**NAME:JAITARAM RUTHVIK REDDY ROLL NO:2403A510B5 BATCH:04**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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. * 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"   **Prompt:**  Function to check if a string is a valid palindrome (ignoring spaces and case). It should return True for phrases like "A man a plan a canal: Panama"  **Output:**    **Observation:**  🡪The program defines a function to check whether a given string is a valid palindrome while ignoring spaces and letter case. It processes the input by removing spaces and converting all characters to lowercase, then compares the processed string with its reverse. If both match, the function returns True; otherwise, it returns False. This demonstrates string manipulation, normalization, and conditional logic in Python.      **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   **Prompt:**  Write a Python code to request input from the user for the number of terms (n) and generate the Fibonacci sequence up to n terms. Display the sequence as space-separated numbers.  **Output:**    **Observation:**  🡪The program prompts the user to enter the number of terms (n) and generates the Fibonacci sequence up to n terms. It initializes the first two terms (0 and 1) and calculates each subsequent term by summing the previous two, using either a loop or recursion. The sequence is then displayed in order, demonstrating iterative computation and sequence generation in Python.    **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   **Prompt:**  Reverse an input using dynamic input with a loop" is a prompt that asks you to write code that takes a user's input (a string, for example) and then reverses it using a looping structure.  **Output:**    **Observation:**  🡪The program defines a function to reverse a given string. It takes the input string, applies slicing with a step of -1 (string[::-1]) or uses built-in functions to reverse it, and then returns the reversed string. This implementation showcases Python’s string slicing feature and emphasizes concise, efficient manipulation of text data.  **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   **Prompt:**  Write a python code for simple calculator that performs addition, subtraction, multiplication, and division based on user input.  **Outpu**t:    **Observation:**  **🡪**The program implements a basic calculator that prompts the user to input two numbers and select an arithmetic operation (+, -, \*, /). Based on the operator selected, the corresponding calculation is performed using conditional statements, and the result is displayed. This ensures interactive user input handling and demonstrates fundamental control flow in Python.  **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()   **Prompt:**  Write a Python function that accepts a file path, opens the file using 'with open()',counts the number of lines using readlines(), and returns the count.  **Output:**    **Observation:**  🡪The program defines a function count\_lines\_in\_file(file\_path) that opens a specified file in read mode, reads all lines using readlines(), and returns the total number of lines by applying len() to the list of lines. The function is invoked with "sample.txt" as the argument, and the result is displayed using print(). This implementation efficiently counts lines and ensures proper file handling using the with open() statement, which automatically closes the file after reading.  **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** | | | | | | | Week1 - wednesday |  |