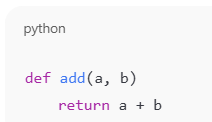
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE** | | | | | **DEPARTMENT OF COMPUTER SCIENCE ENGINEERING** | | | | |
| **ProgramName:**B. Tech | | | | **Assignment Type: Lab** | | | **AcademicYear:**2025-2026 | | |
| **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.     **Expected Output#1**   * Corrected function with syntax fix   **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   **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   **Task Description#4**   * Provide a faulty class definition (missing self in parameters). Let AI fix it     **Expected Output#4**   * Correct \_\_init\_\_() method and explanation   **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   **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 |  |

1. **Task Description#1**

* Paste a function with a missing colon (add(a, b)), and let AI fix the syntax error.



**Expected Output#1**

* Corrected function with syntax fix

**Ans:**

def add(a, b)

    return a + b

**# Fixed function with the missing colon added**

def add(a, b):

    return a + b

result = add(3, 5)

print(result)

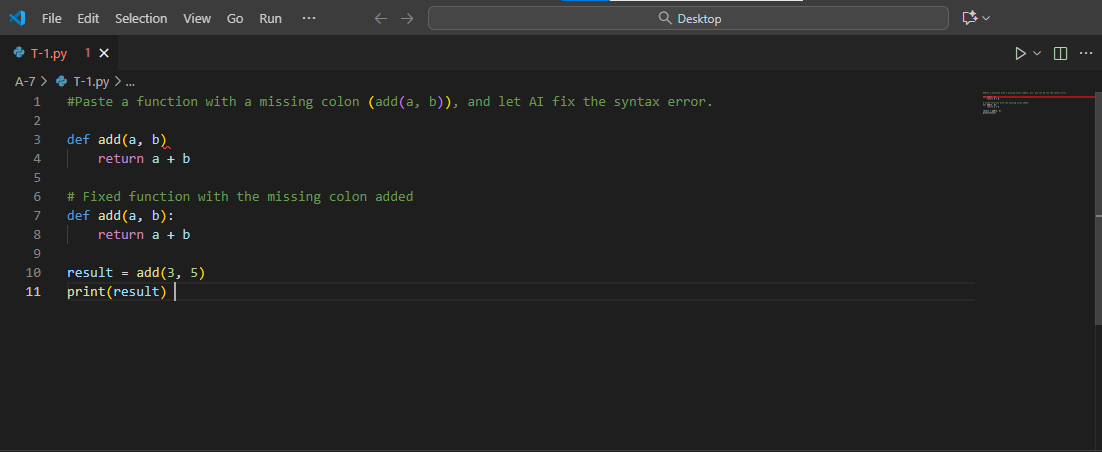


Table 1: AI correcting the syntax error

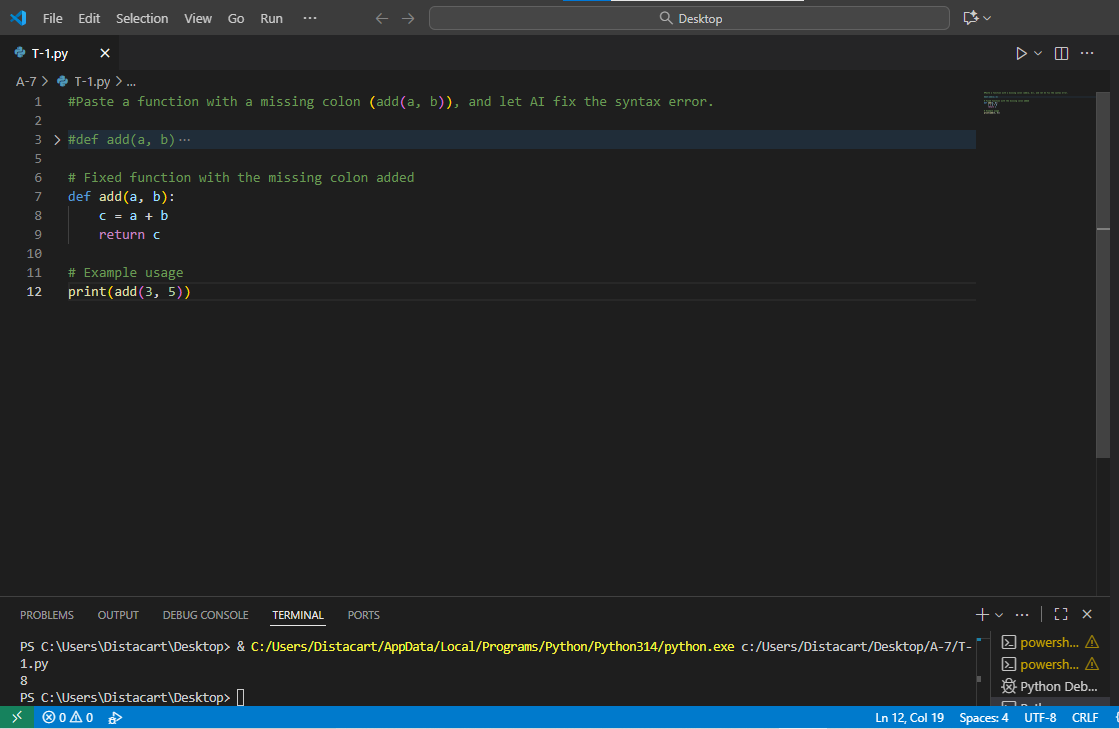
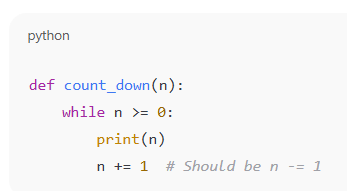


Table 2: Output of the code

1. **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.

Ans: **# Original function with a logic error causing infinite loop**

def count\_up\_to(n):

count = 1

while count <= n:

print(count)

**# Logic error: count is not being incremented, causing infinite loop**

count += 1 **# Fixed by adding this line to increment count**

count\_up\_to(5)

**# Fixed function with the logic error corrected**

def count\_up\_to(n):

count = 1

while count <= n:

print(count)

count += 1 **# Increment count to avoid infinite loop**

count\_up\_to(5)

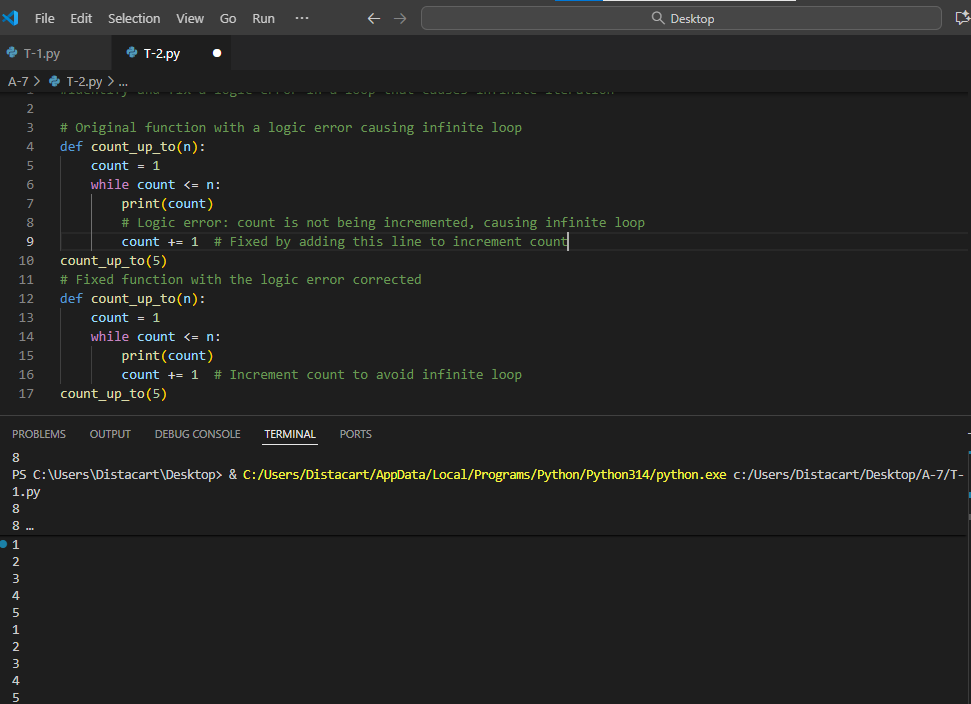
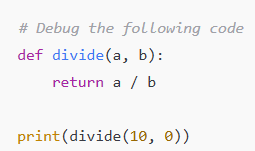


Table 3: corrected code from the Github copilot

1. **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

Ans: **# Original function that may cause division by zero error**

def divide(a, b):

return a / b

**# Fixed function with try-except to handle division by zero**

def divide(a, b):

try:

result = a / b

except ZeroDivisionError:

return "Error: Division by zero is not allowed."

return result

# Example usage

print(divide(10, 2))

print(divide(10, 0))

