Quicksort algorithm → Comparison based sort.

→ Divide N Conquer algo

70%

Partition algo

Q => Given a list of size N, and any one element from the list called as the pivot element. You need to segregate all the elements lesser than pivot on the left side & greater than pivot on the right side in any order.

[10, 7, 8, 9, 1, 5] any order — same list

pivot → 7

$O(N)$

ans → [5, 1, 7, 8, 10, 9] (any order)

<7   7<

length of list ≤ 10^5

j m

→ [1, 5, 7, 9, 10, 8]
0 1 2 3 4 5

m = 0

i = [0, 5]

i = 5

→ we can iterate over all the elements & compare with the pivot

arr[i] < pivotvalue

m is represent the boundary of the left side

So everything on left side of m is lesser than pivot.

→ m will iterate till the index we have got our segregation Boundary.

0 1 2 3 4

[1, 3, 1, 4, 9]

$1 < 1$

$m = 0$

for $i = 0$, len(arr) $- 1$

if $a[i] \leq a[pivot]$:

swap (a[i], a[m])

m + +

swap (a[m], a[pivot])

index pivot = 3

$i = [0, 4]$

$m = \cancel{0}\ \cancel{1}\ \cancel{2}\ 3$

$i = \cancel{0}\ \cancel{1}\ \cancel{2}\ \cancel{3}\ \cancel{4}\ 5$

elmt → 4

0 1 2 3 4

→ 1, 9, 3, 4, 7.

index-pivot = 4

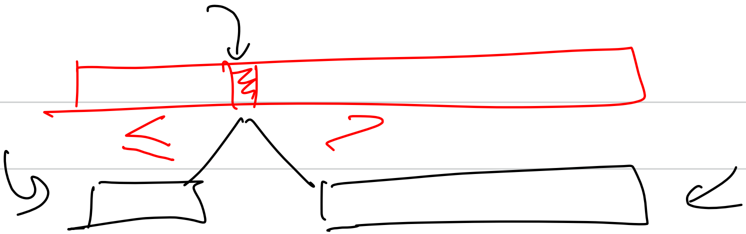
$i = \cancel{0}\ \cancel{1}\ \cancel{2}\ \cancel{3}\ 4$

