ASSIGNMENT-7.5

Name: B.Bhargava Chary

HT NO: 2303A51747

Batch: 24

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

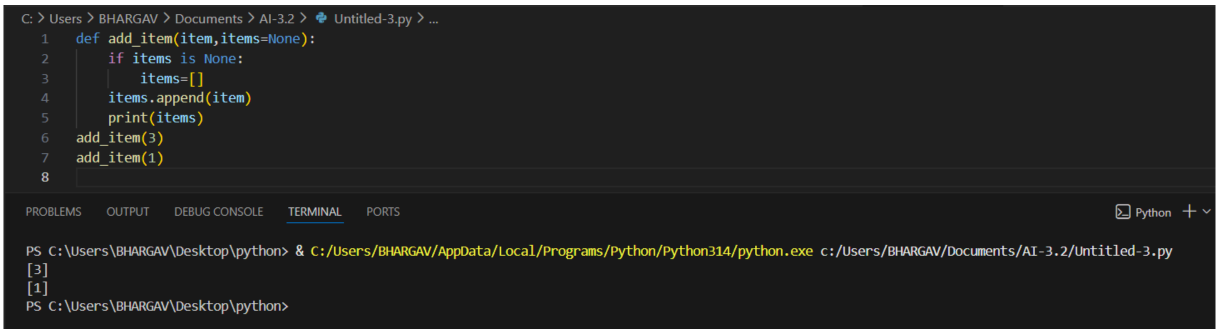
add_item(item, items=[]):

items.append(item) return

items print(add_item(1))

print(add_item(2))

Expected Output: Corrected function avoids shared list bug.



C: > Users > BHARGAV > Documents > AI-3.2 > Untitled-3.py > ...

```python
def add_item(item,items=None):
    if items is None:
        items=[]
    items.append(item)
    print(items)
add_item(3)
add_item(1)

```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe c:/Users/BHARGAV/Documents/AI-3.2/Untitled-3.py
[3]
[1]
PS C:\Users\BHARGAV\Desktop\python>
```

Task 2 (Floating-Point Precision Error)

Task: Analyze given code where floating-point comparison fails. Use AI to correct with

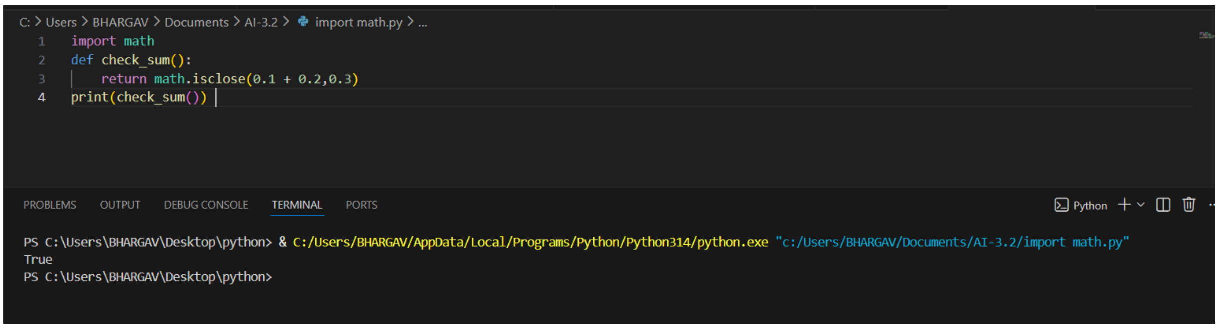
tolerance.

# Bug: Floating point precision issue def

check_sum(): return (0.1 + 0.2) == 0.3

print(check_sum())

Expected Output: Corrected function



C: > Users > BHARGAV > Documents > AI-3.2 > import math.py > ...

