Assignment 7.3 Ai Assisted Coding

Htno:2303a51288

Btno:05

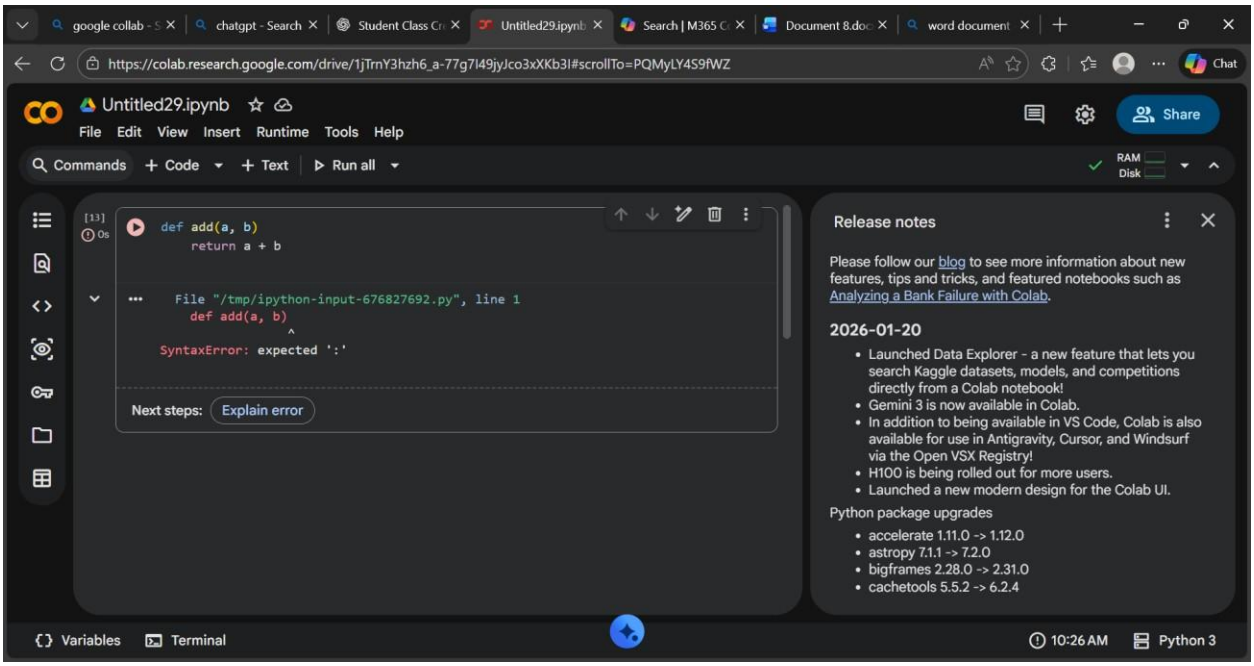
Task 1: Fixing Syntax Errors

Prompt: The following Python function has a syntax error. Identify the issue and correct it. Also explain what the syntax error is.

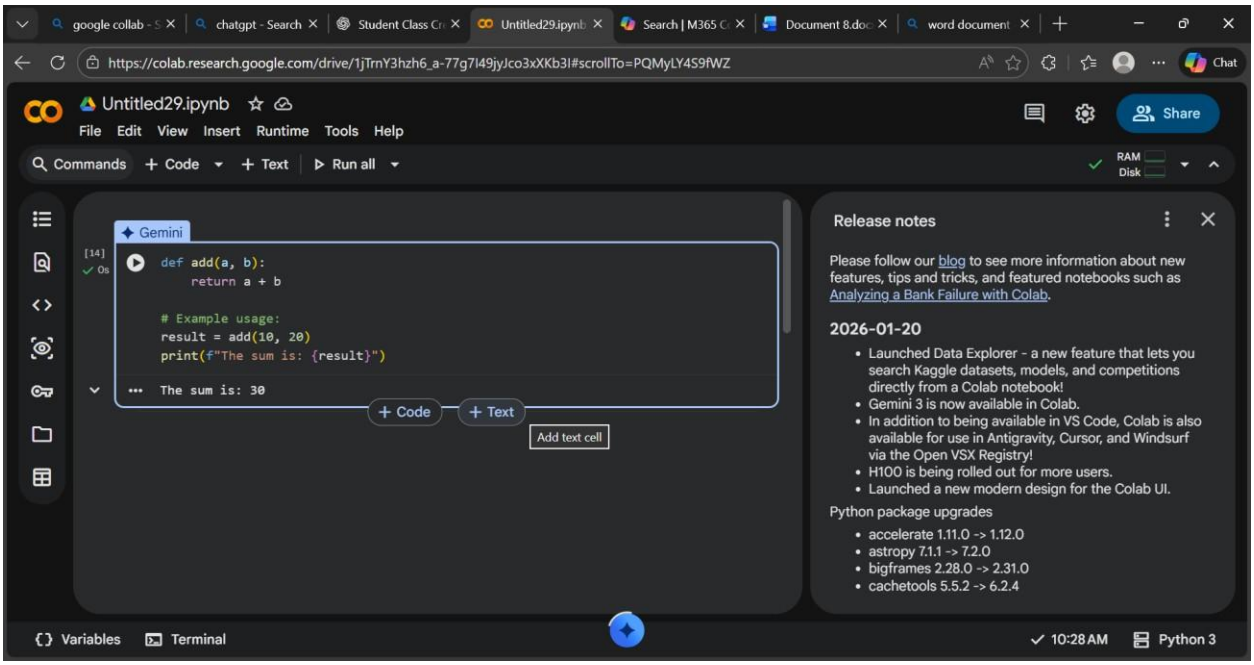
def add(a, b)

