$T(n) = 4T(n/2) + n^2$ $\quad aT(n/b) + \Theta(n^k \log^p n)$

$a = 4,\ b = 2,\ k = 2,\ p = 0$ $\quad \log_2 4 = \log_2 2^2 = 2\log_2 2 = 2$

$a > b^k$

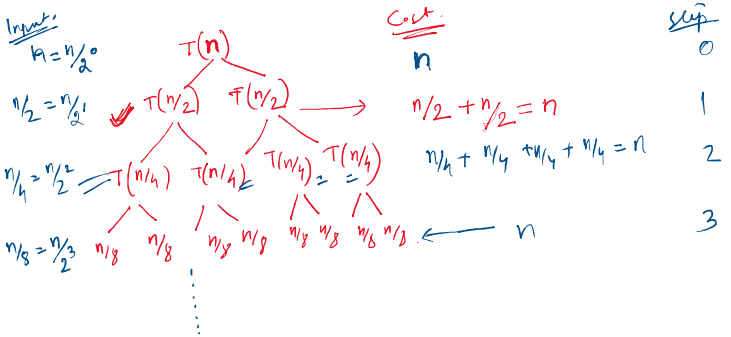
$4 = 2^2$ Cond^n-2nd (a) $\rightarrow$ $T(n) = O(n^{\log_b a} \log^{p+1} n)$

$= O(n^{\log_2 4} \cdot \log^{0+1} n)$

$= O(n^2 \cdot \log n)$

Recursion tree method –

$$T(n) = \begin{cases} 2T(n/2) + n, & n > 1 \\ 1, & n = 1 \end{cases}$$



Total cost =

Steps (0 to k) $\equiv (k+1)$ steps

$n \times (k+1)$

$= nk + n$

$= (n \log_2 n + n)$

$= n \log_2 n.$

Recursion Tree from question.

Recursion stops when input size $\Rightarrow \left(\frac{n}{2^k}\right) = 1$

$\log_a a = 1$

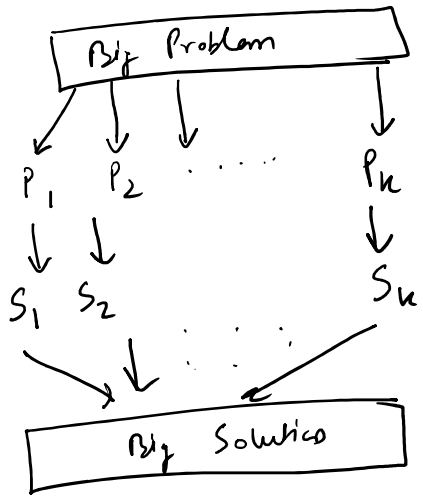
$\Rightarrow n = 2^k$

$\Rightarrow \log_2 n = \log_2 2^k = k \log_2 2 = k$

$\Rightarrow k = \log_2 n$

H/W: $T(n) = \begin{cases} 2T(n/2) + c, & n > 1 \\ 1, & n = 1 \end{cases}$ Using Recursion Tree.

Divide & Conquer method. —— Technique to design algorithm.



(Small sub-problems)

(Solution of small sub-problems)

