Heap sort

In heap sort basically, there are two phases involved in the sorting of elements

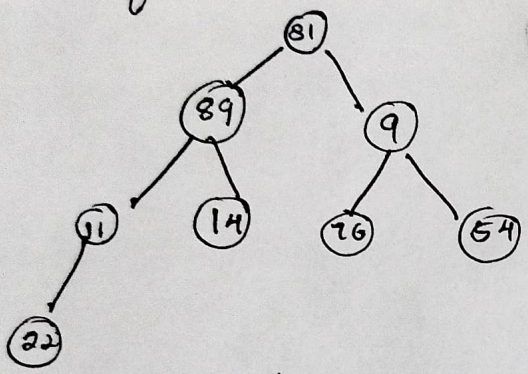
By using the heap sort algorithm,

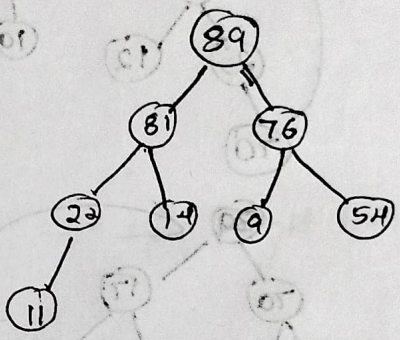
first step : creation of a heap by adjusting the elements of the array

after the creation of heap, now remove the root element of the heap repeatedly

81	89	9	11	14	76	54	22

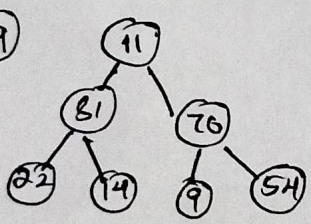
first we have to construct a heap from the given array and convert it into max heap

