Ques: Given an array that represents the size of different ropes. In a single operation we can connect two ropes. cost of connecting two ropes is sum of the length of ropes.

Find the minimum cost of connecting all the ropes

ex: $A = [2, 5, 3, 2, 6]$

Try 1:

$2 + 5 = 7$ → $[7, 3, 2, 6]$

$7 + 3 = 10$ → $[10, 2, 6]$

$10 + 2 = 12$ → $[12, 6]$

$12 + 6 = 18$ → $[18]$

47

Try 2:

$2 + 3 = 5$ → $[5, 5, 2, 6]$

$5 + 2 = 7$ → $[7, 5, 6]$

$5 + 6 = 11$ → $[7, 11]$

$7 + 11 = 18$ [18]

41

* If I pick smaller element first then cost will be minimum.

$x < y < z$

	Case 1	Case 2	Case 3
Step1	$x + y$	$x + z$	$y + z$
Step2:	$(x+y) + z$	$(x+z) + y$	$(y+z) + x$
	$2x + 2y + z$	$2x + 2z + y$	$2y + 2z + x$

Approach 1 : Sort and add.

2 2 3 5 6

(1) Sort

(2) add first two element

(3) insertion sort.

T.C: ~~n log n~~ + N^2

n x (n) log n

T.C = $O(N^2)$

S.C : = $O(1)$

Quiz [1, 2, 3, 4]

1 + 2 = 3
3 + 3 = 6
6 + 4 = 10
19

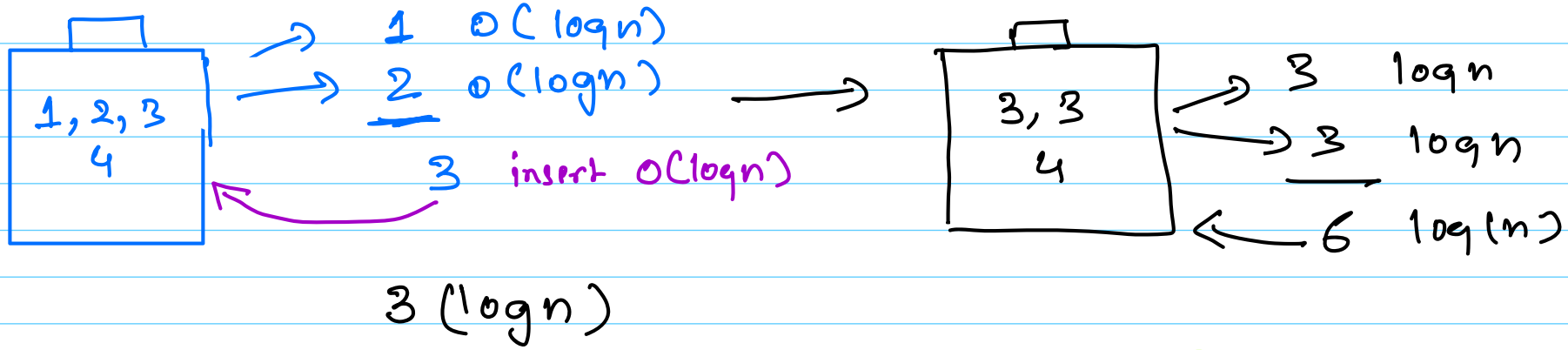
Optimize 2

Heap → (Datastructure)

1) Insertion of element take O(log n)

Min heap.

2) Extraction of min / max element take O(log n)



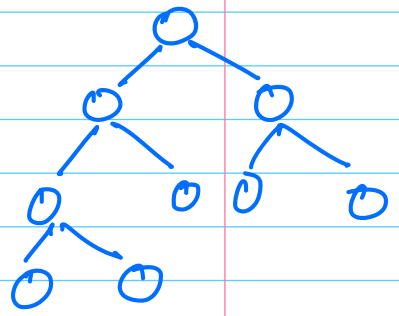
3 (log n)

T.C = n * 3(log n) ⇒ [n log n]

S.C : O(n)

Heap Data Structure. → Min heap (Extraction gives smallest element)
→ Max heap (Extraction give largest element)

↳ This is a binary tree with two special property.



