# SSN College of Engineering, Kalavakkam

# Department of Computer Science and Engineering

# III Semester - CSE 'A ',’B’ & ‘C’

# UCS 1312 Data Structures Lab

# Academic Year: 2019-2020 Batch: 2018-2022

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Name : Rahul Ram M**

**Class : CSE – B**

**Register Number : 185001121**

**Exercise 10: Priority queue using binary heap**

**• Create a priority queue using max heap to store employee details with the fields:**

**Name, id, salary**

**• Insert the details of five employees into the binary heap such that highest paid**

**employee should be given priority.**

**• Relieve one employee from the binary heap.**

**• Display the tree after each operation.**

**//priorityqueue.h**

typedef struct priorityqueue{

int capacity,size;

struct details \*array;

}pq;

typedef struct details{

float salary;

char name[25];

int id;

}det;

pq\* createPQ(int maxele);

void display(pq \*h1);

void display2(det temp);

int isFull(pq \*h1);

int isEmpty(pq \*h1);

pq\* insert(char name1[],int id2,float sal1,pq \*h1);

det deletemax(pq \*h1);

pq\* createPQ(int maxele){

pq \*h1;

h1=(pq \*)malloc(sizeof(pq));

if(h1==NULL){

printf("Fatal Error: Out of space!\n");

}

h1->array=(det \*)malloc(sizeof(det)\*(maxele+1));

if(h1->array==NULL){

printf("Fatal Error: Out of space!\n");

}

h1->capacity=maxele;

h1->array[0].salary=0;

h1->array[0].id=0;

strcpy(h1->array[0].name,"<empty>");

h1->size=0;

return h1;

}

pq\* insert(char name1[],int id2,float sal1,pq \*h1){

int i;

if(isFull(h1)){

printf("Queue is full!\n");

return h1;

}

for(i=++h1->size;h1->array[i/2].salary<sal1 && h1->array[i/2].salary!=0;i/=2){

h1->array[i]=h1->array[i/2];

}

strcpy(h1->array[i].name,name1);

h1->array[i].id=id2;

h1->array[i].salary=sal1;

return h1;

}

det deletemax(pq \*h1)

{

int i, child;

det minelement,lastelement;

if(isEmpty(h1)){

printf("PQueue is empty\n");

return (h1->array[0]);

}

minelement=h1->array[1];

lastelement=h1->array[h1->size--];

for(i=1;(i\*2) <= h1->size;i=child){

child=i\*2;

if(child!=h1->size && h1->array[child+1].salary > h1->array[child].salary)

child++;

if(lastelement.salary < h1->array[child].salary)

h1->array[i]=h1->array[child];

else

break;

}

h1->array[i]=lastelement;

return minelement;

}

void display(pq \*h1){

printf("Displaying Tree!!!\n");

for(int i=1;i<=(h1->size);i++){

printf("%d : %s\t%d\t%.2f\n",i,h1->array[i].name,h1->array[i].id,h1->array[i].salary);

}

}

void display2(det temp){

printf("Displaying Highest Salary Employee!!!\n");

printf("Name : %s\nID : %d\nSalary : %.2f\n",temp.name,temp.id,temp.salary);

}

int isFull(pq \*h1){

if(h1->size==h1->capacity){

return 1;

}

return 0;

}

int isEmpty(pq \*h1){

if(h1->size==0){

return 1;

}

return 0;

}

**//main**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include"priorityqueue.h"

int main(){

pq \*h;

det priorityelement;

char sname[25];

float sal;

int maxemployees,id1,ch;

printf("Enter the number of employees : ");

scanf("%d",&maxemployees);

h=createPQ(maxemployees);

for(int i=0;i<maxemployees;i++){

printf("\nEnter the %dth Employee details!\n",i+1);

printf("Enter Name : ");

scanf("%s",sname);

printf("Enter ID : ");

scanf("%d",&id1);

printf("Enter salary : ");

scanf("%f",&sal);

h=insert(sname,id1,sal,h);

display(h);

}

printf("\nMenu\n1.Insert Employee Details.\n2.Dequeue Highest Salary Employee.\n3.Display Tree.\n4.Exit.\nEnter your choice : ");

scanf("%d",&ch);

while(ch!=4){

switch(ch){

case 1:{

printf("\nEnter Employee Details!\n");

printf("Enter Name : ");

scanf("%s",sname);

printf("Enter ID : ");

scanf("%d",&id1);

printf("Enter salary : ");

scanf("%f",&sal);

h=insert(sname,id1,sal,h);

display(h);

break;

}

case 2:{

priorityelement=deletemax(h);

display2(priorityelement);

display(h);

break;

}

case 3:display(h);break;

case 4:return 0;

default:printf("Invalid Input!\n");

}

printf("\nMenu\n1.Insert Employee Details.\n2.Dequeue Highest Salary Employee.\n3.Display Tree.\n4.Exit.\nEnter your choice : ");

scanf("%d",&ch);

}

return 0;

}

**//Sample I/O:**

Enter the number of employees : 5

Enter the 1th Employee details!

Enter Name : rahul

Enter ID : 123

Enter salary : 56789

Displaying Tree!!!

1 : rahul 123 56789.00

Enter the 2th Employee details!

Enter Name : raghu

Enter ID : 124

Enter salary : 67890

Displaying Tree!!!

1 : raghu 124 67890.00

2 : rahul 123 56789.00

Enter the 3th Employee details!

Enter Name : prithvi

Enter ID : 125

Enter salary : 76544

Displaying Tree!!!

1 : prithvi 125 76544.00

2 : rahul 123 56789.00

3 : raghu 124 67890.00

Enter the 4th Employee details!

Enter Name : prakash

Enter ID : 126

Enter salary : 66666

Displaying Tree!!!

1 : prithvi 125 76544.00

2 : prakash 126 66666.00

3 : raghu 124 67890.00

4 : rahul 123 56789.00

Enter the 5th Employee details!

Enter Name : krishna

Enter ID : 127

Enter salary : 87654

Displaying Tree!!!

1 : krishna 127 87654.00

2 : prithvi 125 76544.00

3 : raghu 124 67890.00

4 : rahul 123 56789.00

5 : prakash 126 66666.00

Menu

1.Insert Employee Details.

2.Dequeue Highest Salary Employee.

3.Display Tree.

4.Exit.

Enter your choice : 2

Displaying Highest Salary Employee!!!

Name : krishna

ID : 127

Salary : 87654.00

Displaying Tree!!!

1 : prithvi 125 76544.00

2 : prakash 126 66666.00

3 : raghu 124 67890.00

4 : rahul 123 56789.00

Menu

1.Insert Employee Details.

2.Dequeue Highest Salary Employee.

3.Display Tree.

4.Exit.

Enter your choice : 4