# **AI ASSISTED CODING**

# **ASSIGNMENT-7.1**

NAME:S.VRINDHA REDDY

HALL NUMBER:2403A51255

BATCH:11

**Task-1: (Syntax Errors – Missing Parentheses in Print Statement)**

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.

**Excepted Output:**

• Corrected code with proper syntax and AI explanation

**Prompt-1:**

I have a Python program that shows an error. Please:

def check\_number(n):

if n = 10:

return "Ten"

else:

return "Not Ten"

Identify the type of error (syntax, logic, runtime, type, or attribute).

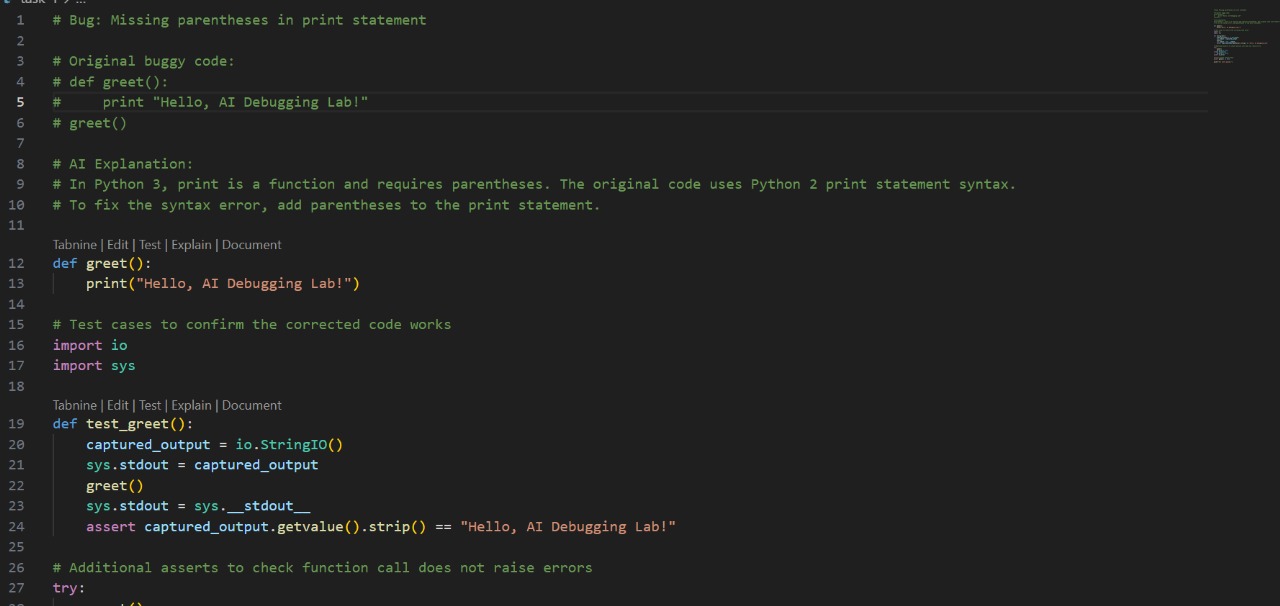
Explain why the error occurs.

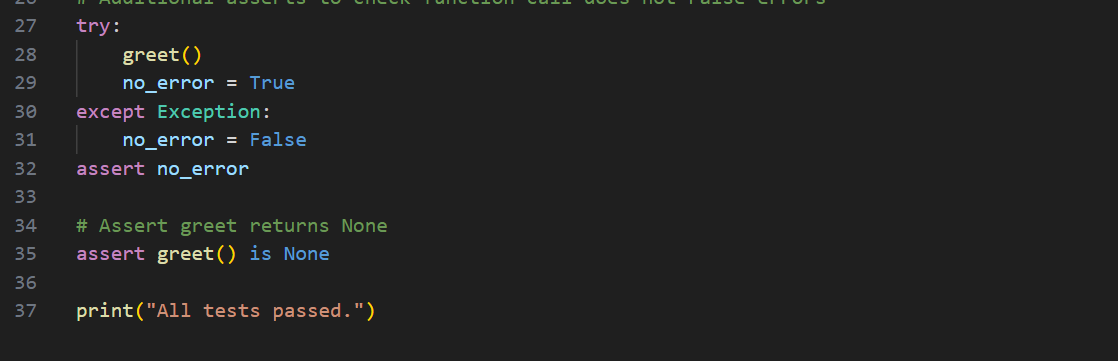
Suggest a corrected version of the code.

