**SR UNIVERSITY**

**ASSIGNMENT NO 12**

**NAME:VASANTA**

**HALL TICKET NO :2403A54127**

**BATCH NO:04(DS)**

**Task 1: Implementing Bubble Sort with AI Comments**

Prompt:

1. Write a basic Bubble Sort implementation in Python.

2. Once your code is complete, ask ChatGPT to:

* Add **inline comments** explaining the key parts of the logic (like comparisons, swapping, loop passes, and early termination).
* Provide a **time complexity analysis** (best, average, worst case).

CODE:

A computer screen shot of a computer screen

AI-generated content may be incorrect.

OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.

EXPLANATION:

A screenshot of a computer program

AI-generated content may be incorrect.

**Task 2: Optimizing Bubble Sort → Insertion Sort**

**PROMPT:**

**Python implementation of Bubble Sort. I want you to analyze it and suggest a more efficient sorting algorithm for a partially sorted array. Then, provide the code for the suggested algorithm and explain why it performs better on this specific type of input.**

**CODE:**

A computer screen shot of a program code

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**EXPLANATION:**

A screenshot of a computer

AI-generated content may be incorrect.

**Task 3: Binary Search vs Linear Search**

**PROMPT:**

**1.Add detailed docstrings to both functions.**

**2.Include performance notes in the docstrings, explaining the time complexity for best and worst-case scenarios.**

**3. Explain the key differences between the two algorithms.**

**4. Explain when Binary Search is the better choice and why.**

**CODE:**

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

**EXPLANATION:**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Task 4: Quick Sort and Merge Sort Comparison**

**PROMPT:**

**compare the performance of Quick Sort and Merge Sort on three types of input:**

1. **A randomly ordered list**
2. **An already sorted list**
3. **A reverse-sorted list**

**CODE:**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**OUTPUT:**

A number on a white background

AI-generated content may be incorrect.

**EXPLANTION:**

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Task 5: AI-Suggested Algorithm Optimization**

**PROMPT:**

**"I wrote a brute-force Python function to find duplicates in a list using nested loops (O(n²) time).  
Can you optimize this algorithm to run in O(n) time using a more efficient approach like sets or dictionaries?  
Please explain how your optimized solution improves the time complexity and why it performs better on large inputs."**

**CODE:**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

**EXPLANATION:**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.