**Exercise 3: Sorting Customer Orders**

**1.Understanding 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. The pass through the list is repeated until the list is sorted.
* **Time Complexity**: O(n^2) in the average and worst cases.

**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**: O(n^2) in the average and worst cases, but O(n) in the best case.

**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**: O(n log n) on average, O(n^2) in the worst case, but this is rare with good pivot selection.

**Merge Sort:**

* **Description**: Merge Sort is also a divide-and-conquer algorithm. It divides the unsorted list into n sublists, each containing one element, then repeatedly merges sublists to produce new sorted sublists until there is only one sublist remaining.
* **Time Complexity**: O(n log n) in all cases.

**4.Analysis**

**Performance Comparison:**

* **Bubble Sort**:
  + **Time Complexity**: O(n^2)
  + **Space Complexity**: O(1)
  + **Efficiency**: Poor for large datasets due to quadratic time complexity.
* **Quick Sort**:
  + **Time Complexity**: O(n log n) on average, O(n^2) in the worst case.
  + **Space Complexity**: O(log n) due to recursive stack space.
  + **Efficiency**: Generally efficient for large datasets due to its average-case performance.

**Quick sort**  is generally preferred over Bubble Sort because it has a much better average-case time complexity (O(n log n)) compared to Bubble Sort's O(n^2). Although Quick Sort has a worst-case time complexity of O(n^2), this is rare in practice with good pivot selection strategies (e.g., using the median of three, random pivot). Moreover, Quick Sort has better cache performance and can be more easily optimized for modern architectures.