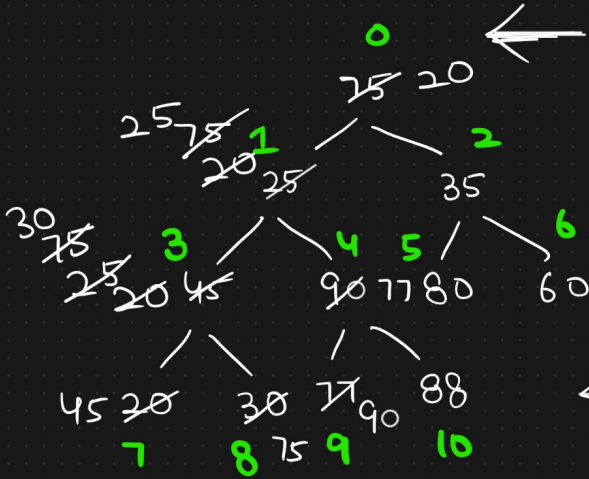
Build Heap → O(n)

This thing was already discussed in the previous lecture-5

75, 25, 35, 45, 90, 80, 60, 20, 30, 77, 88

complete Binary Tree (CBT)



swaps = 3 (almost)

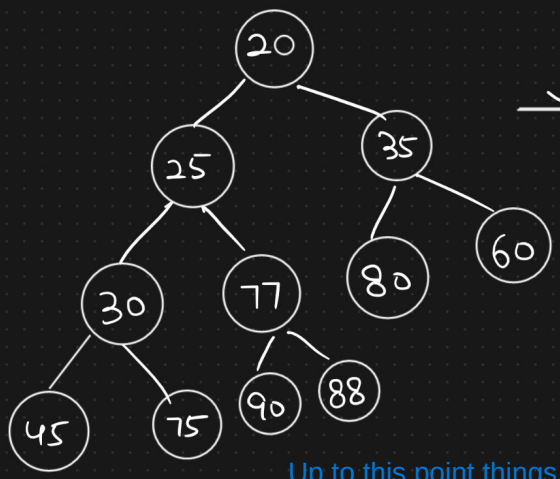
Minheap

swaps = 2 (almost)

swaps = 1 (almost)

Leaf Node

↳ # swaps = 0

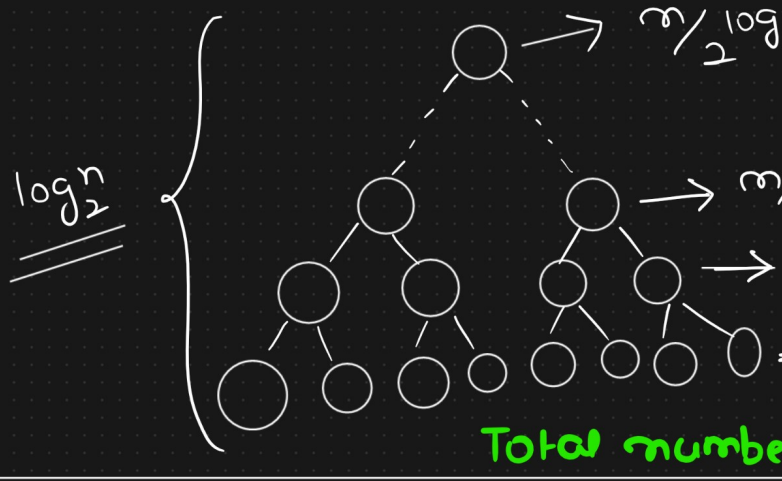


→ Valid Minheap

Up to this point things were discussed in the last lecture-5

Heapify Method

Time complexity to build heap = O(n)



$k = \log_2 n$

⇓

CBT

Total number of swaps

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

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

Divide ①/2 → GP series

$$\frac{S}{2} = n\left(\frac{1}{2^2} + \frac{2}{2^3} + \frac{3}{2^4} + \dots + \frac{\log n - 1}{2^{\log n}} + \frac{\log n}{2^{\log n + 1}}\right) \quad \text{—②}$$

Subtraction of ② from ①

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

↳ valid GP Series

$a = 1/2$ ⇓ first element

(sum of GP series) $r < 1$

$r = \frac{1/2^2}{1/2^1} = 1/2 < 1$ ⇓ common ratio

Sum of GP series $= \frac{a(1-r^n)}{1-r}$; $r < 1$

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

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

$2^{\log n+1} \Rightarrow 2^{\log n} \cdot 2$

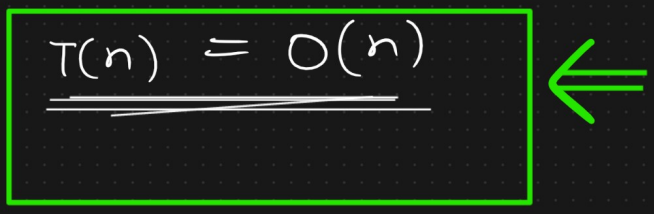
$$2^{\log_2 n} = n^{\log_2 1} = n$$

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

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

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

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



To summarize O(n) is the time complexity in worst case scenario for building the Heap(Min/Max). And the method that is basically used for building the respective Heap is known as Heapify Method