Lec-41

24/12/24 7:16 PM

Heap | Priority Queue

Queue → FIFO

data → priority

| 10 | 20 | 30 | 40 | 50 | 60 |

Rank [Smaller value ↑ higher]

marks [higher → higher]

Heap → 100%

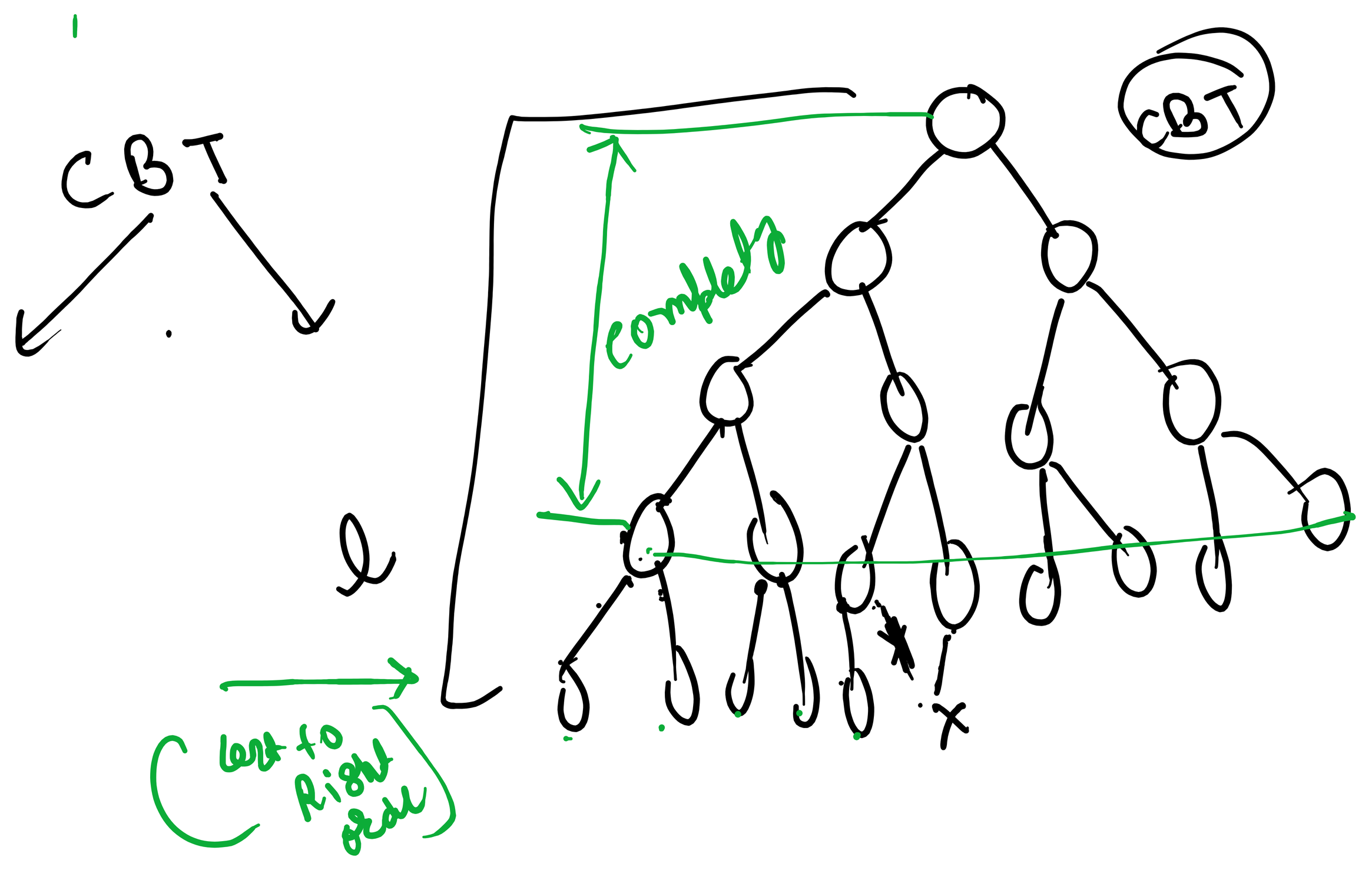
Heap → Priority Queue

Heap → It is a CBT → Binary Tree and added some priority

[CBT → complete Binary tree]

