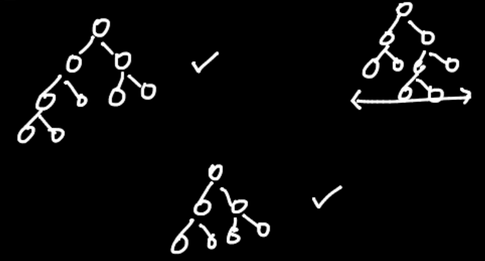
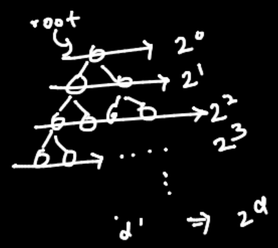
CBT



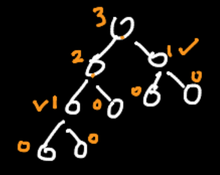
'n'

$h(CBT_n) = \log_2 n$



'n' =

'h' =



$\Rightarrow \left\lfloor \frac{n}{2^{h+1}} \right\rfloor$

$\frac{9}{2^{2+1}} = \left\lfloor \frac{9}{8} \right\rfloor = 1$ ⇒ $n = 9$

✓ 2 ⇒ h=1
✓ 5 ⇒ h=0
✓ 1 ⇒ h=3 ✓
✓ 1 ⇒ h=2

$h=1 \Rightarrow \left\lceil \frac{9}{2^{1+1}} \right\rceil$

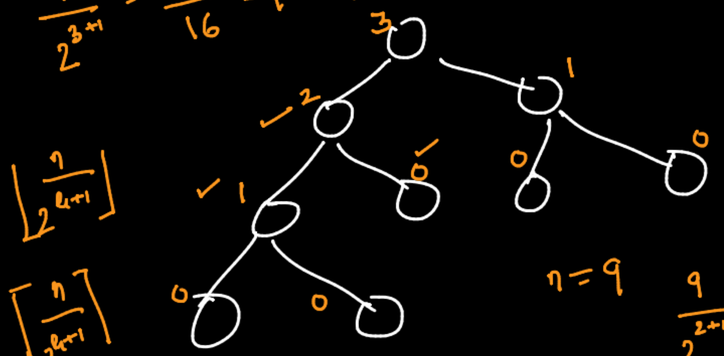
$\Rightarrow \left\lceil \frac{9}{4} \right\rceil$

$\Rightarrow 3$

$\frac{9}{2^{3+1}} = \frac{9}{16} = [0\cdots]$

$\left\lfloor \frac{n}{2^{h+1}} \right\rfloor$

$\left\lceil \frac{n}{2^{h+1}} \right\rceil$



$n = 9$

$\frac{9}{2^{2+1}} = \frac{9}{8} = [1\cdots]$

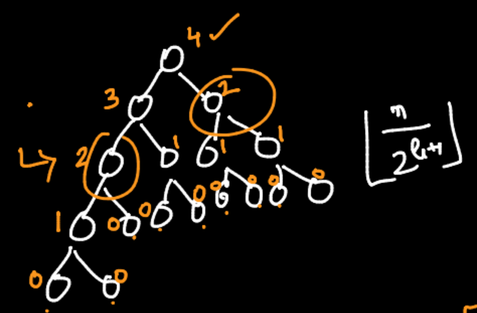
$\frac{9}{2^{0+1}} = \frac{9}{2} = [4\cdots] \Rightarrow \frac{n}{2^{h+1}}$

$= 5$

$\frac{9}{2^{1+1}} = \frac{9}{4} = [2\cdots]$

$n = 17$

$\left\lceil \frac{n}{2^{l+1}} \right\rceil$



$\left\lceil \frac{n}{2^{l+1}} \right\rceil$

$l = 0 \Rightarrow \frac{17}{2^1} = \frac{17}{2} = \lceil 8.\cdot \rceil = 9$ ✓

$l = 2 = \frac{17}{2^{2+1}} = \frac{17}{8} = \lceil 2.\cdot \rceil = 2 \quad \frac{17}{2^{4+1}} = \frac{17}{32} = \lceil 0.\cdots \rceil = 1$

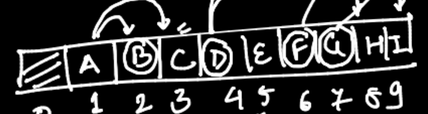
$\sim \frac{n}{2^{l+1}}$

✓ Representation of CBT

⇒

8, 9

Array



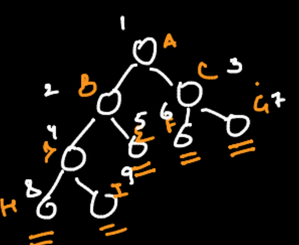
$6/2 = 3$

$\lceil 7/2 \rceil = 3$

for any node @ in idx → left = 2*i
→ right → (2*i + 1)

for any node @ in idx ⇒ $\lfloor i/2 \rfloor$

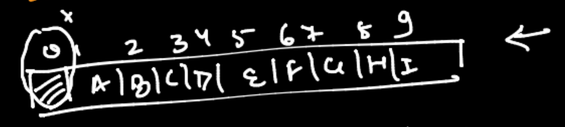
$4/2 = 2$



Logical

non-leaf

$\lfloor n/2 \rfloor$ to 1



starting pt. of leaf = $\lfloor n/2 \rfloor + 1 = \lfloor 9/2 \rfloor + 1$

to n-1 $= \lfloor 4.\cdots \rfloor + 1$

$= 4 + 1 = 5$

Heaps / Priority Queue

		Insertion	find min	deletemin
unsorted	Arrays	O(1)	O(n)	O(n)
	Linked List	O(1)	O(n)	O(n)
sorted	Arrays	O(n)	O(1)	O(1)
	Linked List	O(n)	O(1)	O(1)
unbalanced BST		O(n)	O(n)	O(n)
Balanced BST		$O(\log_2 n)$	$O(\log_2 n)$	$O(\log_2 n)$

