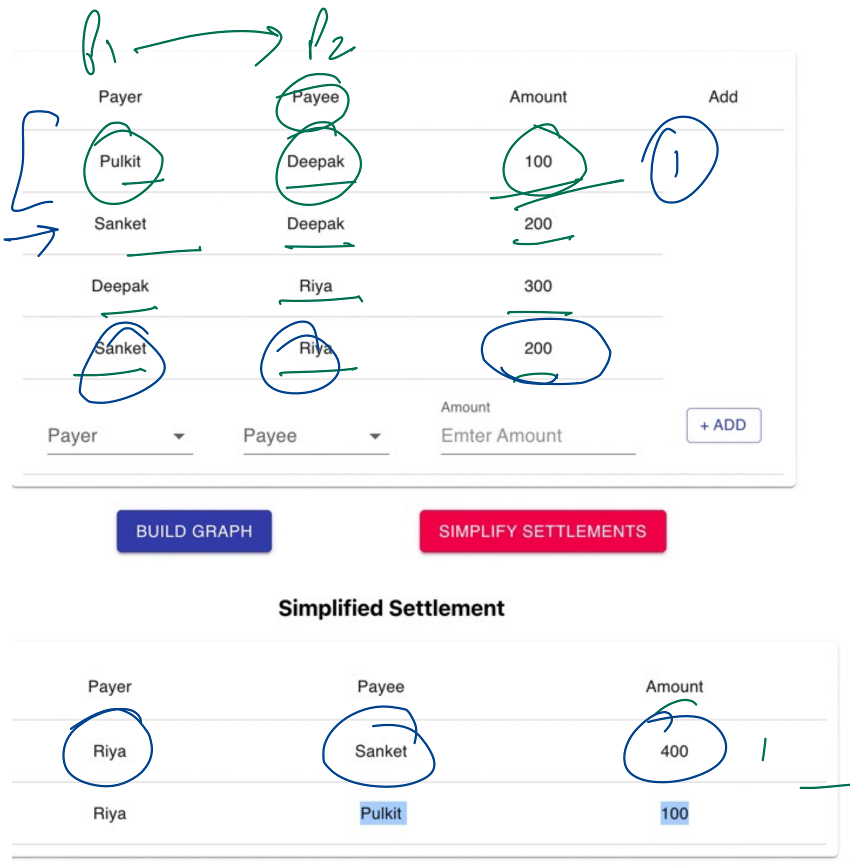
$$\begin{tikzcd}
A \arrow[r, "u"] \arrow[d, "x"'] & D \\
B \arrow[r, "y"'] & C \arrow[u, "z"']
\end{tikzcd}$$

Debit	Credit
A → (-60)	B (50)
	C (10)
A (-10)	B (0)
	C (10)



Payer	Payee	Amount	Add
Pulkit	Deepak	100	
Sanket	Deepak	200	
Deepak	Riya	300	
Sanket	Riya	200	

Amount: Emter Amount

Simplified Settlement

Payer	Payee	Amount
Riya	Sanket	400
Riya	Pulkit	100

Deepak → pulkit (100)

Deepak → Sanke (200)

Riya → Deepak (300)

Riya → Sanket (200)

key value → HashMap

key	value
Deepak	0
Sanket	~~400~~ 0
Riya	~~-500~~ ~~-600~~ 0
Pulkit	~~100~~ 0

1 Calculate the total debt or credit for a person

A B C D → agra

Cost of 1 ticket is 50

A paid for ticket → 200 → A, B, C, D

A → B → 50

A → C → 50

A → D → 50

A → A 50

How to segregate who is under credit & who is under debit.

$p \rightarrow a$ $\qquad a < 0$ ($p \rightarrow$ debit)

$a > 0$ ($p \rightarrow$ credit)

$a == 0$ (p is settled)

1. from all the person who are under debit, get the one with max debit.

from all the persons who are under credit get the one with max credit.

