CS & IT ENGINEERING

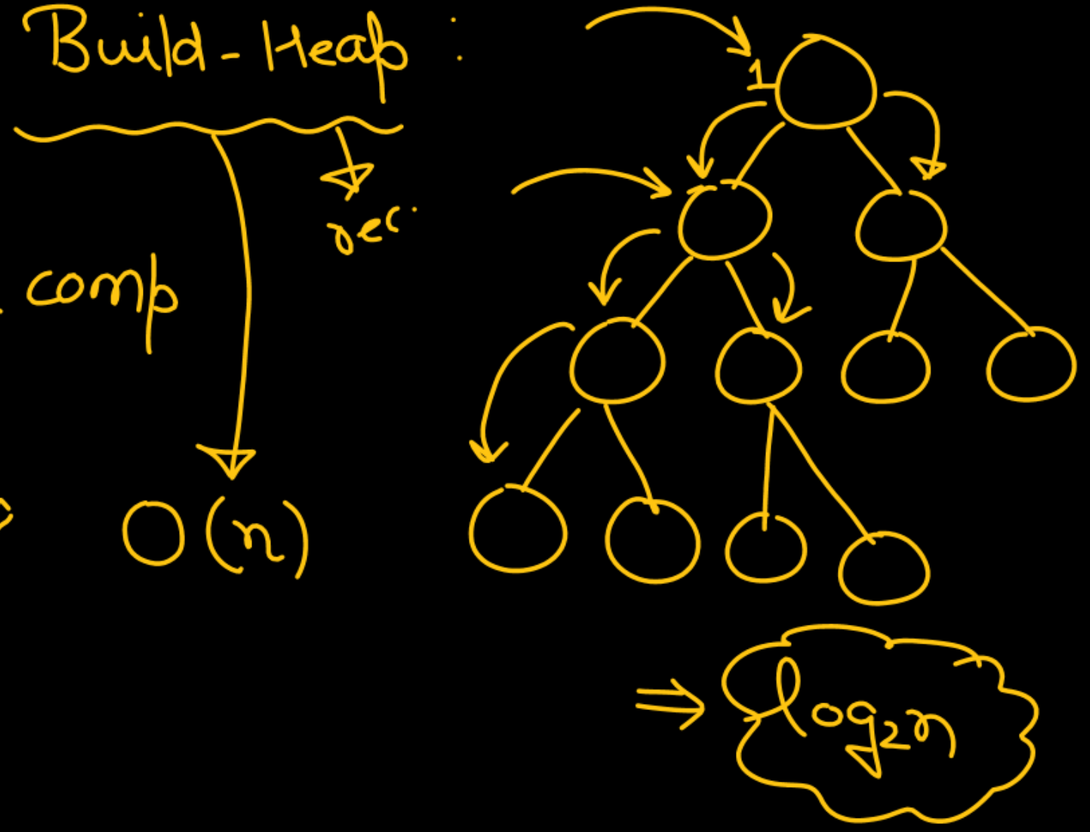
Data Structure

Tree
Chapter- 5
Lec- 09

By- Pankaj Sharma sir

Tree-IX

Build-Heap :



1) Heap Const. by inserting key one after another in a given order

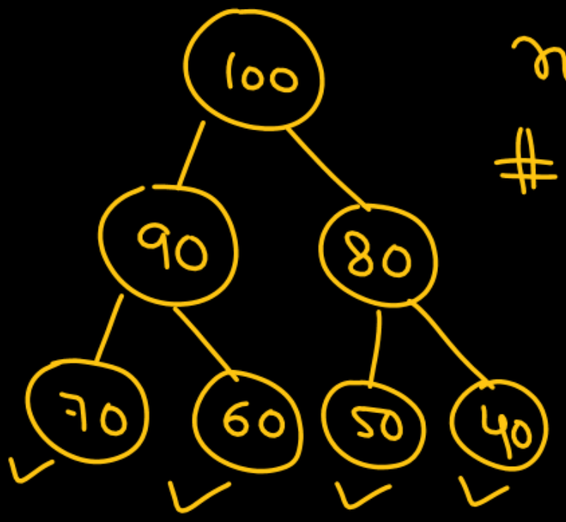
2) Array $\xrightarrow{\text{Convert}}$ Heap

