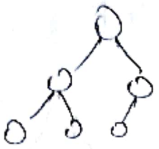
HEAP

↳ Complete Binary Tree.

↳ every level is full except possibly the last left to right



Heap is not BST

MAX heap: a CBT, the value of parent node is greater than value at its child. This property is true for every subtree.

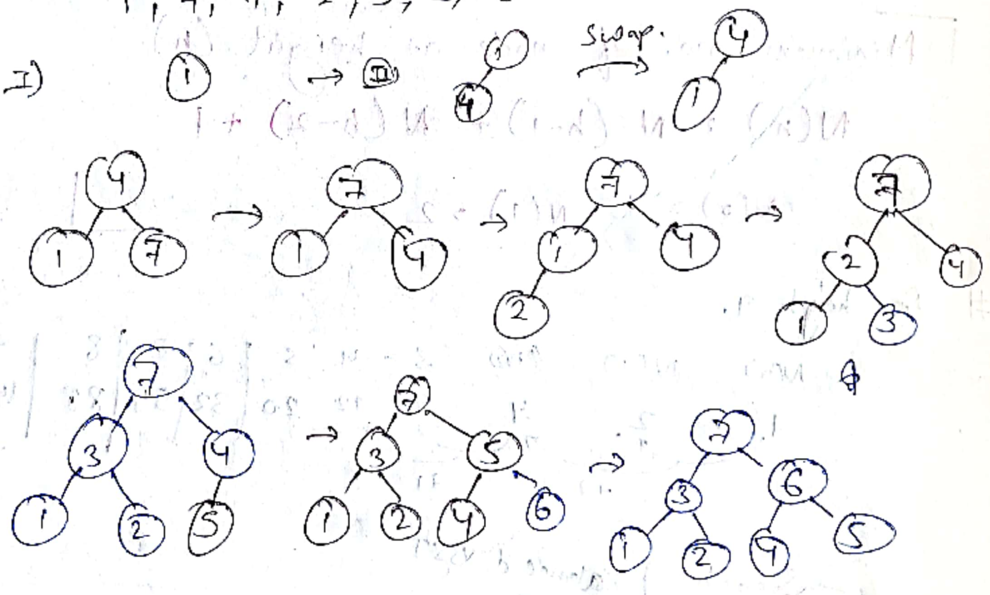
+

MIN Heap
min element

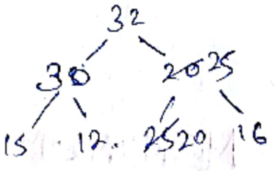
Insertion

1, 4, 7, 2, 3, 5, 6

I)



32, 15, 20, 30, 12, 25, 16 on insert in maxHeap.



Insert Algo.

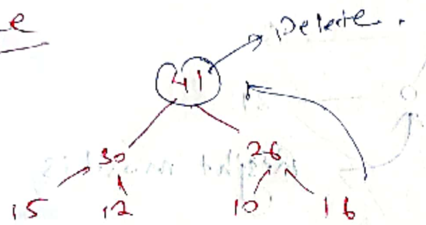
Max heap

```
Insert (a, n, x) {
    a[n] = x ; i = n
    while (i > 1 and a[⌊i/2⌋] < x) {
            Swap (a[i], a[⌊i/2⌋])
            i = ⌊i/2⌋ ;
    }
```

height = $\lfloor \log_2 n \rfloor$

max. Swap = $O(h)$
& insert = $O(\log_2 n)$

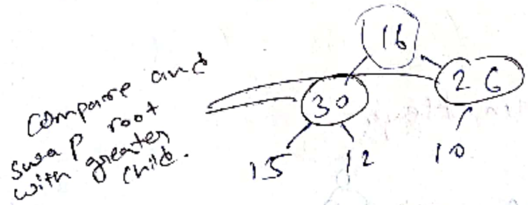
Delete



Priority Queue

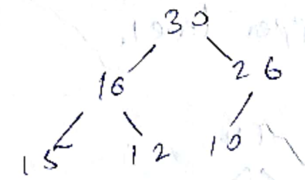
← max heap
∴ when deleted max will be deleted.

