

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		Expected Time to complete
1	Lab 7: AI-Error Debugging with AI: Systematic approaches to finding and		Week4 - Wednesday

	<p>fixing bugs</p> <p>Lab Objectives:</p> <ul style="list-style-type: none"> • 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. <p>Lab Outcomes (LOs):</p> <p>After completing this lab, students will be able to:</p> <ul style="list-style-type: none"> • 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. <p>Task Description#1</p> <ul style="list-style-type: none"> • Paste a function with a missing colon (add(a, b)), and let AI fix the syntax error. <pre style="background-color: #f0f0f0; padding: 10px;">python def add(a, b) return a + b</pre> <p>Expected Output#1</p> <ul style="list-style-type: none"> • Corrected function with syntax fix <p>Task Description#2 (Loops)</p> <ul style="list-style-type: none"> • Identify and fix a logic error in a loop that causes infinite iteration. 	
--	---	--

```
python

def count_down(n):
    while n >= 0:
        print(n)
        n += 1 # Should be n -= 1
```

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.

```
# Debug the following code
def divide(a, b):
    return a / b

print(divide(10, 0))
```

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

```
python

class Rectangle:
    def __init__(length, width):
        self.length = length
        self.width = width
```

Expected Output#4

- Correct __init__() method and explanation

Task Description#5

- Access an invalid list index and use AI to resolve the Index Error.

```
python
```

```
numbers = [1, 2, 3]
print(numbers[5])
```

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

Task Description#1

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

The screenshot shows a code editor window with a dark theme. At the top, there are tabs for "TASK 4.py lab 6 assessment", "TASK 5.py lab 6 assessment", and "TASK 1.py lab 7 assessment". The "TASK 1.py" tab is active. The code in the editor is:

```
python

def add(a, b)
    return a + b

lab 7 assessment > TASK 1.py > ...
1  def add(a, b):
2  |     return a + b
3
4  # Taking only user inputs
5  a = int(input("Enter value for a: "))
6  b = int(input("Enter value for b: "))
7
8  # Displaying sum
9  result = add(a, b)
10 print("The sum is:", result)
```

The line "def add(a, b)" is highlighted in red, indicating a syntax error.

Expected Output#1

- Corrected function with syntax fix

Practical output:

The screenshot shows a terminal window with a dark theme. The title bar says "TERMINAL". The command entered is "python assessment> python -u "c:\Users\rimsha\OneDrive\Desktop\rimsha python assessment\tempCodeRunnerFile.python"" followed by "Enter value for a: 6", "Enter value for b: 4", and "The sum is: 10". The prompt at the end is "PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment>".

```
> < TERMINAL
PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment> python -u "c:\Users\rimsha\OneD
● rive\Desktop\rimsha python assessment\tempCodeRunnerFile.python"
Enter value for a: 6
Enter value for b: 4
The sum is: 10
PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment>
```

Task Description#2 (Loops)

- Identify and fix a logic error in a loop that causes infinite iteration.

The screenshot shows a code editor interface with a dark theme. At the top, there is a search bar containing the text "rimsha python assessment". Below the search bar, there are three tabs: "TASK 5.py lab 6 assessment", "TASK 1.py lab 7 assessment", and "TASK 2.py lab 7 assessment" (which is currently active). The main workspace displays the following Python code:

```
w Go ... ← → Q rimsha python assessment
TASK 5.py lab 6 assessment TASK 1.py lab 7 assessment TASK 2.py lab 7 assessment ...
lab 7 assessment > TASK 2.py > ...
1 def count_down(n):
2     while n >= 0:
3         print(n)
4         n -= 1 # Correct decrement
5
6 # Taking input from user
7 num = int(input("Enter a number to countdown from: "))
8 count_down(num)
```

Expected Output#2

- AI fixes increment/decrement error

Practical output:

```
● PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment> python -u "c:\Users\rimsha\OneDrive\Desktop\rimsha python assessment\tempCodeRunnerFile.py"
Enter a number to countdown from: 8
8
7
6
5
4
3
2
1
0
◆ PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment>
```

Task Description#3

- Debug a runtime error caused by division by zero. Let AI insert try-except.

```
# Debug the following code
def divide(a, b):
    return a / b

print(divide(10, 0))
```

```
lab 7 assessment > TASK 3.py > ...
1 def divide(a, b):
2     try:
3         return a / b
4     except ZeroDivisionError:
5         return "Error: division by zero is not allowed."
6
7 # Example 1
8 print(divide(10, 0)) # division by zero case
9
10 # Example 2
11 print(divide(20, 5)) # valid division
```

Expected Output#3

- Corrected function with safe error handling

Practical output:

```
● PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment> python -u "c:\Users\rimsha\OneDrive\Desktop\rimsha python assessment\tempCodeRunnerFile.py"
Error: division by zero is not allowed.
4.0
❖ PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment>
```