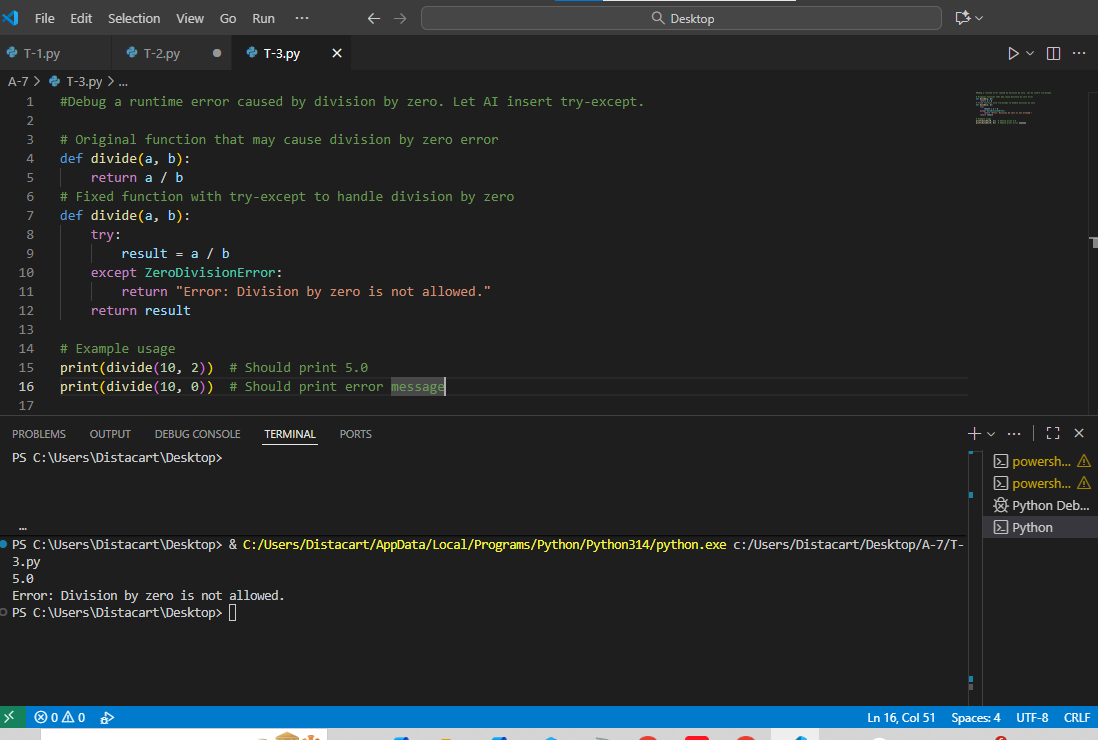
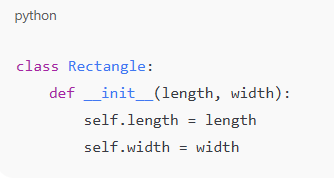


Table 4: Division by zero code and ouptut using AI.

1. Task-Depriction

**Task Description#4**

* Provide a faulty class definition (missing self in parameters). Let AI fix it

****

**Expected Output#4**

* Correct \_\_init\_\_() method and explanation

Ans:

class Person:

def \_\_init\_\_(name, age): **# Logic error: 'self' parameter is missing**

name.name = name

name.age = age

**# Fixed class definition with 'self' parameter added to \_\_init\_\_ method**

class Person:

def \_\_init\_\_(self, name, age): **# Fixed by adding 'self' parameter**

self.name = name

self.age = age

person = Person("Alice", 30)

print(person.name) # Should print "Alice"

