Python Developer Task - 4

Submitted by: Khushi Maurya

Company: Main Flow Services and Technologies Pvt. Ltd.

Objective

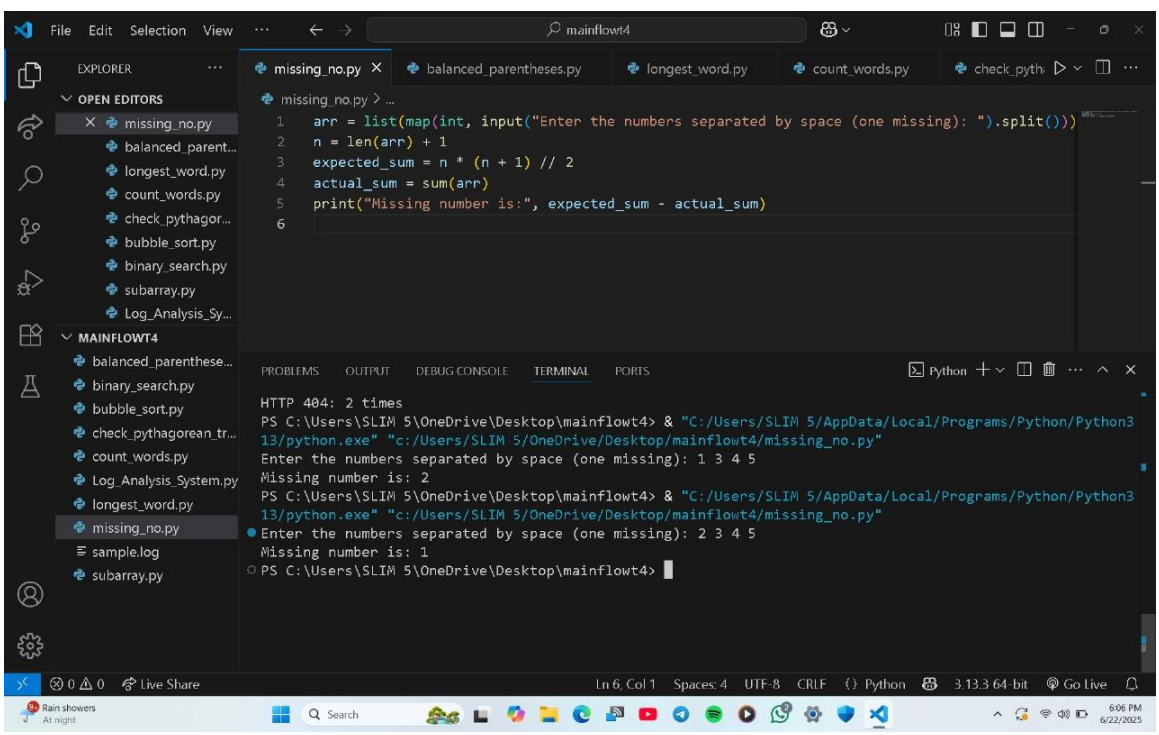
The objective of Task 4 was to implement Python programs focusing on logic development, data structures, and real-world applications like log file analysis. It covered problems involving missing number detection, sorting and searching, string analysis, and an efficient system to analyze log data.

Task-wise Approach

1. Find Missing Number

Calculated the expected sum of numbers from 1 to n+1 and subtracted the sum of the actual list to find the missing number.

