Space complexity

Auxilary space - extra space or temporary space used for algorithm

Space complexity = Total space taken by algorithm w.r.t the input size - include both auxilary space & input space used.

eg - Merge sort = O(n) ~~[illegible]~~ auxilary space
Insertion sort = O(1)
& heap sort

Que
```
for (i=1 ; i≤N;) {
    for (j=1 ; j≤k ; j++) {
        // some operation
           that take time t
    } i = i+k
}
```
time complexity = inner loop O(kt)

↳ O(kt * no of time outer loop run)

$i = 1,\ 1+k,\ 1+2k + \ 1+3k \ \dots \ 1+\underbrace{xk}$

$1 + x k \le N$

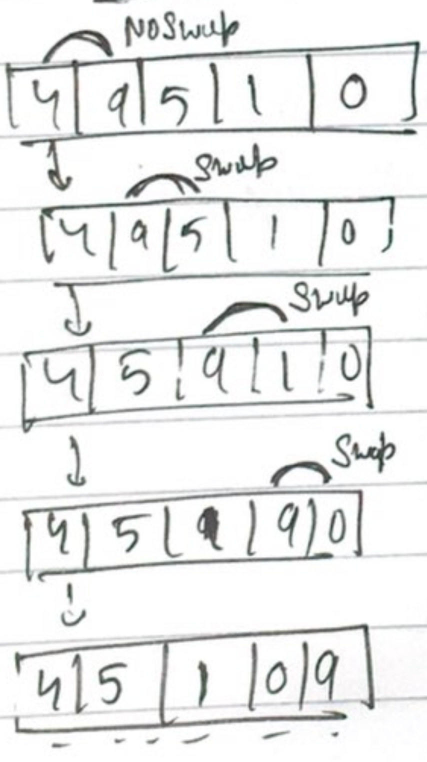
$xk \le N-1$

no of time $\leftarrow x \le \dfrac{N-1}{k}$
outer loop runs

Complexity = $O(k + x \left(\dfrac{~~N~~-1}{k}\right))$

$O(N+)$

Bubble sort



- Worst case & average case Time complexity = $O(N*N)$ when array is reverse sorted
- Best case = $O(N)$ when already sorted
- Auxilary Space = $O(1)$
- Boundary cases = take min time when elements already sorted
- Sorting in place = Yes
- Stable - Yes

Selection Sort

Worst Complexity = $O(n^2)$
Average Complexity = $O(N^2)$
Best case = [illegible] $O(N^2)$
Space Complexity = $O(1)$
Method = Selection
Stable - NO

• it never makes more than $O(n)$ swap and can be useful when memory write is costly

Insertion sort

Time Complexity = $O(n*2)$
Auxiliary Space = $O(1)$
Boundary case = max time when array is reverse sorted, min time when already sorted
Sorting in place = Yes