```python
import math
def check_sum():
    return math.isclose(0.1 + 0.2,0.3)
print(check_sum())
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/import math.py"
True
PS C:\Users\BHARGAV\Desktop\python>
```

Task 3 (Recursion Error – Missing Base Case)

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

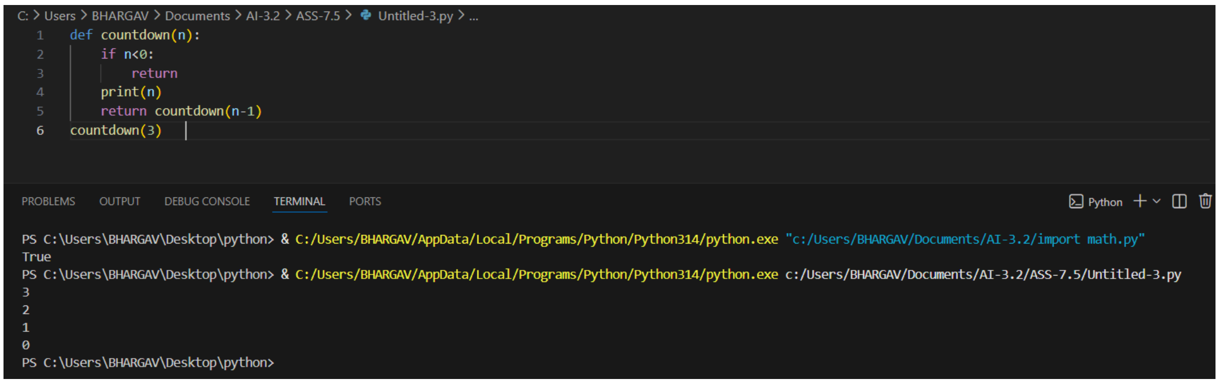
# Bug: No base case def

countdown(n): print(n)

return

countdown(n-1) countdown(5)

Expected Output : Correct recursion with stopping condition.



C: > Users > BHARGAV > Documents > AI-3.2 > ASS-7.5 > Untitled-3.py > ...

```python
def countdown(n):
    if n<0:
        return
    print(n)
    return countdown(n-1)
countdown(3)
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/import math.py"
True
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe c:/Users/BHARGAV/Documents/AI-3.2/ASS-7.5/Untitled-3.py
3
2
1
0
PS C:\Users\BHARGAV\Desktop\python>
```

Task 4 (Dictionary Key Error)

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

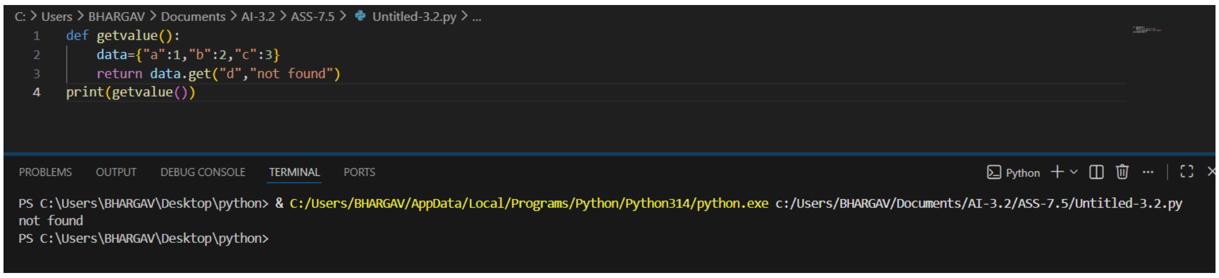
# Bug: Accessing non-existing key

def get_value(): data = {"a": 1, "b":

2} return data["c"]

print(get_value())

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


C: > Users > BHARGAV > Documents > AI-3.2 > ASS-7.5 > Untitled-3.2.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Python

```python
def getvalue():
    data={"a":1,"b":2,"c":3}
    return data.get("d","not found")
print(getvalue())
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe c:/Users/BHARGAV/Documents/AI-3.2/ASS-7.5/Untitled-3.2.py
not found
PS C:\Users\BHARGAV\Desktop\python>
```

Task 5 (Infinite Loop – Wrong Condition)

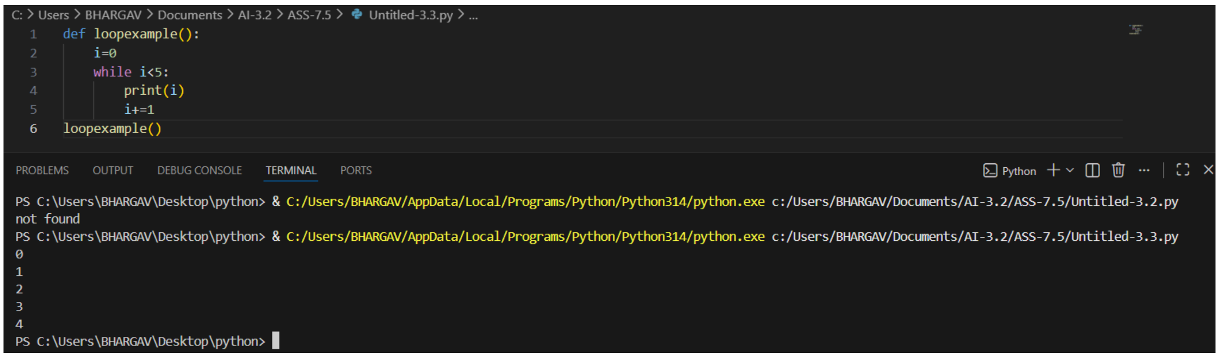
Task: Analyze given code where loop never ends. Use AI to detect and fix it.

# Bug: Infinite loop def loop_example():

i = 0 while i

< 5: print(i)

Expected Output: Corrected loop increments i.


C: > Users > BHARGAV > Documents > AI-3.2 > ASS-7.5 > Untitled-3.3.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Python

