Batch PSP : 55.66 → 58

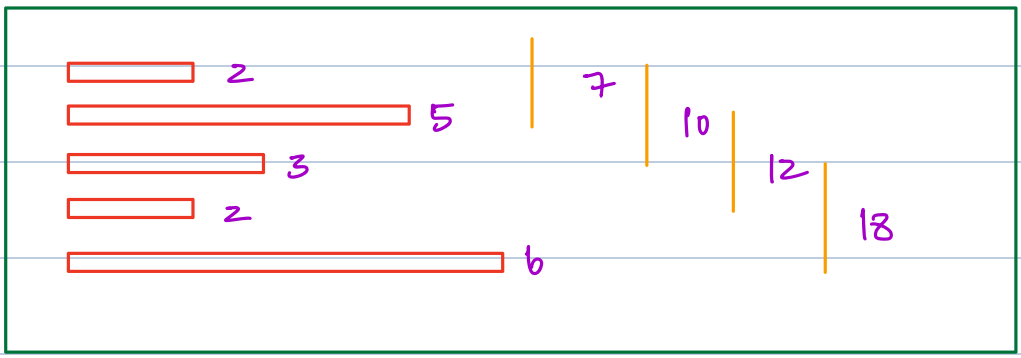
Nov23_PSP_12Apr
sudhakar venkatachalam
Vijay V A
Piyush Kumar
Mateen
Mayur Hadawale
Sai Sharath
Manjunatha I
Harshil Dabhoya
Rajeev
Gobika K
manikandan m
Yash Malviya
Kevin Theodore E
Robin Dhiman

Nov23_PSP_12Apr
Suraj Devraye
Vigneshwaran K
Sarat Patel
MD JASHIMUDDIN
Nitendra Rajput
kameswarreddy Yeddula
Shaurya Srivastava
ALLEN GEOSHAN M
Pranadarth S
Pushkar Deshpande
Pradeep Kumar Chandra
Prashant Kumar Soni
SIJU SAMSON
Mohammed Arshad
Rsr Ram

Q → given N ropes of different sizes. In one operation we can connect 2 ropes & the cost is sum of length of both ropes.

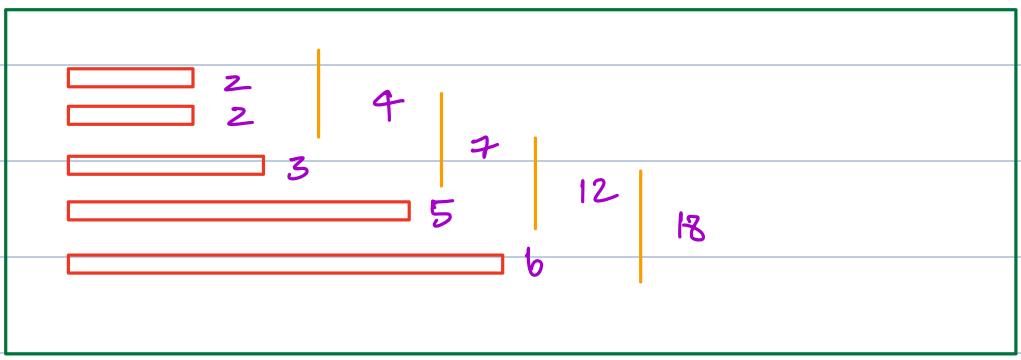
Find min cost to connect all ropes

example



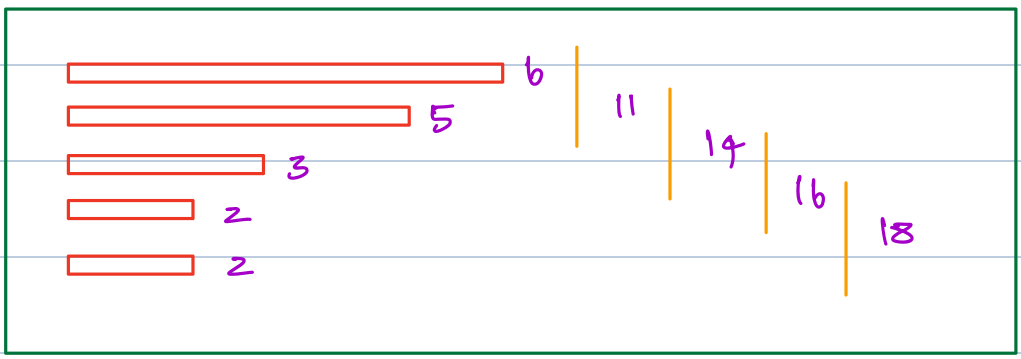
cost = 7 + 10 + 12 + 18

= 47



cost = 4 + 7 + 12 + 18

= 41



cost = 11 + 14 + 16 + 18

= 59

Have we gotten the best answer?