1. Delete the root
2. Copy last element as Root.

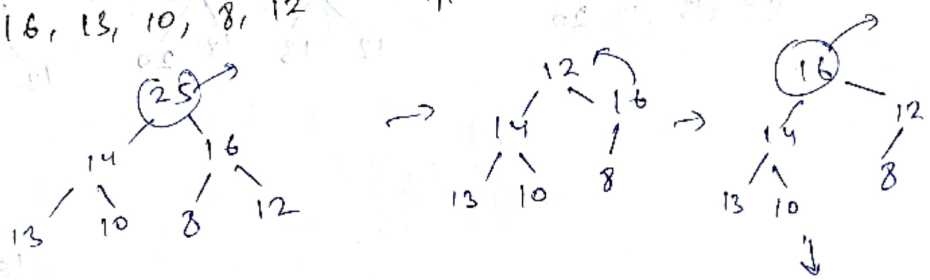


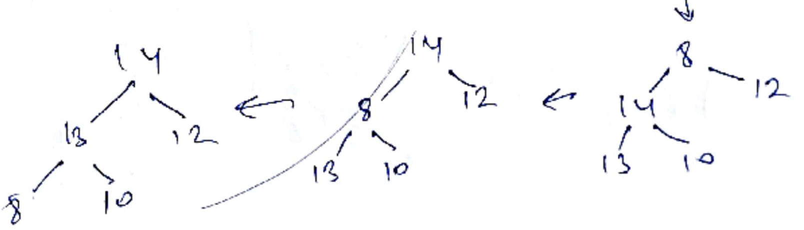
Now Adjust will be called if 1 element is Violating

3. Compare left & right child and find Max.
∴ Swap. And repeat till satisfy.



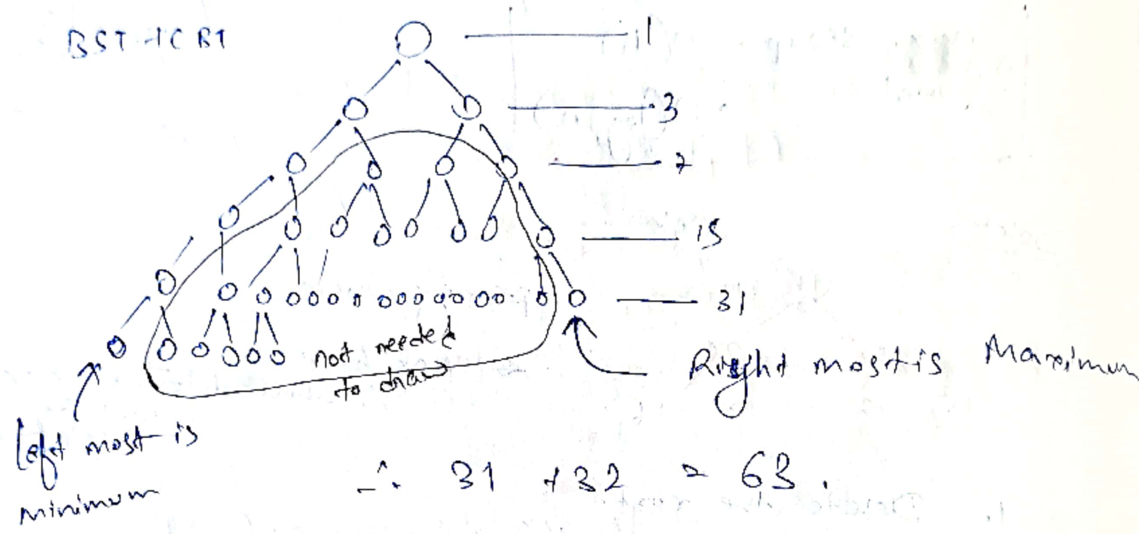
25, 14, 16, 13, 10, 8, 12   And 2 delete operation.