return a + b Input:

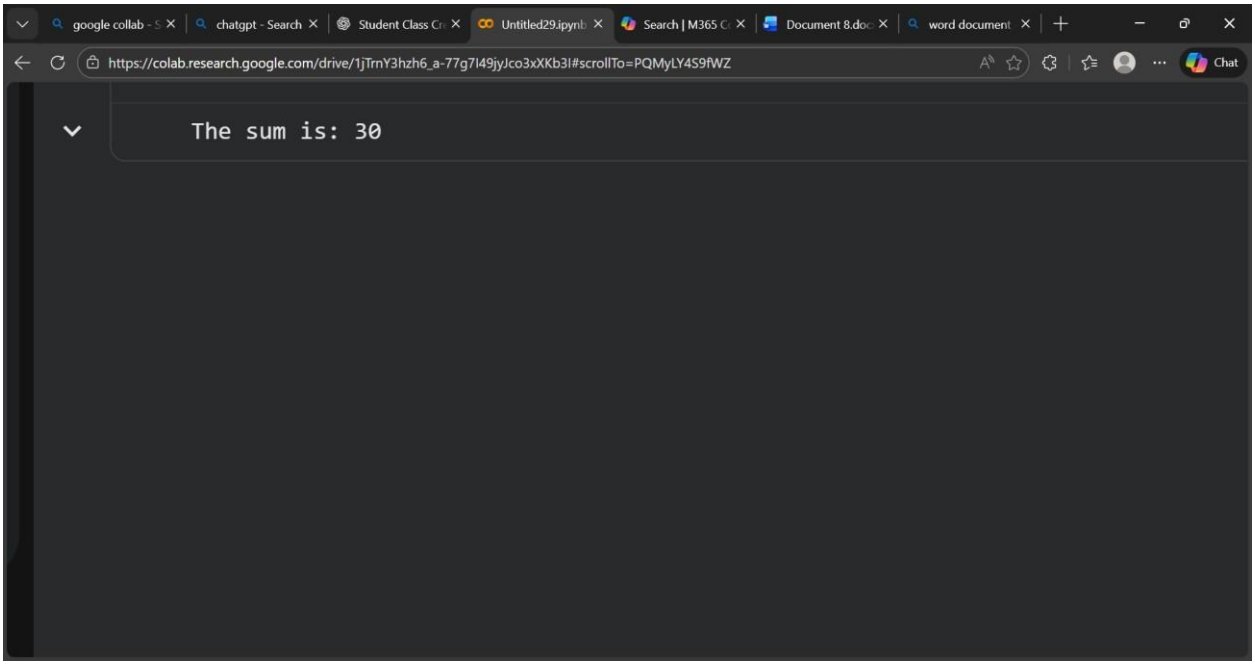
Bug Code:



2) corrected code:



Output:

- 

Explanation:

- In Python, a colon ： is required after defining a function header.
- Without the colon, Python cannot recognize the start of the function block, causing a **SyntaxError**.
- AI correctly identified the missing colon and fixed the function definition.

Task 2: Debugging Logic Errors in Loops

Prompt: The following Python loop runs infinitely. Identify the logic error, correct the loop, and explain the issue.