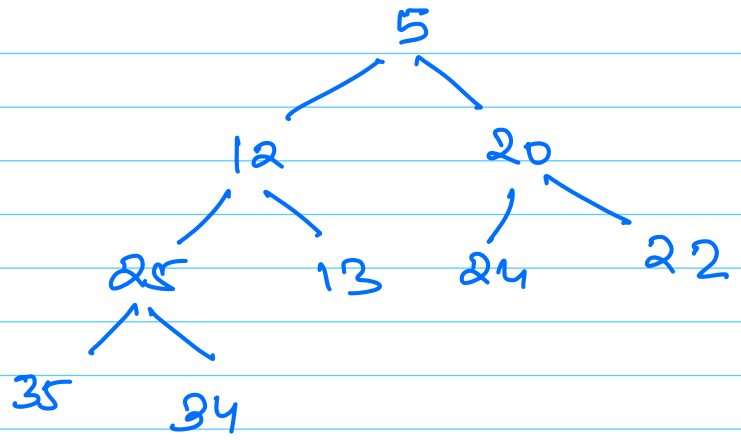
(1) Complete Binary tree: All the levels are completly filled. The last level can be an exception but it should be filled from left to right

(2). Heap order priority:

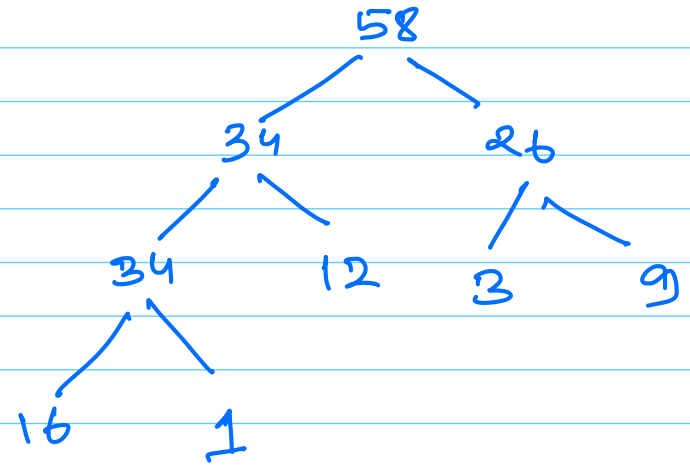
↳ Min heap : Every node should be smaller than all the descendants.

↳ Max heap: Every node should be greater than all its descendants.

Minheap.

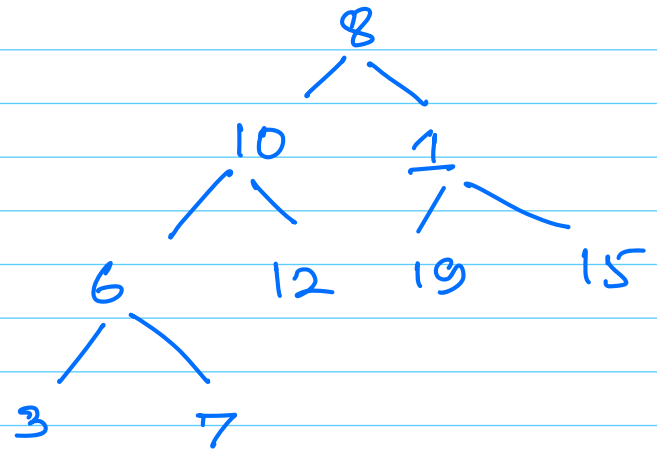


Max heap:



Implementation of heap

* we can implement heap using array. simply because it is a complete binary tree.