And settle them

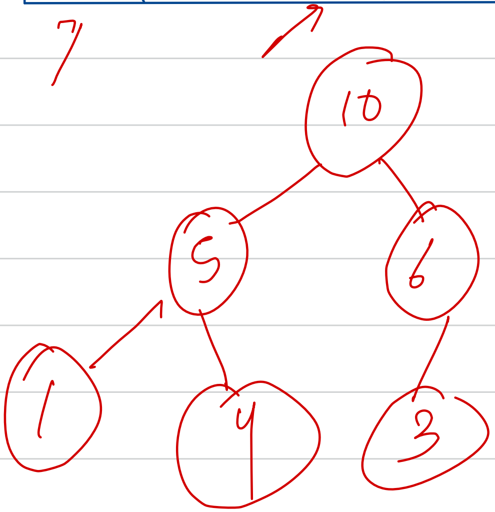
Max heap

① heap is a complete B.T. ←

② Priority of a parent is higher than child.

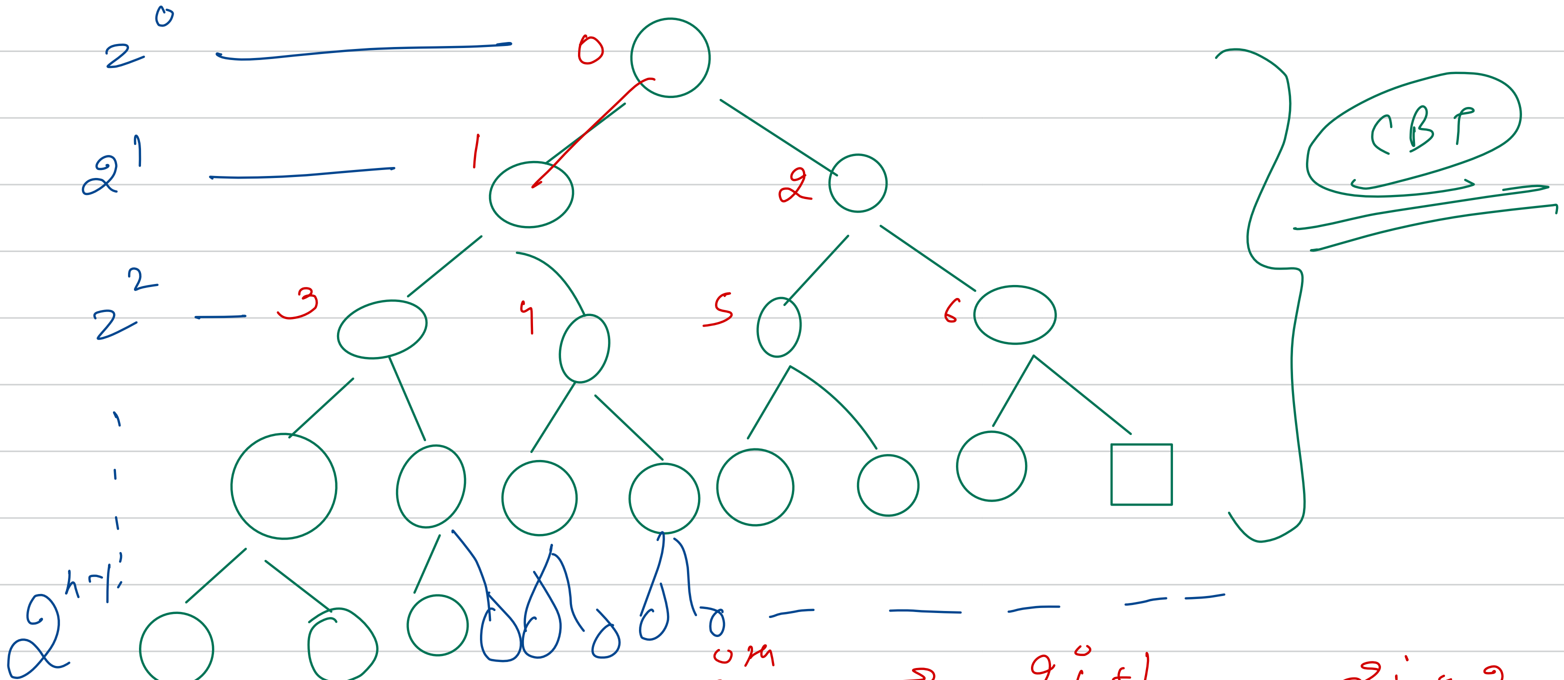
5	1	3	4	10	6

→ $O(n)$



upheapify

$O(n \log n)$



i^{th} parent → $2i+1$, $2i+2$

j^{th} child → $\frac{(j-1)}{2}$ → parent

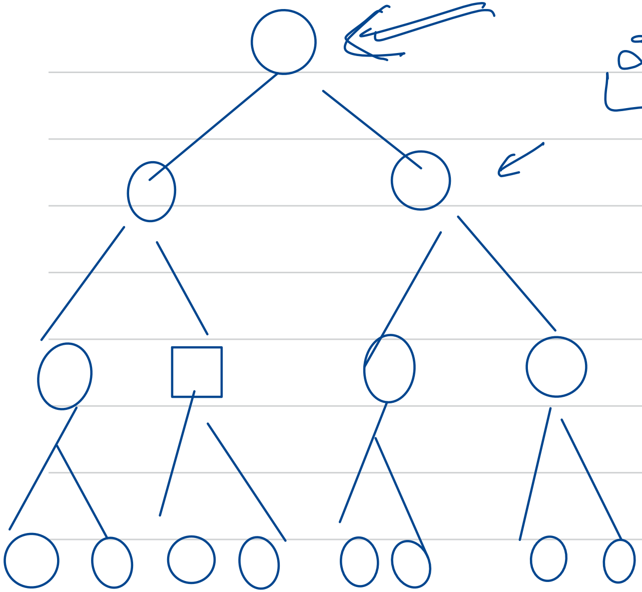
$$\underbrace{2^0 + 2^1 + 2^2 - - - - 2^{h-1}} = n$$

Sum of gp

$$2^h - 1 = n$$

$$2^h = n + 1$$

$$h = \log_2(n+1)$$



$$\underbrace{2^0 + 2^1 + 2^2 - - - - - - 2^{h-2}} + \boxed{2^{h-1}} = n$$

$$2^{h-1} - 1 + 2^{h-1} = n$$

$$2(2^{h-1}) = n+1$$