```python
def loopexample():
    i=0
    while i<5:
        print(i)
        i+=1
loopexample()
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe c:/Users/BHARGAV/Documents/AI-3.2/ASS-7.5/Untitled-3.2.py
not found
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe c:/Users/BHARGAV/Documents/AI-3.2/ASS-7.5/Untitled-3.3.py
0
1
2
3
4
PS C:\Users\BHARGAV\Desktop\python>
```

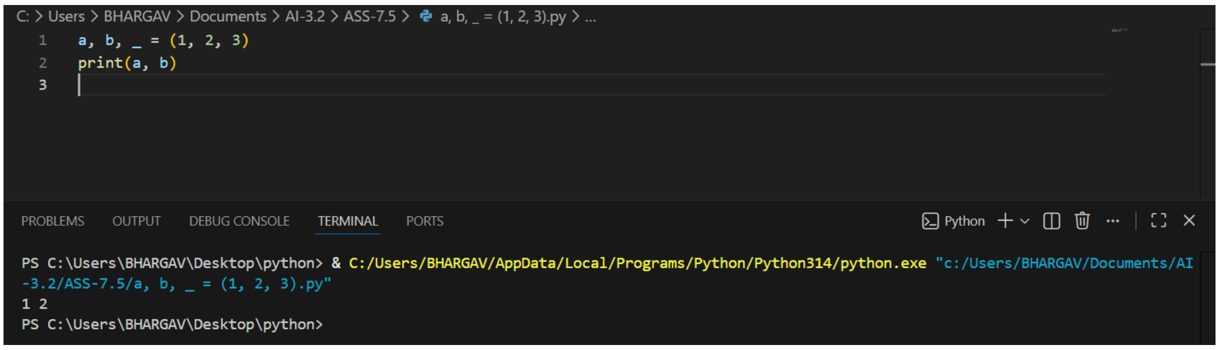
Task 6 (Unpacking Error – Wrong Variables)

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.


C: > Users > BHARGAV > Documents > AI-3.2 > ASS-7.5 > a, b, _ = (1, 2, 3).py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Python

```python
a, b, _ = (1, 2, 3)
print(a, b)
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI
-3.2/ASS-7.5/a, b, _ = (1, 2, 3).py"
1 2
PS C:\Users\BHARGAV\Desktop\python>
```

Task 7 (Mixed Indentation – Tabs vs Spaces)

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

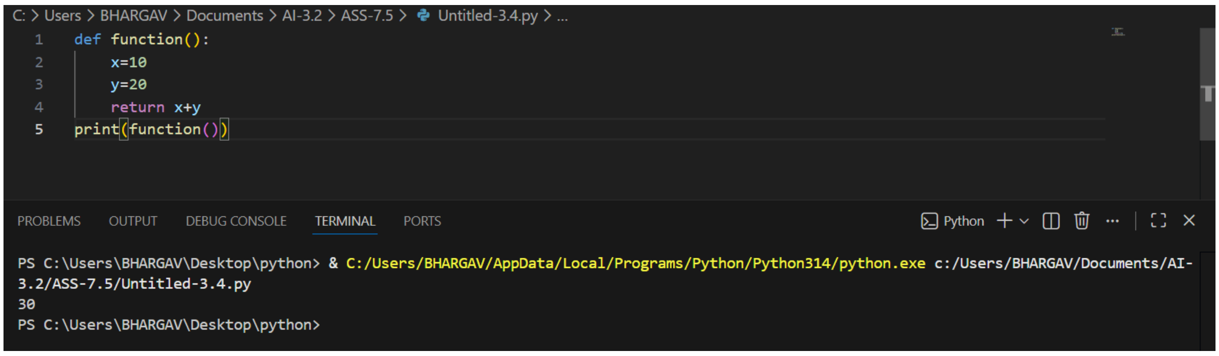
# Bug: Mixed indentation

def func():x = 5 y =

10 return

x+y

Expected Output : Consistent indentation applied.



C: > Users > BHARGAV > Documents > AI-3.2 > ASS-7.5 > Untitled-3.4.py > ...