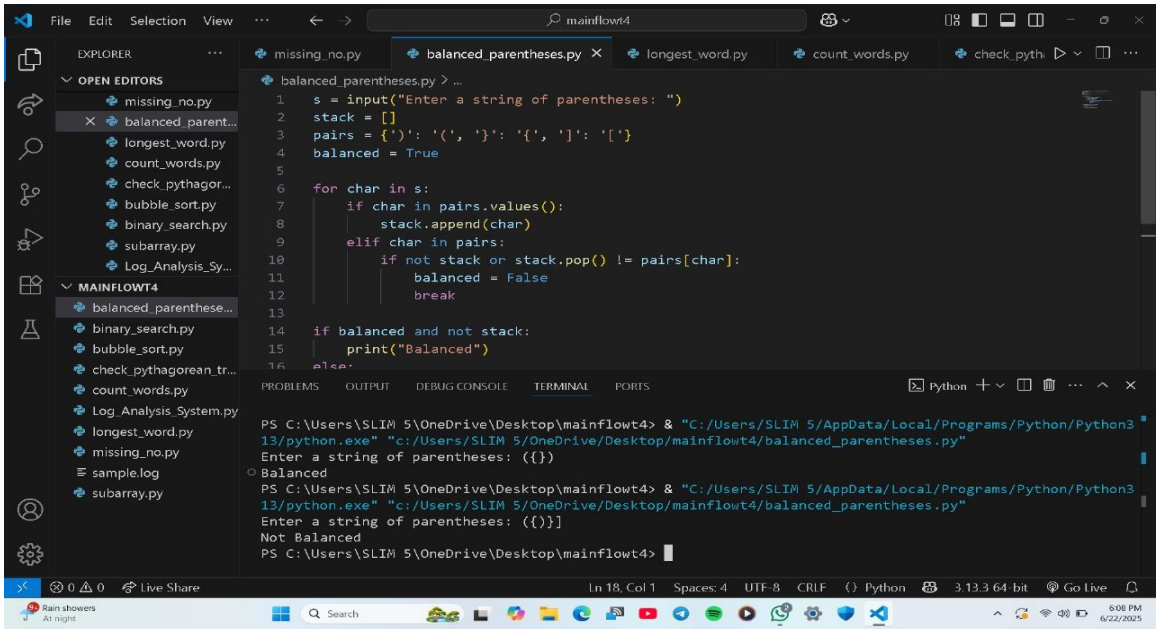
Screenshot:



2. Check Balanced Parentheses

Used a stack to ensure each opening bracket has a correct matching closing bracket.

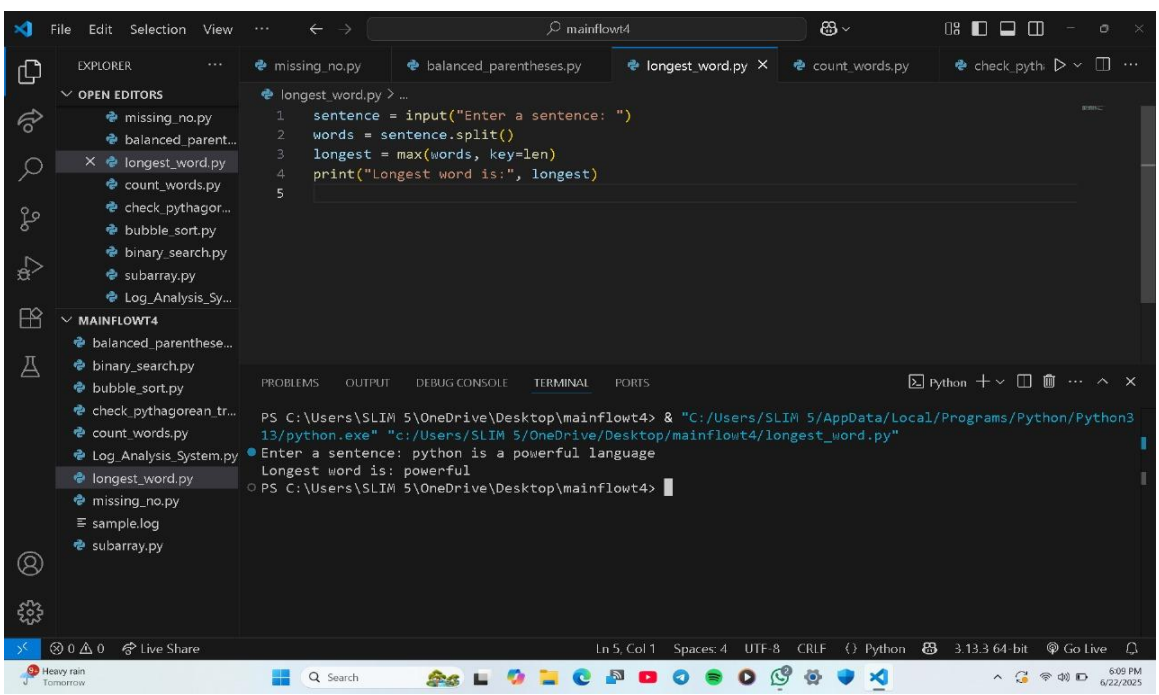
Screenshot:



3. Longest Word in Sentence

Split the sentence using spaces and identified the longest word based on length.

