

LAB ASSIGNMENT-7.3

2303A51337

BATCH-10

➤ **TASK-1:**

PROMPT:

I have a Python function with a syntax error. Detect the issue and correct it.

```
def add(a, b)
```

```
return a + b
```

CODE:

```
def add(a, b):
```

```
return a + b
```

```
print(add(5, 3))
```

OUTPUT:

The screenshot shows a Visual Studio Code (VS Code) interface. The top bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and a search bar labeled 'AI Assisted'. The Explorer sidebar on the left lists files: AI ASSISTED, lab1.4.py, lab1.5.py, lab1.6.py, lab1.7.py, lab1.8.py, lab1.9.py, lab2.3.py, lab2.4.py, lab2.5.py, lab2.6.py, lab2.7.py, lab2.8.py, lab3.3.py, lab3.4.py, lab3.5.py, lab3.6.py, lab3.7.py, lab4.1.py, lab4.2.py, lab4.3.py, lab4.4.py, lab4.5.py, lab4.6.py, lab4.7.py, lab5.1.py, lab5.2.py, lab5.3.py, lab5.4.py, lab6.1.py, lab6.2.py, lab6.3.py, lab6.4.py, lab6.5.py, lab6.6.py, lab6.7.py, lab6.8.py, lab6.9.py, lab7.3.py, and lab7.4.py. The main editor area displays the following Python code:

```
#!/usr/bin/python
# I have a Python function with a syntax error. Detect the issue and correct it.
def add(a, b):
    #return a + b
    return a + b
print(add(5, 3))
```

The terminal at the bottom shows the command run: PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted> & C:/Users/sriva/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/sriva/OneDrive/Desktop/3-2 academics/AI Assisted/lab7.3.py". The status bar indicates the file is 20°C.

EXPLANATION:

The error is because the function definition is missing a colon and is commented out.

It should be written as `def add(a, b):`. After fixing it, the function runs correctly and returns 8 for `add(5, 3)`.

■ TASK-2

PROMPT:

This loop runs infinitely. Identify the logical mistake and fix it.

CODE:

```
def count_down(n):
```

```
    while n >= 0:
```

```
        print(n)
```

```
        n -= 1
```

```
count_down(5)
```

OUTPUT:

The screenshot shows a code editor interface with several tabs open in the background. The active tab is 'lab7.3.py' which contains the following code:

```
def add(a, b):
    return a + b
print(add(5, 3))
#TASK-2:This loop runs infinitely. Identify the logical mistake and fix it.
def count_down(n):
    while n >= 0:
        print(n)
        n -= 1
count_down(5)
```

Below the code editor, the terminal window shows the command being run and the resulting error message:

```
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted & C:/Users/sriva/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/sriva/OneDrive/Desktop/3-2 academics/AI Assisted/lab7.3.py"
Error: Invalid index resolved
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted & C:/Users/sriva/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/sriva/OneDrive/Desktop/3-2 academics/AI Assisted/lab7.3.py"
9
4
3
2
1
0
```

EXPLANATION:

The loop runs infinitely because the value of *i* is never changed inside the loop, so the condition *i* \leq 5 is always true. To fix this, *i* must be incremented inside the loop using *i* += 1. After this correction, the loop prints numbers from 1 to 5 and then stops.

❖ TASK-3:

PROMPT:

This Python function crashes with a runtime error. Identify the issue and add proper error handling.

```
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))
print(divide(10, 2))
```

OUTPUT:

The screenshot shows the Visual Studio Code interface. The Explorer sidebar lists several Python files. The main editor window contains the provided code for Task-3. The terminal at the bottom shows the execution of the code and the resulting error message.

```
#TASK-2:This loop runs infinitely. Identify the logical mistake and fix it.
#TASK-3:This Python function crashes with a runtime error. Identify the issue and add proper error handling.

def divide(a, b):
    try:
        return a / b
    except ZeroDivisionError:
        return "Error: Division by zero is not allowed"
print(divide(10, 0))
print(divide(10, 2))

Error: Division by zero is not allowed
PS C:\Users\sriya\OneDrive\Desktop\3-2 academic\AI Assisted> 
```

EXPLANATION:

The error occurs because dividing by zero causes a `ZeroDivisionError`. Using `try` and `except` prevents the program from crashing by catching this error and returning a message instead. When the divisor is not zero, the function works normally and returns the correct result.

