

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: B. Tech		Assignment Type: Lab	
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/I	Regulation	R24
Date and Day of Assignment	Week4 - Wednesday	BATCH:06	NAME: J.ABHIRAM H.NO:2403A51342
Duration	2 Hours	Applicable to Batches	
Assignment Number: 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	Week4 - Wednesday

	<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): 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 (<code>add(a, b)</code>), and let AI fix the syntax error. <pre>python def add(a, b) return a + b</pre> <p>CODE:</p> <pre>def add(a, b): return a + b print(add(3, 5))</pre> <p>FIX OF ERROR: It must end with a colon (:).</p> <p>Output:</p> <ul style="list-style-type: none"> • Corrected function with syntax fix <pre>8 ~~~ ✓</pre> <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
```

CODE:

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

FIXERROR:

The loop condition is while $n \geq 0$.

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

Output:

```
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

Output:

- 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.

Output:

- 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
Total	2.5 Marks