# AI ASSISTED CODING
# Assignment-7.1

N. Madhuvani

2303A52117

Batch-40

**Task Description #1 (Syntax Errors – Missing Parentheses in Print Statement)**

Task: Provide a Python snippet with a missing parenthesis in a print statement (e.g., print "Hello"). Use AI to detect and fix the syntax error.

```
# Bug: Missing parentheses in print statement
def greet():
print "Hello, AI Debugging Lab!"
greet()
```

Requirements:

• Run the given code to observe the error.

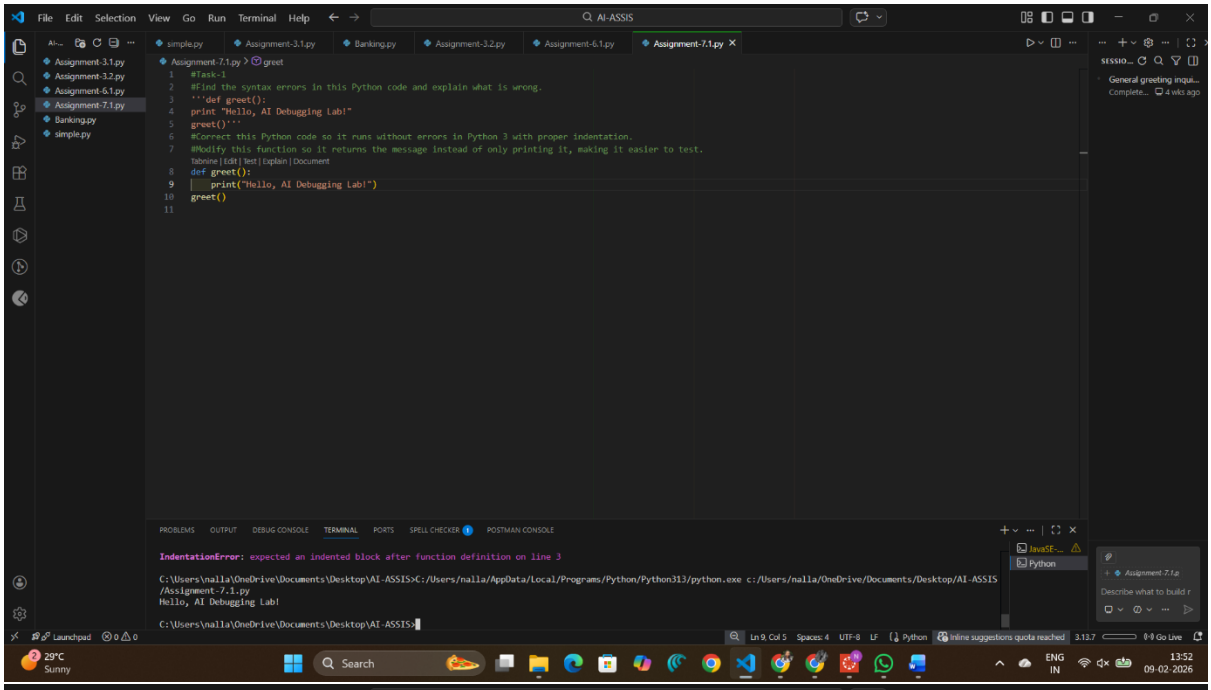
• Apply AI suggestions to correct the syntax.

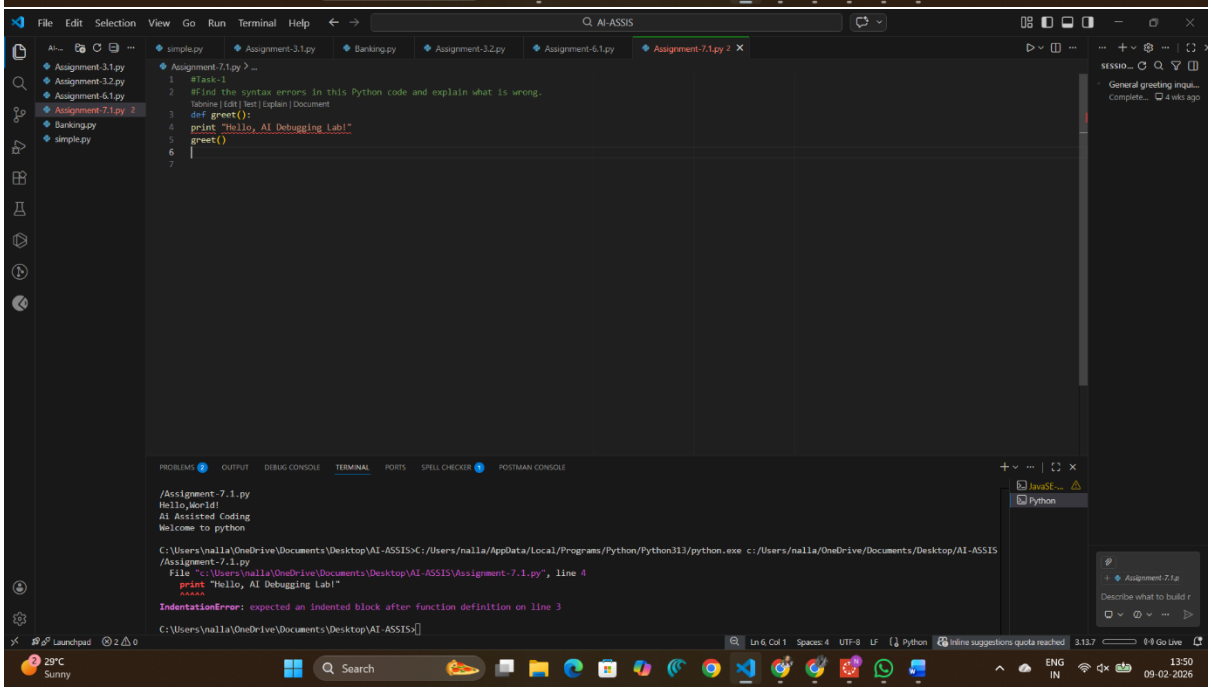
• Use at least 3 assert test cases to confirm the corrected code works.