TASK-4:

PROMPT:

This Python class constructor is incorrect. Identify the problem and correct it.

```
class Student: def __init__(name, age) name = name age = age
```

CODE:

```
class Student:
```

```
    def __init__(self, name, age):
```

```
        self.name = name
```

```
        self.age = age
```

```
s = Student("Srivarsha", 20)
```

```
print(s.name)
```

```
print(s.age)
```

OUTPUT:

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER:** Shows files like lab1.4.py, lab6.4.py, etc., and an AI Assisted folder containing lab7.3.py.
- OPEN EDITORS:** Displays the code for `lab7.3.py`. The code contains several syntax errors:
 - Line 23: `def divide(a, b):`
 - Line 24: `try:`
 - Line 25: `return a / b`
 - Line 26: `except ZeroDivisionError:`
 - Line 27: `return "Error: Division by zero is not allowed"`
 - Line 28: `print(divide(10, 0))`
 - Line 29: `print(divide(10, 2))`
 - Line 30: `if __name__ == "__main__":`
 - Line 31: `#TASK-4: This Python class constructor is incorrect. Identify the problem and correct it.`
 - Line 32: `#class Student:`
 - Line 33: `def __init__(name, age):`
 - Line 34: `# name = name`
 - Line 35: `# age = age`
 - Line 36: `class Student:`
 - Line 37: `def __init__(self, name, age):`
 - Line 38: `self.name = name`
 - Line 39: `self.age = age`
 - Line 40: `s = Student("Srivarsha", 20)`
 - Line 41: `print(s.name)`
 - Line 42: `print(s.age)`
 - Line 43: `if __name__ == "__main__":`
 - Line 44: `s = Student("Srivarsha", 20)`
 - Line 45: `print(s.name)`
 - Line 46: `print(s.age)`
 - Line 47: `if __name__ == "__main__":`
 - Line 48: `s = Student("Srivarsha", 20)`
- PROBLEMS:** Shows an error: `Division by zero is not allowed`.
- TERMINAL:** Shows the command line output:

```
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted & C:/Users/sriva/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/sriva/OneDrive/Desktop/3-2 academics/AI Assisted/lab7.3.py"
5.0
Srivarsha
20
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted>
```
- OUTPUT:** Shows the output of the code execution.
- SIDE BAR:** Shows multiple Python environments.

EXPLANATION:

The constructor is incorrect because it does not use `self`, so the variables `name` and `age` are not stored in the object. `self` is required to refer to the current object, and attributes must be assigned as `self.name` and `self.age`. After fixing this, the constructor correctly initializes the student's name and age, and the object can access them properly.

❖ Task-5

PROMPT:

This code throws an IndexError. Identify the problem and suggest a safe way to access list elements.

```
#numbers = [10, 20, 30]  
#print(numbers[5])
```

CODE:

```
numbers = [10, 20, 30]
```

try:

```
print(numbers[5])
```

```
except IndexError:
```

```
print("Error: Invalid index resolved")
```

OUTPUT:

The screenshot shows the Visual Studio Code interface with the 'AI Assisted' extension active. The top bar has tabs for 'File', 'Edit', 'Selection', 'View', 'Go', 'Run', 'Terminal', 'Help', and 'AI Assisted'. The 'AI ASSISTED' tab is selected, showing a list of files: 'lab1.4.py', 'lab1.py', 'lab1.4.py', 'lab5.4.py', 'lab6.py', 'lab5.4.py', 'lab7.3.py', 'lab6.3.py', 'lab4.5.py', and 'lab5.3.py'. The main editor area displays 'lab7.3.py' with the following code:

```
35     # name = name
36     # age = age
37     ...
38     class Student:
39         def __init__(self, name, age):
40             self.name = name
41             self.age = age
42
43 s = Student("SriVarsha", 20)
44 print(s.name)
45 print(s.age)
46
47 #TASK-5: This code throws an IndexError. Identify the problem and suggest a safe way to access list elements.
48 numbers = [10, 20, 30]
49 #print(numbers[5])
50
51 numbers = [10, 20, 30]
52 try:
53     print(numbers[5])
54 except IndexError:
55     print("Error: Invalid index resolved")
```

The bottom status bar shows the terminal output:

```
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted> & C:/Users/sriva/AppData/Local/Programs/Python/Python313/python.exe "C:/Users/sriva/OneDrive/Desktop/3-2 academics/AI Assisted/lab7.3.py"
Error: Invalid index resolved
PS C:\Users\sriva\OneDrive\Desktop\3-2 academics\AI Assisted> []
```

On the right side, there is a sidebar with several Python-related icons and a list of 'Python' items.

EXPLANATION:

The error occurs because the list has only 3 elements (indexes 0, 1, and 2), but the code tries to access index 5, which does not exist. This causes an `IndexError`. Using a `try-except` block safely handles this situation by catching the error and printing a message instead of crashing the program.