The time complexity of an algorithm estimates how much time the algorithm will use for some input.

Loops

```
for (int i=0 ; i<=n; i++) {
}
```
} $O(n)$

```
for(int i=0; i<=n ; i++) {
    for (int j=0; j<=n; j++) {
    }
}
```
} $O(n^2)$

```
for(int i=0; i<=n ; i++){
  for(int j=0; j<=m ; j++) {
  }
}
```
} $O(nm)$

$$O(1) \leq O(\log n) \leq O(\sqrt{n}) \leq O(n) \leq O(n \log n) \leq O(n^2)$$
$$\leq O(n^3) \leq O(2^n) \leq O(n!)$$

Maximum Subarray

1. Three Loop Approach $O(n^3)$
2. Two Loop Approach $O(n^2)$
3. Single Traversal $O(n)$

The single traversal method is also known as Kadane's Algorithm

So a same problem can be solved in different ways with different complexity