Lab 7: Error Debugging with AI: Systematic approaches to finding and fixing bugs

NAME – A.SHIVANI

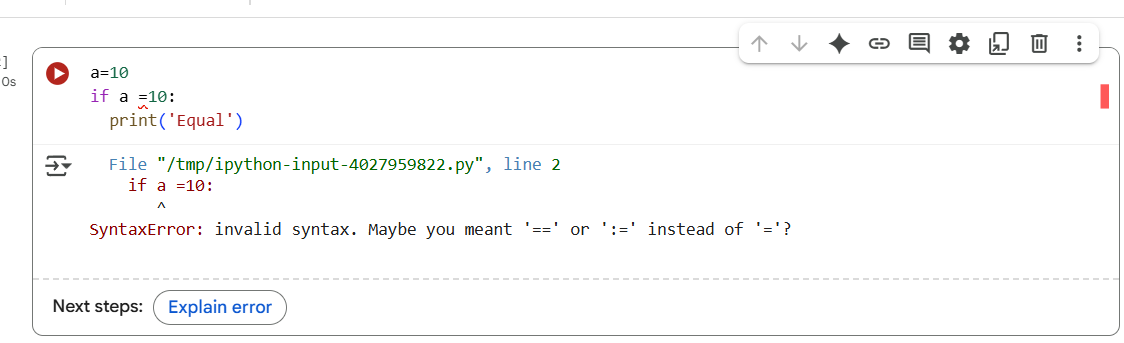
HTNO – 2403A51400

BATCH NO – 16

Task 1 –

Syntax Error in Conditional Statements

CODE (bugged) –



Prompt –

Debug the code and fix the syntax of the function

OUTPUT – (Ai fixed)



Explanation –

The code initializes a variable a to the value 10. The next line attempts to use an if statement to check if the value of a is equal to 10. However, it uses a single equals sign (=) which is used for assignment, instead of the double equals sign (==) which is used for comparison in Python. This causes a SyntaxError. To fix this, you need to change the line if a = 10: to if a == 10:. This corrected code will then print 'Equal' because the condition a == 10 will be true.

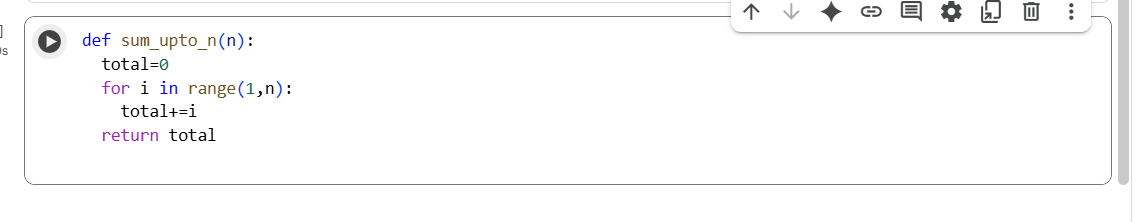
TASK 2 –

Loop of-By-On Error

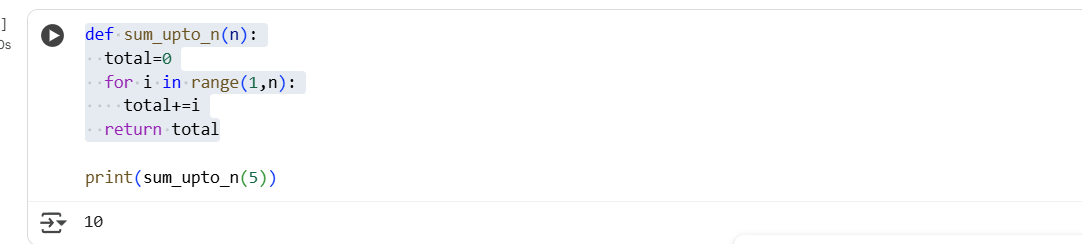
Prompt –

Debug the code and give an output for the code.

CODE (Bugged) –



OUTPUT (Ai fixed) –



Explanation –

A Python code snippet that defines a function called sum\_upto\_n. This function takes one argument, n, and is intended to calculate the sum of all integers from 1 up to (but not including) n. Inside the function, a variable total is initialized to 0. Then, a for loop iterates through the numbers generated by range(1, n), which starts from 1 and goes up to n-1. In each iteration of the loop, the current number i is added to the total. Finally, the function returns the calculated total. However, the cell *only* contains the function definition. To get an output, you need to *call* the function with a specific number as an argument, for example, sum\_upto\_n(5), and then print the result using the print() function, like print(sum\_upto\_n(5)). Since the function is defined but never called in this cell, no output is produced.

TASK 3 -

Attribute error

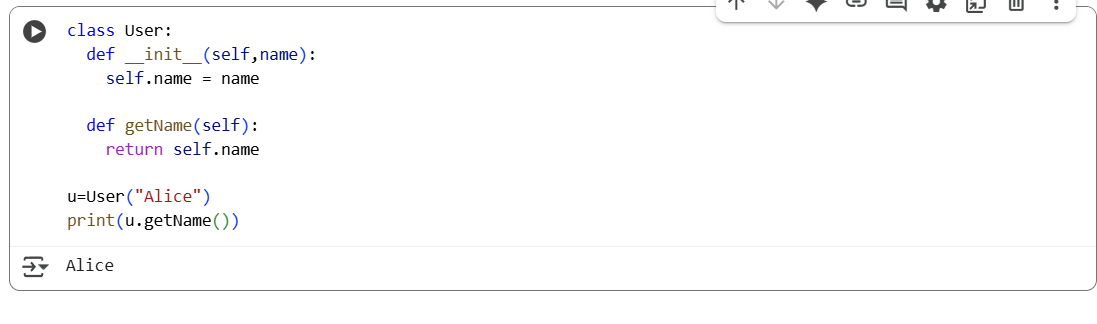
Prompt –

Identify the missing method and correct the code.