nodes at last level

leafs

$$\boxed{2^{h-1}} = \boxed{\frac{(n+1)}{2}}$$

→ How many elements on last level

$$2^{h-1}(h-1) + 2^{h-2}(h-2) - - - - - - - - - 2(1) + 1(0)$$

$$S = \underbrace{2^{h-1}(h-1)} + 2^{h-2}(h-2) - - - \quad 2^{2}(2) + 2^{1}(1) + \cancel{1(0)}$$

$$2S = 2^{h}(h-1) + \underbrace{2^{h-1}(h-2)} - - - - - - - 2^{3}(2) + 2^{2}(1)$$

$$2S - S = 2^{h}(h-1) + \underbrace{2^{h-1}(-1) + 2^{h-2}(-1) - - - - - \cdots 2^{2}(-1)} + 2$$

$$S = 2^{h}(h-1) + (-1)2^{2}(2^{h-2} - 1) + 2$$

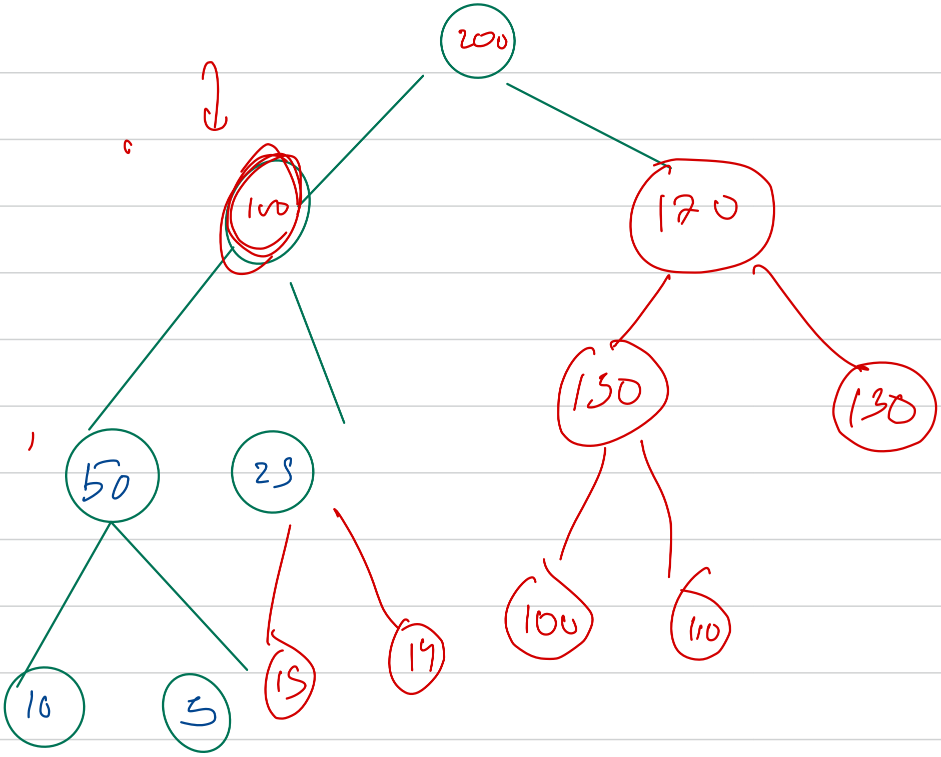
$$S = 2^{h}h - 2^{h} + (-1)(2^{h} - 2) + 2$$