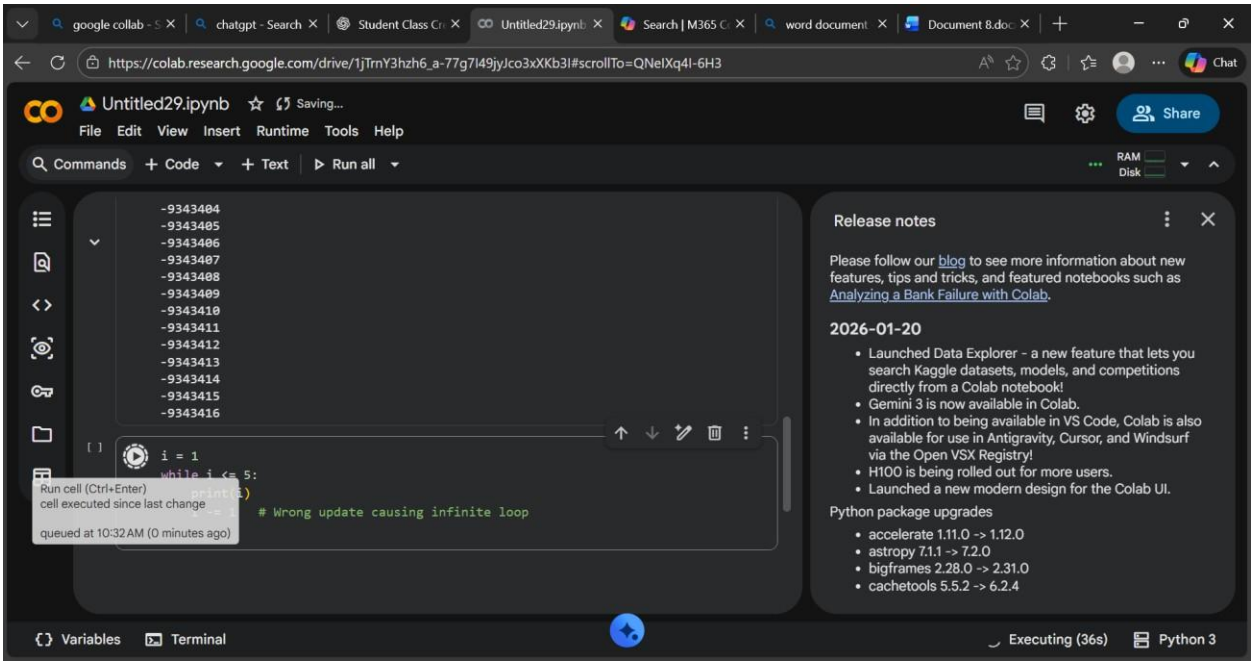
i = 1 while i

<= 5:

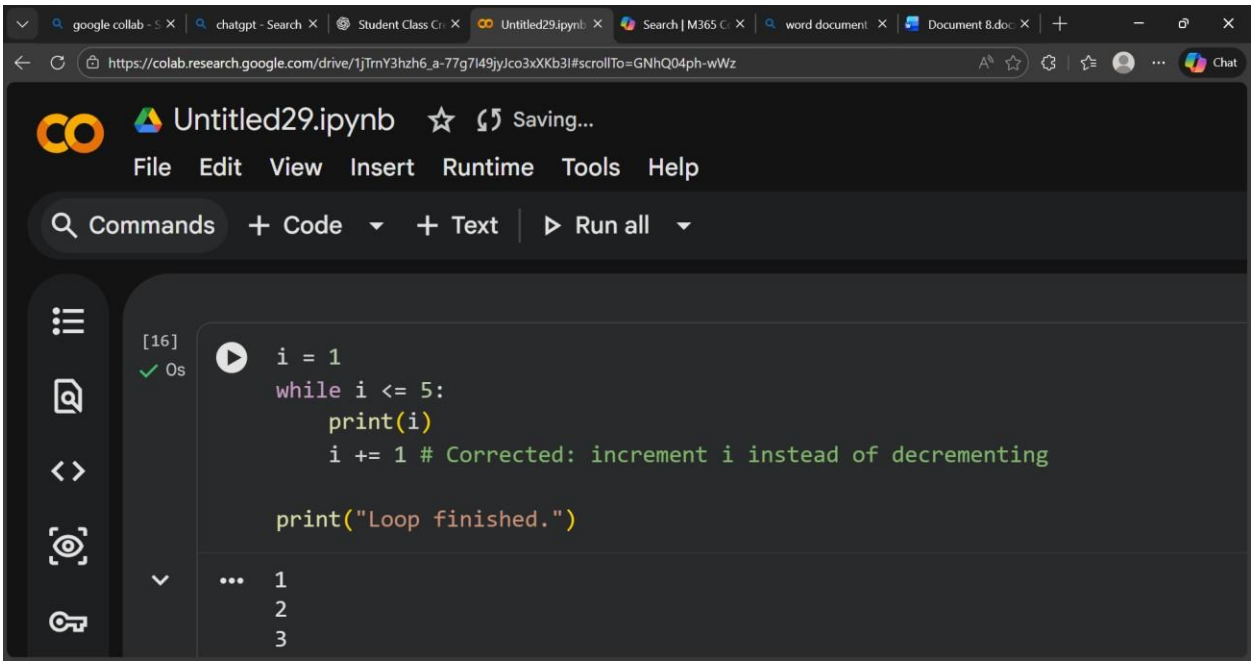
print(i)