{14, 13, 12, 8, 10}

Suppose a BST with 37 distinct element is also a CBT. The tree is stored using array representation of binary heap trees. Assuming that the array indices starts with the 1 largest element of the tree is stored at index X and smallest element is stored at index y then X+y is ___



∴ 31 + 32 = 63.

$$\begin{array}{r} 63 \\ 64 \\ \hline 127 \end{array}$$

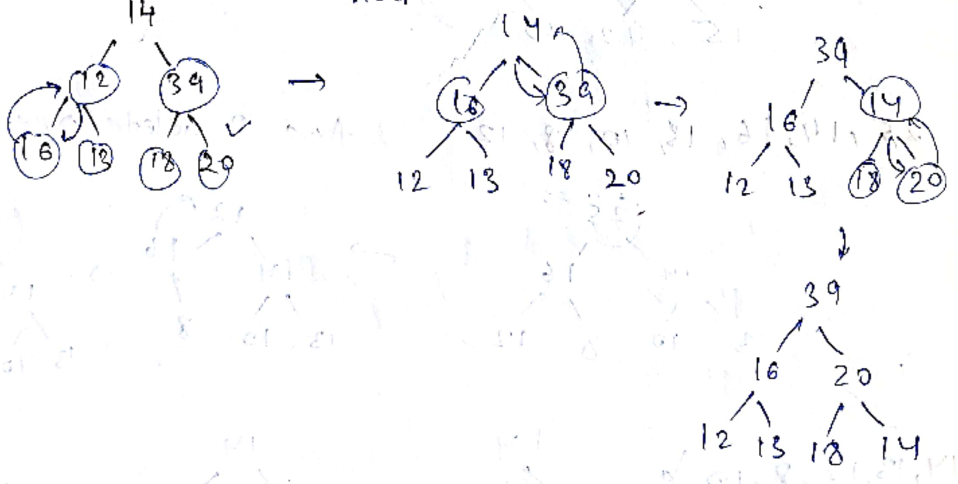
~~Heapify~~

Any tree ⟶ Max/Min Heap.

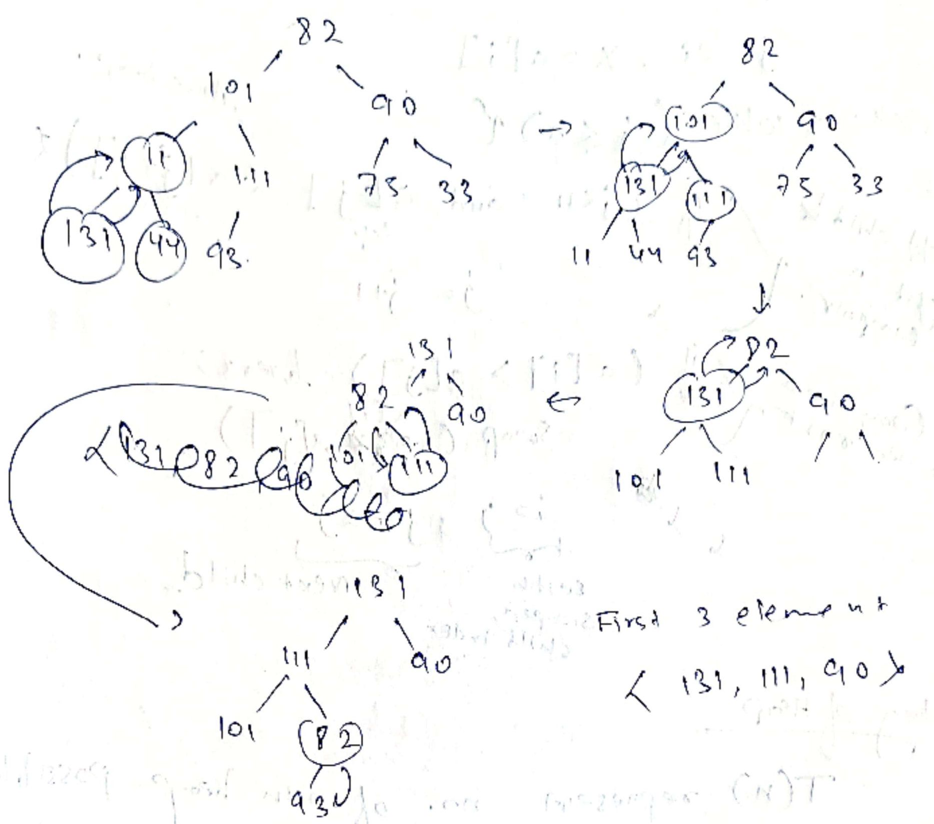
Start with non-leaf node →

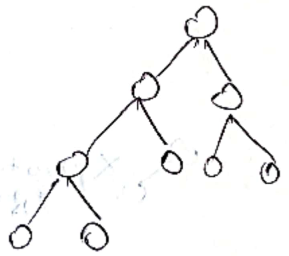
then adjust once. the see the other subtrees and adjust then move to upper level.

