Lista 6 de Laboratório de Programação II
Nome : Messias Feres Curi Melo
Matrícula : 2022003764

Problema 1
1.1

main.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include "deque.h"

int main(){
    Deque *D;
    int elem, escolha;
    printf("-<X Menu X>-\n");
    printf("1 - Criar Deque\n");
    printf("2 - Inserir um item no fim\n");
    printf("3 - Inserir um item no inicio\n");
    printf("4 - Ver o inicio do Deque\n");
    printf("5 - Ver o fim do Deque\n");
    printf("6 - Remover um item do fim\n");
    printf("7 - Remover um item do inicio\n");
    printf("8 - Imprimir o Deque\n");
    printf("9 - Destruir o Deque\n");
    printf("10 - Sair\n");

    while(true){
        printf("Escolha: ");
        scanf("%d", &escolha);

        if(escolha == 1){
            D = criaDeque();
            printf("-> Criado com sucesso!\n");
        }else if(escolha == 2){
            printf("-> Elemento a ser inserido no fim: ");
            scanf("%d",&elem);
            insereFim(D, elem);
            printf("-> Inserido com sucesso!\n");
        }else if(escolha == 3){
            printf("-> Elemento a ser inserido no inicio: ");
            scanf("%d",&elem);
            insereIni(D, elem);
            printf("-> Inserido com sucesso!\n");
        }else if(escolha == 4){
            verInicio(D, &elem);
            printf("-> Inicio: %d\n", elem);
        }else if(escolha == 5){
            verFim(D, &elem);
            printf("-> Fim: %d\n", elem);
        }else if(escolha == 6){
            removeFim(D);
            printf("-> Removido item do fim com sucesso!\n");
        }else if(escolha == 7){
```

```
            removeIni(D);
            printf("-> Removido item do inicio com sucesso!\n");
        }else if(escolha == 8){
            imprime(D);
        }else if(escolha == 9){
            destroiDeque(D);
            printf("-> Destruido com sucesso!\n");
        }else if(escolha == 10){
            printf("-> Finalizando!\n");
            break;
        }else{
            printf("-> Número inválido!\n");
        }
    }

    return 0;
}

```

deque.c

```
deque.c > criaDeque()
#include "deque.h"
#include <stdio.h>
#include <stdlib.h>

Deque* criaDeque(){
    Deque* dq;
    dq = (Deque*) malloc (sizeof(Deque));
    if(dq != NULL){
        dq->qtd = dq->ini = dq->fim = 0;
    }
    return dq;
}

void destroiDeque(Deque *dq){
    if(dq != NULL)
        free(dq);
}

int tamanhoDeque(Deque *dq){
    if(dq == NULL)
        return -1;
    return dq->qtd;
}

int estaCheio(Deque *dq){
    if(dq == NULL)
        return -1;
    return (dq->qtd == MAX);
}

int estaVazio(Deque *dq){
    if(dq == NULL)
        return -1;
    return (dq->qtd == 0);
}

int insereIni(Deque* dq, int elem){
    if(dq == NULL) return 0;
    if(estaCheio(dq)) return 0;
    dq->ini = (dq->ini-1 < 0 ? MAX-1 : dq->ini-1);
    dq->dados[dq->ini] = elem;
    dq->qtd++;
    return 1;
}

int insereFim(Deque* dq, int elem){
    if(dq == NULL) return 0;
```

```c
    if(estaCheio(dq)) return 0;
    dq->dados[dq->fim] = elem;
    dq->fim = (dq->fim+1) % MAX;
    dq->qtd++;
    return 1;
}

int removeIni(Deque* dq){
    if(dq == NULL) return 0;
    if(estaVazio(dq)) return 0;
    dq->ini = (dq->ini+1) % MAX;
    dq->qtd--;
    return 1;
}

int removeFim(Deque* dq){
    if(dq == NULL) return 0;
    if(estaVazio(dq)) return 0;
    dq->fim = (dq->fim-1 < 0 ? MAX-1 : dq->fim-1);
    dq->qtd--;
    return 1;
}

int verInicio(Deque* dq, int* p){
  if(dq == NULL) return 0;
  if(estaVazio(dq)) return 0;
  *p = dq->dados[dq->ini];
  return 1;
}

int verFim(Deque* dq, int* p){
  if(dq == NULL) return 0;
  if(estaVazio(dq)) return 0;
  int i = (dq->fim-1 < 0 ? MAX-1 : dq->fim-1);
  *p = dq->dados[i];
  return 1;
}

void imprime(Deque* dq){
  if(dq == NULL) return;
  if(estaVazio(dq)){
    printf("Deque Vazio!\n");
    return;
  }
  int i = dq->ini;
  printf("Elementos: \n");
  do{
    printf("%d ", dq->dados[i]);
    i = (i + 1) % MAX;
  }while(i != dq->fim);
  printf("\n");
}
```

deque.h

C deque.h > ...

