

AI ASSISTED CODING LAB-7.3

J.Sai Kumar || Batch:-09 || 2303A51562

Task 1: Fixing Syntax Errors.

The screenshot shows a code editor window with the following content:

```
Lab-7.3.py > ...
1 #Task 1: Fixing Syntax Errors.
2 #You are reviewing a Python program where a basic function definition contains a syntax error.
3 #Here is the original code with the syntax error:
4 def add(a, b):
5     return a + b
6
7 print(add(5, 3))
8
9 #The syntax error occurred because the function definition was missing a colon (:) after the parameter list (a, b).
10 #In Python:
11
12 #A colon is mandatory after function definitions.
13
14 #It indicates the start of the function body.
15
16 #Without the colon, Python cannot understand where the function block begins, which results in a SyntaxError.
17
18 #After adding the colon, the program executes successfully.
19
20
```

Below the code editor, the following tabs are visible: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, GITLENS. The TERMINAL tab is currently selected.

In the terminal pane, there are two entries:

- PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING> & C:/Users/saiku/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/saiku/OneDrive/Desktop/AI ASSISTANT CODING/Lab-7.3.py"
- PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING> []

Task 2: Debugging Logic Errors in Loops.

The screenshot shows a code editor window titled "Lab-7.3.py" with the following content:

```
1 #Task 2: Debugging Logic Errors in Loops.
2 #You are debugging a loop that runs infinitely due to a logical mistake.
3 #Here is the give code.
4
5 def count_down(n):
6     while n >= 0:
7         print(n)
8         n += 1
9
10 #Here is the corrected code:
11
12 def count_down(n):
13     while n >= 0:
14         print(n)
15         n -= 1
16 count_down(3)
17
18 #Behavior Analysis:
19 #Initially, n = 3
20 #After each iteration, n decreases by 1
21 #When n becomes -1, the condition n >= 0 becomes False
22 #The loop stops correctly
23 #Infinite iteration issue is resolved
24
```

Below the code editor is a terminal window showing the output of running the script:

```
PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING> & C:/Users/saiku/AppData/Local/Programs/Python/Python314/python Lab-7.3.py
3
2
1
0
```

Task 3: Handling Runtime Errors (Division by Zero).

```
Lab-7.3.py > ...
Lab-7.3.py > ...
1 #Task 3: Handling Runtime Errors (Division by Zero).
2 #A Python function crashes during execution due to a division by zero error.
3 #Here is corrected code with error handling for division by zero:
4 def divide(a, b):
5     try:
6         return a / b
7     except ZeroDivisionError:
8         return "Error: Cannot divide by zero"
9
10 print(divide(10, 0))
11
12 #How the AI Fixed It:
13 #The risky statement a / b is placed inside a try block.
14
15 #If division by zero occurs, Python raises ZeroDivisionError.
16
17 #The except block catches the error.
18
19 #Instead of crashing, the function returns a user-friendly message.
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

● PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING> & C:/Users/saiku/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/saiku/OneDrive/Desktop/AI ASSISTANT CODING/Lab-7.3.py"
Error: Cannot divide by zero
○ PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING>

Task 4: Debugging Class Definition Errors.

```
Lab-7.3.py > ...
Lab-7.3.py > ...
1 #Task 4: Debugging Class Definition Errors.
2 #The following Python class gives an error when creating an object.
3 #Identify the issue in the constructor and correct the code.
4 #Also explain why the fix is required.
5 class Rectangle:
6     def __init__(self, length, width):
7         self.length = length
8         self.width = width
9
10    def display(self):
11        print("Length:", self.length)
12        print("Width:", self.width)
13
14    # Creating object
15    r1 = Rectangle(10, 5)
16    r1.display()
17
18 #AI Explanation of the Object-Oriented Error:
19 #self represents the current instance (object) of the class.
20 #It allows access to instance variables and methods.
21 #When an object is created, Python automatically passes the object as the first argument.
22 #If self is missing, Python throws a TypeError.
23 #Without self, the object cannot store or access its own data.
24
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

● PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING> & C:/Users/saiku/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/saiku/OneDrive/Desktop/AI ASSISTANT CODING/Lab-7.3.py"
Length: 10
Width: 5
○ PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING>

Task 5: Resolving Index Errors in Lists.

The screenshot shows a code editor interface with a tab labeled "Lab-7.3.py" containing the following Python code:

```
1 #Task 5: Resolving Index Errors in Lists.
2 #The following Python code crashes when executed.
3 #Identify the error and modify the code to safely access the list.
4 numbers = [1, 2, 3]
5 index = 5
6
7 if index < len(numbers):
8     print(numbers[index])
9 else:
10    print("Error: Index out of range")
11
12 #AI Explanation of the Error Handling
13
14 #Lists in Python use zero-based indexing.
15
16 #If an index greater than len(list) - 1 is accessed, Python raises an IndexError.
17
18 #Using bounds checking ensures the index is valid before accessing.
19
20 #Using try-except prevents the program from crashing.
```

Below the code editor, the terminal window displays the following output:

```
PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING> & C:/Users/saiku/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/saiku/OneDrive/Desktop/AI ASSISTANT CODING/Lab-7.3.py"
● Error: Index out of range
○ PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING>
```