Nome: João Paulo de Oliveira 11611BCC046

13° Aula prática

Uberlândia

2016

1.Código fonte:

* Main.c:

#include <stdio.h>

#include <stdlib.h>

#include "heap.h"

/\* Size of Queue \*/

int main(){

/\* Main Program \*/

int opn;

PriorityQ p;

que Q = Criafila();

do

{

printf("\n ### Priority Queue Operations(DSC order) ### \n\n");

printf("\n Press 1-Insert, 2-Delete,3-Display,4-Exit\n");

printf("\n Your option ? ");

scanf("%d",&opn);

switch(opn)

{

case 1:

printf("\n\nRead the element and its Priority?");

scanf("%d%d",&p.ele,&p.pr);

PQinsert(p.ele,p.pr,&Q);

break;

case 2:

p = Qdelete(&Q);

if( p.ele != -1)

printf("\n\nDeleted Element is %d \n",p.ele);

break;

case 3:

printf("\n\nStatus of Queue\n\n");

display(Q);

break;

case 4:

printf("\n\nTerminating \n\n");

break;

default:

printf("\n\nInvalid Option !!! Try Again !! \n\n");

break;

}

getch();

}

while(opn != 4);

return 0;

}

* Heap.c:

#include <stdio.h>

#include <stdlib.h>

#include"heap.h"

void PQinsert(int elem, int pri, que\* Q){

int i; /\* Function for Insert operation \*/

if( Qfull(\*Q)) printf("\n\nOverflow!!!!\n\n");

else{

i=Q->costa;

Q->costa++;

while(Q->filaP[i].pr >= pri && i >= 0){

Q->filaP[i+1]=Q->filaP[i];

i--;

}

Q->filaP[i+1].ele=elem;

Q->filaP[i+1].pr=pri;

}

}

int Qfull(que Q){

if(Q.costa==SIZE-1) {

return 1;

}

return 0;

}

int Qempty(que Q){

if(Q.frente > Q.costa) return 1;

return 0;

}

que Criafila(){

que\* Q;

Q = (que\*)malloc(5\*sizeof(que));

Q->costa=0;

Q->frente = 0;

Q->costa = -1;

return \*Q;

}

void display(que Q){

/\* Function to display status of Queue \*/

int i;

if(Qempty(Q)) printf(" \n Empty Queue\n");

else{

printf("Front->");

for(i=Q.frente; i<=Q.costa; i++)

printf("[%d,%d] ",Q.filaP[i].ele,Q.filaP[i].pr);

printf("<-Rear");

}

}

PriorityQ Qdelete(que \*Q){

PriorityQ p;

if(Qempty(\*Q)){

printf("\n\nUnderflow!!!!\n\n");

p.ele=-1;

p.pr=-1;

return(p);

}

else{

p.ele=Q->filaP[Q->frente].ele;

p.pr=Q->filaP[Q->frente].pr;

int i;

for(i=0;i<SIZE;i++){

Q->filaP[i] = Q->filaP[i+1];

}

Q->costa--;

return(p);

}

}

* Heap.h:

#ifndef HEAP\_H\_INCLUDED

#define HEAP\_H\_INCLUDED

#define SIZE 5

typedef struct PRQ{

int ele;

int pr;

} PriorityQ;

typedef struct Queue{

PriorityQ filaP[SIZE];

int frente, costa;

}que;

void PQinsert(int elem, int pri, que\* Q);

void display(que Q);

int Qfull(que Q);

int Qempty(que Q);

que Criafila();

PriorityQ Qdelete(que \*Q);

#endif // HEAP\_H\_INCLUDED

2.Print do funcionamento:

