1. **Types of Sorting:**
2. **Bubble Sort:** A straightforward comparison-based algorithm. It works by repeatedly comparing and swapping adjacent elements if they are in the wrong order until the list is sorted.

Complexity: It has a time complexity of O(n^2), which makes it inefficient for large datasets. Its space complexity is O(1) since it sorts in place.

1. **Insertion Sort:** This algorithm builds the sorted array one item at a time by taking the next item and inserting it into its correct position.

Complexity: It has a time complexity of O(n^2) but performs at O(n) for nearly sorted data. Its space complexity is O(1), making it efficient for small or nearly sorted datasets.

1. **Quick Sort:** A divide-and-conquer algorithm that selects a pivot, partitions the array around the pivot, and recursively sorts the sub-arrays.

Complexity: It has an average time complexity of O(n log n) and a worst-case complexity of O(n^2), which is rare. Its space complexity is O(log n) due to the recursive stack. It is very efficient for large datasets.

1. **Merge Sort:** This algorithm divides the array into halves, sorts them, and merges the sorted halves.

Complexity: It has a consistent time complexity of O(n log n). Its space complexity is O(n) due to the need for a temporary array. It is stable and ideal for large datasets.

**Analysis:**

**Bubble Sort**

Time Complexity:

- Worst Case: O(n^2)

- Average Case: O(n^2)

- Best Case: O(n) (when the array is already sorted)

Space Complexity: O(1) (in-place sorting)

**Quick Sort**

Time Complexity:

- Worst Case: O(n^2) (when the smallest or largest element is always chosen as the pivot)

- Average Case: O(n log n)

- Best Case: O(n log n)

Space Complexity: O(log n) (due to the recursive stack)

Quick Sort is generally preferred over Bubble Sort due to its significantly better average-case time complexity of O(n log n) compared to Bubble Sort's O(n^2). Quick Sort efficiently handles large datasets and is adaptable with optimizations like better pivot selection. Despite its worst-case scenario of O(n^2), this is rare with good pivot choices. Bubble Sort, on the other hand, performs poorly for larger datasets and offers limited optimization potential. Its inherent design of comparing and swapping adjacent elements makes it impractical for anything beyond small or nearly sorted arrays.