Flipkart, Walmart

CBT



① Heap → minheap, maxheap

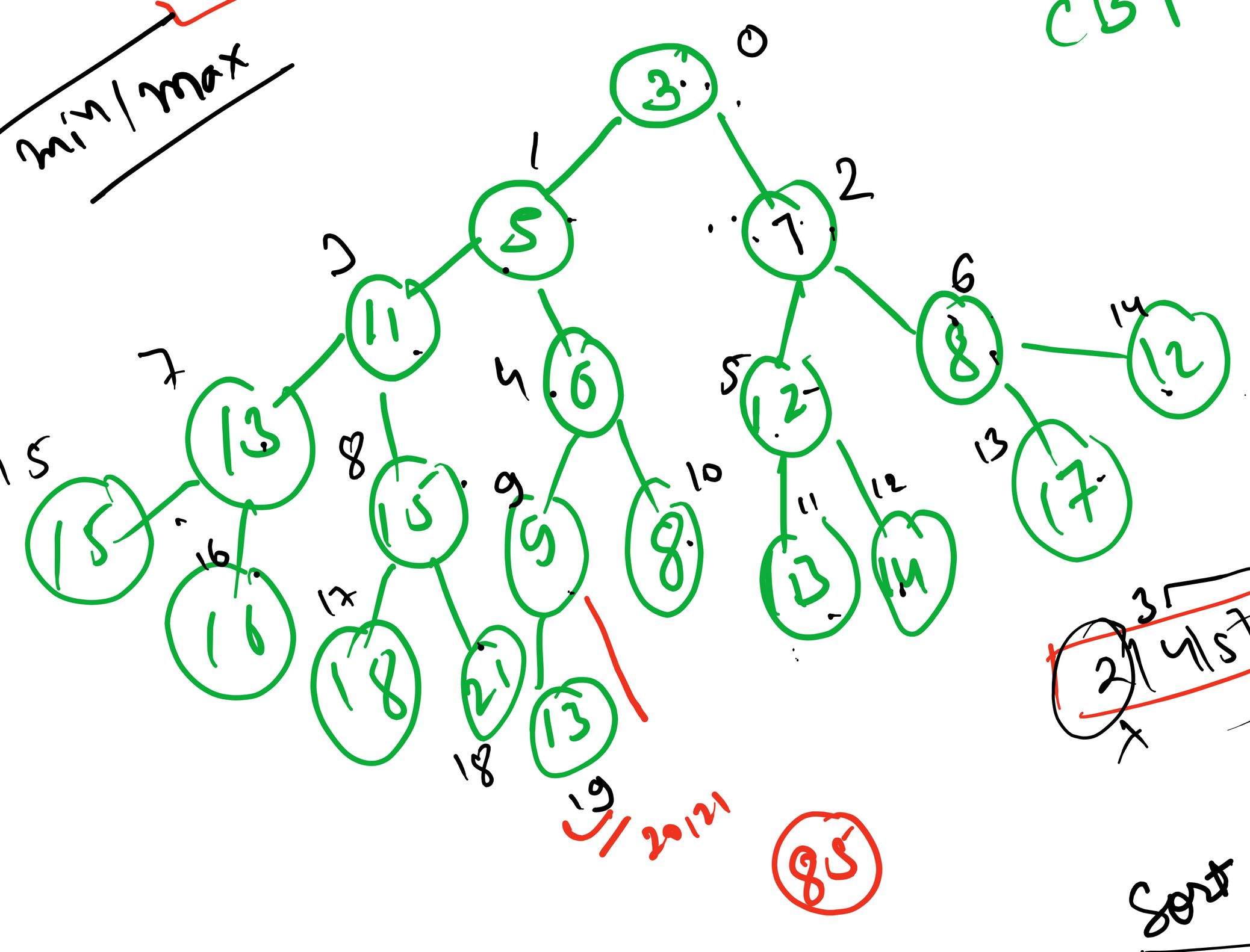
min / max

priority

[parent < [Left Child, Right Child]] Smaller ↑

CBT

min / max → min condition, CBT



$LCI = 2 \times 5 + 1 = 2 \times pi + 1$ ✓

$RCI = 2 \times 5 + 1 = 2 \times pi + 2$ ✓

| 2 | 4 | 5 | 6 | O(n)

