**Nimra**

**199281**

**BESE7B**

LAB TASKS

#include <iostream>

using namespace std;

class HEAP{

public:

int A[100];

int n;

HEAP(){

n = 0;

}

void MIN\_HEAPIFY(int i){

int small;

int l,r;

r=(2\*i)+2;

l=(2\*i)+1;

if (l<=n && A[l]<A[i]){

small=l;

}

else{

small=i;

}

if (r<=n && A[r]<A[small]){

small=r;

}

if (small!=i){

SWAP(i,small);

MIN\_HEAPIFY(small);

}

}

void BUILD\_MIN\_HEAP(){

for(int i=(n/2);i>=0;i--){

MIN\_HEAPIFY(i);

}

}

int TOP(){

if (n!=0){

return A[0];

}

}

void PUSH(int a){

n=n+1;

A[n-1]=a;

BUILD\_MIN\_HEAP();

}

void POP(){

if (n!=0){

A[0]=A[n-1];

n=n-1;

BUILD\_MIN\_HEAP();

}

}

bool isEMPTY(){

if (n==0){

return true;

}

else{

return false;

}

}

void PRINT(){

int level = 0;

for(int i=0;i<=n -1;i++){

if ( level == i){

cout<< "\n";

level =level \* 2+1;

}

cout<<A[i] << " ";

}

cout << "\n";

}

int HEIGHT(int i=0)

{

int track=n;

if (n==0 )

return -1;

else if((2\*i)+1>n || (2\*i)+2>n){

return 0;

}

else

{

int leftDepth = HEIGHT((2\*i)+1);

int rightDepth = HEIGHT((2\*i)+2);

if (leftDepth > rightDepth)

return(leftDepth+1);

else return(rightDepth+1);

}

}

int SIZE(){

return n;

}

private:

void SWAP(int a,int b){

int c;

c=A[a];

A[a]=A[b];

A[b]=c;

}

};

int main()

{

HEAP a;

cout<<"isEmpty boolean = "<<a.isEMPTY()<<endl;

cout<<"Initializing the array"<<endl;

for (int i=0;i<=7;i++){

a.PUSH(rand() % 100);

}

cout<<"applying build min heap on the array here"<<endl;

a.BUILD\_MIN\_HEAP();

a.PRINT();

cout<<"Popped the root element"<<endl;

a.POP();

a.PRINT();

cout<<"isEmpty boolean = "<<a.isEMPTY()<<endl;

cout<<"Height = "<<a.HEIGHT()<<endl;

cout<<"Size = "<<a.SIZE()<<endl;

cout<<"TOP = "<<a.TOP()<<endl;

system("Pause");

}