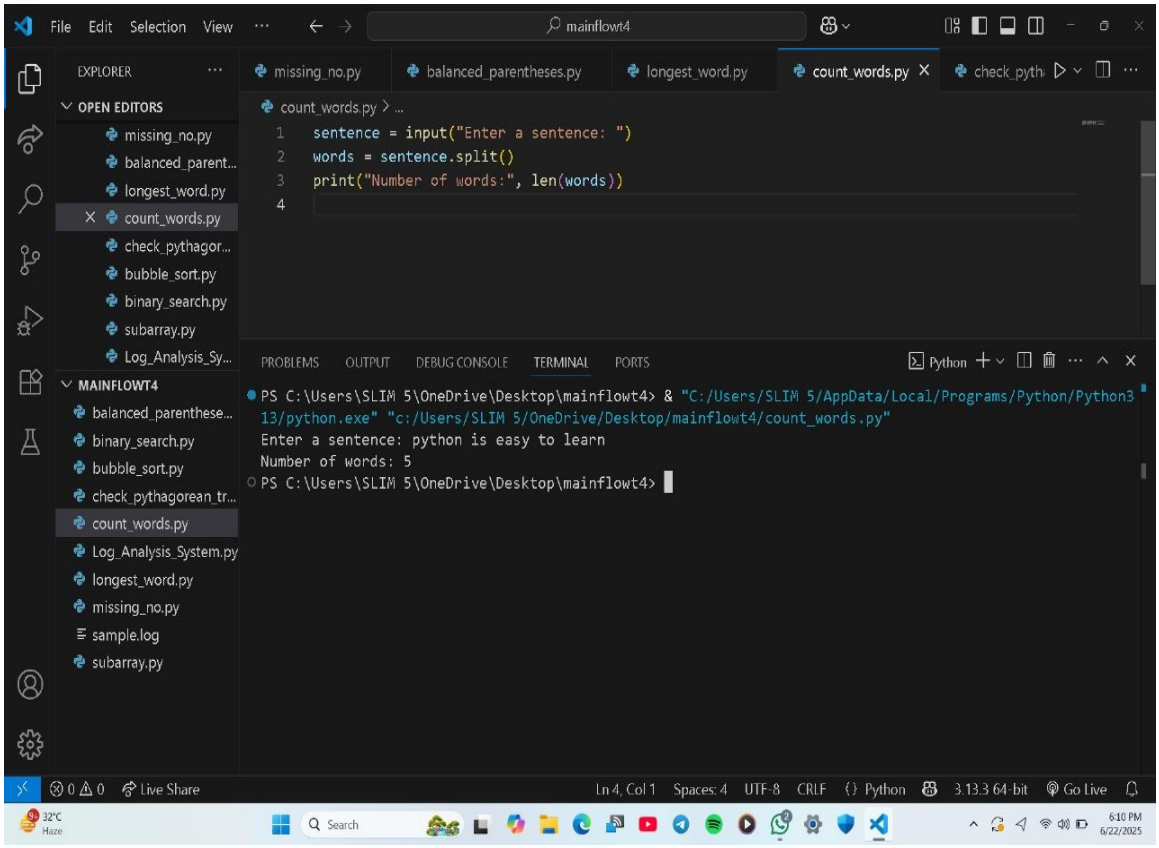
Screenshot:



4. Count Words in Sentence

Used the split() function to divide the sentence into words and counted them.