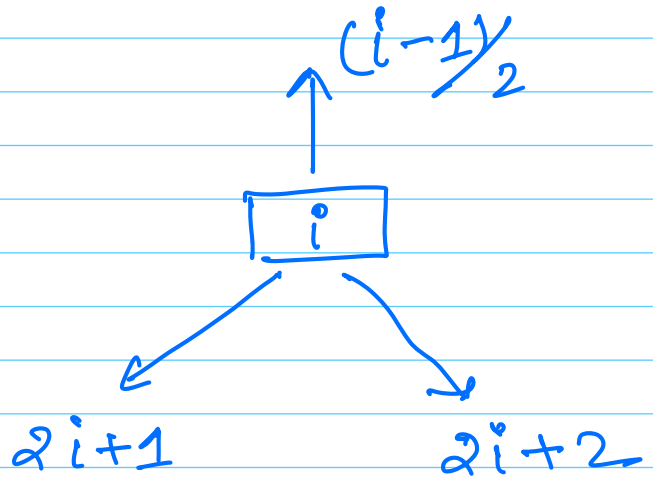


8	10	1	6	12	19	15	3	7
0	1	2	3	4	5	6	7	8

parent i

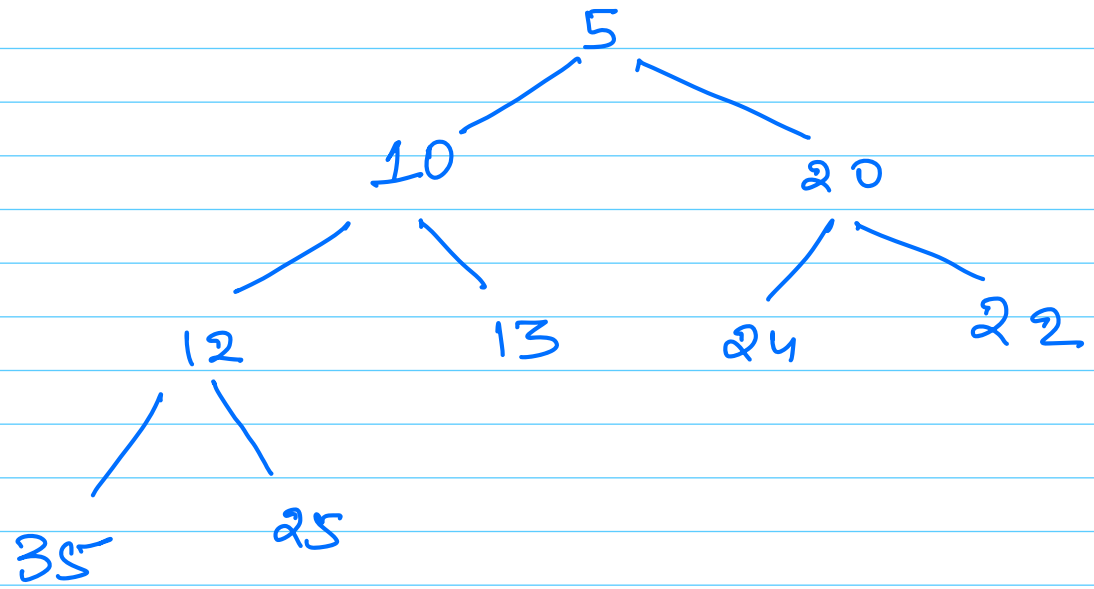
child ⇒ left = 2i+1
right = 2i+2

child : i parent : $(i-1)/2$



Insert a number in Minheap.

insert (10)



T.C: O(log n)