$$S = \boxed{2^{h}h} - 2^{h} - 2^{h} + 2 + 2$$

$$\longrightarrow 2^{h} \longrightarrow 2^{\log_2 n} \longrightarrow \underline{n}$$

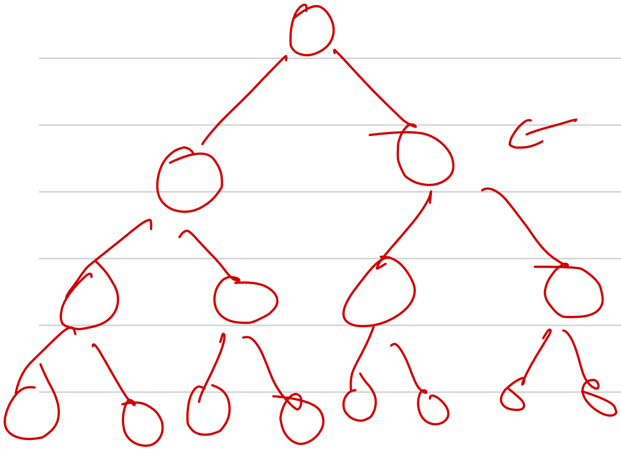
$$\underline{O(n \log n)}$$

MAX
Heap



Downheapify

Remove → O(log n)



$$S = 2^{h-1} \times 0 + \left(2^{h-2} \times 1\right) + \left(2^{h-3} \times 2\right) - - - - \quad 2(h-2) + 1(h-1)$$

$$S = \quad + \left(2^{h-2} \times 1\right) + \left(2^{h-3} \times 2\right) - - - 2^{2} \times (h-3) + 2\,(h-2) + 1\,(h-$$

$$2S = 2^{h-1} \times 1 + 2^{h-2} \times 2 + 2^{h-3} \times 3 - - - - - \; 2^{2}(h-2) + 2(h-1)$$

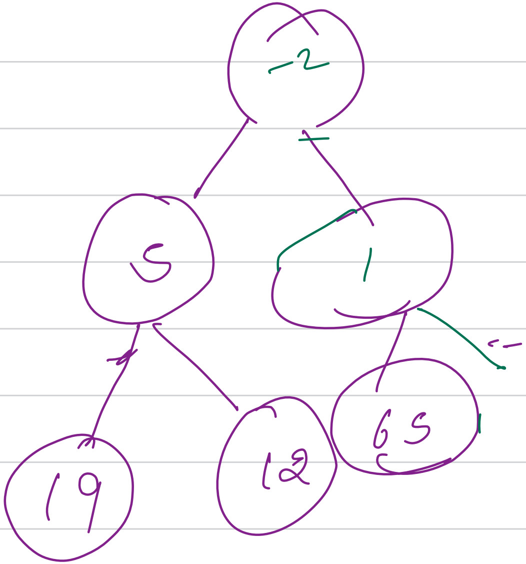
$$2S - S = 2^{h-1} \times 1 + \underbrace{2^{h-2} + 2^{h-3} - - - - - - 2^{2} + 2}_{GP} + (h-1)$$