```
def function():
    x=10
    y=20
    return x+y
print(function())
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe c:/Users/BHARGAV/Documents/AI-3.2/ASS-7.5/Untitled-3.4.py
30
PS C:\Users\BHARGAV\Desktop\python>
```

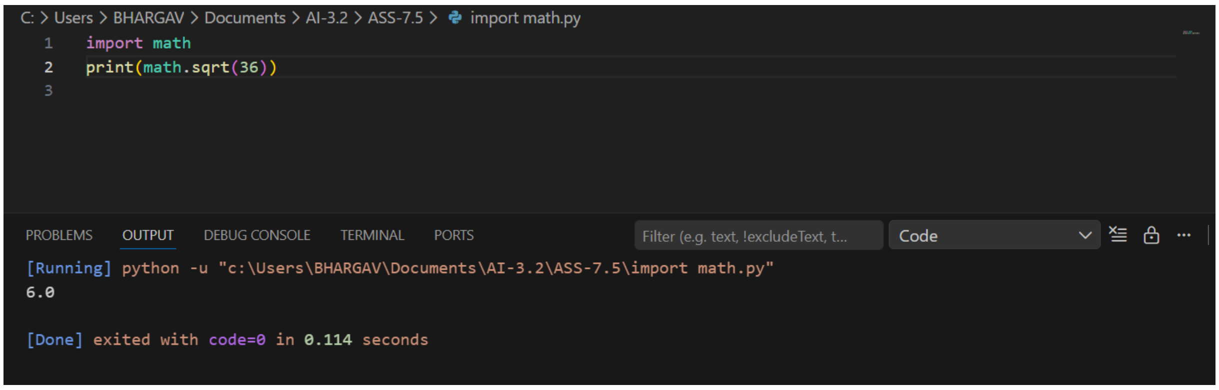
Task 8 (Import Error – Wrong Module Usage)

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



C: > Users > BHARGAV > Documents > AI-3.2 > ASS-7.5 > import math.py

```
import math
print(math.sqrt(36))

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[Running] python -u "c:\Users\BHARGAV\Documents\AI-3.2\ASS-7.5\import math.py"
6.0

[Done] exited with code=0 in 0.114 seconds
```

Task 9 (Unreachable Code – Return Inside Loop)

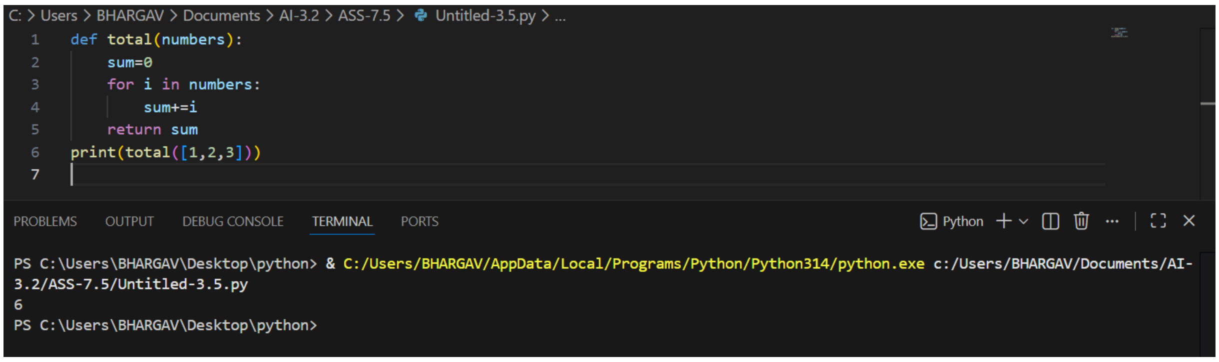
Task: Analyze given code where a return inside a loop prevents full iteration. Use AI to fix it.

# Bug: Early return inside loop def

total(numbers): for n in numbers:

return n print(total([1,2,3]))

Expected Output: Corrected code accumulates sum and returns after loop.


C: > Users > BHARGAV > Documents > AI-3.2 > ASS-7.5 > Untitled-3.5.py > ...