OR

Build – Heap

} $O(n)$

3) Heap $\longrightarrow$ Array



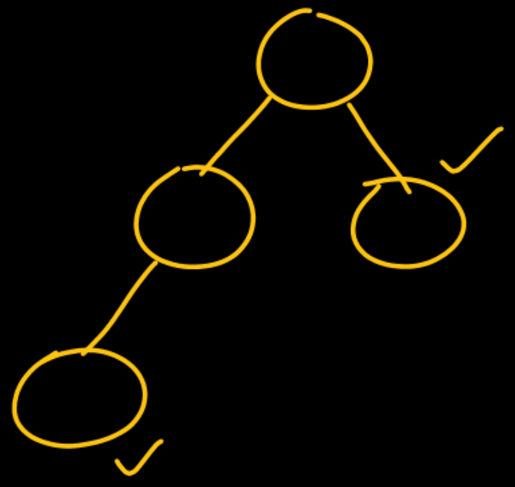
$n = 7$

#leaf node $= \lceil \frac{n}{2} \rceil$

#leaf node $= \lceil \frac{7}{2} \rceil = \lceil 3.5 \rceil$

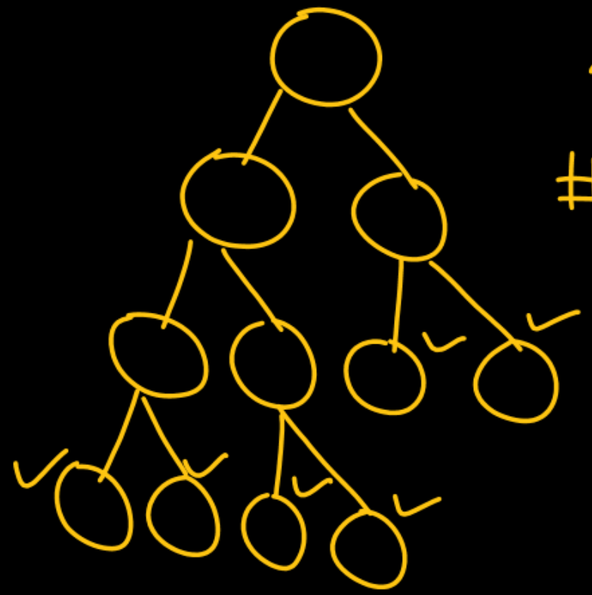
$= 4$

CBT



$n = 4$

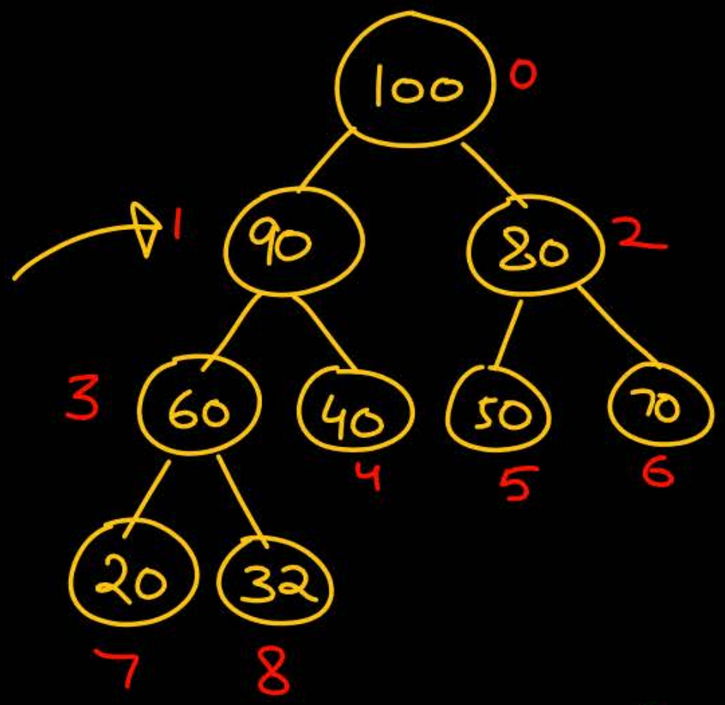
$\#\text{leaf nodes} = \left\lceil \frac{4}{2} \right\rceil = 2$



$n = 11$

$\#\text{leaf nodes} = \left\lceil \frac{11}{2} \right\rceil = \lceil 5.5 \rceil$

