Lab 7.1: Error Debugging with AI – Systematic Approaches to Finding and Fixing Bugs

Name:B.TejasKumar

HallTicketNo:2303A51489

Lab Outcomes

•Detectandcorrectsyntax,logic,andruntimeerrors.
•UnderstandAIexplanationsforbugs.
•Applystructureddebuggingstrategies.
•Refactorbuggycodesafelyandcorrectly.

Task1:SyntaxError–MissingParenthesesinPrintStatement

BuggyCode:
```
defgreet():
   print"Hello,AIDebuggingLab!"
```

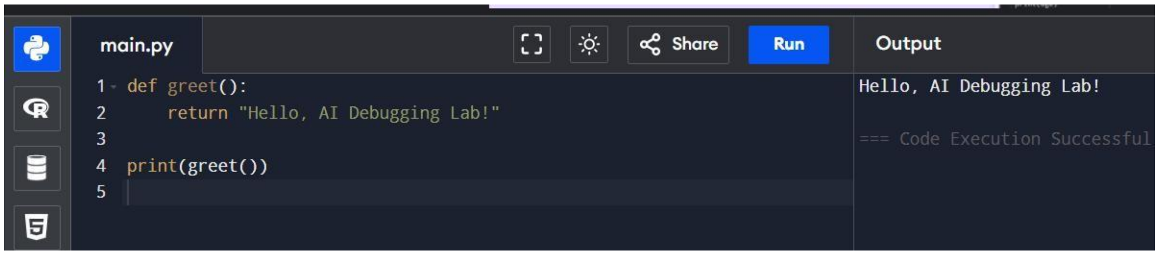
ObservedError:
SyntaxError:Missingparenthesesincallto'print'

AIExplanation:
Python3requiresparenthesesfortheprint()function.

CorrectedCode:
```
defgreet():
   return"Hello,AIDebuggingLab!"

print(greet())
```

Assert Test Cases:
assert greet() == "Hello, AI Debugging Lab!"
assert isinstance(greet(), str)
assert greet().startswith("Hello")

Output:
Hello, AI Debugging Lab!



```python
def greet():
    return "Hello, AI Debugging Lab!"

print(greet())
```

```
Hello, AI Debugging Lab!

=== Code Execution Successful
```

Task 2: Logic Error – Incorrect Condition in If Statement

Buggy Code:

```
def check_number(n):
    if n = 10:
        return "Ten"
```

AI Explanation:
= is assignment, == is comparison.

Corrected Code:

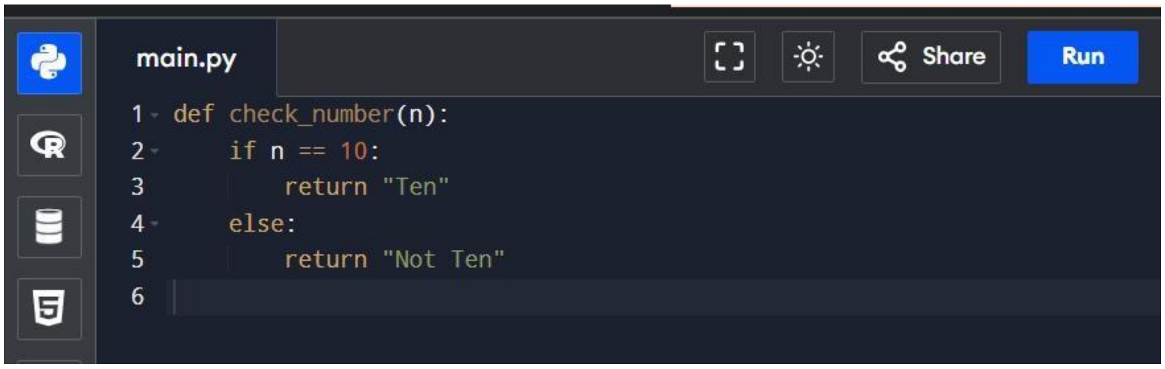
```
def check_number(n):
    if n == 10:
        return "Ten"
    else:
        return "Not Ten"
```

Assert Test Cases:

```
assert check_number(10) == "Ten"
assert check_number(5) == "Not Ten"
assert check_number(0) == "Not Ten"
```

Output:

```
All test cases passed
```



```
def check_number(n):
    if n == 10:
        return "Ten"
    else:
        return "Not Ten"
```