```python
def total(numbers):
    sum=0
    for i in numbers:
        sum+=i
    return sum
print(total([1,2,3]))
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe c:/Users/BHARGAV/Documents/AI-3.2/ASS-7.5/Untitled-3.5.py
6
PS C:\Users\BHARGAV\Desktop\python>
```

Task 10 (Name Error – Undefined Variable)

Task: Analyze given code where a variable is used before being defined. Let AI detect and fix the error.

# Bug: Using undefined variable

def calculate_area(): return length

* width print(calculate_area())

Requirements:

• Run the code to observe the error.

• Ask AI to identify the missing variable definition.

• Fix the bug by defining length and width as parameters.

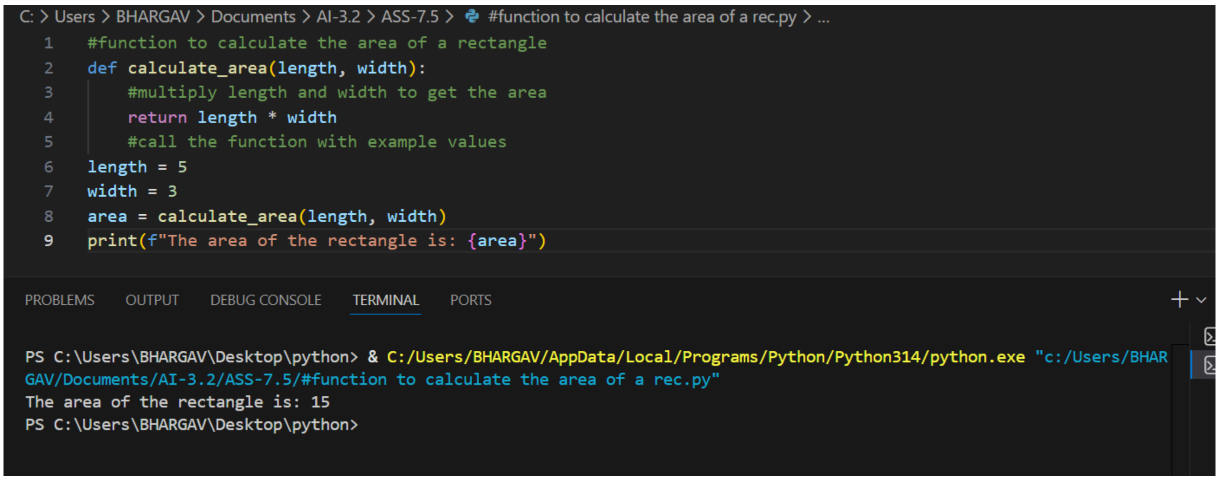
• Add 3 assert test cases for correctness.

Expected Output :

• Corrected code with parameters.

• AI explanation of the bug.

Successful execution of assertions.


C: > Users > BHARGAV > Documents > AI-3.2 > ASS-7.5 > #function to calculate the area of a rec.py > ...

```python
#function to calculate the area of a rectangle
def calculate_area(length, width):
    #multiply length and width to get the area
    return length * width
    #call the function with example values
length = 5
width = 3
area = calculate_area(length, width)
print(f"The area of the rectangle is: {area}")
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/AI-3.2/ASS-7.5/#function to calculate the area of a rec.py"
The area of the rectangle is: 15
PS C:\Users\BHARGAV\Desktop\python>
```

Task 11 (Type Error – Mixing Data Types Incorrectly)

Task: Analyze given code where integers and strings are added incorrectly. Let AI detect and fix the error.

# Bug: Adding integer and string def

add_values(): return 5 +

"10" print(add_values())

Requirements:• Run the code to observe the error.

• AI should explain why int + str is invalid.

• Fix the code by type conversion (e.g., int("10") or str(5)).

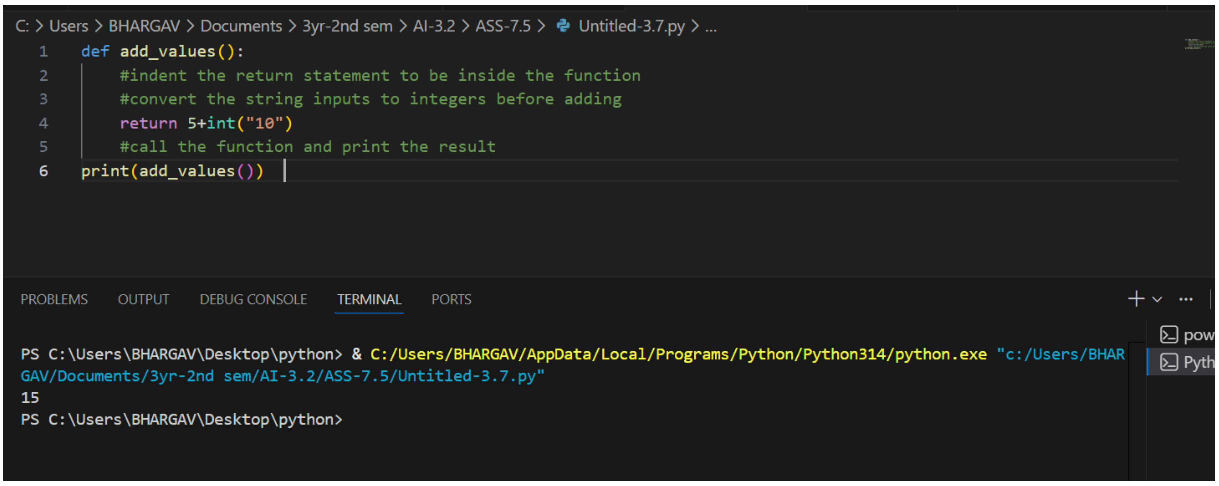
• Verify with 3 assert cases.

Expected Output #6:

• Corrected code with type handling.

• AI explanation of the fix.

Successful test validation.


C: > Users > BHARGAV > Documents > 3yr-2nd sem > AI-3.2 > ASS-7.5 > Untitled-3.7.py > ...