i -= 1

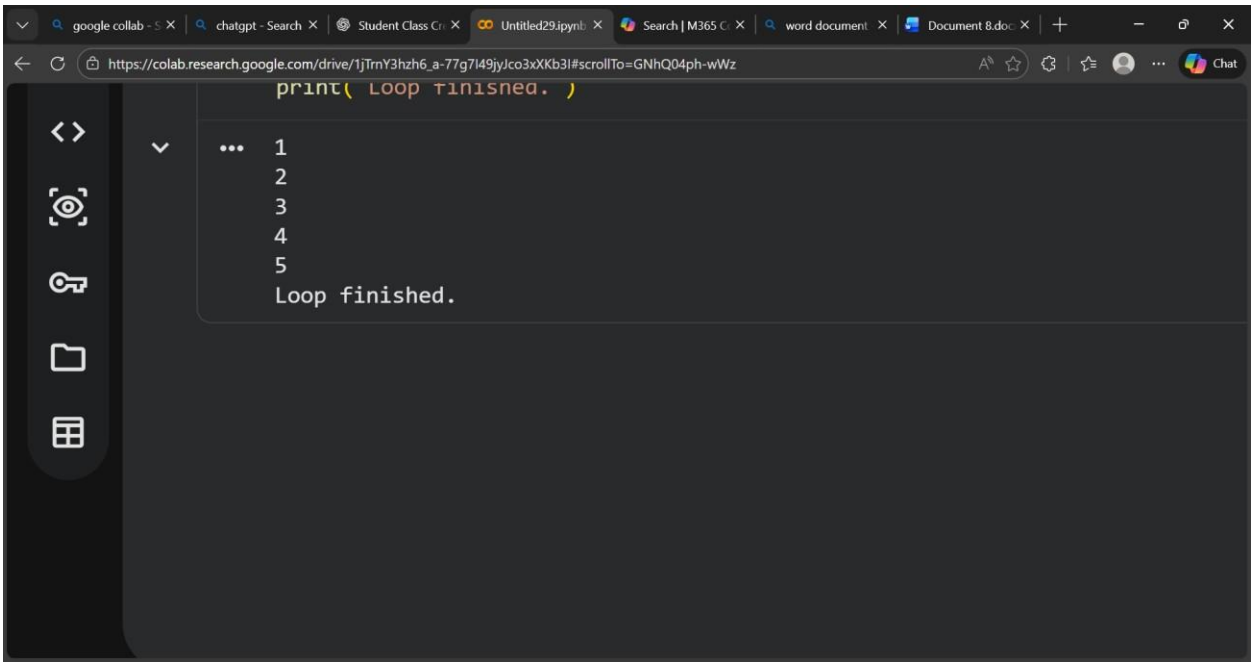
Input: Bug code:



Corrected code:



Output:



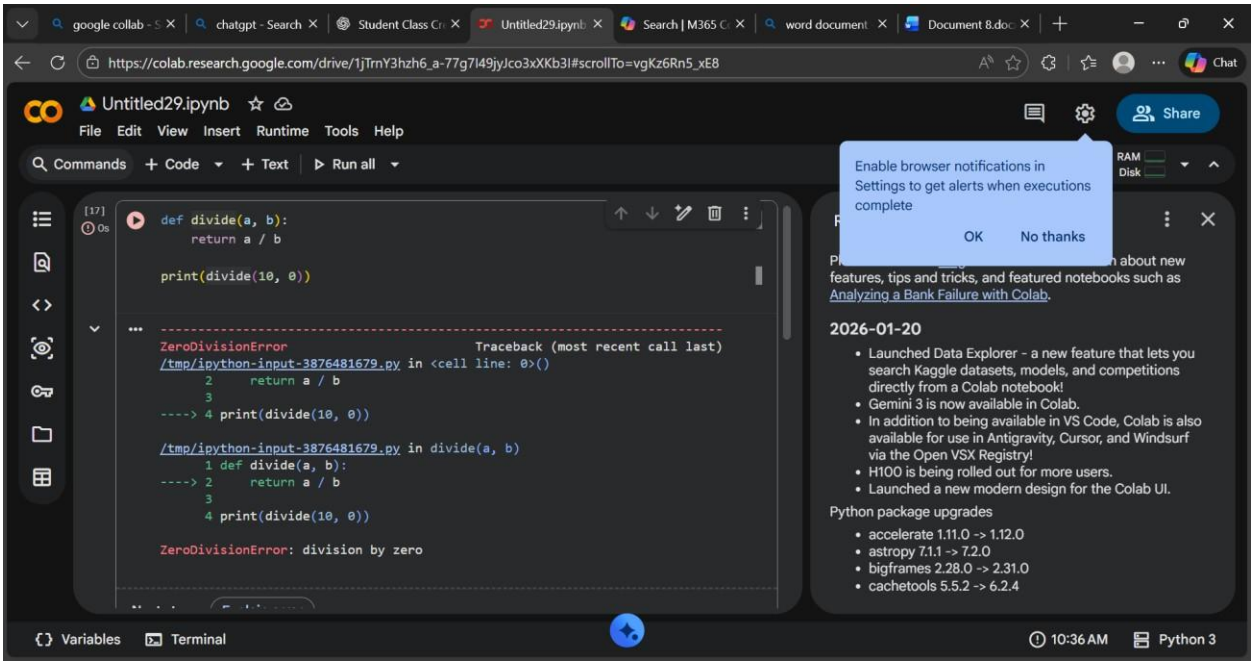
Explantion: The variable `i` was decreasing (`i -= 1`) while the condition required it to increase, causing an infinite loop.