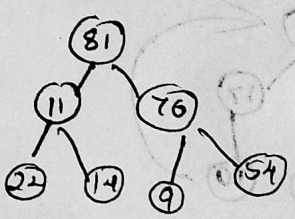




89	81	76	22	14	9	54

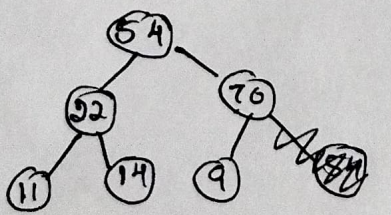
89

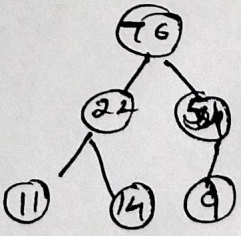


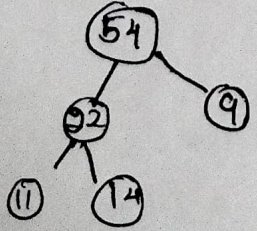


54 | 76 | 81 | 89

81	11	76	22	14	9	54

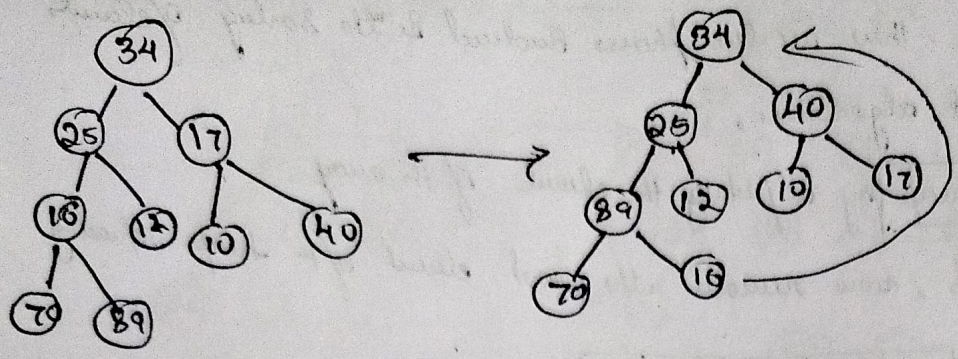




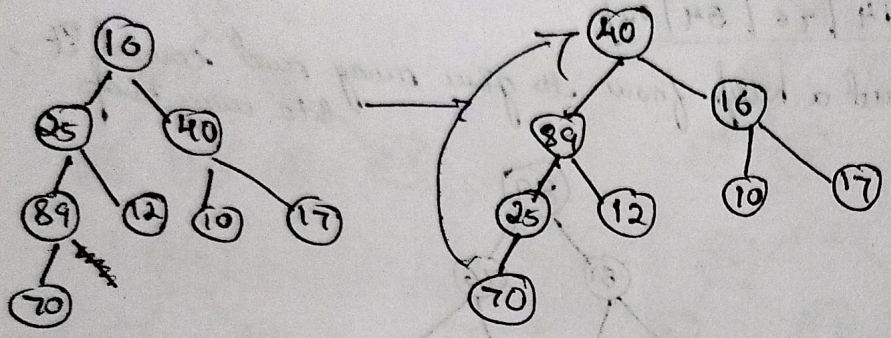


22

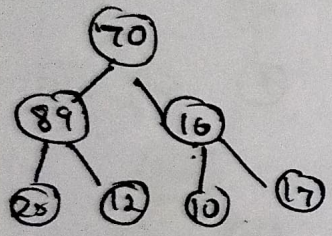
34	25	17	16	12	10	40	70	89

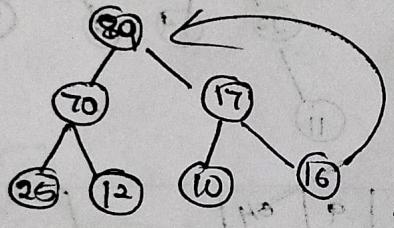


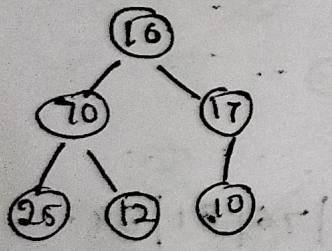
34

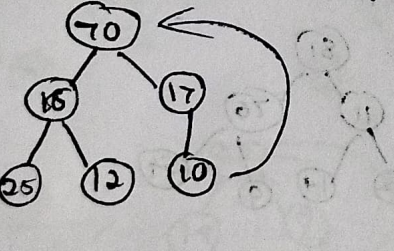


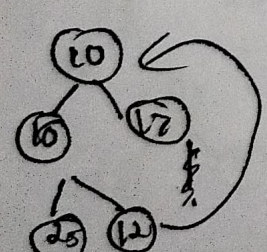
70 | 89 | 40 | 34

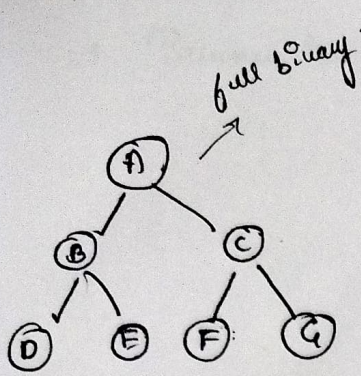


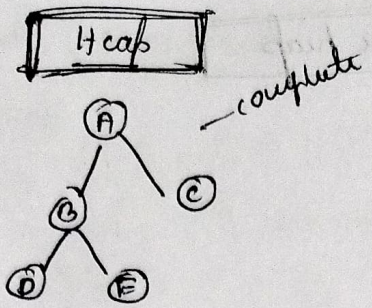


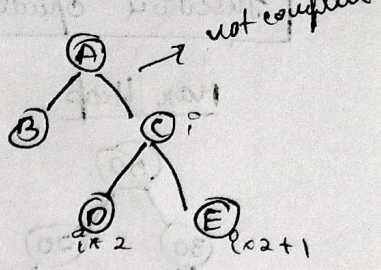












	1	2	3	4	5	6	7
T	A	B	C	D	E	F	G

	1	2	3	4	5
	A	B	C	D	E

	1	2	3	4	5	6	7
	A	B	C	-	-	D	E

If a node is at index = i

If left child is at ⟶ $2*i$

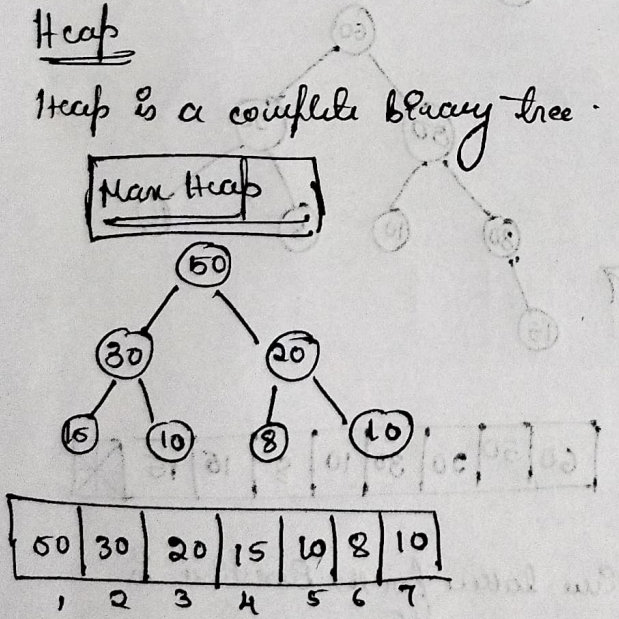
If right child is at ⟶ $2*i+1$

Its parent is at ⟶ $[\frac{i}{2}]$

- A complete binary tree is a full binary tree upto height h-1. and elements are filled from left to right.
