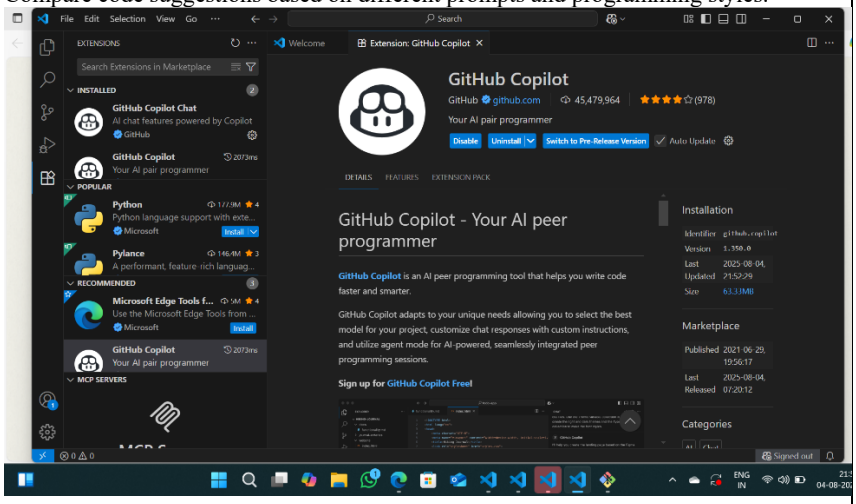
SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
ProgramName:B. Tech		**Assignment Type: Lab**	**AcademicYear:**2025-2026
CourseCoordinatorName	Venkataramana Veeramsetty		
Instructor(s)Name	Dr. V. Venkataramana (Co-ordinator) Dr. T. Sampath Kumar Dr. Pramoda Patro Dr. Brij Kishor Tiwari Dr.J.Ravichander Dr. Mohammand Ali Shaik Dr. Anirodh Kumar Mr. S.Naresh Kumar Dr. RAJESH VELPULA Mr. Kundhan Kumar Ms. Ch.Rajitha Mr. M Prakash Mr. B.Raju Intern 1 (Dharma teja) Intern 2 (Sai Prasad) Intern 3 (Sowmya) NS_2 (Mounika)		
CourseCode	24CS002PC215	**CourseTitle**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment	Week1 - Tuesday	**Time(s)**	
Duration	2 Hours	**Applicableto Batches**	24CSBTB01 To 24CSBTB39
AssignmentNumber:1.2(Present assignment number)/**24**(Total number of assignments)			

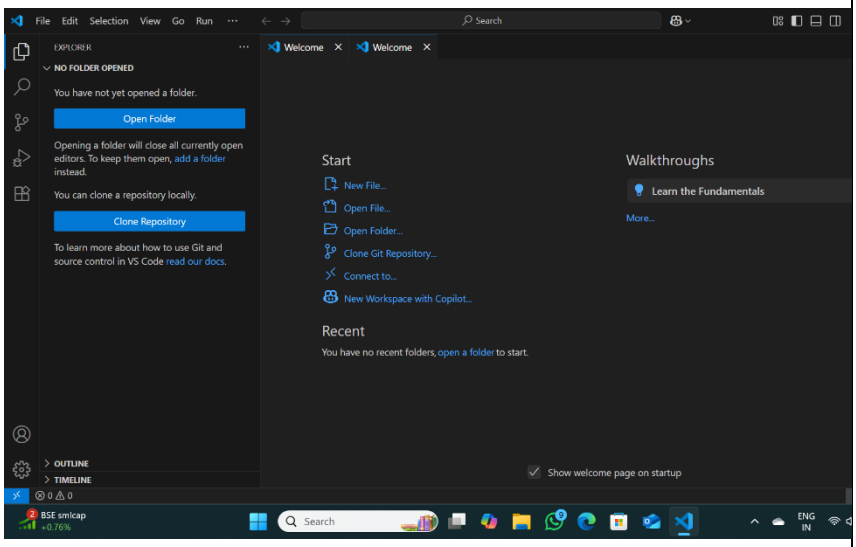
Q.No.	Question	*ExpectedTime to complete*
1	Lab 1: Environment Setup – GitHub Copilot and VS Code Integration **Lab Objectives:** ● To install and configure GitHub Copilot in Visual Studio Code. ● To explore AI-assisted code generation using GitHub Copilot.	Week1 - wednesday