$= 6$

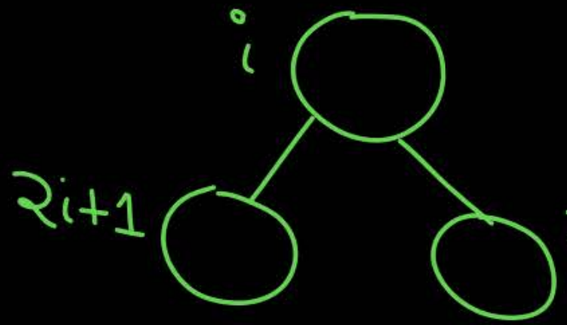


100	90	80	60	40	50	70	20	32
0	1	2	3	4	5	6	7	8

index of key 100 ⇒ 0

" " its left child = 2×0+1 = 1

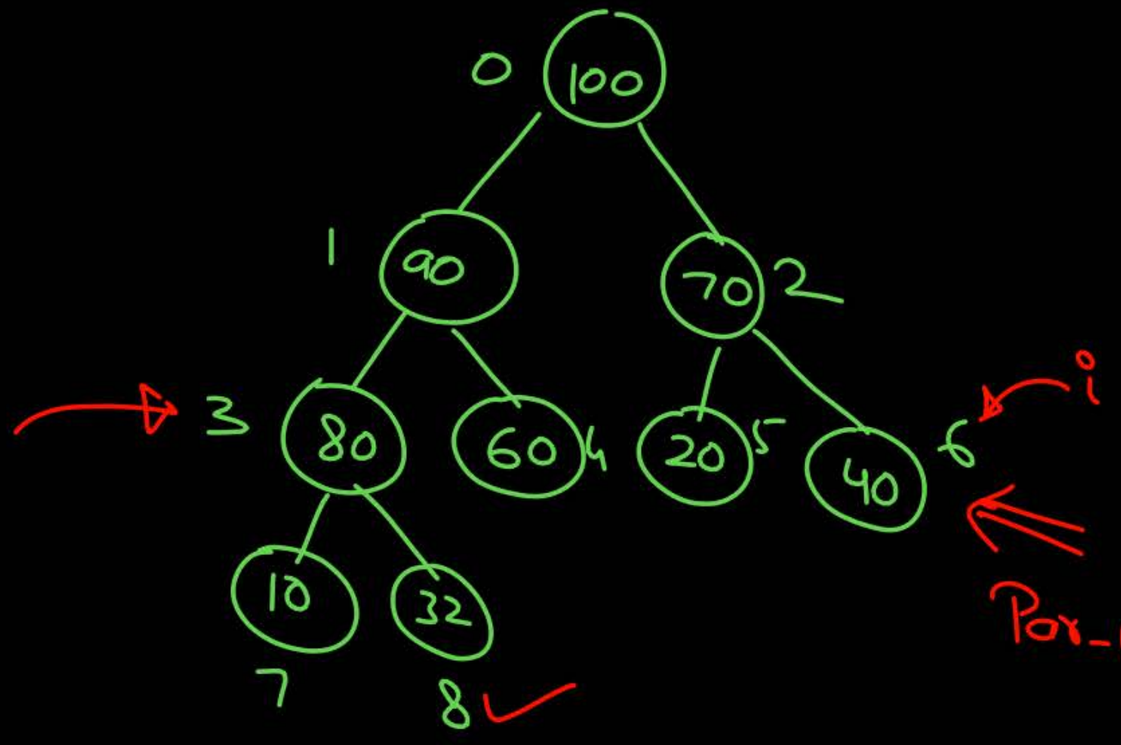
" " " Right child = 2×0+2 = 2



If u know index of a node we can find **Parent-node index**

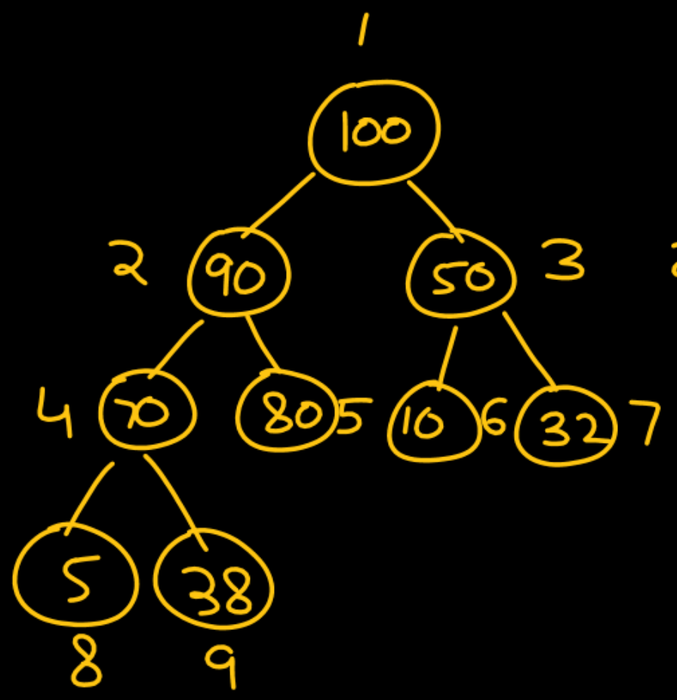
Node $\Rightarrow$ index i

Par_Node $\Rightarrow \left\lfloor \frac{(i-1)}{2} \right\rfloor$



Par_index $= \left\lfloor \frac{6-1}{2} \right\rfloor = \left\lfloor \frac{5}{2} \right\rfloor = [2.5] = 2$

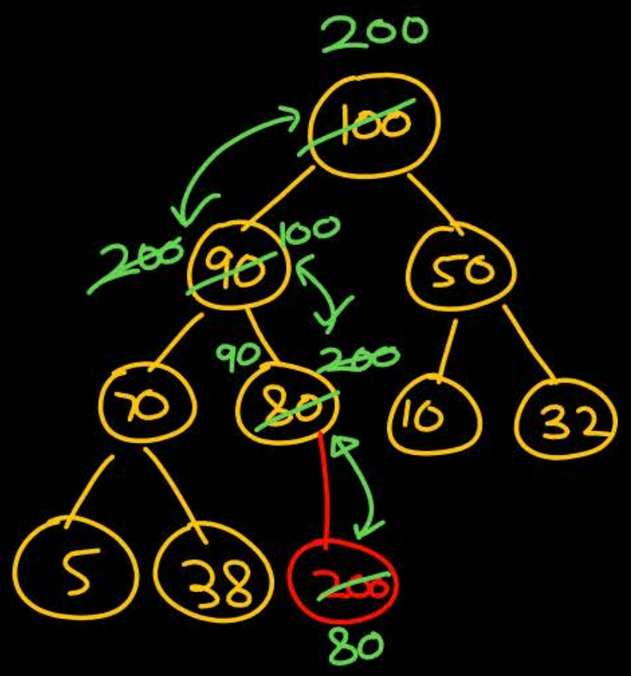
$\frac{8-1}{2} = \frac{7}{2} = [3.5] = 3$