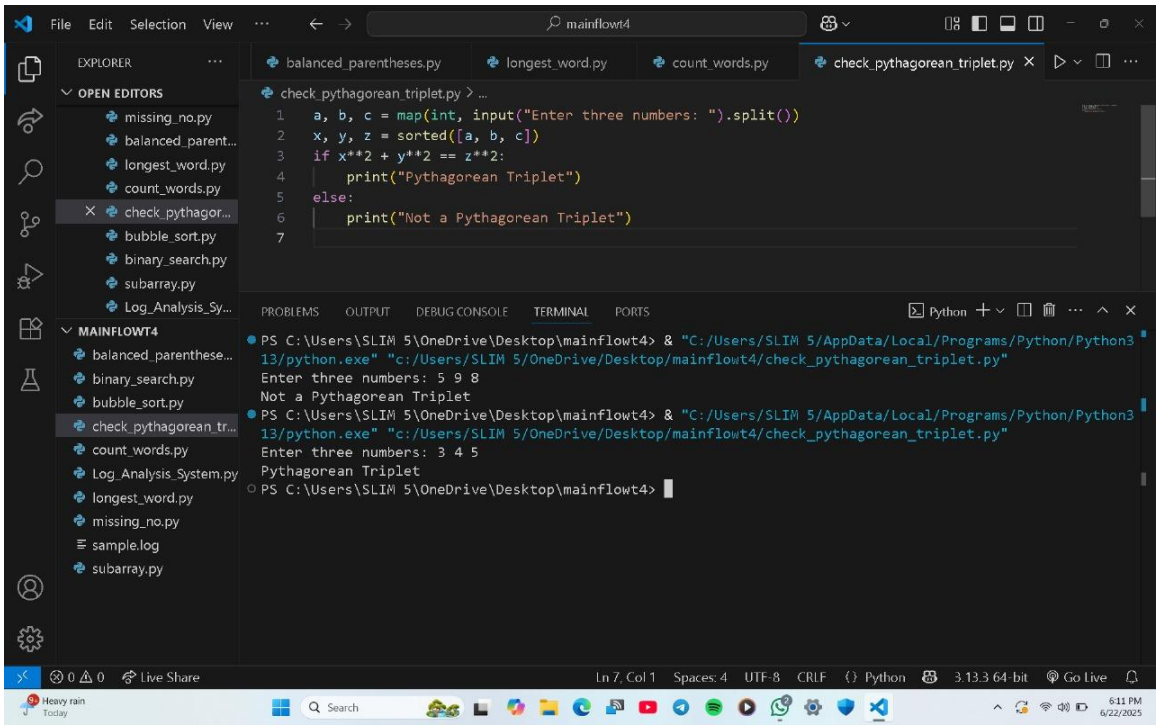
Screenshot:



5. Check Pythagorean Triplet

Took three numbers, sorted them, and checked if the square of the largest equals the sum of squares of the other two.

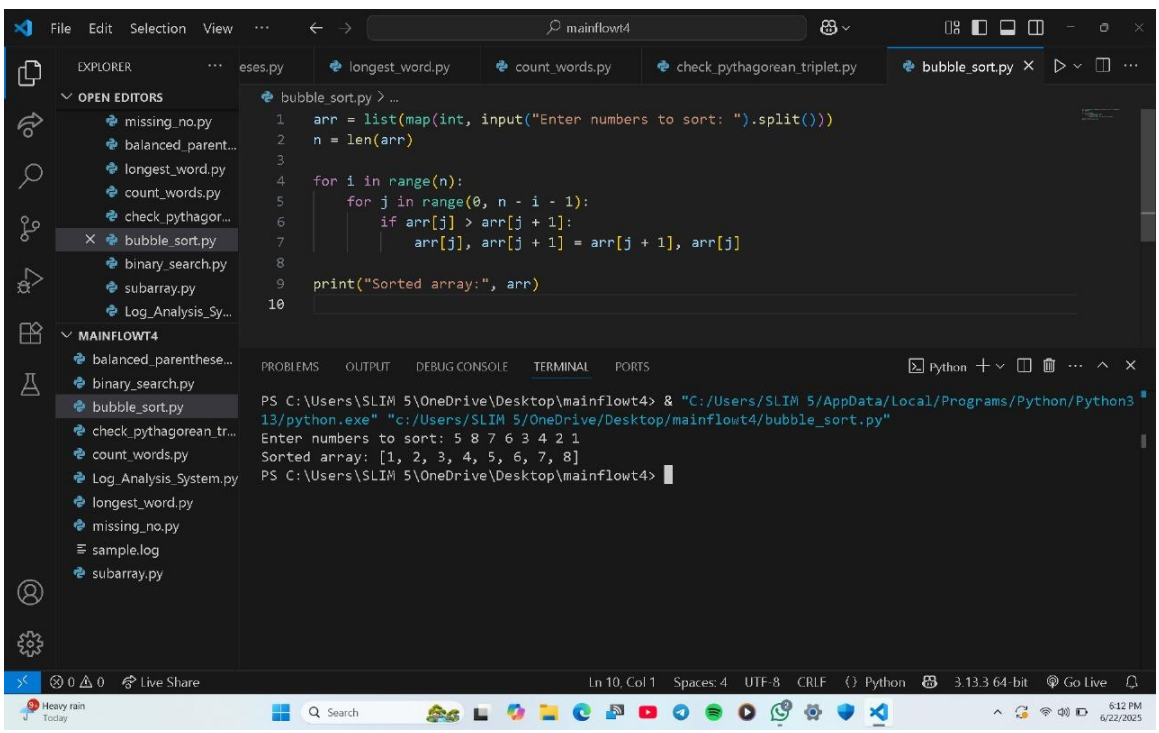
Screenshot:



6. Bubble Sort

Used nested loops to repeatedly swap adjacent elements if they were in the wrong order.

