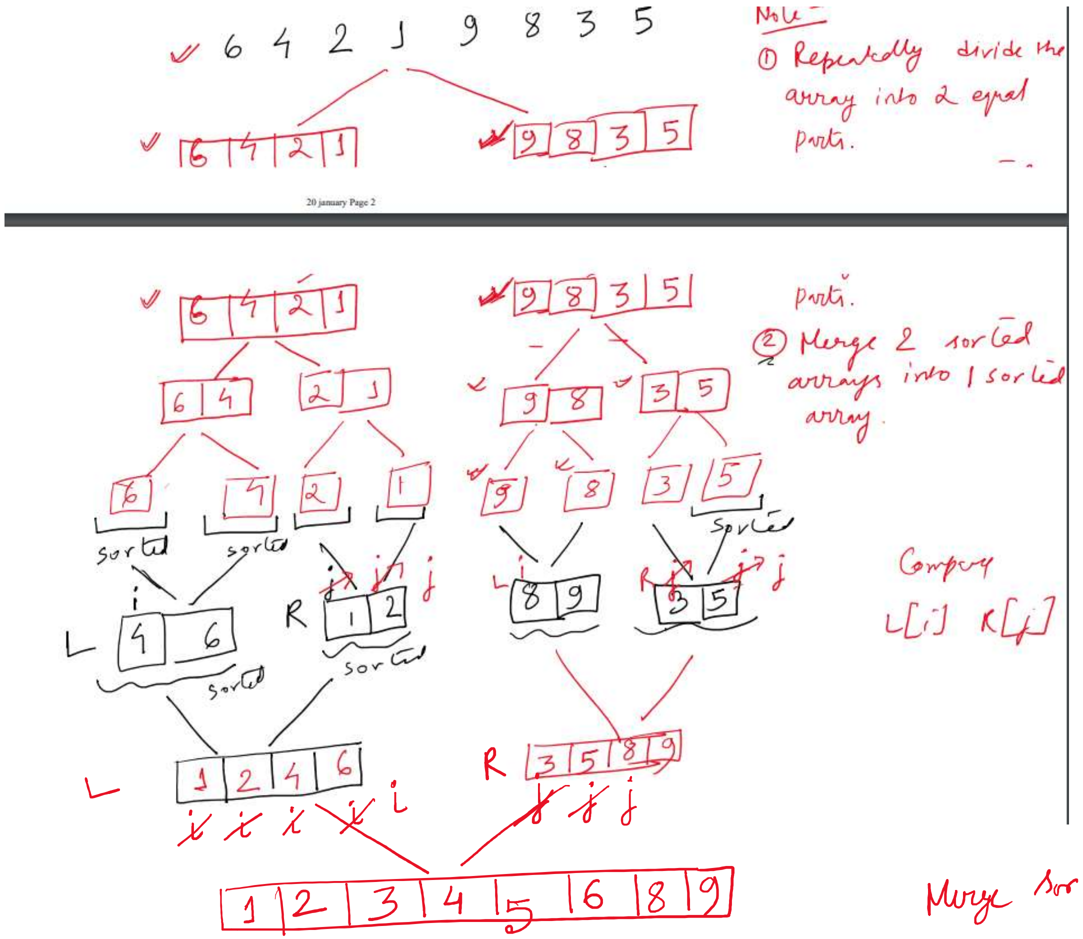
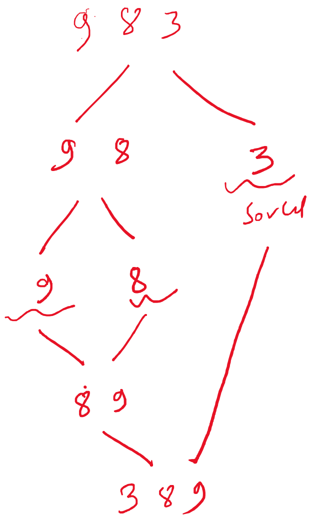


Note -

① Repeatedly divide the array into 2 equal parts.

② Merge 2 sorted arrays into 1 sorted array.

Compare L[i] R[j]



Merge Sort is Recursive.

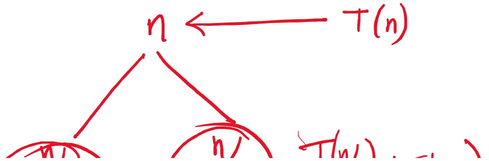
Recursive Code → Recurrence Relation —Solve→ Time complexity

TC - T(n)

Suppose - Merge Sort ka code T(n)

$$\boxed{T(n) = T(n/2) + T(n/2) + n}$$

← Merge Karne ka

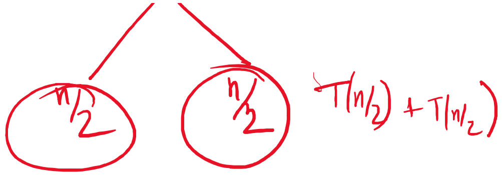


$$\boxed{T(n) = T(n/2) + T(n/2) + n}$$

Actual problem

2 subproblems

Merge karne ka Time



Master Thm

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

$a=2$, $b=2$, $k=1$, $p=0$

$a \quad b^k$ $\qquad k=1$

$2 = 2^1$

2nd condn.

$p=0$

$$\textcircled{2a} \rightarrow n^{\log_b a} . \log^{p+1} n = n^{\boxed{\log_2 2}} . \log^{0+1} n$$

$$= n . \log n$$

Quick Sort - Divide & Conquer Technique.