Max heap.



82, 101, 90, 11, 111, 75, 33, 131, 44, 93.





For CBT

Total node = 9

leaf node = 5 · ⇐ $\lceil \frac{9}{2} \rceil$

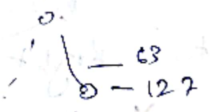
Property.

non leaf nod = $\lfloor \frac{9}{2} \rfloor$ = 4

min heap with 169 distinct element. No. of leaf node

∴ $\left\lceil \frac{169}{2} \right\rceil$ = 85

Jadi and Korman.



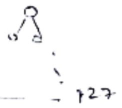
∴ last height = 169 − 127
= 42 leaf

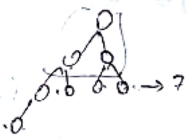
And. 127 − 63 − $\left\lceil \frac{42}{2} \right\rceil$

= 127 − 84 = 43

∴ 42 + 43 = 85 ✓

85





$$\begin{array}{r} 169 \\ -127 \\ \hline 42 \end{array}$$

$$\begin{array}{r} 127 \\ -21 \\ \hline 106 \end{array}$$

$$\begin{array}{r} 8 \\ -7 \\ \hline 1 \end{array}$$

$\frac{7-1}{6}$

7 − 1 − 3

4. $\underbrace{7-1}_{6-3}$ = 3

n = no. of nodes   i = index.

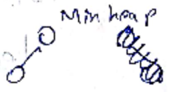
```
Algorithm Adjust (a,n,i) {
        j = 2i , x = a[i]
                                                      Max heap .
leaf check → while (j ≤ n) {
    left child &     if (j<n  and  a[j] < a[j+1]) {
    right  "                           left     right
    compare.                    j = j+1
                         }
    Compare with root    if (a[i] > a[j]) break;
                         swap (a[i], a[j])
                         i = j ; j = 2j
                      with swaped child index    next child.
  }
}
```

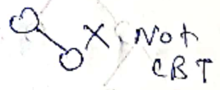
Counting of Heap

T(n) represent no. of min heap possible with n nodes.

n = 1   T(1) = 1

n = 2   T(2) = ~~2~~ 1

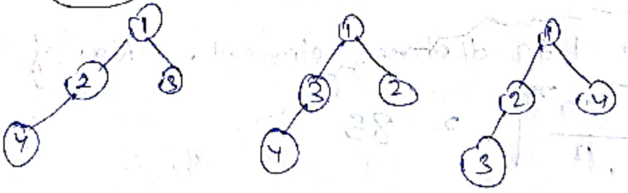




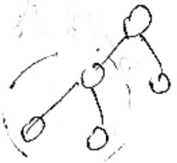
n = 3   T(3) = 2.

n = 4   T(4) = 3

1, 2, 3, 4 → Node left $^3C_2$



n = 5   T(5)



1 2 3 4 5

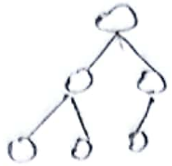
Left   Right

$^4C_3$ × 4 ← individual right combo with left

2 × 4 = 8

$$T(n) = {}^{n-1}C_k \; \underbrace{T(k)}_{\text{From CBT}} * T(n-1-k)$$

$T(6) = ?$



Left = 3 Right = 2

$$T(6) = {}^5C_3 \; \underset{\Downarrow}{T(3)} * T(2)$$

$$= \frac{5 \times \overset{2}{\cancel{4}} \times \cancel{3}}{\cancel{3} \times \cancel{2}} \times 2 * 1$$

$$= 20$$

- No. of labelled binary tree (n node) = $n! \times \underbrace{\text{no. of unlabelled BT with n node}}_{\frac{1}{n+1}\,{}^{2n}C_n}$.

- No. of BST with n node = no. of unlabelled BT with n node