0	1	2	3	4	5	6	7	8
5	10	20	12	13	24	22	35	25

1. Complete Binary Tree : add the element at last of arrays

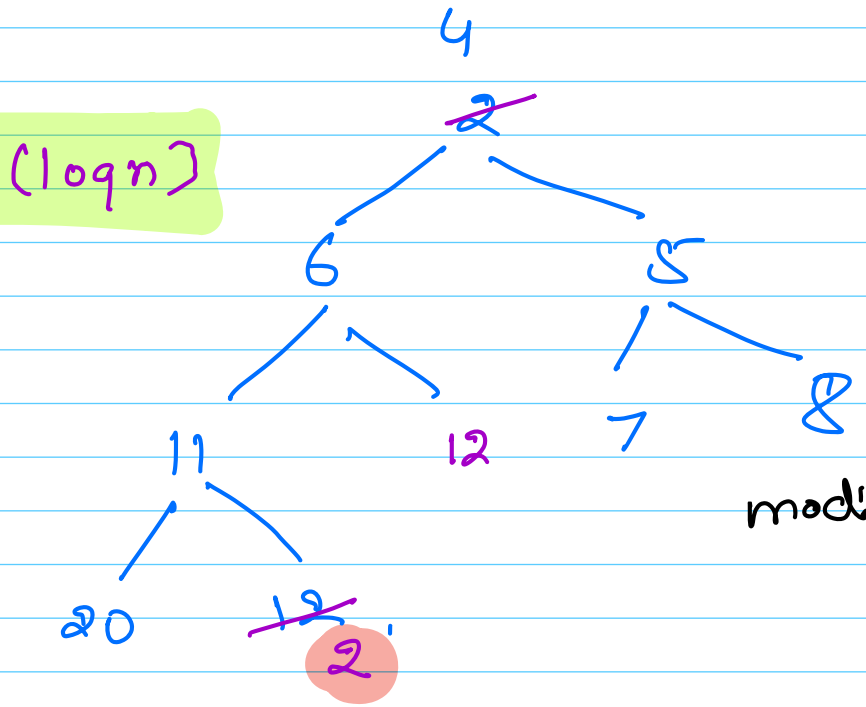
2. if (arr[pi] > arr[i]) swap.

Pseudo code

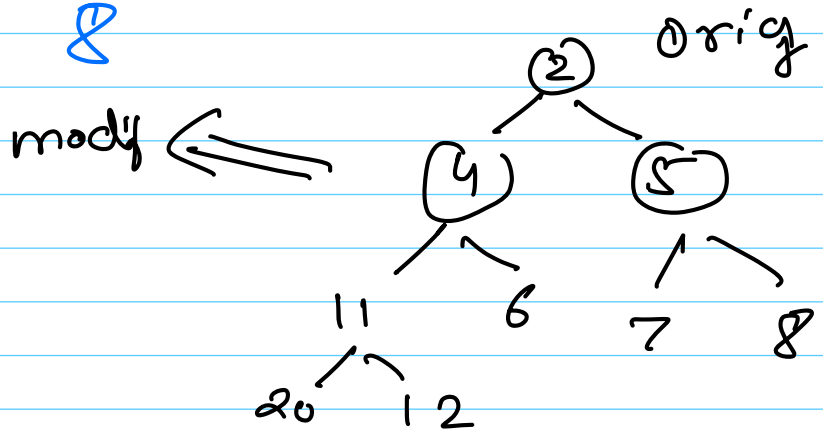
```
heap[]
insert (heap, k)
{
        heap.add(k)
        i = heap.size - 1.
        while (i>0){
            pi = (i-1)//2
            if (heap[pi] > heap[i]){
                swap(heap, pi, i);
                i = pi
            }
            else
            {
                break;
            }
    }.
```

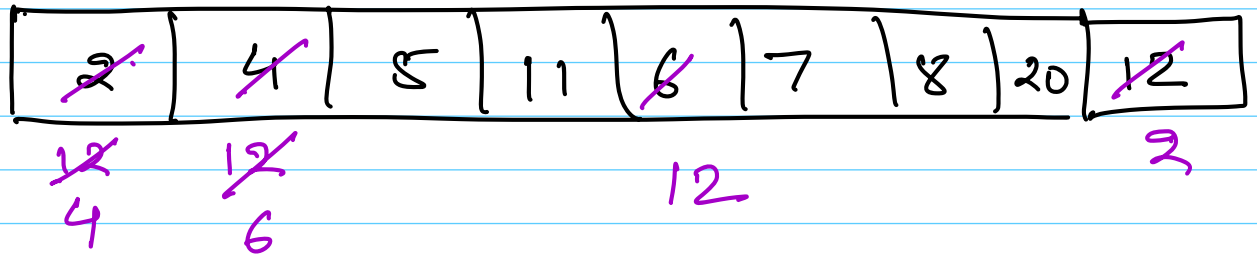
2) Extract Min Element

T.C: O(logn)



* Swap from smaller child.





