

# LAB EXAM – 1 (01-09-2025)

HALL-TICKET NUMBER: 2403A51410

NAME: AMGOTH VIKAS NAYAK

## QUESTION 1 :

Prompt:

```
33
34 def main():
35     # Sample list of numbers
36     numbers = [64, 34, 25, 12, 22, 11, 90, 88, 76, 45]
37
38     print("Original list:", numbers)
39     print("-" * 50)
40
41     # Method 1: Using built-in sort() method (modifies original list)
42     numbers_copy1 = numbers.copy()
43     numbers_copy1.sort()
44     print("1. Using sort() method (ascending):", numbers_copy1)
45
46     # Method 2: Using built-in sort() method with reverse=True (descending)
47     numbers_copy2 = numbers.copy()
48     numbers_copy2.sort(reverse=True)
49     print("2. Using sort() method (descending):", numbers_copy2)
50
51     # Method 3: Using sorted() function (creates new list)
52     sorted_asc = sorted(numbers)
53     print("3. Using sorted() function (ascending):", sorted_asc)
54
55     # Method 4: Using sorted() function with reverse=True (descending)
56     sorted_desc = sorted(numbers, reverse=True)
57     print("4. Using sorted() function (descending):", sorted_desc)
58
59     print("-" * 50)
60     print("Manual Sorting Algorithms:")
61     print("-" * 50)
62
63     # Method 5: Bubble Sort
64     numbers_copy3 = numbers.copy()
65     bubble_sorted = bubble_sort(numbers_copy3)
66     print("5. Bubble Sort:", bubble_sorted)
67
68     # Method 6: Selection Sort
69     numbers_copy4 = numbers.copy()
```

Write a python program for sorting a list of numbers

LABEXAM-1.py

write a python programme for sorting a list of numbers

Restore checkpoint

I'll help you write a Python program for sorting a list of numbers. Let me create a comprehensive program that demonstrates different sorting methods.

LABEXAM-1.py +103

```
Python Program for Sorting a List of Numbers
def bubble_sort(arr):
    """Bubble sort algorithm implementation"""
    n = len(arr)
    for i in range(n):
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]
    return arr
```

Perfect! I've created a comprehensive Python program for sorting a list of numbers. The program includes:

Features:

- 1. Built-in Python Methods:
  - sort() method (modifies original list)
  - sorted() function (creates new list)
  - Both ascending and descending order

1 file edited +12 -1

LABEXAM-1.py

Plan, search, build anything

Agent Out Auto

Cursor Tab Ln 103, Col 11 Spaces: 4 UTF-8 CRLF Python 3.13.7 (venv: venv)

26°C Rain

Search

ENG US

09:50 01-09-2025

## CODE GENERATED BY AI AND ITS OUTPUT:

The screenshot displays a code editor with a file explorer on the left, a central code editor, and a right-hand panel with AI-generated content.

**File Explorer (Left):** Shows a project named 'VIKASAIAC' with files including 'venv', '7-4-3.py', '7-4-1.py', '7-4-2.py', '7-4-4.py', '7-4-5.py', 'aasgn2task2\_fixed.py', 'aasgn2task2.py', 'aasgn2task1.py', 'aasgn2task3.py', 'LAB ASSIGNMENT-65-243A51410.pdf', 'LAB-1.1.py', 'LAB-3-3.py', 'LAB-3-4.py', 'LAB-3-5.py', 'LAB-3-6.py', 'LAB-3.T-2.py', 'LAB-3.py', 'LABEXAM-1.py', and 'users.txt'.

**Code Editor (Center):** Displays the content of 'LABEXAM-1.py'. The code defines a `main()` function that sorts a list of numbers using six different methods:

```
33
34 def main():
35     # Sample list of numbers
36     numbers = [64, 34, 25, 12, 22, 11, 90, 88, 76, 45]
37
38     print("Original list:", numbers)
39     print("-" * 50)
40
41     # Method 1: Using built-in sort() method (modifies original list)
42     numbers_copy1 = numbers.copy()
43     numbers_copy1.sort()
44     print("1. Using sort() method (ascending):", numbers_copy1)
45
46     # Method 2: Using built-in sort() method with reverse=True (descending)
47     numbers_copy2 = numbers.copy()
48     numbers_copy2.sort(reverse=True)
49     print("2. Using sort() method (descending):", numbers_copy2)
50
51     # Method 3: Using sorted() function (creates new list)
52     sorted_asc = sorted(numbers)
53     print("3. Using sorted() function (ascending):", sorted_asc)
54
55     # Method 4: Using sorted() function with reverse=True (descending)
56     sorted_desc = sorted(numbers, reverse=True)
57     print("4. Using sorted() function (descending):", sorted_desc)
58
59     print("-" * 50)
60     print("Manual Sorting Algorithms:")
61     print("-" * 50)
62
63     # Method 5: Bubble Sort
64     numbers_copy3 = numbers.copy()
65     bubble_sorted = bubble_sort(numbers_copy3)
66     print("5. Bubble Sort:", bubble_sorted)
67
68     # Method 6: Selection Sort
69     numbers_copy4 = numbers.copy()
```

**Right Panel (AI Assistant):** Contains a chat interface with the prompt "Write a python program for sorting a list of numbers" and the response:

LABEXAM-1.py +103 ✓

write a python programme for sorting a list of numbers

Restore checkpoint

I'll help you write a Python program for sorting a list of numbers. Let me create a comprehensive program that demonstrates different sorting methods.