$m = 0$

Revisit

$<\ \&\ \leq$

quicksort



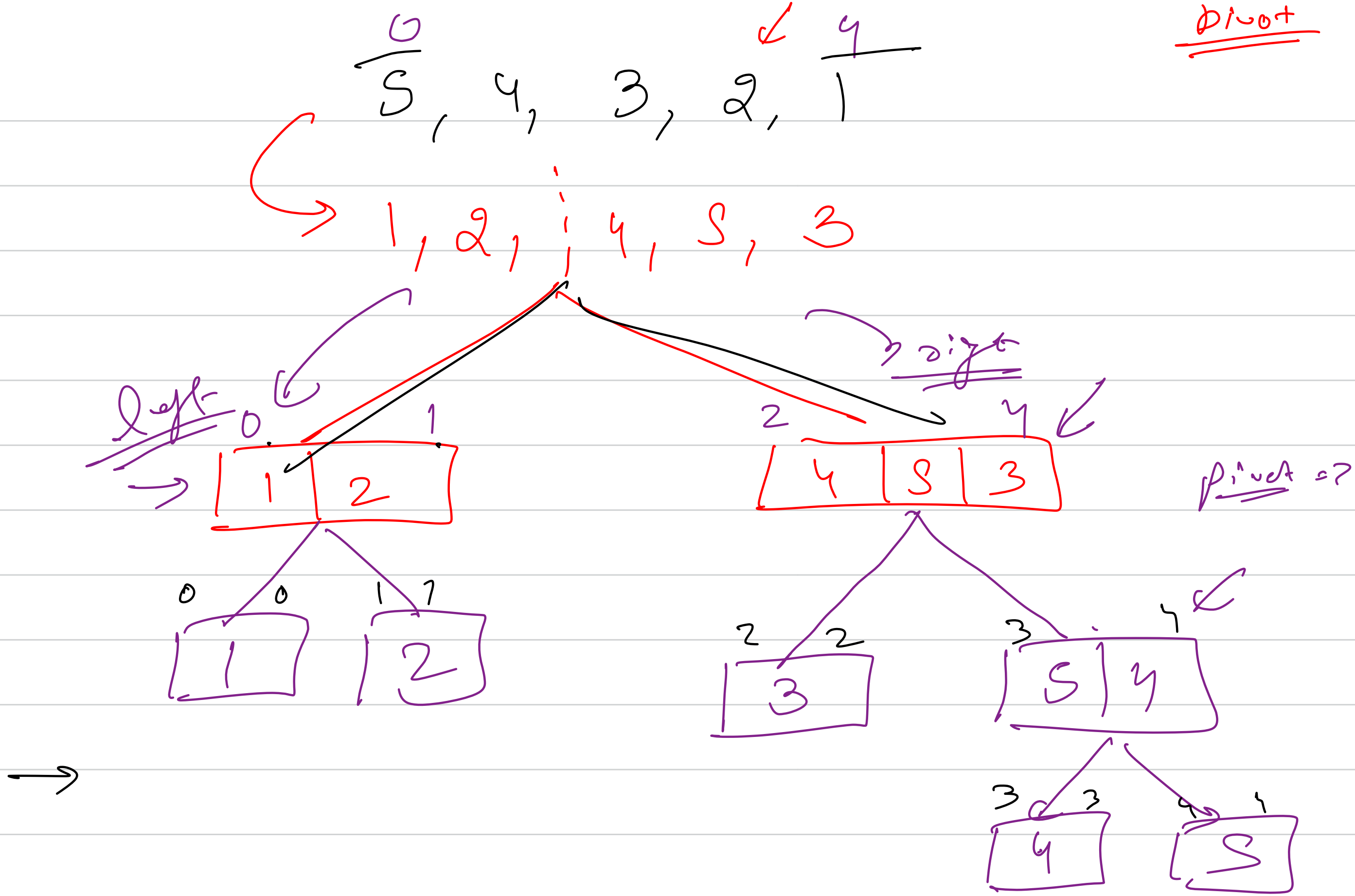
→ pick any random pivot

→ partition the list based on the pivot

→ recursively apply the same above logic

→ left part

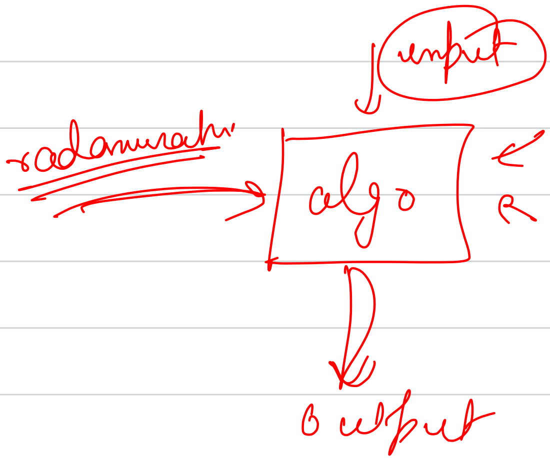
→ right part


pivot
5, 4, 3, 2, 1
1, 2, 4, 5, 3
left
right
pivot =?
1 | 2
4 | 5 | 3
1
2
3
5 | 4
4
5

Q- How to select pivot?

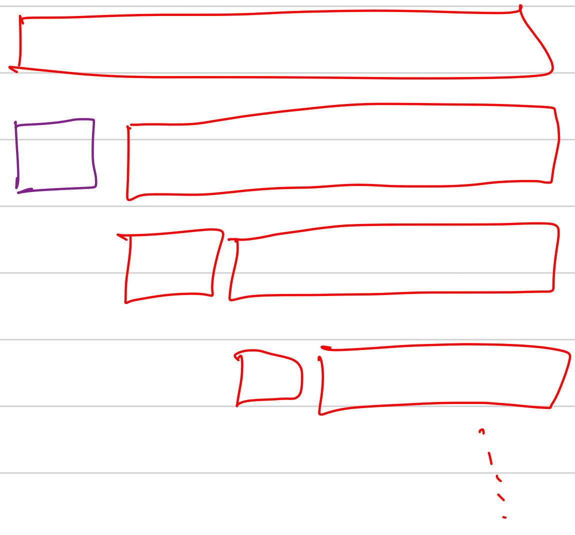
We will ans it later

Quicksort → Randomized algorithm



we have to choose pivot

1, 10000, 1000, 10



mid → element at mid index is the smallest element

→ $O(n^2)$

Segregate based on median

closest to mean

We should have the element of randomness

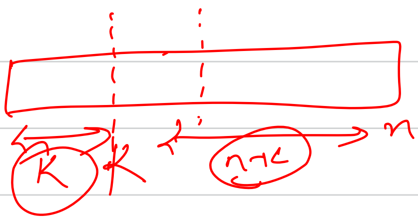
↳ choose pivot randomly.

