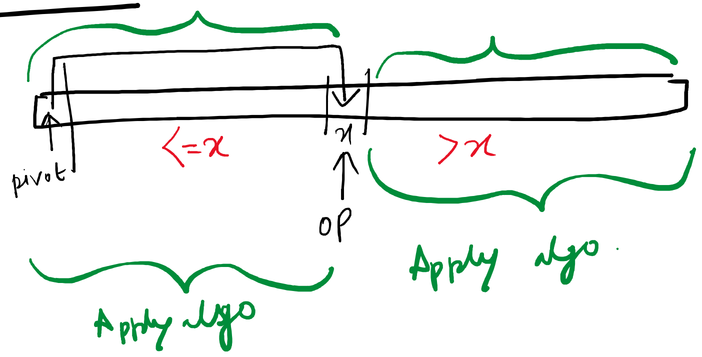
27 January 2025 23:18

Quick Sort - Based on divide & conquer technique.



Note -

① pivot element

↓

Aapka choice

Generally,

pivot = arr[0]

or

pivot = arr[last-element]

35 50 15 25 80 20 90 45.

P q ~~q~~ ~~q~~

⇓

35 20 15 25 80 50 90 45

~~p~~ ~~p~~ ~~p~~ p ~~q~~

q ~~q~~

⇓

25 20 15 35 80 50 90 45

op

pivot = 35

p, q

① p ⟶ arr[p] <= pivot.

q ⟵ arr[q] > pivot.

② Chk if p & q have crossed each other

↳ swap(pivot, arr[q]);

③ Chk if p & q have not crossed each other.

Swap (arr[p], arr[q])

left array - 25 20 15 pivot = 25, p

p q

p p

⇓

15 20 25 ← OP

Right arr - 80 50 90 45 pivot = 80, ~~p = 50~~ p

p p q

arr[q] > pivot

arr[p] <= pivot.

⇓

80 50 45 90 45 > 80 ✗

pivot

p q

q p

45 <= 80

45 > 80 ✗

⇓

45 50 80 90

OP

Merge Sort

TC → O(n log n)

SC → O(n).

Quick Sort - SC → O(1).

Worst Case

TC — ① Best case

② Pivot element ka OP middle mein hon.

$$T(n) = T(n/2) + T(n/2) + n$$

$$T(n) = 2T(n/2) + n \longrightarrow$$

$$T(n) = n \log n.$$

② Worst Case

Agar array already sorted.

$\xrightarrow{n-1}$

$\leftarrow$ 1 2 3 4 5

q p $\Downarrow$ ~~q~~

pivot = 1

$$T(n) = T(n-1) + n \longrightarrow$$

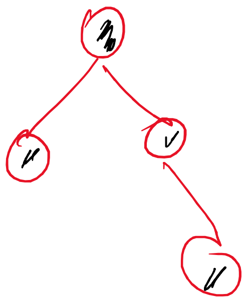
$$O(n^2).$$

Heap Sort.

Tree —

Binary Tree → Tree that have

Binary Tree → Tree that have 0 or 1 or 2 children.

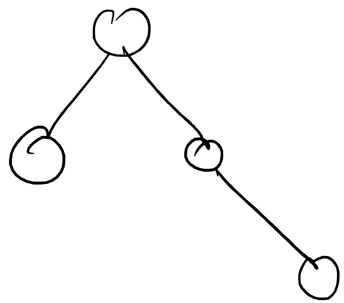


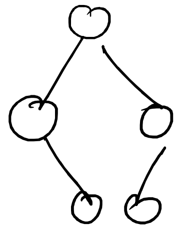
Leaf node – Node jiska 0 children.

Heap – ① Almost complete binary tree.

↓

Node add karo toh left direction se karo.

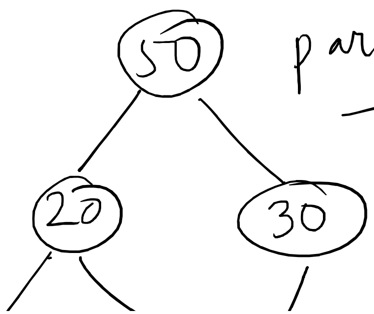




② Max-Heap → Parent > child

Min Heap → Parent < child.

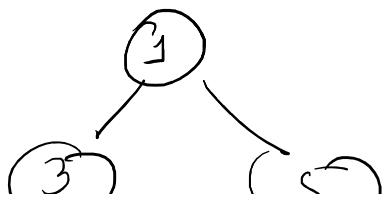
Max heap –

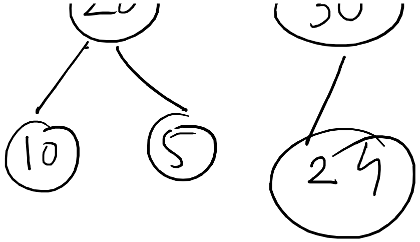


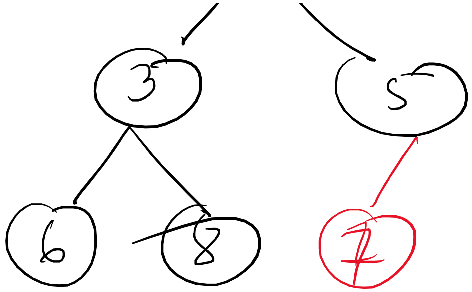
par > child

Min heap –

parent < child







Heapify – 10 20 15 12 40 25 18.

Represent it in the form of Almost complete binary tree.