

## Insertion Sort

```
for (int i = 1; i < size; +i) {
```

Time complexity.

```
    int temp = arr[i]
```

$\cdot n$

```
    int j = i - 1
```

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

```
    while (j >= 0 & arr[j] > temp) {
```

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

```
        arr[j + 1] = arr[j]
```

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

```
    j = j - 1;
```

~~$\cdot \frac{1}{2}(n(n-1)/2)$~~

```
    arr[j + 1] = temp;
```

$\cdot 1(n-1)$

}

∴ Time complexity:

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

$$= \frac{3n^2}{2} + \frac{5n}{2} - 3$$

∴  $O(n^2)$  is the time complexity!

Space complexity

(at time and for storing the array)

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

$$= 6 + n$$

∴  $O(n)$  is the space complexity!

Best case:  $O(n)$  array is already sorted

$O(n^2)$  is worst case

Drawn  
20/10/20

### Selection Sort:

```

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

```

Time complexity

$n$

$1(n-1)$

~~$n(n-1)$~~

~~$n(n-1)$~~

~~$n(n-1)$~~

$3(n-1)$

### Time Complexity

$$n(n-1) + n + 3(n-1)$$

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

$$= \frac{5n^2}{2} + \frac{7n}{2} + \frac{11}{2}$$

$$n^2 - n \approx n^2$$

$O(n^2)$  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:-

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

$O(n)$  is worst case

Final  
30/01