100	90	50	70	80	10	32	5	38
1	2	3	4	5	6	7	8	9

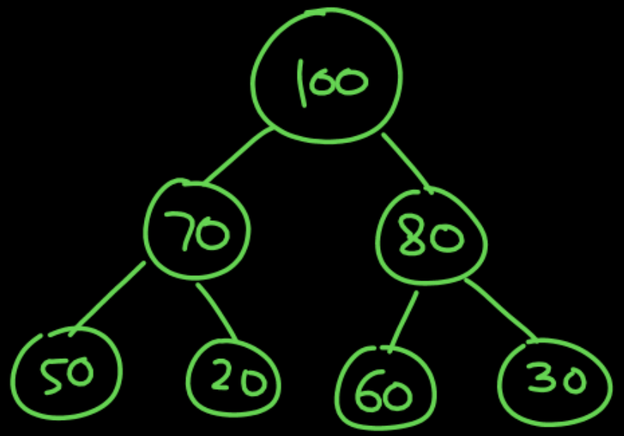
$$\text{Node} \Rightarrow \underset{i}{\text{index}}$$

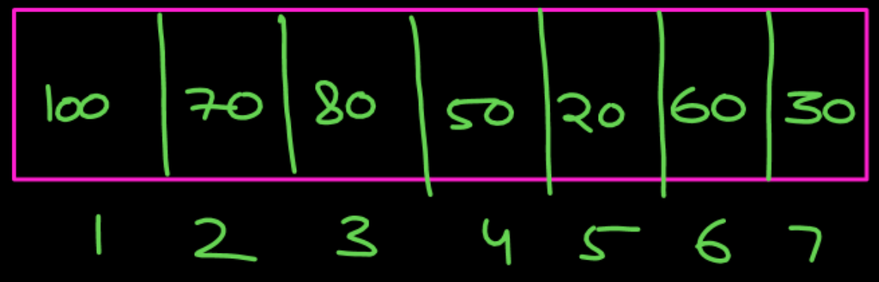
$$\text{Par\_Node} \Rightarrow \left\lfloor \frac{i}{2} \right\rfloor$$



insert 200 in given max-heap

$O(\log_2 n)$



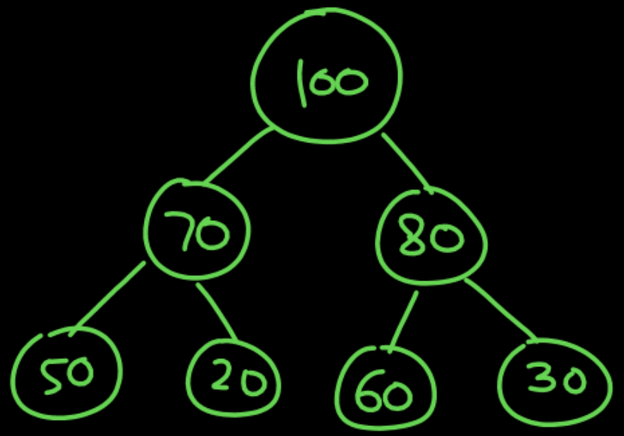


Constant time

Find the maximum element in a given max-heap

$\Rightarrow$

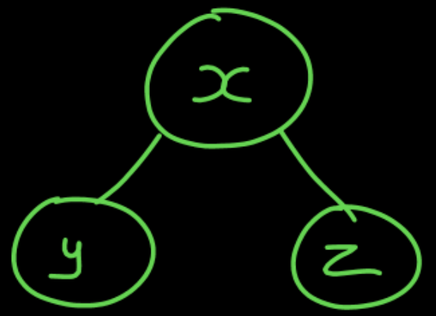
return A[1]



100	70	80	50	20	60	30
1	2	3	4	5	6	7

Find the minimum element in a given max-heap.

Max - heap



Min value
$\Rightarrow x$?
y ?
z ?