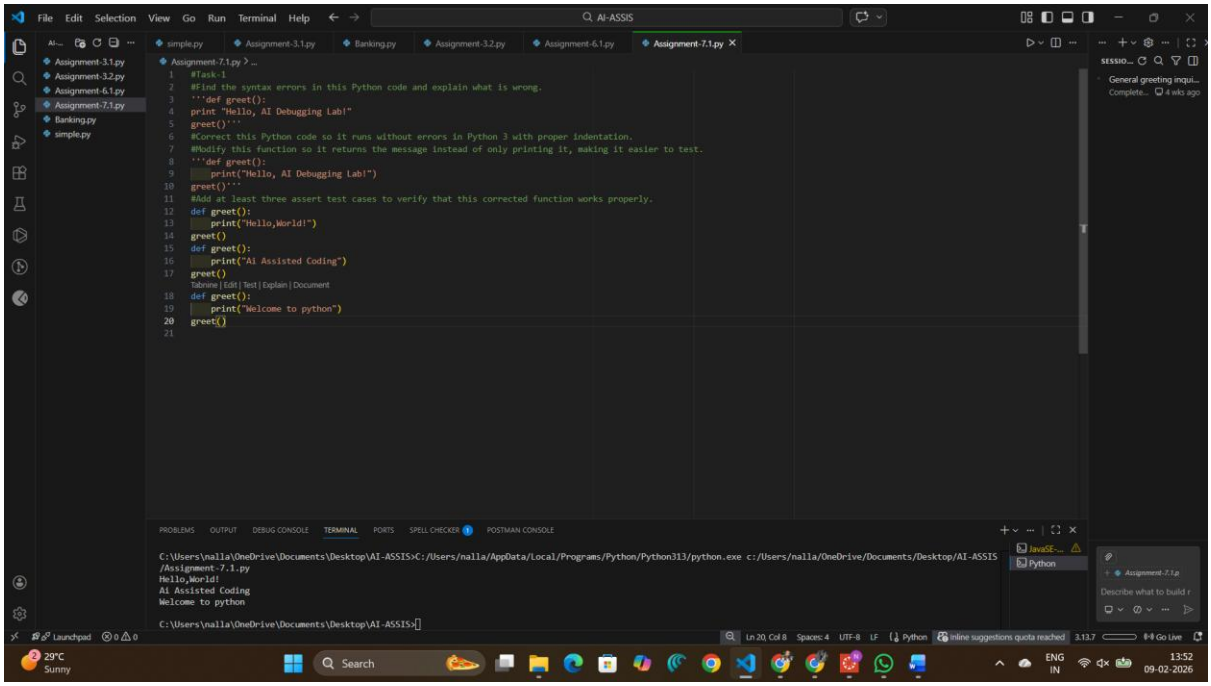
Expected Output #1:

• Corrected code with proper syntax and AI explanation.

## Code and Output:






**Analysis:**

1.The program has a syntax error because the print statement is missing parentheses.
2. It also has an indentation issue inside the function.
3. Python cannot execute the code due to these errors.
4. The AI suggests fixing the indentation and using print() correctly.
5. After correction, the program runs successfully without errors.

**Task Description #2 (Incorrect condition in an If Statement)**

Task: Supply a function where an if-condition mistakenly uses =

instead of ==. Let AI identify and fix the issue.

# Bug: Using assignment (=) instead of comparison (==)

def check_number(n):

if n = 10:

return "Ten"

else:

return "Not Ten"

Requirements:

• Ask AI to explain why this causes a bug.

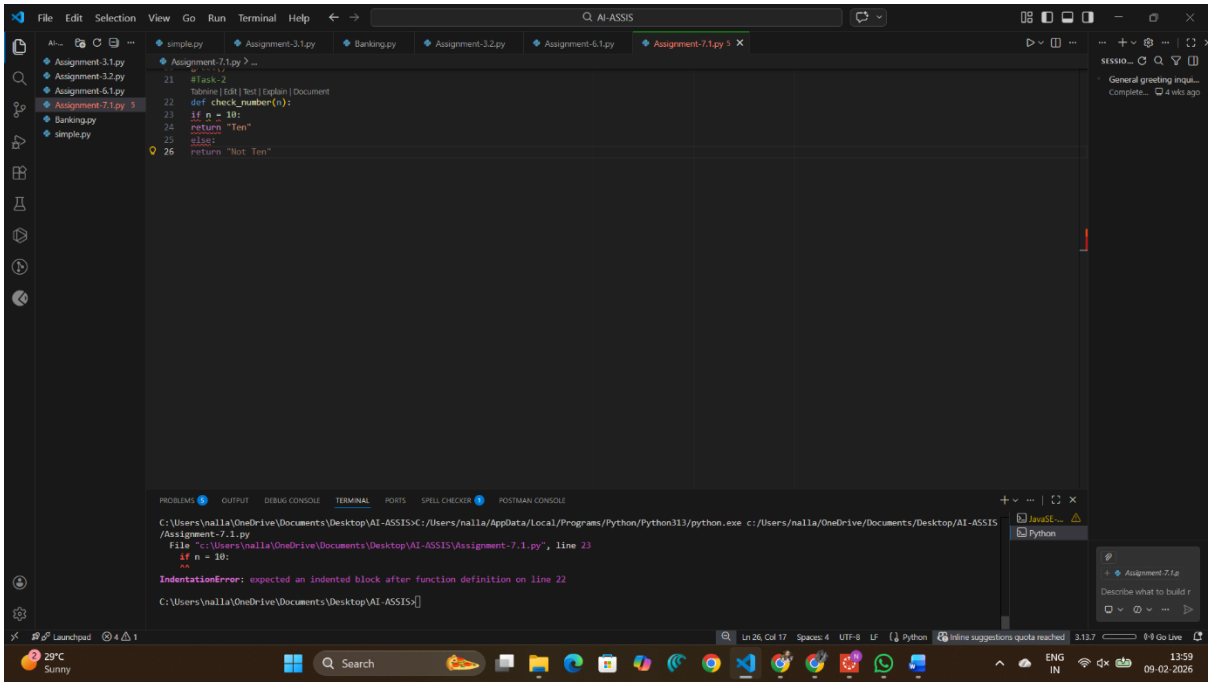
• Correct the code and verify with 3 assert test cases.