No

let us say we have 3 ropes of length → $x < y < z$

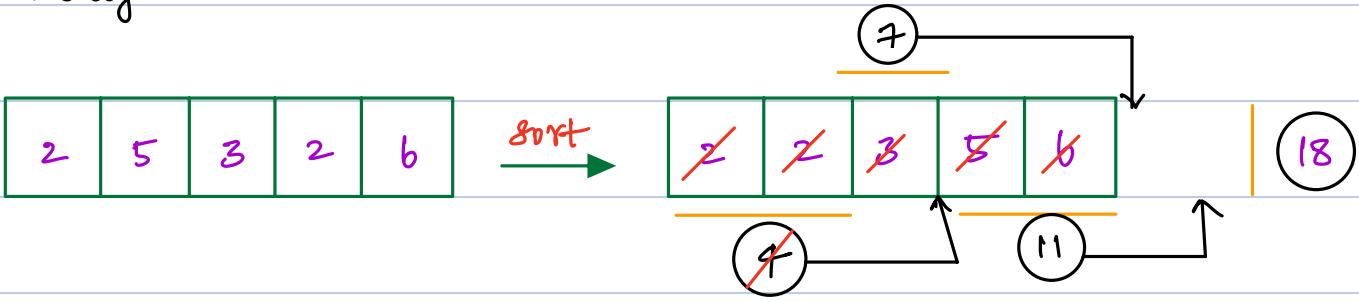
Case 1	Case 2	Case 3
connect x, y, z	connect x, z, y	connect y, z, x
$x+y$	$x+z$	$y+z$
$x+y+z$	$x+y+z$	$y+z+x$

Observation

connect small length ropes at each stage to get min cost

Solution 1

Sort the input length of ropes and connect sequentially



total cost = $4+7+11+18$

$= 40$

→ after calculating sum, insert it to correct spot using insertion sort

T.C = $O(n^2)$ S.C = $O(1)$

Quiz 1

Minimum cost of connecting all ropes

1	2	3	4

Step 1: 1, 2, 3, 4 → cost = 3

Step 2: 3, 3, 4 → cost = 6

Step 3: 6, 4 → cost = 10

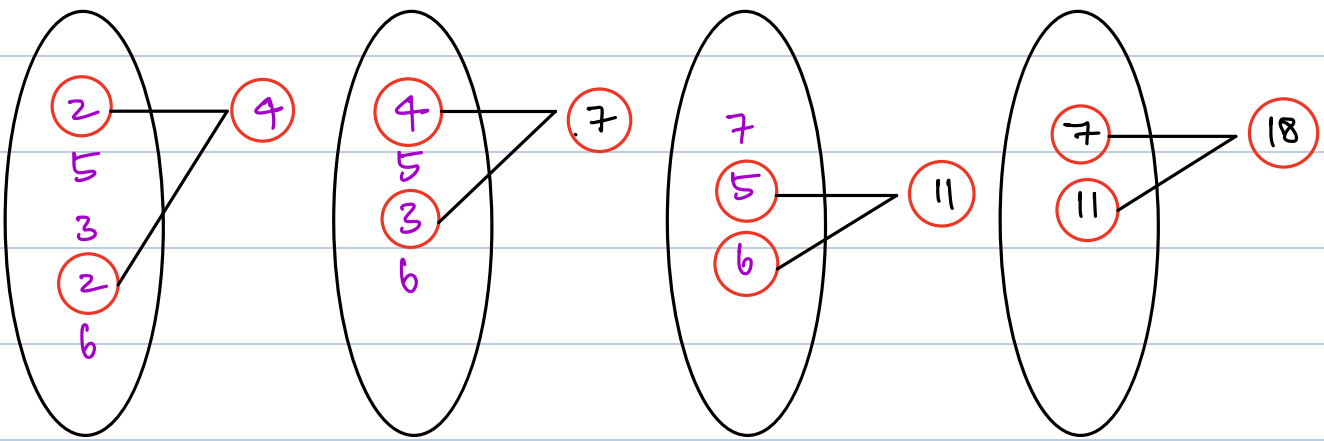
total = 19

Connecting the ropes Optimization

We will use Heaps or priority Queue to solve this problem effectively

Insertion of element → O(log n)

get min / max element → O(log n)