```c
#ifndef DEQUE_H
#define DEQUE_H

#define MAX 100

typedef struct{
    int qtd, ini, fim;
    int dados[MAX];
}Deque;

Deque* criaDeque();

void destroiDeque(Deque *dq);

int tamanhoDeque(Deque *dq);

int estaCheio(Deque *dq);

int estaVazio(Deque *dq);

int insereIni(Deque* dq, int elem);

int insereFim(Deque* dq, int elem);

int removeIni(Deque* dq);

int removeFim(Deque* dq);

int verInicio(Deque* dq, int* p);

int verFim(Deque* dq, int* p);

void imprime(Deque* dq);

#endif
```

Console

```
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/d1$ gcc main.c deque.h deque.c -o d1
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/d1$ ./d1
-<X Menu X>-
1 - Criar Deque
2 - Inserir um item no fim
3 - Inserir um item no inicio
4 - Ver o inicio do Deque
5 - Ver o fim do Deque
6 - Remover um item do fim
7 - Remover um item do inicio
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!
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PROG II/Lista6/d1$
```

1.2

main.c

C main.c > main()

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include "deque.h"

int main(){
    Deque *D;
    int elem, escolha;
    printf("-<X Menu X>-\n");
    printf("1 - Criar Deque\n");
    printf("2 - Inserir um item no fim\n");
    printf("3 - Inserir um item no inicio\n");
    printf("4 - Ver o inicio do Deque\n");
    printf("5 - Ver o fim do Deque\n");
    printf("6 - Remover um item do fim\n");
    printf("7 - Remover um item do inicio\n");
    printf("8 - Imprimir o Deque\n");
    printf("9 - Destruir o Deque\n");
    printf("10 - Sair\n");

    while(true){
        printf("Escolha: ");
        scanf("%d", &escolha);

        if(escolha == 1){
            D = criaDeque();
            printf("-> Criado com sucesso!\n");
        }else if(escolha == 2){
            printf("-> Elemento a ser inserido no fim: ");
            scanf("%d",&elem);
            insereFim(D, elem);
            printf("-> Inserido com sucesso!\n");
        }else if(escolha == 3){
            printf("-> Elemento a ser inserido no inicio: ");
            scanf("%d",&elem);
            insereIni(D, elem);
            printf("-> Inserido com sucesso!\n");
        }else if(escolha == 4){
            verInicio(D, &elem);
            printf("-> Inicio: %d\n", elem);
        }else if(escolha == 5){
            verFim(D, &elem);
            printf("-> Fim: %d\n", elem);
        }else if(escolha == 6){
            removeFim(D);
            printf("-> Removido item do fim com sucesso!\n");
        }else if(escolha == 7){
            removeIni(D);
            printf("-> Removido item do inicio com sucesso!\n");
        }else if(escolha == 8){
            imprime(D);
        }else if(escolha == 9){
            destroiDeque(D);
            printf("-> Destruido com sucesso!\n");
        }else if(escolha == 10){
            printf("-> Finalizando!\n");
            break;
        }else{
            printf("-> Número inválido!\n");
        }
    }

    return 0;
}

```

deque.c

C deque.c > ...

