

.vscode\dsa\Heaps_implement.cpp

```
1  #include<iostream>
2  #include<queue>
3  using namespace std ;
4
5  class heap{ // max heap
6      public :
7          int arr[100] ;
8          int size ;
9
10         heap() // default constructor
11         {
12             arr[0] = -1 ;
13             size = 0 ;
14         }
15
16         void insert(int val) // T.C. = O(logn)
17         {
18             size = size+1 ;
19             int index = size ;
20             arr[index] = val ;
21
22             while(index > 1)
23             {
24                 int parent = index/2 ;
25                 if(arr[parent] < arr[index])
26                 {
27                     swap(arr[parent],arr[index]) ;
28                     index = parent ;
29                 }
30                 else{
31                     return ;
32                 }
33             }
34         }
35
36         void print()
37         {
38             for(int i=1; i<=size; i++)
39             {
40                 cout<<arr[i]<<" " ;
41             }cout<<endl;
42         }
43
44         void deleteFromHeap() // T.C. = O(logn)
45         // ham root node ko delete krte hai
46         {
47             if(size == 0)
48             {
49                 cout<<"nothing to delete"<<endl;
50                 return ;
51             }
```

```
52
53     arr[1] = arr[size] ; // root node and last node ko swap krdiya ya value replace kr di
54     size-- ; // remove last element
55
56     // ab first node jo new bani hai usko shi jagah par pahunchana hai
57     int i = 1 ;
58     while(i < size)
59     {
60         int leftIndex = 2*i ;
61         int rightIndex = 2*i+1 ;
62
63         if(leftIndex < size && arr[i] < arr[leftIndex])
64         {
65             swap(arr[i],arr[leftIndex]) ;
66             i = leftIndex ;
67         }
68         else if(rightIndex < size && arr[i] < arr[rightIndex])
69         {
70             swap(arr[i],arr[rightIndex]) ;
71             i = rightIndex ;
72         }
73         else{
74             return ;
75         }
76     }
77 }
78 };
79
80 void heapify(int arr[], int n, int i) // T.C. = O(logn)
81 {
82     int largest = i ;
83     int left = 2*i ;
84     int right = 2*i+1 ;
85
86     if(left <= n && arr[largest]<arr[left])
87     {
88         largest = left ;
89     }
90     if(right <= n && arr[largest]<arr[right])
91     {
92         largest = right ;
93     }
94     if(largest != i)
95     {
96         swap(arr[largest],arr[i]) ;
97         heapify(arr,n,largest) ;
98     }
99 }
100
101 void heapSort(int arr[], int n) // T.C. = O(nlogn)
102 {
103     int size = n ;
104     while(size > 1)
105     {
```

```
106         swap(arr[size],arr[1]) ;
107         size-- ;
108
109         heapify(arr,size,1) ;
110     }
111 }
112
113 int main()
114 {
115     heap h ;
116     h.insert(50) ;
117     h.insert(55) ;
118     h.insert(53) ;
119     h.insert(52) ;
120     h.insert(54) ;
121     h.print() ;
122
123     h.deleteFromHeap() ;
124     h.print() ;
125
126     int arr[6] = {-1,54,53,55,52,50} ;
127     int n=5 ;
128     for(int i=n/2;i>0;i--)
129     {
130         heapify(arr,n,i) ;
131     }
132     cout<<"printing the array now = "<<endl;
133     for(int i=1;i<=n;i++)
134     {
135         cout<<arr[i]<<" " ;
136     }cout<<endl ;
137
138     heapSort(arr,n) ;
139     cout<<"printing sorted array = "<<endl;
140     for(int i=1;i<=n;i++)
141     {
142         cout<<arr[i]<<" " ;
143     }cout<<endl ;
144
145
146     cout<<"using priority queue here - "<<endl ;
147     priority_queue<int> pq ; // max heap
148
149     pq.push(4) ;
150     pq.push(2) ;
151     pq.push(5) ;
152     pq.push(3) ;
153     cout<<"element at top - "<<pq.top()<<endl ;
154     pq.pop() ;
155     cout<<"element at top - "<<pq.top()<<endl ;
156     cout<<"size = "<<pq.size()<<endl;
157
158
159
```

```
160     // min heap
161     priority_queue<int, vector<int>, greater<int> > minHeap ;
162
163
164     return 0 ;
165 }
```