

Name:D.Ashmitha

Enrollno:2403a510g5

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	Applicable to Batches	
AssignmentNumber:7.3(Present assignment number)/24(Total number of assignments)			
Q.No.	Question	Expected Time to complete	
1	Lab 6: AI-Based Code Completion – Classes, Loops, and Conditionals <b>Lab Objectives:</b> <ul style="list-style-type: none"> <li>To identify and correct syntax, logic, and runtime errors in Python programs using AI</li> </ul>	Week4 - Wednesday	

	<div>tools.</div> <div><ul style="list-style-type: none"><li>• To understand common programming bugs and AI-assisted debugging suggestions.</li><li>• To evaluate how AI explains, detects, and fixes different types of coding errors.</li><li>• To build confidence in using AI to perform structured debugging practices.</li></ul></div> <div><b>Lab Outcomes (LOs):</b> After completing this lab, students will be able to:</div> <div><ul style="list-style-type: none"><li>• Use AI tools to detect and correct syntax, logic, and runtime errors.</li><li>• Interpret AI-suggested bug fixes and explanations.</li><li>• Apply systematic debugging strategies supported by AI-generated insights.</li><li>• Refactor buggy code using responsible and reliable programming patterns.</li></ul></div> <div><b>Task Description#1</b><ul style="list-style-type: none"><li>• Paste a function with a missing colon (add(a, b)), and let AI fix the syntax error.</li></ul></div> <div><pre>python  def add(a, b)     return a + b</pre></div> <div>CODE:</div> <div><pre>def add(a, b):     return a + b  print(add(3, 5))</pre></div> <div><b>FIX OF ERROR:</b> It must end with a colon (:).</div> <div><b>Expected Output#1</b><ul style="list-style-type: none"><li>• Corrected function with syntax fix</li></ul></div> <div><pre>8 ~~~</pre><div></div></div> <div><b>Task Description#2 (Loops)</b><ul style="list-style-type: none"><li>• Identify and fix a logic error in a loop that causes infinite iteration.</li></ul></div>	
--	--	--

python

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

**CODE:**

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

**FIXERROR:**

**The loop condition is while n >= 0.**

**Inside the loop, n += 1 makes n increase forever, so it never becomes less than 0 → infinite loop**

**Expected Output#2**

- AI fixes increment/decrement error

```
5  
4  
3  
2  
1  
0
```

**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))
```

**CODE:**

```
def divide(a, b):
    try:
        return a / b
    except ZeroDivisionError:
        return "Error: Division by zero is not allowed"

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

#### FIXERROR:

This will raise a `ZeroDivisionError` at runtime because `b = 0`.  
In Python, division by zero is not allowed

#### Expected Output#3

- Corrected function with safe error handling

```
Error: Division by zero is not allowed
```

#### 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
```

#### CODE:

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

#### FIXERROR:

In Python, instance methods (including `__init__`) must include `self` as the first parameter. Without `self`, Python doesn't know which object's attributes (`length` and `width`) to assign.

#### Expected Output#4

- Correct `__init__()` method and explanation

```
rect = Rectangle(10, 5)
print(rect.length)  # Output: 10
print(rect.width)   # Output: 5
```

#### Task Description#5

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

```
python

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

#### CODE:

##### Fix 1: Check index before accessing

```
numbers = [1, 2, 3]
index = 5

if index < len(numbers):
    print(numbers[index])
else:
    print("Index out of range")
```

##### Fix 2: Use try-except

```
numbers = [1, 2, 3]

try:
    print(numbers[5])
except IndexError:
    print("Error: Invalid index")
```

##### Fix 3: Safe access with default value

```
numbers = [1, 2, 3]
index = 5

value = numbers[index] if index < len(numbers) else None
print(value)    # Output: None
```

**FIXERROR:**

**IndexError: list index out of range**

**Expected Output#5**

- AI suggests checking length or using safe access logic

**Fix 1: Check index before accessing:**

Index out of range

Fix 2: Use try-except:

Error: Invalid index

Fix 3: Safe access with default value:

None

**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
<b>Total</b>	<b>2.5 Marks</b>