- To analyze the accuracy and effectiveness of Copilot's code suggestions.
- To understand prompt-based programming using comments and code context

Lab Outcomes (LOs):
After completing this lab, students will be able to:

- Set up GitHub Copilot in VS Code successfully.
- Use inline comments and context to generate code with Copilot.
- Evaluate AI-generated code for correctness and readability.
- Compare code suggestions based on different prompts and programming styles.



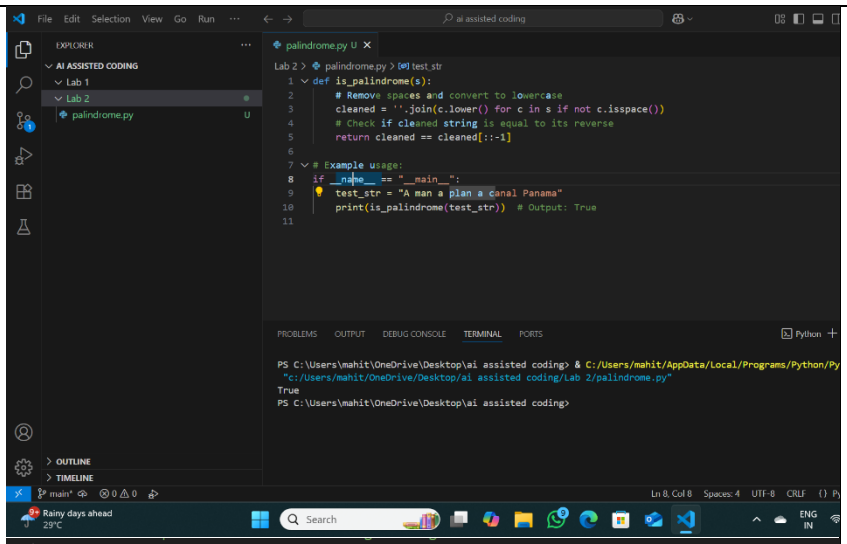


Task Description#1

- Write a comment: # Function to check if a string is a valid palindrome (ignoring spaces and case) and allow Copilot to complete it.

Expected Output#1

- A function that correctly returns True for phrases like "A man a plan a canal Panama"




```
def is_palindrome_alt(s):
    cleaned = ''.join(c.lower() for c in s if not c.isspace())
    left, right = 0, len(cleaned) - 1
    while left < right:
        if cleaned[left] != cleaned[right]:
            return False
        left += 1
        right -= 1
    return True

print(is_palindrome_alt("A man a plan a canal Panama"))  # Output: True
print(is_palindrome_alt("racecar"))  # Output: True
print(is_palindrome_alt("hello world"))  # Output: False
```

Output:

```
PS C:\Users\mahit\OneDrive\Desktop\ai assisted coding\Lab 2> & C:/Users/mahit/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/mahit/OneDrive/Desktop/ai assisted coding/Lab 2/palindrome.py"
True
```

Task Description#2

- Generate a Python function that returns the Fibonacci sequence up to n terms. Prompt with only a function header and docstring

Expected Output#2

- AI completes the function logic using loop or recursion with accurate output

```
#fibonacci sequence using loop
def fibonacci(nterms):
    sequence = []
    a, b = 0, 1
    for _ in range(nterms):
        sequence.append(a)
        a, b = b, a + b
    return sequence

# Example usage:
nterms = int(input("Enter number of terms: "))
sequence = fibonacci(nterms)
print("Loop:", sequence)
```

(variable) sequence: list

```python
# Alternative implementation using recursion
def fibonacci_recursive(n):
    if n <= 1:
        return n
    else:
        return fibonacci_recursive(n-1) + fibonacci_recursive(n-2)

# Generate the sequence using recursion
nterms = int(input("Enter number of terms: "))
sequence_recursive = [fibonacci_recursive(i) for i in range(nterms
print("Recursive:", sequence_recursive)
```

Output:

```
Enter number of terms: 5
Loop: [0, 1, 1, 2, 3]
Enter number of terms: 3
Recursive: [0, 1, 1]
```

Task Description#3

- Write a comment like # Function to reverse a string and use Copilot to generate the function.

