Connecting the ropes

2   5   2   6   3

You can connect two ropes together at a time.
Cost of connecting two ropes → sum of length of ropes.
Find the minimum cost of connecting all the ropes.

			Cost
2 + 5	→	7	7 +
7 + 2	→	9	9 +
9 + 6	→	15	15 +
15 + 3	→	18	18
			49

$\underbrace{2, 2}_{}, \underbrace{3, 5, 6}_{}$

7   11

			Cost
2 + 2	→	4	4 +
4 + 3	→	7	7 +
7 + 5	→	12	12 +
12 + 6	→	18	18
			41

			Cost
2 + 2	→	4	4 +
3 + 4	→	7	7 +
5 + 6	→	11	11 +
7 + 11	→	18	18
			40

idea → Always pick the 2 smallest ropes & combine them.

2, 2, 3, 5, 6

4, 3, 5, 6 $\xRightarrow{\text{sort}}$ 3, 4, 5, 6

7, 5, 6 $\xRightarrow{\text{sort}}$ 5, 6, 7

11, 7 $\xRightarrow{\text{sort}}$ 7, 11

18

Cost

4
+
7
+
11
+
18

40

D.S
- → insert( ) → $\log_2 N$
- → extractMin / removeMin( ) ↳ $\log_2 N$

⇓

Heap

Heap Data Structure

⇓

Binary Tree

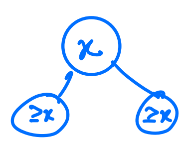
- ↓ **Structure**

  Complete Binary Tree (C.B.T)

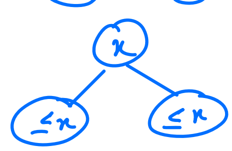
  All levels must be completely filled & last level must be filled from left to right.

- ↓ **Order of elements**

  Heap Order Property (HOP)

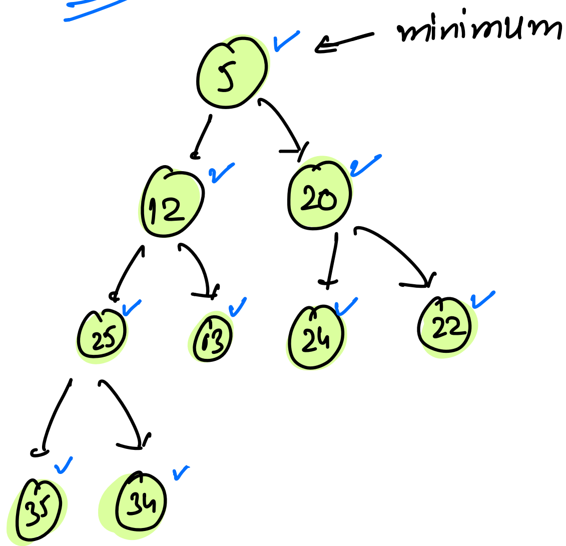
  

  min-Heap

  max-Heap

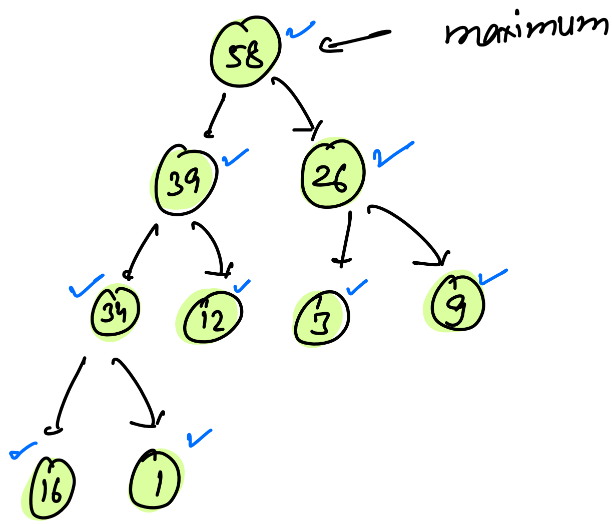
min-Heap



C.B.T ✓

H.O.P ✓

max-Heap.



