|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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** | | | Week4 - Wednesday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicableto**  **Batches** | |  | | | |
| **AssignmentNumber:7.3**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***ExpectedTime***  ***to complete*** |  |
|  | 1 | Lab 7: AI-Error Debugging with AI: Systematic approaches to finding and fixing bugs  **Lab Objectives:**   * To identify and correct syntax, logic, and runtime errors in Python programs using AI tools. * To understand common programming bugs and AI-assisted debugging suggestions. * To evaluate how AI explains, detects, and fixes different types of coding errors. * To build confidence in using AI to perform structured debugging practices.   **Lab Outcomes (LOs):**  After completing this lab, students will be able to:   * Use AI tools to detect and correct syntax, logic, and runtime errors. * Interpret AI-suggested bug fixes and explanations. * Apply systematic debugging strategies supported by AI-generated insights. * Refactor buggy code using responsible and reliable programming patterns.   **Task Description#1**   * Paste a function with a missing colon (add(a, b)), and let AI fix the syntax error.     Prompt  **Expected Output#1**   * Corrected function with syntax fix   **Prompt:** Correct the python code and allow user to give input to test the code.  **Code:**    **Code Explanation:**   1. Function definition — def add(a,b): defines a function called add that takes two parameters a and b. 2. Return statement — return a+b adds the two numbers together and returns the result. 3. First input — a = int(input("Enter first number: ")) prompts the user to enter a number and stores it as an integer in variable a. 4. Second input — b = int(input("Enter second number: ")) prompts the user to enter another number and stores it as an integer in variable b. 5. Print result — print("Result:", add(a,b)) calls the add function with the two numbers, gets the sum, and prints it on the screen.   **Output:**    **Task Description#2 (Loops)**   * Identify and fix a logic error in a loop that causes infinite iteration.     **Expected Output#2**   * AI fixes increment/decrement error   **Prompt:** Correct the python code and allow user to give input to test the code.  **Code:**    **Code Explanation:**   1. Function definition — def count\_down(n): defines a function called count\_down that takes one parameter n (the starting number). 2. While loop — while n >= 0: repeats the code block as long as n is greater than or equal to 0. 3. Print number — print(n) prints the current value of n to the screen. 4. Decrement — n -= 1 decreases n by 1 each time (the fix from the original bug where it was n += 1). 5. User input — n = int(input("Enter a number to count down from: ")) asks the user to enter a starting number, then calls count\_down(n) to count down from that number to 0.   **Output:**    **Task Description#3**   * Debug a runtime error caused by division by zero. Let AI insert try-except.     **Expected Output#3**   * Corrected function with safe error handling   **Prompt:** Correct the python function and allow the user to give user input to check the code.  **Code:**    **Code Explanation:**   1. Function definition — def divide(a, b): creates a function that takes two numbers: a (numerator) and b (denominator). 2. Zero check — if b == 0: checks if the divisor is zero to prevent a crash. 3. Error message — Returns "Error: Cannot divide by zero" if b is 0 instead of crashing. 4. Safe division — return a / b performs the division only when b is not zero. 5. User input & call — Gets two numbers from the user and calls divide(a, b) to display the result or error message.   **Output:**    **Task Description#4**   * Provide a faulty class definition (missing self in parameters). Let AI fix it     **Expected Output#4**   * Correct \_\_init\_\_() method and explanation   **Prompt:** Correct the errors in the code and allow user to check the code by giving the user input.  **Code:**    **Code Explantion:**   1. Class definition — class Rectangle: creates a blueprint for rectangle objects. 2. Constructor — def \_\_init\_\_(self, length, width): initializes length and width attributes when a Rectangle is created. 3. Area method — area() returns length × width. 4. Perimeter method — perimeter() returns 2 × (length + width). 5. User test — Gets length and width from user input, creates a Rectangle object, and prints area and perimeter.   **Output:**    **Task Description#5**   * Access an invalid list index and use AI to resolve the Index Error.     **Expected Output#5**   * AI suggests checking length or using safe access logic   **Prompt:** Correct the index error in the code by using safe access logic and allow user to check the code by giving the user input.  **Code:**      **Code Explanation:**   1. Defines numbers = [1, 2, 3]. 2. Prompts user for an index with input() and converts to int. 3. Checks if 0 <= index < len(numbers): to ensure index is valid. 4. Prints numbers[index] when valid. 5. Prints an error message showing the list length when invalid   **Output:**    **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** | | --- | --- | | Identification of bugs | 0.5 | | Application of AI-suggested fixes | 0.5 | | Explanation and understanding of errors | 0.5 | | Corrected code functionality | 0.5 | | Report structure and reflection | 0.5 | | **Total** | **2.5 Marks** | | | | | | | Week4 - Wednesday |  |