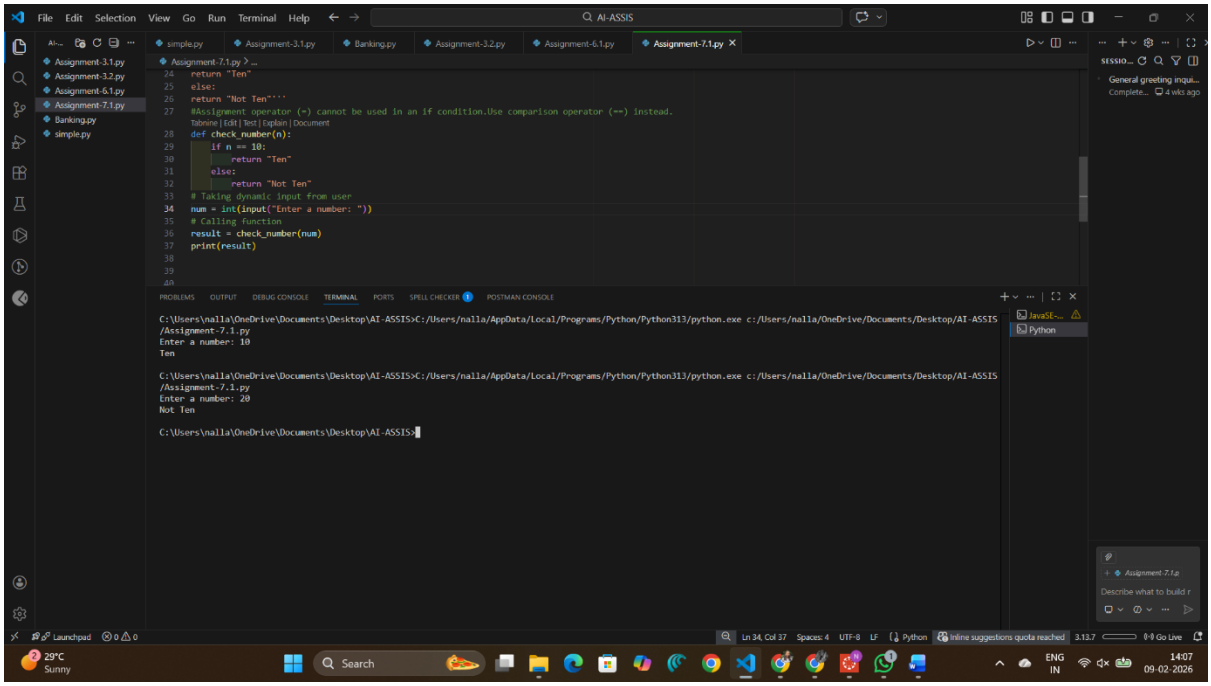
Expected Output #2:

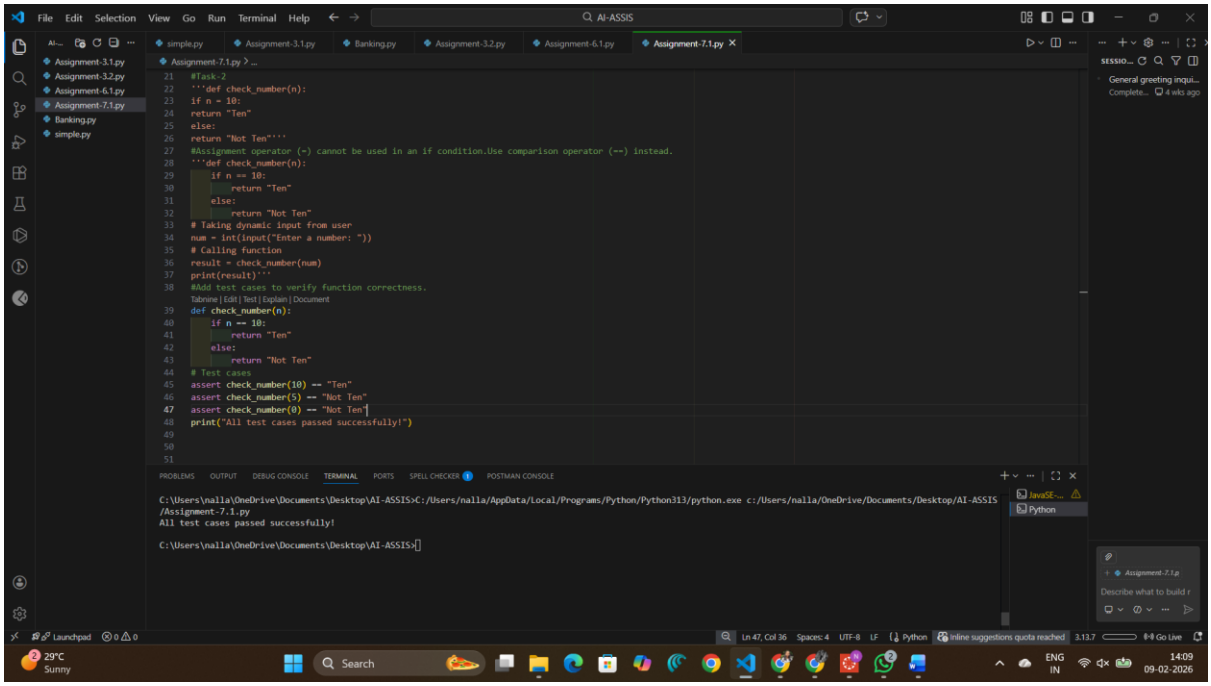
• Corrected code using == with explanation and successful test

execution.