per step → $3 * \log n = O(\log n)$

overall T.C → $O(n \log n)$

Heaps / Priority Queue

Structure → Complete Binary Tree

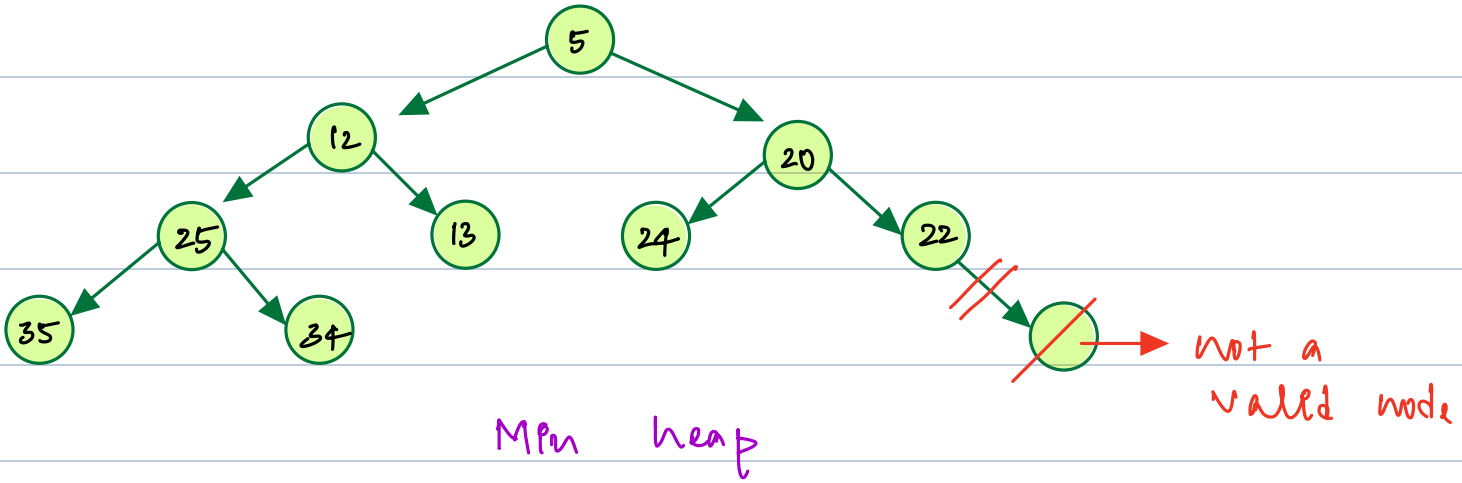
All levels all completely filled. The last level can be an exception but should be filled from left to right

Type → Min Heap ∀ nodes data <= children nodes

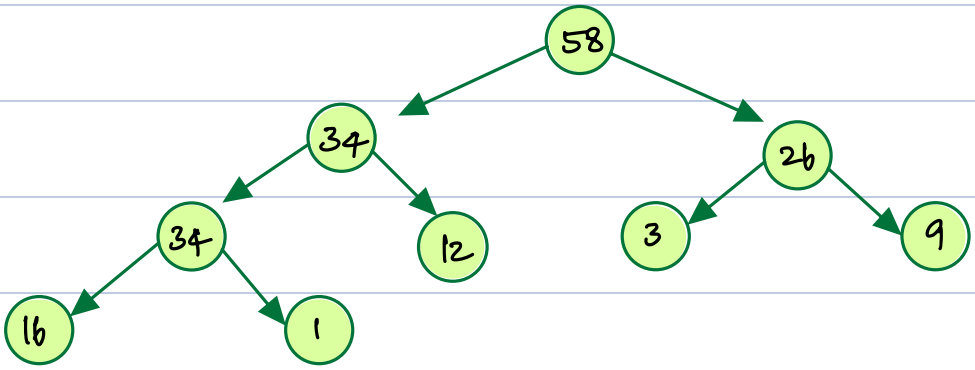
→ Max Heap ∀ nodes data >= children node

No relationship between left & right children

Example :