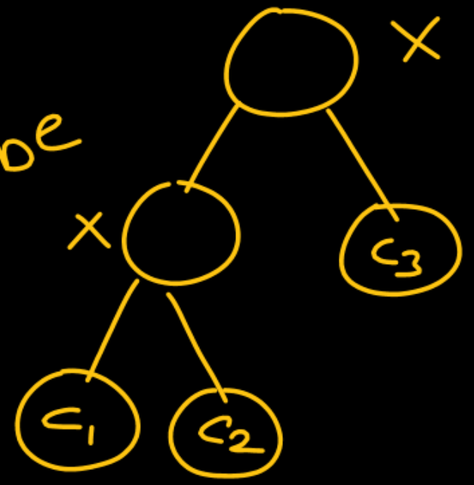
It can not be x ✓

$x > y$
$x > z$

$y, z < x$

Min can be either y or z ✓

✓ In a Max-heap Min element must be a leaf node



Min c_1, c_2, c_3

10 element

Comparision ⇒ 9

$\lceil \frac{n}{2} \rceil$

$\#comp = \lceil \frac{n}{2} \rceil - 1$

Max-heap is given

(i) Find_Max — constant time

(ii) Find_Min — $O(n)$

(iii) Insert a key — $O(\log_2 n)$

(iv) Search a key - $O(n)$

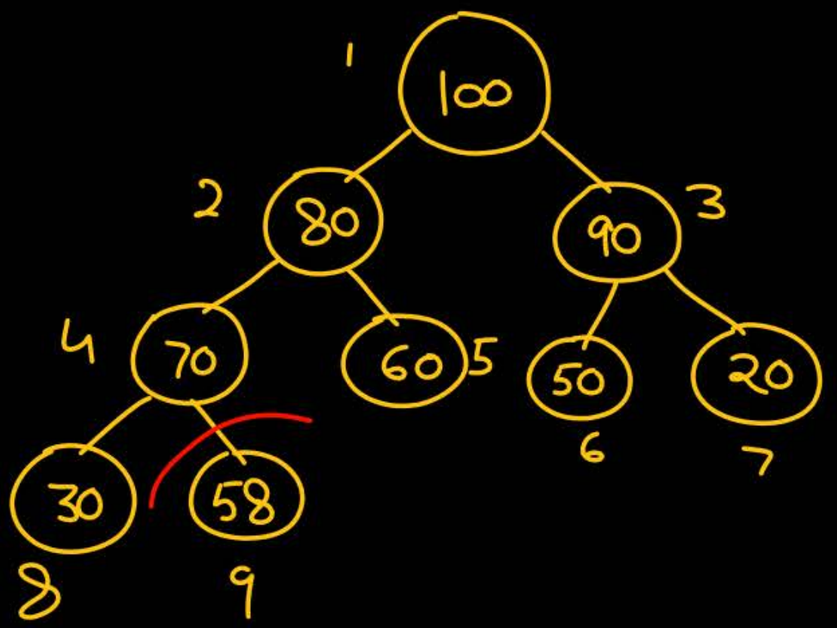
Min-heap is given

(i) Find_Min $\Rightarrow$ Constant time

(ii) Find_Max $\Rightarrow$ $O(n)$

(iii) Insert a key $\Rightarrow$ $O(\log_2 n)$

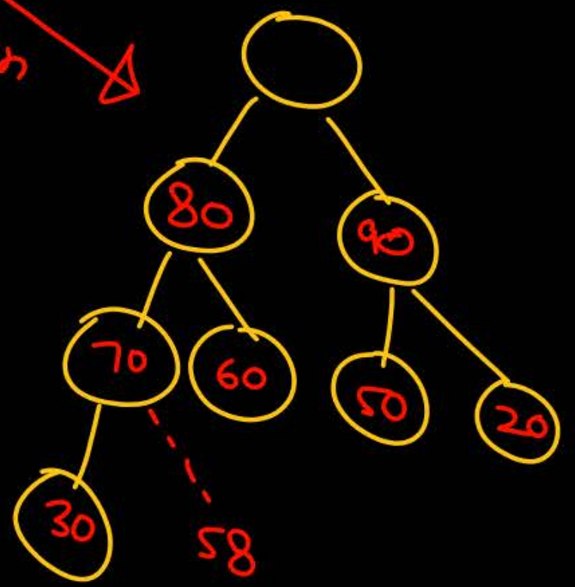
(iv) Search a key $\Rightarrow$ $O(n)$



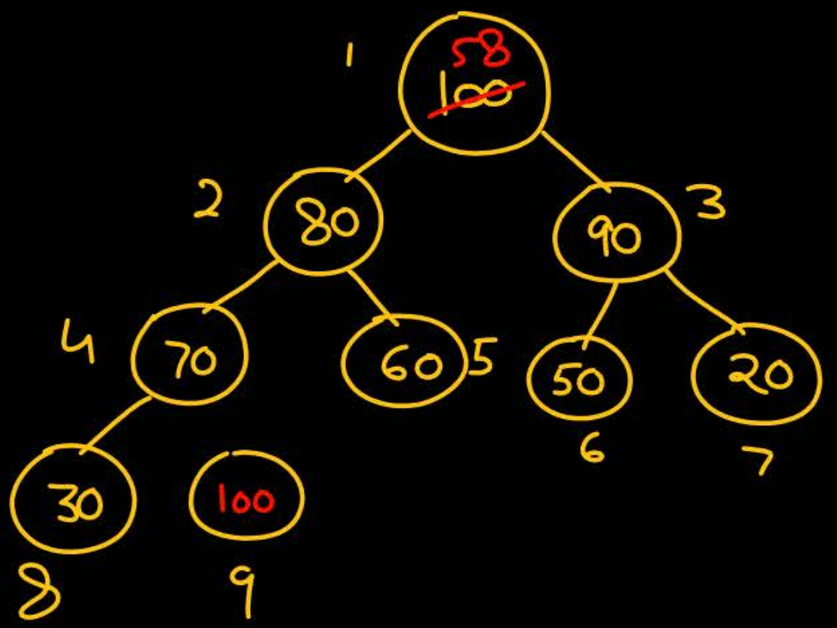
100	80	90	70	60	50	20	30	58
1	2	3	4	5	6	7	8	9

Extract_Max :

