

Date: / /

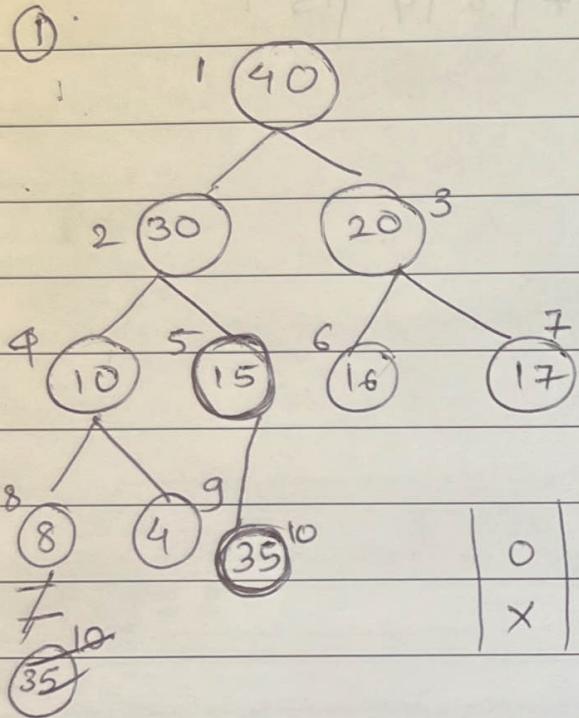
1- Inserting 35.

Input : 40, 30, 20, 10, 15, 16, 17, 8, 4.

Array :-

0	1	2	3	4	5	6	7	8	9	10
x	40	30	20	10	15	16	17	8	4	35

Added 35



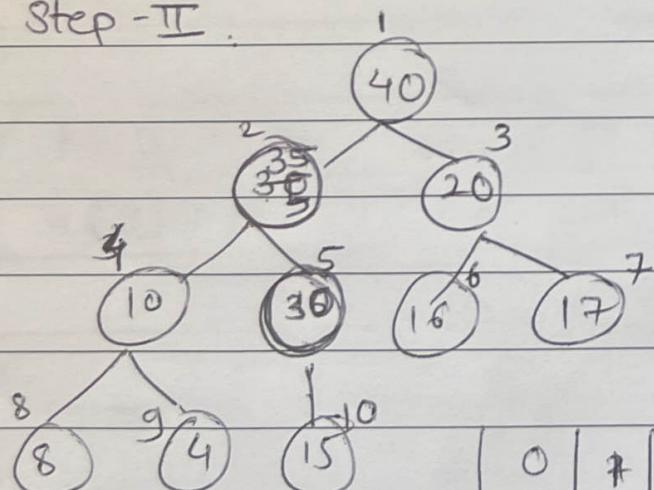
Since,

35 is greater than 15
(parent), will swap.

After swapping array :-

0	1	2	3	4	5	6	7	8	9	10
x	40	30	20	10	35	16	17	8	4	15

Step - II .



Again,

35 is greater than 30 (parent) so we will swap.

After swapping :-

0	1	2	3	4	5	6	7	8	9	10
x	40	35	20	10	30	16	17	8	4	15

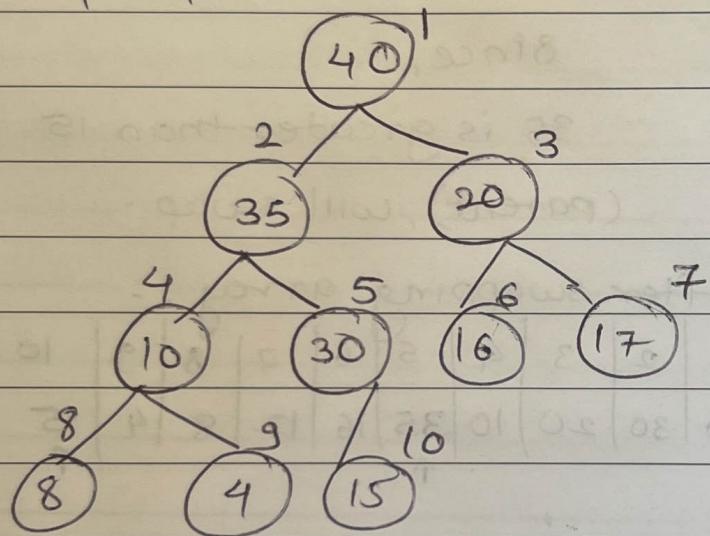
Date: / /

since, 40 (parent) is greater than 35,
there will be no more swaps.

Final array looks like :-

0	1	2	3	4	5	6	7	8	9	10
X	40	35	20	10	30	16	17	8	4	15

Heap representation :-



Date: / /

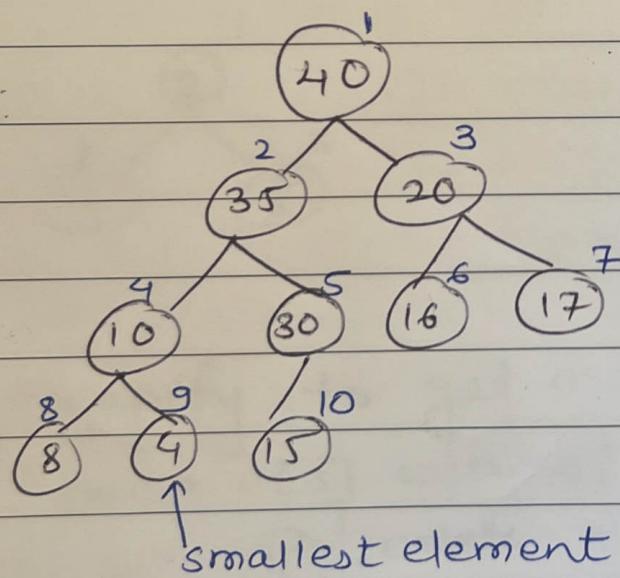
② In a MAX heap containing n numbers, the smallest element can be found in :-

complexity : $O(n)$

- In a MAXHEAP, the smallest element can be found on the last level of heap or the second last level of the heap.
- The time complexity to reach to last level is $O(n)$

For. ex.

- $A = \{ 40, 35, 20, 10, 30, 16, 17, 8, 4, 15 \}$ - From 1st que.



- To find 4, we need to traverse to all the all the levels of the heap.

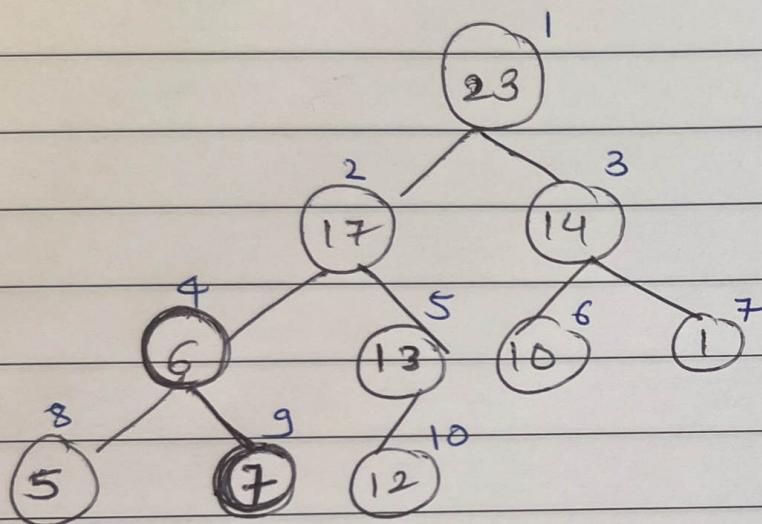
③ Is the array with values

Date: / /

23, 17, 14, 6, 13, 10, 1, 5, 7, 12 a MAX HEAP - ?

ANS:- Let's put it in heap representation :-

1	2	3	4	5	6	7	8	9	10
23	17	14	6	13	10	1	5	7	12



→ This is not a MAX-heap.

Since, MAX-heap should have root node values greater than its leaf node.

- At $a[4]$ which 6, it has two children

$$a[8] = 5$$

$$a[9] = 7$$

$\therefore a[9] = 7$ is greater than its parent node. It is not a MAX-HEAP

4.

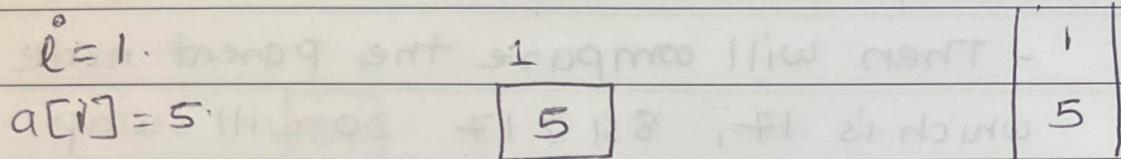
Date: / /

Building a Heap.

Top to Bottom.

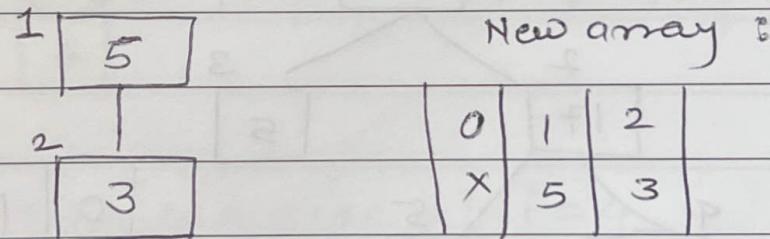
$$A = \{5, 3, 17, 10, 84, 19, 6, 22, 9\}$$

i	0	1	2	3	4	5	6	7	8	9
	X	5	3	17	10	84	19	6	22	9



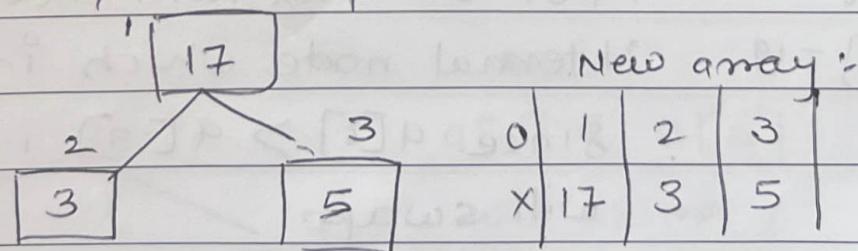
$i=2$ since 3 is less than 5.

$a[2] = 3$.



$i=3$ since 17 is greater than 5, will swap.