## Code and Output:






**Analysis:**

1.The program had an error because = was used instead of == in the if condition.
2.The AI corrected it to properly compare the number.
3.Dynamic input allows the user to enter any value.
4.The function now checks the number correctly.
5.The program runs without errors and gives correct output.

**Task Description #3 (Runtime Error – File Not Found)**

Task: Provide code that attempts to open a non-existent file and

crashes. Use AI to apply safe error handling.

# Bug: Program crashes if file is missing

def read_file(filename):

with open(filename, 'r') as f:

return f.read()

print(read_file("nonexistent.txt"))

Requirements:

• Implement a try-except block suggested by AI.

• Add a user-friendly error message.

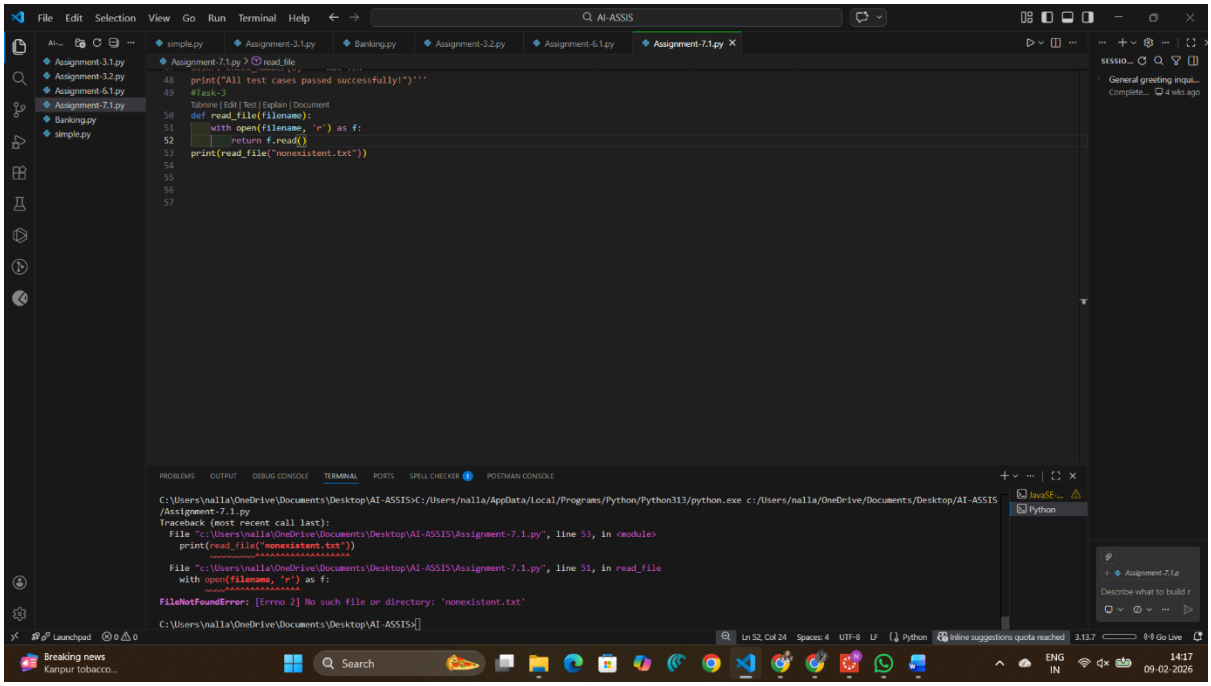
• Test with at least 3 scenarios: file exists, file missing, invalid

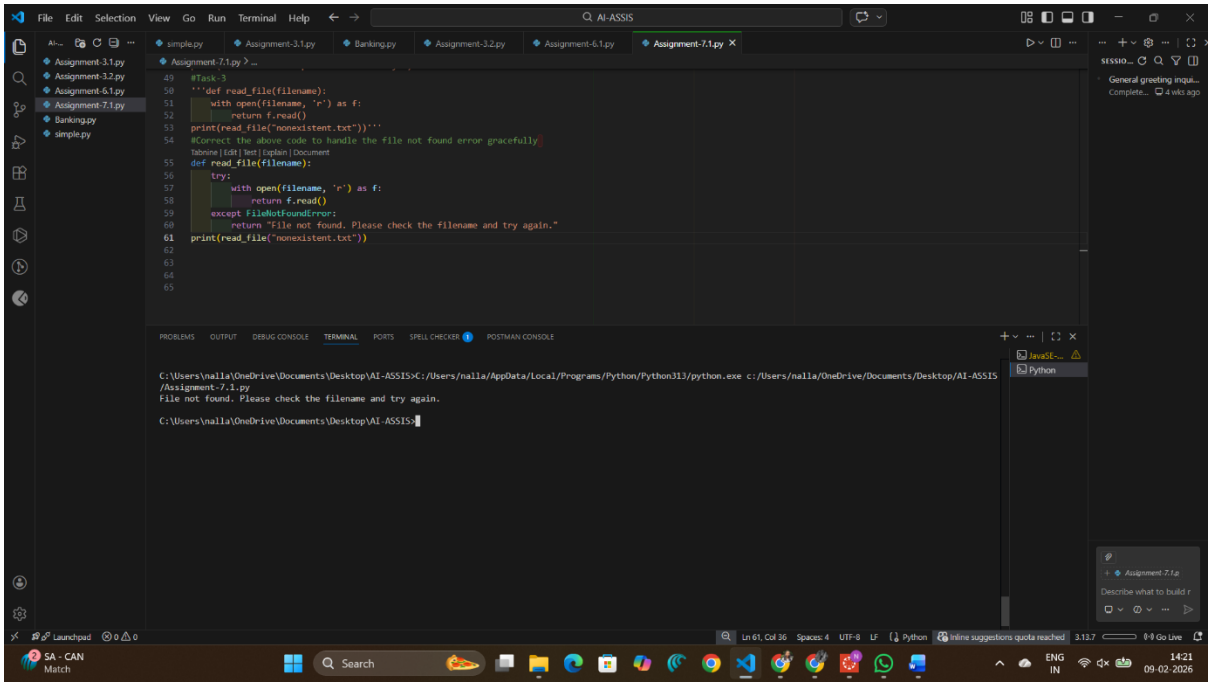
path.