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

NO* alocarNO(){
    return (NO*) malloc (sizeof(NO));
}

void liberarNO(NO* q){
    free(q);
}

Deque* criaDeque(){
    Deque* dq;
    dq = (Deque*) malloc (sizeof(Deque));
    if(dq != NULL){
        dq->qtd = 0;
        dq->ini = NULL;
        dq->fim = NULL;
    }
    return dq;
}

void destroiDeque(Deque *dq){
    if(dq != NULL){
        NO* aux;
        while(dq->ini != NULL){
            aux = dq->ini;
            dq->ini = dq->ini->prox;
            liberarNO(aux);
        }
        free(dq);
    }
}

int tamanhoDeque(Deque *dq){
    if(dq == NULL)
        return -1;
    return dq->qtd;
}

int estaVazio(Deque *dq){
    if(dq == NULL)
        return -1;
    return (dq->qtd == 0);
}
```

```c

int insereIni(Deque* dq, int elem){
    if(dq == NULL) return 0;
    NO* novo = alocarNO();
    if(novo == NULL) return 0;
    novo->info = elem;
    novo->ant = NULL;
    if(estaVazio(dq)){
        novo->prox = NULL;
        dq->fim = novo;
    }else{
        dq->ini->ant = novo;
        novo->prox = dq->ini;
    }
    dq->ini = novo;
    dq->qtd++;
    return 1;
}

int insereFim(Deque* dq, int elem){
    if(dq == NULL) return 0;
    NO* novo = alocarNO();
    if(novo == NULL) return 0;
    novo->info = elem;
    novo->prox = NULL;
    if(estaVazio(dq)){
        novo->ant = NULL;
        dq->ini = novo;
    }else{
        dq->fim->prox = novo;
        novo->ant = dq->fim;
    }
    dq->fim = novo;
    dq->qtd++;
    return 1;
}

int removeIni(Deque* dq){
    if(dq == NULL) return 0;
    if(estaVazio(dq)) return 0;
    NO* aux = dq->ini;
    if(dq->ini == dq->fim){
        dq->ini = dq->fim = NULL;
    }else{
        dq->ini = dq->ini->prox;
        dq->ini->ant = NULL;
```

```
    }
    liberarNO(aux);
    dq->qtd--;
    return 1;
}

int removeFim(Deque* dq){
    if(dq == NULL) return 0;
    if(estaVazio(dq)) return 0;
    NO* aux = dq->fim;
    if(dq->ini == dq->fim){
        dq->ini = dq->fim = NULL;
    }else{
        dq->fim = dq->fim->ant;
        dq->ini->prox = NULL;
    }
    liberarNO(aux);
    dq->qtd--;
    return 1;
}

int verInicio(Deque* dq, int* p){
  if(dq == NULL) return 0;
  if(estaVazio(dq)) return 0;
  *p = dq->ini->info;
  return 1;
}

int verFim(Deque* dq, int* p){
  if(dq == NULL) return 0;
  if(estaVazio(dq)) return 0;
  *p = dq->fim->info;
  return 1;
}

void imprime(Deque* dq){
  if(dq == NULL) return;
  if(estaVazio(dq)){
    printf("Deque Vazio!\n");
    return;
  }
  NO* aux = dq->ini;
  printf("Elementos:\n");
  while(aux != NULL){
    printf("%d ", aux->info);
    aux = aux->prox;
  }
  printf("\n");
}
```

deque.h

C deque.h > ...

```c
#ifndef DDE_H
#define DDE_H

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

typedef struct{
    int qtd;
    struct NO* ini;
    struct NO* fim;
}Deque;

NO* alocarNO();

void liberarNO(NO* q);

Deque* criaDeque();

void destroiDeque(Deque *dq);

int tamanhoDeque(Deque *dq);

int estaVazio(Deque *dq);

int insereIni(Deque* dq, int elem);

int insereFim(Deque* dq, int elem);

int removeIni(Deque* dq);

int removeFim(Deque* dq);

int verInicio(Deque* dq, int* p);

int verFim(Deque* dq, int* p);

void imprime(Deque* dq);

#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 inicio
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 início com sucesso!
Escolha: 8
Elementos:
54 32
Escolha: 9
-> Destruido 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
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include "fpse.h"

int main(){
    FilaP* F;
    int elem, escolha, prio;
    printf("-<X Menu X>-\n");
    printf("1 - Criar Fila\n");
    printf("2 - Inserir um item pela prioridade\n");
    printf("3 - Ver o início da Fila\n");
    printf("4 - Remover um item\n");
    printf("5 - Imprimir a Fila\n");
    printf("6 - Mostrar o tamanho da Fila\n");
    printf("7 - Destruir a Fila\n");
    printf("8 - Sair\n");

    while(true){
        printf("Escolha: ");
        scanf("%d", &escolha);