LABEXAM-1.py +103 ✓

Python Program for Sorting a List of Numbers

```
def bubble_sort(arr):
    """Bubble sort algorithm implementation"""
    n = len(arr)
    for i in range(n):
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]
    return arr
```

Perfect! I've created a comprehensive Python program for sorting a list of numbers. The program includes:

**Features:**

1. Built-in Python Methods:
  - `sort()` method (modifies original list)
  - `sorted()` function (creates new list)
  - Both ascending and descending order

> 1 file edited +12 -1

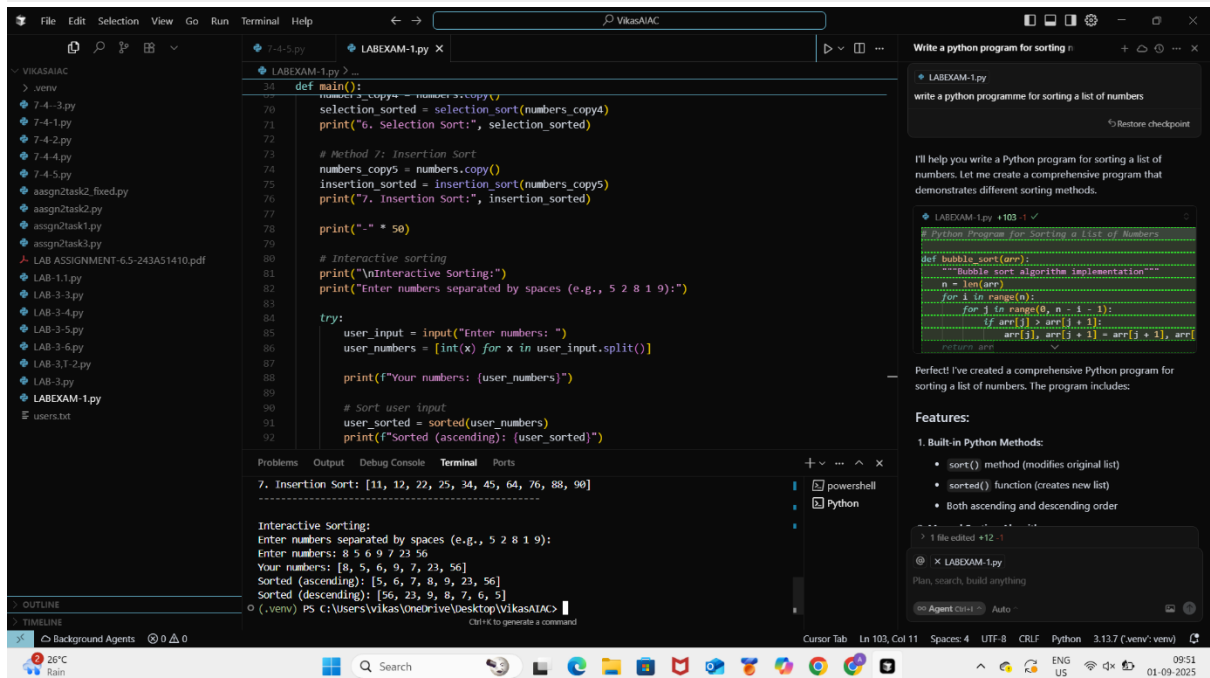
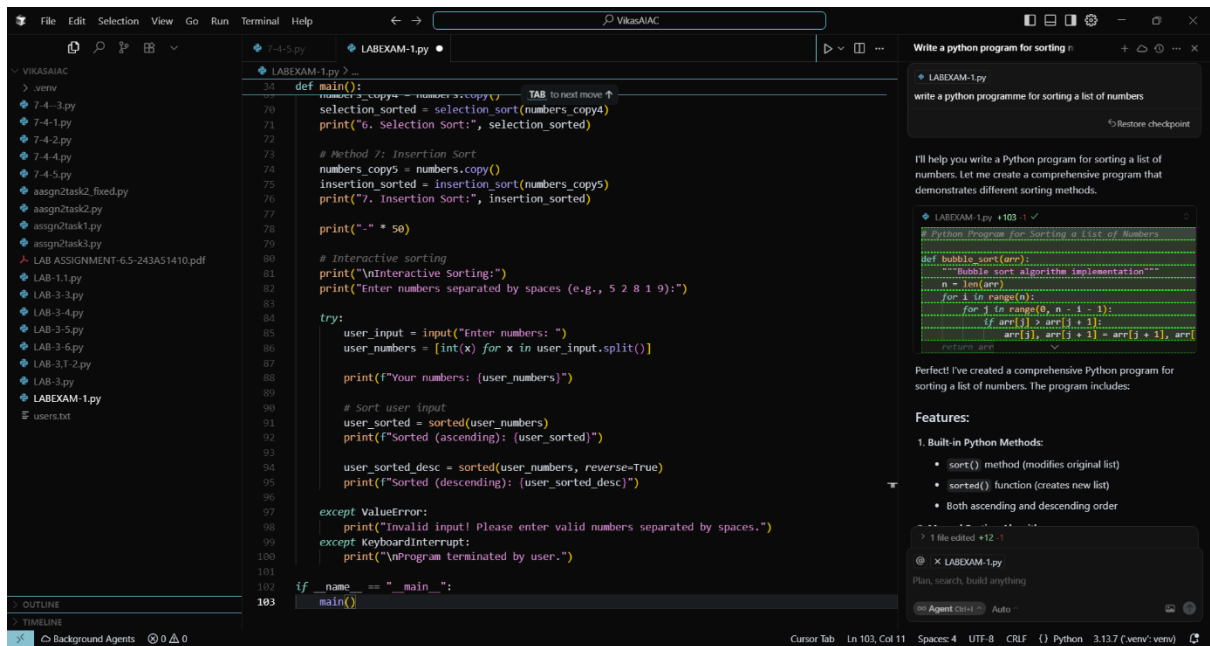
LABEXAM-1.py