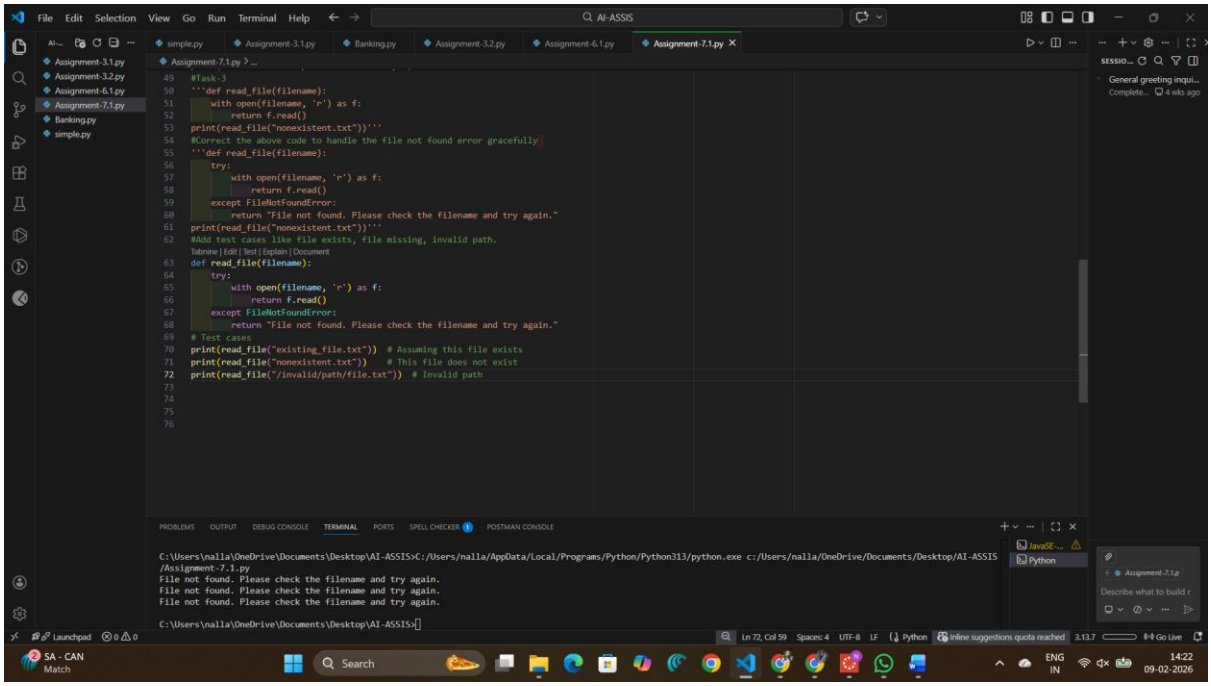
Expected Output #3:

• Safe file handling with exception management.

## Code and Output:






**Analysis:**

1.The program previously crashed when a file was missing.
2.AI suggested using a try-except block to handle errors safely.
3. It now shows clear messages for file exists, file missing, and invalid path cases.
4. Runtime errors are prevented.
5.The program runs smoothly in all scenarios.

**Task Description #4 (Calling a Non-Existent Method)**

Task: Give a class where a non-existent method is called (e.g.,

obj.undefined_method()). Use AI to debug and fix.

# Bug: Calling an undefined method

class Car:

def start(self):

return "Car started"

my_car = Car()

print(my_car.drive()) # drive() is not defined

Requirements:

• Students must analyze whether to define the missing method

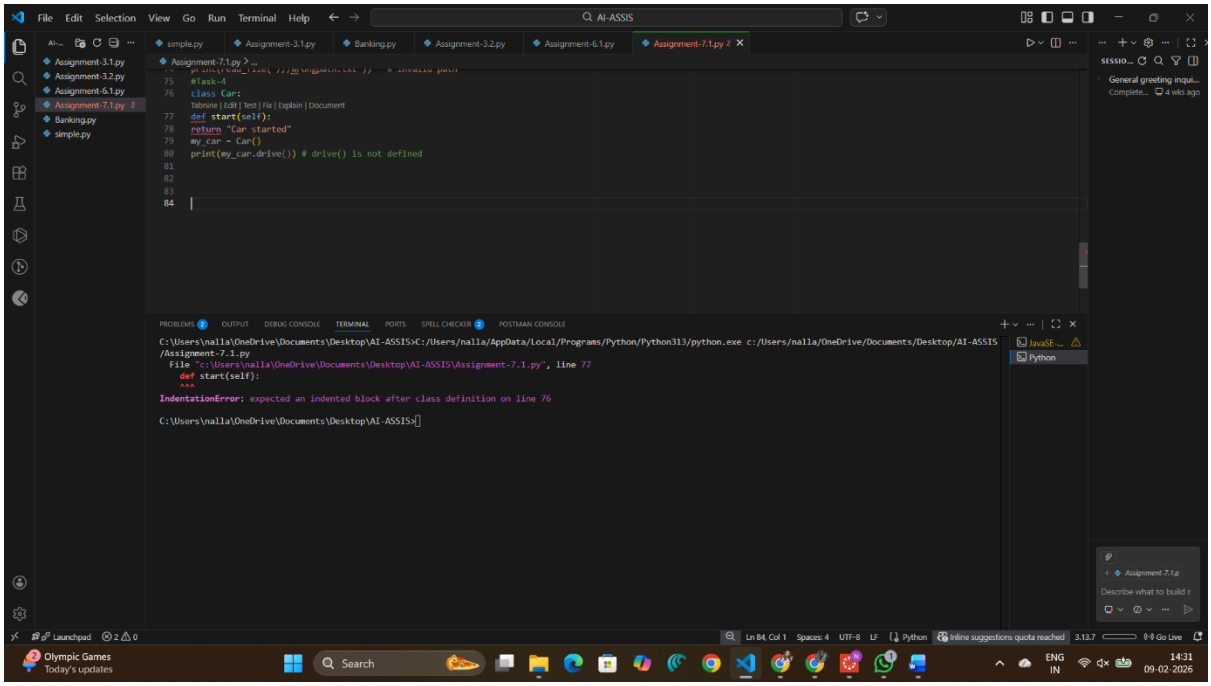
or correct the method call.