```python
def add_values():
    #indent the return statement to be inside the function
    #convert the string inputs to integers before adding
    return 5+int("10")
    #call the function and print the result
print(add_values())
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/3yr-2nd sem/AI-3.2/ASS-7.5/Untitled-3.7.py"
15
PS C:\Users\BHARGAV\Desktop\python>
```

Task 12 (Type Error – String + List Concatenation)

Task: Analyze code where a string is incorrectly added to a list.

# Bug: Adding string and list def

combine(): return "Numbers: "

+ [1, 2, 3] print(combine())

Requirements:

• Run the code to observe the error.

• Explain why str + list is invalid.

• Fix using conversion (str([1,2,3]) or " ".join()).

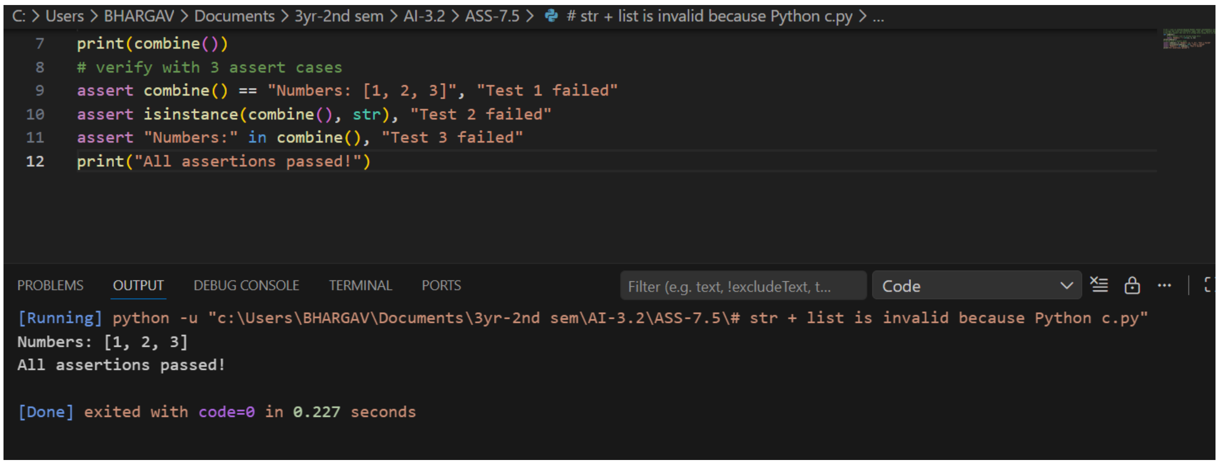
• Verify with 3 assert cases.

Expected Output:

• Corrected code

• Explanation

• Successful test validation



C: > Users > BHARGAV > Documents > 3yr-2nd sem > AI-3.2 > ASS-7.5 > # str + list is invalid because Python c.py > ...

```python
print(combine())
# verify with 3 assert cases
assert combine() == "Numbers: [1, 2, 3]", "Test 1 failed"
assert isinstance(combine(), str), "Test 2 failed"
assert "Numbers:" in combine(), "Test 3 failed"
print("All assertions passed!")
```

```
[Running] python -u "c:\Users\BHARGAV\Documents\3yr-2nd sem\AI-3.2\ASS-7.5\# str + list is invalid because Python c.py"
Numbers: [1, 2, 3]
All assertions passed!