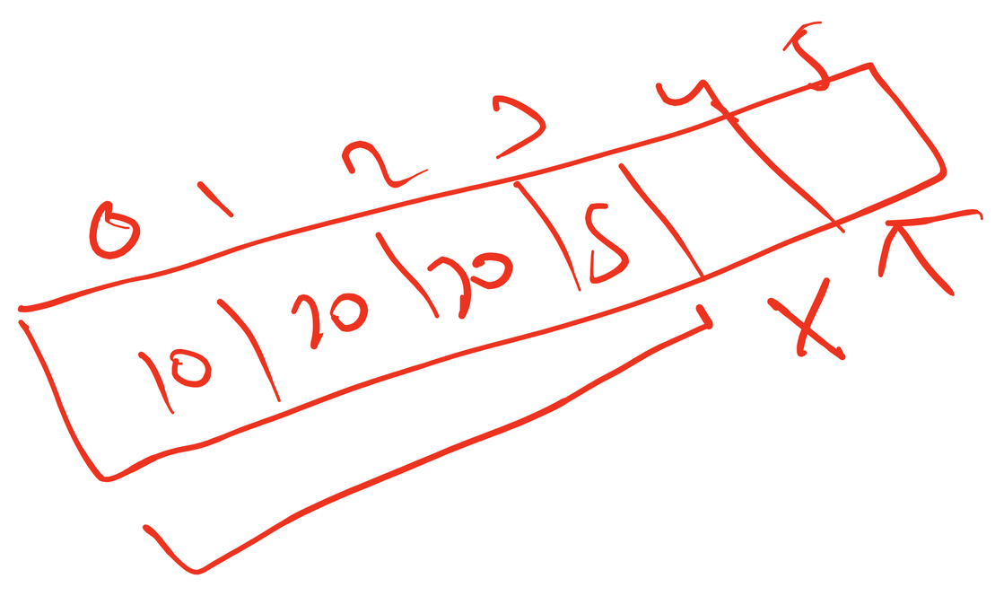
$$pi = \frac{ci - 1}{2}$$

$= \frac{11-1}{2} = 5$

$\frac{12-2}{2} = 5$

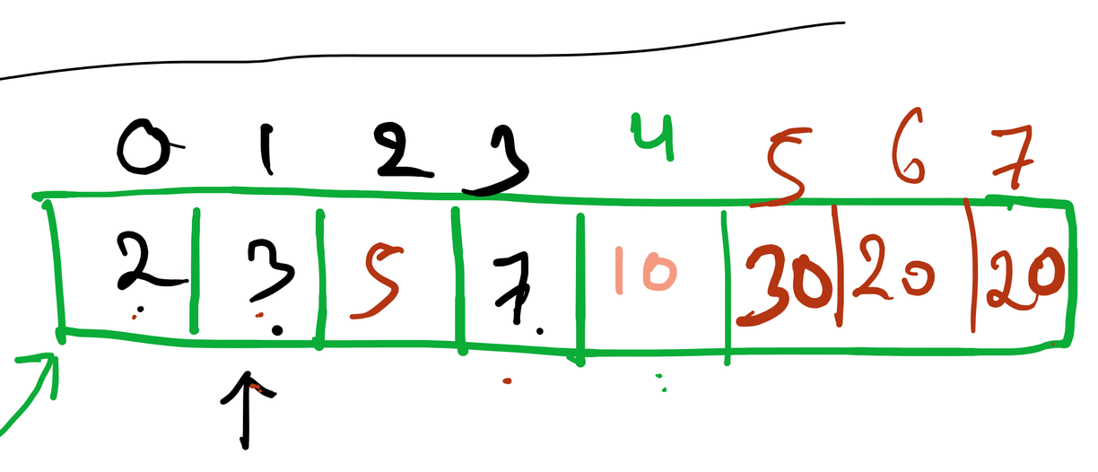
CBT ?

[10, 20, 30, 5, 7, 3, 20, 2]



	Sort	unsorted	heap
Add	O(n)	O(1)	log(n)
remove	O(n)	O(n)	log(n)
get	O(1)	O(n)	O(1)

[10, 20, 30, 5, 7, 3, 20, 2]

