Priority queue - User's priority matters

Why do we need priority queues?

- ***On shopping websites, every user has different priorities like price, color, size, etc.***
- ***We need some data structure that can sort products on the basis of the user's priority***
- ***This data structure should take less time as we want to provide user results as fast as possible.***

Possible Candidates for making priority queue :

Data structure	push()	front()	pop()
Heaps	O(log(n))	O(log(n))	O(log(n))
Arrays(unsorted)	O(1)	O(n)	O(n)
Arrays(sorted)	O(n)	O(1)	O(1)
Linked list(unsorted)	O(1)	O(n)	O(n)
Linked list(sorted)	O(n)	O(1)	O(1)
BST	O(n)	O(n)	O(n)
Hashmap	O(1)	O(n)	O(n)

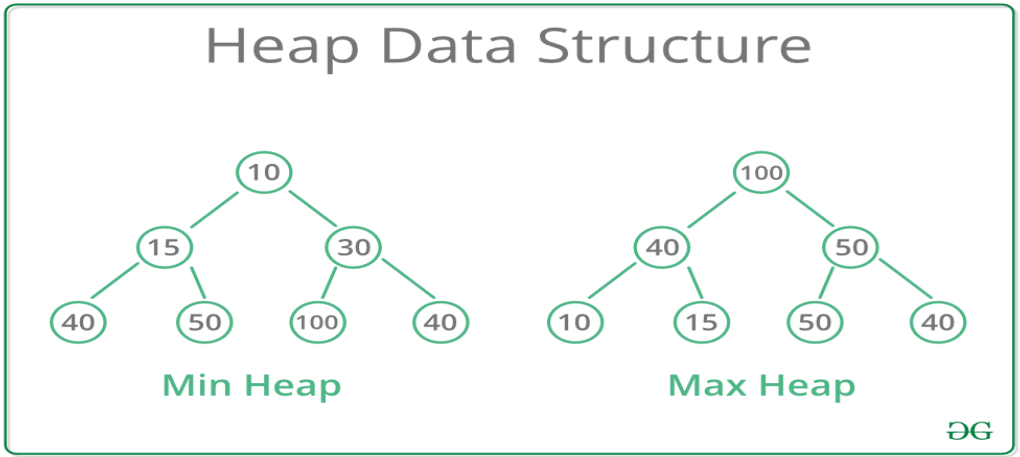
What is heap?

A Heap is a special Tree-based data structure in which the tree is a complete binary tree.

2 major properties:

1) Complete Binary tree

2) Heap order property



Complete Binary Tree

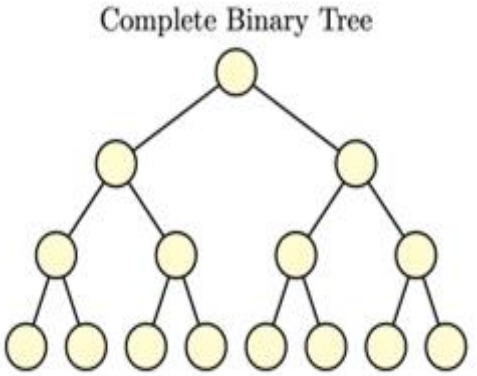
- For a tree to be CBT, all the positions except the last level should be completely filled.
- Also, the last level should be filled from left to right

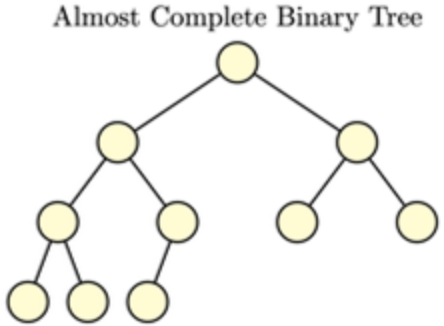
-> Using general maths to calculate the height of CBT for n nodes

- After solving various equations the result was:-

$$O(\log_2(n)) <= h <= O(\log_2(n))$$

- Where h is the height of CBT and n is the number of nodes in CBT





Heap order property :