[Done] exited with code=0 in 0.227 seconds
```

Task 13 (Type Error – Multiplying String by Float)

Task: Detect and fix code where a string is multiplied by a float.

# Bug: Multiplying string by float def

repeat_text(): return "Hello"

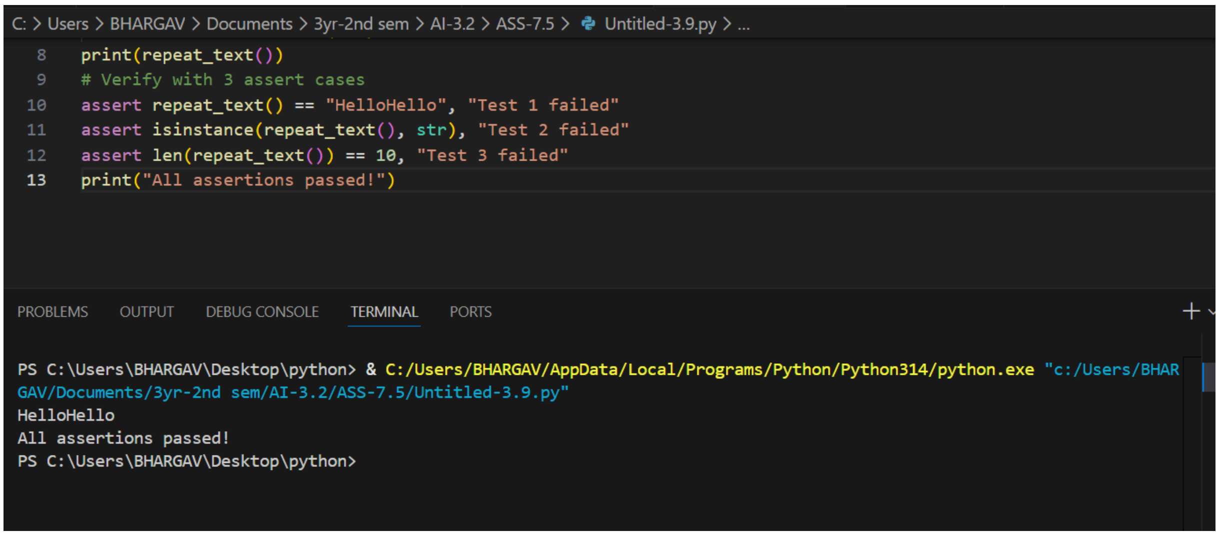
* 2.5 print(repeat_text())

Requirements: • Observe the error.

• Explain why float multiplication is invalid for strings.

• Fix by converting float to int.

• Add 3 assert test cases



C: > Users > BHARGAV > Documents > 3yr-2nd sem > AI-3.2 > ASS-7.5 > Untitled-3.9.py > ...

```python
print(repeat_text())
# Verify with 3 assert cases
assert repeat_text() == "HelloHello", "Test 1 failed"
assert isinstance(repeat_text(), str), "Test 2 failed"
assert len(repeat_text()) == 10, "Test 3 failed"
print("All assertions passed!")
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHAR
GAV/Documents/3yr-2nd sem/AI-3.2/ASS-7.5/Untitled-3.9.py"
HelloHello
All assertions passed!
PS C:\Users\BHARGAV\Desktop\python>
```

Task 14 (Type Error – Adding None to Integer)

Task: Analyze code where None is added to an integer.

# Bug: Adding None and integer def

compute(): value = Nonereturn value + 10

print(compute())

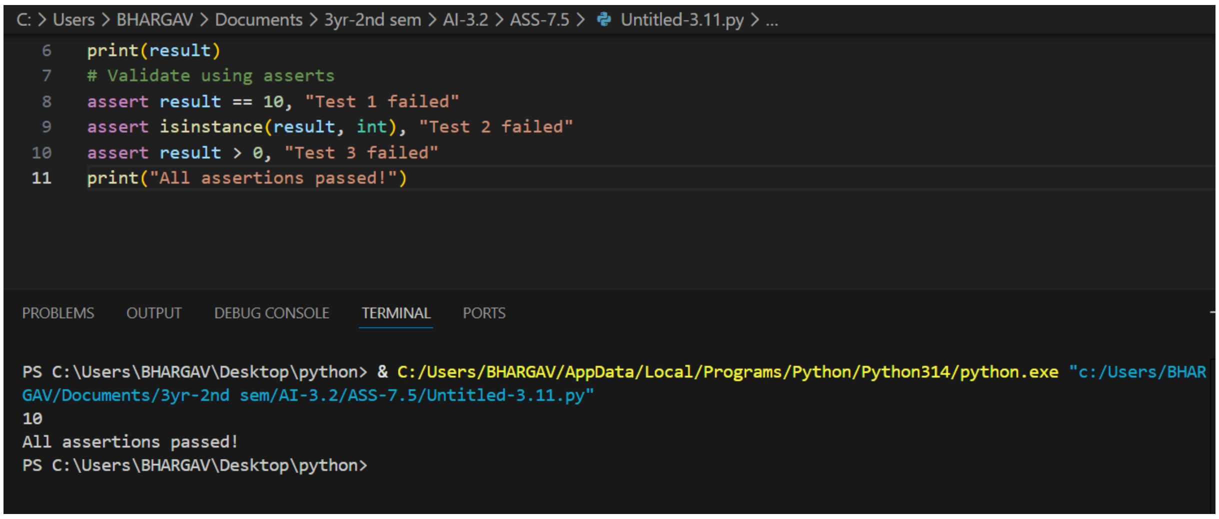
Requirements:

• Run and identify the error.

• Explain why NoneType cannot be added.

• Fix by assigning a default value.

• Validate using asserts.



C: > Users > BHARGAV > Documents > 3yr-2nd sem > AI-3.2 > ASS-7.5 > Untitled-3.11.py > ...