Implementation

```
heap[]
swap ( heap, 0, heap.length -1)
heapify ( heap[], i )
{

    while ( 2i+1 < N )
    {       // add a check for 2i+2
      x =   min ( heap[i], heap[2i+1],
                           heap [2i+2] )

      if ( x == heap[i] )
           break

      else if ( x == heap[2i+1] )
      {      swap ( heap, i, 2i+1)
             i = 2i+1

      }
```

else
{

swap (heap, i, 2i+2)
i = 2i+2

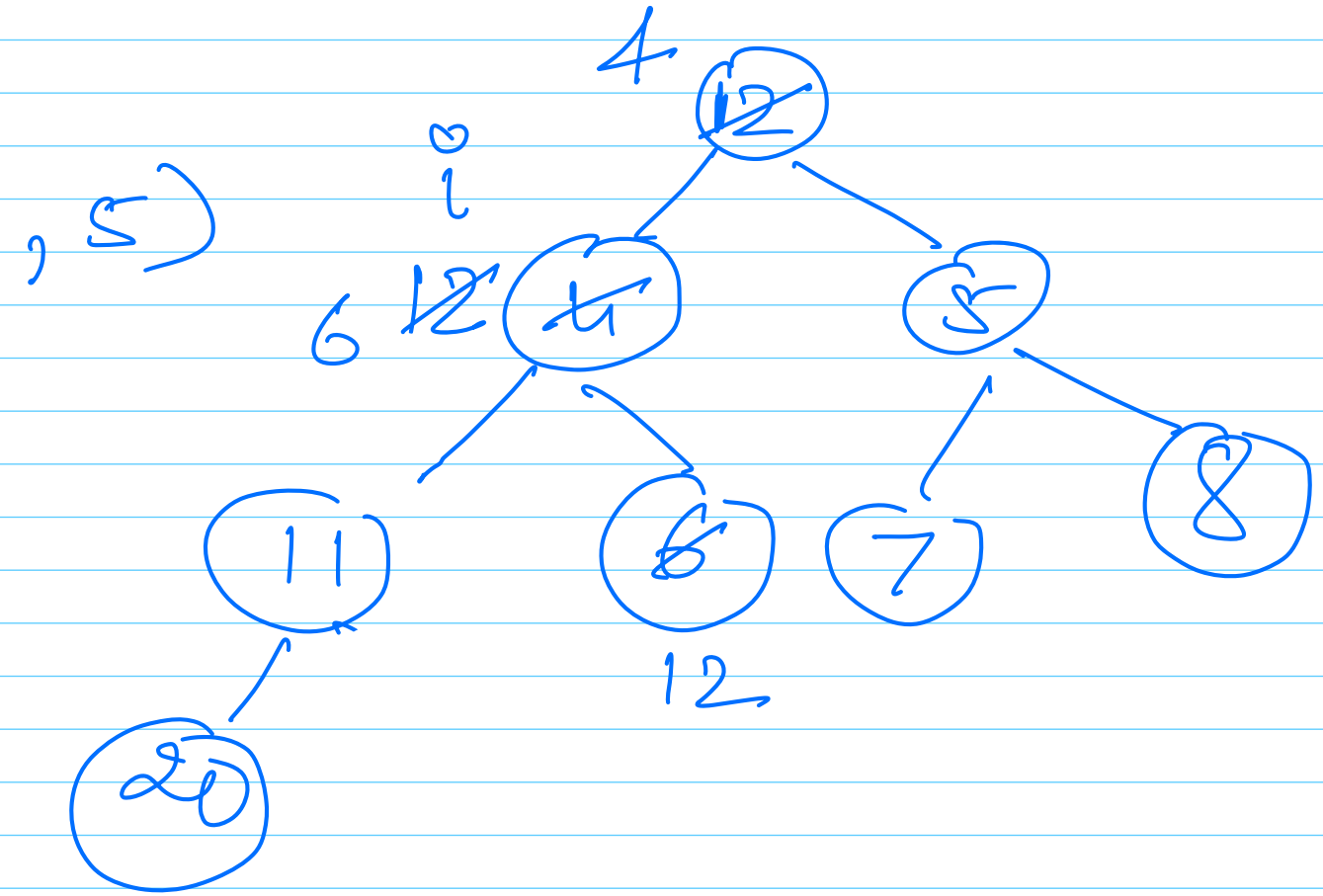
}

}

}.

min(12, 4, 5)
↳ 4

min(12, 11, 6)
↳ 6



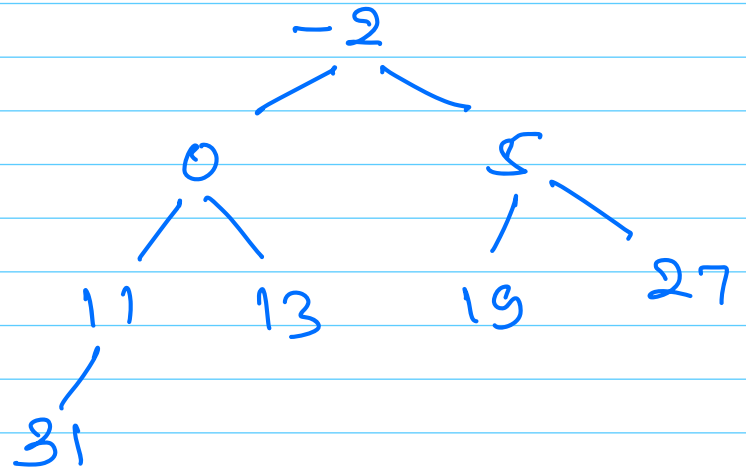
Build a heap

(Min heap)

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

Idea 1 => 1) Sort the array

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



T.C: nlogn

Idea 2 : insert (heap, k)

↓

log (n)