Structure after deletion

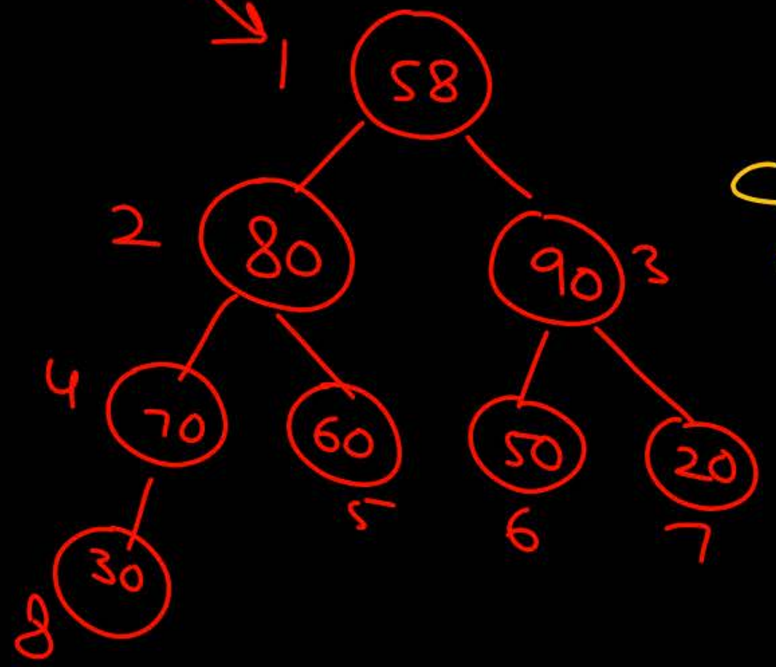


CBT with 8 node



Extract_Max : $n = 9$

Heapify

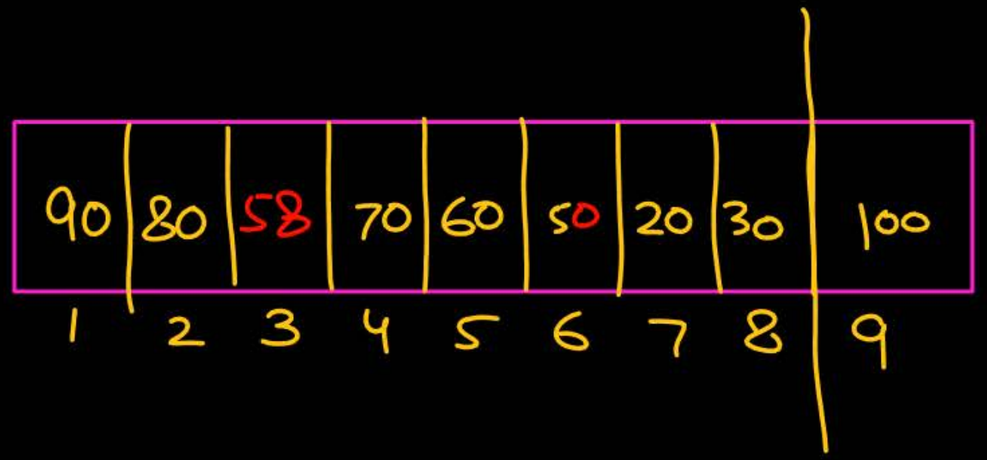


CBT with 8 node

$A[1] \leftrightarrow A[n]$

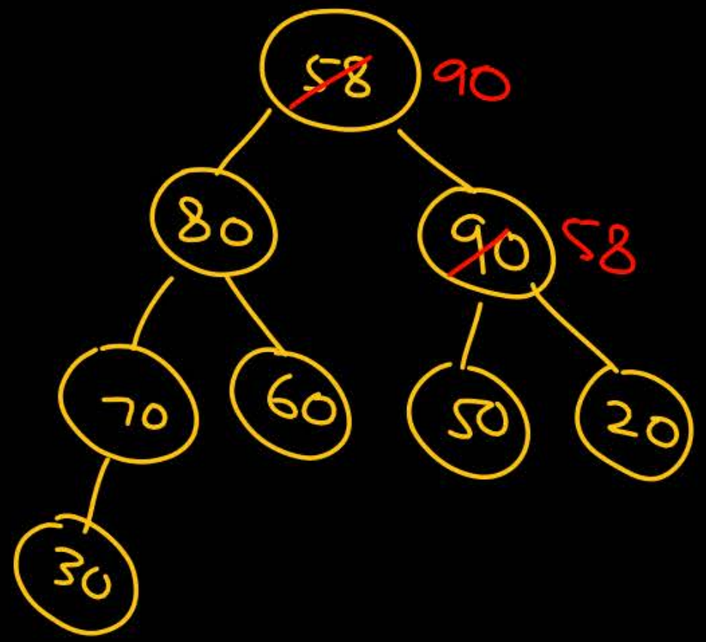
$n = n-1$

Heapify(A, 1, n)



Extract_Max :

$n = 9$



CBT with 8 node

distinct 3 keys are given 10, 20, 30

20 Problems

How many max-heaps are possible?

