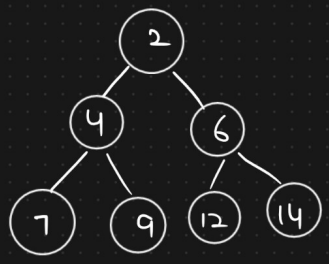
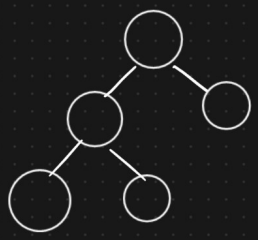
Heap Data Structure



2 child or no child
↳ Leaf nodes

full Binary Tree



Almost complete Binary Tree

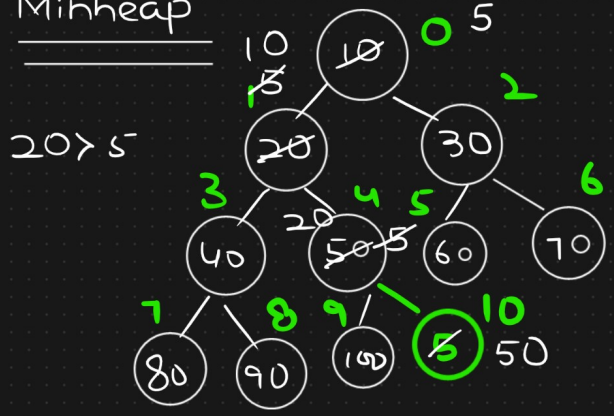
{ 1) Left to right
2) upper level completely filled ⇓ Lower level

{ 1) Minheap → Parent node data < Child node data
2) Maxheap
↳ Parent node data > Child node data

Insertion

10 > 5

Minheap



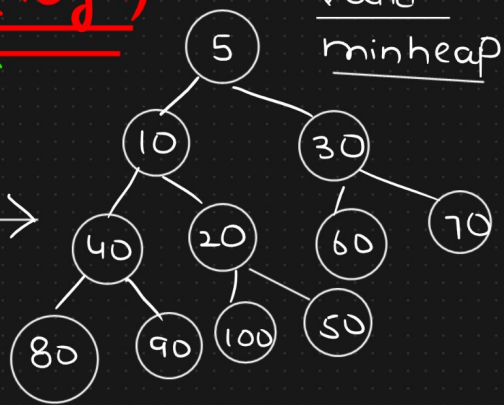
20 > 5

50 > 5

3 ⟺ O(log n)

SWAPS ⟺

Valid minheap



Minheapify Bottom

Array

0	1	2	3	4	5	6	7	8	9	10
10	20	30	40	50	60	70	80	90	100	5

Height of CBT → Minheap / Maxheap

Property

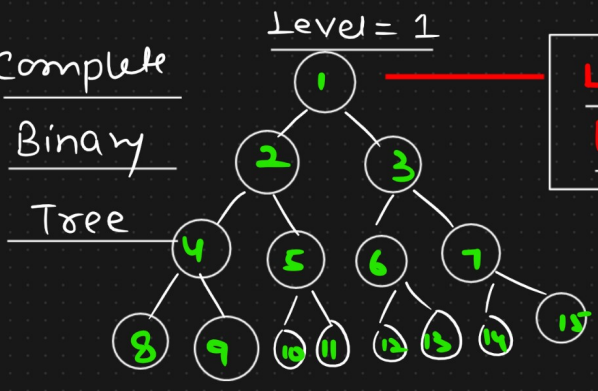
Complete Binary Tree

Worst Case Scenario

1. number of node $= 2^h - 1$

$= 2^4 - 1$

$= 15$



Level = 0
Height = 1

$n = 2^h - 1$

$n+1 = 2^h$

$h = \log_2(n+1)$

Height = Level + 1

2.