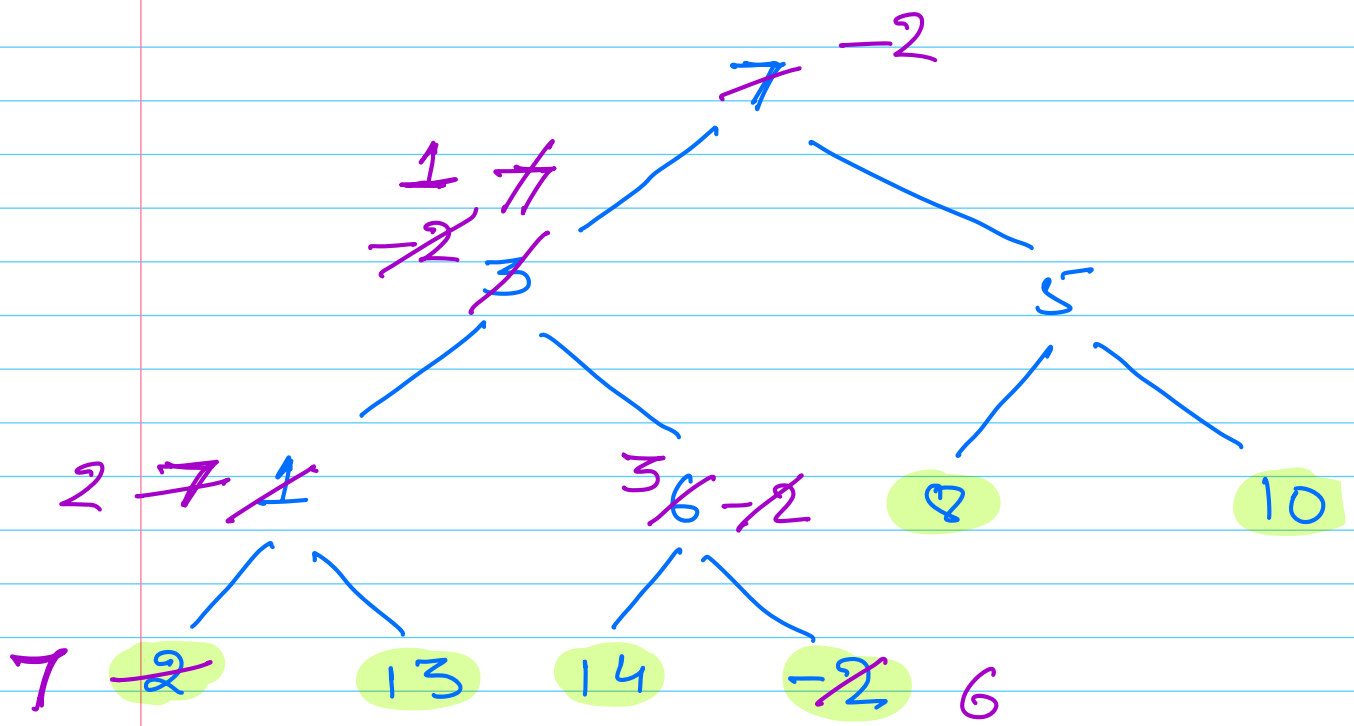
↓

all element

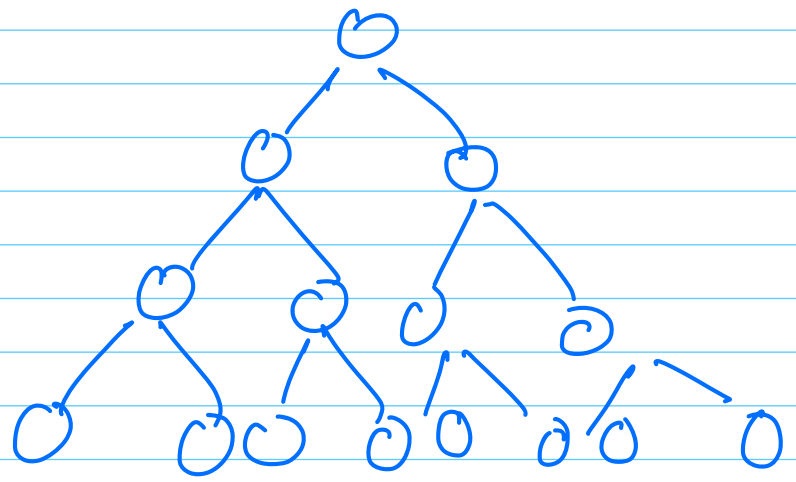
T.C : nlogn

Idea 3

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



Total elements = N



	Swaps
1	3
N/8	2
N/4	1
N/2	0

$$S = N/2 * 0 + N/4 * 1 + N/8 * 2 \cdots\cdots$$

$$S = \frac{N}{2}\left[\frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \frac{4}{16} + \cdots\right]$$

↓

Arithmatic geometric progress

$$S = \frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \frac{4}{16} + \cdots$$

$$\frac{S}{2} = \frac{1}{4} + \frac{2}{8} + \frac{3}{16} + \frac{4}{32} + \cdots$$

$$S - \frac{S}{2} = \underbrace{\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \cdots}_{\text{GP.}}$$

Sum of infinite $= \frac{a}{1-r}$

$= a = 1/2$

$r = 1/2$

$$= \cdot \frac{1/2}{1 - 1/2}$$

$$S - S/2 = 1$$

$$\cdot \frac{S}{2} = 1$$

$$S = \underline{\underline{2}}$$

Overall iteration $= 2 \times N/2$

$= O(N)$

Total time complexity $= O(N)$

$(n/2)-1$

```
for (int i = (n/2 - 1); i >= 0; i--
{
    heapify (heap[], i);
}
```

