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.	Week1 - wednesday

- To explore AI-assisted code generation using GitHub Copilot.

- 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"

```
[ ] Start coding or generate with AI.

def is_palindrome(s):
    processed_s = "".join(s.split()).lower()
    return processed_s == processed_s[::-1]

string1 = "Race car"
string2 = "hello world"
string3 = "A man a plan a canal Panama"

print(f"'{string1}' is a palindrome: {is_palindrome(string1)}")
print(f"'{string2}' is a palindrome: {is_palindrome(string2)}")
print(f"'{string3}' is a palindrome: {is_palindrome(string3)}")

'Race car' is a palindrome: True
'hello world' is a palindrome: False
'A man a plan a canal Panama' is a palindrome: True
```

PROMPT: write a python program using Function to check if a string is a valid palindrome (ignoring spaces and case)

OSERVATION: Based on the output of the code you ran, it seems like the is palindrome function correctly identifies palindromes even when they have spaces and different capitalization.

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

ommands + Code + Text ▷ Run all

```python
def generate_fibonacci_sequence(n):
  if n <= 0:
    return []
  elif n == 1:
    return [0]
  else:
    sequence = [0, 1]
    while len(sequence) < n:
      next_term = sequence[-1] + sequence[-2]
      sequence.append(next_term)
    return sequence
try:
  num_terms = int(input("Enter the number of Fibonacci terms to generate: "))
  fibonacci_sequence = generate_fibonacci_sequence(num_terms)
  print(f"The Fibonacci sequence up to {num_terms} terms is: {fibonacci_sequence}")
except ValueError:
  print("Invalid input. Please enter an integer.")
```

```
Enter the number of Fibonacci terms to generate: 5
The Fibonacci sequence up to 5 terms is: [0, 1, 1, 2, 3]
```

PROMPT: write a Python program that returns the Fibonacci sequence input from the user.

OBSERVATION: This code takes a number from the user and gives you back the Fibonacci sequence up to that many numbers. It builds the sequence by adding the last two numbers together repeatedly..

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
def reverse_string(s):
  return s[::-1]

user_string = input("Enter a string to reverse: ")
reversed_string = reverse_string(user_string)
print(f"Original string: {user_string}")
print(f"Reversed string: {reversed_string}")
```

```
Enter a string to reverse: hello how are you
Original string: hello how are you
Reversed string: uoy era woh olleh
```

PROMPT: Write a python code for Function to reverse a string from the user

OBSERVATION: This code asks the user to type in some text. It then flips that text around. Finally, it shows you both the original text and the reversed version.

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

```python
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:
    return "Error: Division by zero"
  return x / y

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

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

  if choice in ('1', '2', '3', '4'):
    try:
      num1 = float(input("Enter first number: "))
      num2 = float(input("Enter second number: "))
    except ValueError:
      print("Invalid input. Please enter numbers.")
      continue
```

Variables Terminal

```python
    if choice == '1':
      print(num1, "+", num2, "=", add(num1, num2))

    elif choice == '2':
      print(num1, "-", num2, "=", subtract(num1, num2))

    elif choice == '3':
      print(num1, "*", num2, "=", multiply(num1, num2))

    elif choice == '4':
      print(num1, "/", num2, "=", divide(num1, num2))
    break
  else:
    print("Invalid input.")
```

```
Select operation:
1. Add
2. Subtract
3. Multiply
4. Divide
Enter choice(1/2/3/4): 2
Enter first number: 34
Enter second number: 22
34.0 - 22.0 = 12.0
```

PROMPT: write a python program that simulates a basic and simple calculator with operations (add, subtract, multiply, divide).

OBSERVATION:

- It defines functions for addition, subtraction, multiplication, and division.
- It prompts the user to select an operation.
- It takes two numbers as input from the user.
- It handles potential errors like division by zero and invalid input.
- It performs the selected operation and displays the result.

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()

```python
def count_lines_in_file(filename):
  try:
    with open(filename, 'r') as f:
      line_count = sum(1 for line in f)
    return line_count
  except FileNotFoundError:
    return f"Error: File '{filename}' not found."
  except Exception as e:
    return f"An error occurred: {e}"

# Example usage:
file_name = input("Enter the name of the file: ")
number_of_lines = count_lines_in_file(file_name)

if isinstance(number_of_lines, int):
  print(f"The file '{file_name}' has {number_of_lines} lines.")
else:
  print(number_of_lines)
```

```
Enter the name of the file: my_test_file.txt
The file 'my_test_file.txt' has 3 lines.
```

PROMPT: write a python code using function that reads a file and returns the number of lines..

OBSERVATION: The (count lines in file) function counts lines in a file. It includes robust error handling with try-except. It specifically catches (File Not Found Error) and general exceptions. Using with open ensures proper file closing. It returns the line count or an error message. The example usage prompts for the filename and shows the result.

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**