AI Assisted Coding

Assignment – 7.1

K.VAMSHIDHAR|| 2303A51OH7|| Batch:- 8

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()

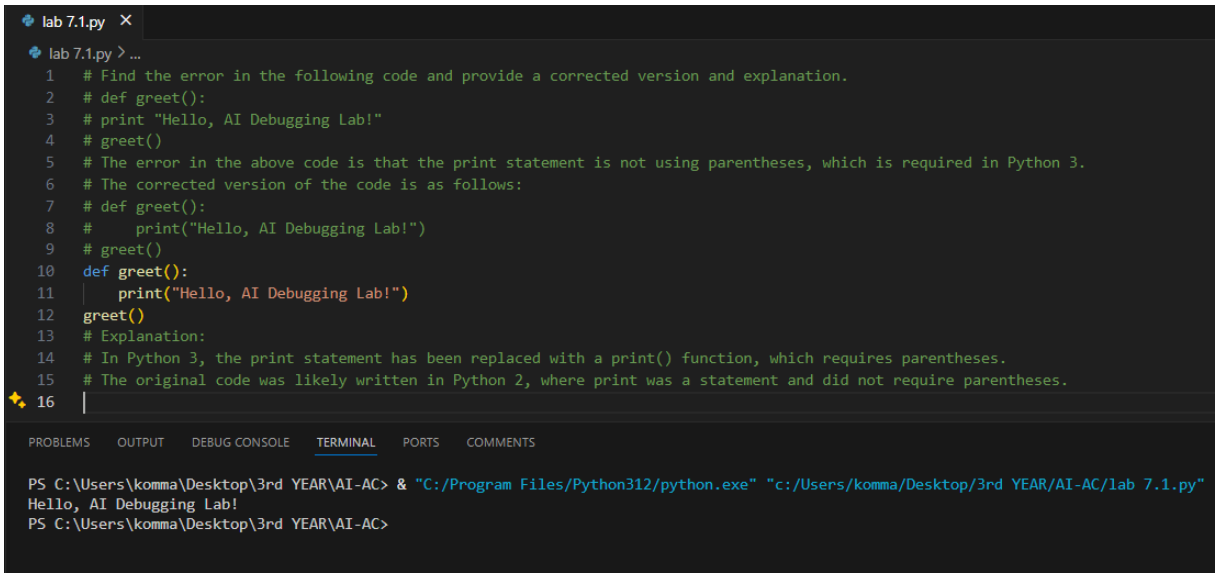
Requirements:

• 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.



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.

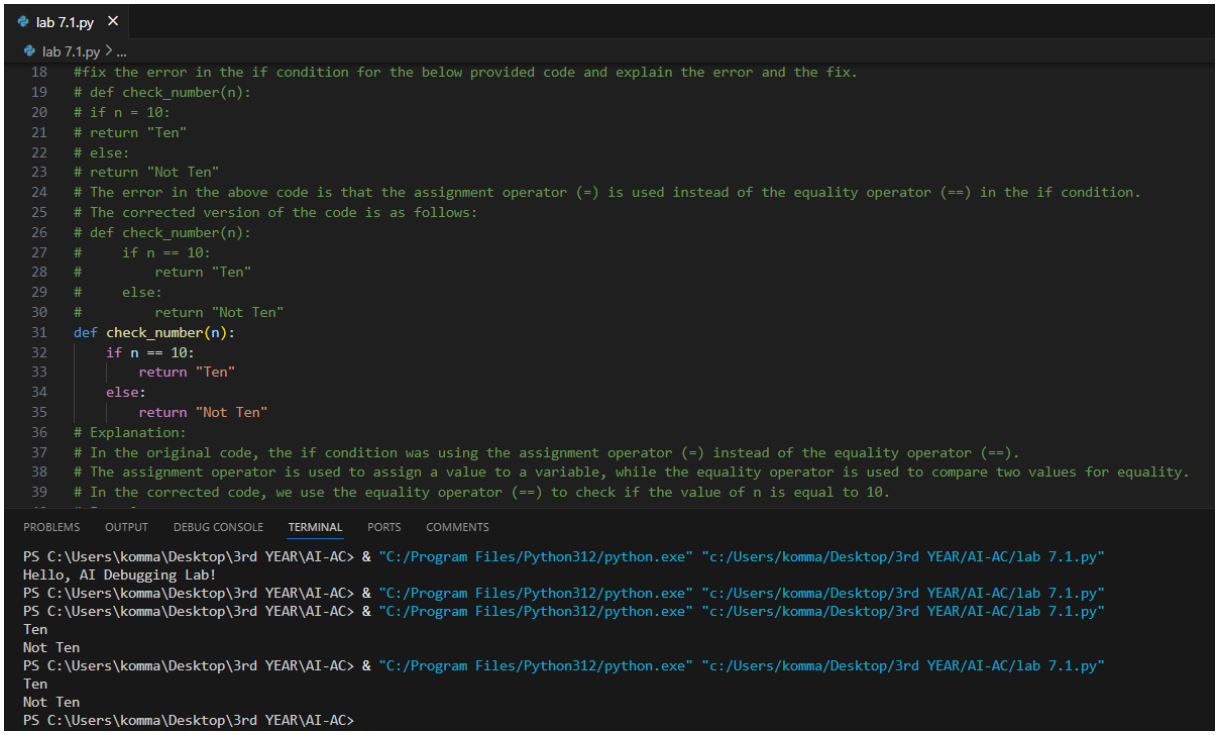
Bug: Using assignment (=) instead of comparison (==)