$\Rightarrow$ $n \log n$

Que

Merge N Sorted array :

a = [2, 3, 11, 15, 20]

b = [1, 5, 7, 9]

c = [0, 2, 4]

d = [3, 4, 5, 6, 7, 8]

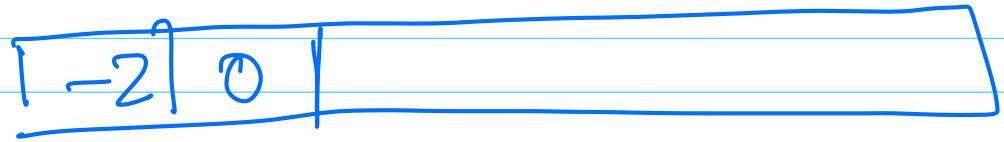
e = [-2, 5, 10, 20]

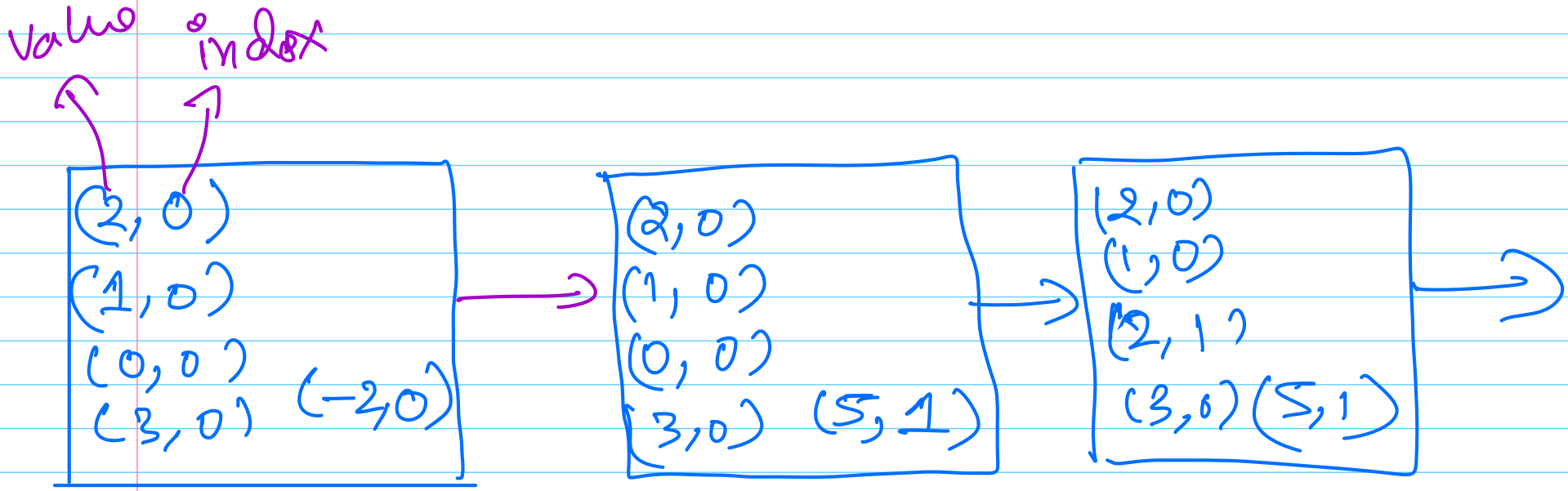
] K array

Idea 1: take two array and merge them and do for all.

$= O(n^2)$

Idea 2 minheap.





1) add first element of all the array with their index.

2) Extract an element
   - add in the merged array
   - pick next element of same array and insert.

T.C: $K(\log n + \log n)$

$= 2K \log(n)$

$= K \log(n)$

H.W