Leaf node $= \left\lceil \frac{n}{2} \right\rceil = \left\lceil \frac{15}{2} \right\rceil = 8$

non-leaf node $= \left\lfloor \frac{n}{2} \right\rfloor = 7$

$l = \log_2(n+1)$

Total number of elements $\Rightarrow n 2^n$

Time complexity $\Rightarrow$ of insertion in minheap

✓ a) $O(n)$ c) $O(\log n)$ ✗

✗ b) $O(n \log n)$ d) $O(n^2)$ ✗

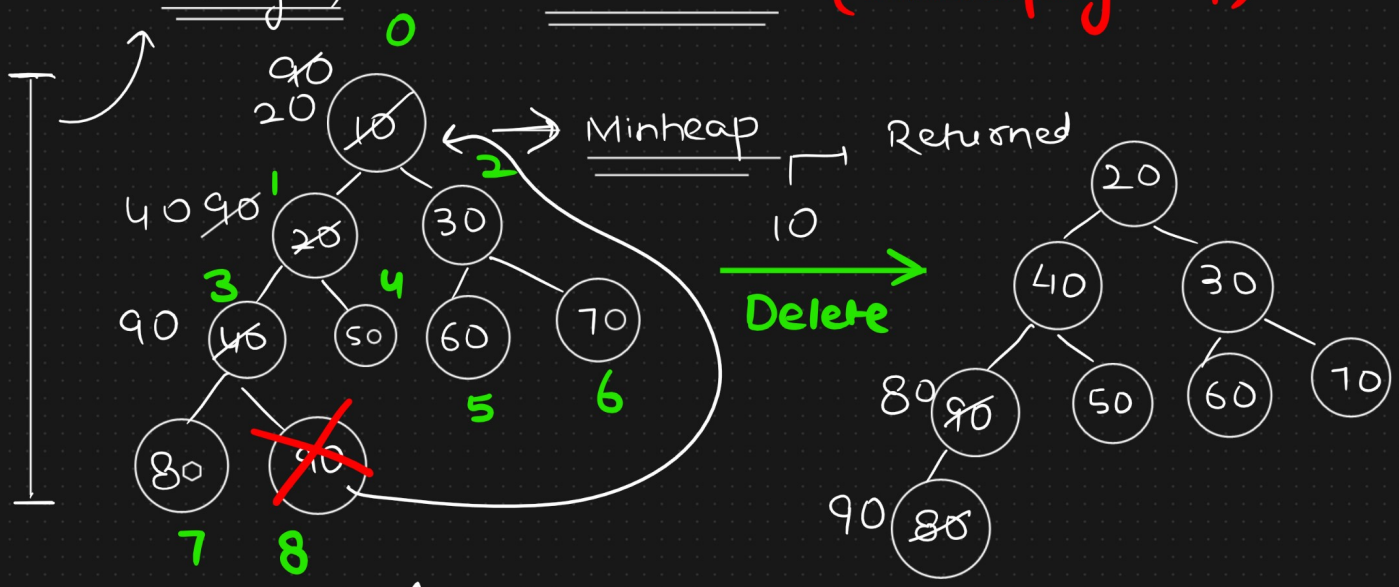
n ——— $\log n$

$n 2^n$ ——— $\log(n \cdot 2^n)$ → $\log n + \log 2^n$ → $\log n + n \cdot \log 2$

