

AI ASSISTED CODING

LAB-7.5

B.Gayathri

2303A51717

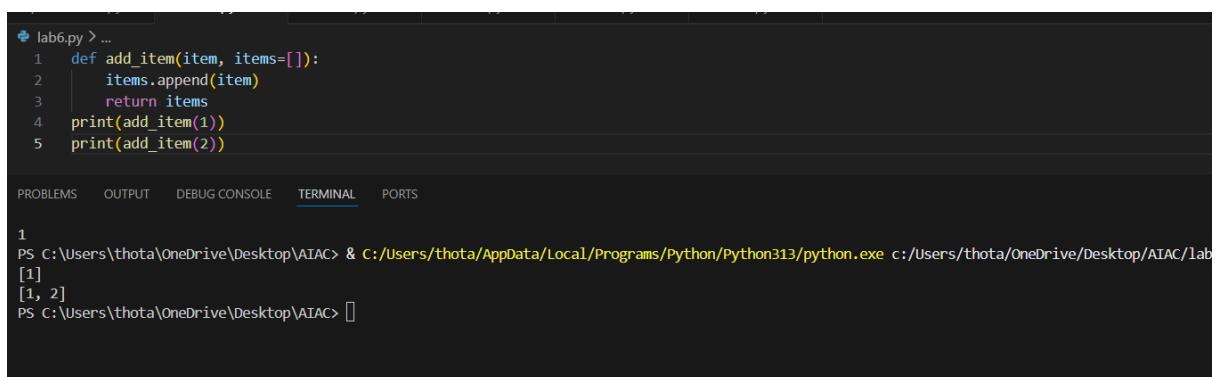
BATCH-11

Task 1 (Mutable Default Argument – Function Bug)

Task: Analyze given code where a mutable default argument causes unexpected behavior. Use AI to fix it.

```
# Bug: Mutable default argument
def add_item(item, items=[]):
    items.append(item)
    return items
print(add_item(1))
print(add_item(2))
```

Expected Output: Corrected function avoids shared list bug.



The screenshot shows a terminal window with the following content:

```
lab6.py > ...
1 def add_item(item, items=[]):
2     items.append(item)
3     return items
4 print(add_item(1))
5 print(add_item(2))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

1 PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab
[1]
[1, 2]
PS C:\Users\thota\OneDrive\Desktop\AIAC> []
```

Task 2 (Floating-Point Precision Error)

Task: Analyze given code where floating-point comparison fails.

Use AI to correct with tolerance.

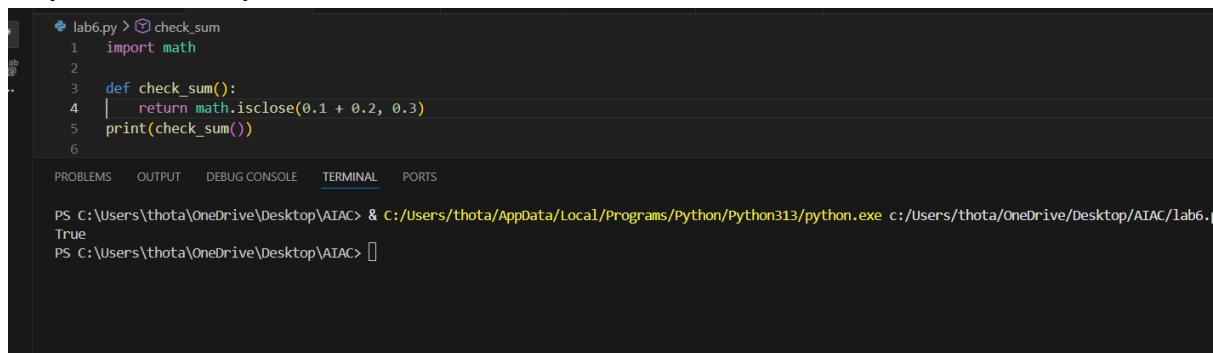
```
# Bug: Floating point precision issue

def check_sum():

    return (0.1 + 0.2) == 0.3

print(check_sum())
```

Expected Output: Corrected function



A screenshot of a terminal window. The code in the editor is:

```
lab6.py > check_sum
1 import math
2
3 def check_sum():
4     return math.isclose(0.1 + 0.2, 0.3)
5 print(check_sum())
6
```

The terminal output shows the command being run and the result:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab6.py
True
PS C:\Users\thota\OneDrive\Desktop\AIAC> []
```

Task 3 (Recursion Error – Missing Base Case)

Task: Analyze given code where recursion runs infinitely due to missing base case. Use AI to fix.