C.B.T ✓

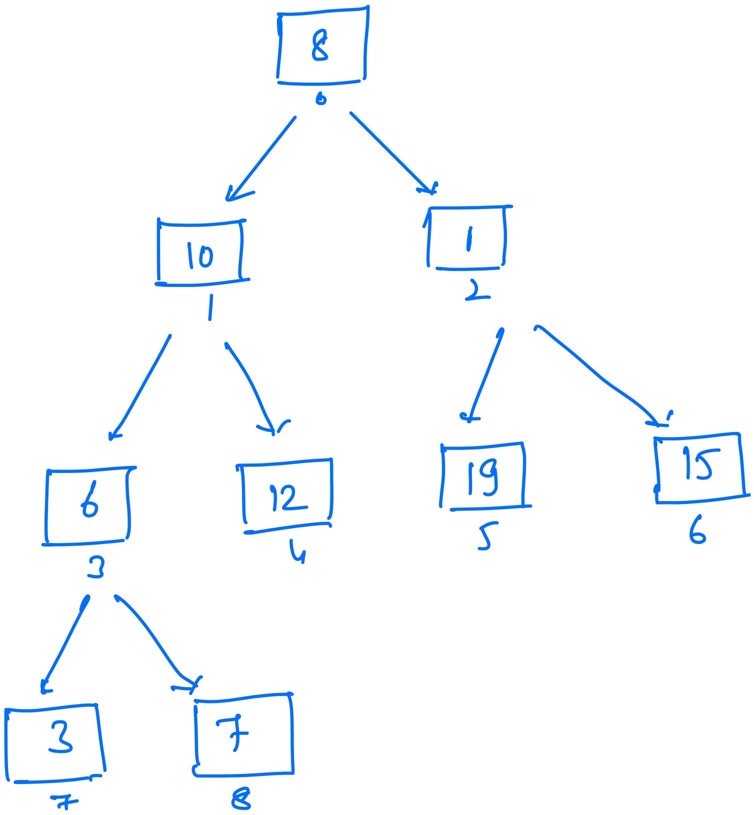
H.O.P

minHeap → getMin() → O(1)

maxHeap → getMax() → O(1)

Visualise Arrays as Binary Tree [C.B.T]