- height of a complete binary tree will always be $\log n$. because unnecessarily we will not go to the next level once the previous level gets filled.

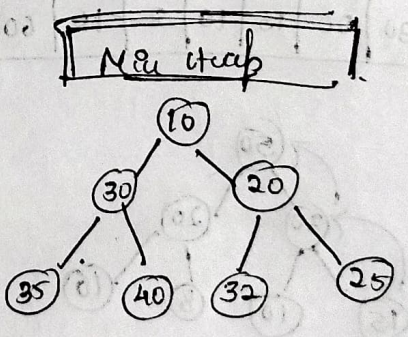
Heap

- Heap is a complete binary tree.



	1	2	3	4	5	6	7
	50	30	20	15	10	8	10

- Max heap is a complete binary tree satisfying the condition that every a node greater than all it's descendent
- duplicate are allowed here

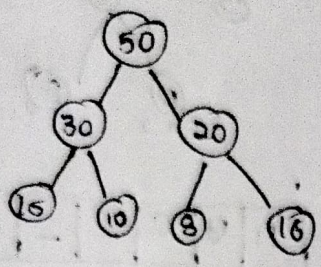


	1	2	3	4	5	6	7
	10	30	20	35	40	32	25

- Smallest element will be having in the root
- Min heap is a complete binary tree satisfying the condition that every node having a element smaller than all it's decendance

Insertion operation in Max heap

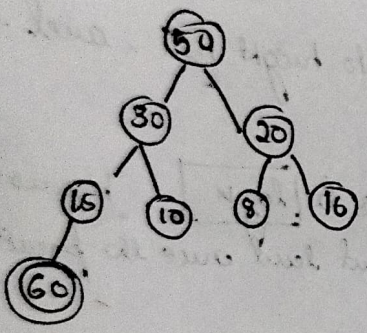
Max Heap



50	30	20	15	10	8	16	
1	2	3	4	5	6	7	

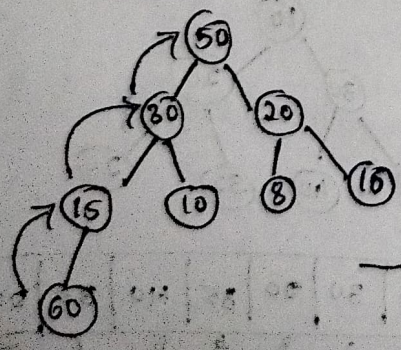
(1) Insert a elmt 60

50	30	20	16	10	8	15	60
							8

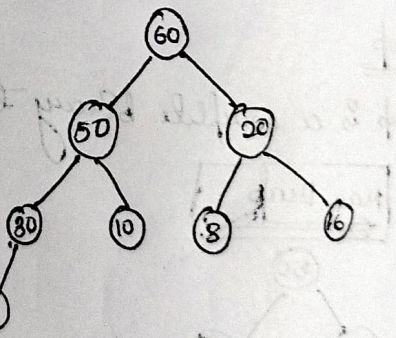


(2) (3) compare

50	30	20	16	10	8	16	60



(3)