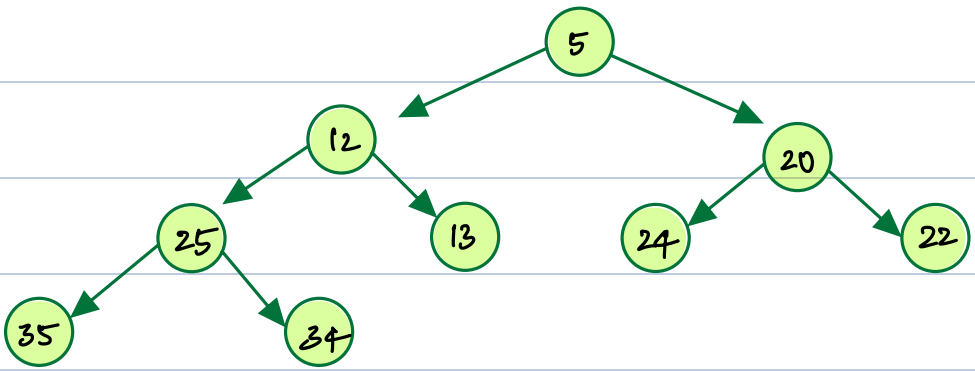


Min heap



Max Heap

Array Implementation of Heaps



Min Heap

Complete

B. Tree

Visualize

5	12	20	25	13	24	22	35	34
0	1	2	3	4	5	6	7	8

parent	left	Right
0	1	2
1	3	4
2	5	6
3	7	8

← Index

Given parent index i

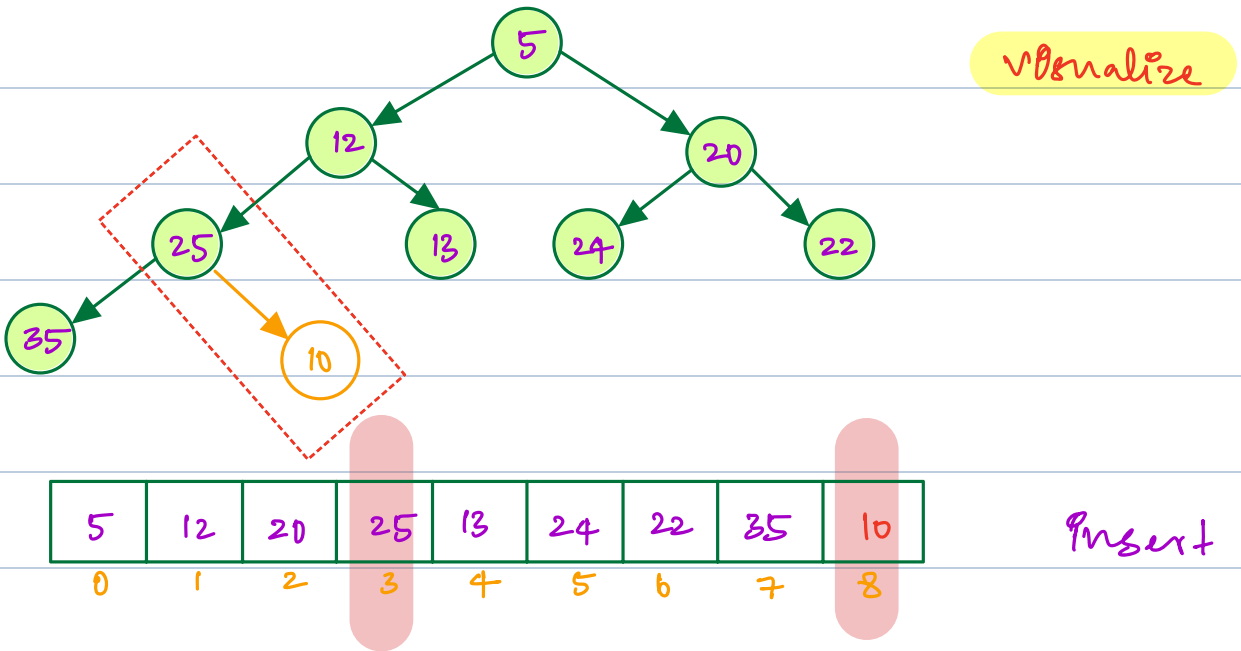
left child index = 2i+1

right child index = 2i+2

Given child index i (left or Right)