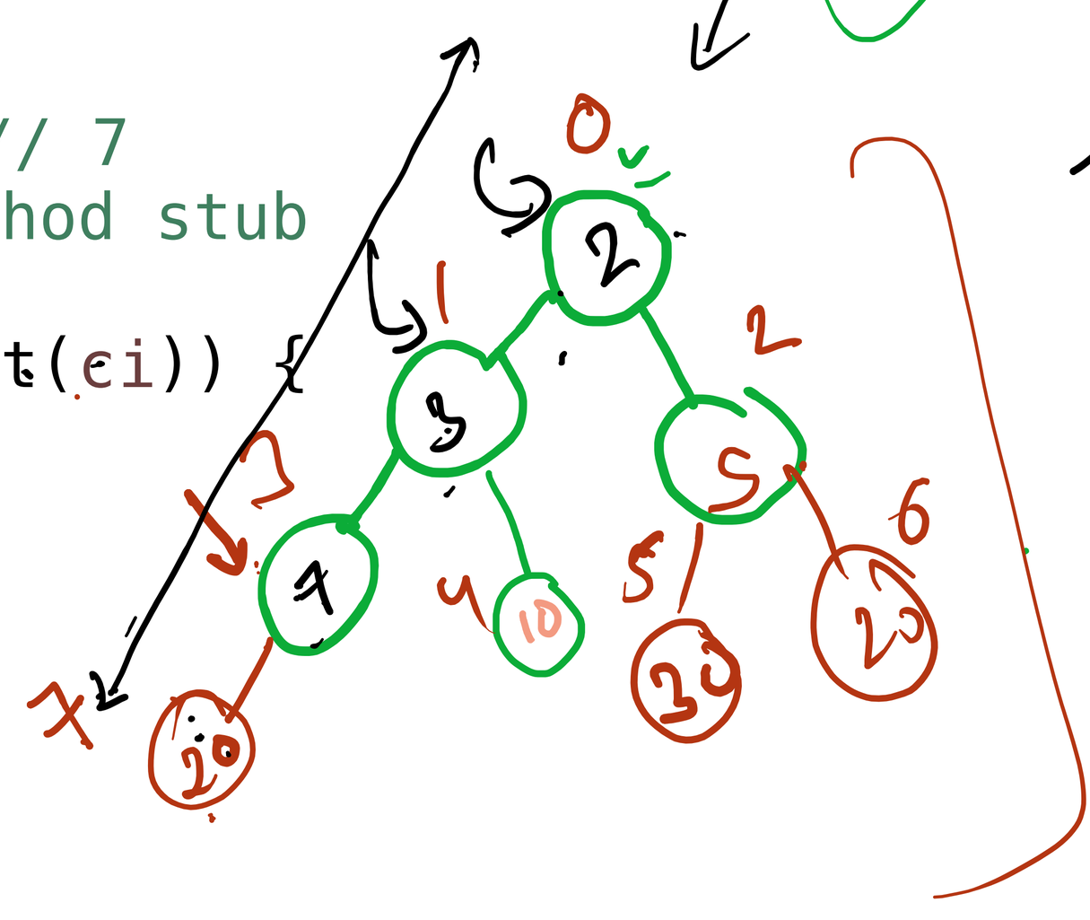


parent = 20
child = 2

```
2
[2, 3, 5, 7, 10, 30, 20, 20]
```

① upheapify

```java
private void upheapify(int ci) {// 7
	// TODO Auto-generated method stub
	int pi = (ci - 1) / 2;
	if (list.get(pi) > list.get(ci)) {
		swap(pi, ci);
		upheapify(pi);
	}
}
```



$\frac{0-1}{2} = 0$

$\frac{7-1}{2} = 3$

$\frac{3-1}{2} = 1$

$\frac{2-1}{2} = 0$

$\frac{1-1}{2} = 0$

$\frac{0-1}{2} = \frac{-1}{2} = 0$

$\frac{4-1}{2} = 1$

$\frac{5-1}{2} = 2$

$2^h = n$

$h = \log(n)$

log(n)

O(n)

[10, 20, 30, 5, 7, 3, 20, 2]

