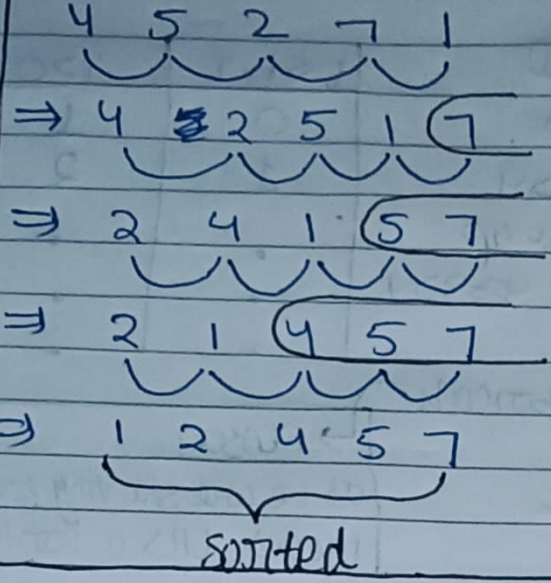
BUBBLE SORT*

best case = $\Omega(N)$

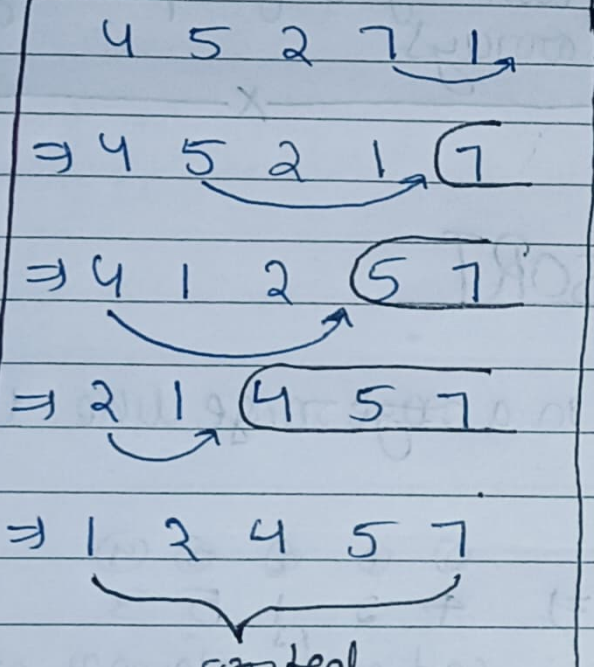


- Time complexity ⇒ $O(N^2)$
- stable algorithm ✓

⇓

means if eq ⇒ ① ③ ② ⑤ ③
after sorting ⇒ ② ① ③ ③ ⑤
(means black ball was after blue ball and after sorting order remains same ∴ stability)

SELECTION SORT**



- Time complexity ⇒ $O(n^2)$
- Best case ⇒ ~~$O(N^2)$~~ $O(N^2)$ (same)
- stable algorithm ✗
- performs well on small arrays.

* if arr[i] < arr[i-1] ⇒ swap ....

** put largest element at last index, put second largest element at second last index ....

INSERTION SORT

5 3 4 1 2 → is ① > ① → swap
⇒ 3 5 4 1 2 → is 5 > 4 → swap
⇒ 3 4 5 1 2 → now check, is 3 > 4 → no ⇒ loop break

i <= n-2	j > 0
0	1
1	2
.	.
.	.
.	.

AND SO ON ...

→ REASON
as we are starting from 0, means LHS is sorted

✡ Time complexity ⇒ $O(N^2)$ (best case = $O(N)$)

✡ advantage ⇒ stable, works good for partially sorted arrays

CYCLE SORT

⚠ when given numbers in a ~~range~~ range like 1→N apply cyclic sort.

EXAMPLE ⇒ range 1→5 ⇒

⓪	①	②	③	④
4	2	1	5	3

(at every element at correct index which is value - 1 in this case)

∴ as 4 is at i=0, swap with i=3 (the correct position for 4)

...

✡ Time complexity = O(N)