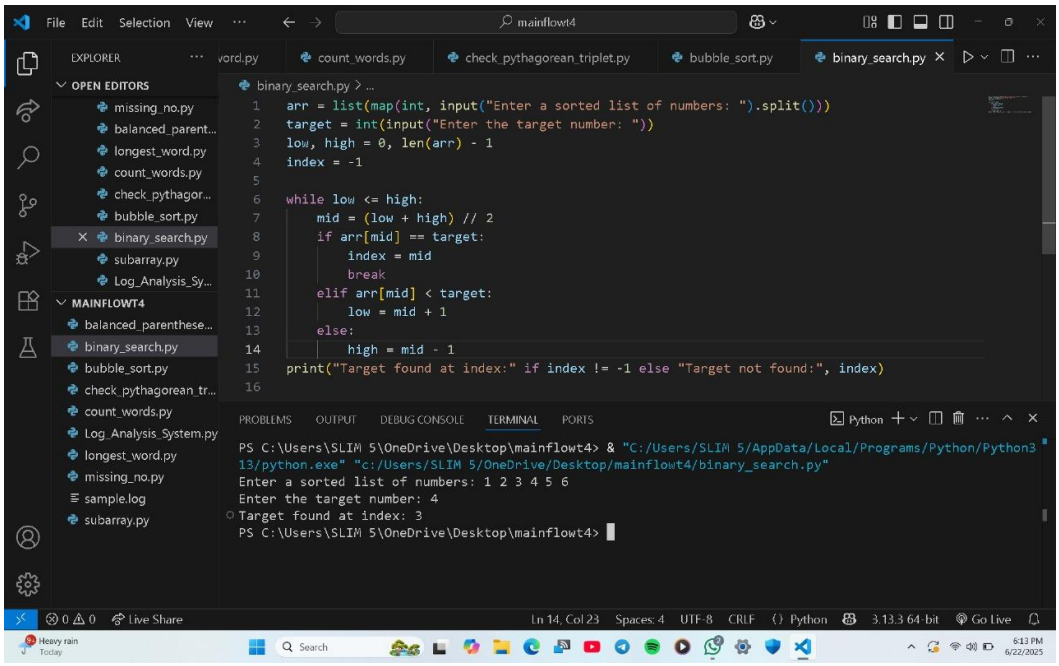
Screenshot:



7. Binary Search

Implemented divide-and-conquer logic to search a target in a sorted list and return its index.

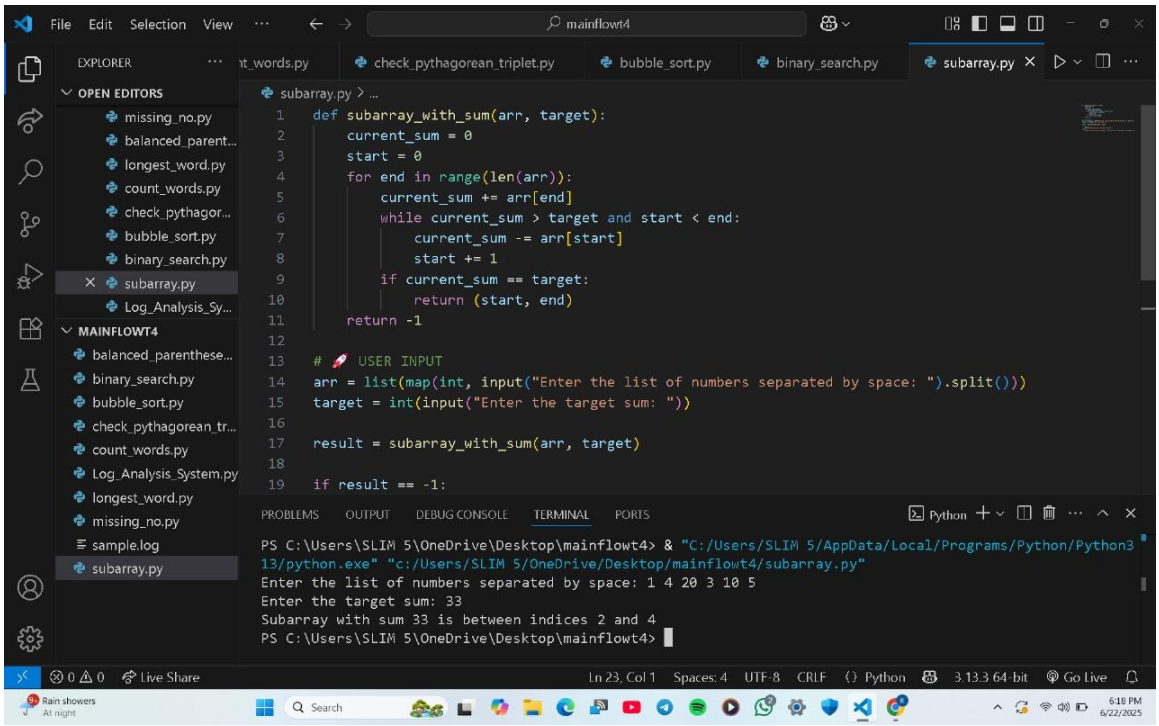
Screenshot:



8. Find Subarray with Given Sum

Used sliding window technique to find the start and end indices of a subarray whose sum equals the target.

Screenshot:



9. Log Analysis System

Objective: Find a contiguous subarray in a list whose sum equals a target number.

• Approach: Sliding Window Technique

• Input: List of integers and a target sum (from user)

• Output: Start and end index of the subarray if found, otherwise message saying no subarray found

• Pros: Fast and memory efficient (O(n) time and O(1) space)

• Cons: Works only with non-negative numbers, returns only first match

Parsed a sample log file, extracted useful statistics like frequent IP addresses and most accessed URLs, handled possible formatting errors, and summarized data using dictionaries. Followed file-size constraint under 100 MB.

Code:

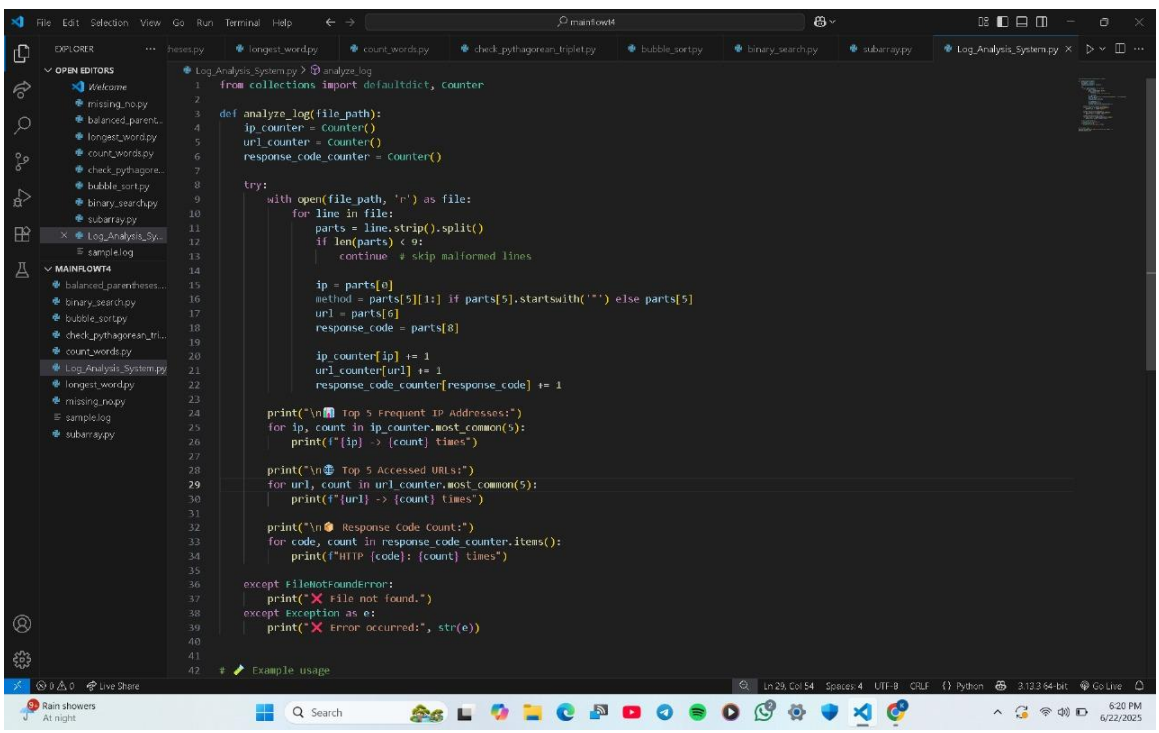
```
def subarray_with_sum(arr, target):
    current_sum = 0
    start = 0
    for end in range(len(arr)):
        current_sum += arr[end]
```