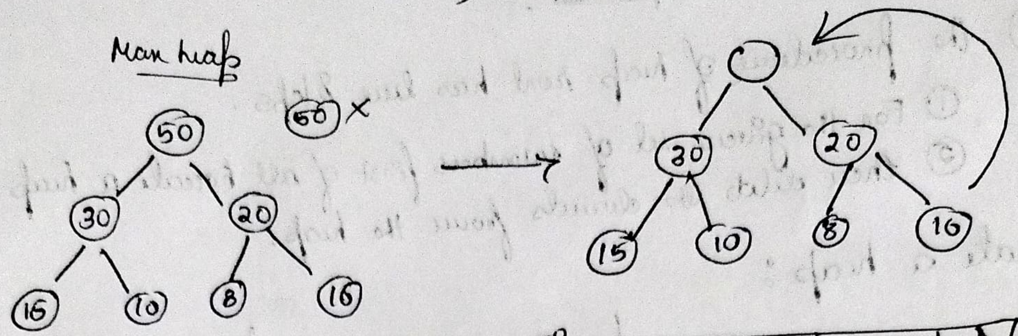


60	50	20	30	10	8	15	15

- Time taken for the insertion is O(1) to O(log n)
  - O(1): minimum (if there is no swapping)
  - O(log n): maximum (if there is a swapping)

- Remove the element (Root element)