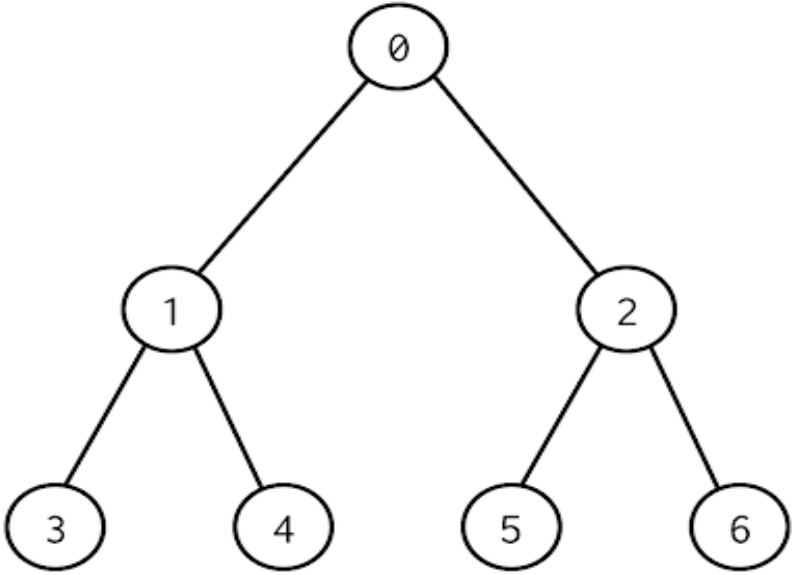
- ***For a min-heap, the parent should always be smaller than both of it's child elements. The same property must be recursively true for all subtrees in that Binary Tree.***

- ***For a max-heap, the parent should always be greater than both of its child elements. The same property must be recursively true for all subtrees in that Binary Tree.***

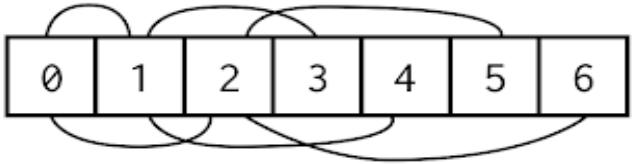
Internal working:

-> Storing the tree in the form of an array.

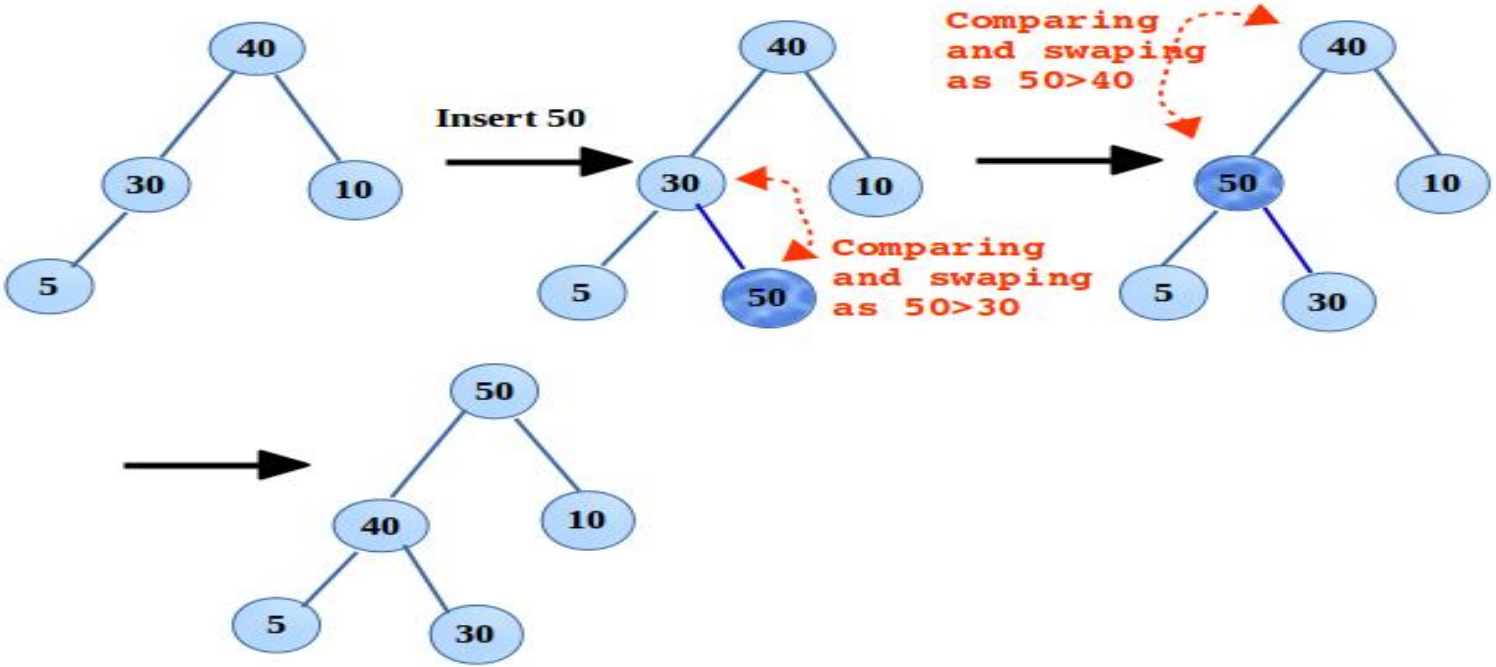
- As traversing whole tree was costing a lot of time.
- So, we store the tree in linear form using array.
- Left Child :- 2 * i + 1
- Right Child :- 2 * i + 2