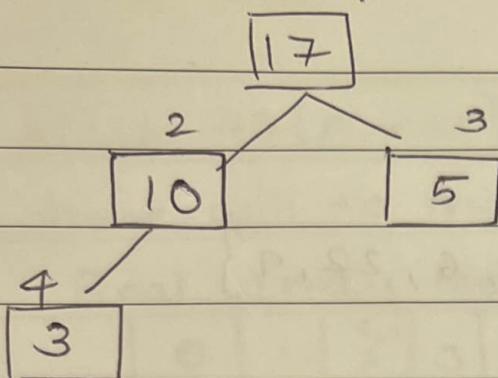
$a[3] = 17$.



$i=4$ since 10 is greater than 3

$a[4] = 10$ will swap

Date: / /

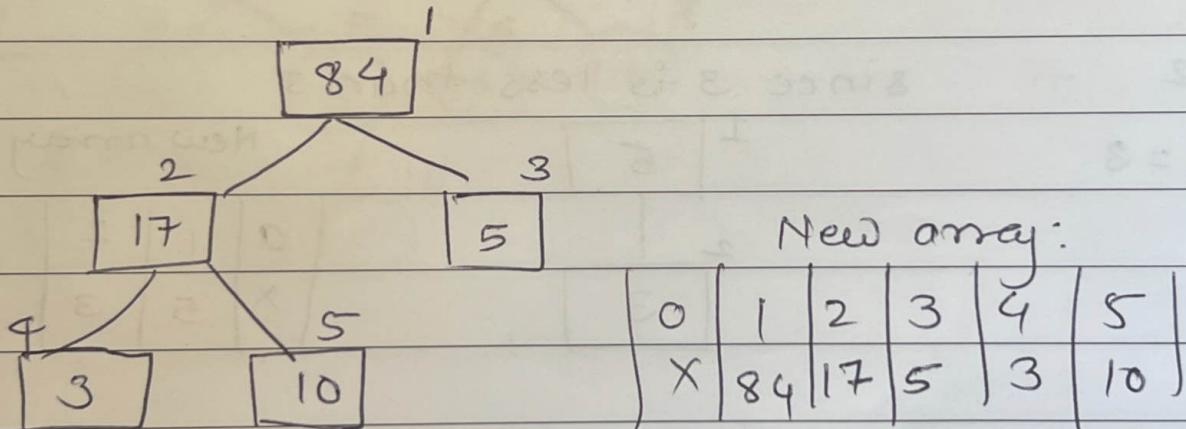


New array:-

0	1	2	3	4
X	17	10	5	3

$i = 5$ - since 84 is greater than 10, will
 $a[5] = 84$ swap with 10 first:

- Then will compare the parent node
 which is 17, $84 > 17$ so will swap
 84 with 17.

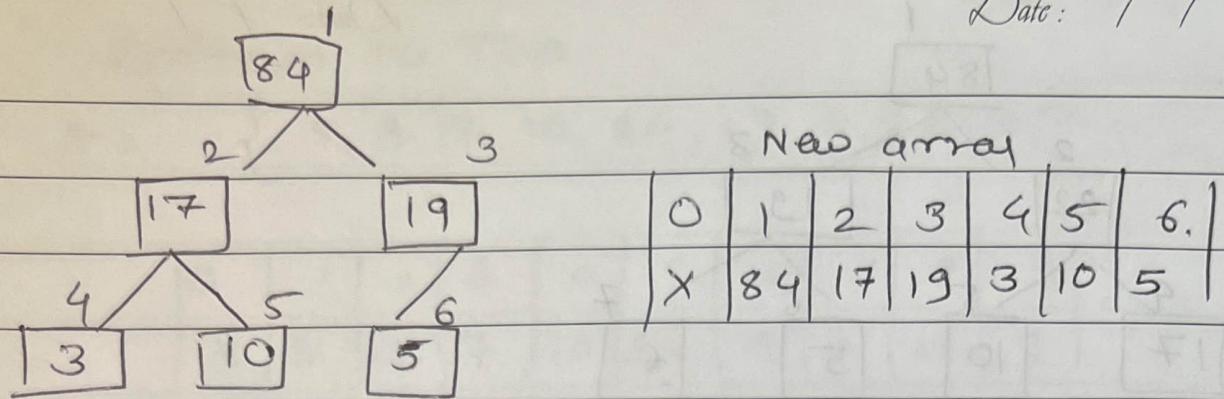


New array:

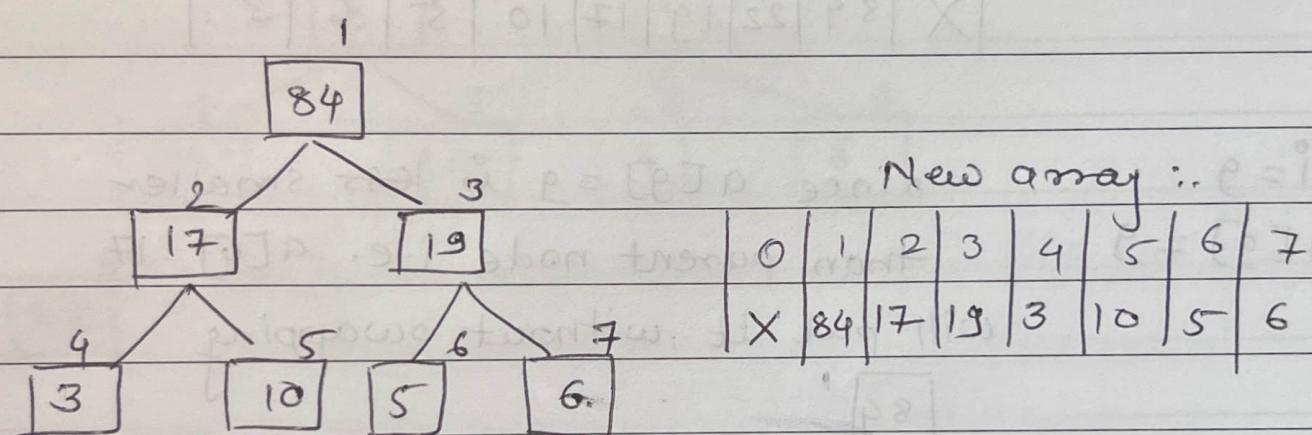
0	1	2	3	4	5
X	84	17	5	3	10

$i = 6$: For 6th index, will check with internal
 $a[6] = 19$ internal node which index 3. $a[3] = 5$
 , since $a[6] > a[3]$ i.e. $19 > 5$
 will swap.

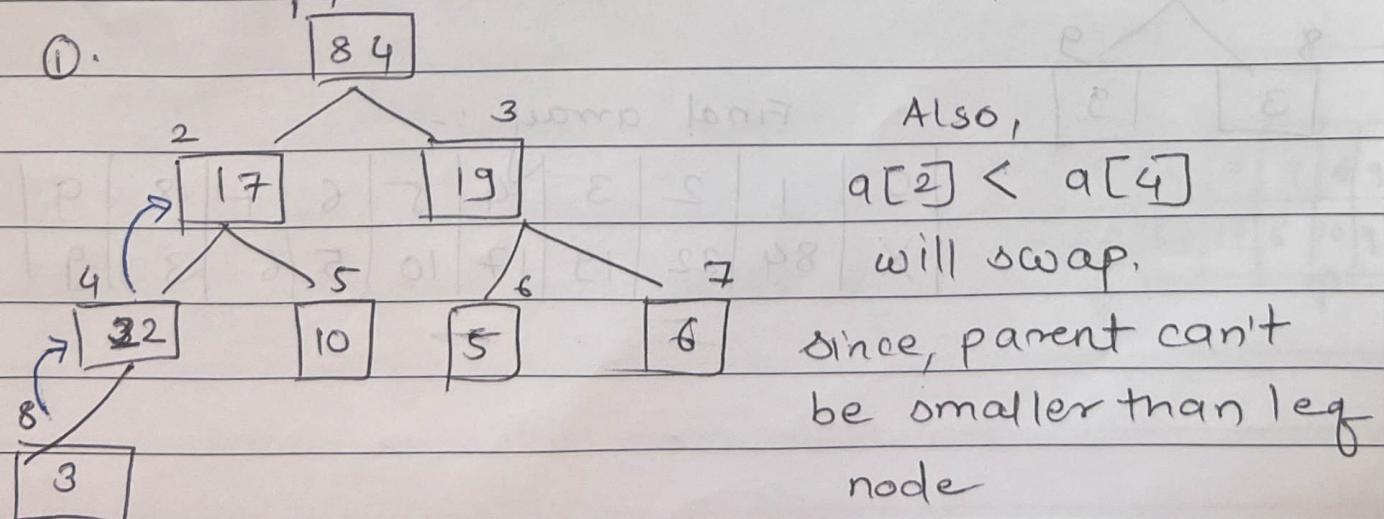
Date: / /



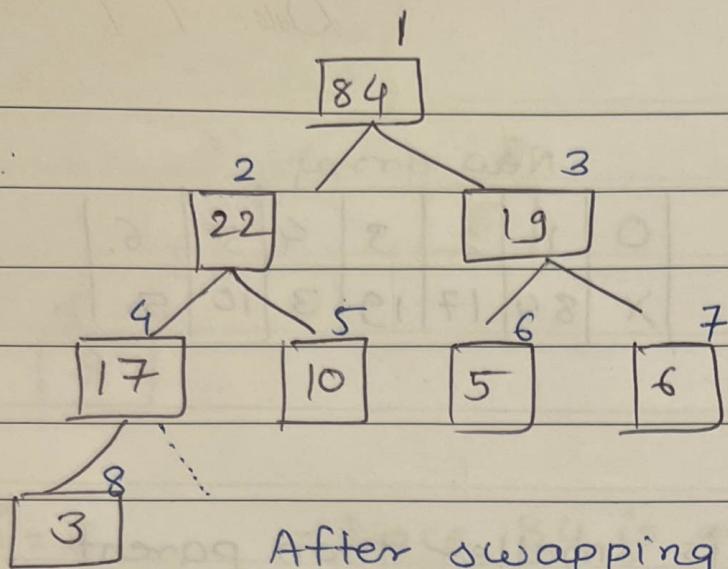
$i=7$ since 6 is smaller than parent
 $a[7]=6$ node at index 3, will keep as is.



$i=8$ We will have leaf node at $i=4$
 $a[8]=22$ which $a[4]=3$, smaller than
parent, so we will swap.



