**Bubble Sort**

* **Description**: A simple comparison-based algorithm that repeatedly steps through the list, compares adjacent elements, and swaps them if they are in the wrong order. It continues this process until the list is sorted.
* **Time Complexity**:
  + Best Case: O(n) (when the list is already sorted)
  + Average Case: O(n^2)
  + Worst Case: O(n^2)

**Insertion Sort**

* **Description**: Builds the sorted list one item at a time by repeatedly picking the next item and inserting it into the correct position among the previously sorted items.
* **Time Complexity**:
  + Best Case: O(n) (when the list is already sorted)
  + Average Case: O(n^2)
  + Worst Case: O(n^2)

**Quick Sort**

* **Description**: A divide-and-conquer algorithm that selects a 'pivot' element, partitions the array into elements less than the pivot and elements greater than the pivot, and recursively applies the same process to the subarrays.
* **Time Complexity**:
  + Best Case: O(n log n)
  + Average Case: O(n log n)
  + Worst Case: O(n^2) (when the pivot is the smallest or largest element)

**Merge Sort**

* **Description**: A divide-and-conquer algorithm that divides the list into halves, recursively sorts each half, and then merges the sorted halves.
* **Time Complexity**:
  + Best Case: O(n log n)
  + Average Case: O(n log n)
  + Worst Case: O(n log n)

4a.) **Time Complexity Comparison**

* **Bubble Sort**:
  + **Time Complexity**: O(n^2) in both average and worst cases.
  + **Space Complexity**: O(1) - No extra space is needed beyond the input array.
* **Quick Sort**:
  + **Time Complexity**: O(n log n) on average, but O(n^2) in the worst case (with poor pivot choices).
  + **Space Complexity**: O(log n) due to recursion stack space.

4b.) **Why Quick Sort is Preferred**

* **Efficiency**: Quick Sort generally performs better than Bubble Sort due to its O(n log n) average time complexity compared to O(n^2) for Bubble Sort.
* **Scalability**: Quick Sort is more scalable and performs well with large datasets. It also tends to be faster in practice due to better cache performance and fewer overall operations.