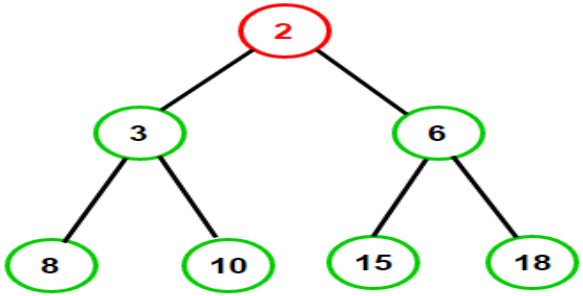
a[i] has children a[2*i+1] and a[2*i+2]



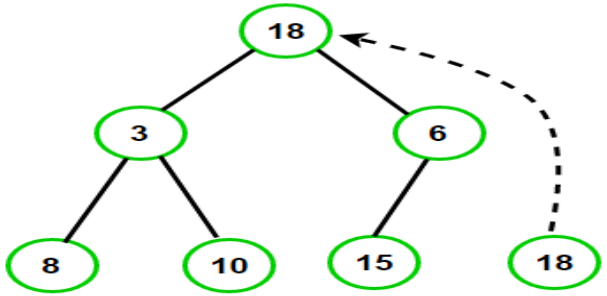
push () or up heapify



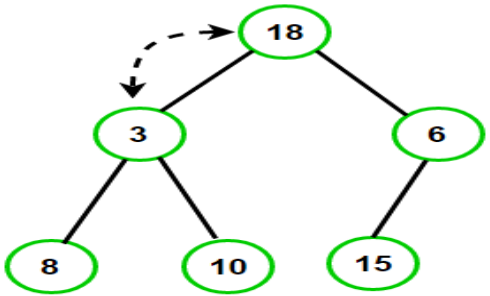
pop () or down heapify



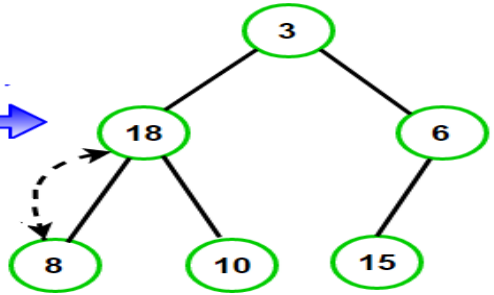
Pop() called on min heap



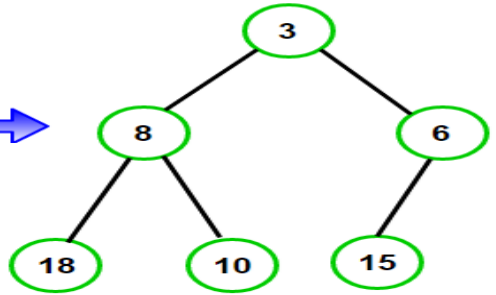
Replace the root of the heap with the last element on the last level and call
Heapify-down(root)



Swap root node with its smaller child
swap(18, min(3, 6))