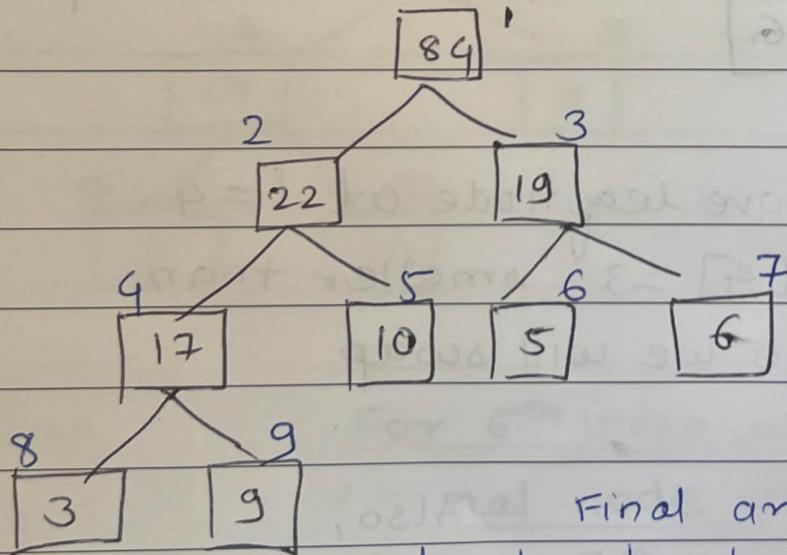
Date: / /



After swapping, twice final array:

0	1	2	3	4	5	6	7	8
X	84	22	19	17	10	5	6	3

$i = 9$ since $a[9] = 9$ is less smaller
 $a[9] = 9$ than parent node i.e. $a[4] = 17$
 will put it, without swapping



Final array :-

0	1	2	3	4	5	6	7	8	9
X	84	22	19	17	10	5	6	3	9

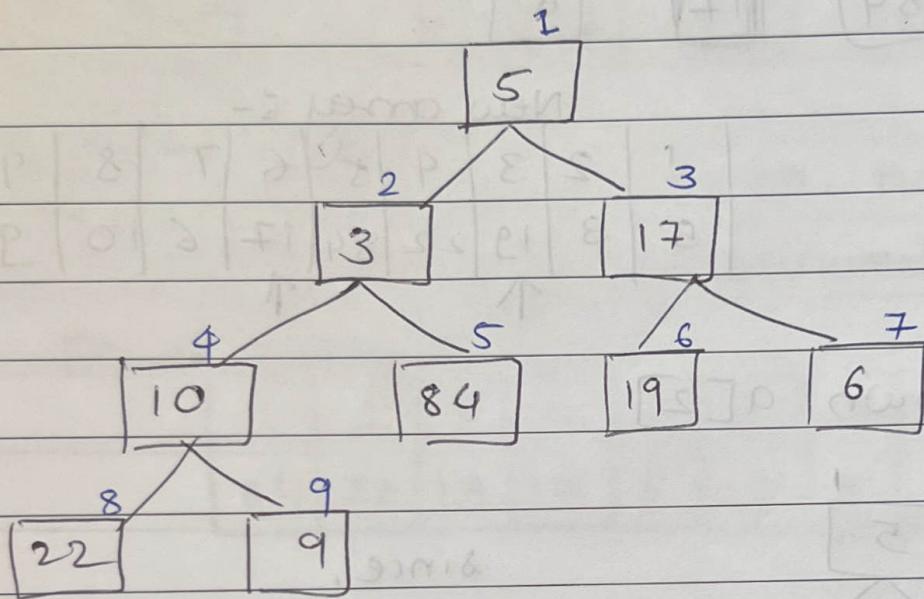
4. Building a heap

Date: / /

- Bottom to Top.

$$A = \{ 5, 3, 17, 10, 84, 19, 6, 22, 9 \}$$

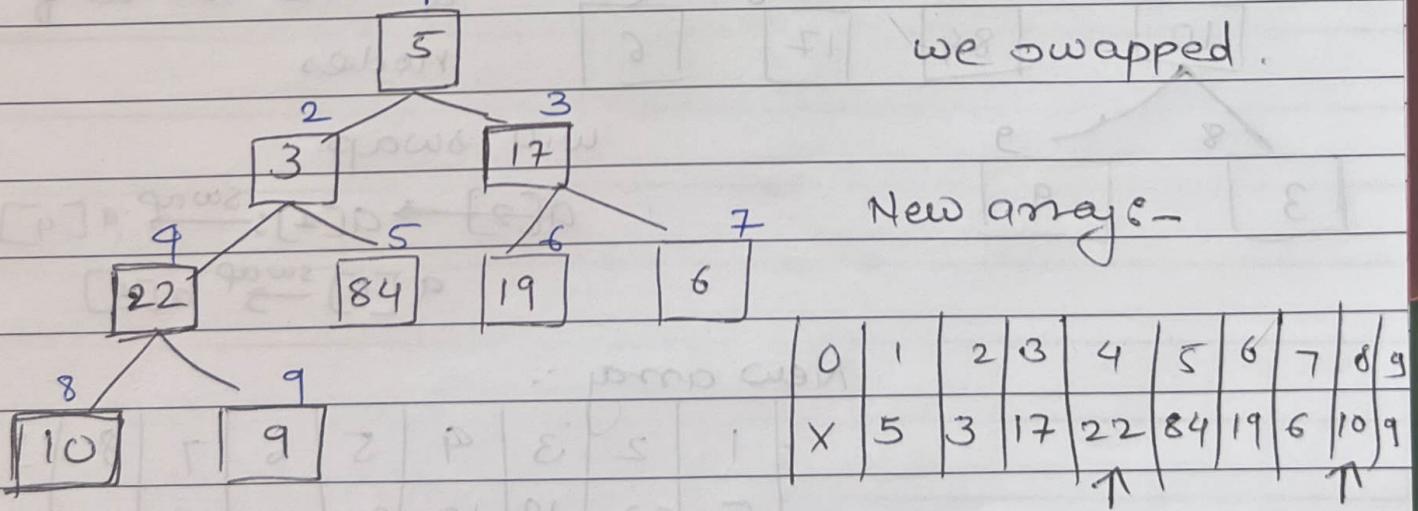
0	1	2	3	4	5	6	7	8	9
X	5	3	17	10	84	19	6	22	9.



Heapify $a[4]$

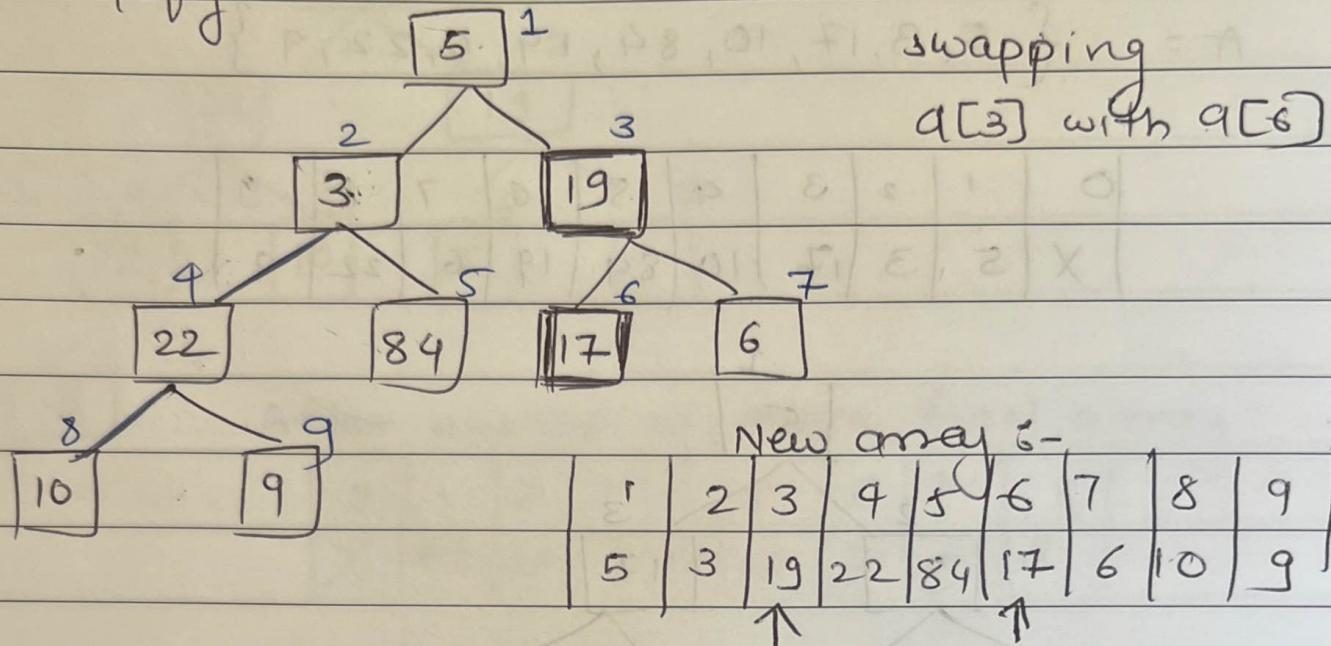
since $a[4] > a[8]$

we swapped.