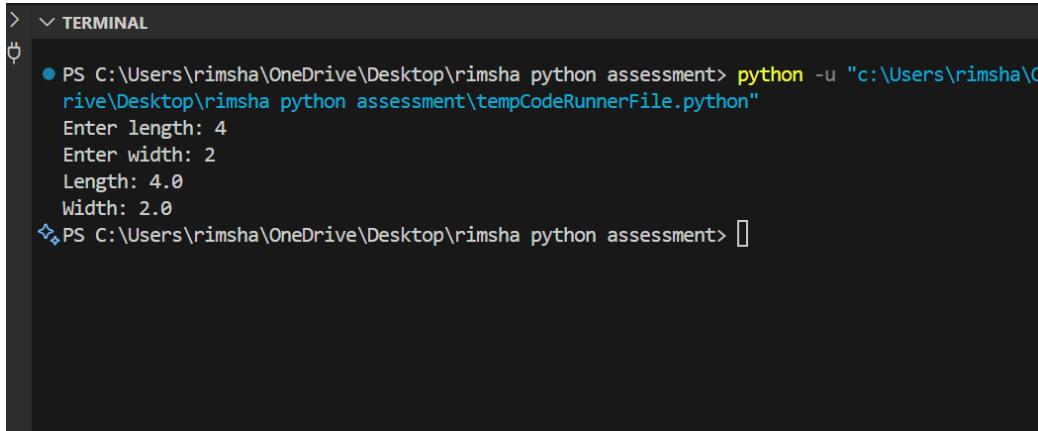
Task Description#4

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

Expected Output#4

- Correct `__init__()` method and explanation

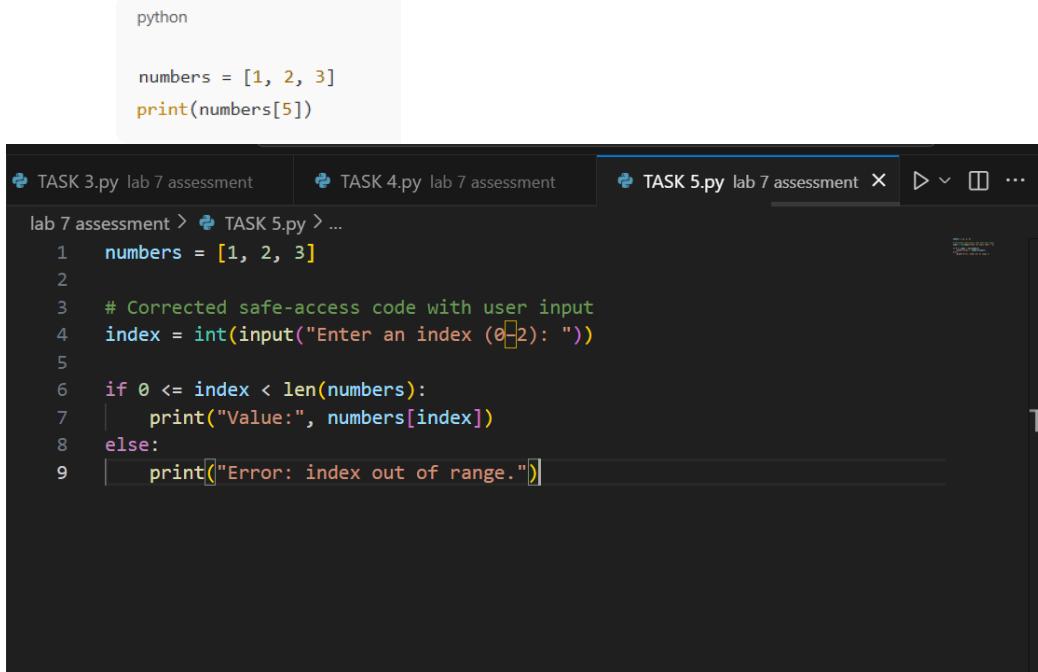
Practical output:



```
> ▾ TERMINAL
❯ PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment> python -u "c:\Users\rimsha\OneDrive\Desktop\rimsha python assessment\tempCodeRunnerFile.py"
Enter length: 4
Enter width: 2
Length: 4.0
Width: 2.0
PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment> []
```

Task Description#5

- Access an invalid list index and use AI to resolve the Index Error.



python

```
numbers = [1, 2, 3]
print(numbers[5])
```

TASK 3.py lab 7 assessment

TASK 4.py lab 7 assessment

TASK 5.py lab 7 assessment

```
lab 7 assessment > TASK 5.py > ...
1 numbers = [1, 2, 3]
2
3 # Corrected safe-access code with user input
4 index = int(input("Enter an index (0-2): "))
5
6 if 0 <= index < len(numbers):
7     print("Value:", numbers[index])
8 else:
9     print("Error: index out of range.")
```

Expected Output#5

AI suggests checking length or using safe access logic

Practical output:

```
width: 2.0
● PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment> python -u "c:\Users\rimsha\OneD
rive\Desktop\rimsha python assessment\tempCodeRunnerFile.python"
Enter an index (0-2): 1
Value: 2
● PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment> python -u "c:\Users\rimsha\OneD
rive\Desktop\rimsha python assessment\tempCodeRunnerFile.python"
Enter an index (0-2): 0
Value: 1
● PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment> python -u "c:\Users\rimsha\OneD
rive\Desktop\rimsha python assessment\tempCodeRunnerFile.python"
Enter an index (0-2): 2
Value: 3
❖ PS C:\Users\rimsha\OneDrive\Desktop\rimsha python assessment> █
```