$\Rightarrow \log n + n$

$\Rightarrow O(n)$

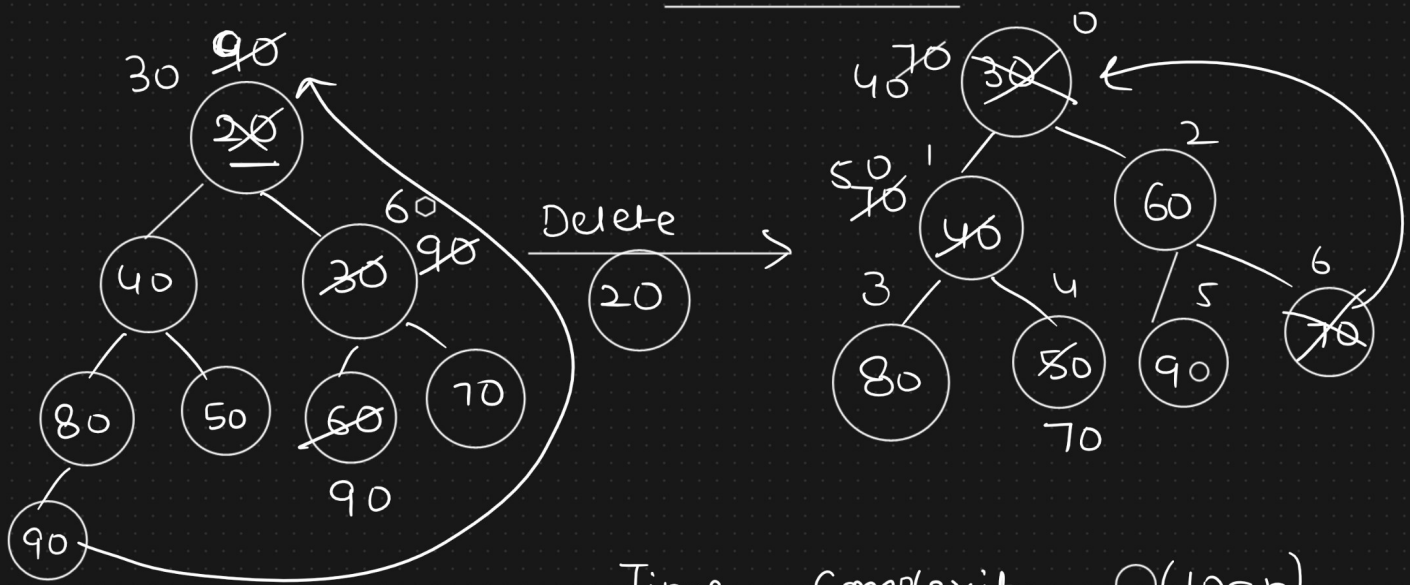
$O(\log n)$ Deletion (Minheapify Top)



Array

0	1	2	3	4	5	6	7	8
10	20	30	40	50	60	70	80	90

Deletion $\longrightarrow$ min. value in the tree



Time Complexity = $O(\log n)$

HeapSort

1) Create/build minheap ⟶ n
2) Delete the elements from minheap & store it.

$\Downarrow$ n . (Delete) $\Rightarrow$ $n * \log n$ =

$n + n.\log n$

Sorted array (Ascending order)

$\searrow$ $O(n \log n)$

10	20	30	40	50	60	70	80	90

(Ascending order)

Inplace ⟶ Space complexity - O(1)

