ASSIGNMENT 6

AIM:-

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

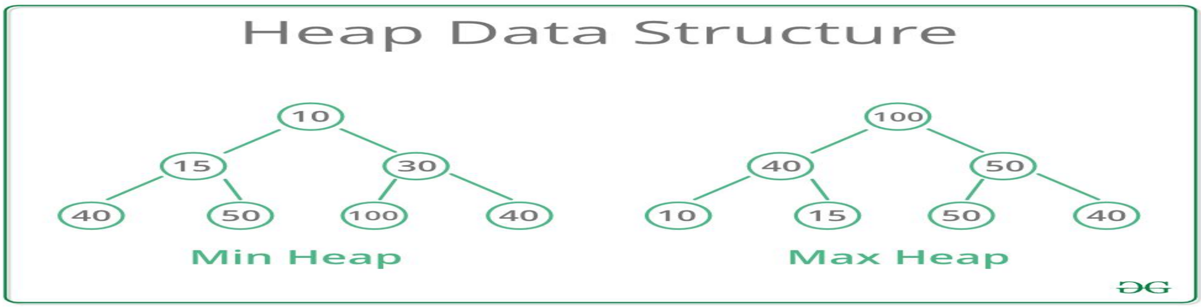
OBJECTIVE:-

Implement heap data structure and function associated with it like build a heap,heap sort, heapify and determine max and min marks.

THEORY:-

A Heap is a special Tree-based data structure in which the tree is a complete binary tree. Generally, Heaps can be of two types:

1. **Max-Heap**: In a Max-Heap the key present at the root node must be greatest among the keys present at all of it's children. The same property must be recursively true for all sub-trees in that Binary Tree.
2. **Min-Heap**: In a Min-Heap the key present at the root node must be minimum among the keys present at all of it's children. The same property must be recursively true for all sub-trees in that Binary Tree.



Heap Sort

Heap sort algorithm is divided into two basic parts:

- Creating a Heap of the unsorted list/array.
- Then a sorted array is created by repeatedly removing the largest/smallest element from the heap, and inserting it into the array. The heap is reconstructed after each removal.

Initially on receiving an unsorted list, the first step in heap sort is to create a Heap data structure(Max-Heap or Min-Heap). Once heap is built, the first element of the Heap is either largest or smallest(depending upon Max-Heap or Min-Heap), so we put the first element of the heap in our array. Then we again make heap using the remaining elements, to again pick the first element of the heap and put it into the array. We keep on doing the same repeatedly untill we have the complete sorted list in our array.

ALGORITHM:-

HEAP SORT

```
void heapsort(int a[], int n)
   {
     for(int i=(n/2)-1; i>=0;i--)
     {
       heapify(a,n,i);
     }
     for(int i=(n-1);i>=0;i--)
     {
         int temp= a[0];
         a[0]= a[i];
         a[i]= temp;
       heapify (a,i,0);
       }
     }
```

HEAPIFY:

```
void heapify(int a[],int n, int i)
{
         int largest=i;
         int l= (2*i)+1;
         int r=(2*i)+2;
         if(l<n && a[l]>a[largest])
         largest=l;
         if(r<n && a[r]>a[largest])
         largest=r;

         if(largest!=i)
         {
         int t= a[i];
         a[i]=a[largest];
         a[largest]=t;
         heapify(a,n,largest);
```

```
     }
}
```

DISPLAY ARRAY

```
void    printarray(int a[],int n)
{
       for(int i=0;i<n;i++)
             {
               cout<<a[i]<<"";
               cout<<"\n";
               }
               }
```

MAX VALUE

```
               voidmaximum(int a[],int n)
               {
                      cout<<"MAXIMUM MARKS:"<<a[n-1]<<endl;
               }
```