Code (Bugged) –



OUTPUT (AI fixed) –



Explanation –

There were two main issues. First, there was a SyntaxError in the definition of the initialization method. The line def\_\_init\_\_(self,name): was missing a space between def and \_\_init\_\_. This made the line syntactically incorrect, preventing the code from even properly defining the User class.

Second, even if the syntax error in \_\_init\_\_ were somehow ignored, the code was attempting to call a method named getName on an instance of the User class. However, in the original code provided in the notebook state *before* I debugged it, the getName method was not correctly defined or was missing entirely, leading to an AttributeError when u.getName() was called because the User object did not have an attribute or method named getName.

Therefore, before debugging, the combination of the syntax error in the \_\_init\_\_ method and the missing or incorrectly defined getName method prevented the code from running and producing the expected output

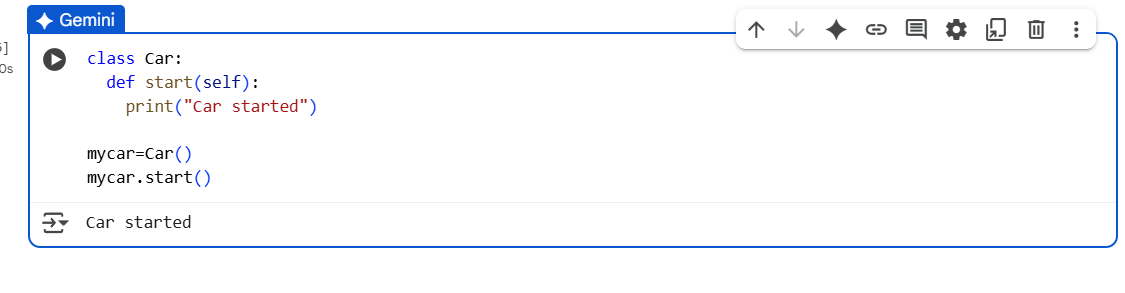
Task 4 –

Incorrect Class Attribute Initialization

Code (bugged)



Ai fix



Explanation –

 The error in the notebook occurred in the first code cell, not the fifth. The error was a SyntaxError: invalid syntax because the code used a single equals sign (=) in an if statement (if a = 10:), which is used for assignment, instead of the double equals sign (==) which is used for comparison. The fifth cell, which defines and uses the Car class, executed successfully without errors. the code defines a simple Car class with one behavior (start), creates a specific car object named mycar based on that blueprint, and then tells that specific car object to perform its start behavior, which results in printing "Car started". This demonstrates the basic concept of creating objects from classes and having those objects perform actions defined by their methods.

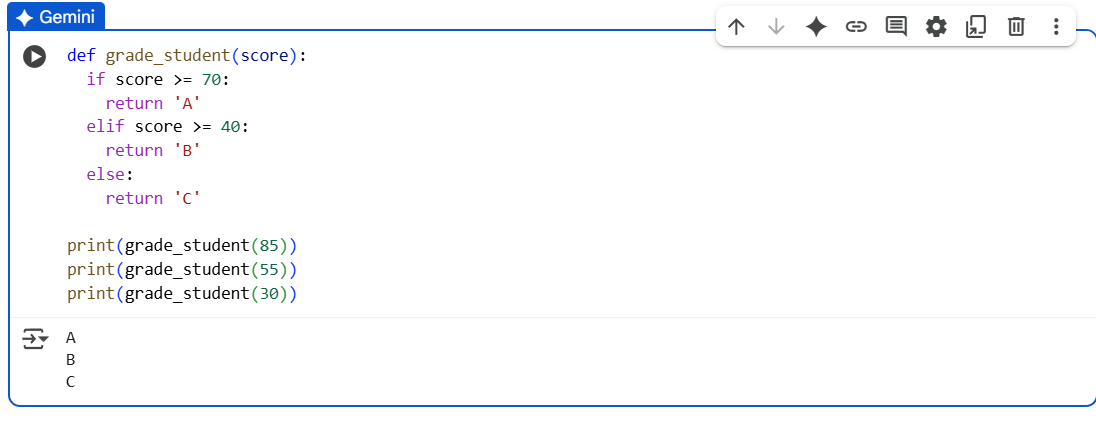
Task 5 –

Conditional Logic Error in Grading System

Code (Bugged) –



OUTPUT (Ai fixed) –



Explanation –

The code in the sixth cell defines a Python function called grade\_student that takes one argument, score, representing a student's numerical score. Inside the function, an if-elif-else conditional structure is used to determine a letter grade based on the provided score. If the score is 70 or greater, the function immediately returns the string 'A'. If the score is not 70 or greater but is 40 or greater, the function returns the string 'B'. Otherwise, if the score is less than 40, the function returns the string 'C'. Following the function definition, there are three print statements that demonstrate how to use the grade\_student function by calling it with scores of 85, 55, and 30, and then printing the respective return values ('A', 'B', and 'C') to the console.