20	30	40	50	10

Insertion

comparison → $\log n$

swaps → $\log n$

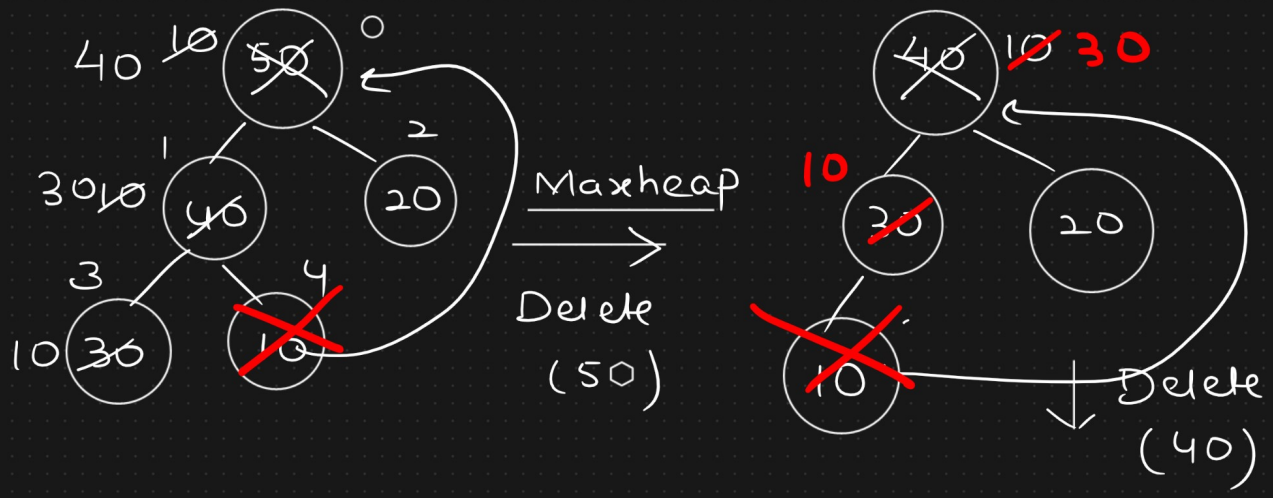
$O(\log n)$

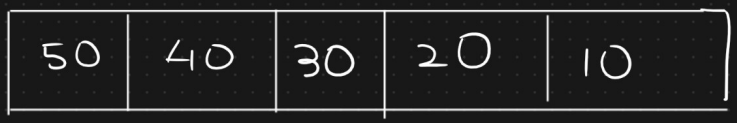
Deletion

comparison → $2 \log n$

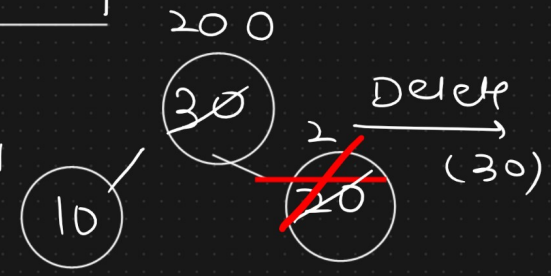
swap → $\log n$

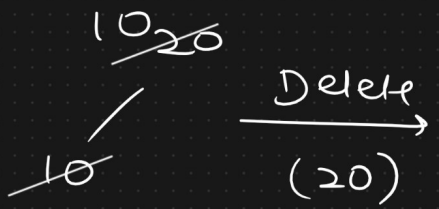
$O(\log n)$

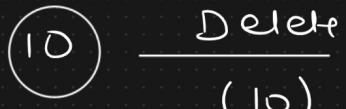




Sorted array (Descending order)

