**Heap Sort:**

#include<iostream>

#include<conio.h>

using namespace std;

class BinaryMinHeap{

public:

int \*data;

int heapS;

int arrS;

int gtLeftChild(int indN){

return (2 \* indN) + 1;

}

int gtRightChild(int indN){

return (2 \* indN) + 2;

}

int getPind(int indN){

return (indN - 1) / 2;

}

BinaryMinHeap(int size){

data = new int[size];

heapS = 0;

arrS = size;

}

int getMinimum(){

if(isEmpty() == true){

cout << "\nHeap is Empty.";

}else{

return data[0];

}

return -1;

}

void display(){

if(isEmpty() == true){

cout << "\nHeap is Empty.";

}

else{

cout << "\nElements are Present in Heap are ";

for(int i = 0; i < heapS; i++)

cout << data[i] << " ";

}

}

void insert(int val){

cout << "\nYou want to insert this value " << val << " to Heap.";

if(heapS == arrS){

cout << "\nSorry to Inform you Heap is Full.";

}

else{

data[heapS] = val;

heapS++;

reheapUp(heapS - 1);

}

}

void reheapUp(int nodI){

int parN, temp;

if(nodI != 0){

parN = getPind(nodI);

if(data[parN] > data[nodI]){

temp = data[parN];

data[parN] = data[nodI];

data[nodI] = temp;

reheapUp(parN);

}

}

}

bool isEmpty(){

if(heapS == 0){

return true;

}

return false;

}

void remove(){

if(isEmpty() == true){

cout << "\nHeap is Empty.";

}

else{

cout << "\nRemoved Element is" << data[0] << " from Heap.";

data[0] = data[heapS - 1];

heapS--;

reheapDown(0);

}

}

void reheapDown(int nodI){

int lc,rc,mInd,temp;

lc = gtLeftChild(nodI);

rc = gtRightChild(nodI);

if(rc >= heapS){

if(lc >= heapS){

return;

}else{

mInd = lc;

}

}else{

if(data[lc] < data[rc]){

mInd = lc;

}else{

mInd = rc;

}

}

if(data[mInd] < data[nodI]){

temp = data[mInd];

data[mInd] = data[nodI];

data[nodI] = temp;

}

}

void sort(){

cout << "\nSorted Heap:";

int oheapS = heapS;

int sortArr[heapS], i;

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

sortArr[i] = data[0];

cout << sortArr[i] << " ";

remove();

}

}

void checkSpace()

{

if(heapS==0)

{

cout << "\nHeap is Empty.";

}

else if(heapS == arrS)

{

cout << "\nSorry to Inform you Heap is Full.";

}

else

{

int k = arrS - heapS;

cout << "\nYou can add " << k << " more Elements in the given heap with their size is " << arrS;

}

}

};

int main(){

int size;

int k;

cout << "\nTo Create Heap Press 1:";

cin >> k;

if (k != 1){

return 0;

}

cout << "\nEnter size of Heap: ";

cin >> size;

BinaryMinHeap bn(size);

int ch, p;

cout << "1) Insert element to Heap: " << endl;

cout << "2) Delete element from Heap: " << endl;

cout << "3) Display all the elements of Heap:" << endl;

cout << "4) Display all the elements in Sorted form in Heap: " << endl;

cout << "5) Check Available Space in the Heap: " << endl;

cout << "6) Display Minimum Element in the Heap: " << endl;

cout << "7) Exit" << endl;

do {

cout << "\nEnter your choice : " << endl;

cin >> ch;

switch (ch) {

case 1:

cout << "\nEnter Element you Want insert in the Heap : ";

cin >> p;

bn.insert(p);

break;

case 2:

cout << "\nBefore Element removed: ";

bn.display();

bn.remove();

cout << "\nAfter Element removed: ";

bn.display();

break;

case 3:

cout << "\nDisplay Elements in the Heap: ";

bn.display();

break;

case 4:

cout << "\nDisplay Sorted Heap: ";

bn.sort();

break;

case 5:

cout << "\nAvailable Space in the Heap is ";

bn.checkSpace();

break;

case 6:

cout << "\nDisplay Minimum Element in Min Heap: " << bn.getMinimum();

break;

case 7:

exit(0);

default:

cout << "Invalid choice" << endl;

}

} while (ch != 7);

return 0;

}

**Output:**







