**Exercise 3: Sorting Customer Orders**

**1. Understand Sorting Algorithms**

**Bubble Sort**

* **Description**: Bubble Sort is a simple sorting algorithm that repeatedly steps through the list, compares adjacent elements and swaps them if they are in the wrong order.
* **Time Complexity**:
  + Best Case: O(n)
  + Average Case: O(n^2)
  + Worst Case: O(n^2)

**Insertion Sort**

* **Description**: Insertion Sort builds the final sorted array one item at a time. It is much less efficient on large lists than more advanced algorithms such as quicksort, heapsort, or merge sort.
* **Time Complexity**:
  + Best Case: O(n)
  + Average Case: O(n^2)
  + Worst Case: O(n^2)

**Quick Sort**

* **Description**: Quick Sort is a divide-and-conquer algorithm. It works by selecting a 'pivot' element from the array and partitioning the other elements into two sub-arrays, according to whether they are less than or greater than the pivot.
* **Time Complexity**:
  + Best Case: O(n log n)
  + Average Case: O(n log n)
  + Worst Case: O(n^2)

**Merge Sort**

* **Description**: Merge Sort is an efficient, stable, comparison-based, divide-and-conquer sorting algorithm. Most implementations produce a stable sort, meaning that the implementation preserves the input order of equal elements in the sorted output.
* **Time Complexity**:
  + Best Case: O(n log n)
  + Average Case: O(n log n)
  + Worst Case: O(n log n)

**4. Analysis**

**Time Complexity Comparison**

* **Bubble Sort**:
  + Best Case: O(n)
  + Average Case: O(n^2)
  + Worst Case: O(n^2)
* **Quick Sort**:
  + Best Case: O(n log n)
  + Average Case: O(n log n)
  + Worst Case: O(n^2) (rare, occurs when the smallest or largest element is always chosen as the pivot)

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

* **Efficiency**: Quick Sort is much more efficient for large datasets due to its average time complexity of O(n log n), compared to Bubble Sort’s O(n^2).
* **Practical Performance**: Quick Sort generally has better cache performance than Bubble Sort and other O(n log n) algorithms because of its in-place nature.
* **Adaptability**: Quick Sort can be implemented to be stable, and its partitioning logic can be adapted to various data structures.