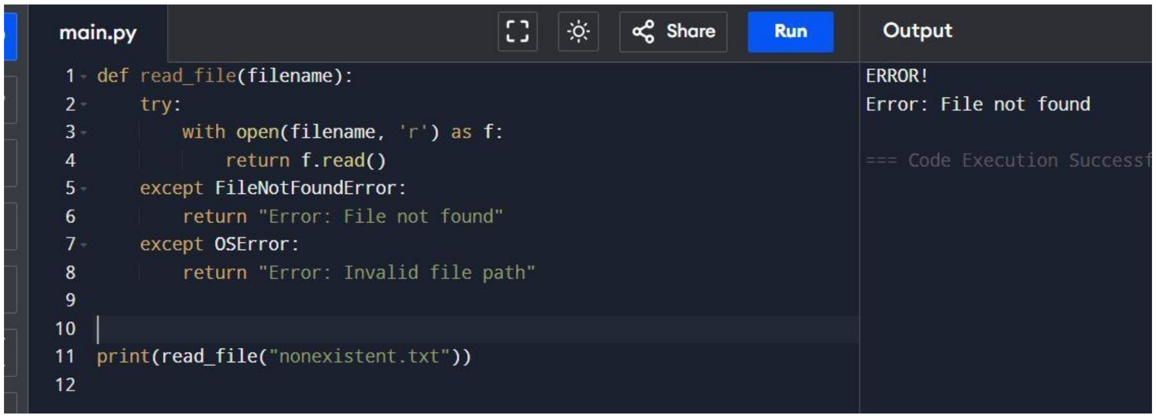
Task 3: Runtime Error – File Not Found

Corrected Code:

```
def read_file(filename):
    try:
        with open(filename, 'r') as f:
```

```
        return f.read()
    except FileNotFoundError:
        return "Error: File not found"
    except OSError:
        return "Error: Invalid file path"
```

Output:
Error: File not found



```
def read_file(filename):
    try:
        with open(filename, 'r') as f:
            return f.read()
    except FileNotFoundError:
        return "Error: File not found"
    except OSError:
        return "Error: Invalid file path"


print(read_file("nonexistent.txt"))
```

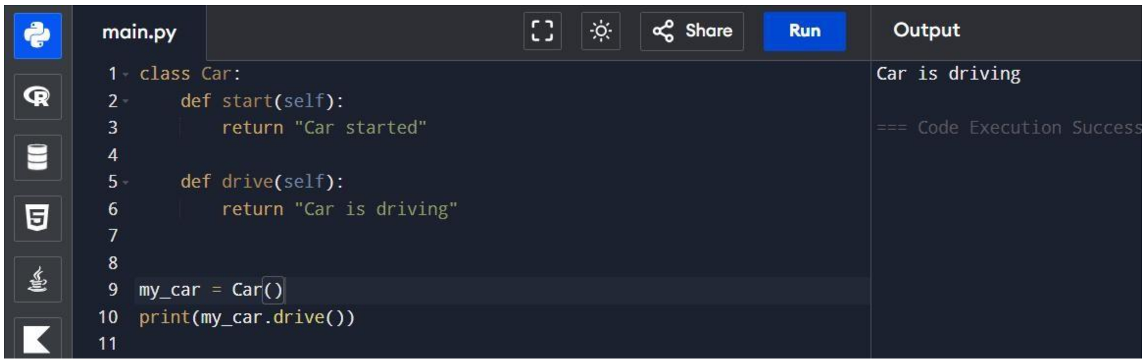
```
ERROR!
Error: File not found

=== Code Execution Successf
```

Task 4: Calling a Non-Existent Method

Corrected Code:

```
class Car:
    def start(self):
        return "Car started"
    def drive(self):
        return "Car is driving"
```

Output:
Car is driving



```
class Car:
    def start(self):
        return "Car started"

    def drive(self):
        return "Car is driving"


my_car = Car()
print(my_car.drive())
```

```
Car is driving

=== Code Execution Success
```

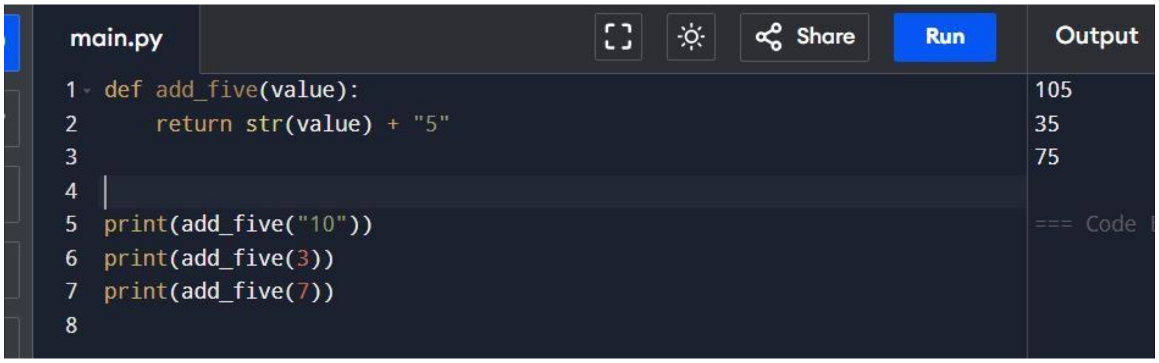
Task 5: TypeError – Mixing Strings and Integers

Solution 1:

```
def add_five(value):
    return int(value) + 5
```

Solution 2:

```
def add_five(value):
    return str(value) + "5"
```



```
def add_five(value):
    return str(value) + "5"


print(add_five("10"))
print(add_five(3))
print(add_five(7))
```

```
105
35
75

=== Code
```

Conclusion

This lab demonstrated how AI-assisted debugging helps identify errors, explain bugs clearly, and suggest safe fixes.