Quick Sort:

Arr[] = {5, 4, 3, 2, 1}

There is a concept of 'pivot' in Quick Sort which acts as a reference point.

* pivot:

Choose any element as a reference. → after first pass all the elements '<' (less than pivot) will be on the left hand side of pivot and the elements greater than pivot will be on the right side.

(*) Recursive Quick Sort:

arr = {5, 4, 3, 2, 1}

p = 4 (pivot)

↓

[1, 3, 2, 4, 5] → 1st pass may or may not be sorted

→ After every pass we put pivot at the correct position.

Recursion will call the LHS and RHS of pivot.

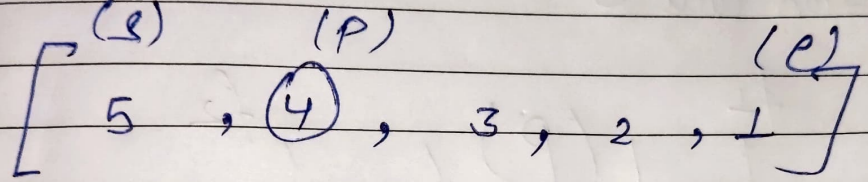
1, 3, 2 ↙ ↘ [5] → sorted.

[1, 2, 3] ↳ pivot = 3

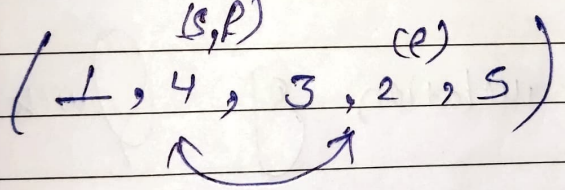
In merge sort, even when the array get sorted the recursive call checks until the base condition but in QuickSort, that doesn't happen

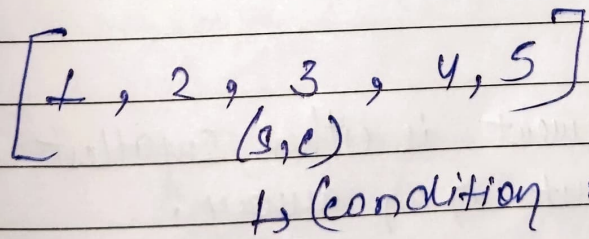
(*) Here, we will have void return type as we are not creating new array, we are sorting the existing array.

(*) How to put pivot :



we make a condition for the pivot and if any elements violates it, we changes its position.





(*) Checking Violation :

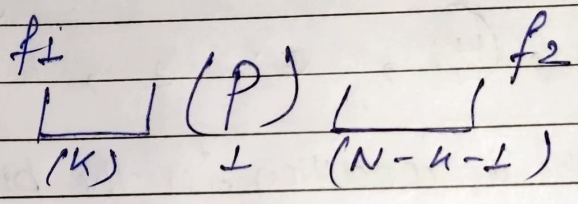
→ while n[s] < p:
// keep moving foward.
s++;

→ while n[e] > p:
e--;

(*) The idea is to put pivot (p) in its correct position.

→ Pick a random element as pivot. Or,
→ Corner elements. Or
→ Pick the middle elements.

(*) Recursive relation of this approach is that we are dividing the approach with two function calls



Total elements = N

→ $T(N) = T(K) + T(N-K-1) + O(N)$ ← for pivot

↳ Recurrence relation of Quick Sort.

→ Worst Case:

If the pivot element is either smallest or largest element in an array.

Eg: [3, 5, 8, 9, 20, 34, 18, 66]

↳ N-1 ; ↳ (P)

when (k = 0).

$T(N) = T(0) + T(N-1) + O(N)$

$= O(N^2)$

→ Best Case :

Pivot is in the middle of element

$k = N/2$

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

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

$$= O(N \log N)$$ [Through Akra - Bazzi formula]

(*) Notes :

→ Not Stable

→ Merge Sort is better in linked list due to memory allocations. → not continous.

→ MS takes $O(N)$ extras space.

Hybrid Sorting Algos : (Tim Sort)

↳ Intrnal algorithms uses Merge Sort + Insertion Sort

↳ Works well with partially sorted data.