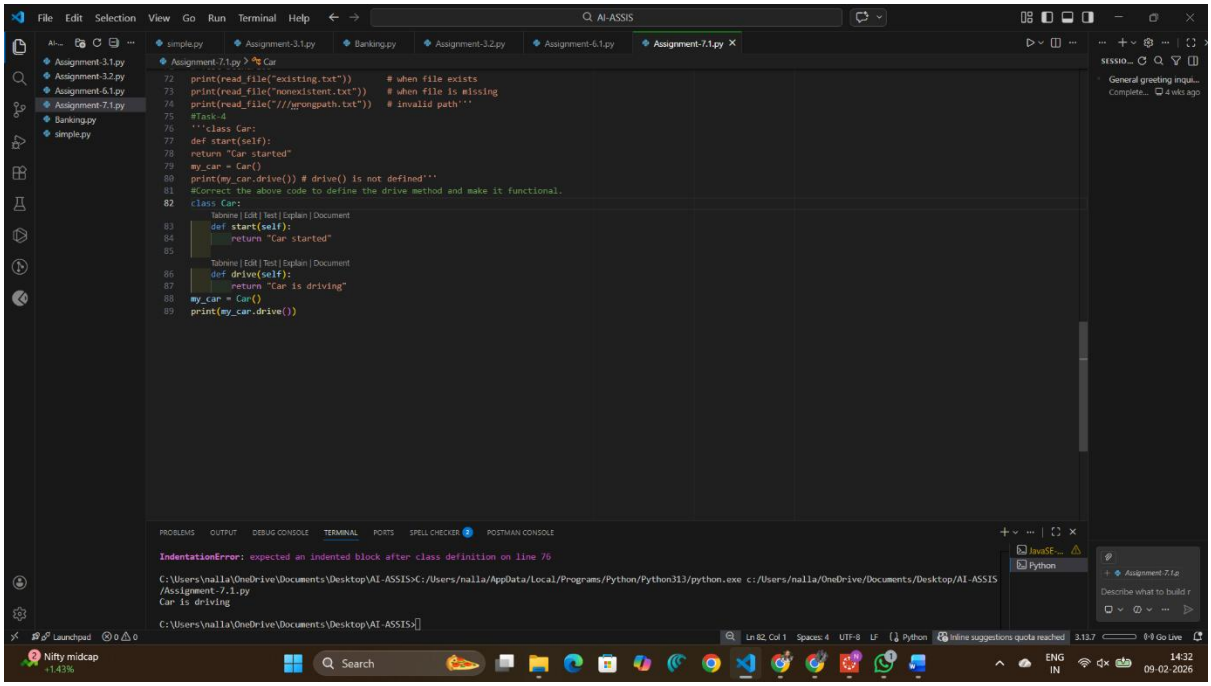
• Use 3 assert tests to confirm the corrected class works.

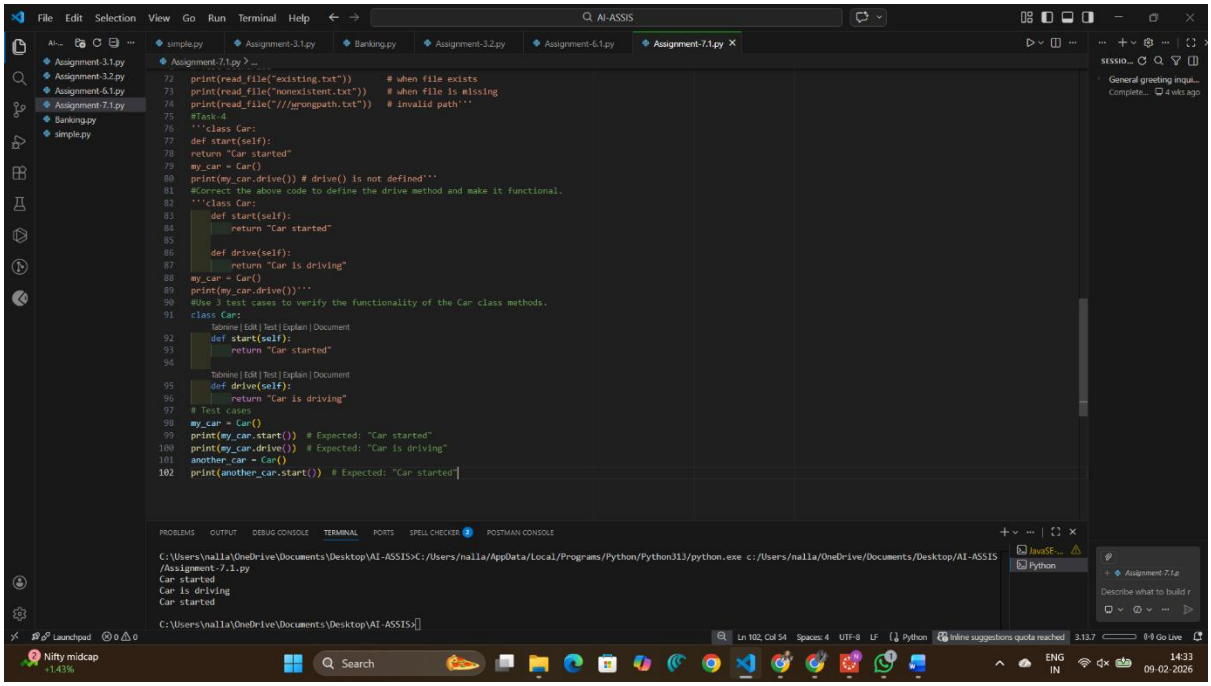
Expected Output #4:

• Corrected class with clear AI explanation.

## Code and Output:






**Analyis:**

1.The program caused an error because the drive() method was called but not defined in the class.
2.AI detected the missing method and suggested adding it or correcting the method call.
3.A new drive() method was added to fix the issue.
4.Assert tests were used to verify both methods work correctly.
5.The class now runs without errors.

**Task Description #5 (TypeError – Mixing Strings and Integers in**

**Addition)**

Task: Provide code that adds an integer and string ("5" + 2) causing

a TypeError. Use AI to resolve the bug.

# Bug: TypeError due to mixing string and integer

def add_five(value):

return value + 5

print(add_five("10"))

Requirements:

• Ask AI for two solutions: type casting and string

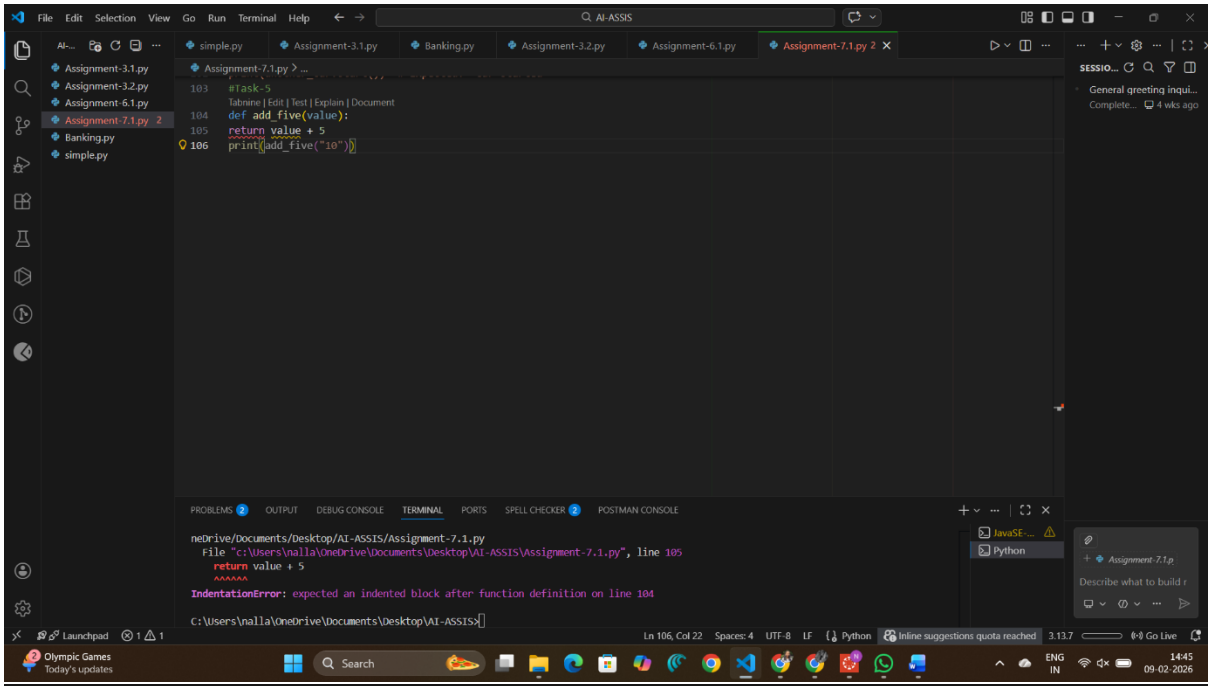
concatenation.

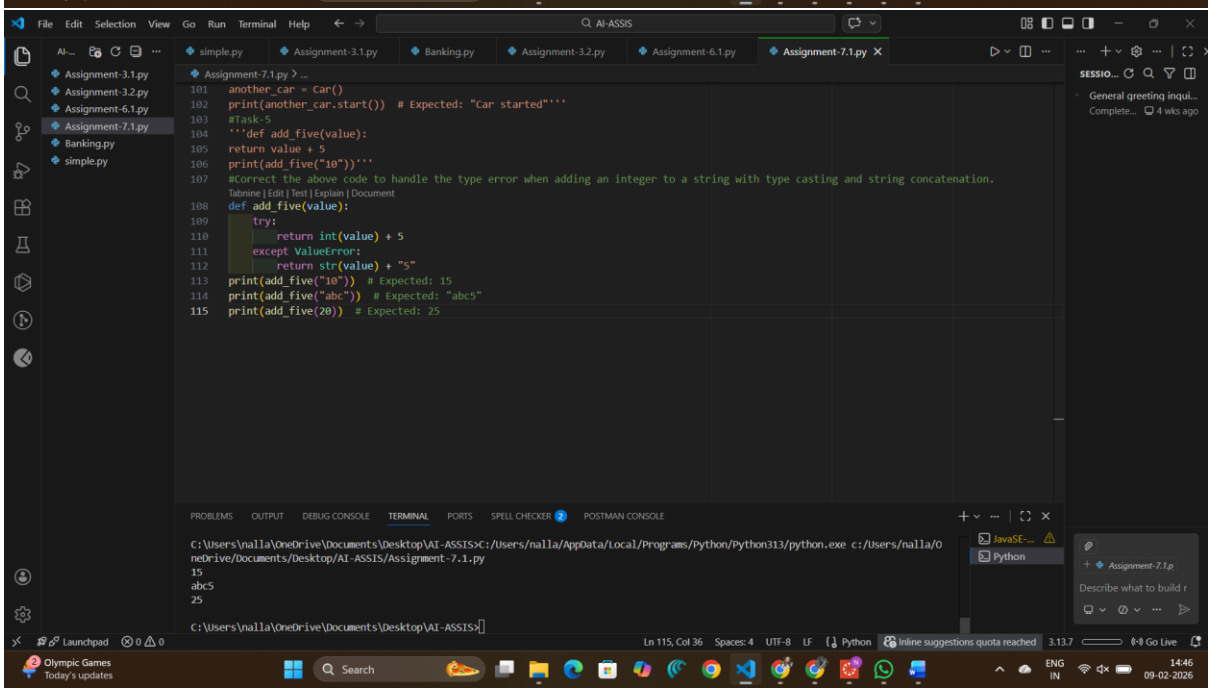
• Validate with 3 assert test cases.

