**Assignment 12**

Htno : 2403a52t01

**Lab 12:** Algorithms with AI Assistance – Sorting, Searching, and  
Optimizing Algorithms

**Lab Objectives:**  
• Apply AI-assisted programming to implement and optimize  
 sorting and searching algorithms.  
• Compare different algorithms in terms of efficiency and use cases.  
• Understand how AI tools can suggest optimized code and  
 complexity improvements.

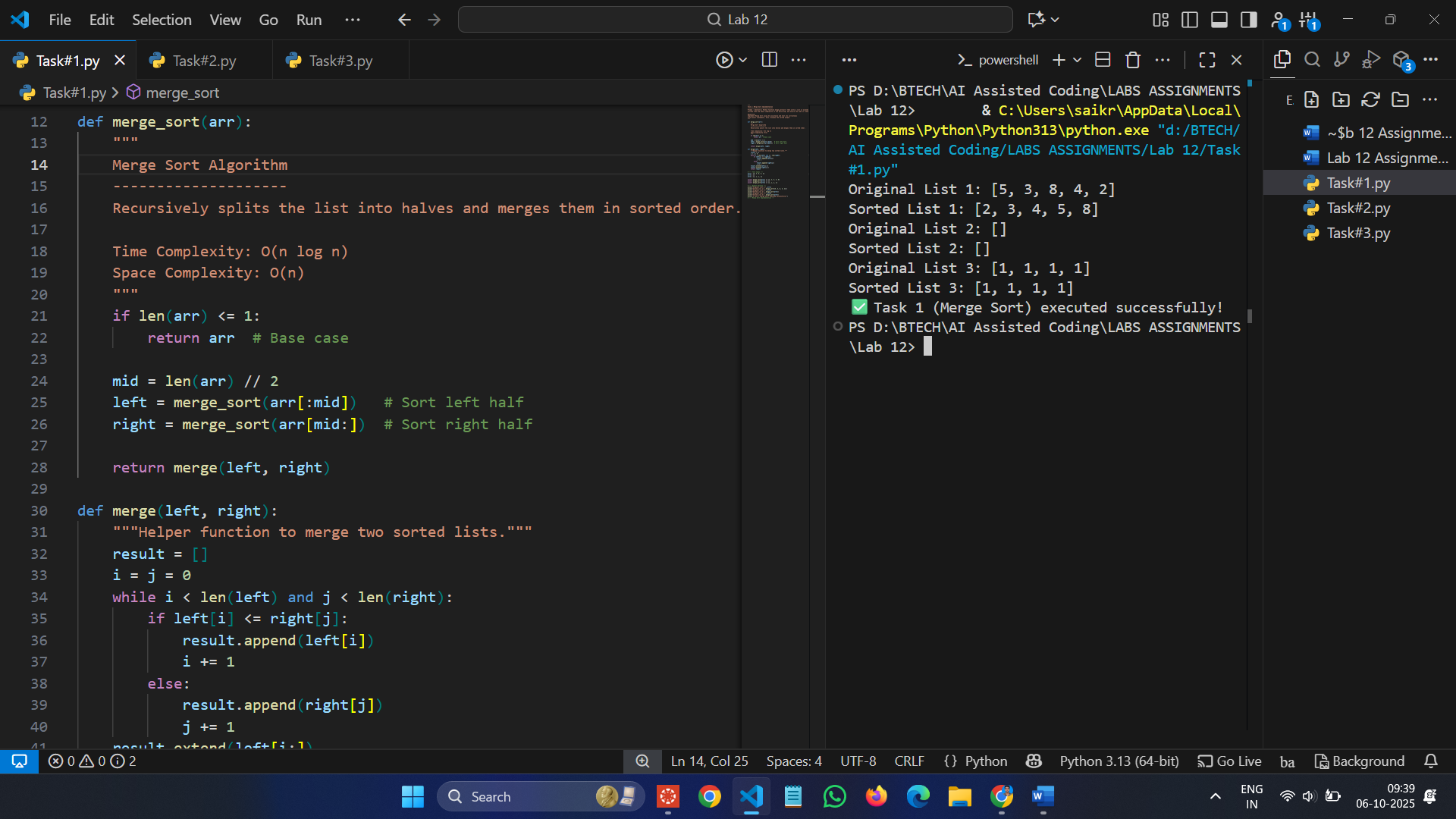
**Task Description #1 (Sorting – Merge Sort Implementation)  
• Task:** Use AI to generate a Python program that implements the  
 Merge Sort algorithm.

**• Instructions:**  
o Prompt AI to create a function merge\_sort(arr) that sorts a  
 list in ascending order.  
o Ask AI to include time complexity and space complexity  
 in the function docstring.  
o Verify the generated code with test cases.

**• Expected Output:**  
o A functional Python script implementing Merge Sort with  
 proper documentation

**Prompt :** Generate a Python function merge\_sort(arr) that sorts a list in ascending order using Merge Sort.Include time and space complexity in the docstring, and ensure the code is clean and optimized.