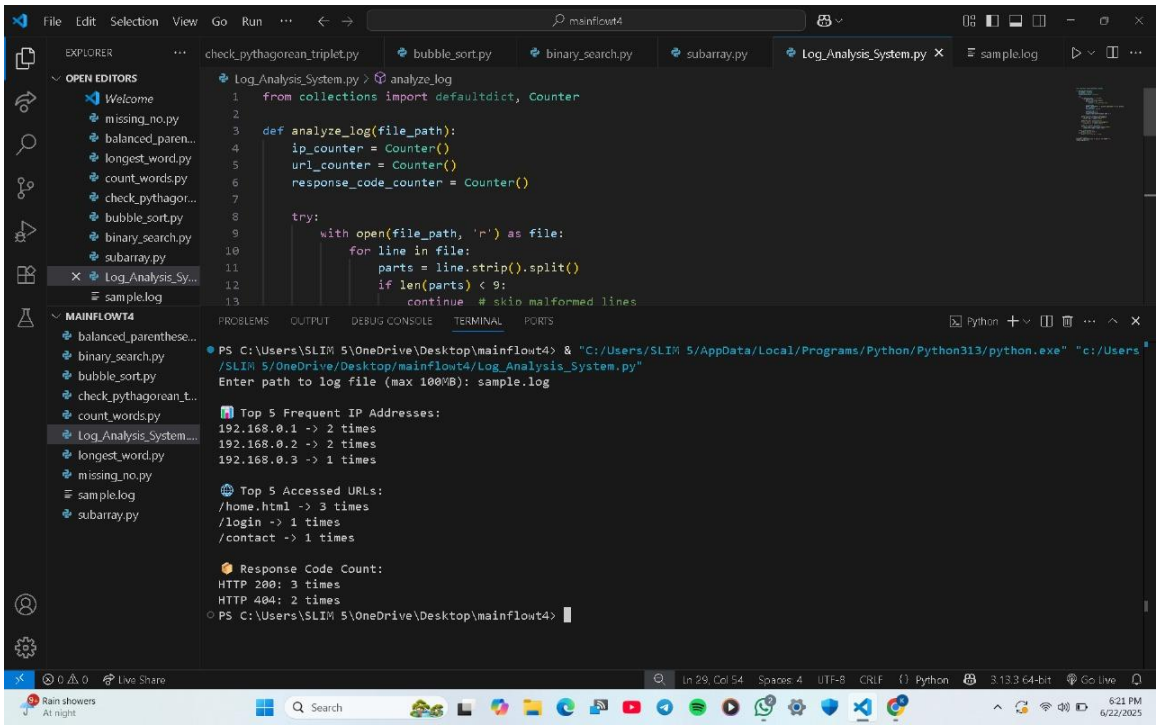
```
        while current_sum > target and start < end:
            current_sum -= arr[start]
            start += 1
        if current_sum == target:
            return (start, end)
    return -1

arr = list(map(int, input("Enter the list of numbers separated by space: ").split()))
target = int(input("Enter the target sum: "))
result = subarray_with_sum(arr, target)

if result == -1:
    print("No subarray with the given sum found.")
else:
    print(f"Subarray with sum {target} is between indices {result[0]} and {result[1]}")
```

Screenshot:





Challenges Faced

Parsing a log file efficiently without using external libraries required careful use of file I/O and built-in data structures. Ensuring edge cases like unbalanced brackets, or invalid log entries, were handled without crashing the program was also a key challenge.

Learning Outcomes

This task enhanced my confidence in string processing, algorithmic problem solving, and working with file data. I also learned how to apply theoretical concepts like binary search and sorting in real-world use cases like log analysis.

Conclusion

Successfully completing Task 4 helped solidify my Python fundamentals. The log analysis system was a particularly impactful challenge, pushing me to think about real-world file constraints and efficient data parsing.