Eg

1 + 2 + 3 + 4

= 10

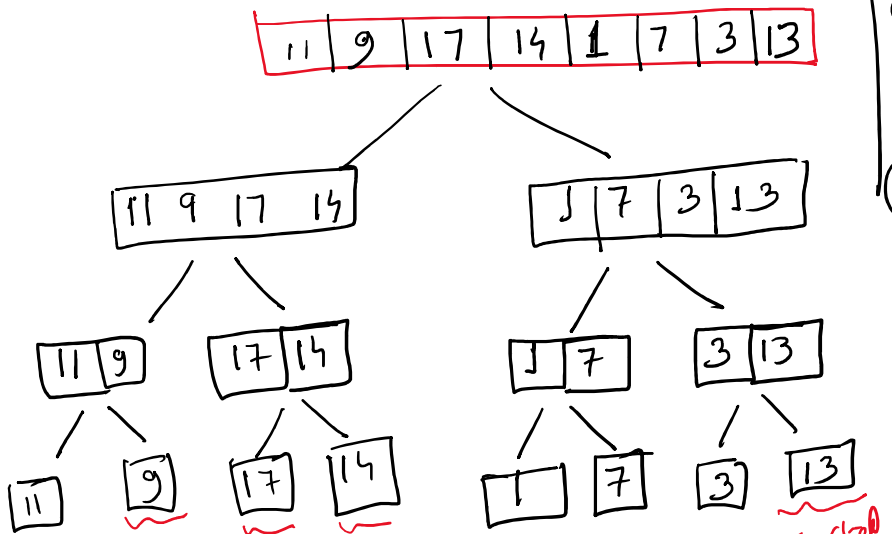
(1+2) + (3+4)

P1 P2

S1 = 3 S2 = 7

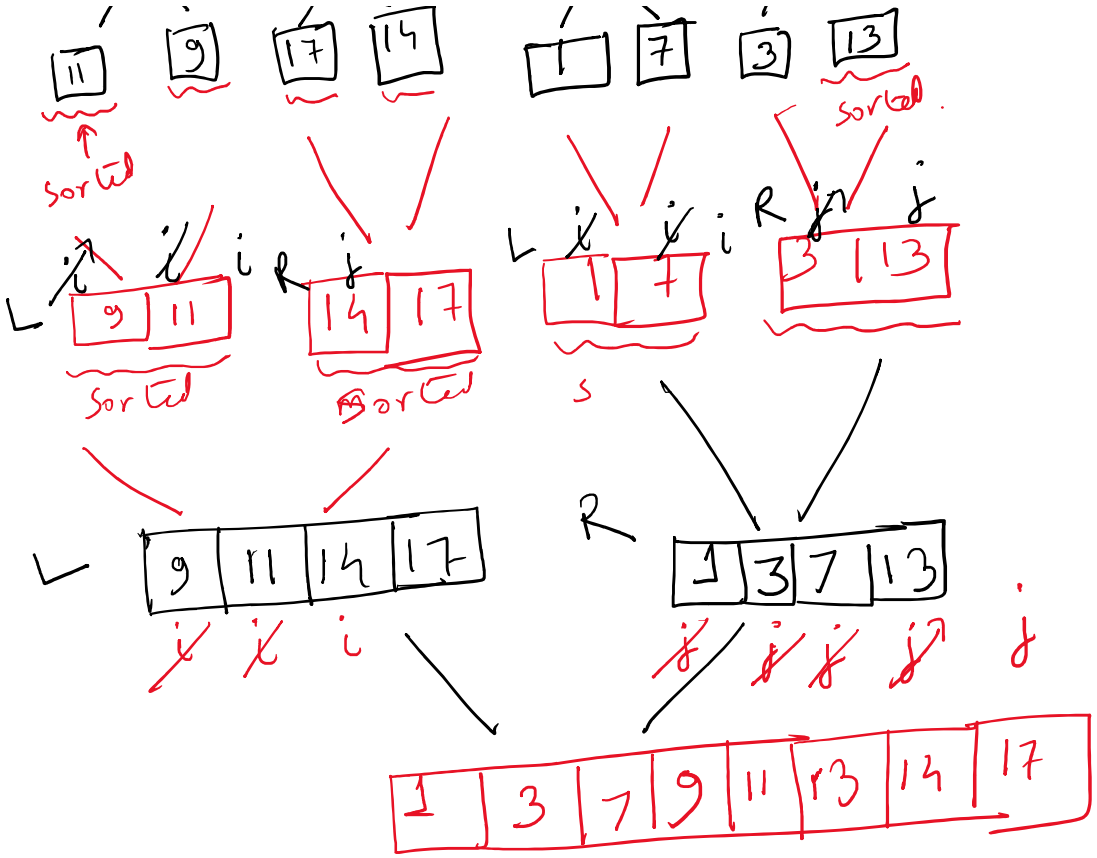
10

Merge Sort - Based on divide & conquer method.