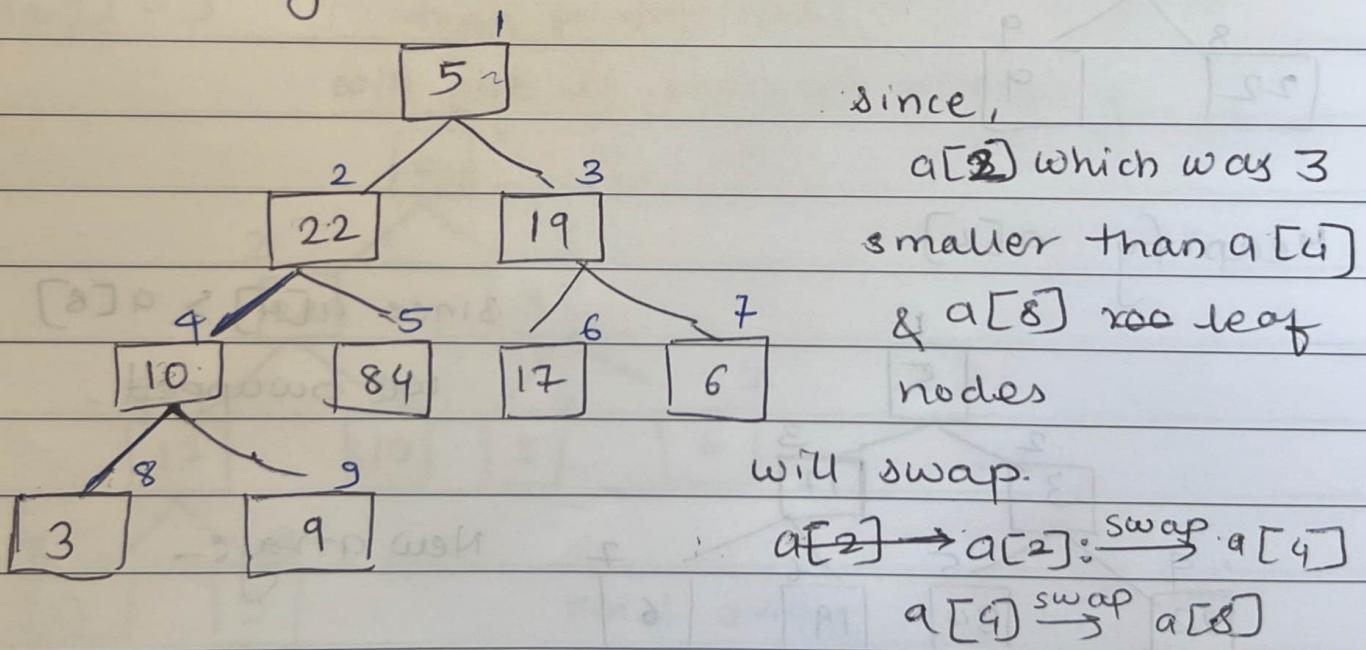


Date: / /

② Heapify down $a[3]$

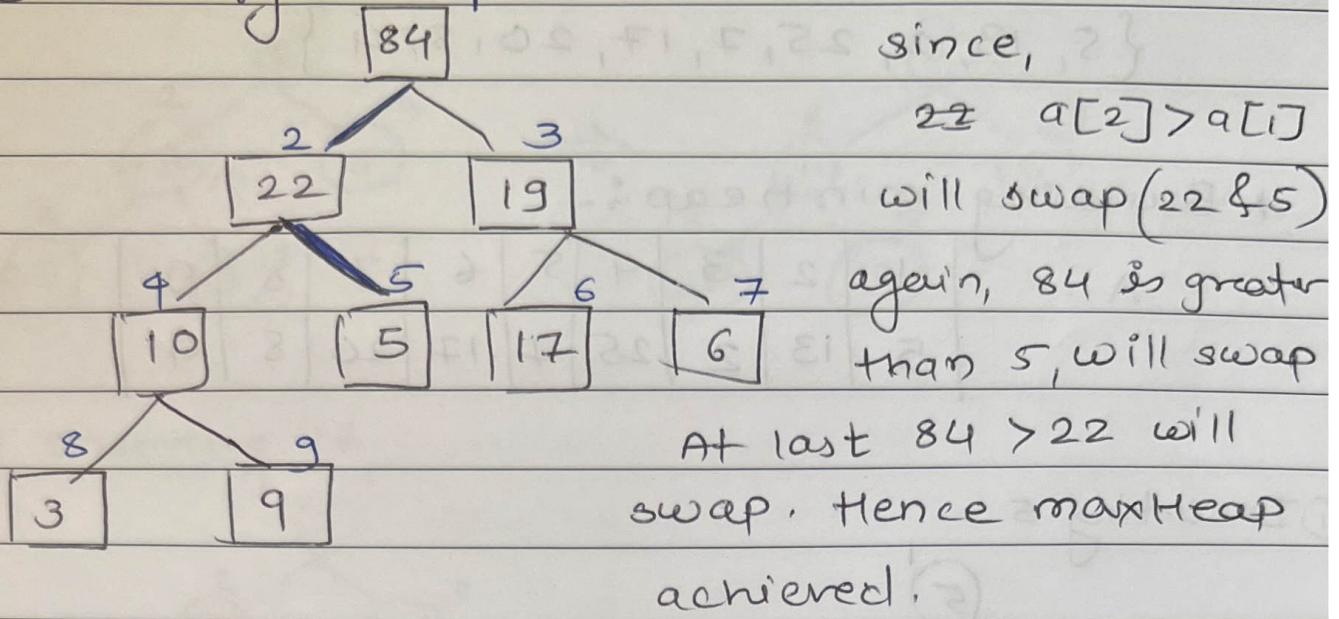


③ Heapify down $a[2]$



New array :-

1	2	3	4	5	6	7	8	9
5	22	19	10	84	17	6	3	9

(4) Heapify down, $a[1]$ 

Final array :

1	2	3	4	5	6	7	8	9
84	22	19	10	5	17	6	3	9

5. Build minheap & maxheap

{5, 13, 2, 25, 7, 17, 20, 8, 4}

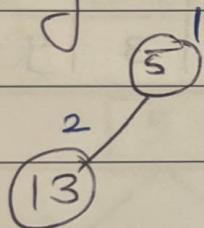
+ Building min heap :-

A =	1	2	3	4	5	6	7	8	9
	5	13	2	25	7	17	20	8	4

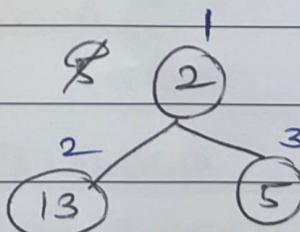
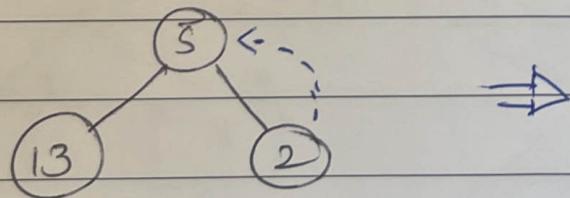
① Inserting 5



