### **EXERCISE 3 – SORTING CUSTOMER ORDERS**

### 

### **Step 1: Understand Sorting Algorithms**

**Bubble Sort**:

* **Description**: Bubble Sort repeatedly steps through the list, compares adjacent elements, and swaps them if they are in the wrong order. This process continues until no swaps are needed.
* **Time Complexity**: O(n^2) in the worst and average cases.

**Insertion Sort**:

* **Description**: Insertion Sort builds the sorted array one item at a time, by repeatedly taking the next element and inserting it into the correct position in the already-sorted part of the array.
* **Time Complexity**: O(n^2) in the worst and average cases.

**Quick Sort**:

* **Description**: Quick Sort is a divide-and-conquer algorithm that picks a pivot element and partitions the array into two sub-arrays: elements less than the pivot and elements greater than the pivot. It then recursively sorts the sub-arrays.
* **Time Complexity**: O(n log n) on average, O(n^2) in the worst case.

**Merge Sort**:

* **Description**: Merge Sort is a divide-and-conquer algorithm that divides the array into two halves, recursively sorts them, and then merges the sorted halves.
* **Time Complexity**: O(n log n) in all cases.

**Step 4: Analysis**

**Performance Comparison:**

* **Bubble Sort**:
  + **Time Complexity**: O(n^2) in the worst and average cases.
  + **Explanation**: Bubble Sort repeatedly compares and swaps adjacent elements, which leads to a quadratic number of comparisons and swaps as the array size increases.
* **Quick Sort**:
  + **Time Complexity**: O(n log n) on average, O(n^2) in the worst case.
  + **Explanation**: Quick Sort divides the array into smaller sub-arrays and sorts them recursively. Although the worst-case time complexity is O(n^2), it can be mitigated by choosing a good pivot. On average, Quick Sort performs much better due to the divide-and-conquer approach.

**Why Quick Sort is Generally Preferred Over Bubble Sort:**

Quick Sort is generally preferred over Bubble Sort because it has a much better average time complexity of O(n log n), making it more efficient for large datasets. Bubble Sort's O(n^2) time complexity makes it impractical for large datasets due to the excessive number of comparisons and swaps required. Quick Sort's divide-and-conquer strategy allows it to handle larger arrays more efficiently, making it a preferred choice for sorting in most scenarios.