Note -

1. Repeatedly divide the array into 2 equal parts.
2. Merge 2 sorted arrays into 1 sorted array.

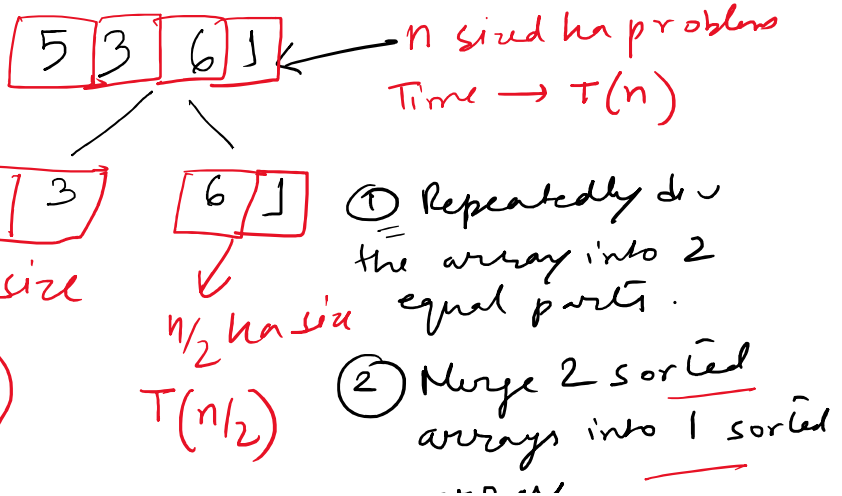


Compare
L[i] R[j]

Time Complexity

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

Sub-problems solve karne ka time | merge



① Repeatedly divide the array into 2 equal parts.

② Merge 2 sorted arrays into 1 sorted

ka time | $T(n/2)$ $T(n/2)$ ⌣ arrays into 1 sorted array.

$T(n) = 2T(n/2) + n$ $\quad aT(n/b)$ $\quad (n^k \log^p n)$

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

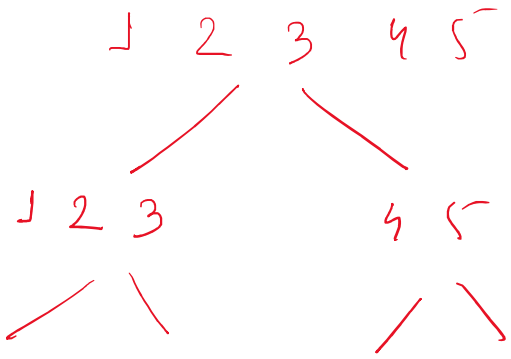
$a \quad b^k$

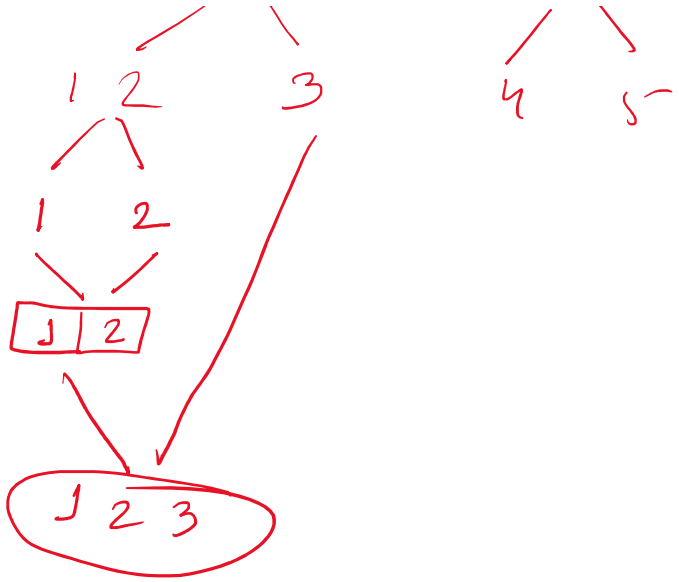
$2 = 2^1$

Cond^n 2a $\rightarrow$ $T(n) = O\left(n^{\log_b a} \cdot \log^{p+1} n\right) = O\left(n^{\log_2 2} \cdot \log^{0+1} n\right)$