```python
print(result)
# Validate using asserts
assert result == 10, "Test 1 failed"
assert isinstance(result, int), "Test 2 failed"
assert result > 0, "Test 3 failed"
print("All assertions passed!")
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHAR
GAV/Documents/3yr-2nd sem/AI-3.2/ASS-7.5/Untitled-3.11.py"
10
All assertions passed!
PS C:\Users\BHARGAV\Desktop\python>
```

Task 15 (Type Error – Input Treated as String Instead of Number)

Task: Fix code where user input is not converted properly.

# Bug: Input remains string def

sum_two_numbers():

a = input("Enter first number: ") b =

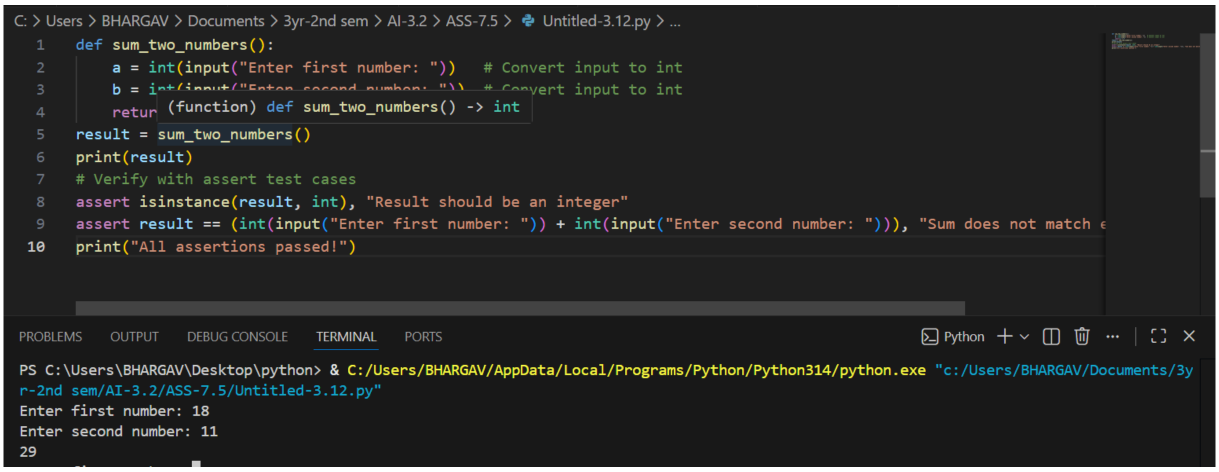
input("Enter second number: ")

return a + b

print(sum_two_numbers())

Requirements:

- Explain why input is always string.
- Fix using int() conversion.
- Verify with assert test cases.



C: > Users > BHARGAV > Documents > 3yr-2nd sem > AI-3.2 > ASS-7.5 > Untitled-3.12.py > ...

```python
def sum_two_numbers():
    a = int(input("Enter first number: "))   # Convert input to int
    b = int(input("Enter second number: "))  # Convert input to int
    retur (function) def sum_two_numbers() -> int
result = sum_two_numbers()
print(result)
# Verify with assert test cases
assert isinstance(result, int), "Result should be an integer"
assert result == (int(input("Enter first number: ")) + int(input("Enter second number: "))), "Sum does not match e
print("All assertions passed!")
```

```
PS C:\Users\BHARGAV\Desktop\python> & C:/Users/BHARGAV/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/BHARGAV/Documents/3y
r-2nd sem/AI-3.2/ASS-7.5/Untitled-3.12.py"
Enter first number: 18
Enter second number: 11
29
```