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.



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:

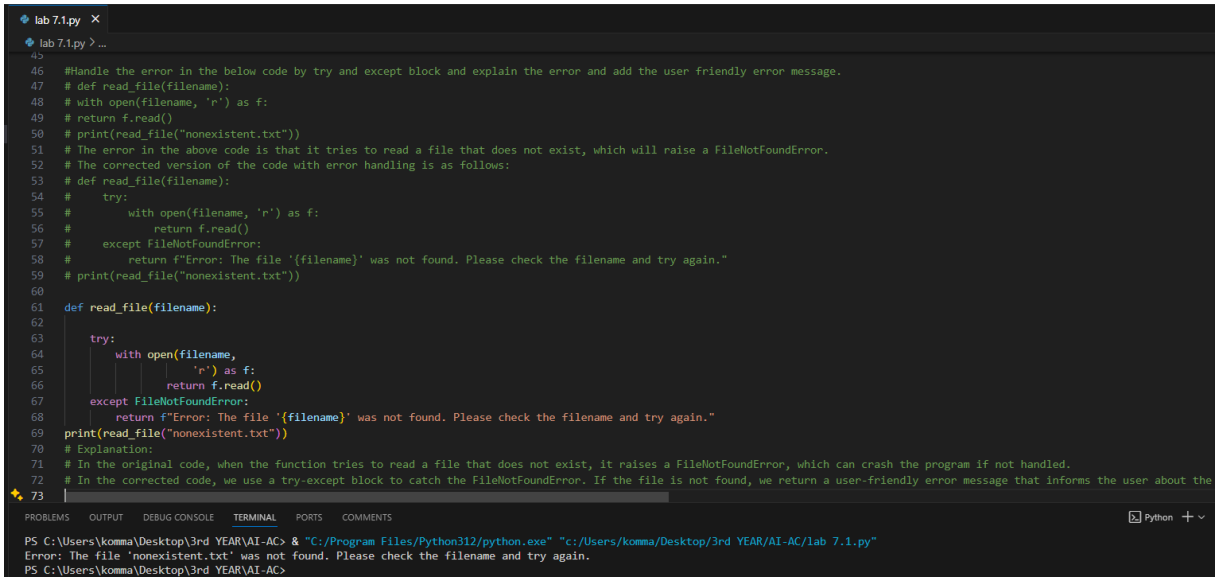
return f.read() print(read_file("nonexistent.txt"))

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.



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.

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.

```python

#In the below code fix the bug related to the Calling an undefined method and explain the error and the fix.
# class Car:
# def start(self):
# return "Car started"
# my_car = Car()
# print(my_car.drive()) # drive() is not defined
# The error in the above code is that the method `drive()` is called on the `my_car` object, but it is not defined in the `Car` class.
# The corrected version of the code is as follows:
class Car:
    def start(self):
        return "Car started"

    def drive(self):
        return "Car is driving"
my_car = Car()
print(my_car.drive())  # Output: Car is driving
# Explanation:
# In the original code, the `drive()` method was called on the `my_car` object, but it was not defined in the `Car` class, which would result in an AttributeError.
# In the corrected code, we defined the `drive()` method within the `Car` class, which allows us to call it on the `my_car` object without any errors. Now, when we call `my_car.dri
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  COMMENTS                                                    Python

PS C:\Users\komma\Desktop\3rd YEAR\AI-AC> & "C:/Program Files/Python312/python.exe" "c:/Users/komma/Desktop/3rd YEAR/AI-AC/lab 7.1.py"
Car is driving
PS C:\Users\komma\Desktop\3rd YEAR\AI-AC>
```

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.