3 nodes CBT ⇒

10, 20

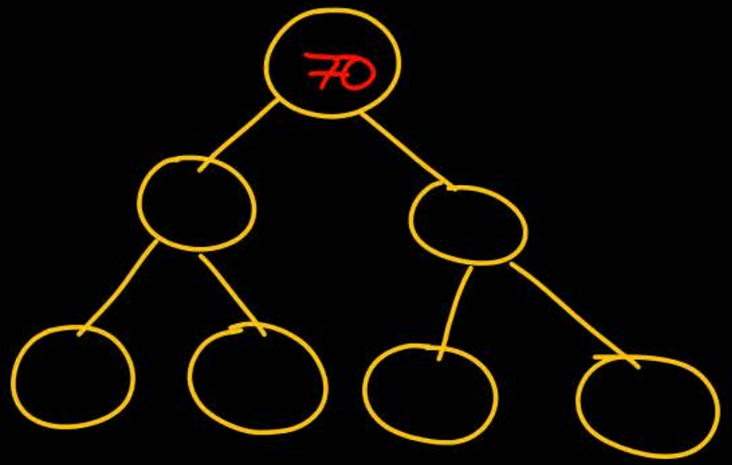
Max-heap

$n = 3 \Rightarrow 2$

10, 20, 30, 40, 50, 60, 70

Max-heaps are possible

Every node



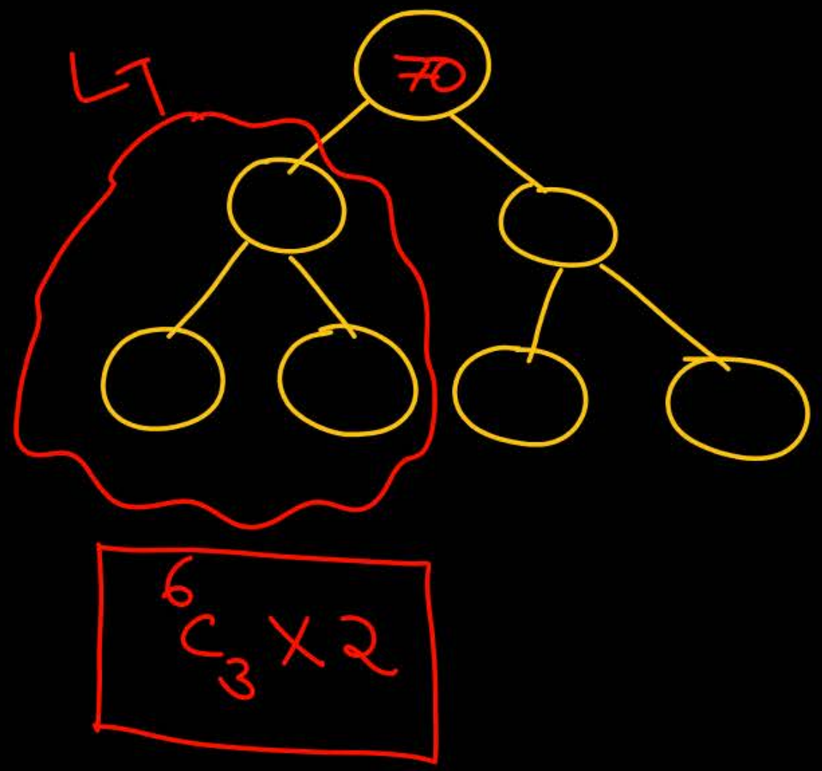
Out of 7 keys

⇒ Root node ⇒ 1 way

(Maximum value)

