**Exercise 3: Sorting Customer Orders**

**1. Understanding Sorting Algorithms**

1. **Bubble Sort**:

Bubble Sort repeatedly compares adjacent pairs of elements and swaps them if they are in the wrong order. This process is repeated until the array is sorted.

1. **Insertion Sort**:

Insertion Sort builds the final sorted array one item at a time by picking elements and inserting them into their correct position.

1. **Quick Sort**:

Quick Sort divides the array into two sub-arrays based on a pivot element and recursively sorts the sub-arrays. The pivot element is placed in its correct position.

1. **Merge Sort**:

Merge Sort divides the array into two halves, recursively sorts them, and then merges the sorted halves.

**4. Analysis of Bubble Sort vs. Quick Sort**

1. **Time Complexity Comparison**:
   * **Bubble Sort**: has a time complexity of O(n^2) in both average and worst cases. This means that the execution time grows quadratically with the number of elements, making it impossible for large datasets.
   * **Quick Sort**: has a time complexity of O(n log n) on average, which is significantly better than Bubble Sort for larger datasets. However, its worst-case time complexity is O(n^2) if the pivot is poorly chosen, though this is mitigated in practice with good pivot selection strategies.
2. **Why Quick Sort is Preferred**:

Quick Sort is more efficient for larger datasets with its average time complexity of O(n log n), compared to Bubble Sort's O(n^2). It operates in-place with minimal additional storage, and generally performs better in practice due to fewer comparisons and swaps.