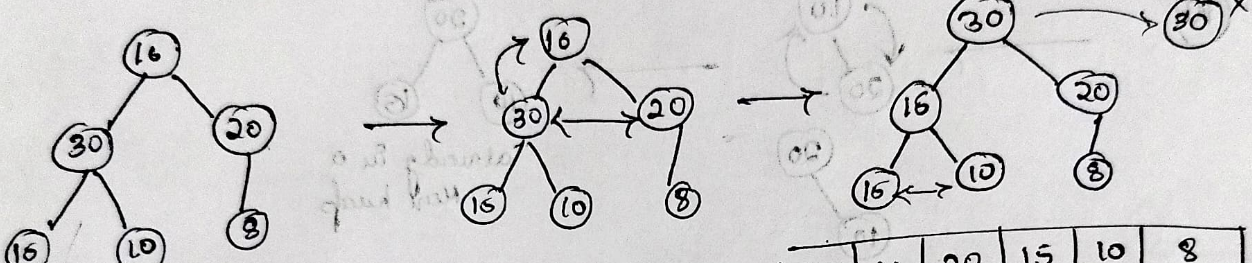
Max heap



50	30	20	15	10	8	16
1	2	3	4	5	6	7

50

16	30	20	16	10	8	X
1	2	3	4	5	6	

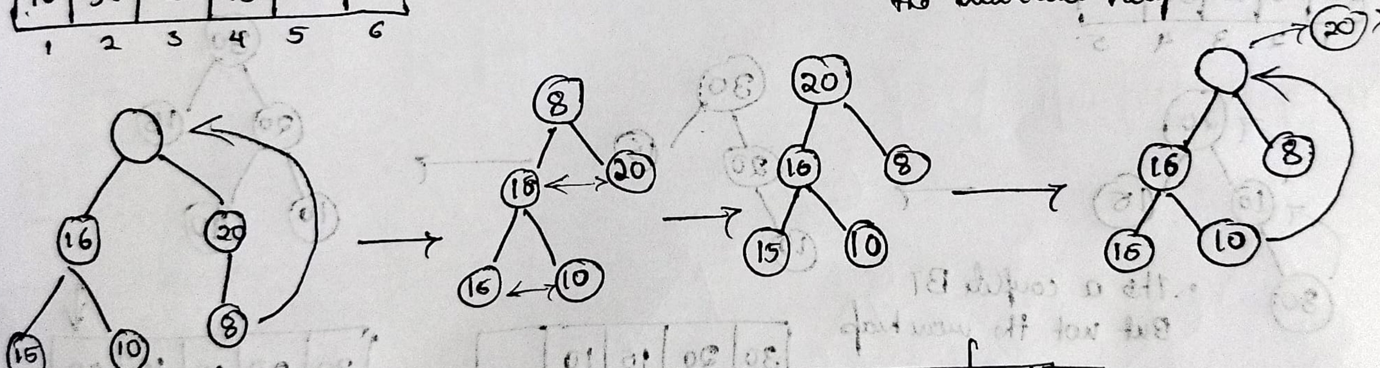


30	16	20	15	10	8
1	2	3	4	5	6

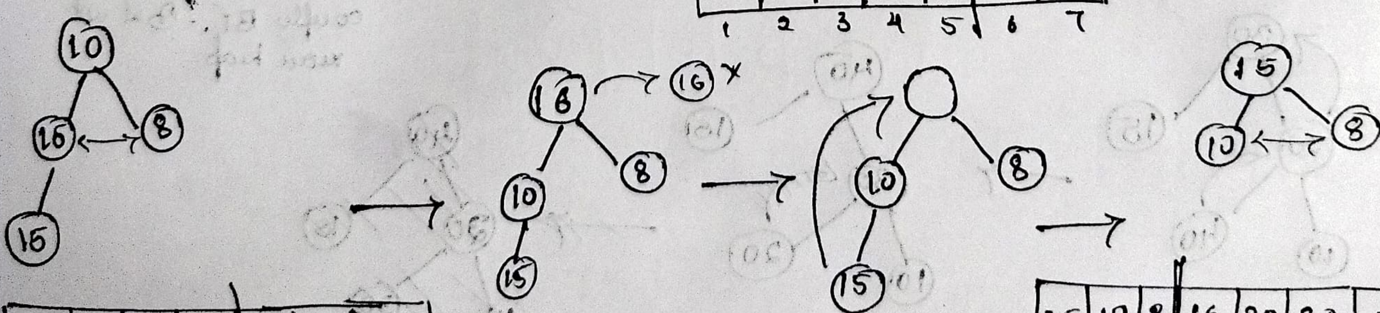
- This is a complete binary But not the max heap