```
# Bug: No base case
```

```
def countdown(n):

    print(n)

    return countdown(n-1)

countdown(5)
```

Expected Output : Correct recursion with stopping condition

The screenshot shows a code editor with multiple tabs at the top: palindrome.py, lab6.py, lab4.py, lab2.py, lab5.py, and lab1.py. The lab6.py tab is active, displaying the following Python code:

```
def countdown(n):
    print(n)
    if n == 0:
        return
    countdown(n-1)
countdown(5)
```

Below the code editor is a terminal window with the following output:

```
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab6.py
5
4
3
2
1
0
PS C:\Users\thota\OneDrive\Desktop\AIAC>
```

Task 4 (Dictionary Key Error)

Task: Analyze given code where a missing dictionary key causes error. Use AI to fix it.

```
# Bug: Accessing non-existing key

def get_value():

    data = {"a": 1, "b": 2}

    return data["c"]

print(get_value())
```

Expected Output: Corrected with .get() or error handling.

The screenshot shows a code editor with multiple tabs at the top: palindrome.py, lab6.py, lab4.py, lab2.py, lab5.py, and lab1.py. The lab6.py tab is active, displaying the following Python code:

```
def get_value():
    data = {"a": 1, "b": 2}
    return data.get("c", "Key not found")
print(get_value())
```

Below the code editor is a terminal window with the following output:

```
PS C:\Users\thota\OneDrive\Desktop\AIAC> ^C
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab6.py
Key not found
PS C:\Users\thota\OneDrive\Desktop\AIAC>
```

Task 5 (Infinite Loop – Wrong Condition)

Task: Analyze given code where loop never ends. Use AI to detect and fix it.

```
# Bug: Infinite loop

def loop_example():

    i = 0

    while i < 5:

        print(i)
```

Expected Output: Corrected loop increments i.

The screenshot shows the VS Code interface. The top bar has tabs for 'palindrome.py', 'lab6.py' (which is active), 'lab4.py', 'lab2.py', 'lab5.py', and 'lab1.py'. Below the tabs, the code editor shows the following Python script:

```
lab6.py > ...
1 def loop_example():
2     i = 0
3     while i < 5:
4         print(i)
5         i += 1
6 loop_example()
```

Below the code editor is a terminal window with the following output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\thota\OneDrive\Desktop\AIAC> ^C
0
1
2
3
4
PS C:\Users\thota\OneDrive\Desktop\AIAC> []
```

Task 6 (Unpacking Error – Wrong Variables)

Task: Analyze given code where tuple unpacking fails. Use AI to fix it.

```
# Bug: Wrong unpacking

a, b = (1, 2, 3)
```

Expected Output: Correct unpacking or using _ for extra values.

A screenshot of a terminal window from a code editor. The code in the editor is:

```
lab6.py > ...
1 a, b, _ = (1, 2, 3)
```

The terminal output shows the command being run and the result:

```
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab6.py
PS C:\Users\thota\OneDrive\Desktop\AIAC> []
```

Task 7 (Mixed Indentation – Tabs vs Spaces)

Task: Analyze given code where mixed indentation breaks execution. Use AI to fix it.

Bug: Mixed indentation

```
def func():
```

```
    x = 5
```

```
    y = 10
```

```
    return x+y
```

Expected Output : Consistent indentation applied.

A screenshot of a terminal window from a code editor. The code in the editor is:

```
lab6.py > ...
1 def func():
2     x = 5
3     y = 10
4     return x+y
5 print(func())
```

The terminal output shows the command being run and the result:

```
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab6.py
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab6.py
15
PS C:\Users\thota\OneDrive\Desktop\AIAC> []
```

Task 8 (Import Error – Wrong Module Usage)

Task: Analyze given code with incorrect import. Use AI to fix.

Bug: Wrong import

```
import maths
```

```
print(maths.sqrt(16))
```

Expected Output: Corrected to import math

The screenshot shows a code editor window with a dark theme. In the top-left corner, there is a file icon followed by the text "lab6.py". Below this, two lines of Python code are displayed:

```
1 import math
2 print(math.sqrt(16))
```

At the bottom of the editor, there is a terminal window showing the following command-line interface session:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\thota\OneDrive\Desktop\AIAC> & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab6.py
PS C:\Users\thota\OneDrive\Desktop\AIAC> ^C ...
● PS C:\Users\thota\OneDrive\Desktop\AIAC & C:/Users/thota/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thota/OneDrive/Desktop/AIAC/lab6.py
4.0
○ PS C:\Users\thota\OneDrive\Desktop\AIAC> []
```