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

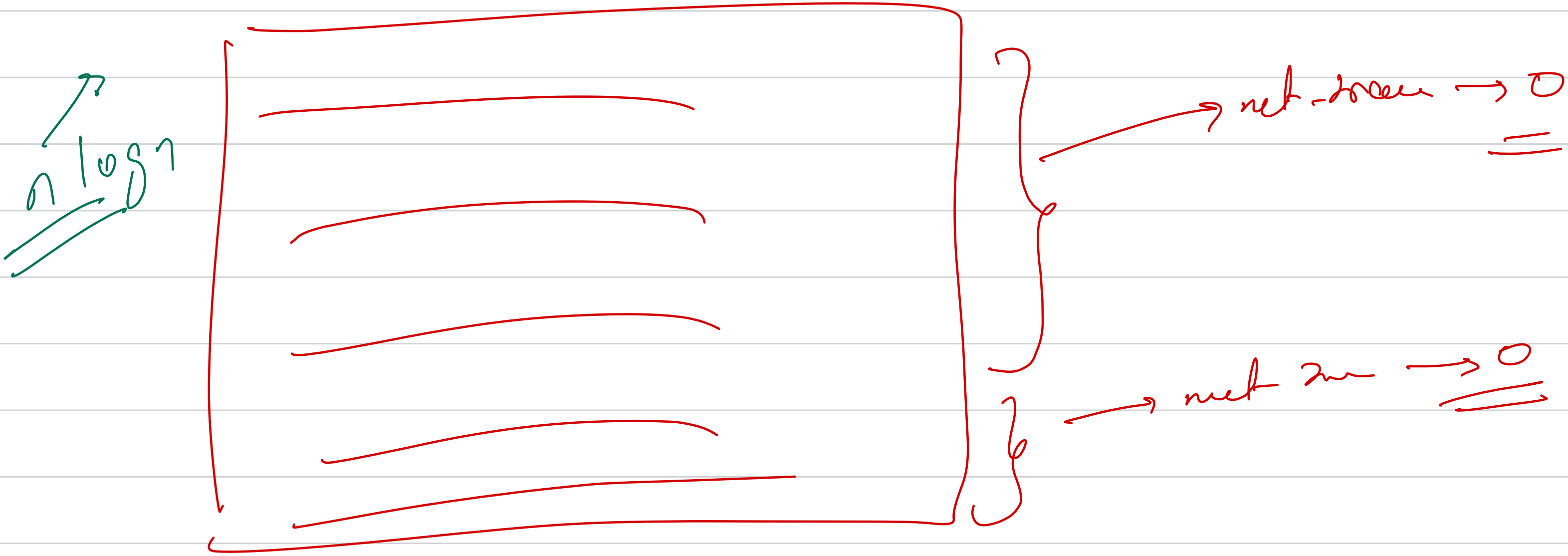
$$= 2^{h-1} + 2^{h-1} - 2 + h - 1$$

$$\Rightarrow 2 \times 2^{h-1} - 3 + h \Rightarrow 2^{h} - 3 + h$$

$$n - 3 + \log n \rightarrow O(n)$$


-2
5
1
19
12
65
-1

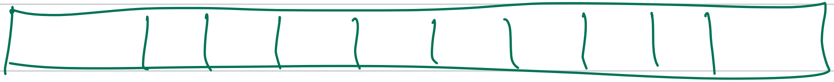
⇒ Heap → net transaction = 0



TC → $O(n + n\log n)$ → $O(n\log n)$

NP

sol^n is not in polynomial time, but verifiable in Polynom



NP Complete

Find all & print subsets which sum to zero

Backtracking

2, 3, -5, 1, 2, -1, 3, -5

K group

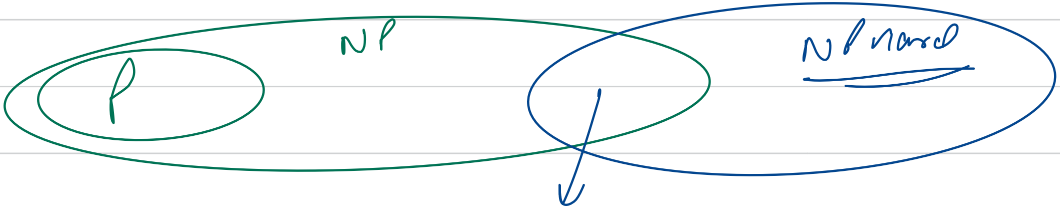
{2, 3, -5}

{1, 2, -1, 3, -5}

P →

NP →

NP-Hard → atleast as hard as the hardest proble in NP



NPC → brute force → algo can soln & correctness is verifiable in polynomial time install node

install yarn

yarn install

yarn . start → local mach

npm run build → build folder → github
&
netlify