- Maximum time → logn
- When we are delete we get the maximum heap

16	30	20	15	10	8
1	2	3	4	5	6

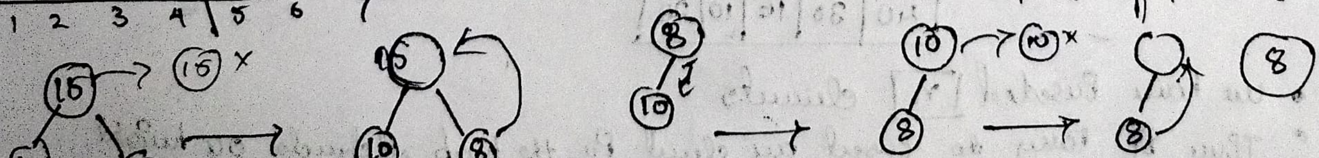


20	16	8	15	10	30	50
1	2	3	4	5	6	7



10	16	8	15	20	30	50
1	2	3	4	5	6	7

15	10	8	16	20	30	50
1	2	3	4	5	6	7



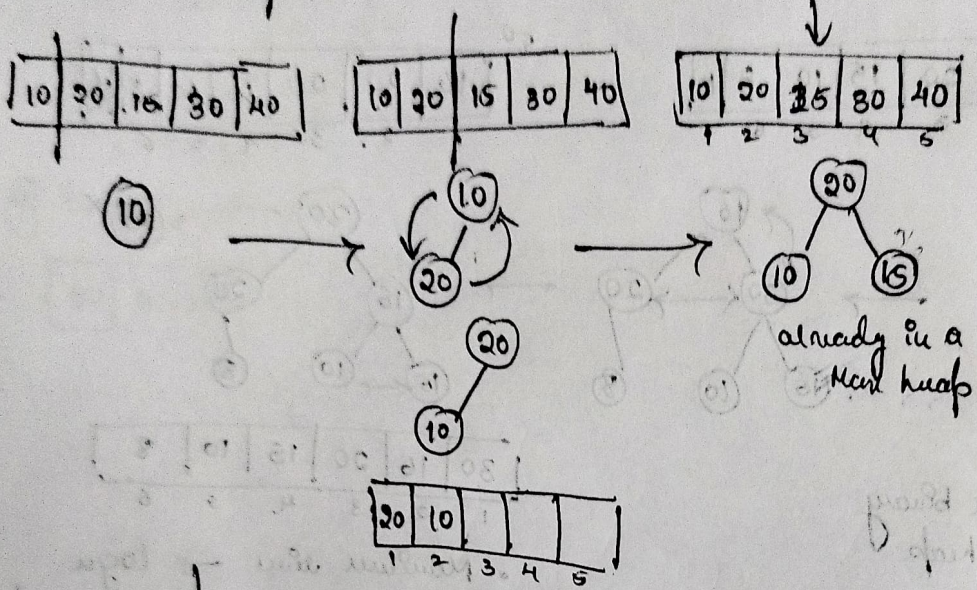
8	10	15	16	20	30	50

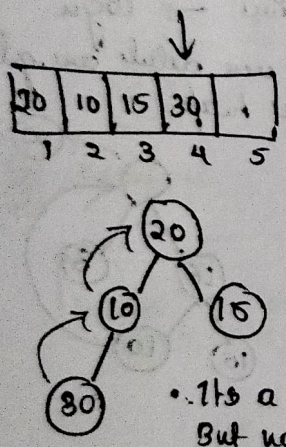
8	10	15	16	20	30	50

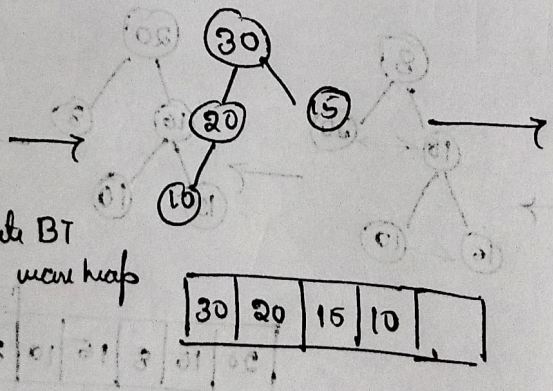
Heap sort

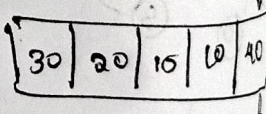
① The procedure of heap sort has two steps.
   ① For the given set of numbers first of all create a heap
   ② then delete the elements from the heap.

create a heap:

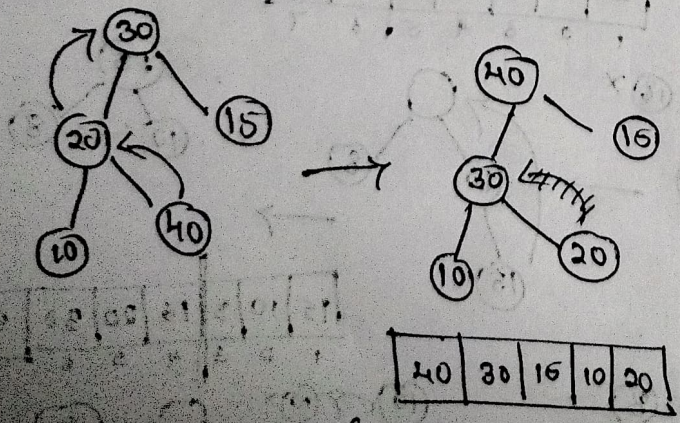


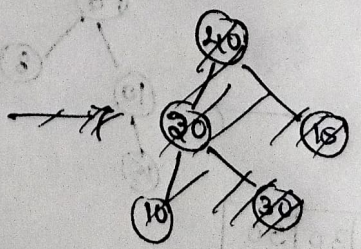






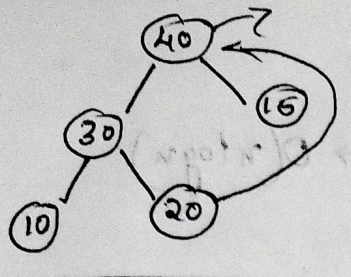
complete BT . But not max heap

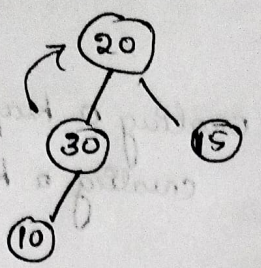


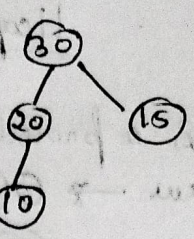


- we have inserted [n] elements
- Time it takes to insert an element in the heap, depends on height [n log n]

Second step :





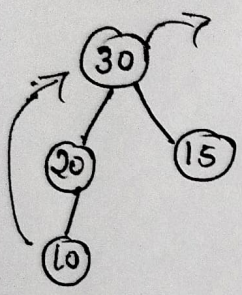


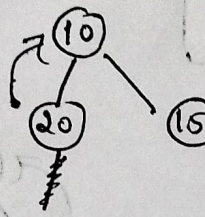
40	30	15	10	20

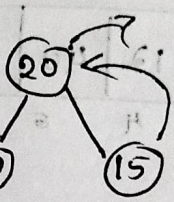
this is a complete binary tree but not the max heap