Changing it to `i += 1` allows the loop to reach the stopping condition and terminate correctly.

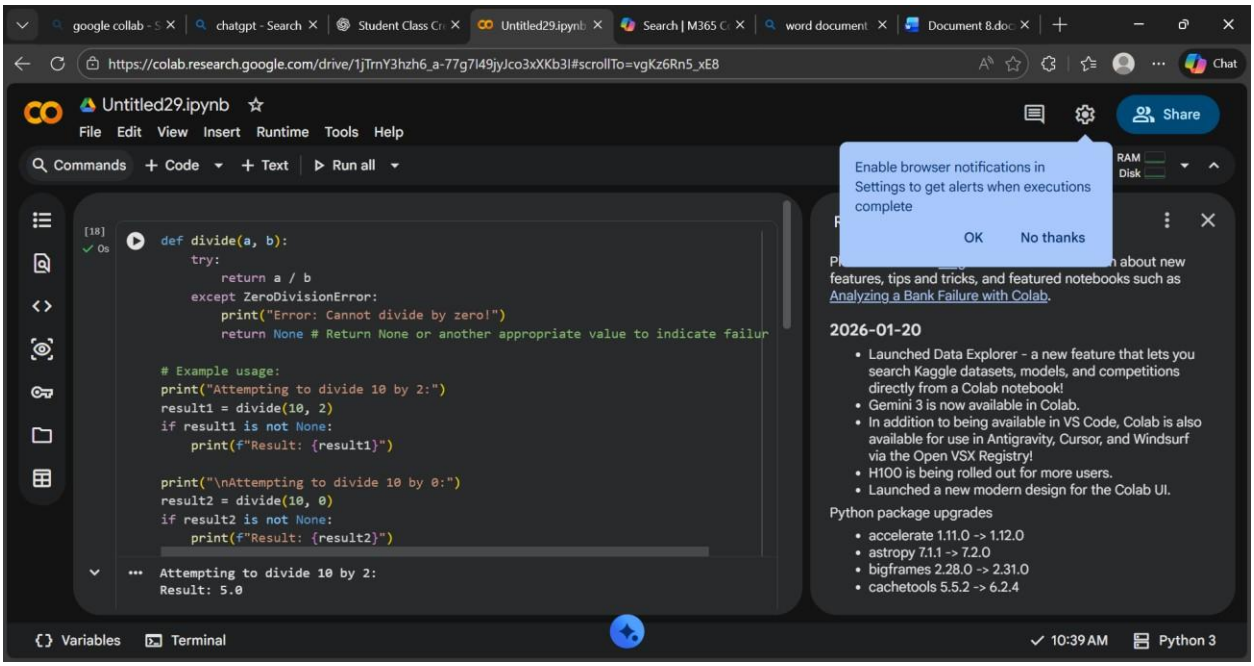
Task 3: Handling Runtime Errors (Division by Zero)

Prompt: This Python code causes a runtime error. Identify the problem, fix it using tryexcept, and explain the issue. def divide(a, b): return a / b print(divide(10, 0))