8	10	1	6	12	19	15	3	7
0	1	2	3	4	5	6	7	8



$0 \to 1 \quad (2*0+1)$

$0 \to 2 \quad (2*0+2)$

$1 \to 3 \quad (2*1+1)$

$1 \to 4 \quad (2*1+2)$

$2 \to 5 \quad (2*2+1)$

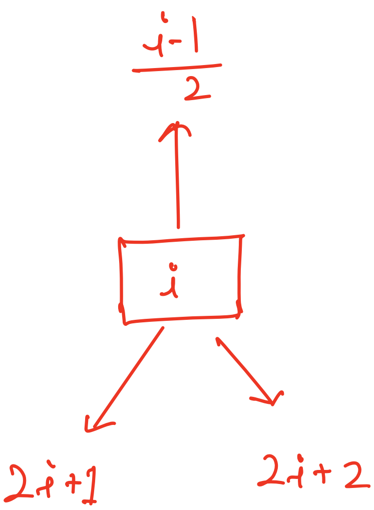
$2 \to 6 \quad (2*2+2)$

$3 \to 7 \quad (2*3+1)$

$3 \to 8 \quad (2*3+2)$

$4 \to 9 \quad (2*4+1)$

$4 \to 10 \quad (2*4+2)$

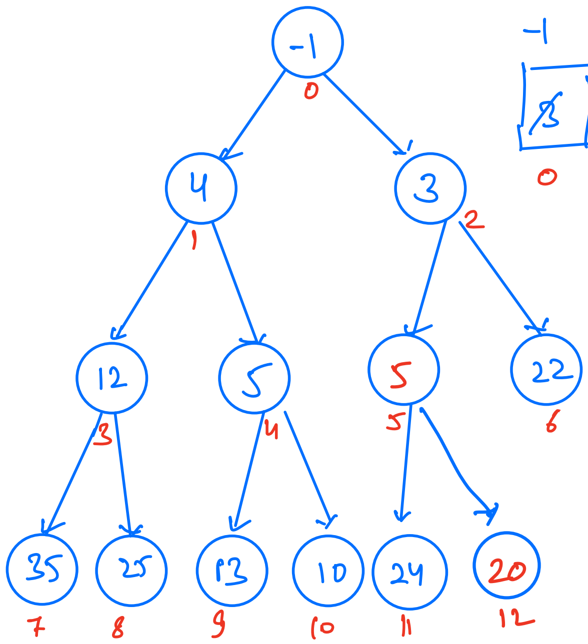


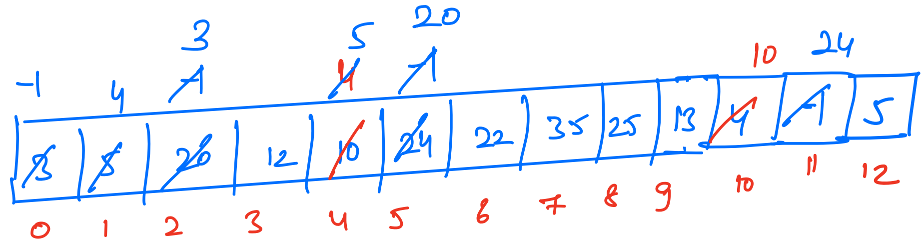
$i \xrightarrow{\text{left}} 2i+1$

$i \xrightarrow{\text{right}} 2i+2$

$ci \longrightarrow \frac{ci-1}{2}$

min-Heap. (insertion)





insert (-1)

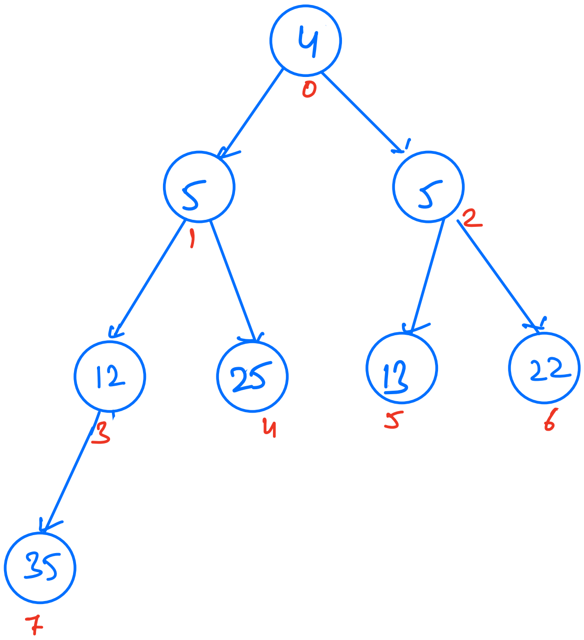
i	pi	arr[pi] < arr[i]
11	$\frac{11-1}{2} = \frac{10}{2} = 5$	No ⇒ swap
5	$\frac{5-1}{2} = \frac{4}{2} = 2$	No ⇒ swap
2	$\frac{2-1}{2} = \frac{1}{2} = 0$	No ⇒ swap

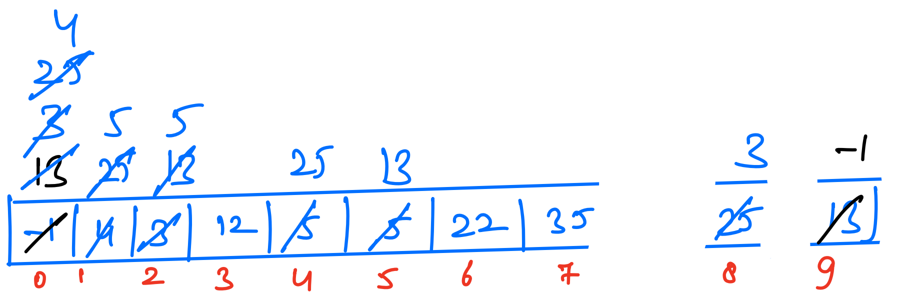
T.C → $O(Ht.)$ → $O(\log_2 N)$

code. →

```
void insert( arr[], x) {
    arr.addLast(x);
    i = arr.size() - 1;
    while( i > 0 ){
        pi = (i-1)/2;
        if( arr[pi] > arr[i]){
            swap( arr[i] with arr[pi])
            i = pi;
        }
        else{
            break;
        }
    }
}
```

