Analysis of Algorithm

Amypo Technologies Pvt Ltd

Concepts:

- Sorting
- Bubble Sort
- Insertion Sort
- Selection Sort

Sorting Algorithm:

Sorting is the process of arranging the elements of an array so that they can be placed either in ascending or descending order.

The different types of sorting algorithms are

- Bubble Sort
- Insertion Sort
- Selection Sort
- Quick Sort

Bubble Sort Algorithm :

- Bubble sort works on the repeatedly swapping of adjacent elements until they are not in the intended order.
- It is called bubble sort because the movement of array elements is just like the movement of air bubbles in the water.
- Bubbles in water rise up to the surface; similarly, the array elements in bubble sort move to the end in each iteration.
- The average and worst-case complexity of Bubble sort is $O(n^2)$, where n is a number of items.

Bubble Sort Complexity:

The time complexity of bubble sort in the best case, average case, and worst case.

Time Complexity:

Case	Time Complexity
Best Case	$O(n)$
Average Case	$O(n^2)$
Worst Case	$O(n^2)$

- Best Case Complexity - It occurs when there is no sorting required, i.e. the array is already sorted. The best-case time complexity of bubble sort is O(n).
- Average Case Complexity - It occurs when the array elements are in jumbled order that is not properly ascending and not properly descending. The average case time complexity of bubble sort is $O(n^2)$.
- Worst Case Complexity - It occurs when the array elements are required to be sorted in reverse order.

Space Complexity:

Space Complexity	O(1)
Stable	YES

- The space complexity of bubble sort is O(1). It is because, in bubble sort, an extra variable is required for swapping.
- The space complexity of optimized bubble sort is O(2). It is because two extra variables are required in optimized bubble sort

Insertion Sort Algorithm:

- The idea behind the insertion sort is that first take one element, iterate it through the sorted array.
- Although it is simple to use, it is not appropriate for large data sets as the time complexity of insertion sort in the average case and worst case is $O(n^2)$, where n is the number of items.
- Insertion sort is less efficient than the other sorting algorithms like heap sort, quick sort, merge sort, etc.

Insertion sort has various advantages such as:

- Simple implementation
- Efficient for small data sets
- Adaptive, i.e., it is appropriate for data sets that are already substantially sorted.

Algorithm:

- Step 1 - If the element is the first element, assume that it is already sorted. Return 1.
- Step2 - Pick the next element, and store it separately in a key.
- Step3 - Now, compare the key with all elements in the sorted array.
- Step 4 - If the element in the sorted array is smaller than the current element, then move to the next element. Else, shift greater elements in the array towards the right.
- Step 5 - Insert the value.
- Step 6 - Repeat until the array is sorted.

Insertion sort complexity:

Best Case Complexity : It occurs when there is no sorting required, i.e. the array is already sorted. The best-case time complexity of insertion sort is O(n).

Average Case Complexity : It occurs when the array elements are in jumbled order that is not properly ascending and not properly descending. The average case time complexity of insertion sort is $O(n^2)$.

Worst Case Complexity : It occurs when the array elements are required to be sorted in reverse order. That means suppose you have to sort the array elements in ascending order, but its elements are in descending order. The worst-case time complexity of insertion sort is $O(n^2)$.

Space Complexity:

- Space Complexity - O(1)
- Stable - Yes

The space complexity of insertion sort is O(1). It is because, in insertion sort, an extra variable is required for swapping.

Selection Sort Algorithm:

- Selection sort finds the smallest element in the array and place it on the first place on the list, then it finds the second smallest element in the array and place it on the second place.
- This process continues until all the elements are moved to their correct ordering.
- It carries running time $O(n^2)$ which is worst than insertion sort.
- The average and worst-case complexity of selection sort is $O(n^2)$, where n is the number of items. Due to this, it is not suitable for large data sets.

Selection sort complexity:

Time Complexity:

- **Best Case Complexity** - It occurs when there is no sorting required, i.e. the array is already sorted. The best-case time complexity of selection sort is $O(n^2)$.

- **Average Case Complexity** - It occurs when the array elements are in jumbled order that is not properly ascending and not properly descending. The average case time complexity of selection sort is $O(n^2)$.

- **Worst Case Complexity** - It occurs when the array elements are required to be sorted in reverse order. That means suppose you have to sort the array elements in ascending order, but its elements are in descending order. The worst-case time complexity of selection sort is $O(n^2)$.

Space Complexity:

- **Space Complexity** - O(1)
- **Stable** - Yes
- The space complexity of selection sort is O(1). It is because, in selection sort, an extra variable is required for swapping.