KATHMANDU UNIVERSITY

Department of Computer Science and Engineering

Dulikhel, Kavre

Mini Project

[Code No: COMP 206]

Submitted By:

Rajaram Karki (Roll No: 23)

Submitted To:

Department of Computer Science and Engineering

Rajani Chulyadyo (Ma'am)

In the program 5, 4, 10, 16, 2, 11, 1 are inserted in order.
They are stored in the vector as following

5						

4	5					

4	5	10				

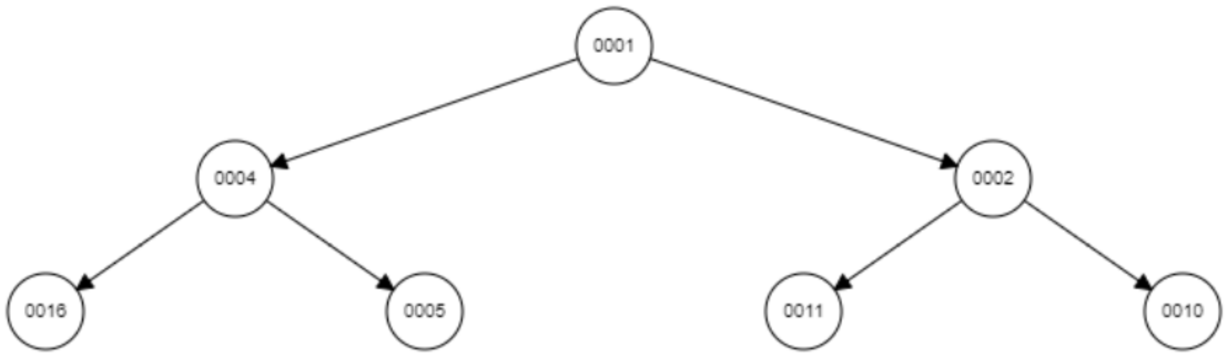
4	5	10	16			

2	4	10	16	5		

2	4	10	16	5	11	

1	4	2	16	5	11	10

The heap becomes



Since the data are stored starting from 0th index, the left child of an index i becomes (2*i + 1) and similarly right child index becomes (2*i + 2). The parent of an index i becomes (i-1)/2.

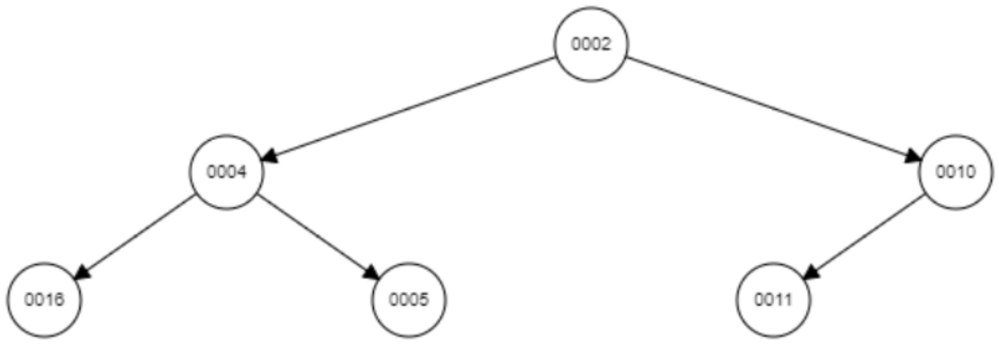
While inserting elements into the heap, if the inserted element is lower than its parent's element than we swap the two.

Upon deleting the min-element (root node), the last element takes the place on root then, it is compared with its lowest children (among the right and left children) then swapped if it is lower.

Upon deleting minimum element i.e. 1

2	4	10	16	5	11	

The heap becomes



In this way we can implement ascending priority queue using min heap. The element with lowest priority (i.e., minimum element) is removed. Insert an element with arbitrary priority. And return an element with minimum priority.

Time Complexity of the program

For the time complexity of heap implementation, we have the following major operations

InsertKey – The time complexity of inserting an element to the heap is O(1) for the best case when the inserted element doesn't violate the heap property. If it does violate the heap property, reheap_up must be called which makes the time complexity O(logn).

DeleteKey – DeleteKey removes the minimum element from the node. Doing so violates the heap property, then calling reheap_down makes the time complexity O(logn).

Top – Top returns the minimum element/root in the heap, so it completes in O(1).

Display() – For display, it traverse through the whole vector so, it's time complexity is O(n) where n is the number of elements in the heap.

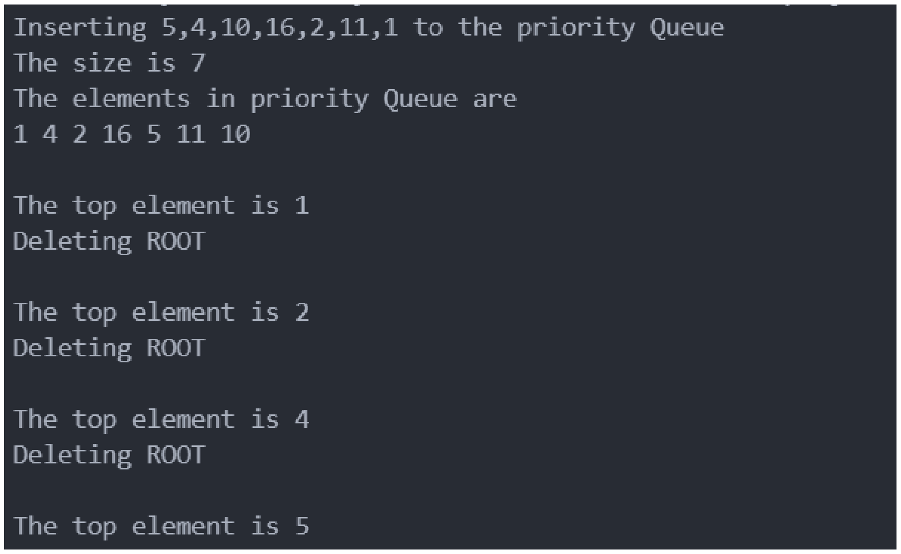
So, the overall time complexity of the program becomes O(n).

Similarly for priority queue

For enqueue, InsertKey is used with time complexity O(logn)
For dequeue, DeleteKey is used with time complexity O(logn)
And for top/peek, top is used with time complexity O(1)
The same display function is used with time complexity O(n).

So, the overall time complexity of the program becomes O(n).

Output of Program



```
Inserting 5,4,10,16,2,11,1 to the priority Queue
The size is 7
The elements in priority Queue are
1 4 2 16 5 11 10

The top element is 1
Deleting ROOT

The top element is 2
Deleting ROOT

The top element is 4
Deleting ROOT

The top element is 5
```

GitHub Repository