20	30	15	10	

30	20	15	10	40



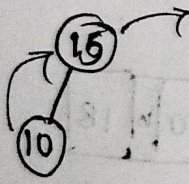


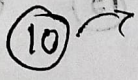


10	20	15		40

10	20	15	30	40

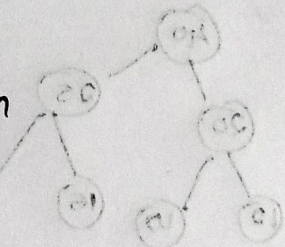
15	10	20	30	40

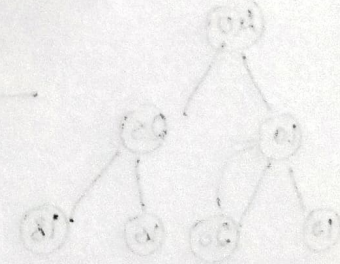




10	15	20	30	40

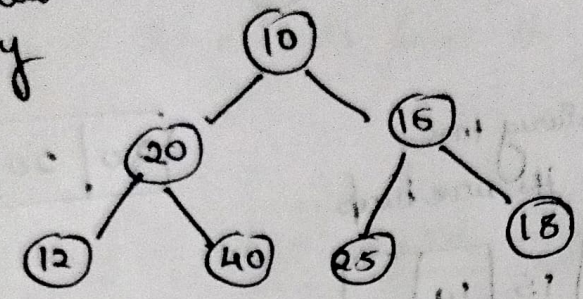
- creating of heap → nlogn
- Deletion of heap → nlogn

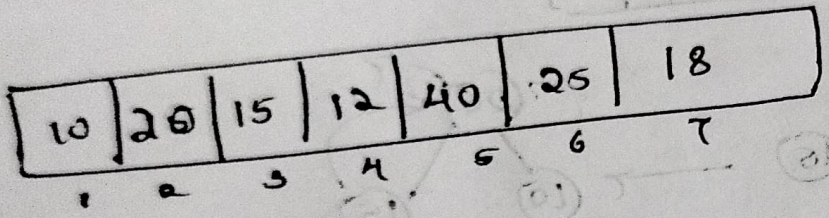


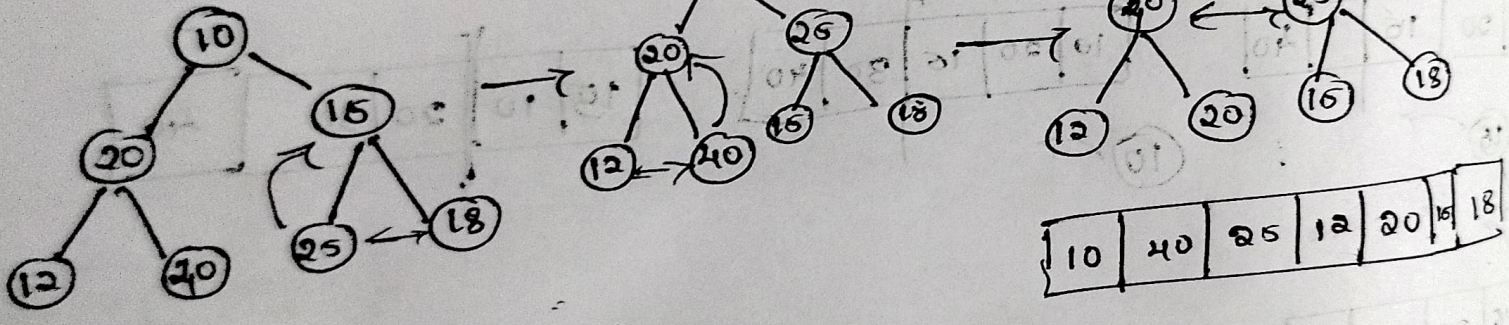


Heapify

- Heapify is a procedure for creating a heap
- Time taken → $O(n)$ creating a heap → $O(n \log n)$ heapify







Priority Queue

- The elements will have the priority and inserting and deletion are based on the priority.

smaller number higher priority or larger number higher priority

insertion → log n
deletion → log n

- so heap is a data for implementing the priority.