parent index = $\frac{(i-1)}{2}$

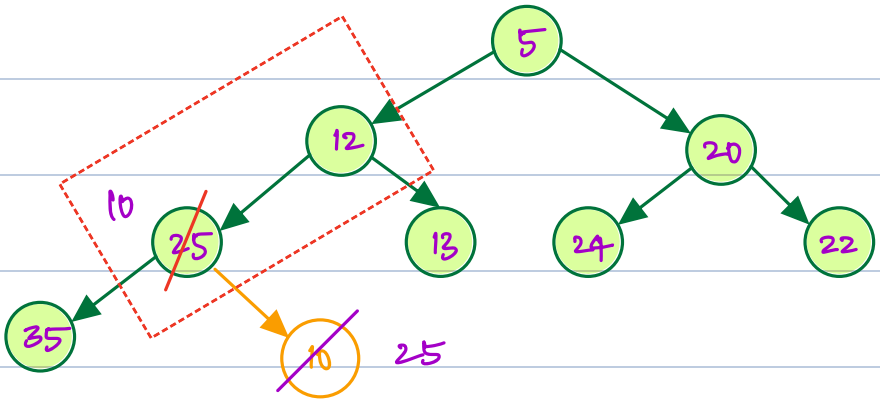
Insertion in Min Heap

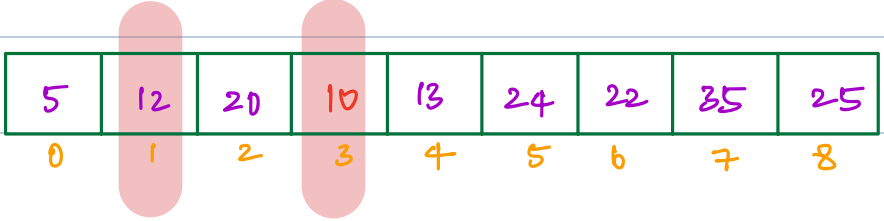


child index 8, ⟶ parent index $\frac{(8-1)}{2}$

3

arr[3] <= arr[8] ⟶ swap as not true.

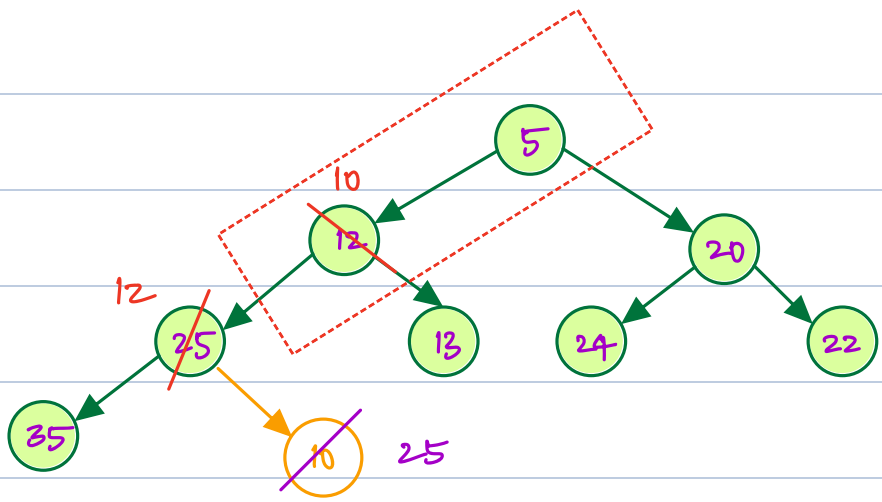


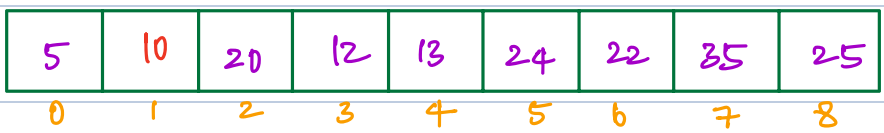


child index (3) ⟶ parent index $\frac{(3-1)}{2}$

(1)

arr[1] <= arr[3] ⟶ swap since not true





child index 1 ⟶ parent index 0

Stop

Quiz 2

Time complexity of Insertion

O(log(n))