MINIMUM VALUE

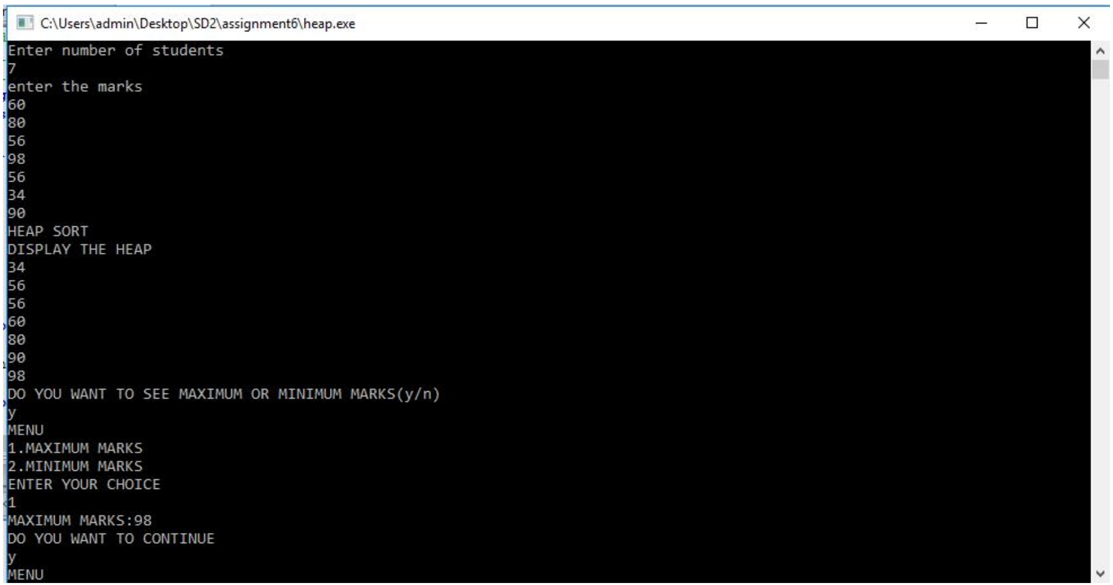
```
               void minimum(int a[],int n)
               {
                      cout<<"MINIMUM MARKS:"<<a[0]<<endl;
               }
```

CODE:-

```
#include<iostream>
using namespace std;
class heap
{
public:
void printarray(int a[], int n);
void heapsort(int a[], int n);
void minimum(int a[],int n);
void maximum(int a[],int n);
};
void heapify(int a[],int n,int i);
void heap:: heapsort(int a[], int n)
   {
     for(int i=(n/2)-1; i>=0;i--)
     {
       heapify(a,n,i);
```

