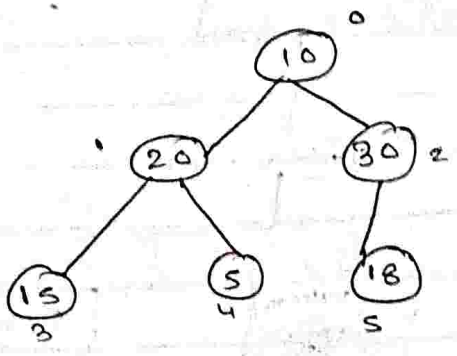
Heapify.

arr →

10	20	30	15	5	18
0	1	2	3	4	5



Suppose. In heapify function we had given an index 0 and its a max heap.

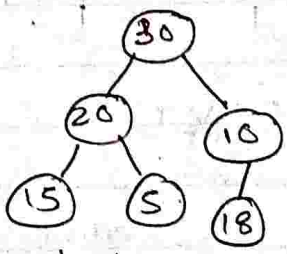
heapify (0).

compare (10, 20, 30)

max Element = 30.

30 is not a parent node.

∴ swap 30 & 10.



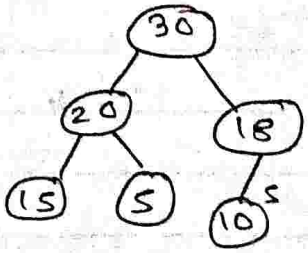
0	1	2	3	4	5
30	20	10	15	5	18

It will recursively call heapify(2), where 10 is stored

max (10, 18) = 18.

18 is not a parent node.

∴ swap 10 and 18.



0	1	2	3	4	5
30	20	18	15	5	10

heapify (5) is a leaf node, ∴, it will end here.

Build heap :-

Given an array, convert it to a heap using heapify algo.

arr →

7	1	2	5	8	4
0	1	2	3	4	5

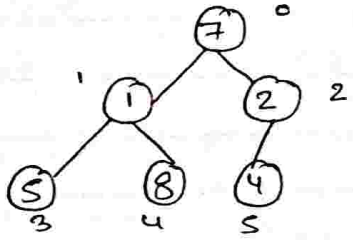
leaf Node begins from $\frac{N}{2}$ to N-1. where N = 6.

∴ 3 to 5, leaf Nodes are present.

So, we will call heapify function in Build heap from 2 to 0.

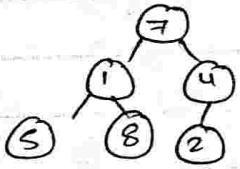
heapify (2).

For Iteration = 2



Compare (2,4) = 4.

4 is not a parent ∴ swap 2, 4.



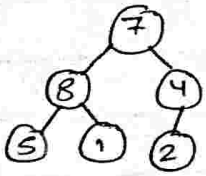
7	1	4	5	8	2

Now, 2 is a leaf node, ∴ we will stop here.

For Iteration = 1.

compare (1, 5, 8) = 8.

8 is not a parent then swap.

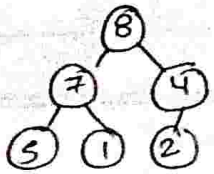


7	8	4	5	1	2

For Iteration = 0.

compare (7, 8, 4) = 8.

8 is not a parent node. ∴ swap 7 & 8.



8	7	4	5	1	2

It will recursively check other but maximum node is parent. So no swapping