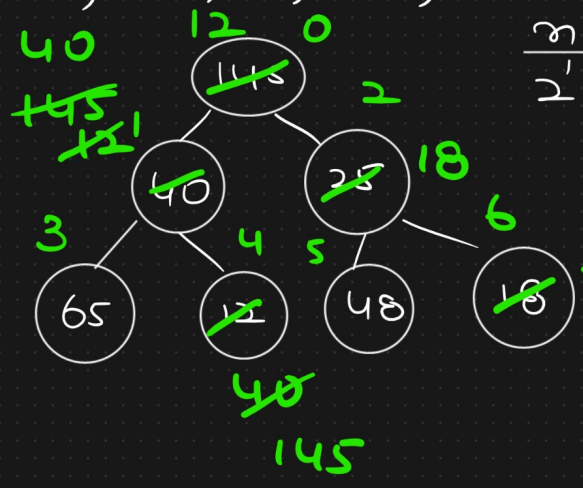






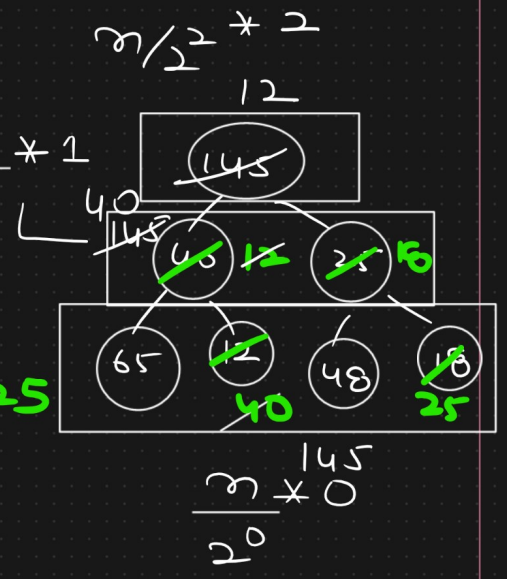
Build of minheap

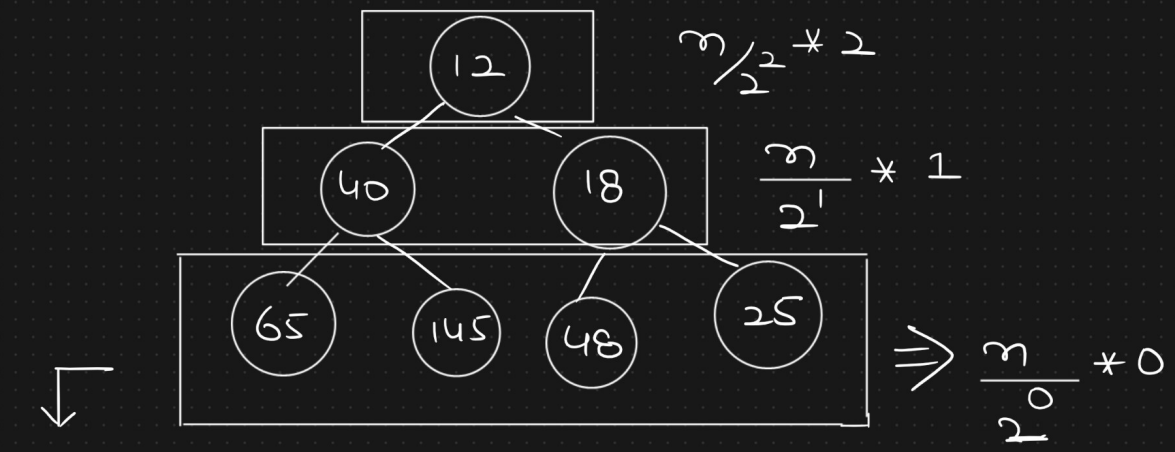
145, 40, 25, 65, 12, 48, 18



Minheap

$$\begin{cases} n//2 - 1 \\ 6//2 - 1 \\ = 2 \end{cases}$$





Total no. of nodes in leaf node = n

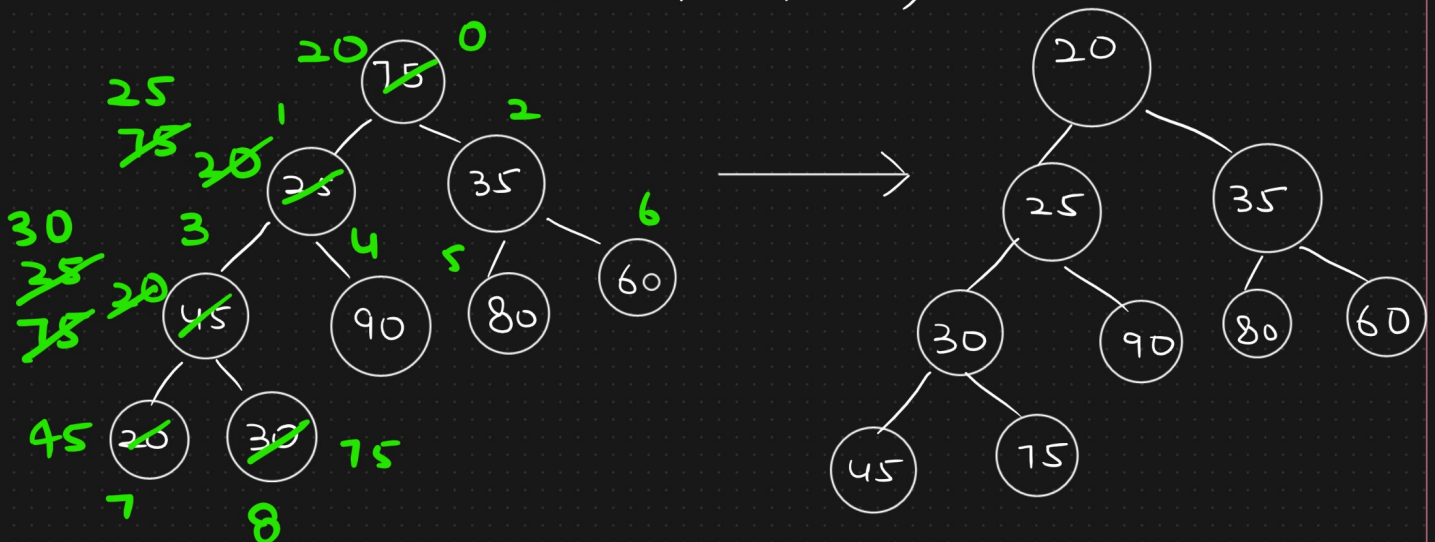
Total number of swaps

$$S = \frac{n}{2^0} * 0 + \frac{n}{2^1} * 1 + \frac{n}{2^2} * 2 + \cdots + \frac{n}{2^{\log n}} * \log n$$

$$= n\left(\frac{1}{2^1} + \frac{2}{2^2} + \frac{3}{2^3} + \cdots + \frac{\log n}{2^{\log n}}\right) \quad \text{---- (1)}$$

$$\frac{S}{2} = n\left(\frac{1}{2^2} + \frac{2}{2^3} + \cdots + \frac{\log n - 1}{2^{\log n}} + \frac{\log n}{2^{\log n + 1}}\right) \quad \text{(2)}$$

75, 25, 35, 45, 90, 80, 60, 20, 30



$S - S/2$ (GP Series)

$$\frac{S}{2} = n\left[\left(\frac{1}{2^1} + \frac{1}{2^2} + \frac{1}{2^3} + \frac{1}{2^4} + \cdots + \frac{1}{2^{\log n}}\right) - \frac{\log n}{2^{\log n+1}}\right]$$

$r = 1/2$

$$S = \frac{a(1-r^n)}{1-r} \quad r < 1$$