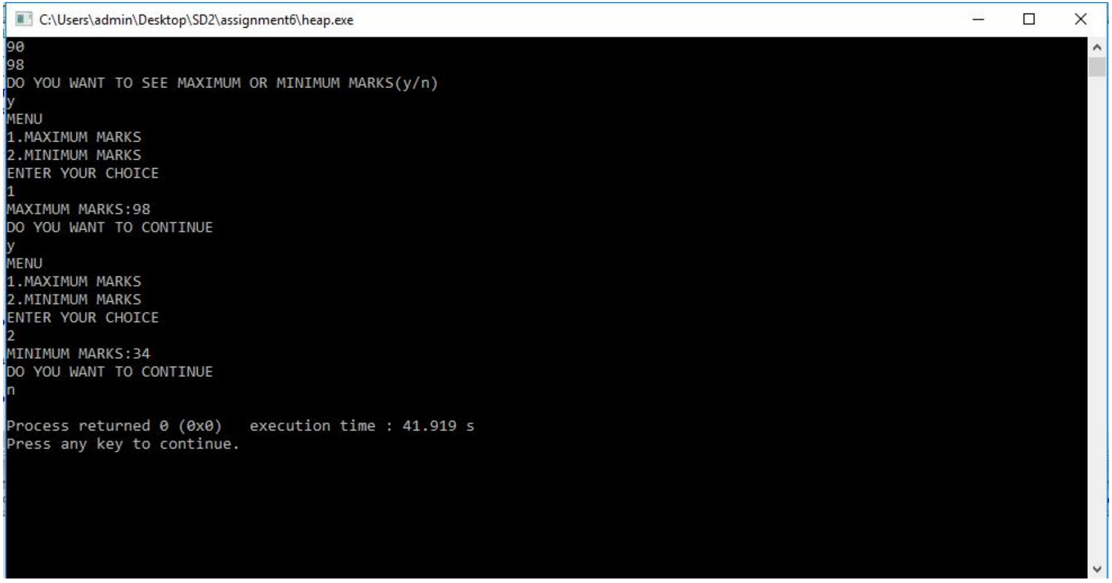
```
      }
      for(int i=(n-1);i>=0;i--)
      {
          int temp= a[0];
          a[0]= a[i];
          a[i]= temp;
        heapify (a,i,0);
        }
      }
void heapify(int a[],int n, int i)
{
          int largest=i;
          int l= (2*i)+1;
          int r=(2*i)+2;
          if(l<n && a[l]>a[largest])
          largest=l;
          if(r<n && a[r]>a[largest])
          largest=r;

          if(largest!=i)
          {
          int t= a[i];
          a[i]=a[largest];
          a[largest]=t;
          heapify(a,n,largest);
          }
}
void heap:: printarray(int a[],int n)
{
        for(int i=0;i<n;i++)
              {
                cout<<a[i]<<"";
                cout<<"\n";
                }
                }
                void heap::maximum(int a[],int n)
                {
                        cout<<"MAXIMUM MARKS:"<<a[n-1]<<endl;
                }
                void heap::minimum(int a[],int n)
                {
                        cout<<"MINIMUM MARKS:"<<a[0]<<endl;
                }
int main()
{
    heap h;
    int a[100],n;
    cout<<"Enter number of students"<<endl;
    cin>>n;
    cout<<"enter the marks"<<endl;
    for(int i=0;i<n;i++)
        {
        cin>>a[i];
        }
        cout<<"HEAP SORT"<<endl;
        h.heapsort(a,n);
        cout<<"DISPLAY THE HEAP"<<endl;
        h.printarray(a,n);
        char ch;
        int choice;
        cout<<"DO YOU WANT TO SEE MAXIMUM OR MINIMUM
MARKS(y/n)"<<endl;
        cin>>ch;
        while(ch=='y')
        {
        cout<<"MENU"<<endl;
        cout<<"1.MAXIMUM MARKS"<<endl;
        cout<<"2.MINIMUM MARKS"<<endl;
        cout<<"ENTER YOUR CHOICE"<<endl;
        cin>>choice;
        switch(choice)
                {
                case 1:
                        h.maximum(a,n);
                        break;
                case 2:
                        h.minimum(a,n);
                        break;
                default:
                        cout<<"SORRY!WRONG CHOICE"<<endl;
                        break;
                }
                cout<<"DO YOU WANT TO CONTINUE"<<endl;
                cin>>ch;
        }
return 0;
    }
```

OUTPUT:-



C:\Users\admin\Desktop\SD2\assignment6\heap.exe

```
Enter number of students
7
enter the marks
60
80
56
98
56
34
90
HEAP SORT
DISPLAY THE HEAP
34
56
56
60
80
90
98
DO YOU WANT TO SEE MAXIMUM OR MINIMUM MARKS(y/n)
y
MENU
1.MAXIMUM MARKS
2.MINIMUM MARKS
ENTER YOUR CHOICE
1
MAXIMUM MARKS:98
DO YOU WANT TO CONTINUE
y
MENU
```



C:\Users\admin\Desktop\SD2\assignment6\heap.exe

```
90
98
DO YOU WANT TO SEE MAXIMUM OR MINIMUM MARKS(y/n)
y
MENU
1.MAXIMUM MARKS
2.MINIMUM MARKS
ENTER YOUR CHOICE
1
MAXIMUM MARKS:98
DO YOU WANT TO CONTINUE
y
MENU
1.MAXIMUM MARKS
2.MINIMUM MARKS
ENTER YOUR CHOICE
2
MINIMUM MARKS:34
DO YOU WANT TO CONTINUE
n

Process returned 0 (0x0)   execution time : 41.919 s
Press any key to continue.
```

CONCLUSION:-

We have successfully Implemented heap data structure and function associated with it like build a heap,heap sort, heapify,etc.