24 | Friday

8 am

QuickSort

- Sort almost in "place", not require additional array
- Avg. performance $O(n \log n)$ & worst case $O(n^2)$

Divide - Conquer - Combine algorithm.

1 pm

i j

17, 12, 6, 19, 23, 8, 5, 10

pivot element = 12

increment i till find element greter than 12

Decrement j until find element less than 12

when find swap them

∴ Two parts will created elements > 12 & elements < 12

Take individual part and repeat the process

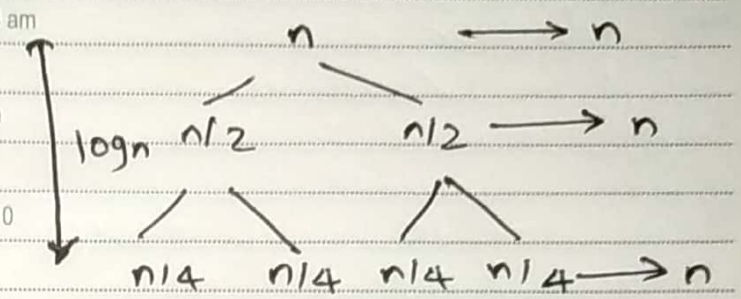
Analysis:

Note: If we lucky, partition splits the array evenly

$O(n \log n)$

25 | Saturday

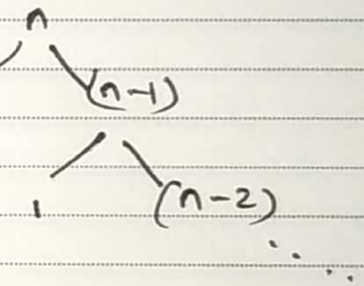
8 am



$O(n \log n)$

Worst case → $O(n^2)$

1 pm



26 | Sunday

How can we make sure that we are usually lucky?

- Partition around 'middle' element.
- Partion around random element (works well in practice)

Randomized Quicksort running time

worst $O(n^2)$

Note: Best $O(n \log_2 n)$

Avg. $O(n \log_2 n)$

Insertion is BST avg. time is $O(n \log n)$.