        if(escolha == 1){
            F = criaFila();
            printf("-> Criado com sucesso!\n");
        }else if(escolha == 2){
            printf("-> Elemento a ser inserido: ");
            scanf("%d",&elem);
            printf("-> Prioridade do elemento: ");
            scanf("%d",&prio);
            inserirPrio(F, elem, prio);
            printf("-> Inserido com sucesso!\n");
        }else if(escolha == 3){
            verIni(F, &elem);
            printf("-> Início: %d\n", elem);
        }else if(escolha == 4){
            removeIni(F);
            printf("-> Removido com sucesso!\n");
        }else if(escolha == 5){
            imprime(F);
        }else if(escolha == 6){
            int tamanho = tamanhoFila(F);
            printf("-> O tamanho da fila é: %d\n", tamanho);
        }else if(escolha == 7){
            destroiFila(F);
            printf("-> Destruido com sucesso!\n");
        }else if(escolha == 8){
            break;
        }else{
            printf("-> Número inválido!\n");
        }
    }

    return 0;
}
```

fpse.c

C fpse.c > tamanhoFila(FilaP *)

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

NO* alocarNO(){
    return (NO*) malloc (sizeof(NO));
}

void liberarNO(NO* q){
    free(q);
}

FilaP* criaFila(){
    FilaP* fp;
    fp = (FilaP*) malloc (sizeof(FilaP));
    if(fp != NULL)
        *fp = NULL;
    return fp;
}

int estaVazia(FilaP* fp){
    if(fp == NULL) return -1;
    return ((*fp) == NULL);
}

int tamanhoFila(FilaP* fp){
    if (fp == NULL) return 0;
    int tamanho = 0;
    NO* aux = *fp;
    while (aux != NULL) {
        tamanho++;
        aux = aux->prox;
    }
    return tamanho;
}

int inserirPrio(FilaP* fp, int elem, int pri){
    if(fp == NULL) return 0;
    NO* novo = alocarNO();
    if(novo == NULL) return 0;

    novo->info = elem;
    novo->prio = pri;

    if(estaVazia(fp)){
        novo->prox = *fp;
        *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
#ifndef FPSE_H
#define FPSE_H

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

typedef struct NO* FilaP;

NO* alocarNO();

void liberarNO(NO* q);

FilaP* criaFila();

int estaVazia(FilaP* fp);

int tamanhoFila(FilaP* fp);

int inserirPrio(FilaP* fp, int elem, int pri);

int removeIni(FilaP* fp);

int verIni(FilaP* fp, int* p);

void imprime(FilaP* fp);

void destroiFila(FilaP* fp);

#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
-> Destruido 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
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include "fpheap.h"

int main(){
    FilaP* F;
    int elem, escolha, prio;
    printf("-<X Menu X>-\n");
    printf("1 - Criar Fila\n");
    printf("2 - Inserir um item pela prioridade\n");
    printf("3 - Ver o início da Fila\n");
    printf("4 - Remover um item\n");
    printf("5 - Imprimir a Fila\n");
    printf("6 - Mostrar o tamanho da Fila\n");
    printf("7 - Destruir a Fila\n");
    printf("8 - Sair\n");

    while(true){
        printf("Escolha: ");
        scanf("%d", &escolha);

        if(escolha == 1){
            F = criaFila();
            printf("-> Criado com sucesso!\n");
        }else if(escolha == 2){
            printf("-> Elemento a ser inserido: ");
            scanf("%d",&elem);
            printf("-> Prioridade do elemento: ");
            scanf("%d",&prio);
            inserirPrio(F, elem, prio);
            printf("-> Inserido com sucesso!\n");
        }else if(escolha == 3){
            verIni(F, &elem, &prio);
            printf("-> Valor inicial: %d e prioridade: %d\n", elem, prio);
        }else if(escolha == 4){
            removeIni(F);
            printf("-> Removido com sucesso!\n");
        }else if(escolha == 5){
            imprime(F);
        }else if(escolha == 6){
            int tamanho = tamanhoFila(F);
            printf("-> O tamanho da fila é: %d\n", tamanho);
        }else if(escolha == 7){
            destroiFila(F);
            printf("-> Destruído com sucesso!\n");
        }else if(escolha == 8){
            printf("-> Finalizando!\n");
            break;
        }else{
            printf("-> Número inválido!\n");
        }
    }

    return 0;
}
```

fpheap.c

fpheap.c > ...

