NO temp;
35     temp.info = a->info;
36     temp.prio = a->prio;
37     a->info = b->info;
38     a->prio = b->prio;
39     b->info = temp.info;
40     b->prio = temp.prio;
41 }
42
43 void ajustaHeapInsere(Filap* fp, int filho){
44     NO temp;
45     int pai = (filho-1)/2;
46     int prioPai = fp->dados[pai].prio;
47     int prioFilho = fp->dados[filho].prio;
```

```

48  while(filho > 0 && prioPai < prioFilho){
49      trocaNO(&fp->dados[filho], &fp->dados[pai]);
50      filho = pai;
51      pai = (pai-1)/2;
52      prioPai = fp->dados[pai].prio;
53      prioFilho = fp->dados[filho].prio;
54  }
55  }
56
57  int inserirPrio(FilaP* fp, int elem, int pri){
58      if(fp == NULL) return 0;
59      if(estaCheia(fp)) return 0;
60      fp->dados[fp->qtd].info = elem;
61      fp->dados[fp->qtd].prio = pri;
62      ajustaHeapInsere(fp, fp->qtd);
63      fp->qtd++;
64      return 1;
65  }
66
67  void ajustaHeapRemove(FilaP* fp, int pai){
68      NO temp;
69      int filho = 2*pai + 1;
70  while(filho < fp->qtd){
71      if(filho < fp->qtd-1)
72          if(fp->dados[filho].prio < fp->dados[filho+1].prio)
73              filho++;
74
75      if(fp->dados[pai].prio > fp->dados[filho].prio)
76          break;
77
78      trocaNO(&fp->dados[pai], &fp->dados[filho]);
79      pai = filho;
80      filho = 2*pai + 1;
81  }
82  }
83
84  int removeIni(FilaP* fp){
85      if(fp == NULL) return 0;
86      if(estaVazia(fp)) return 0;
87
88      fp->qtd--;
89      fp->dados[0].info = fp->dados[fp->qtd].info;
90      fp->dados[0].prio = fp->dados[fp->qtd].prio;
91      ajustaHeapRemove(fp, 0);
92      return 1;
93  }
94
95  int verIni(FilaP* fp, int* valor, int* pri){
96      if(fp == NULL) return 0;
97      if(estaVazia(fp)) return 0;
98      *valor = fp->dados[0].info;
99      *pri = fp->dados[0].prio;
100     return 1;
101 }
102
103 void imprime(FilaP* fp){
104     if(fp == NULL) return;
105     if(estaVazia(fp)){
106         printf("Fila Vazia!\n");
107         return;
108     }
109     printf("Elementos:\n");
110     int i;
111     for(i=0; i<fp->qtd; i++)
112         printf("[%d, %d] (%d) -- ", fp->dados[i].prio, fp->dados[i].info, i);
113     printf("\n");
114 }
115

```

## fpheap.h

```
C fpheap.h > ...
1  #ifndef FPHEAP_H
2  #define FPHEAP_H
3
4  #define MAX 100
5  typedef struct NO{
6      int info, prio;
7  }NO;
8  typedef struct{
9      int qtd;
10     NO dados[MAX];
11 }FilaP;
12
13 FilaP* criaFila();
14
15 void destroiFila(FilaP* fp);
16
17 int tamanhoFila(FilaP *fp);
18
19 int estaCheia(FilaP *fp);
20
21 int estaVazia(FilaP *fp);
22
23 void trocaNO(NO* a, NO* b);
24
25 void ajustaHeapInsere(FilaP* fp, int filho);
26
27 int inserirPrio(FilaP* fp, int elem, int pri);
28
29 void ajustaHeapRemove(FilaP* fp, int pai);
30
31 int removeIni(FilaP* fp);
32
33 int verIni(FilaP* fp, int* valor, int* pri);
34
35 void imprime(FilaP* fp);
36
37 #endif
```

## Console

```
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/f2$ gcc main.c fpheap.h fpheap.c -o f2
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/f2$ ./f2
-<X Menu X>-
1 - Criar Fila
2 - Inserir um item pela prioridade
3 - Ver o início da Fila
4 - Remover um item
5 - Imprimir a Fila
6 - Mostrar o tamanho da Fila
7 - Destruir a Fila
8 - Sair
Escolha: 1
-> Criado com sucesso!
Escolha: 2
-> Elemento a ser inserido: 10
-> Prioridade do elemento: 2
-> Inserido com sucesso!
Escolha: 2
-> Elemento a ser inserido: 13
-> Prioridade do elemento: 1
-> Inserido com sucesso!
Escolha: 3
-> Valor inicial: 10 e prioridade: 2
Escolha: 2
-> Elemento a ser inserido: 32
-> Prioridade do elemento: 5
-> Inserido com sucesso!
Escolha: 5
Elementos:
[5, 32] (0) -- [1, 13] (1) -- [2, 10] (2) --
Escolha: 4
-> Removido com sucesso!
Escolha: 6
-> O tamanho da fila é: 2
Escolha: 7
-> Destruído com sucesso!
Escolha: 8
-> Finalizando!
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/f2$
```