10, 20, 30, 40, 50, 60, 70

Max-heaps are possible



keys remaining

$= 7 - 1 = 6$ keys

① keys in left subtree = 3

For left subtree

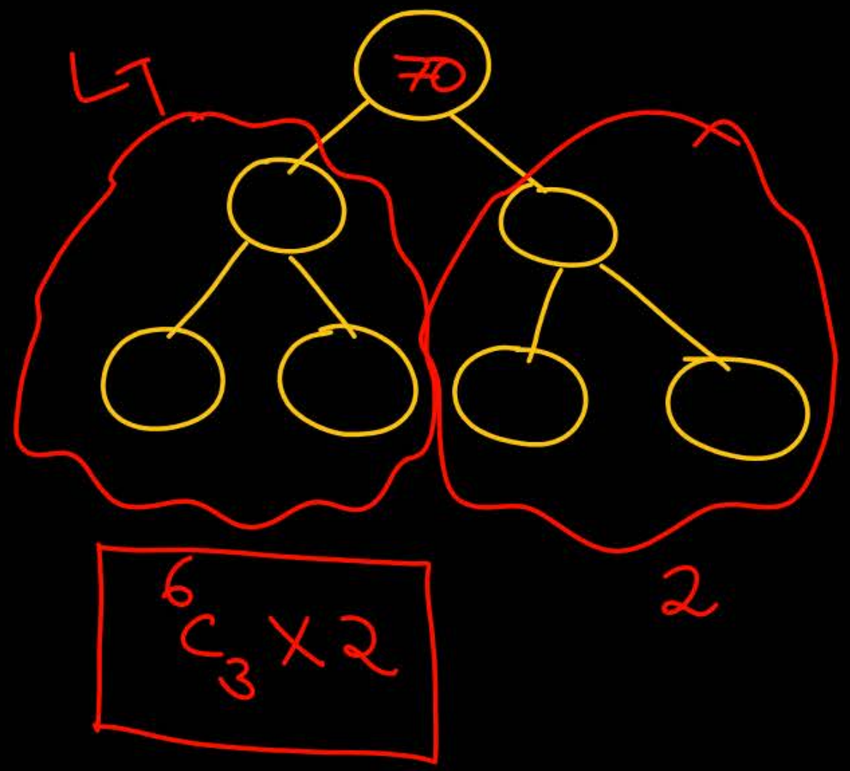
⇒ any 3 keys can be picked

No. of Heaps possible with these 3 selected key = 2

${}^{6}C_{3}$

10, 20, 30, 40, 50, 60, 70

Max-heaps are possible



keys rem = 3

RT

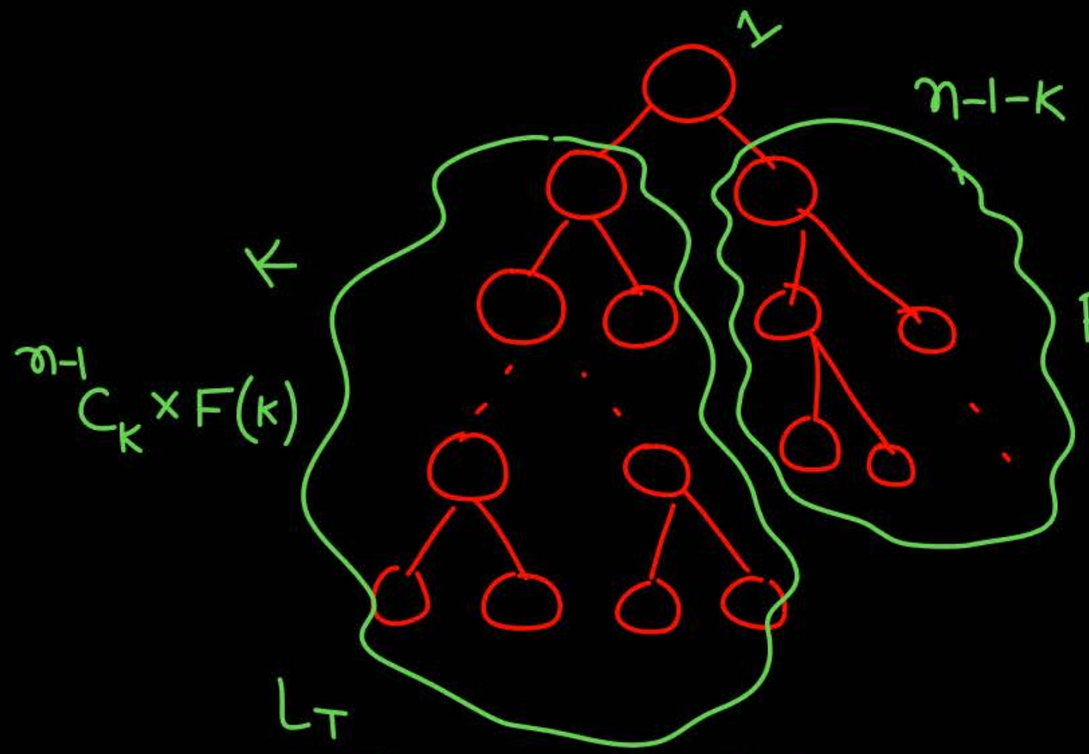
heaps possible with 3 keys = 2

$$\Rightarrow 1 \times (^6C_3 \times 2) \times (2)$$

$$= \frac{6!}{3!3!} \times 2 \times 2$$

$$= \frac{6 \times 5 \times 4 \times 3!}{3!3!} \times 2 \times 2 = 80$$

Max heaps with n distinct keys ?



let

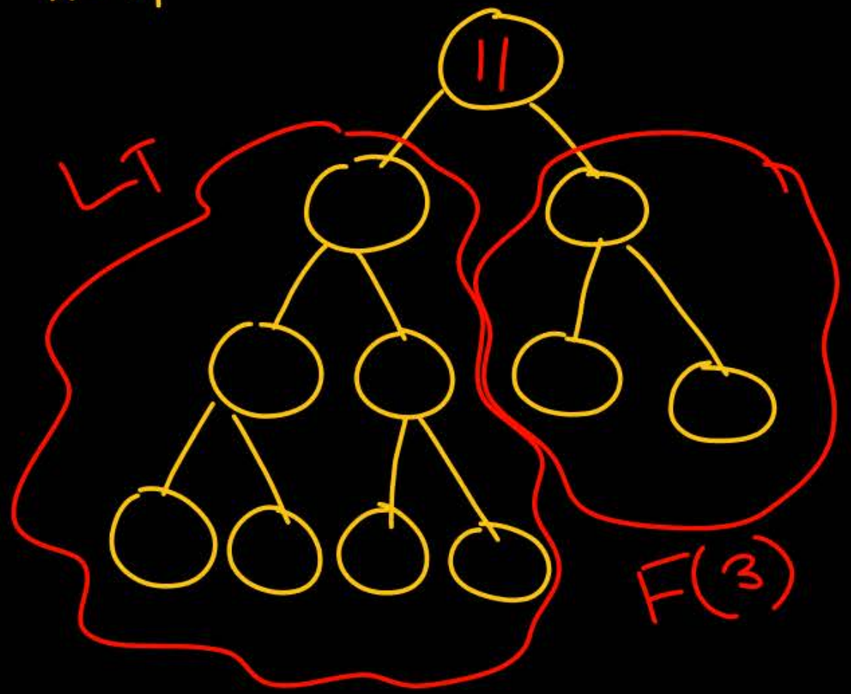
F(n) : No. of max heaps possible with n distinct keys

Root → 1 way (Max. key)

⇒ n-1 keys में से k को select

$$F(n) = 1 \times {}^{n-1}C_k \cdot F(k) \times F(n-1-k)$$

$n = 11$



1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

Remain. keys = 10

we can select any 7 keys for LT

$= {}^{10}C_7$

$$F(11) = 1 \times \left\{ {}^{10}C_7 \times F(7) \right\} \times F(3)$$

$F(7)$ → 80

$F(3)$ → 2

7 keys

$F(7)$

H.W

15 distinct
20 distinct

$F(11) = {}^{10}C_7 \times 80 \times 2$


THANK YOU GW
SOLDIERS !