Plan, search, build anything

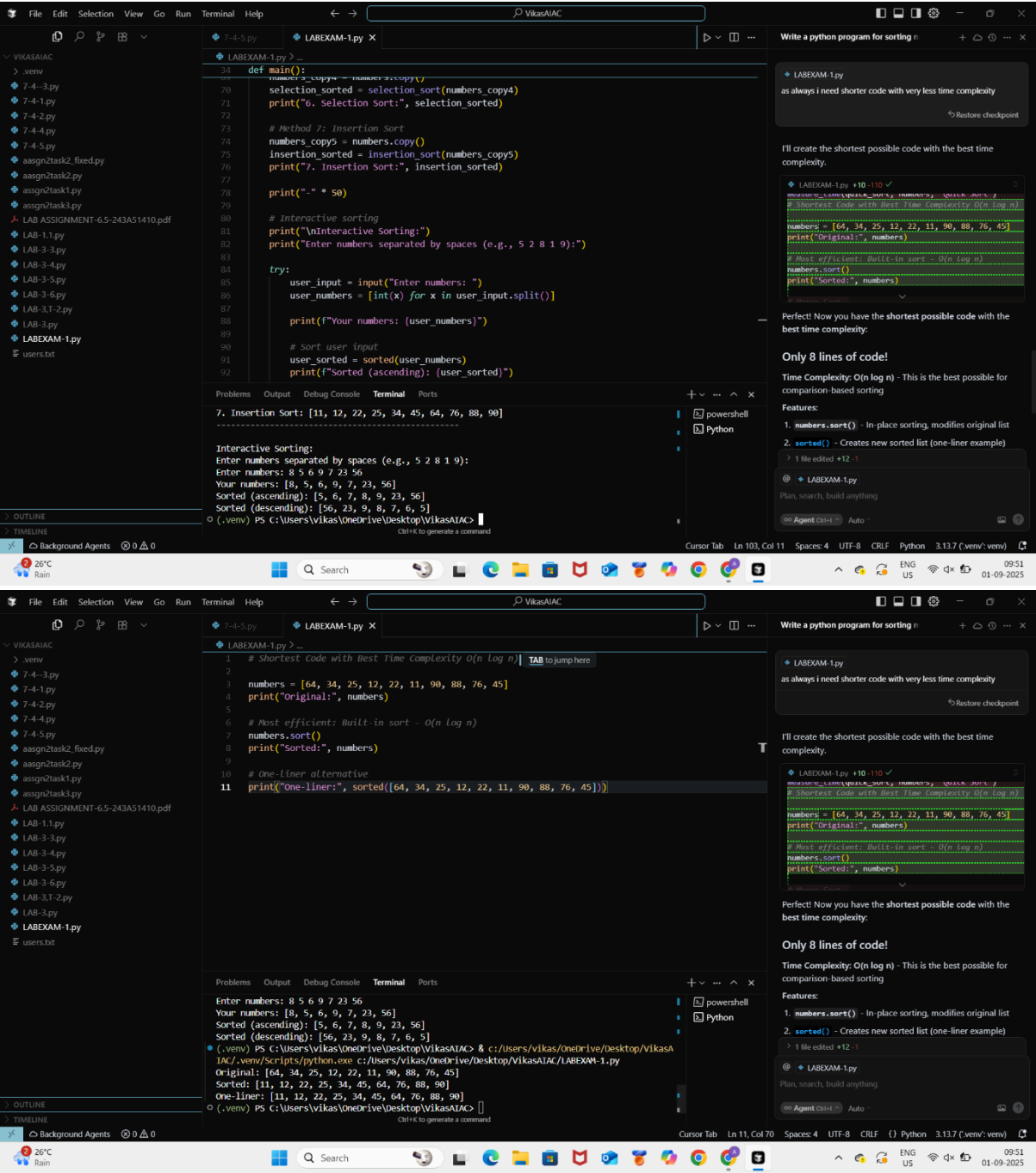
Agent Ctrl+L Auto

Cursor Tab Ln 103, Col 11 Spaces: 4 UTF-8 CRLF Python 3.13.7 (venv: venv)

26°C Rain 09:50 01-09-2025

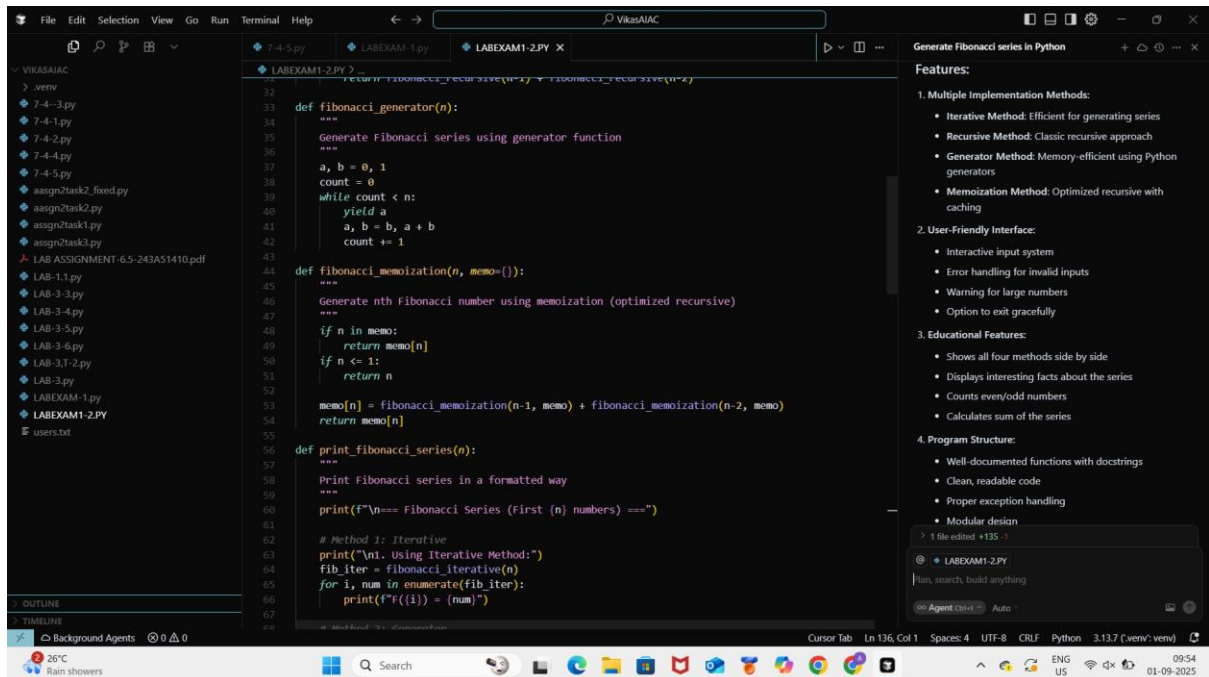


RE-PROMPTED AI CODE AND ITS OUTPUT:



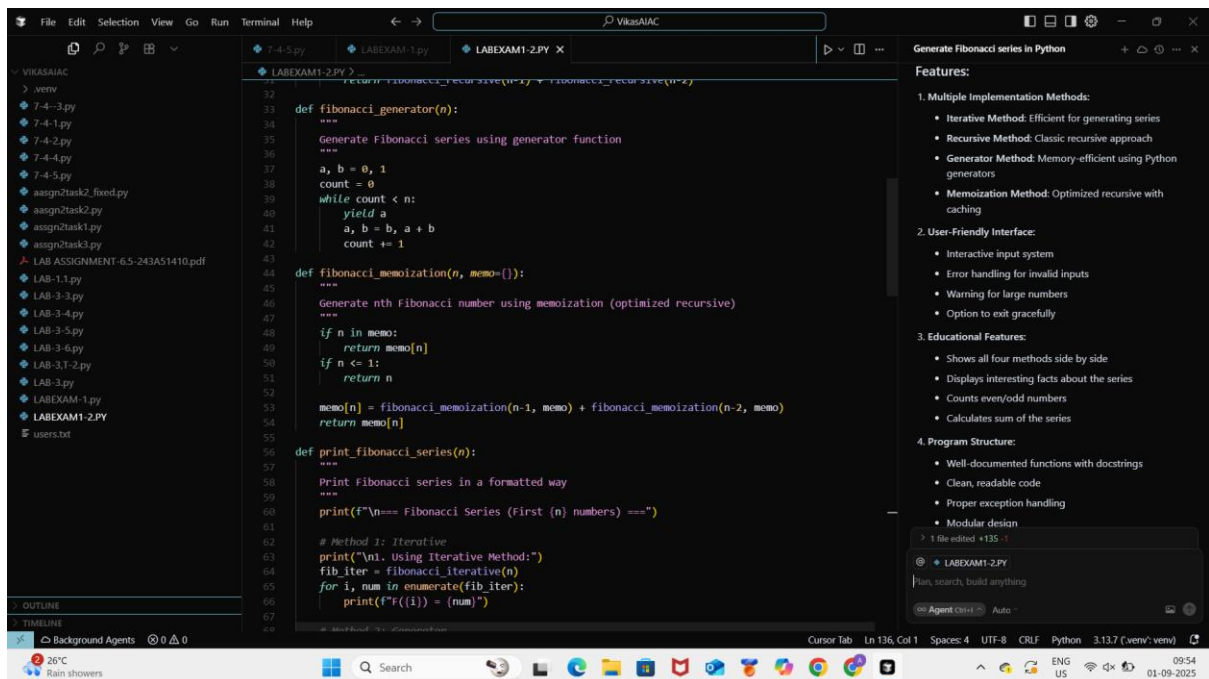
# QUESTION 2 :

## PROMPT:

