ASSIGNMENT:6

AIM :

Read the marks obtained by the students of second year in an online examination of a particular subject. Find out maximum and minimum marks obtained in that subject using heap data structure.

OBJECTIVE :

To study and learn the concepts of heap data structure.

THEORY :

Heap definition- It is a Complete (Binary) Tree with each node having HEAP PROPERTY. Elements are filled level by level from left- to-right. If A is a parent node of B, then the key (the value) of node A is ordered with respect to the key of node B with the same ordering applying across the heap.

Types of heap: 1) Min heap

2) Max heap

- MAX HEAP definition:
  - Complete (Binary) tree with the property that the **value of each node** is at least as large as the value of its children (i.e. >= value of its children)
- MIN HEAP definition:
  - Complete (Binary) tree with the property that the **value of each node** is at most as large as the value of its children (i.e. <= value of its children)

ALGORITHM :

To maintain the max heap property i.e. MAXHEAPIFY

MAX-HEAPIFY(A, i, n)

1. l ← LEFT(i)
2. r ← RIGHT(i)
3. **if** l ≤ n and A[l] > A[i]
4. **then** largest ←l
5. **else** largest ←i
6. **if** r ≤ n and A[r] > A[largest]
7. **then** largest ←r
8. **if** largest ≠ i
9. **then** exchange A[i] ↔ A[largest]
10. MAX-HEAPIFY(A, largest, n)

PROGRAM :

```
#include <iostream>

using namespace std;

class Heap{
private :
    int * max_heap;
    int * min_heap;
    int capacity, currEmptyPos, currEmptyPosMin;
public :
    Heap(int capacity){
        currEmptyPos = 0;
        currEmptyPosMin = 0;
        this->capacity = capacity;
```

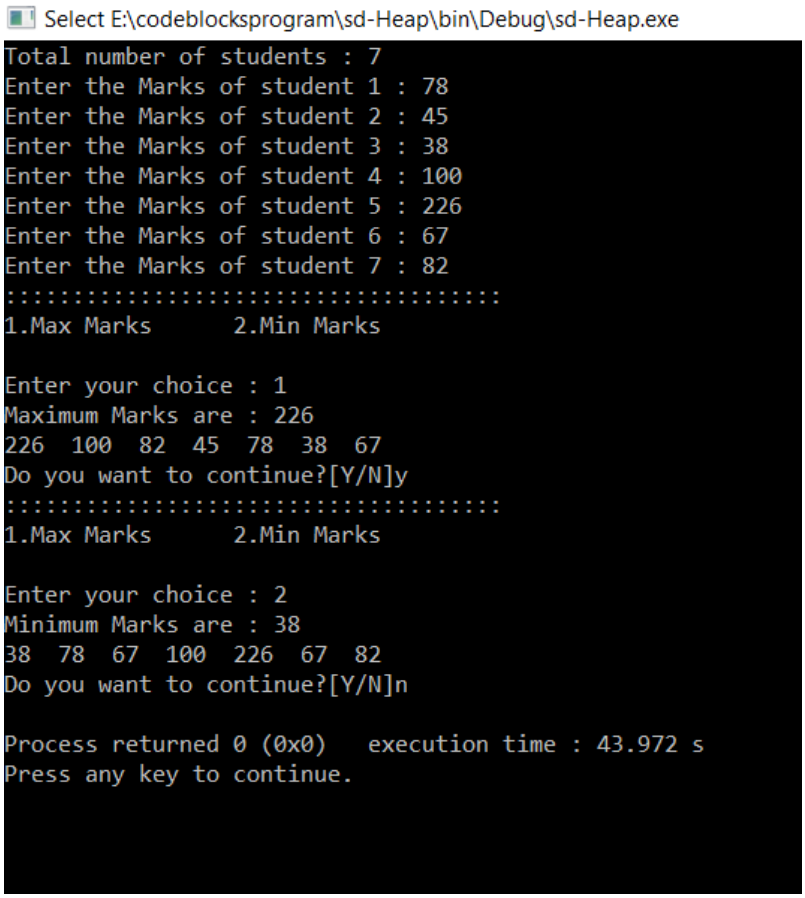
```
        max_heap = new int[capacity];
        min_heap = new int[capacity];
        for(int i=0; i<capacity; i++){
            max_heap[i] = 0;
            min_heap[i] =9999;
        }
    }
    void max_heapify(int pos){

        //<<"called."<<pos<<endl;
        int left = (pos*2)+1;
        int right = (pos*2)+2;
        int largest = pos;
        if( pos < ((currEmptyPos+1)/2)){
            if(max_heap[left]>max_heap[largest]){
            largest = left;
            }
            if(max_heap[right]>max_heap[largest]){
                largest = right;
            }
        }
        if(largest == pos){
            return;
        }
        else{
            int temp = max_heap[largest];
            max_heap[largest] = max_heap[pos];
            max_heap[pos] = temp;
            max_heapify((pos-1)/2);
        }
    }
```

```
void min_heapify(int pos){

    //cout<<"called."<<pos<<endl;
    int left = (pos*2)+1;
    int right = (pos*2)+2;
    int smallest = pos;
    if( pos < ((currEmptyPosMin+1)/2)){
        if(min_heap[left]<min_heap[smallest]){
            smallest = left;
        }
        if(min_heap[right]<min_heap[smallest]){
            smallest = right;
        }
    }
    if(smallest == pos){
        return;
    }
    else{
        int temp = min_heap[smallest];
        min_heap[smallest] = max_heap[pos];
        min_heap[pos] = temp;
        min_heapify((pos-1)/2);
    }
}

void insertIntoHeap(int val){
    //cout<<capacity<<endl;
    if(currEmptyPos< capacity){
        max_heap[currEmptyPos] = val;
        max_heapify((currEmptyPos-1)/2);
        min_heap[currEmptyPosMin] = val;
        min_heapify((currEmptyPosMin-1)/2);
```

```
            currEmptyPos++;
            currEmptyPosMin++;
        }
    }

    int getMaxTop(){return max_heap[0];}
    int getMinTop(){return min_heap[0];}

    void print_max_heap(){
        for(int i = 0;i<capacity;i++){
            cout<<max_heap[i]<<"  ";
        }
        cout<<endl;
    }
    void print_min_heap(){
        for(int i = 0;i<capacity;i++){
            cout<<min_heap[i]<<"  ";
        }
        cout<<endl;
    }
};

void setStudentsMarks(){
    cout<<"Total number of students : ";
    int noOfStu;
    cin>>noOfStu;
    Heap stuMarks(noOfStu);
    for(int i = 0; i < noOfStu ; i++){
        cout<<"Enter the Marks of student "<<i+1<<" : ";
        int marks;
        cin>>marks;
        stuMarks.insertIntoHeap(marks);
```

```
    }
    char ch;
    do{
        cout<<"::::::::::::::::::::::::::::::::::::"<<endl;
        cout<<"1.Max Marks"<<"      "<<"2.Min Marks"<<endl;
        cout<<endl<<"Enter your choice : ";
        int choice;
        cin>>choice;
        if(choice == 1){
            cout<<"Maximum Marks are :
"<<stuMarks.getMaxTop()<<endl;
            stuMarks.print_max_heap();

        }
        else{
            cout<<"Minimum Marks are :
"<<stuMarks.getMinTop()<<endl;
            stuMarks.print_min_heap();
        }

        cout<<"Do you want to continue?[Y/N]";
        cin>>ch;
    }while(ch=='y' || ch=='Y');

}

int main()
{
    setStudentsMarks();
    return 0;
}
```

OUTPUT:



CONCLUSION:

We successfully implemented heap data structure.