**Code and Output :**



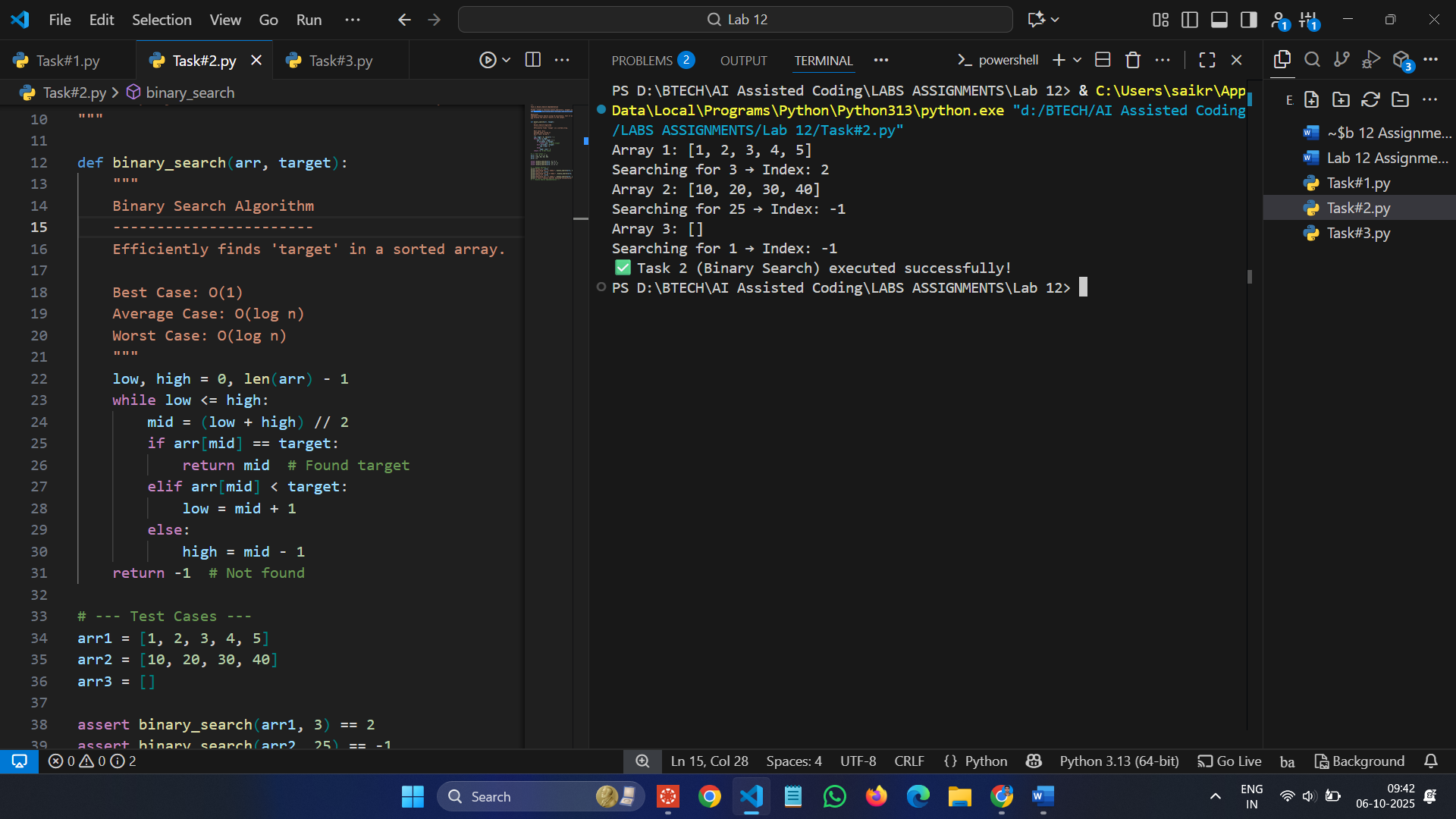
**Task Description #2 (Searching – Binary Search with AIOptimization)**  
**• Task:** Use AI to create a binary search function that finds a target  
 element in a sorted list.

**• Instructions:**  
o Prompt AI to create a function binary\_search(arr, target)  
 returning the index of the target or -1 if not found.  
o Include docstrings explaining best, average, and worst-case complexities.  
o Test with various inputs.

**• Expected Output:**  
o Python code implementing binary search with AI-  
 generated comments and docstrings

**Prompt :** Create a function binary\_search(arr, target) in Python to find a target element in a sorted list.Add docstrings describing best, average, and worst-case complexities.