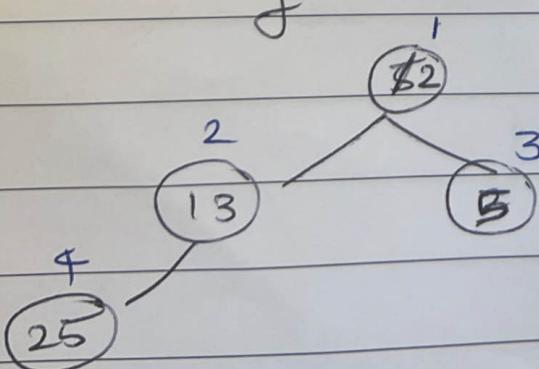
② Inserting 13



③ Inserting 2

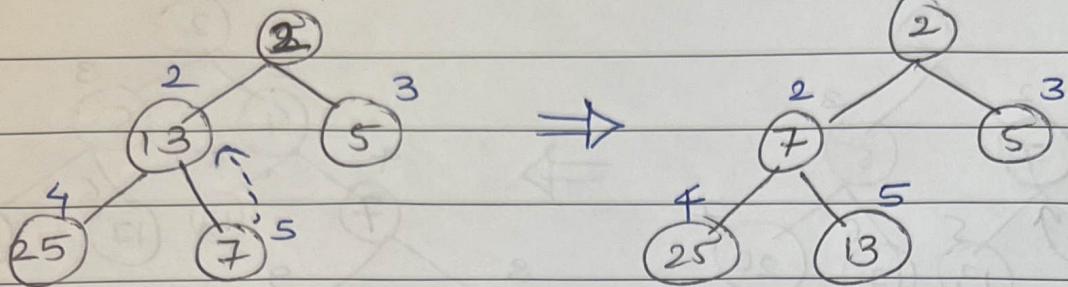


④ Inserting 25

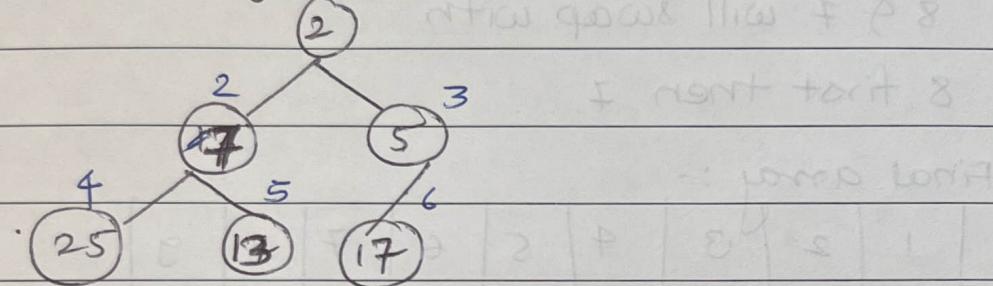


Date: / /

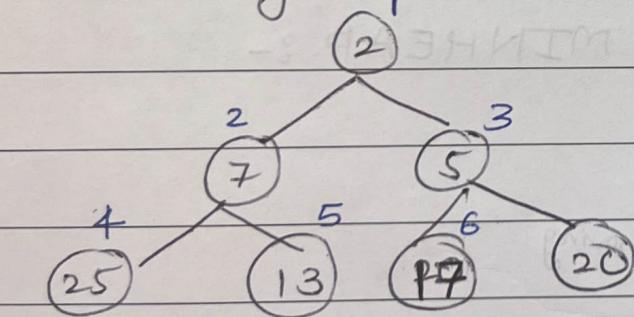
⑤ Inserting 7



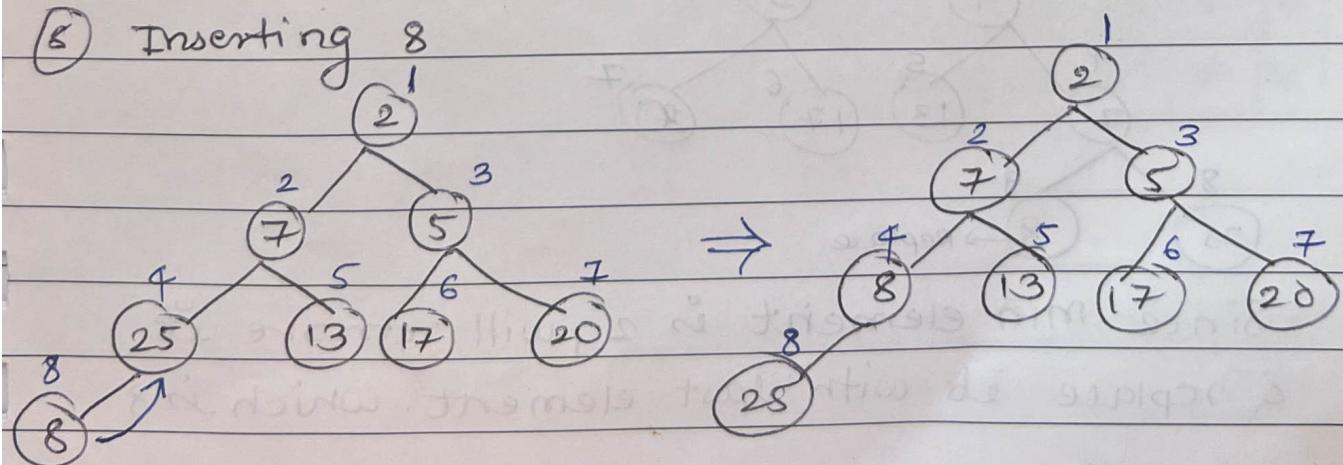
⑥ Inserting 17



⑦ Inserting 20

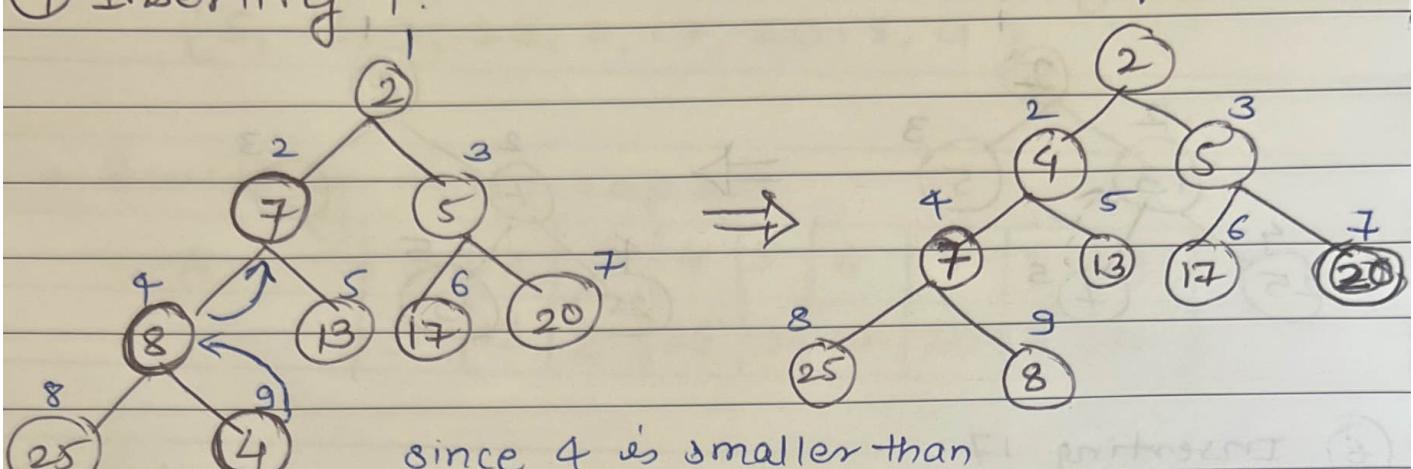


⑧ Inserting 8



Date: / /

⑨ Inserting 4.



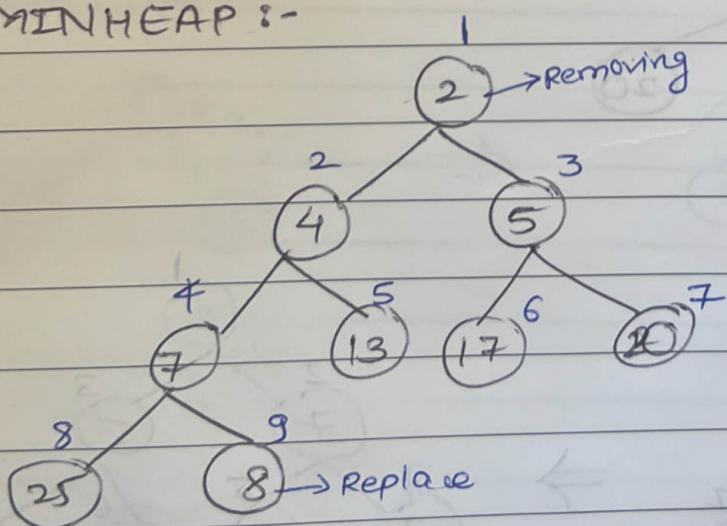
since 4 is smaller than
8 & 7 will swap with
8 first then 7

Final array :-

1	2	3	4	5	6	7	8	9
2	4	5	7	13	17	20	25	8

* Deleting min from MINHEAP :-

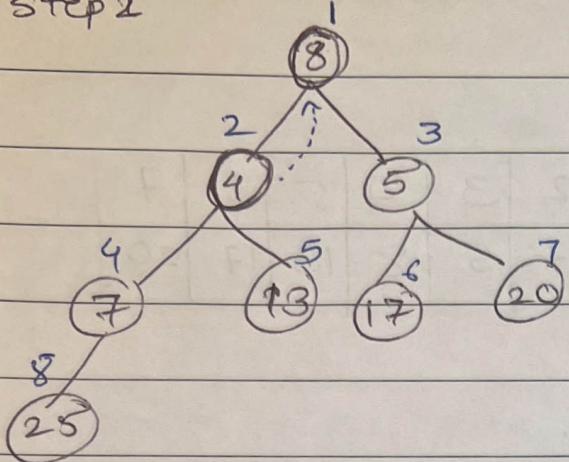
MINHEAP :-



since min element is 2, will remove it
& replace it with last element. which is 8

Date: / /

step 2

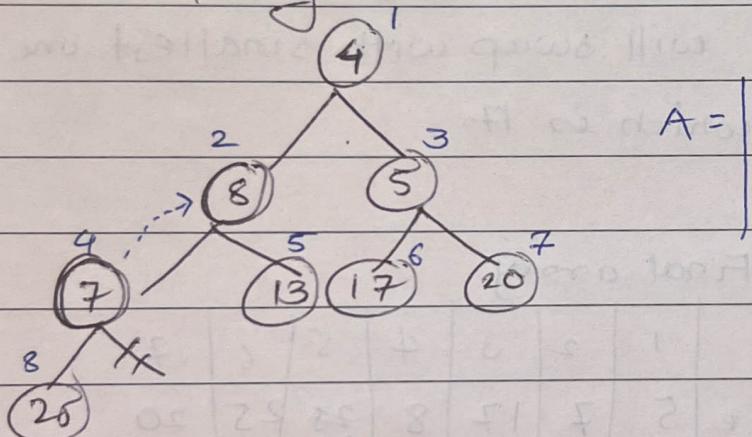


$A =$

1	2	3	4	5	6	7	8
8	4	5	7	13	17	20	25

Since, it's a MINHEAP, Parent node cannot be greater than either leaf node. (Heapify)

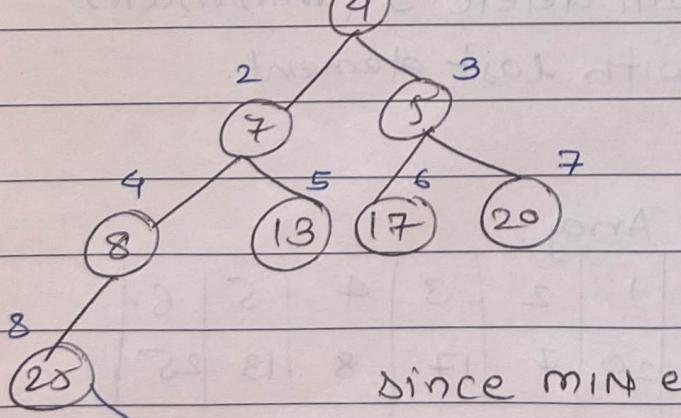
swapping 8 with 4.



1	2	3	4	5	6	7	8
4	8	5	7	13	17	20	25

swapping 8 with 7.

* Remove

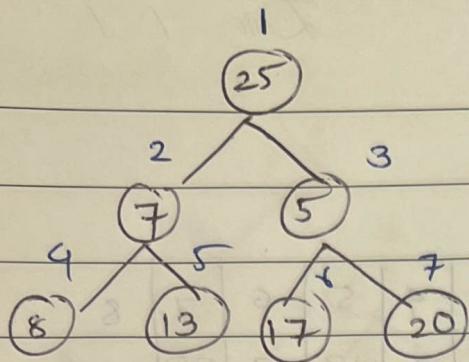


1	2	3	4	5	6	7	8
4	7	5	8	13	17	20	25

Replace

since min element is 4, will delete 4, & replace it with last element.

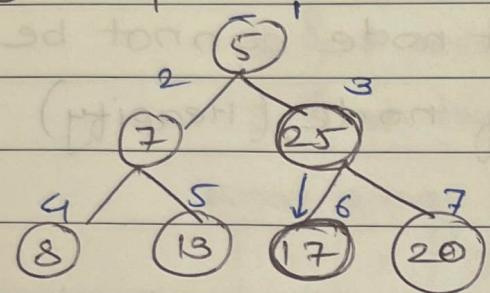
Date: / /



1	2	3	4	5	6	7
25	7	5	8	13	17	20

∴ swapping to get MINHEAP.

① swap 25, with smallest element 5.

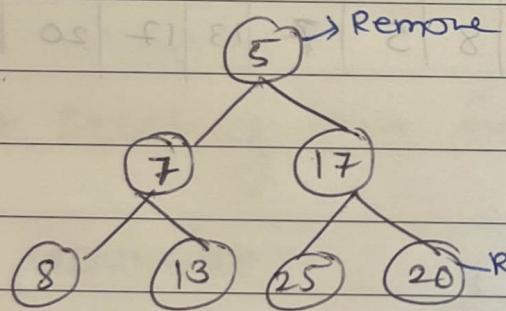


since, 25 is greater

than its leaf nodes,

will swap with smallest one

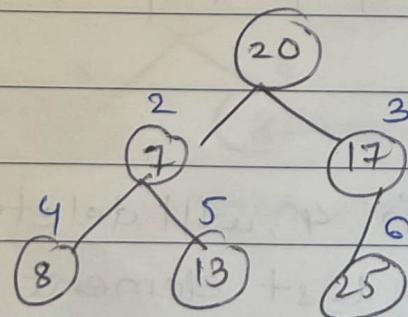
which is 17.



Final array

1	2	3	4	5	6	7
5	7	17	8	25	13	20

∴ since, it's MINHEAP, will delete 5 (minimum element) & replace it with last element.

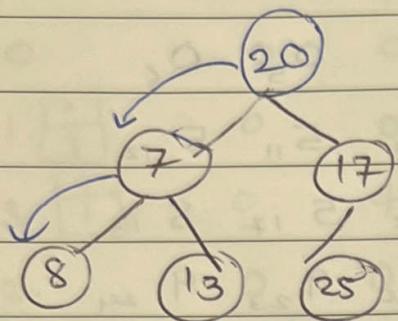


Array:

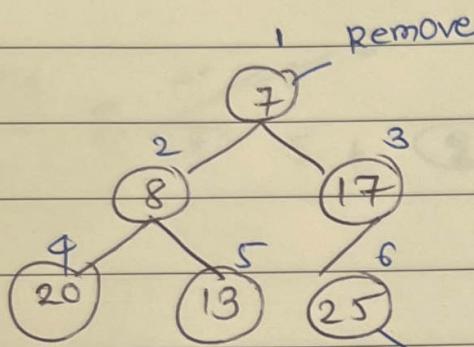
1	2	3	4	5	6
20	7	17	8	13	25

Date: / /

Heapify, to get minimum element at root.