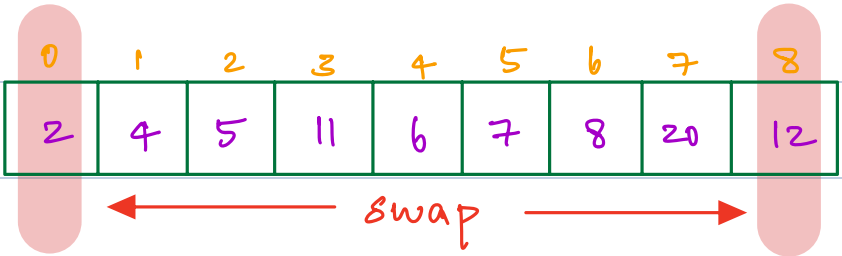
pseudo code

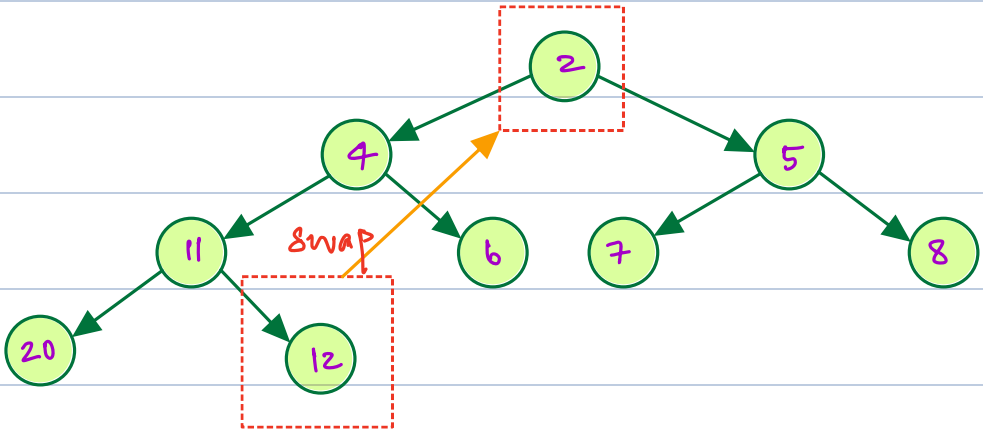
```
// assume we have an existing heap called heap
heaps.append(val)
i = heaps.length - 1;
while (i >= 0) {
    pi = (i-1)/2;
    if (heaps[pi] > heaps[i])
        swap(heaps, pi, i)
        i = pi
    else
        break;
}
```

Extract Min

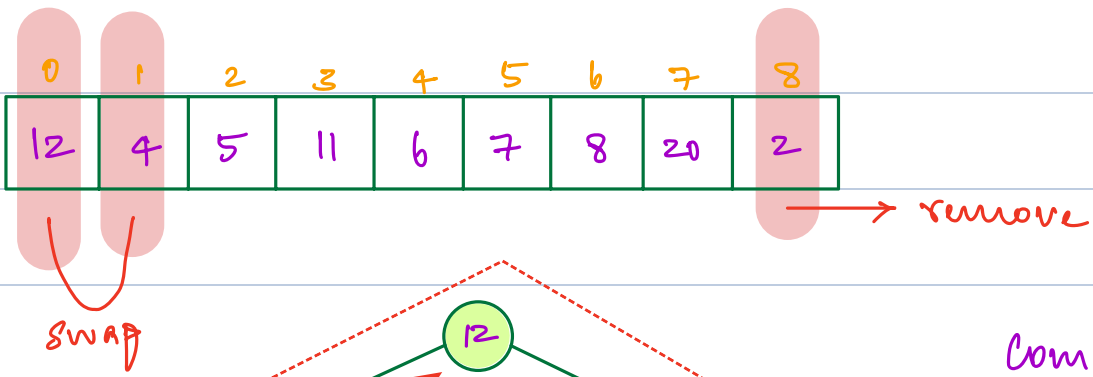
In a min heap where do we have the least element ?

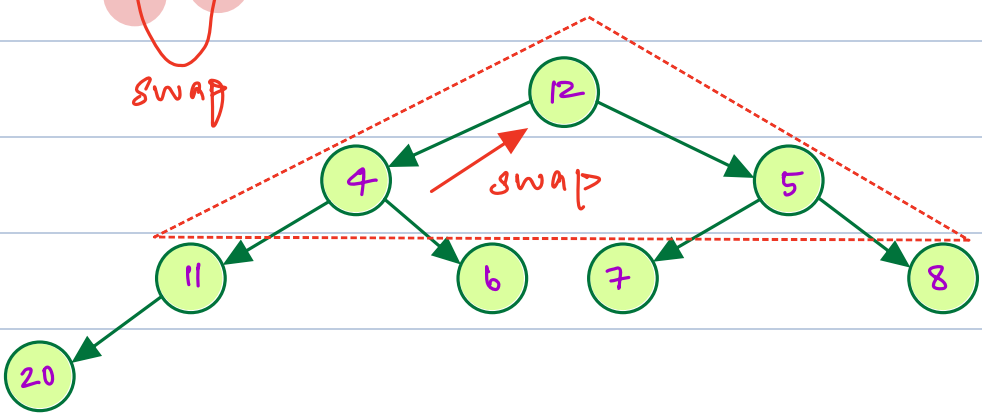
ROOT NODE





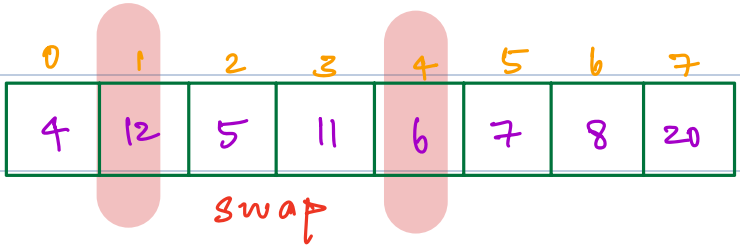
which node can we remove such that it is still complete Binary Tree