Extract-Min / Remove-Min





Code :-

```
swap( arr[0] with arr[size-1])

arr.removeLast();

heapify( arr[], 0);

void heapify( arr[], i ){
    while( 2i+1 < size ){
        x = Min( arr[i], arr[2i+1], arr[2i+2] );
        if( arr[i] == x){
            break;
        }
        else if ( arr[2i+1] == x){
            swap(arr[i] with arr[2i+1]);
            i = 2i+1;
        }
        else{
            swap (arr[i] with arr[2i+1]);
            i = 2i+2;
        }
    }
}
```

T.C → $O(\log_2 N)$

S.C → $O(1)$

2 5 2 6 3

You can connect two ropes together at a time.
Cost of connecting two ropes → sum of length of ropes.
Find the minimum cost of connecting all the ropes.

Cost = 0+4+7+11+18
= 40.

min. Heap

2 + 2 = 4
3 + 4 = 7
5 + 6 = 11
7 + 11 = 18

Code:

```
Heap<long> heap;
for(i=0; i<N; i++){
        heap.insert(arr[i]);
}

long cost = 0
while( heap.size() > 1){
    long x = heap.removeMin();
    long y = heap.removeMin();

        cost += (x+y);
        heap.insert(x+y);
}

return cost;
```

$\boxed{\begin{array}{ll} T.C \rightarrow & O(N \log N) \\ S.C \rightarrow & O(N) \end{array}}$

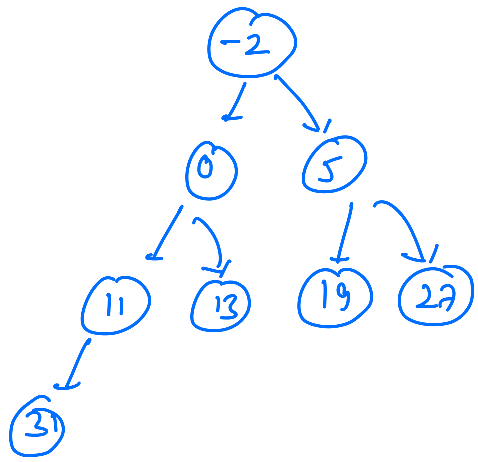
Build a Heap [Interview Problem]

[5, 13, -2, 11, 27, 31, 0, 19]

① Sort the array

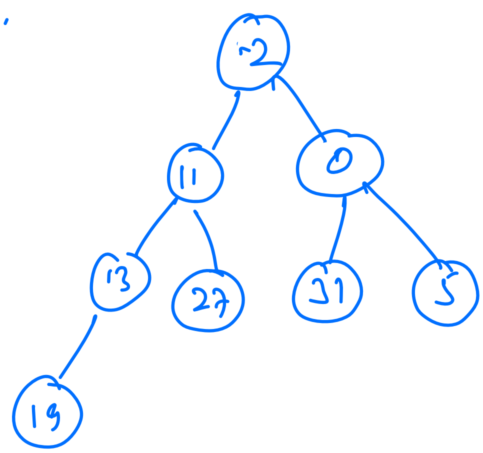
[-2, 0, 5, 11, 13, 19, 27, 31]

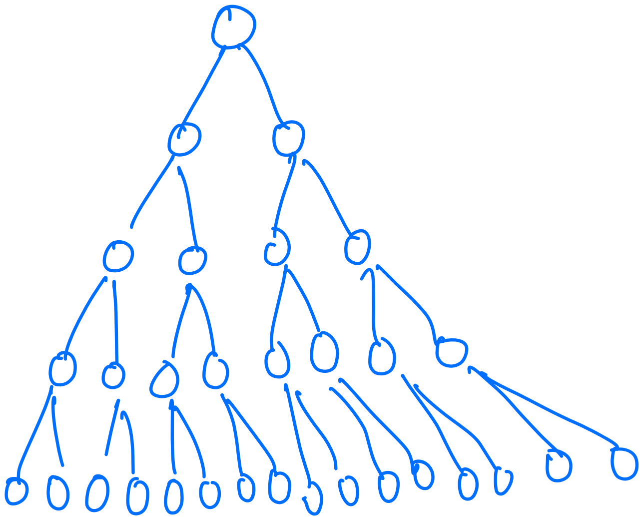
T.C → $O(N \log N)$



② call insert(arr[i]) for all the elements.