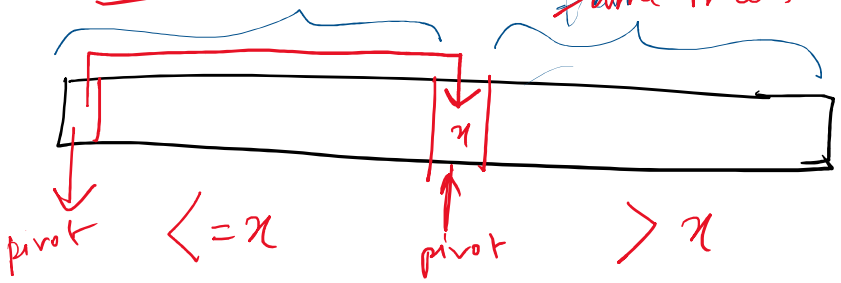
$= O(n \log^1 n)$

SC $= O(n)$ //





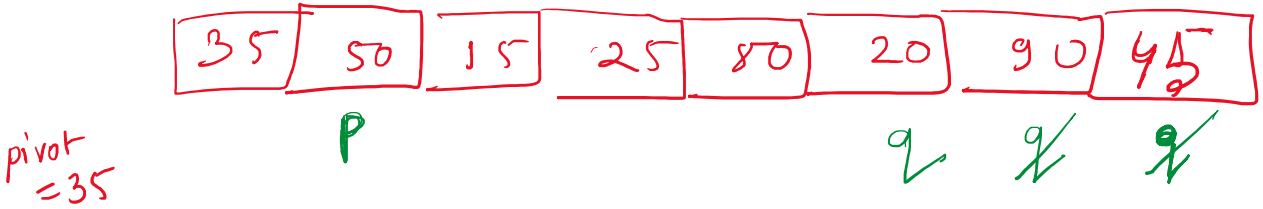
Quick Sort-



Note- pivot element

↓

Aapko choose karna hain



35 20 15 25 80 50 90 45

p p p p q

q q

Note -

① p →

arr[p] <= pivot.