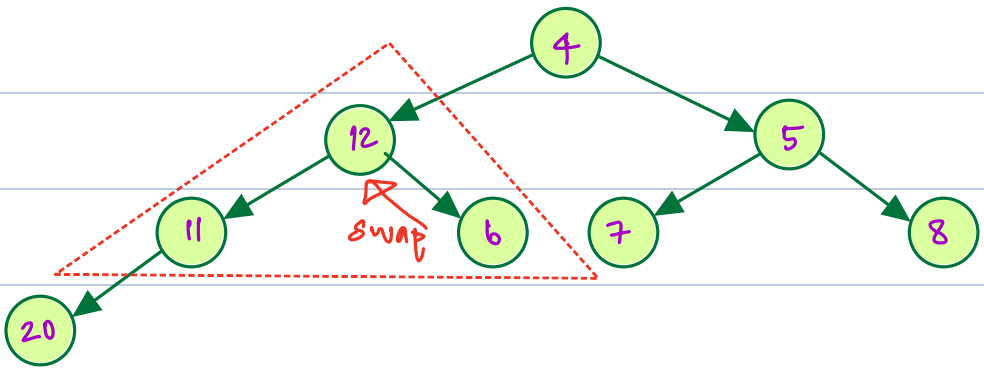


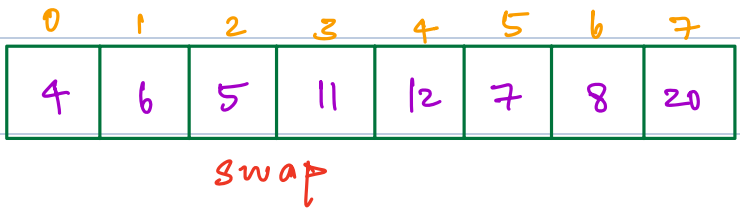
Compare the modified node with its left and right child

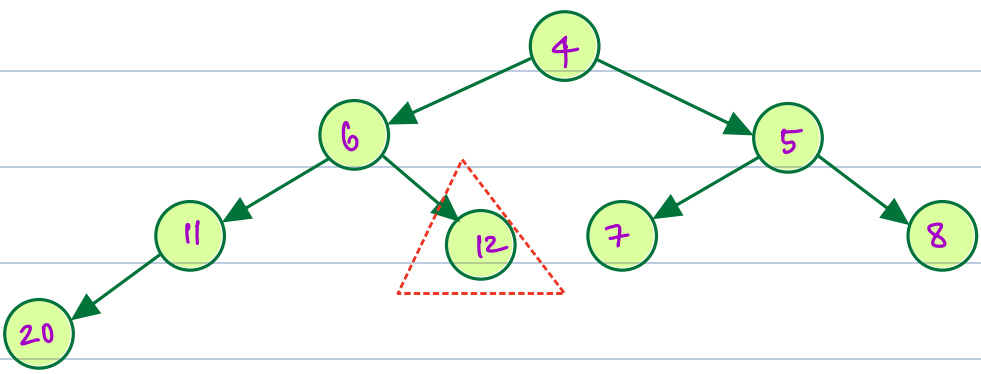
swap with smallest child data



compare with children and swap with smallest child







no more children to swap

pseudo code

```
swap (heap, 0, n-1);
heap.remove (n-1);
heapify (heap, 0);
```

```
void heapify ( heap [], i) {
    while (2i+1 < N) {
        // handle if right child exist or
        // not
        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;
        }
    }
}
```

- T.C = O(log n)

Break till 10:40 pm

Build a Heap

A =	5	13	-2	11	27	31	0	19
	0	1	2	3	4	5	6	7

↓ sort the array

Idea 1 :

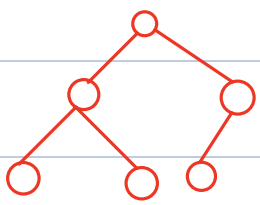
-2	0	5	11	13	19	27	31
0	1	2	3	4	5	6	7

T.C = O(nlogn) S.C = O(n)