Heap / PQ

- insertion $O(\log_2 n)$
- deletion min/max $O(\log_2 n)$
- find min/max $O(1)$

- min heap
- max heap

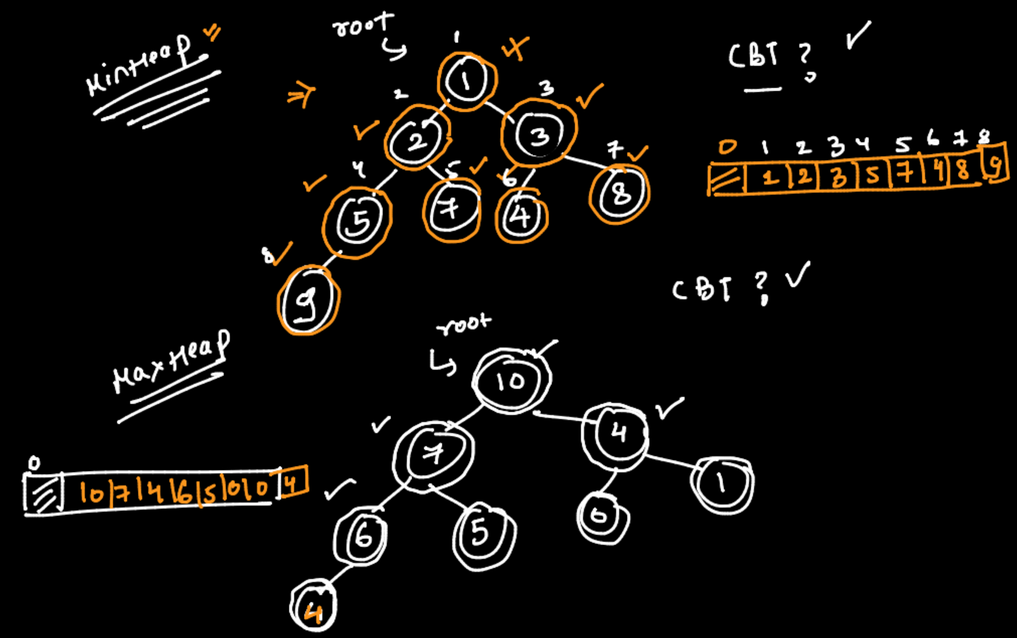
Heap (MinHeap)

↳ CBT

↳ that satisfies heap property

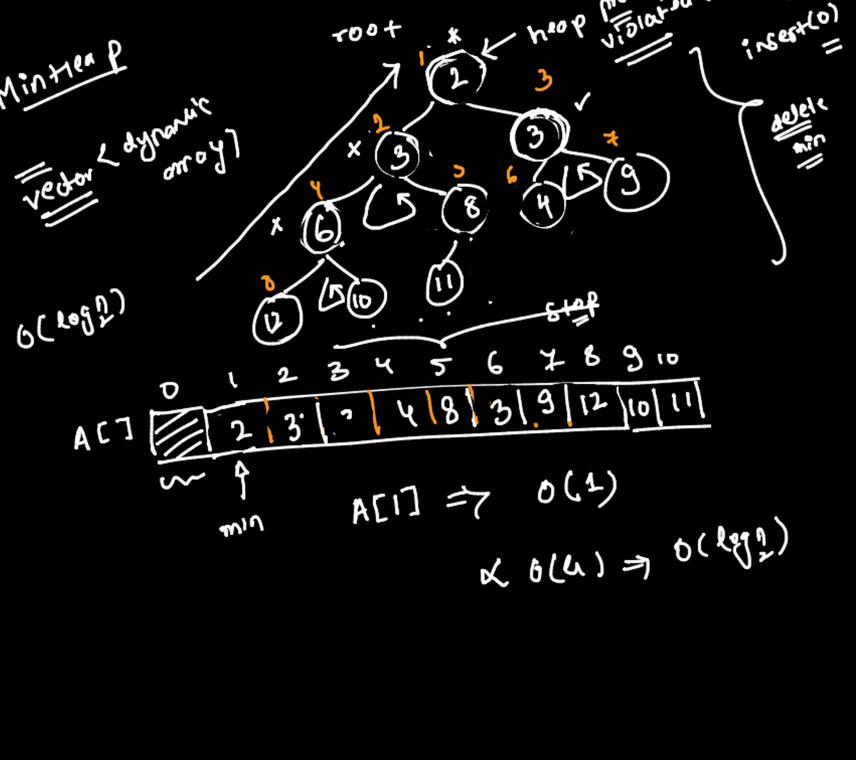
⇒ (MinHeap) for each node its value should ≤ its child nodes

(MaxHeap) for each node its value should ≥ its child node



MinHeap

vector < dynamic array >



A[1] ⇒ O(1)

∝ O(1) ⇒ O(log2 n)

Heapify

log2 n ∝ O(1)

0 1 2 3 4 5 6
70 20 30 40 50 60
20 70 70
40