Provide at least 3 assert test cases to confirm the fix works.

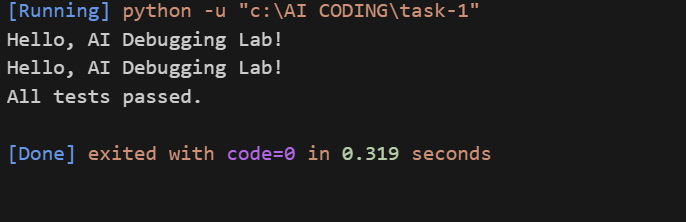
Show the expected output after correction.

**Code:**



****

**Output:**

****

**Task-2:** **(Logic Error – Incorrect Condition in an If Statement)**

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.

**Prompt-2:**

I have a Python program that shows an error. Please:

def check\_number(n):

if n = 10:

return "Ten"

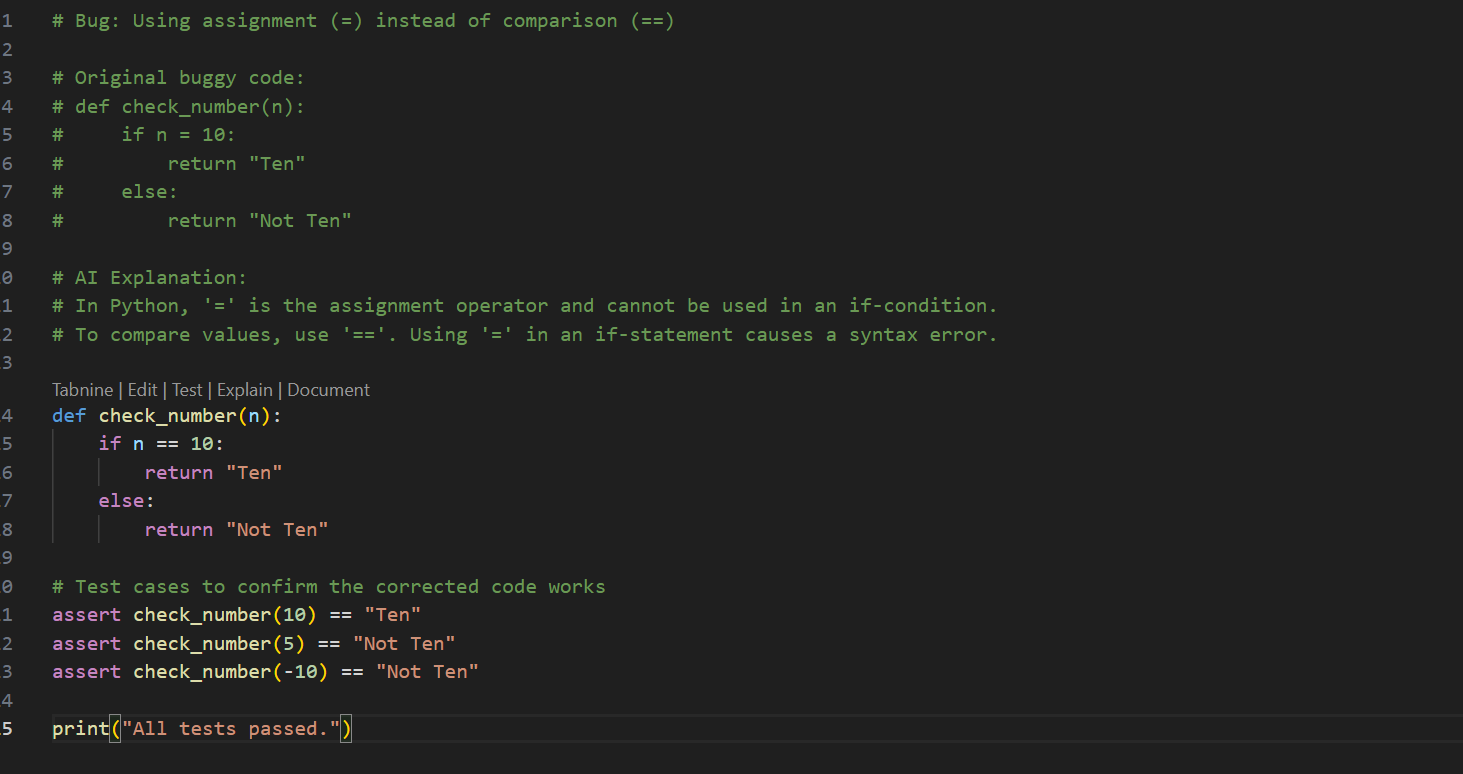
else

return "Not Ten"

Identify the type of error (syntax, logic, runtime, type, or attribute).