↳ $T(n)$ → no. of operations required to apply quicksort on list of size n

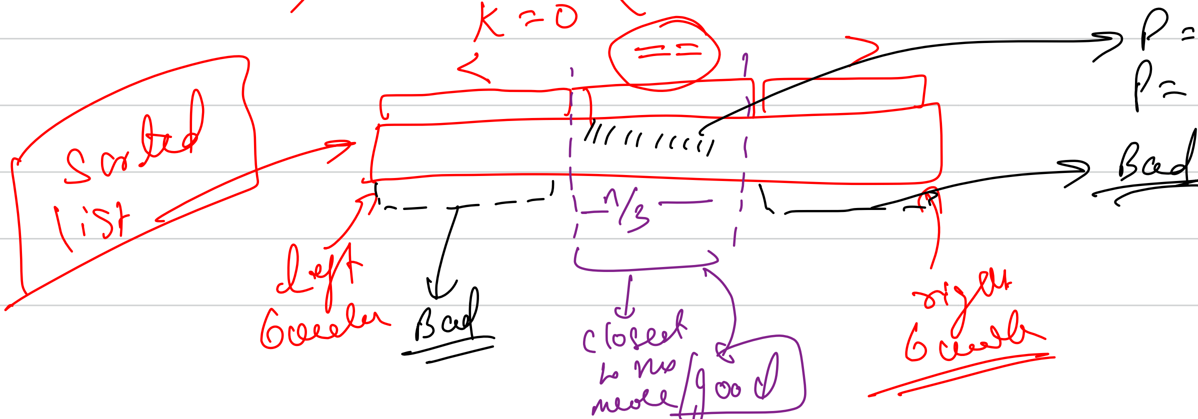
We will get the estimation of $T(n)$

$$\alpha(T(n)) = \underbrace{\sum \underbrace{X}_{\text{value } x} \cdot \underbrace{P(T(n) = x)}_{\text{probability } T(n)=x}}_{\text{general eq}^n}$$

(Maths is req'd)



$$\hookrightarrow \alpha(T(n)) = \sum_{k=0}^{n-1} \left(\alpha(T(k)) + \alpha(T(n-k)) + n \right)$$



$P = 1/3 \leftarrow$ good

$P = 2/3 \leftarrow$ Bad

n

$$\alpha(T(n)) = \sum_{k=0}^{n-1} \left(\alpha(T(k)) + \alpha(T(n-k)) + n \right) <=$$

$$n + \underbrace{\frac{1}{3}}_{\text{probability of good split}} \times \underbrace{\left(T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) \right)}_{\text{good split}} + \underbrace{\frac{2}{3}}_{\text{probability of bad split}} T(n)$$

$$T(n) \leq n + \frac{1}{3}\left(T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right)\right) + \frac{2}{3} T(n)$$

$$\frac{1}{3}T(n) \le n + \frac{1}{3}\left(T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right)\right)$$

$$T(n) \le 3n + T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right)$$

find the tightest upper bound $\quad T(n) \le c \times n \log n$

$$T(n) \le 3n + c \times \frac{n}{3} \log \frac{n}{3} + c \times \frac{2n}{3} \log \frac{2n}{3}$$

$$T(n) \leq 3n + C \times \frac{n}{3} \log \frac{n}{3} + C \times \frac{2n}{3} \log \frac{2n}{3}$$

$$\leq n\left(3 + \frac{C}{3}\log n + \frac{2C}{3}\log n - \frac{C}{3}\log 3 - \frac{C}{3}\log\frac{3}{2}\right)$$

$$\leq n\left(3 + C\log n - \frac{C}{3}\log 3 - \frac{C}{3}\log\frac{3}{2}\right)$$

$$T(n) \leq 3n + Cn\log n - \frac{cn}{3}\log 3 - \frac{cn}{3}\log\frac{3}{2}$$

$$T(n) \leq \Theta(n\log n)$$

$$O(n^2)$$

middle

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + O(n)$$

$$O(n \log n)$$

→ TC → $\Theta(n\log n)$; $\Omega(n\log n)$; $O(n^2)$

SC → $O(\log n)$

Inplace → Yes

Stable → No

pseudorandom

seed

start value

current time

Heapsort

Radix sort

Bucket

→ Tim Sort → sorted()

sort()

→ Java → Arrays.sort()

insertion sort + merge sort

→ $O(n \log n)$