|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE** | | | | | **DEPARTMENT OF COMPUTER SCIENCE ENGINEERING** | | | | |
| **Program Name:** B. Tech | | | | **Assignment Type: Lab** | | | **Academic Year:**2025-2026 | | |
| **Course Coordinator Name** | | | | 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) | | | | | | |
| **Course Code** | | | 24CS002PC215 | **Course Title** | | AI Assisted Coding | | | |
| **Year/Sem** | | | II/III | **Regulation** | | R24 | | | |
| **Date and Day**  **of Assignment** | | | Week4 - Monday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicable to**  **Batches** | |  | | | |
| **Assignment Number:7.1** (Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***Expected Time***  ***to complete*** |  |
|  | 1 | **Lab 7: 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** (Syntax Errors – Missing Parentheses in Print Statement)  Task: Provide a Python snippet with a missing parenthesis in a print statement (e.g., print "Hello"). Use AI to detect and fix the syntax error.  # Bug: Missing parentheses in print statement  def greet():  print "Hello, AI Debugging Lab!"  greet()  Requirements:   * Run the given code to observe the error. * Apply AI suggestions to correct the syntax. * Use at least 3 assert test cases to confirm the corrected code works.   Expected Output #1:   * Corrected code with proper syntax and AI explanation.   **Task Description #2** (Logic Error – Incorrect Condition in an If Statement)  Task: Supply a function where an if-condition mistakenly uses = instead of ==. Let AI identify and fix the issue.  # Bug: Using assignment (=) instead of comparison (==)  def check\_number(n):  if n = 10:  return "Ten"  else:  return "Not Ten"  Requirements:   * Ask AI to explain why this causes a bug. * Correct the code and verify with 3 assert test cases.   Expected Output #2:   * Corrected code using == with explanation and successful test execution.   **Task Description #3** (Runtime Error – File Not Found)  Task: Provide code that attempts to open a non-existent file and crashes. Use AI to apply safe error handling.  # Bug: Program crashes if file is missing  def read\_file(filename):  with open(filename, 'r') as f:  return f.read()  print(read\_file("nonexistent.txt"))  Requirements:   * Implement a try-except block suggested by AI. * Add a user-friendly error message. * Test with at least 3 scenarios: file exists, file missing, invalid path.   Expected Output #3:   * Safe file handling with exception management.   **Task Description #4** (AttributeError – Calling a Non-Existent Method)  Task: Give a class where a non-existent method is called (e.g., obj.undefined\_method()). Use AI to debug and fix.  # Bug: Calling an undefined method  class Car:  def start(self):  return "Car started"  my\_car = Car()  print(my\_car.drive()) # drive() is not defined  Requirements:   * Students must analyze whether to define the missing method or correct the method call. * Use 3 assert tests to confirm the corrected class works.   Expected Output #4:   * Corrected class with clear AI explanation.   **Task Description #5** (TypeError – Mixing Strings and Integers in Addition)  Task: Provide code that adds an integer and string ("5" + 2) causing a TypeError. Use AI to resolve the bug.  # Bug: TypeError due to mixing string and integer  def add\_five(value):  return value + 5  print(add\_five("10"))  Requirements:   * Ask AI for two solutions: type casting and string concatenation. * Validate with 3 assert test cases.   Expected Output #5:   * Corrected code that runs successfully for multiple inputs.   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 - Monday |  |
|  |  |  | | | | | |  |  |

**AI ASSITED CODE LAB ASSIGMENT 7.1**

**Name:Jarupula rakesh**

**Roll.no:2403a51321**

**#task1:**

**Prompt:**

Provide a Python example with a syntax error caused by missing parentheses in a print statement. Then debug it by adding the correct parentheses, and include at least three assert test cases to verify the fixed code works correctly. Show all code together in one block**.**

**Code:**

**A screen shot of a computer program

AI-generated content may be incorrect.**

**Output:**

**A screen shot of a computer program

AI-generated content may be incorrect.**

**Observation:**

he error happened because the original code used the old Python 2 style print statement without parentheses. In Python 3, print is a function, so it must be written as print("Hello, AI Debugging Lab!"). After fixing the syntax, the program ran correctly, displaying the greeting and passing all three assert tests, which confirmed that the function produced the expected output

**task2:  
prompt:**Provide a Python function with a bug where an if condition mistakenly uses = instead of ==. Explain why this causes an error, fix the code, and include at least three assert test cases. Show all code together with the expected runtime output.

**Code:**

**A computer screen shot of a program code

AI-generated content may be incorrect.**

**Output:**

**A black rectangle with white dots

AI-generated content may be incorrect.**

**Observation:**The function check\_number correctly checks whether the input is equal to 10 using the == operator. When you pass 10, it prints “The number is 10”; for 5 or "abc", it prints “The number is not 10.” All three assert tests confirmed that the function behaves as expected, and the program ends with All tests passed,” proving the logic and comparison are correct**.**

**Task3**

**Prompt:**

Write a Python program to read a file safely using try-except.  
Show friendly error messages if the file is missing, path is invalid, or permission denied.  
Test with three cases: existing file, missing file, and invalid path**.A screenshot of a computer program

AI-generated content may be incorrect.**

**Observation:**

updated code uses a try-except block to handle errors safely when opening files. It stops the program from crashing by showing clear messages if a file is missing, the path is invalid, or there is no permission. Testing with three cases — an existing file, a missing file, and an invalid path — proves that the solution works well in every situation. This makes the program more user-friendly and reliable.

**Task4**

**Code:  
A screenshot of a computer program

AI-generated content may be incorrect.**

**Output:** **A black background with white text

AI-generated content may be incorrect.**

**Observation:**

The error occurred because the program tried to call a method drive() that was not defined in the Car class. To fix this, either the method call should be corrected to an existing method, or the missing method should be added to the class. In the corrected version, the drive() method was added, so both start() and drive() work correctly. Three assert tests confirm that the methods return the expected results and that no errors occur, making the program safe and reliable.

**Task5:**  
prompt:  
rite a Python program to add 5 to a value safely.Handle cases where the input is a string or an integer.Provide two solutions: type casting and string concatenation.  
Include 3 assert tests for each solution to check correctness.

**Code:** **A screenshot of a computer program

AI-generated content may be incorrect.**

Output

**A black background with white text

AI-generated content may be incorrect.**

**Observation**

The error happened because a string and an integer were added.It was fixed using type casting and string concatenation.Multiple assert tests confirmed the code works for different inputs.Now the program runs safely without any TypeErrors.