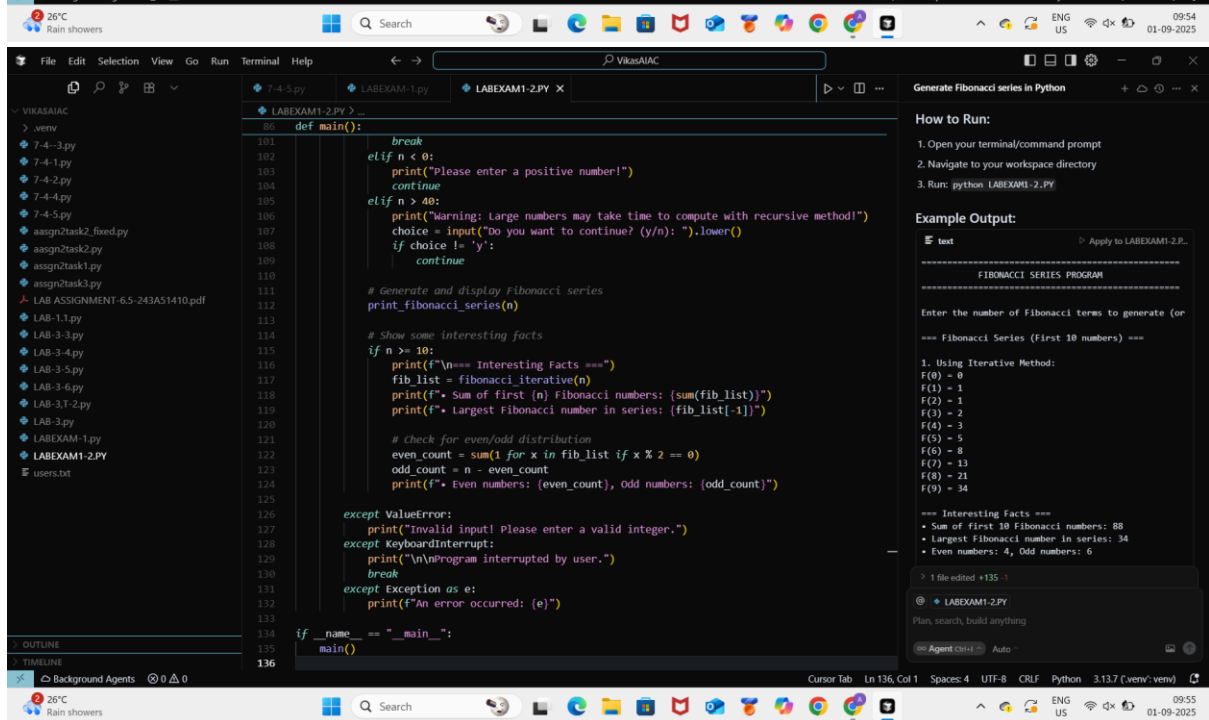
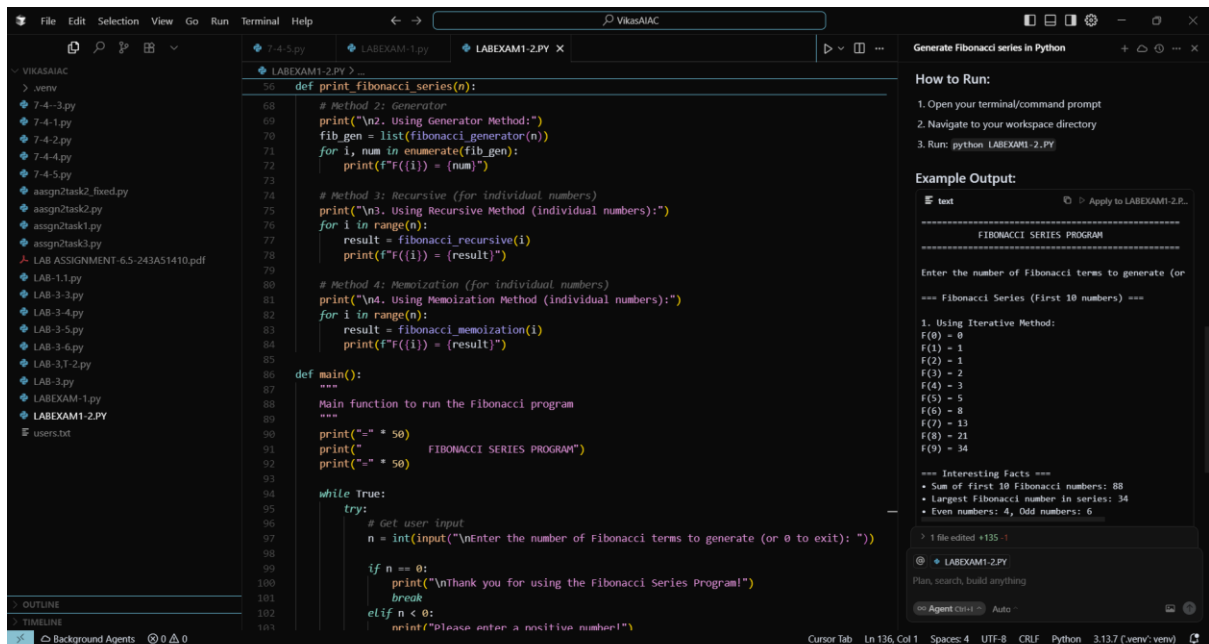