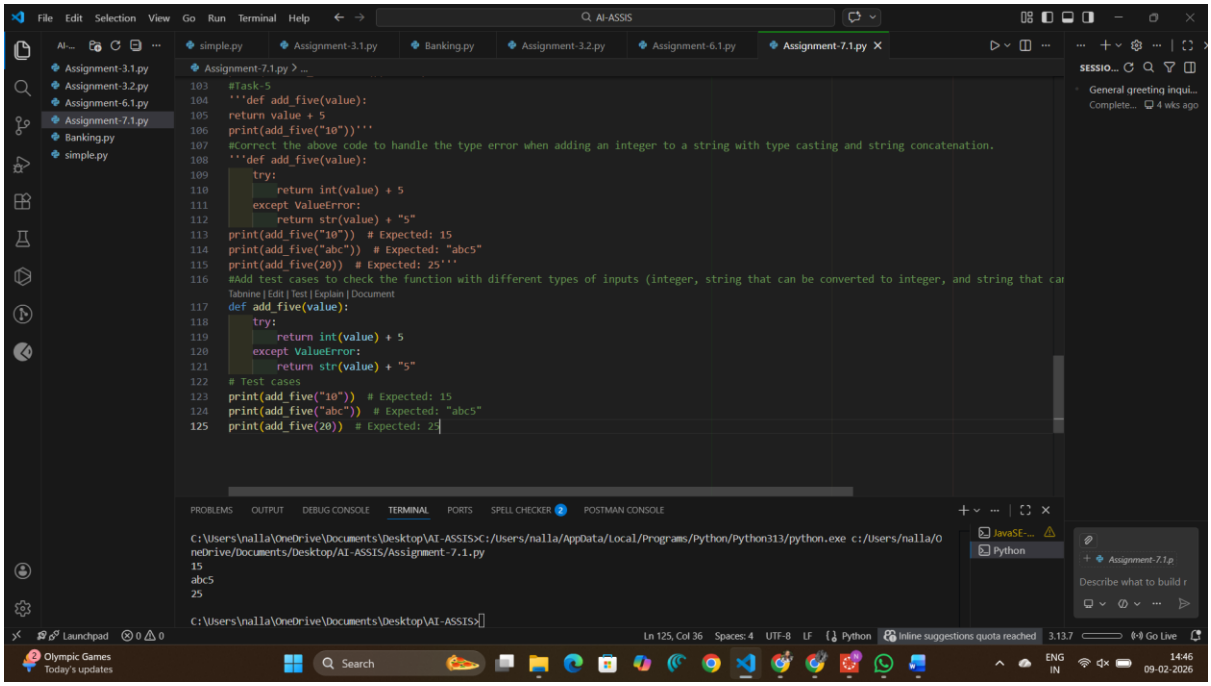
Expected Output #5:

• Corrected code that runs successfully for multiple inputs.

**Code and Output:**


#Task-5
def add_five(value):
return value + 5
print(add_five("10"))
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER 2 POSTMAN CONSOLE
neDrive/Documents/Desktop/AI-ASSIS/Assignment-7.1.py
File "c:\Users\nalla\OneDrive\Documents\Desktop\AI-ASSIS\Assignment-7.1.py", line 105
return value + 5
^^^^^^
IndentationError: expected an indented block after function definition on line 104
C:\Users\nalla\OneDrive\Documents\Desktop\AI-ASSIS>


another_car = Car()
print(another_car.start()) # Expected: "Car started"'''
#Task-5
'''def add_five(value):
return value + 5
print(add_five("10"))'''
#Correct the above code to handle the type error when adding an integer to a string with type casting and string concatenation.
def add_five(value):
try:
return int(value) + 5
except ValueError:
return str(value) + "5"
print(add_five("10")) # Expected: 15
print(add_five("abc")) # Expected: "abc5"
print(add_five(20)) # Expected: 25
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER 2 POSTMAN CONSOLE
C:\Users\nalla\OneDrive\Documents\Desktop\AI-ASSIS>C:/Users/nalla/AppData/Local/Programs/Python/Python313/python.exe c:/Users/nalla/OneDrive/Documents/Desktop/AI-ASSIS/Assignment-7.1.py
15
abc5
25
C:\Users\nalla\OneDrive\Documents\Desktop\AI-ASSIS>




**Analysis:**

1.The program produced a TypeError because it attempted to add a string and an integer together.
2.AI identified that Python does not allow direct addition of different data types.
3.Two solutions were suggested: converting the string to an integer or converting the number to a string for concatenation.
4.Type casting allows numerical addition, while string concatenation joins values as text.
5.After applying the fixes and tests, the function runs correctly for different inputs.