Expected Output#3

- Auto-completed reverse function

```python
#Reverse string using slicing
def reverse_string(s):
    return s[::-1]

# Example usage
if __name__ == "__main__":
    input_str = input("Enter a string to reverse: ")
    print("Reversed string:", reverse_string(input_str))

 # Alternative way to reverse a string without using slicing
 def reverse_string_alt(s):
    reversed_str = ""
    for char in s:
        reversed_str = char + reversed_str
    return reversed_str
 input_str = input("Enter a string to reverse (alternative method): ")
 print("Reversed string (alternative method):", reverse_string_alt(input_str
```

Output:

```
PS C:\Users\mahit\OneDrive\Desktop\ai assisted coding\Lab 2> & C:/Users/mahit/AppData/Local/
thon/Python313/python.exe "c:/Users/mahit/OneDrive/Desktop/ai assisted coding/Lab 2/reverse.
Enter a string to reverse: red
Reversed string: der
Enter a string to reverse (alternative method): blue
Reversed string (alternative method): eulb
```

Task Description#4

- Generate a program that simulates a basic calculator (add, subtract, multiply, divide). Write the comment: # Simple calculator with 4 operations and let AI complete it.

Expected Output#4

- Fully working calculator with input/output and operator selection logic

```
def add(x, y):
    return x + y

def subtract(x, y):
    return x - y

def multiply(x, y):
    return x * y

def divide(x, y):
    if y == 0:
        raise ValueError("Cannot divide by zero.")
    return x / y

def main():
    print("Simple Calculator")
    print("Select operation:")
    print("1. Add")
    print("2. Subtract")
    print("3. Multiply")
    print("4. Divide")

    choice = input("Enter choice (1/2/3/4): ")

    if choice not in ('1', '2', '3', '4'):
        print("Invalid input")
        return

    try:
        num1 = float(input("Enter first number: "))
        num2 = float(input("Enter second number: "))
    except ValueError:
        print("Invalid number input.")
        return
```

```
    try:
        num1 = float(input("Enter first number: "))
        num2 = float(input("Enter second number: "))
    except ValueError:
        print("Invalid number input.")
        return

    if choice == '1':
        print(f"{num1} + {num2} = {add(num1, num2)}")
    elif choice == '2':
        print(f"{num1} - {num2} = {subtract(num1, num2)}")
    elif choice == '3':
        print(f"{num1} * {num2} = {multiply(num1, num2)}")
    elif choice == '4':
        try:
            result = divide(num1, num2)
            print(f"{num1} / {num2} = {result}")
        except ValueError as e:
            print(e)

f __name__ == "__main__":
    main()
```

Output:

```
Select operation:
1. Add
2. Subtract
3. Multiply
4. Divide
Enter choice (1/2/3/4): 3
Enter first number: 34
Enter second number: 2
34.0 * 2.0 = 68.0
```

Task Description#5

- Use a comment to instruct AI to write a function that reads a file and returns the number of lines..

Expected Output#5

- Functional implementation using open() or with open() and readlines()

```
#write a code use function that reads a file and returns the number of the line
def read_file(file_path):
    try:
        with open('wsm.txt', 'r') as file:
            lines = file.readlines()
            return len(lines)
    except FileNotFoundError:
        print(f"The file at {file_path} was not found.")
        return 0
    except Exception as e:
        print(f"An error occurred: {e}")
        return 0
print(read_file('wsm.txt'))  # Example usage, replace 'wsm.txt' with your file
```

Output:

```
PS C:\Users\mahit\OneDrive\Desktop\ai assisted coding\Lab 2> & C:/Users/mahit/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/mahit/OneDrive/Desktop/ai assisted coding/Lab 2/linecount.py"
8
```

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots

Evaluation Criteria:

Criteria	Max Marks
Task #1	0.5
Task #2	0.5
Task #3	0.5
Task #4	0.5
Task #5	0.5
Total	**2.5 Marks**