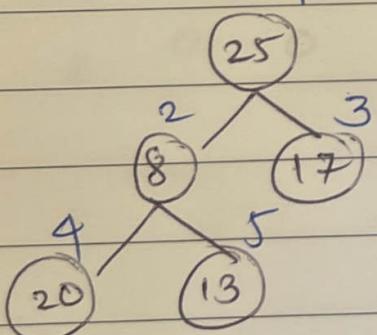
- ① swap 20 with 7
- ② swap 20 with 8.



since, all the leaf nodes
are ~~not~~ smaller than leaf
node, its a MINHEAP

replace	1	2	3	4	5	6	
	7	8	17	20	13	25	

+ Remove minimum element i.e. 7 & replace it
with 25,



Array :-

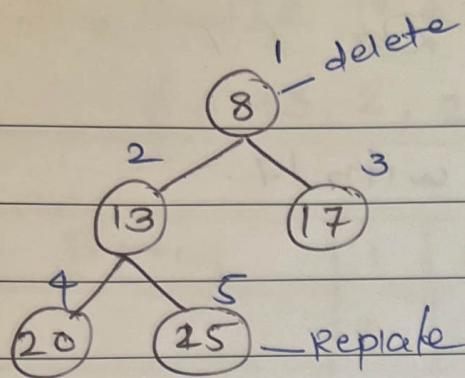
1	2	3	4	5
25	8	17	20	13

* heapify, to get MINHEAP.



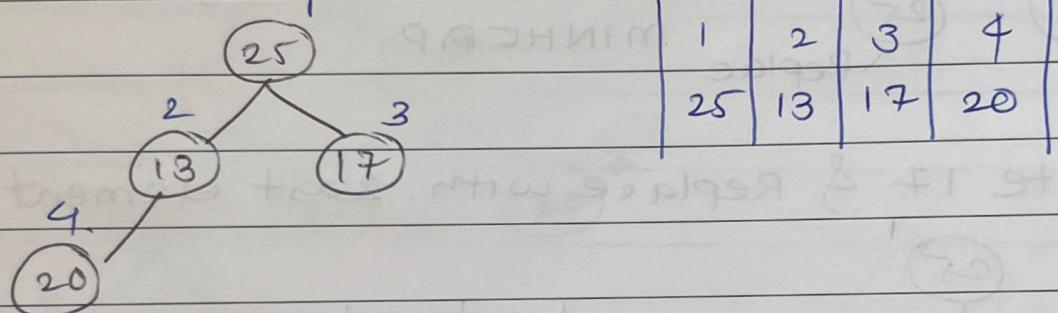
- ① swap 25 with 8
- ② swap 25 with 13.

Date: / /

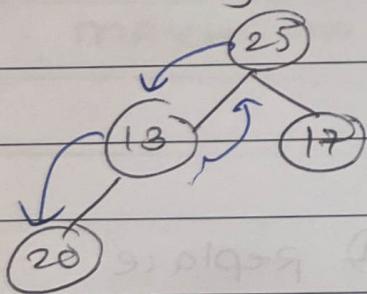


since, all the leaf nodes
are smaller than root node
its MINHEAP.

* For Delete min element (8) & replace with last one.

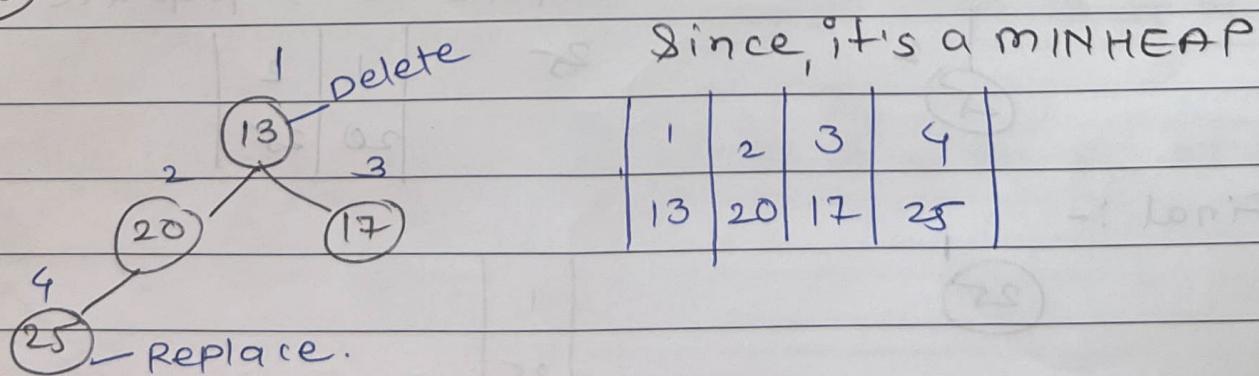


+ Heapify to get MINHEAP :-

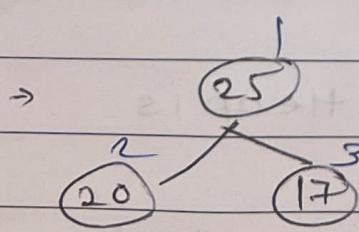


① swap 25 with 13

② swap 25 with 20.



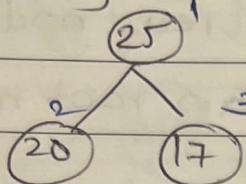
Since, it's a MINHEAP



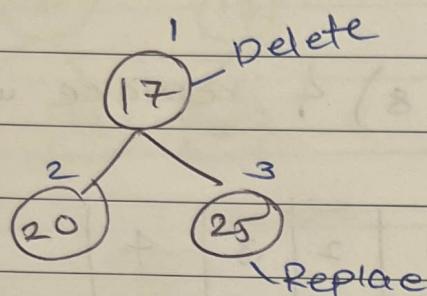
1	2	3
25	20	17

Date: / /

* Heapify to get MINHEAP:-

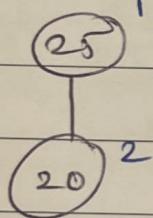


Replace 25 with 17.



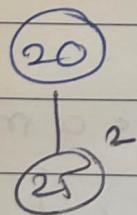
since, root node is
greater than leaf its not
MINHEAP.

- Delete 17 & Replace with last element



1	2
25	20

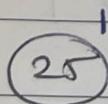
* Heapify to get MINHEAP



Delete 20 & replace

25	1	2
20	25	

Final :-



1
25

Delete last element 25 & Now heap is
Empty.

5 * Build MAXHEAP.

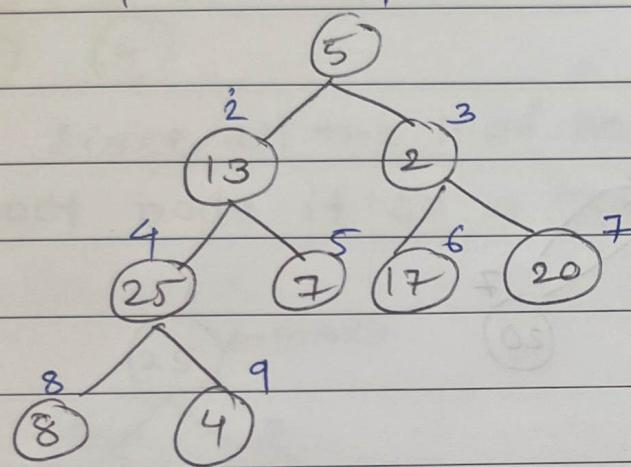
Date: / /

{ 5, 13, 2, 25, 7, 17, 20, 8, 4 }

Array : | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

| 5 | 13 | 2 | 25 | 7 | 17 | 20 | 8 | 4 |

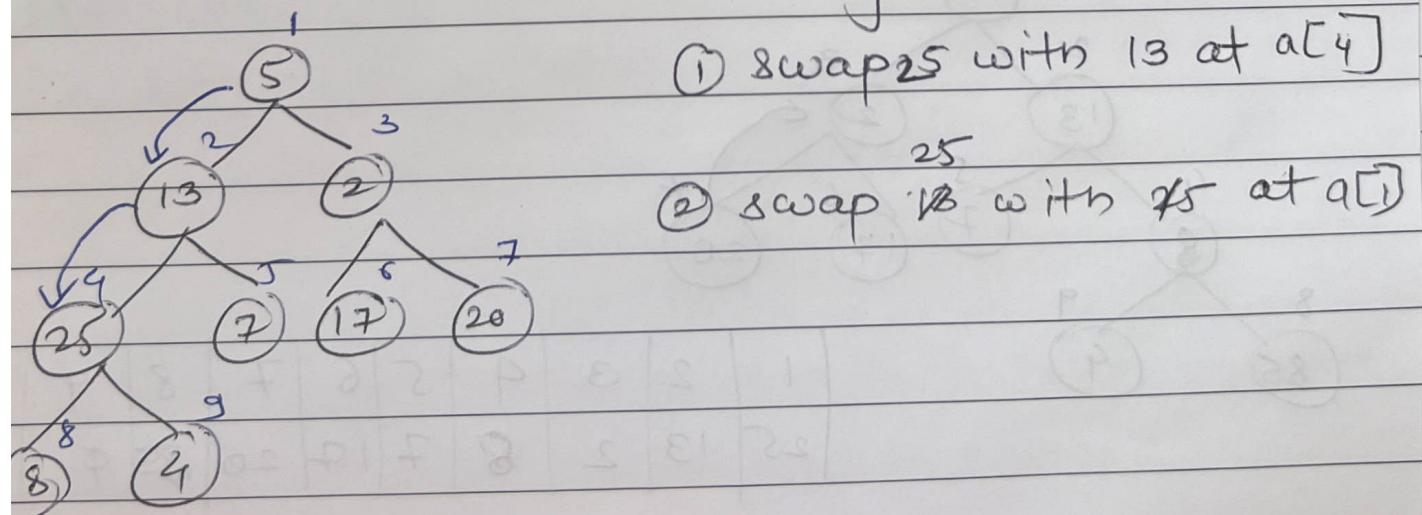
: heap :-



∴ MAXHEAP : has largest element as root node.