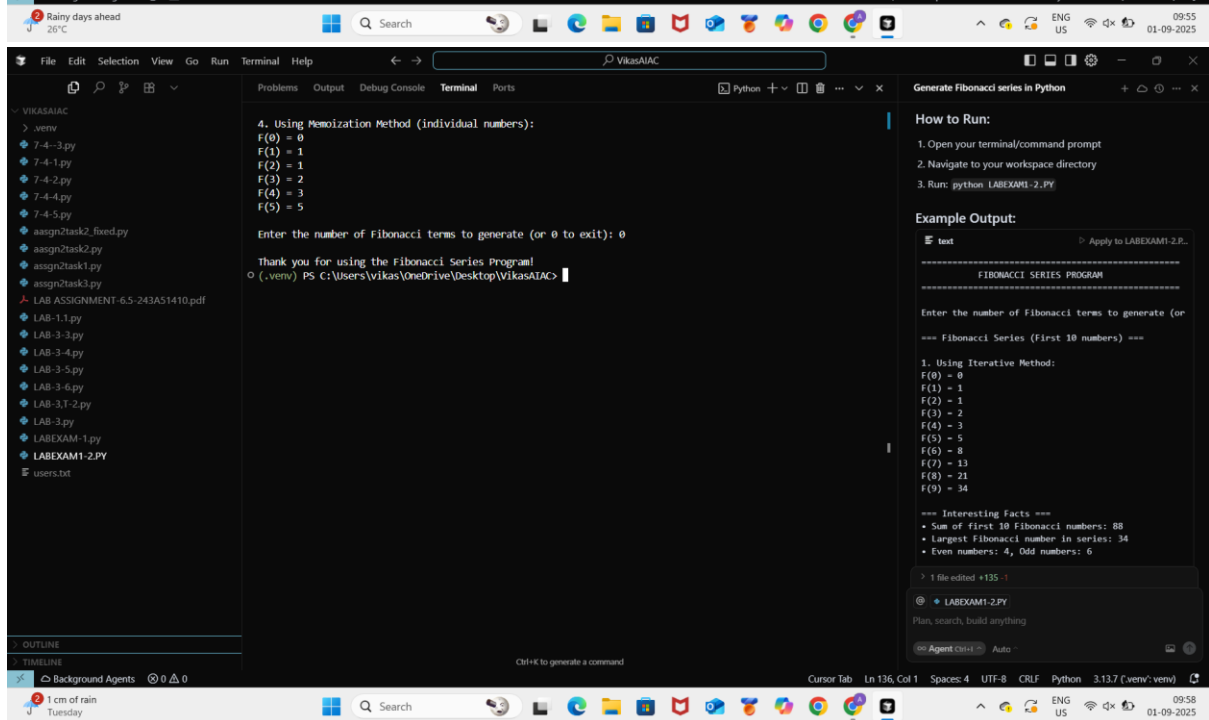
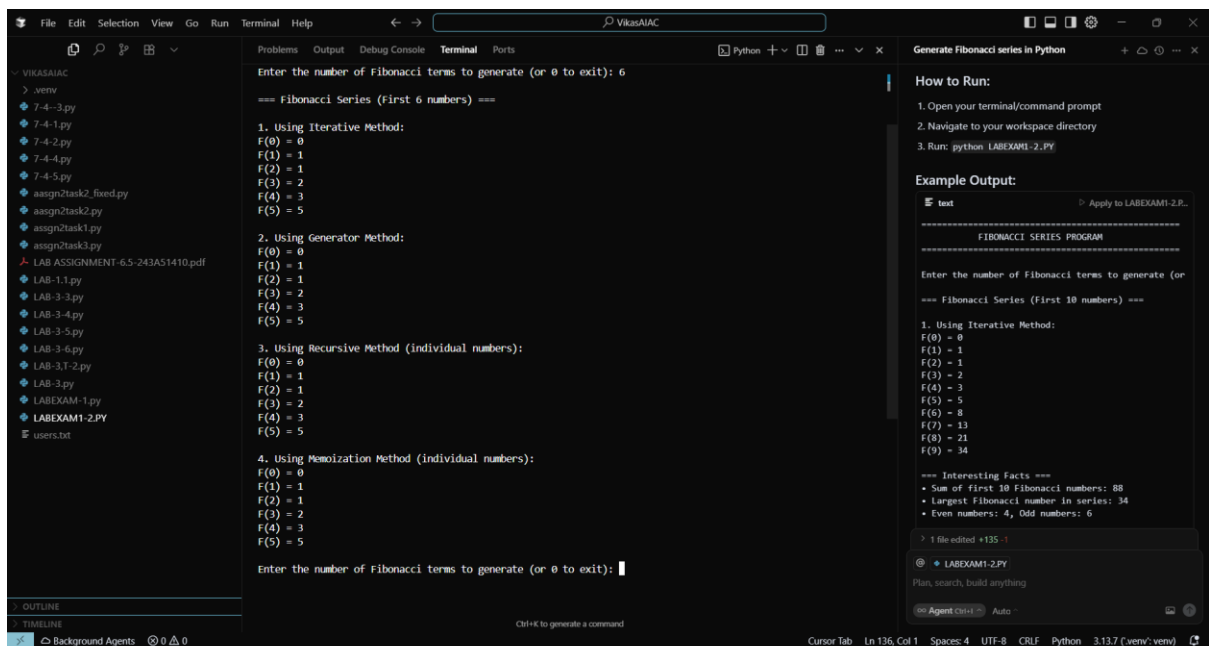
```
File Edit Selection View Go Run Terminal Help
LABEXAM1-2.PY
32 return fibonacci_recursive(n-2) + fibonacci_recursive(n-3)
33
34 def fibonacci_generator(n):
35     """
36     Generate Fibonacci series using generator function
37     """
38     a, b = 0, 1
39     count = 0
40     while count < n:
41         yield a
42         a, b = b, a + b
43         count += 1
44
45 def fibonacci_memoization(n, memo={}):
46     """
47     Generate nth Fibonacci number using memoization (optimized recursive)
48     """
49     if n in memo:
50         return memo[n]
51     if n <= 1:
52         return n
53     memo[n] = fibonacci_memoization(n-1, memo) + fibonacci_memoization(n-2, memo)
54     return memo[n]
55
56 def print_fibonacci_series(n):
57     """
58     Print Fibonacci series in a formatted way
59     """
60     print(f"\n=== Fibonacci Series (First {n} numbers) ===")
61
62     # Method 1: Iterative
63     print("\n1. Using Iterative Method:")
64     fib_iter = fibonacci_iterative(n)
65     for i, num in enumerate(fib_iter):
66         print(f"{i}) = {num}")
67
68 # Method 2: Recursive
69 print("\n2. Using Recursive Method:")
70 fib_recursive = fibonacci_recursive(n)
71 for i, num in enumerate(fib_recursive):
72     print(f"{i}) = {num}")
73
74 # Method 3: Generator
75 print("\n3. Using Generator Method:")
76 fib_gen = fibonacci_generator(n)
77 for i, num in enumerate(fib_gen):
78     print(f"{i}) = {num}")
79
80 if __name__ == "__main__":
81     n = 10
82     print_fibonacci_series(n)
83     fibonacci_recursive(n)
84     fibonacci_generator(n)
85     fibonacci_memoization(n)
```