1. Identify the type of error.
2. Explain why using = instead of == causes a bug.
3. Correct the code with the proper comparison operator.
4. Provide at least 3 assert test cases.
5. Show the expected output for the test cases**.**

**Code:**

****

**Output:**

**A black screen with white text

AI-generated content may be incorrect.**

**Task-3:** **(Runtime Error – File Not Found)**

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.

**Prompt-3:**

I have a Python program that shows an error. Please:

def read\_file(filename):

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

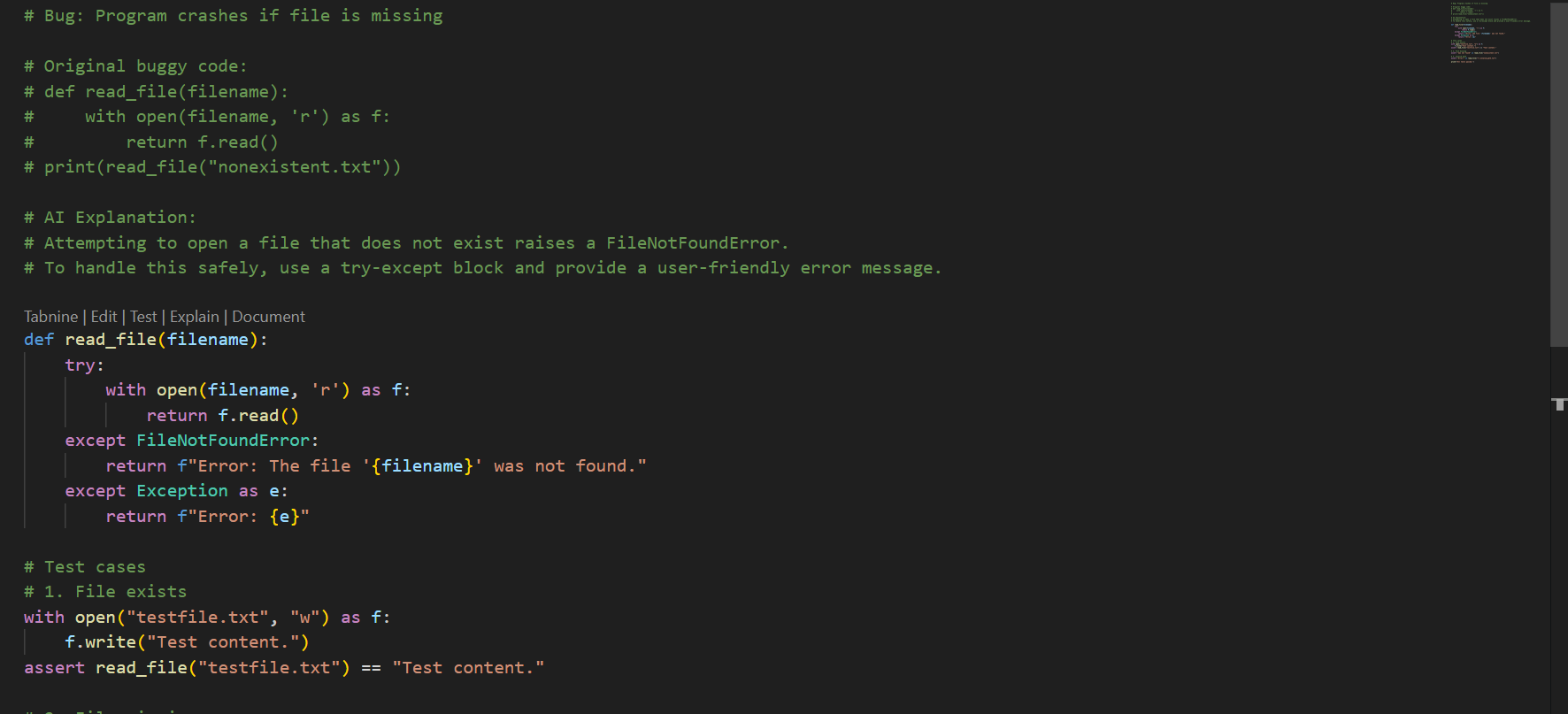
return f. read ()

print(read\_file("nonexistent.txt"))

Identify the type of error (syntax, logic, runtime, type, or attribute).

1. Identify the type of error.
2. Explain why the program crashes when the file does not exist.
3. Add safe error handling using try-except.
4. Provide 3 test scenarios:
   * File exists
   * File missing
   * Invalid path
5. Show the expected output in each case.

**Code:**

****

**A screen shot of a computer program

AI-generated content may be incorrect.**

**Output:**

A black background with purple and white text

AI-generated content may be incorrect.

**Task-4: (Attribute Error – Calling a Non-Existent Method)**

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**.**

**Prompt-4:**

The following program calls a method that does not exist in the class. Please debug it.

class Car:

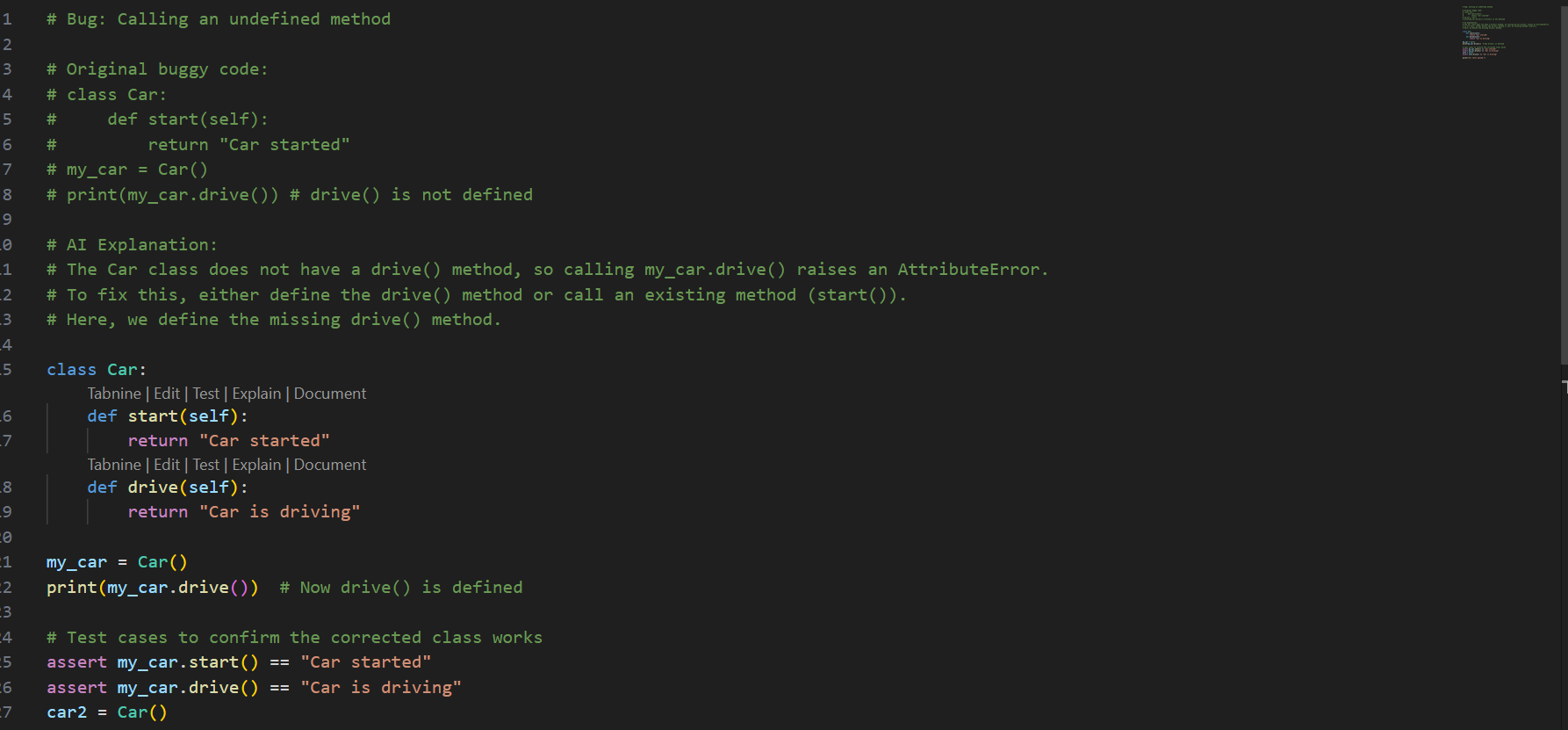
def start(self):

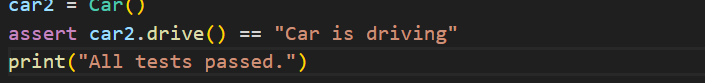
return "Car started"

my\_car = Car()

print (my\_car. drive ())

**Code:**





**Output:**

A black screen with white text

AI-generated content may be incorrect.