$LCI = 2 \times 0 + 1 = 1$

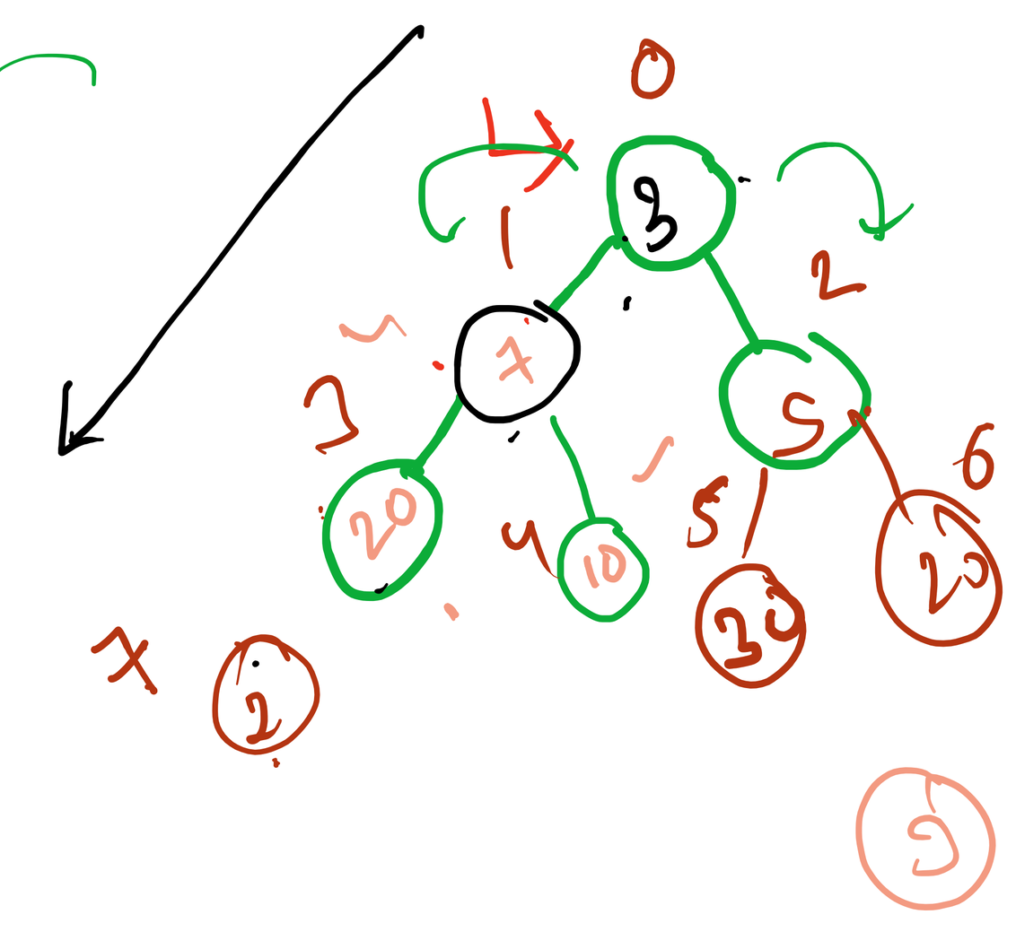
$RCI = 2 \times 0 + 2 = 2$

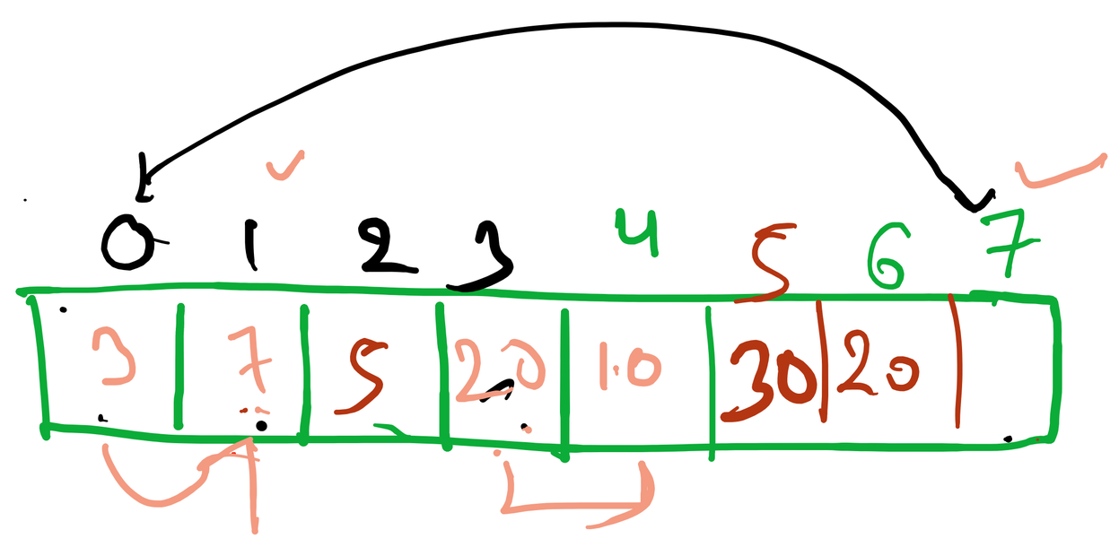
0 1, 2

pi, lci, rci

min

mini = 0





O(1)

O(1)

```
2
[2, 3, 5, 7, 10, 30, 20, 20]
2
[3, 7, 5, 20, 10, 30, 20]
```

```java
private void downheapify(int pi) {
	// TODO Auto-generated method stub
	int lci = 2 * pi + 1;
	int rci = 2 * pi + 2;
	int mini = pi;
	if (list.get(lci) < list.get(mini)) {
		mini = lci;
	}
	if (list.get(rci) < list.get(mini)) {
		mini = rci;
	}
	if (mini != pi) {
		swap(mini, pi);
		downheapify(mini);
	}

}
```

$2 \times 3 + 1 = 7$

$2 \times 3 + 2 = 8$