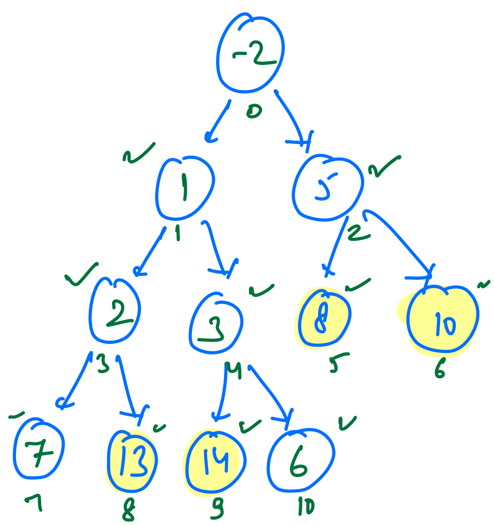
T.C → $O(N \log N)$





All leaf nodes → H.O.P is already valid

arr[] → [7, 3, 5, 1, 6, 8, 10, 2, 13, 14, -2]

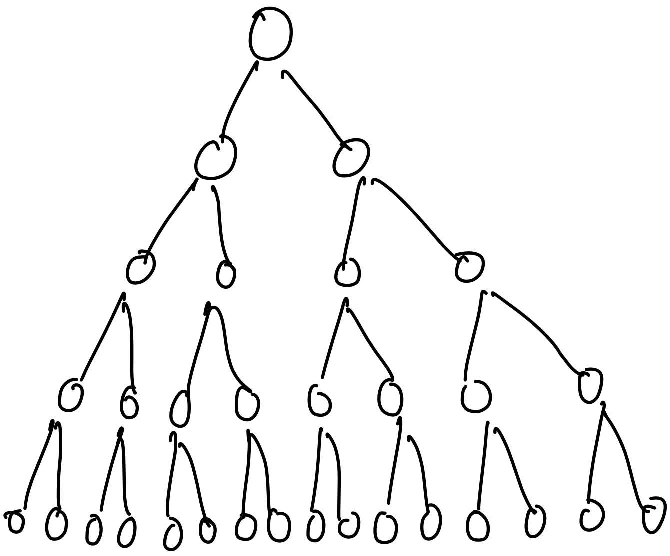


Idx of last non-leaf node ?

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

```
for( i = N/2 - 1 ; i >= 0 ; i-- ){
    heapify (arr[], i);
}
```

total elements = N



		swaps.
1	*	
|		
$N/16$	*	3
$N/8$	*	2
$N/4$	*	1
$N/2$	*	0

$$\text{total swaps} = \cancelto{0}{\frac{N}{2} * 0} + \frac{N}{4} * 1 + \frac{N}{8} * 2 + \frac{N}{16} * 3 + \frac{N}{32} * 4 + \text{---}$$

$$= \frac{N}{2}\left[\underbrace{\frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \frac{4}{16} + \frac{5}{32} + \text{---}}_{A.G.P}\right]$$

$$S = \frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \frac{4}{16} + \frac{5}{32} + \text{---} \rightarrow (1)$$

$$\frac{S}{2} = \frac{1}{4} + \frac{2}{8} + \frac{3}{16} + \frac{4}{32} + \frac{5}{64} + \text{---} \rightarrow (2)$$

$$-$$

$$\frac{S}{2} = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} + \text{---}$$

$$\frac{S}{2} = \frac{\frac{1}{2}}{1 - \frac{1}{2}} = \frac{\cancel{1/2}}{\cancel{1/2}} = 1 \qquad S_\infty = \frac{a}{1-r}$$

$$\boxed{S = 2}$$

total swaps = N $\Rightarrow$ [T.C $\Rightarrow$ O(N)]

→ sort again & again

→ min/max again & again

] ⇒ Heap/ Priority Queue

Next Session → → Heap Sort

→ Median of Stream of Integers