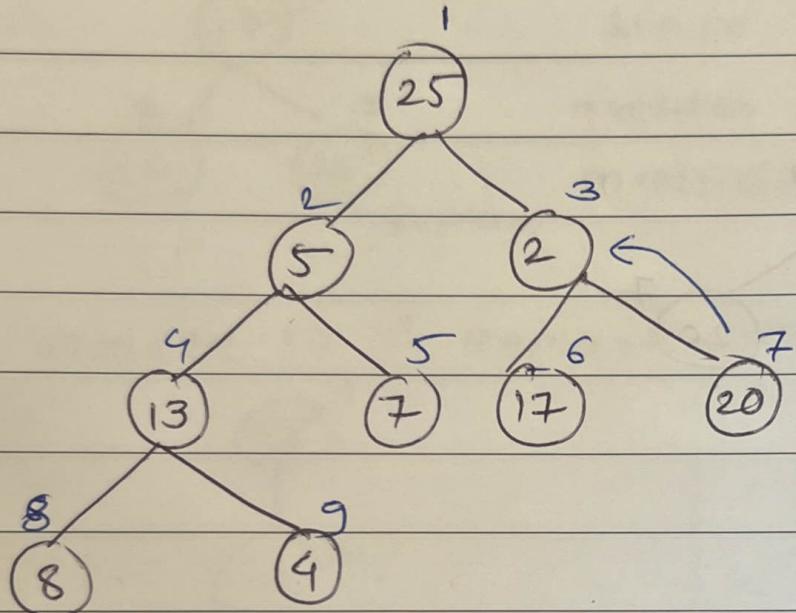
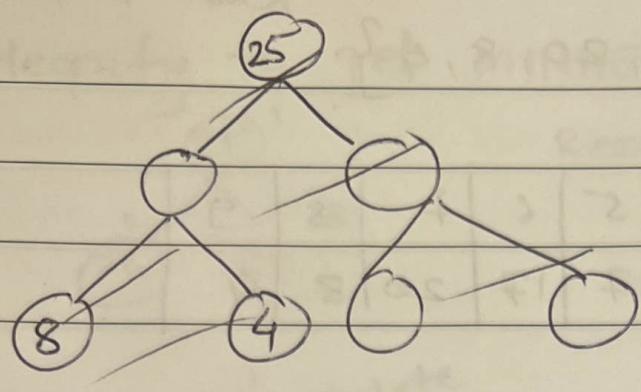
Starting from leaf node

① swap 25 with 13 at a[4]

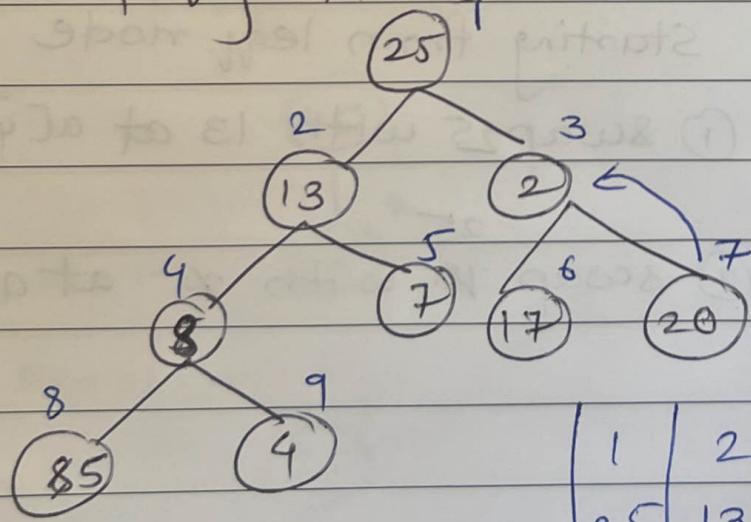


② swap 13 with 25 at a[1]

Date: / /



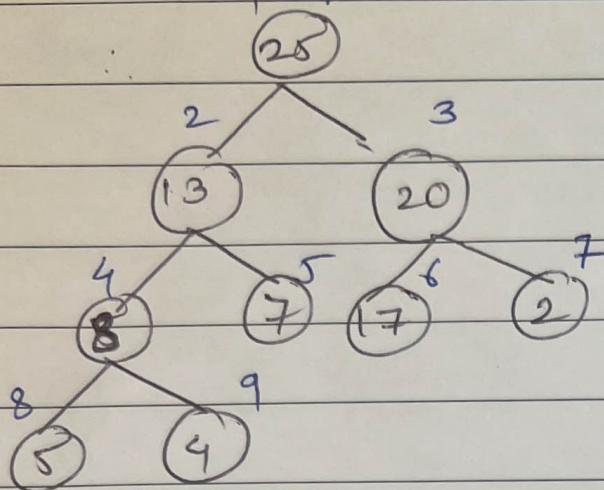
Heapify # 9[4]



1	2	3	4	5	6	7	8	9
25	13	2	8	7	17	20	85	4

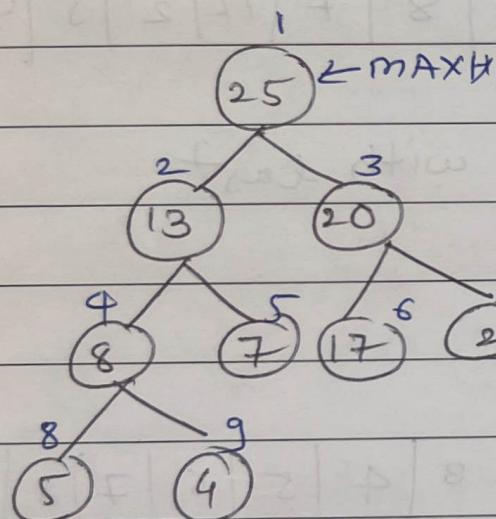
Date: / /

Heapify $a[3]$



1	2	3	4	5	6	7	8	9
25	13	20	8	7	17	2	5	4

since, all the leaf nodes are smaller than root node it's a MAXHEAP.



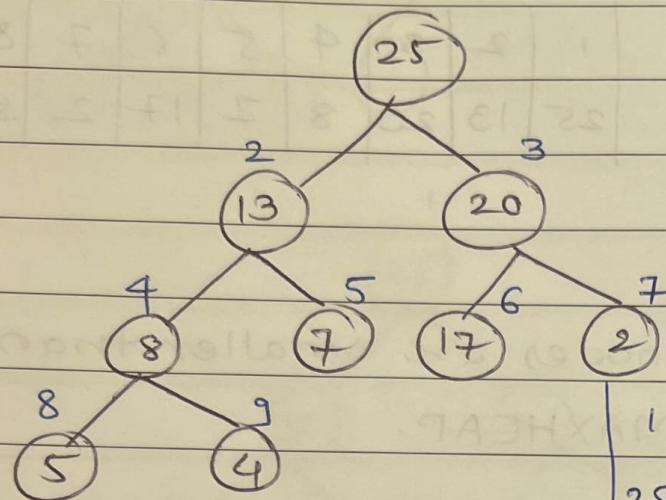
Final array :-

1	2	3	4	5	6	7	8	9
25	13	20	8	7	17	2	5	4

Date: / /

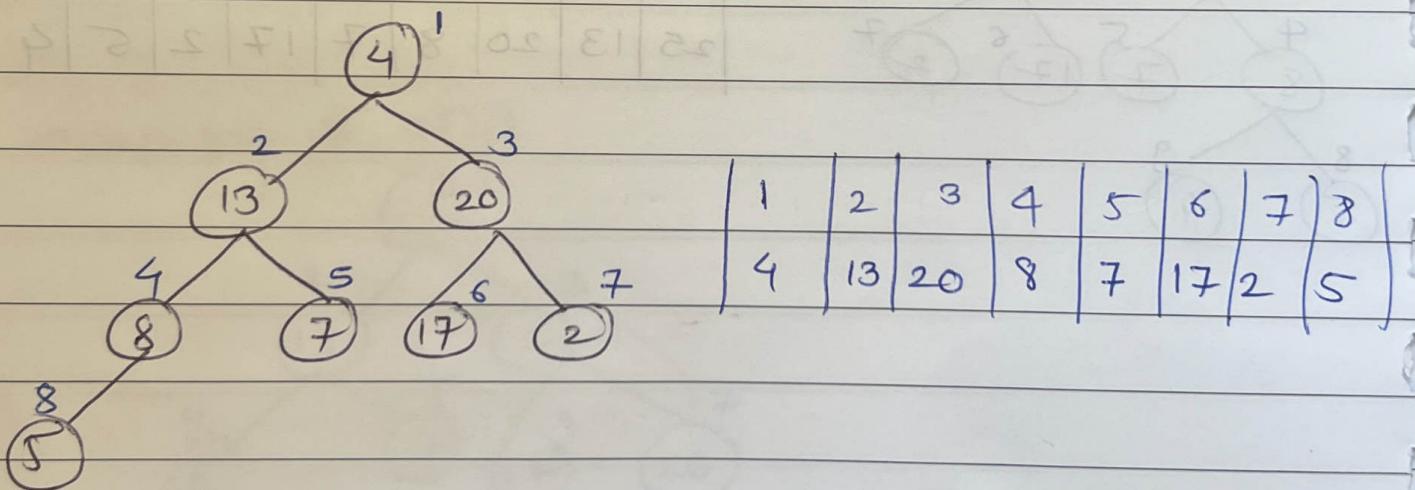
* Deleting max element from MAXHEAP.

MAXHEAP :-



1	2	3	4	5	6	7	8	9
25	13	20	8	7	17	2	5	4

- ① Delete, MAX → 25 & replace it with last element which is 4.



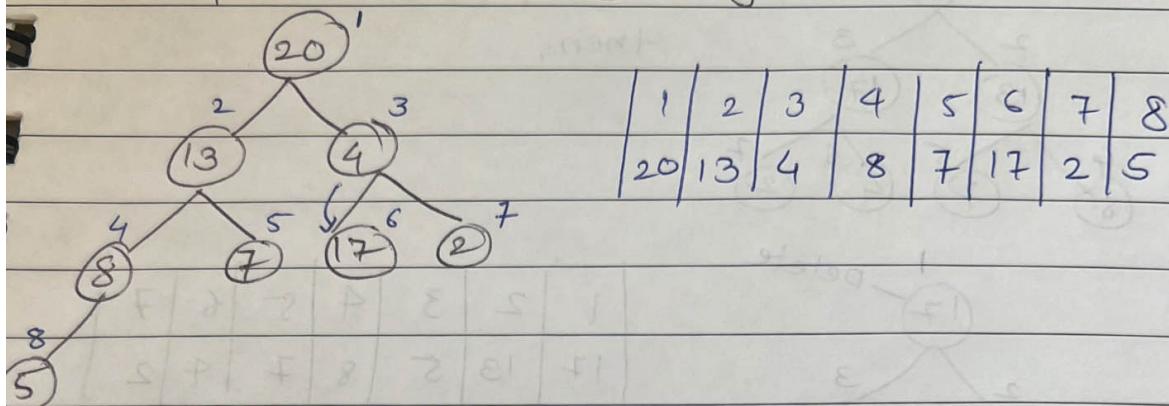
Heapify to get MAXHEAP.

∴ a[4] is already sorted. All the leaf
 ∴ a[3] ⇒ 20 already sorted : } nodes are
 ∴ a[2] ⇒ 13 → 11 → } smaller than
 root node.