**Task-5: (TypeError – Mixing Strings and Integers in Addition)**

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

**Prompt-5:**

The following code mixes strings and integers, causing a TypeError. Please provide two possible fixes:

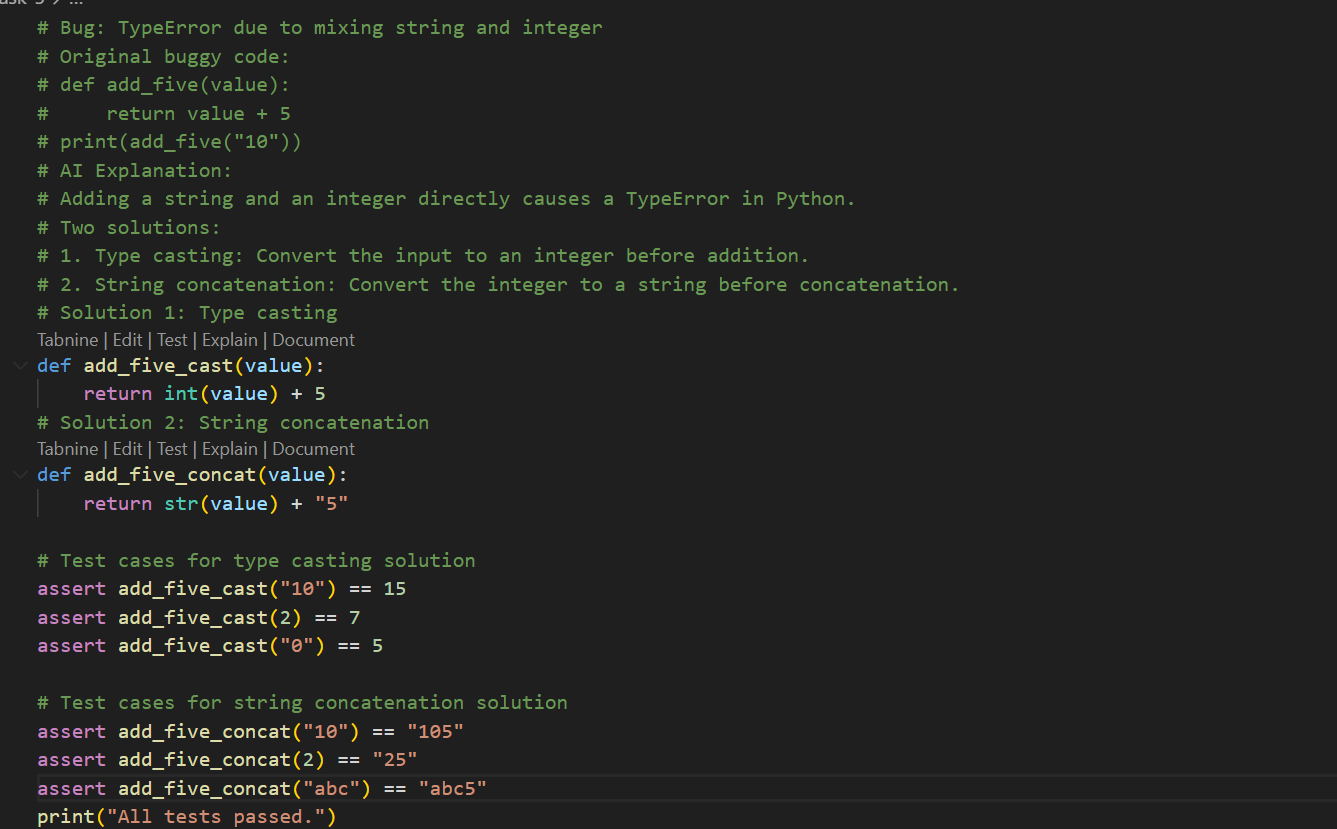
1. Convert input to integer.
2. Use string concatenation.

def add\_five(value):

return value + 5

print (add\_five ("10"))

**Code:**

****

**Output:**

**A black background with orange text

AI-generated content may be incorrect.**