**Exercise 3: Sorting Customer Orders**

**Scenario:**

You are tasked with sorting customer orders by their total price on an e-commerce platform. This helps in prioritizing high-value orders.

**Steps:**

1. **Understand Sorting Algorithms:**
   * Explain different sorting algorithms (Bubble Sort, Insertion Sort, Quick Sort, Merge Sort).

Sorting algorithms are essential for organizing data in a specified order, typically in numerical or lexicographical order.

1. Bubble Sort

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. This process is repeated until the list is sorted.

2. Insertion Sort

Insertion Sort builds the final sorted list one item at a time. It is much less efficient on large lists than more advanced algorithms such as quicksort, heapsort, or merge sort.

3. Quick Sort

Quick Sort is a highly efficient sorting algorithm and is based on the divide-and-conquer approach. 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.

4. Merge Sort

Merge Sort is an efficient, stable, and comparison-based sorting algorithm. It is also based on the divide-and-conquer approach.

1. **Analysis:**
   * Compare the performance (time complexity) of Bubble Sort and Quick Sort.

Worst-case: Both Bubble Sort and Quick Sort have a worst-case time complexity of O(n ^ 2) However, this worst case for Quick Sort is avoidable with good pivot selection.

Best-case: Quick Sort O(n \* log(n)) is significantly faster than Bubble Sort O(n) for large n.

Average-case: Quick Sort O(n \* log(n)) performs much better on average compared to Bubble Sort O(n ^ 2)

* + Discuss why Quick Sort is generally preferred over Bubble Sort.

Quick Sort is generally preferred over Bubble Sort due to its superior efficiency and performance. While Bubble Sort has a time complexity of O(n^2) in both average and worst cases, making it inefficient for large datasets, Quick Sort has an average-case complexity of O(n log n), which is significantly faster. Even though both can have a worst-case time complexity of O(n^2), Quick Sort can be optimized with good pivot selection to avoid this scenario. Consequently, Quick Sort's ability to handle large datasets efficiently makes it a more practical and widely used sorting algorithm compared to Bubble Sort.