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

FilaP* criaFila(){
    FilaP* fp;
    fp = (FilaP*) malloc (sizeof(FilaP));
    if(fp != NULL)
        fp->qtd = 0;
    return fp;
}

void destroiFila(FilaP* fp){
    if(fp != NULL)
        free(fp);
}

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

int estaCheia(FilaP *fp){
  if(fp == NULL) return -1;
  return (fp->qtd == MAX);
}

int estaVazia(FilaP *fp){
  if(fp == NULL) return -1;
  return (fp->qtd == 0);
}

void trocaNO(NO* a, NO* b){
    NO temp;
    temp.info = a->info;
    temp.prio = a->prio;
    a->info = b->info;
    a->prio = b->prio;
    b->info = temp.info;
    b->prio = temp.prio;
}

void ajustaHeapInsere(FilaP* fp, int filho){
    NO temp;
    int pai = (filho-1)/2;
    int prioPai = fp->dados[pai].prio;
    int prioFilho = fp->dados[filho].prio;
```

```c
    while(filho > 0 && prioPai < prioFilho){
        trocaNO(&fp->dados[filho], &fp->dados[pai]);
        filho = pai;
        pai = (pai-1)/2;
        prioPai = fp->dados[pai].prio;
        prioFilho = fp->dados[filho].prio;
    }
}

int inserirPrio(FilaP* fp, int elem, int pri){
    if(fp == NULL) return 0;
    if(estaCheia(fp)) return 0;
    fp->dados[fp->qtd].info = elem;
    fp->dados[fp->qtd].prio = pri;
    ajustaHeapInsere(fp, fp->qtd);
    fp->qtd++;
    return 1;
}

void ajustaHeapRemove(FilaP* fp, int pai){
    NO temp;
    int filho = 2*pai + 1;
    while(filho < fp->qtd){
        if(filho < fp->qtd-1)
            if(fp->dados[filho].prio < fp->dados[filho+1].prio)
                filho++;

        if(fp->dados[pai].prio > fp->dados[filho].prio)
            break;

        trocaNO(&fp->dados[pai], &fp->dados[filho]);
        pai = filho;
        filho = 2*pai + 1;
    }
}

int removeIni(FilaP* fp){
    if(fp == NULL) return 0;
    if(estaVazia(fp)) return 0;

    fp->qtd--;
    fp->dados[0].info = fp->dados[fp->qtd].info;
    fp->dados[0].prio = fp->dados[fp->qtd].prio;
    ajustaHeapRemove(fp, 0);
    return 1;
}

int verIni(FilaP* fp, int* valor, int* pri){
  if(fp == NULL) return 0;
  if(estaVazia(fp)) return 0;
  *valor = fp->dados[0].info;
  *pri = fp->dados[0].prio;
  return 1;
}

void imprime(FilaP* fp){
    if(fp == NULL) return;
    if(estaVazia(fp)){
        printf("Fila Vazia!\n");
        return;
    }
    printf("Elementos:\n");
    int i;
    for(i=0; i<fp->qtd; i++)
        printf("[%d, %d] (%d) -- ", fp->dados[i].prio, fp->dados[i].info, i);
    printf("\n");
}

```

fpheap.h

C fpheap.h > ...

```c
#ifndef FPHEAP_H
#define FPHEAP_H

#define MAX 100
typedef struct NO{
    int info, prio;
}NO;
typedef struct{
    int qtd;
    NO dados[MAX];
}FilaP;

FilaP* criaFila();

void destroiFila(FilaP* fp);

int tamanhoFila(FilaP *fp);

int estaCheia(FilaP *fp);

int estaVazia(FilaP *fp);

void trocaNO(NO* a, NO* b);

void ajustaHeapInsere(FilaP* fp, int filho);

int inserirPrio(FilaP* fp, int elem, int pri);

void ajustaHeapRemove(FilaP* fp, int pai);

int removeIni(FilaP* fp);

int verIni(FilaP* fp, int* valor, int* pri);

void imprime(FilaP* fp);

#endif
```

Console

```
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PRCG II/Lista6/f2$ gcc main.c fpheap.h fpheap.c -o f2
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PRCG II/Lista6/f2$ ./f2
-<X Menu X>-
1 - Criar Fila
2 - Inserir um item pela prioridade
3 - Ver o inicio 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
-> Destruido com sucesso!
Escolha: 8
-> Finalizando!
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/Facul/+LAB PRCG II/Lista6/f2$
```