## CODE GENERATED BY AI AND IT'S OUTPUT:

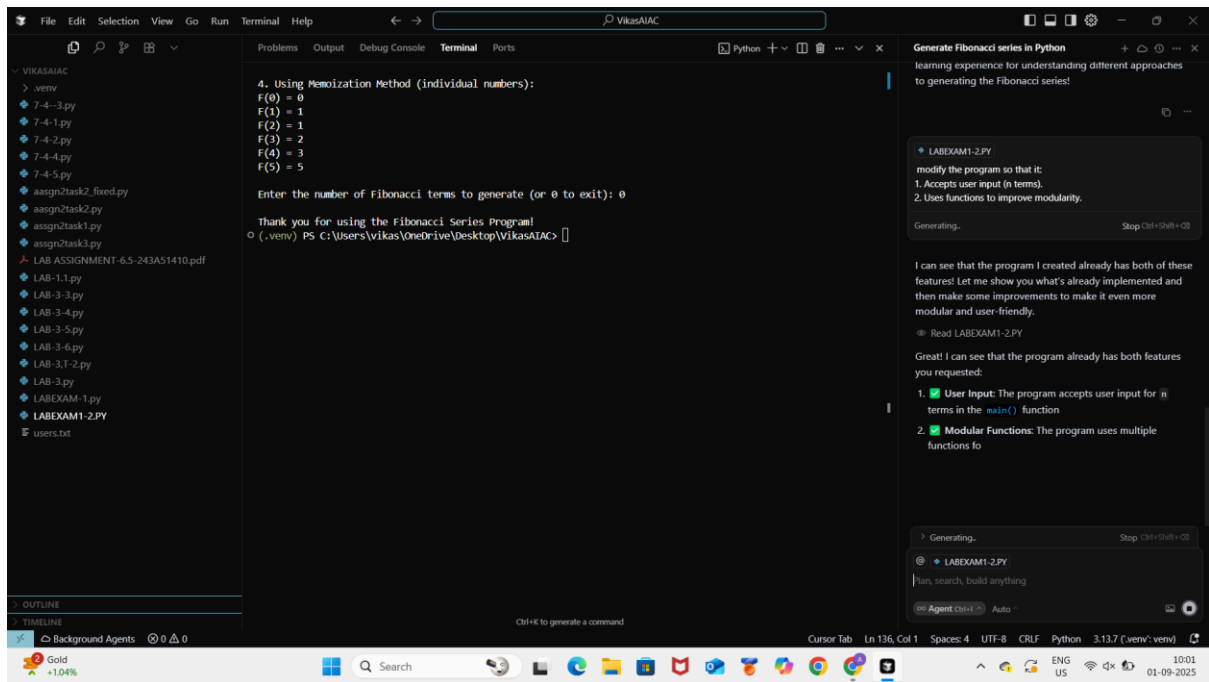


```
File Edit Selection View Go Run Terminal Help
LABEXAM1-2.PY
32 return fibonacci_recursive(n-2) + fibonacci_recursive(n-3)
33
34 def fibonacci_generator(n):
35     """
36     Generate Fibonacci series using generator function
37     """
38     a, b = 0, 1
39     count = 0
40     while count < n:
41         yield a
42         a, b = b, a + b
43         count += 1
44
45 def fibonacci_memoization(n, memo={}):
46     """
47     Generate nth Fibonacci number using memoization (optimized recursive)
48     """
49     if n in memo:
50         return memo[n]
51     if n <= 1:
52         return n
53     memo[n] = fibonacci_memoization(n-1, memo) + fibonacci_memoization(n-2, memo)
54     return memo[n]
55
56 def print_fibonacci_series(n):
57     """
58     Print Fibonacci series in a formatted way
59     """
60     print(f"\n=== Fibonacci Series (First {n} numbers) ===")
61
62     # Method 1: Iterative
63     print("\n1. Using Iterative Method:")
64     fib_iter = fibonacci_iterative(n)
65     for i, num in enumerate(fib_iter):
66         print(f"{i}) = {num}")
67
68 # Method 2: Recursive
69 print("\n2. Using Recursive Method:")
70 fib_recursive = fibonacci_recursive(n)
71 for i, num in enumerate(fib_recursive):
72     print(f"{i}) = {num}")
73
74 # Method 3: Generator
75 print("\n3. Using Generator Method:")
76 fib_gen = fibonacci_generator(n)
77 for i, num in enumerate(fib_gen):
78     print(f"{i}) = {num}")
79
80 if __name__ == "__main__":
81     n = 10
82     print_fibonacci_series(n)
83     fibonacci_recursive(n)
84     fibonacci_generator(n)
85     fibonacci_memoization(n)
```









## RE-PROMPTING TO ADD OTHER FEATURES AND IT'S CODE:

