

## Insertion Sort

```
for (int i = 1; i < size; ++i) {
    int temp = arr[i]
    int j = i - 1
    while (j >= 0 && arr[j] > temp) {
        arr[j+1] = arr[j]
        j--
    }
    arr[j+1] = temp;
}
```

Time complexity

$$\begin{aligned}
 & n \\
 & 1 + (n-1) \\
 & 1 + (n-1) \\
 & \frac{n(n-1)}{2} \\
 & 1 + (n(n-1)/2) \\
 & \text{---} \\
 & 1 + (n-1)
 \end{aligned}$$

∴ Time complexity:

$$\begin{aligned}
 & n + (n-1) + (n-1) + \frac{n(n-1)}{2} + \frac{n(n-1)}{2} + \frac{n(n-1)}{2} + (n-1) \\
 & = \frac{3n^2}{2} + \frac{5n}{2} - 3
 \end{aligned}$$

∴  $O(n^2)$  is the time complexity!

Space complexity

(at time and for storing the array)

$$\begin{aligned}
 & 1 + 1 + 1 + 1 + 1 + 1 + n \\
 & = 6 + n
 \end{aligned}$$

∴  $O(n)$  is the space complexity!

Best case  $O(n)$  array is already sorted  
 $O(n^2)$  is worst case

*Sam*  
20/01

Selection Sort:

```
for (int i = 0; i < size - 1; ++i) {
    int minI = i;
    for (int j = i + 1; j < size; ++j) {
        if (arr[j] < arr[minI]) {
            minI = j;
        }
    }
    swap(arr[i], arr[minI]);
}
```

Time complexity

$$n$$

$$1(n-1)$$

$$\frac{n(n-1)}{2}$$

$$\frac{n(n-1)}{2}$$

$$\frac{n(n-1)}{2}$$

$$3(n-1)$$

Time Complexity

$$n(n-1) + n + (2(n-1))$$

$$\cancel{n + (n-1) + \frac{n(n-1)}{2} + \frac{n(n-1)}{2} + n(n-1) + 3(n-1)}$$

$$= \frac{3n^2}{2} + \frac{7n}{2} + \frac{11}{2} \quad n^2 - n \approx n^2$$

$O(n^2)$  is the time complexity!

Space Complexity:

for line & for storing the array

$$1 + 1 + 1 + 3 + n$$

$$\therefore 7 + n$$

$\therefore O(n)$  is space complexity!

Best case :-

$O(n^2)$  array is already sorted but only no. of swaps complexity which doesn't affect the overall complexity.

$O(n^2)$  is worst case

Done  
30/01