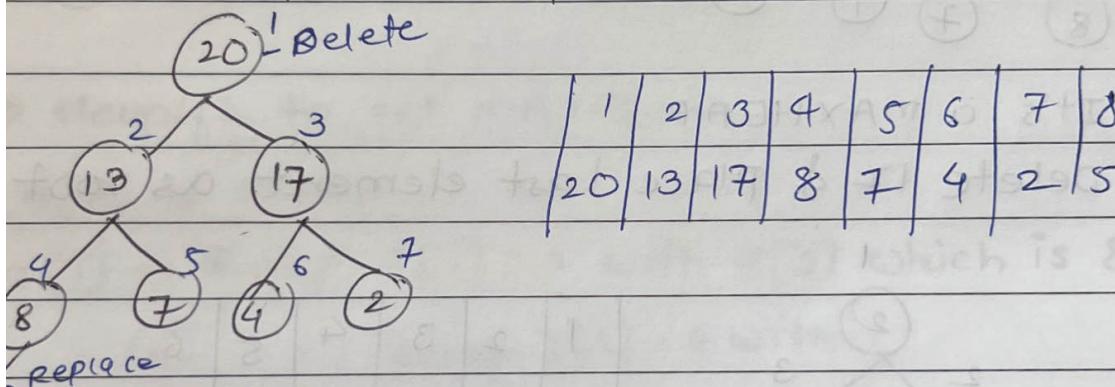
Date: / /

$a[0] \Rightarrow 4$.

swap $a[i]$ with $a[3]$ to get MAXHEAP.

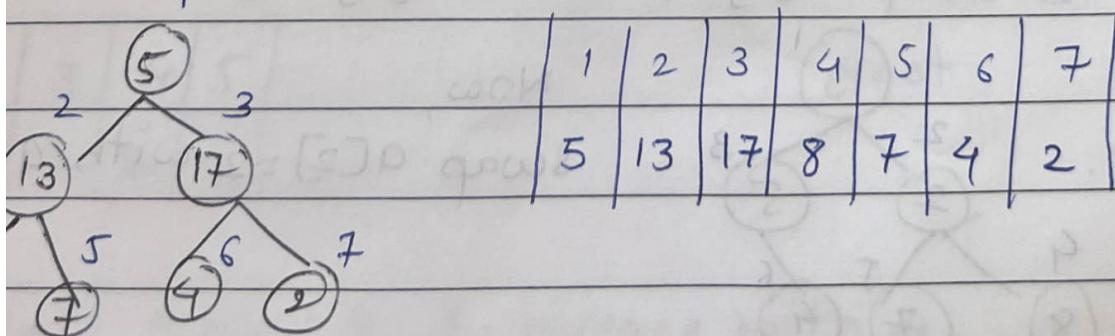


At $a[3]$, leaf nodes are greater than root node, will swap.



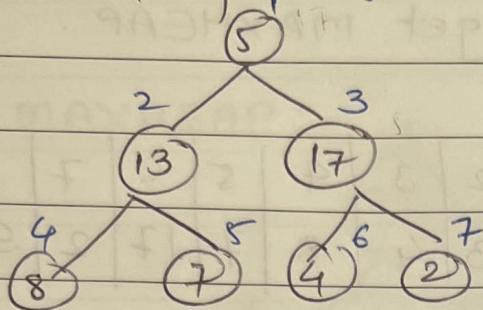
since, all the leaf nodes are smaller than root node, its MAXHEAP.

Delete MAX i.e. 20 & replace with last element



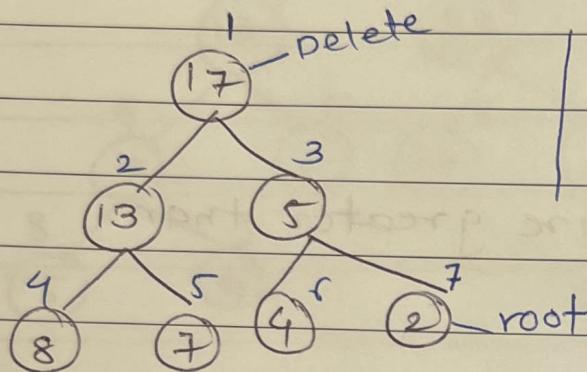
Date: / /

Heapify to get MAXHEAP.



swap $a[3] = 17$ with $a[1] = 5$

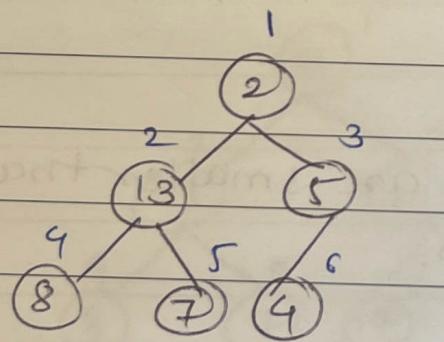
then,



1	2	3	4	5	6	7
17	13	5	8	7	4	2

- It's a MAXHEAP.

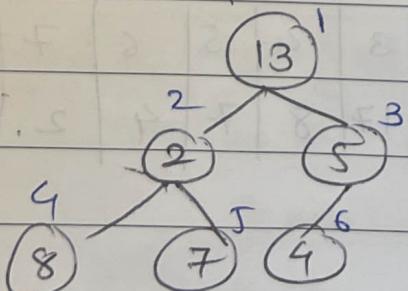
→ Delete 17 & place last element as root.



1	2	3	4	5	6
2	13	5	8	7	4

→ Heapify to get MAXHEAP.

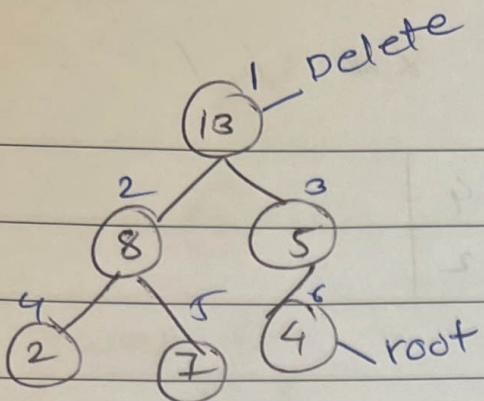
① swap $a[1] = 2$ with $a[2] = 13$.



Now,

swap $a[2] = 2$ with $a[4] = 4$

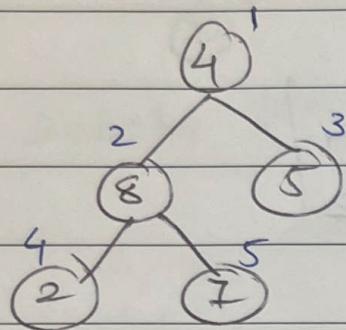
Date: / /



It's a MAXHEAP

1	2	3	4	5	6
13	8	5	2	7	4

* Delete 13 & replace with 4.

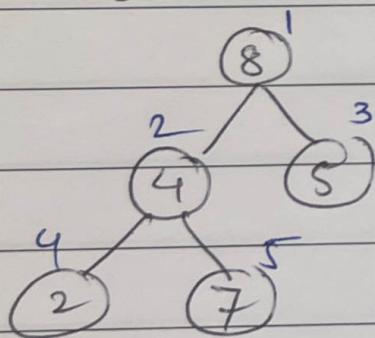


1	2	3	4	5
4	8	5	2	7

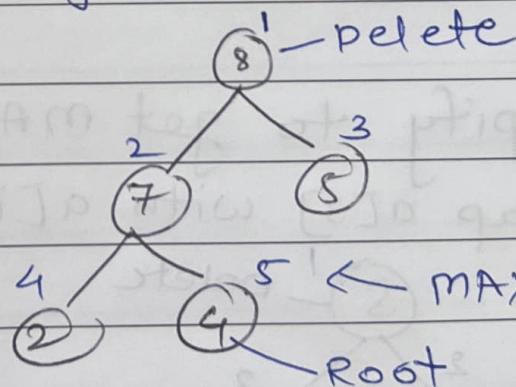
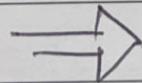
→ Heappify to get MAXHEAP

$a[2] \Rightarrow$ sortep

$a[1] \Rightarrow$ swap $a[1] = 4$ with $a[2]$ which is 8



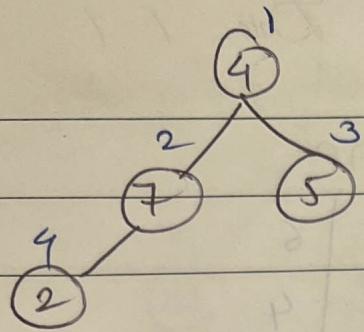
swap $a[2] = 4$ with
 $a[5] = 7$ to get MAXHEAP



1	2	3	4	5
8	7	5	2	4

* Delete MAX = 8 & replace with 4

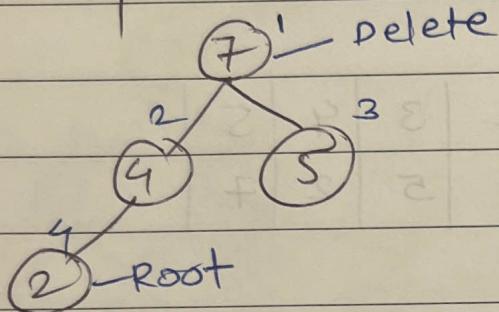
Date: / /



1	2	3	4
4	7	5	2

* Heapify to get MAXHEAP

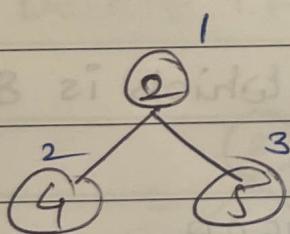
- swap $a[2] = 7$ with $a[1] = 4$



1	2	3	4
7	4	5	2

It's a MAXHEAP

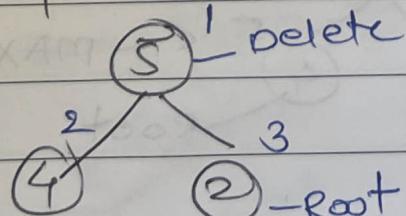
→ Delete max = 7 & replace with 2.



1	2	3
2	4	5

* Heapify to get MAX HEAP

swap $a[3] = 5$ with $a[1] = 2$

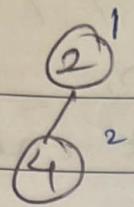


1	2	3
5	4	2

② -root It's a MAXHEAP.

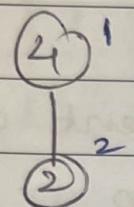
- Delete $a[1] = 2$ & Replace with 2

Date: / /



1	2
2	4

swap for MAXHEAP



1	2
4	2

- Delete $a[1] = 4$.

2	1
2	

- Final delete $a[0] = 2$ & MAXHEAP is empty now.