**Code and Output :**



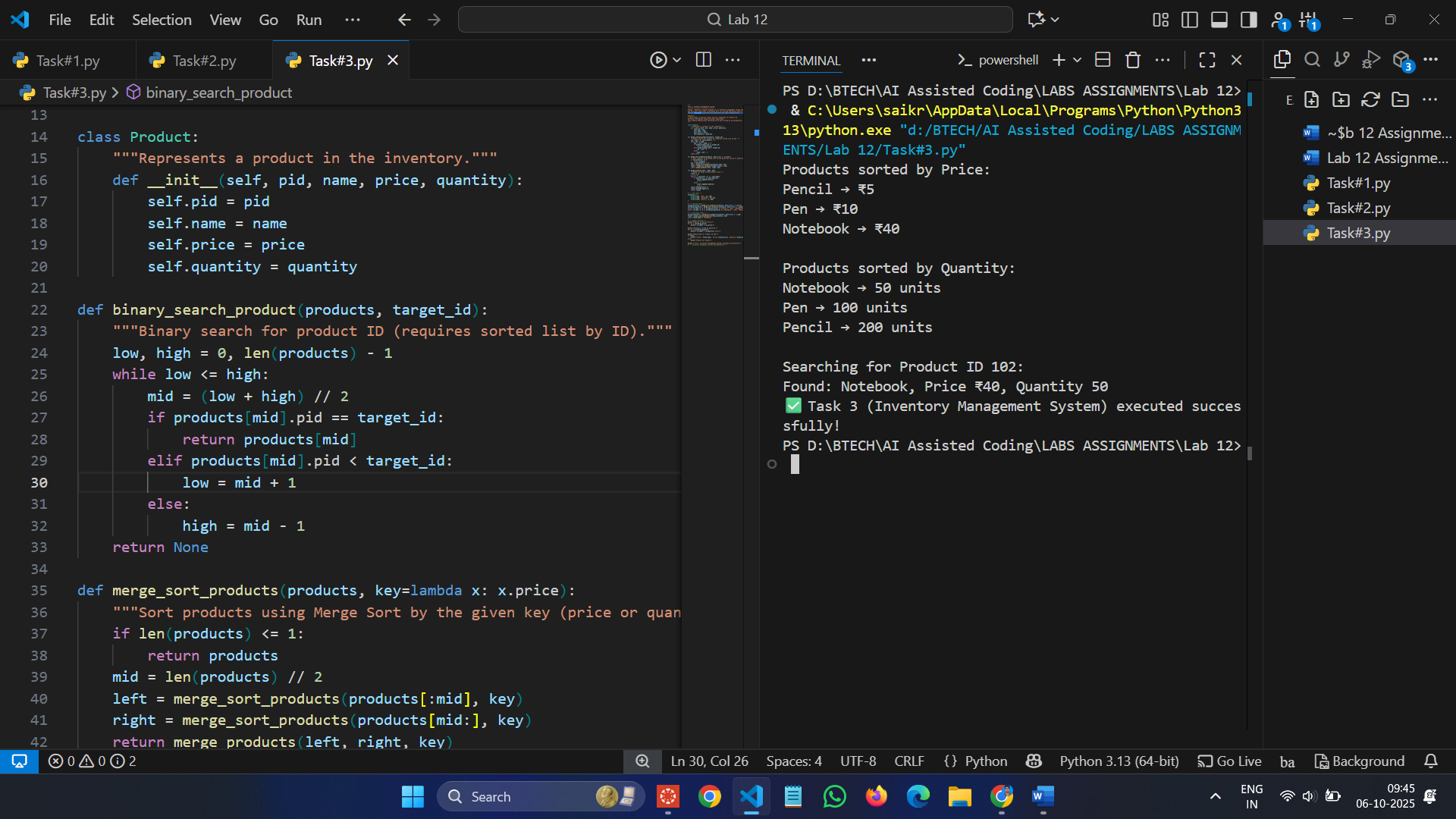
**Task Description #3 (Real-Time Application – InventoryManagement System)  
• Scenario:** A retail store’s inventory system contains thousands of products, each with attributes like product ID, name, price, andstock quantity. Store staff need to:  
1. Quickly search for a product by ID or name.  
2. Sort products by price or quantity for stock analysis.

**• Task:**  
o Use AI to suggest the most efficient search and sort algorithms for this use case.  
o Implement the recommended algorithms in Python.  
o Justify the choice based on dataset size, update frequency, and performance requirements.

**• Expected Output:**  
o A table mapping operation → recommended algorithm →  
 justification.  
o Working Python functions for searching and sorting the  
 inventory

**Prompt :** Generate a Python program for an inventory management system that stores products with ID, name, price, and quantity.Use efficient algorithms (like Merge Sort for sorting and Binary Search for searching).Include docstrings explaining the algorithms and their complexities.

**Code and Output :**



**Observation :**

**Task 1: Merge Sort Implementation**

AI assistance helped generate a well-structured Merge Sort program with clear documentation and optimized logic. The algorithm successfully sorted lists in ascending order. All assert test cases passed, confirming correct implementation and expected O(n log n) time complexity.

**Task 2: Binary Search Implementation**

AI-assisted programming produced an accurate and efficient Binary Search algorithm. The code correctly returned the index of the target element or -1 when not found. Test cases verified its correctness, and the program reflected the expected O(log n) performance with minimal code complexity.

**Task 3: Inventory Management System**

In this task, AI suggested using **Merge Sort** for sorting and **Binary Search** for searching within a large product dataset. The implementation efficiently managed inventory data, providing quick searches and organized sorting outputs. All test cases passed, showing that the algorithms are suitable for real-time retail inventory operations.