② q ←

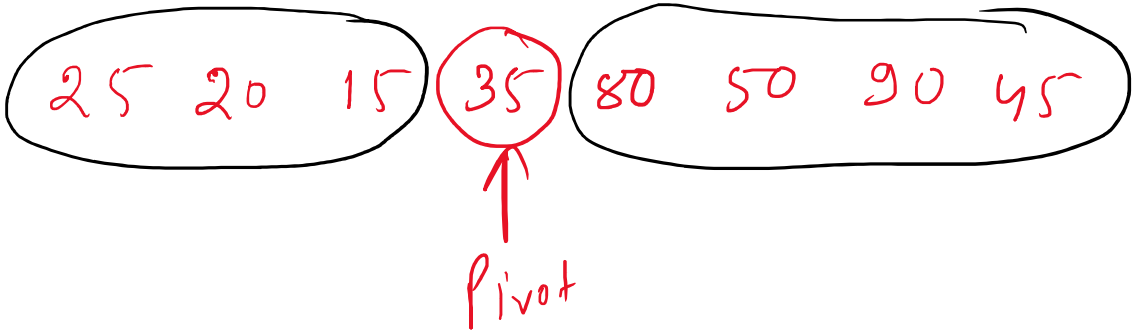
arr[q] > pivot

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

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

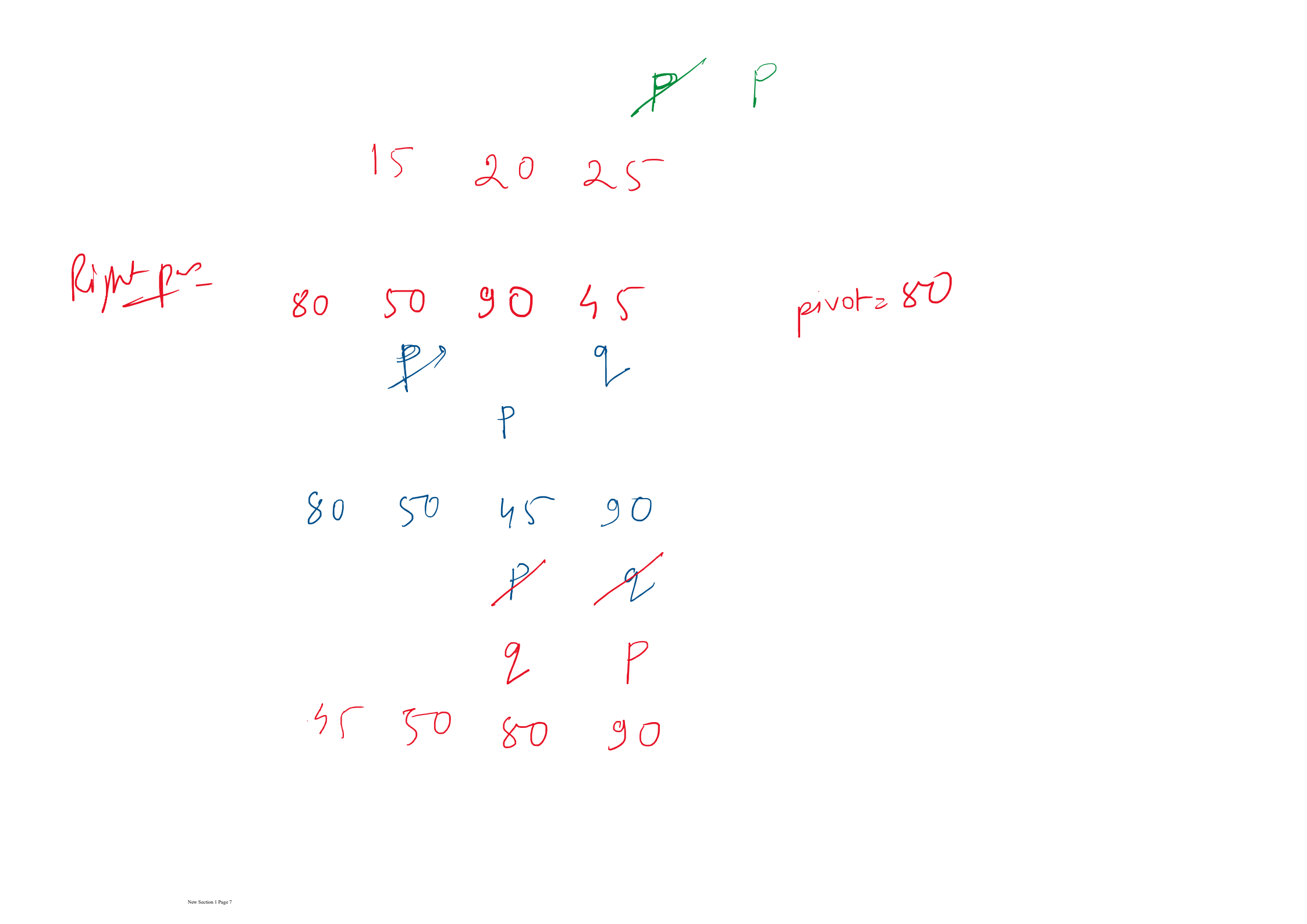
④ If p and q crossed each other

swap (arr(q), pivot);



Lft array - 25 20 15 pivot = 25

p q


15 20 25
Right pt
80 50 90 45
pivot = 80
80 50 45 90
45 50 80 90