Sorting Algorithms

Exercise 1: Exploration of Bubble Sort

Objective: Explore and implement the Bubble Sort algorithm, analyze its performance, and enhance its efficiency.

Task 1: Implement the conventional Bubble Sort algorithm.

Task 2: Enhance the Bubble Sort algorithm. Propose and implement an optimization strategy.

Task 3: Analyze the time complexity for both the conventional and enhanced versions of Bubble Sort.

Additional Inquiry: Investigate the impact of diverse input datasets on the performance of Bubble Sort.

Exercise 2: Delving into Insertion Sort

Objective: Implement and optimize the Insertion Sort algorithm and evaluate its computational complexity.

Task 1: Implement the traditional Insertion Sort algorithm.

Task 2: Optimize your Insertion Sort implementation.

Task 3: Examine and report the time complexity for both the standard and optimized versions of Insertion Sort.

Additional Inquiry: Explore the scenarios where Insertion Sort might outperform more complex sorting algorithms.

Exercise 3: Analyzing Selection Sort

Objective: Gain a comprehensive understanding of the Selection Sort algorithm.

Task 1: Implement the fundamental Selection Sort algorithm.

Task 2: Advance the Selection Sort algorithm.

Task 3: Conduct a time complexity analysis for both the basic and advanced versions of Selection Sort.

Additional Inquiry: Investigate the behavior of Selection Sort on nearly sorted lists.

Exercise 4: Investigating Merge Sort (Fusion Sort)

Objective: Dissect and enhance the Merge Sort algorithm, and examine its computational efficiency.

Task 1: Implement the classic Merge Sort algorithm.

Task 2: Enhance your Merge Sort implementation.

Task 3: Analyze the time complexity for both the standard and improved versions of Merge Sort.

Additional Inquiry: Evaluate the performance of Merge Sort with large datasets.

Exercise 5: Quick Sort In-Depth

Objective: Delve into the Quick Sort algorithm, refine its implementation, and scrutinize its time complexity.

Task 1: Implement the basic Quick Sort algorithm.

Task 2: Refine the Quick Sort algorithm.

Task 3: Evaluate the time complexity for both the basic and enhanced Quick Sort implementations.

Additional Inquiry: Examine how the choice of pivot influences Quick Sort's efficiency.