35 50 15 25 80 20 90 45.

pivot =35

p

$\cancel{q}$ $\cancel{q}$

$50 > 35$

Note -

① pivot element.

② pivot element = 1st element.

③ $p \rightarrow arr[p] <= pivot.$

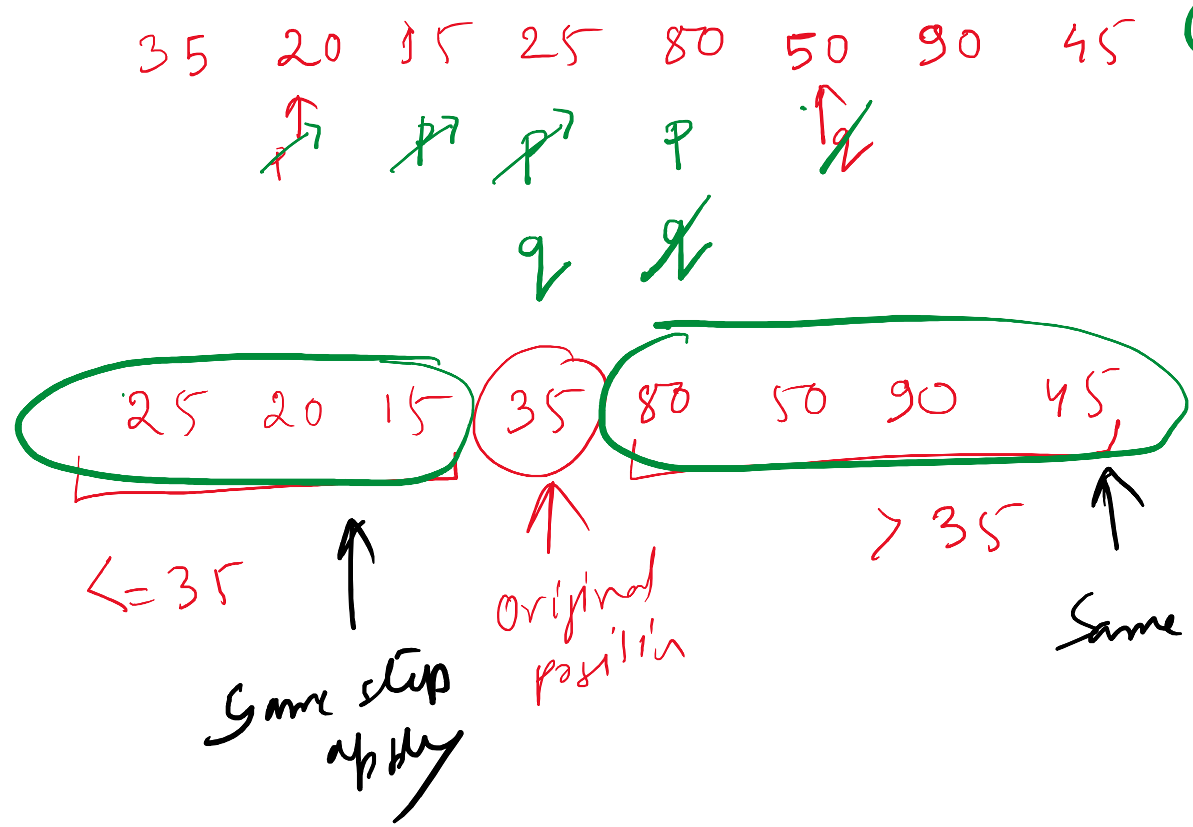
$q \rightarrow arr[q] > pivot.$

= 50 > 35

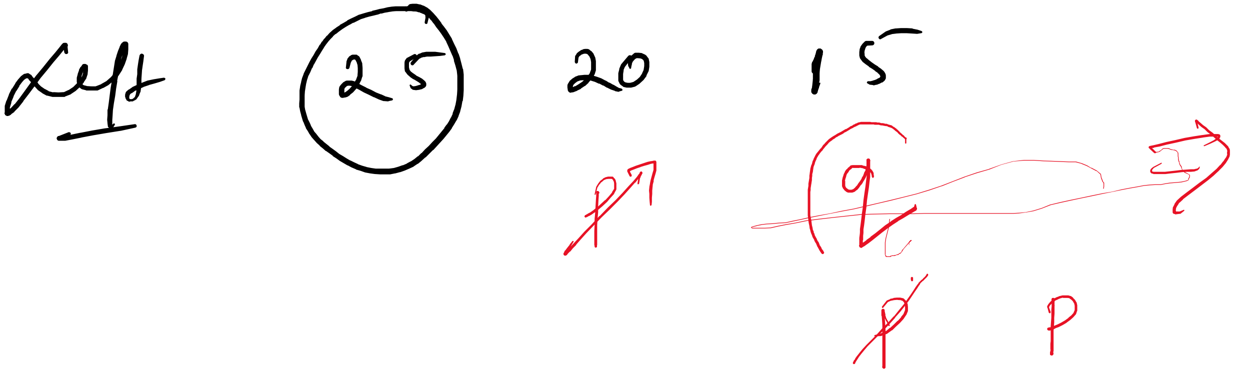
q → arr[q] > pivot.

(4) If p&q havenot crossed each other, swap them

(5) If p&q crossed each other, = swap (pivot, arr[q]);



pivot = 25



swap (arr(q), pivot)

15 20 25

80 50 90 45 pivot=80

~~p~~ p q

swap(arr(p), arr(q))

80 50 45 90

~~p~~ q ~~q~~

swap(pivot, arr[q])

p

45 50 80 90

<=pivot original pos >pivot