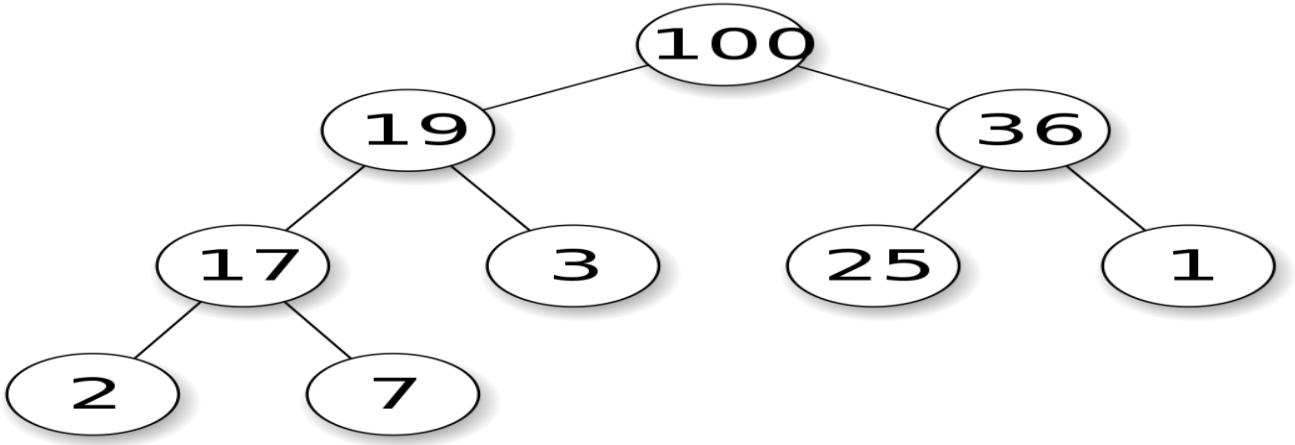
Swap node 18 with its smaller child
swap(18, min(8, 10))



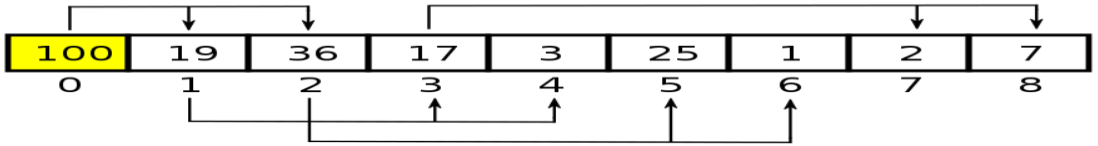
Resultant Min Heap

front ()

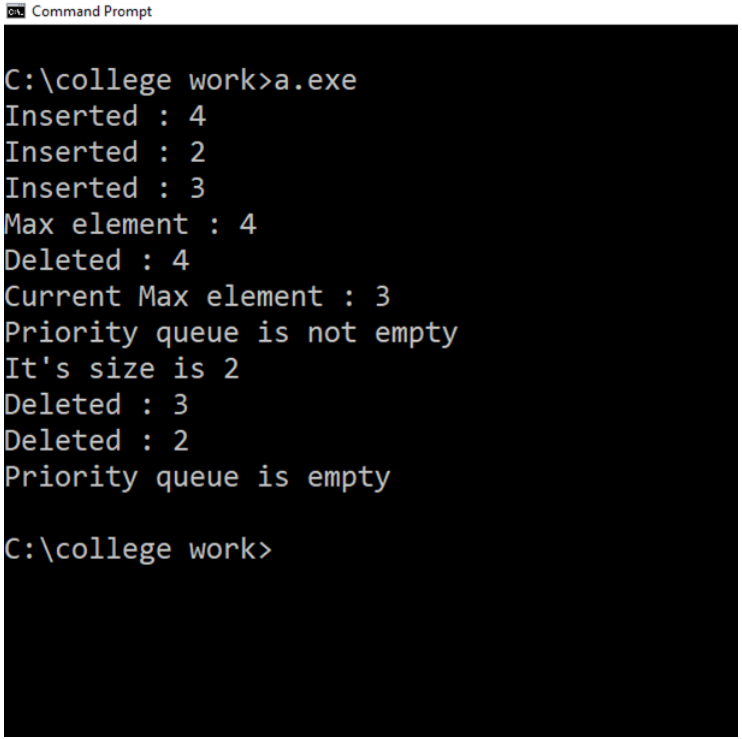
Tree representation



Array representation



Demo


Command Prompt
C:\college work>a.exe
Inserted : 4
Inserted : 2
Inserted : 3
Max element : 4
Deleted : 4
Current Max element : 3
Priority queue is not empty
It's size is 2
Deleted : 3
Deleted : 2
Priority queue is empty
C:\college work>

Thank you!