Build heap without sorting

Observations

# nodes	1	2	3	4	5	6	...
# leaves	1	1	2	2	3	3	...

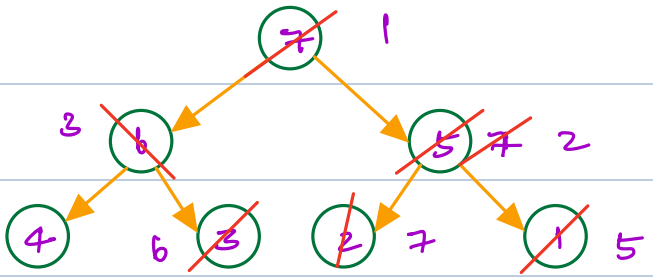


→ $\frac{\#\text{ nodes} + 1}{2}$ Almost Half

arr =

7	6	5	4	3	2	1
0	1	2	3	4	5	6

→ construct MIPn Heap



# nodes	swaps / node
$n/8$	2
$n/4$	1
$n/2$	0

Time complexity depends on no. of swaps.

$$\# \text{ swaps} = 0 * \frac{n}{2} + 1 * \frac{n}{4} + 2 * \frac{n}{8} + \dots$$

$$= \sum i * \frac{n}{2^{i+1}}$$

$$= \frac{n}{2} \sum \frac{i}{2^i} \longrightarrow \text{AGP}$$

expanding $\frac{i}{2^i}$

$$S = \frac{1}{2} + \frac{2}{2^2} + \frac{3}{2^3} + \dots$$

$$\frac{S}{2} = \frac{1}{2^2} + \frac{2}{2^3} + \frac{3}{2^4} + \dots$$

$$S - \frac{S}{2} = 1/2 + 1/2^2 + 1/2^3 + \dots$$

$$\frac{S}{2} = \boxed{\frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^3} + \dots} \longrightarrow \text{GP}$$

$$\frac{S}{2} = \frac{1/2}{1 - 1/2} = 1 \Rightarrow S = 2$$

$$T.C = n/2 * S = n/2 * 2 = \boxed{n}$$

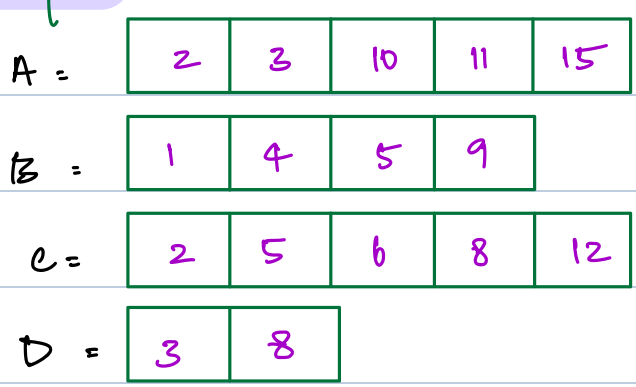
pseudo code

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

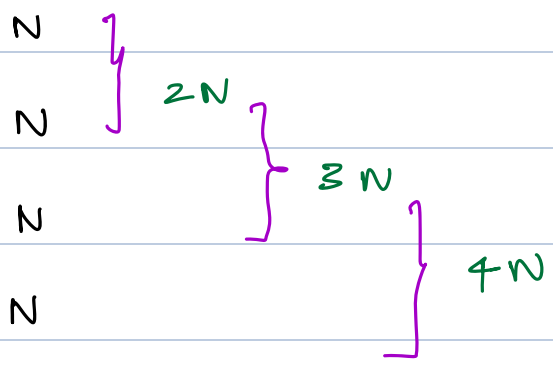
Quiz 3: T.C = O(n)

Q → Merge given K sorted arrays into a single sorted array

example



K arrays each size N

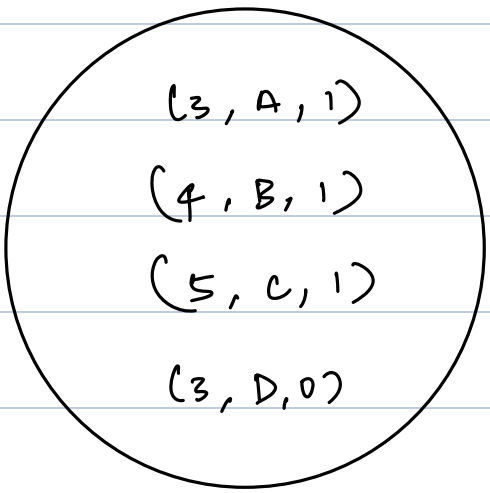


Overall T.C = $N * (2 + 3 + 4 + \dots K)$

$= O(N * K^2)$

Get smallest from all K arrays at every step

Ques 4: Min Heap



(Array, reference, i)

Steps

1. Add zeroth index values to initialize the min Heap
2. pick the smallest element from Heap and insert the next element from same array

Final array

1 2 2

$T.C = O(x \log n)$

x is count of all elements in array