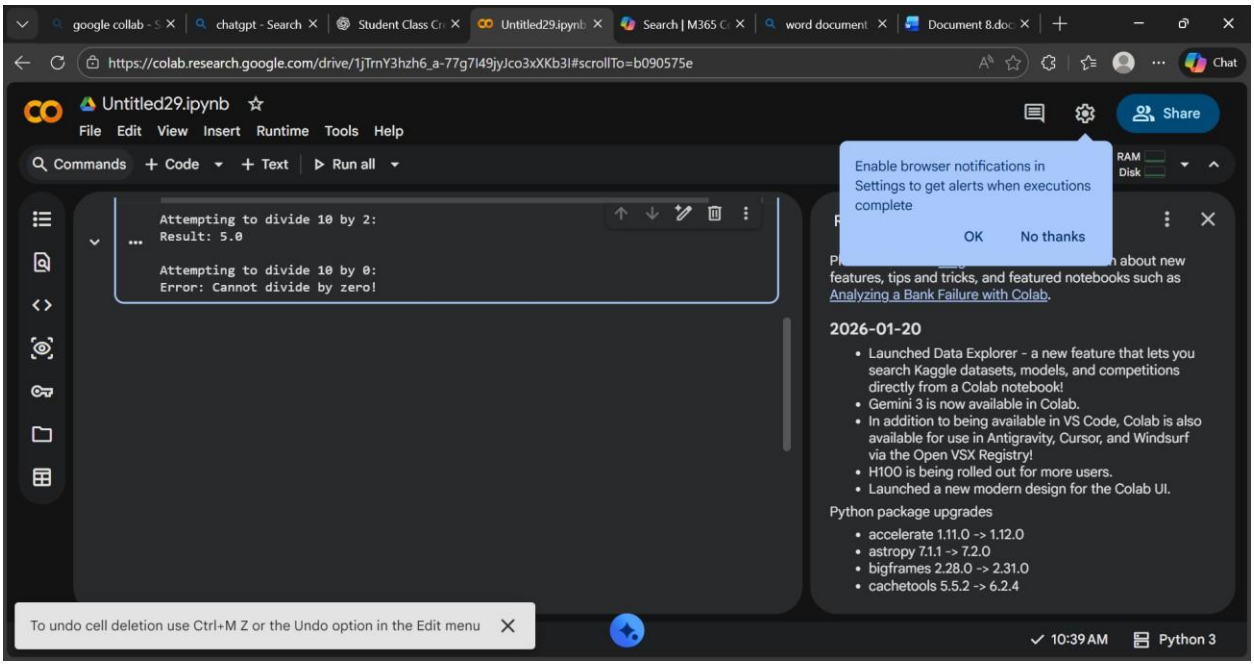
Input:Bug Code



Corrected Code:



Output:



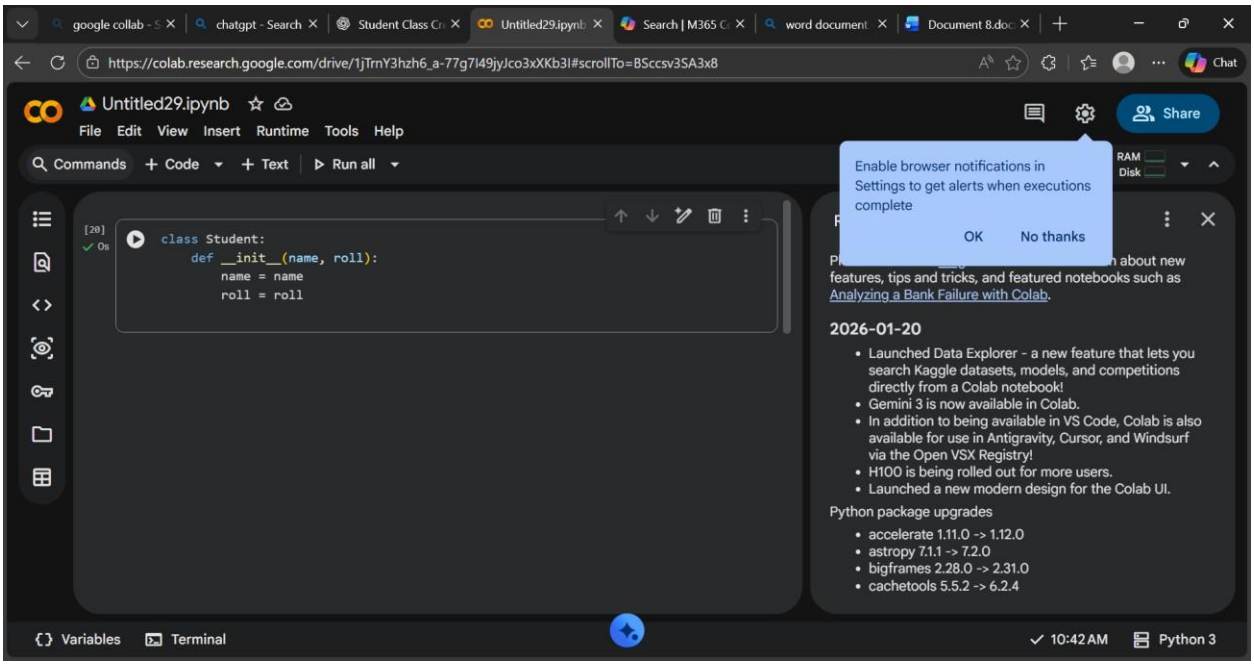
Explanation: he program crashes because division by zero is not allowed in Python, causing a `ZeroDivisionError`.