$$\frac{S}{2} = n\left(\frac{\frac{1}{2}\left(1 - \frac{1}{2^{\log n}}\right)}{1 - \frac{1}{2}}\right) - \frac{\log n}{2^{\log n+1}}$$

$$\frac{S}{2} = n\left[\left(\frac{2^{\log n} - 1}{2^{\log n}}\right) - \frac{\log n}{2^{\log n} \cdot 2}\right]$$

$\Rightarrow 2^{\log_2 n}$

$\Rightarrow n^{\log_2 2} = n$

$$\frac{S}{2} = n\left\{\left(\frac{n-1}{n}\right) - \frac{\log n}{n \cdot 2}\right\}$$

$$\frac{S}{2} = \cancel{n} \cdot \frac{n-1}{\cancel{n}} - \cancel{n} \cdot \frac{\log n}{\cancel{n} \cdot 2}$$

$$\frac{S}{2} = n - 1 - \frac{\log n}{2}$$

$$S = 2n - 2 - \log n$$

$$= O(n) \Leftarrow$$

https://docs.python.org/3/library/heapq.html

Psuedocode

$O(n)$

Psuedocode

```
buildheap(arr, n):
    StartIndx = n//2 - 1

    for i in range(StartIndx, -1, -1):
        ↳ (StartIndex, 0)

        heapify(arr, n, i)
```

i → Parent node

create a min-heap or max-heap

```
heapify(arr, n, i):
    smallest = i
    l = 2*i + 1
    r = 2*i + 2
```

* → minheap

```
    arr(l) < arr(smallest):
        smallest = l

    arr(r) < arr(smallest):
        smallest = r

    heapify(arr, n, smallest)
```

Recursion