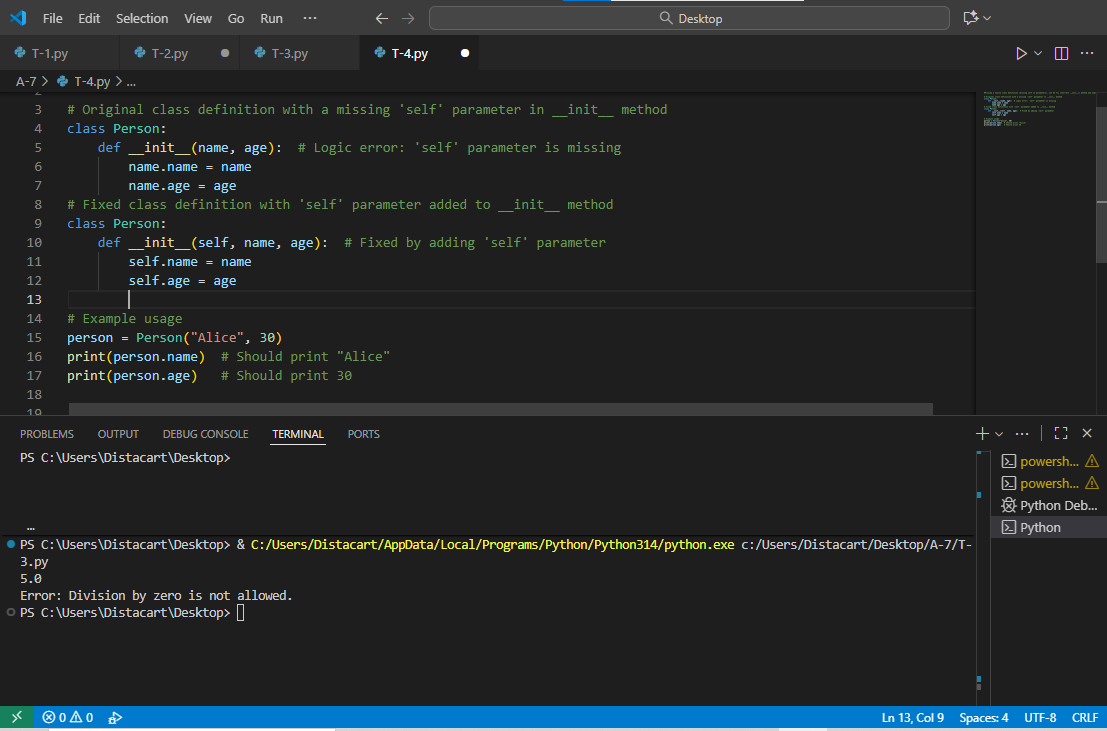
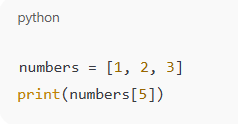


Table 5: AI fixed the code with error message

1. **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.

Ans:

my\_list = [10, 20, 30, 40, 50]

try:

**# Attempting to access an index that is out of range**

print(my\_list[10])

except IndexError as e:

print("IndexError encountered:", e)

print("The index you are trying to access is out of range.")

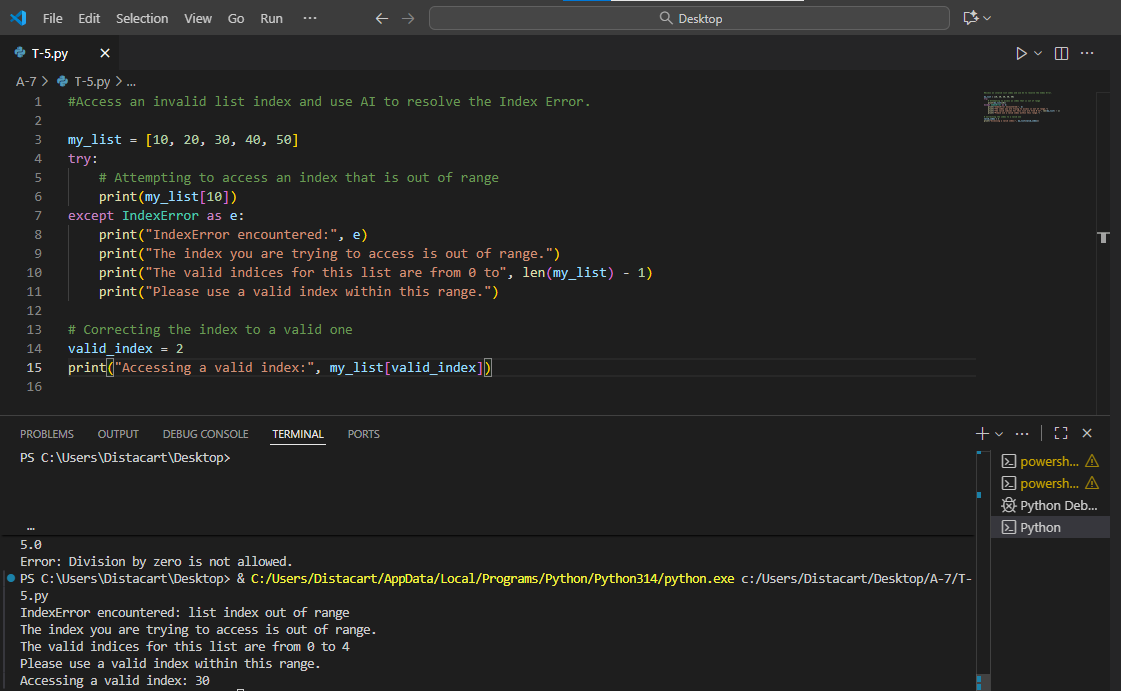
print("The valid indices for this list are from 0 to", len(my\_list) - 1)

print("Please use a valid index within this range.")

**# Correcting the index to a valid one**

valid\_index = 2

print("Accessing a valid index:", my\_list[valid\_index])



**Table 6: Task-6 code and output**