Using `try-except` prevents the crash and safely handles the error.

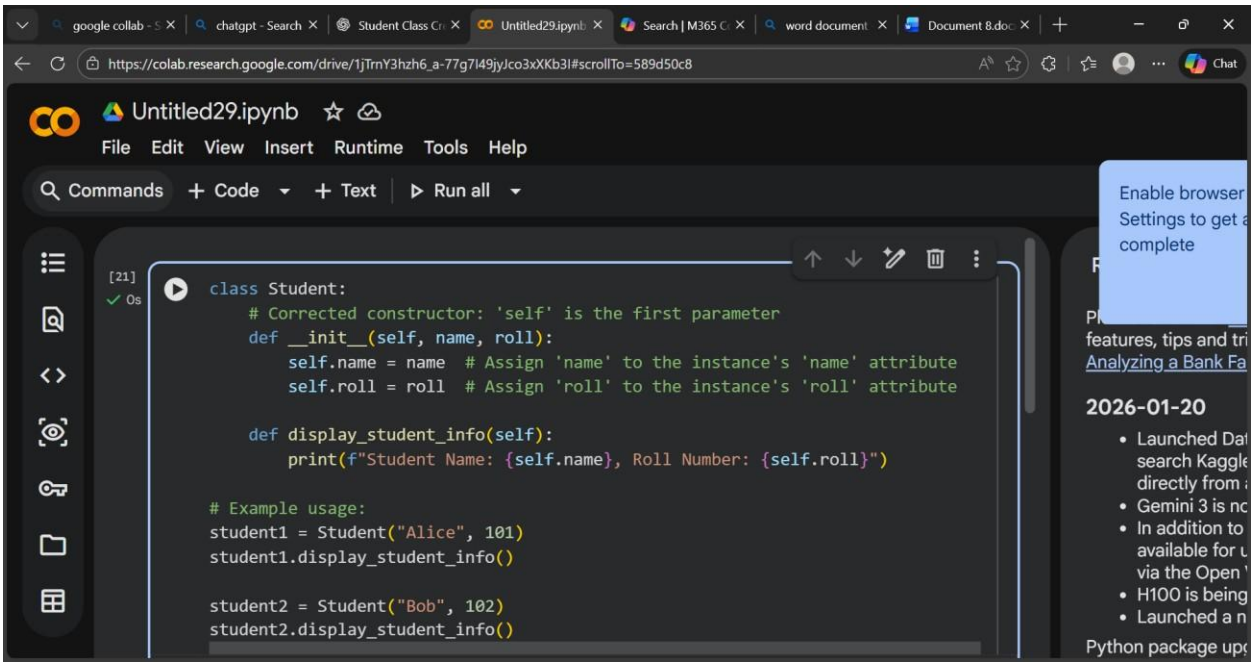
Task 4: Debugging Class Definition Errors

Prompt: The following Python class has an error in the constructor. Identify the issue, correct the class definition, and explain why the fix is needed.
class Student: def **init**(name, roll): name = name roll = roll

