

Lab Assignment- 7.1

NAME: M. Geethika

ID.NO: 2303A52276

Subject: AI Assisted Coding

Task #1 – Syntax Error (Missing Parentheses in print)

Prompt: Use AI to detect the syntax error caused by missing parentheses in the print statement, correct it, and validate the fix using at least 3 assert test cases.

The screenshot shows the BLACKBOX AI Agent interface. On the left, there's a sidebar with various icons and a 'TASK' section containing the file path '@/Lab7.1_2256.py'. The main area is titled 'correct the given code' and displays a Python script with syntax highlighting. The code has several lines underlined with red, indicating errors. The terminal tab at the bottom shows command-line output related to the task.

```
#Task #1 - Syntax Error (Missing Parentheses in print)
def greet():
    return "Hello, AI Debugging Lab!"

print(greet())

# Validation with assert test cases
assert greet() == "Hello, AI Debugging Lab!"
assert len(greet()) > 0
assert "Hello" in greet()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

PS C:\Users\kotas\OneDrive\AI assist> & C:/Users/kotas/AppData/Local/Microsoft/Windows/e "c:/Users/kotas/OneDrive/AI assist/Lab7.1_2256.py"
● Hello, AI Debugging Lab!
○ PS C:\Users\kotas\OneDrive\AI assist>

AI Explanation

Python 3 requires parentheses in `print()`. Without them, it raises a **SyntaxError**.

Code Explanation

The syntax was corrected by adding parentheses to the `print` function.

Task #2 – Incorrect If Condition (= vs ==)

Prompt

Ask AI to explain why using `=` instead of `==` causes a bug, correct the code, and verify the fix using 3 assert test cases.

```

13     #Task #2 - Incorrect If Condition (= vs ==)
14     def check_number(n):
15         if n == 10:
16             return "Ten"
17         else:
18             return "Not Ten"
19
20     # Validation with assert test cases for check_number
21     assert check_number(10) == "Ten"
22     assert check_number(5) == "Not Ten"
23     assert check_number(0) == "Not Ten"
24
25

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

PS C:\Users\kotas\OneDrive\AI assist> & C:/Users/kotas/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/kotas/OneDrive/AI assist/Lab7.1_2256.py"
● Hello, AI Debugging Lab!
● PS C:\Users\kotas\OneDrive\AI assist> & C:/Users/kotas/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/kotas/OneDrive/AI assist/Lab7.1_2256.py"
Hello, AI Debugging Lab!
○ PS C:\Users\kotas\OneDrive\AI assist>

AI Explanation

= is an assignment operator, not comparison. Conditions require ==.

Code Explanation

Using == ensures proper comparison instead of assignment.

Task #3 – Runtime Error (File Not Found)

Prompt

Use AI to add safe error handling using try-except, provide a user-friendly message, and validate with 3 test scenarios and assert checks.

```

26     def read_file(filename):
27         with open(filename, 'r') as f:
28             return f.read()
29     except FileNotFoundError:
30         return "Error: The file '{}' was not found. Please check the file path and try again.".format(filename)
31
32
33     print(read_file("nonexistent.txt"))
34
35     # Validation with assert test cases for read_file
36     assert "Error:" in read_file("nonexistent.txt")
37     assert "not found" in read_file("nonexistent.txt")
38     assert read_file("nonexistent.txt") == "Error: The file 'nonexistent.txt' was not found. Please check the file path and try again."
39

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

PS C:\Users\kotas\OneDrive\AI assist> & C:/Users/kotas/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/kotas/OneDrive/AI assist/Lab7.1_2256.py"
● Hello, AI Debugging Lab!
● PS C:\Users\kotas\OneDrive\AI assist> & C:/Users/kotas/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/kotas/OneDrive/AI assist/Lab7.1_2256.py"
Hello, AI Debugging Lab!
● PS C:\Users\kotas\OneDrive\AI assist> & C:/Users/kotas/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/kotas/OneDrive/AI assist/Lab7.1_2256.py"
Hello, AI Debugging Lab!
○ PS C:\Users\kotas\OneDrive\AI assist>
Error: The file 'nonexistent.txt' was not found. Please check the file path and try again.
○ PS C:\Users\kotas\OneDrive\AI assist>

AI Explanation

The error occurs because the program tries to open a non-existent file, so AI suggests using a try–except block to handle the exception safely.

Code Explanation

try-except prevents crashes and handles runtime errors gracefully.

Task #4 – Calling a Non-Existential Method

Prompt

Use AI to analyze the bug, either define the missing method or correct the method call, and validate using 3 assert tests.

The screenshot shows the AI assist interface in a code editor. The code editor displays a Python script named `Lab7.1_2256.py`. The code defines a `Car` class with two methods: `start` and `drive`. The `start` method is correctly implemented, returning "Car started". The `drive` method is also correctly implemented, returning "Car is driving". A task card on the left side of the interface provides instructions: "First, the task is to use AI to analyze the bug, either define the missing method or correct the method call, and validate using 3 assert tests." Below the code editor is a terminal window showing command-line interactions. It shows the user navigating to a directory, running a command, and receiving an error message: "Error: The file 'nonexistent.txt' was not found. Please check the file path and try again." This indicates that the code is attempting to use a non-existent file, which is the bug being analyzed.

AI Explanation

`drive()` does not exist, causing **AttributeError**.

Code Explanation

The missing method was defined to eliminate the **AttributeError**.

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

Prompt

Ask AI for two solutions (type casting and string concatenation), fix the bug, and validate using 3 assert test cases.

The screenshot shows a code editor interface with the following details:

- File:** Lab7.1_2256.py
- Content:**

```
55     #Task Description #5 (TypeError - Mixing Strings and Integers in Addition)
56     # Solution 1: Type casting - convert string to int if necessary, then add 5
57     def add_five(value):
58         if isinstance(value, str):
59             value = int(value)
60         return value + 5
61
62     # Solution 2: String concatenation - if string, append "5", else convert to string and app
63     def add_five_concat(value):
64         if isinstance(value, str):
65             return value + "5"
66         else:
67             return str(value) + "5"
68
69     print(add_five("10")) # Uses type casting
70     print(add_five_concat("10")) # Uses string concatenation
71
72     # Validation with assert test cases for add_five (type casting)
73     assert add_five(10) == 15
74     assert add_five("10") == 15
75     assert add_five(5) == 10
76
77     # Validation with assert test cases for add_five_concat (string concatenation)
78     assert add_five_concat("10") == "105"
79     assert add_five_concat(10) == "105"
80     assert add_five_concat("hello") == "hello5"
```
- Terminal:**

```
PS C:\Users\kotas\OneDrive\AI assist> Error: The file 'nonexistent.txt' was not found. Please check the file path and try again.
```

AI Explanation

The error happens due to adding incompatible data types, so AI suggests type casting or string concatenation to fix it.

Code Explanation

The error occurred due to mixing data types. Fixes include converting to integer or concatenating strings.