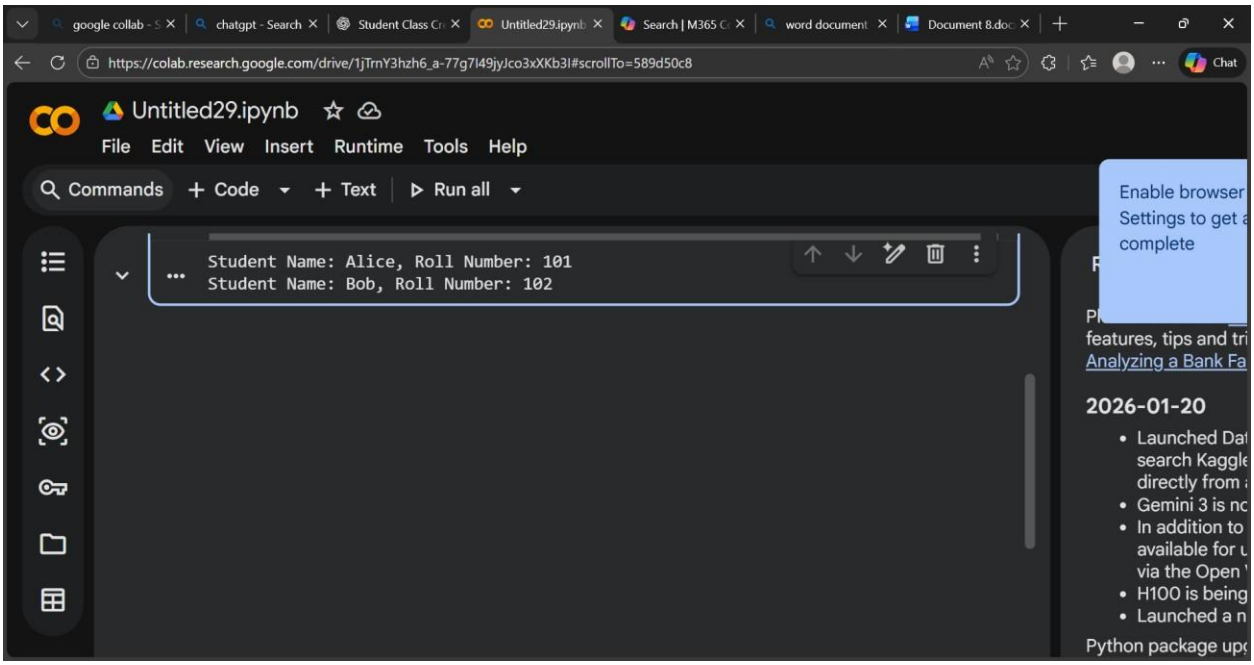
Input:Bug Code



Corrected code:



Output:



Explanation: The constructor was missing the `self` parameter, which is required to refer to the object instance.

Using `self.name` and `self.roll` stores values inside the object properly. Task 5:

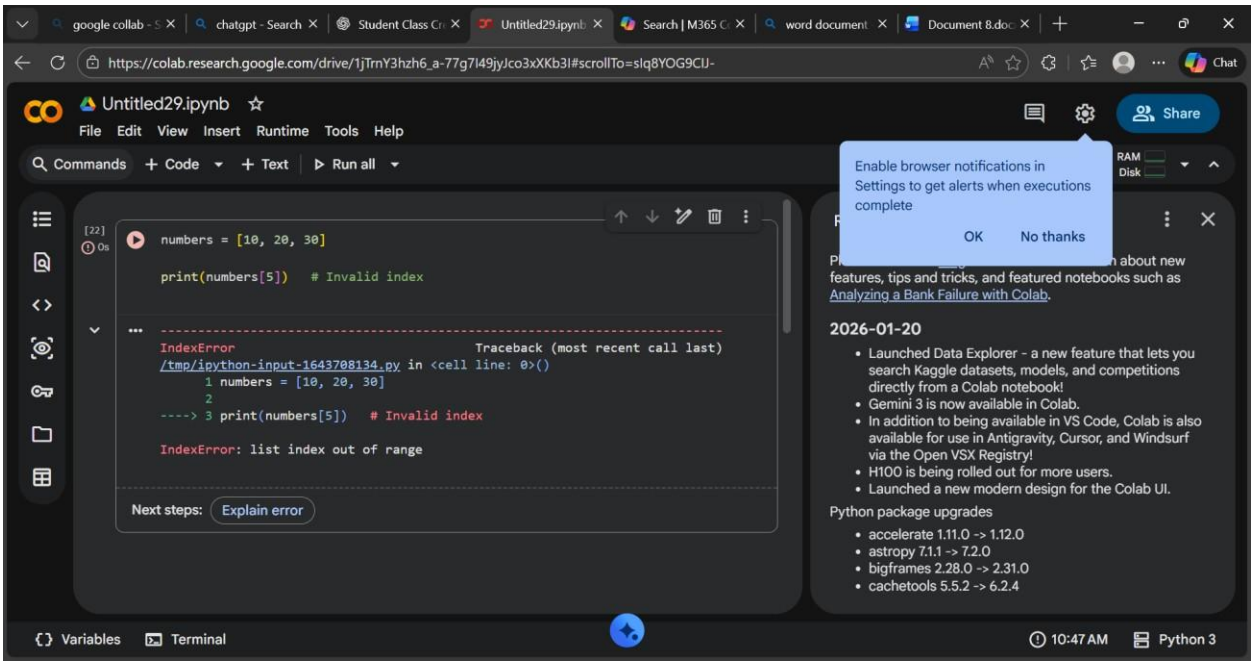
Resolving Index Errors in Lists

Prompt: This Python code causes an IndexError. Identify the issue, correct the code using safe access methods, and explain the problem.

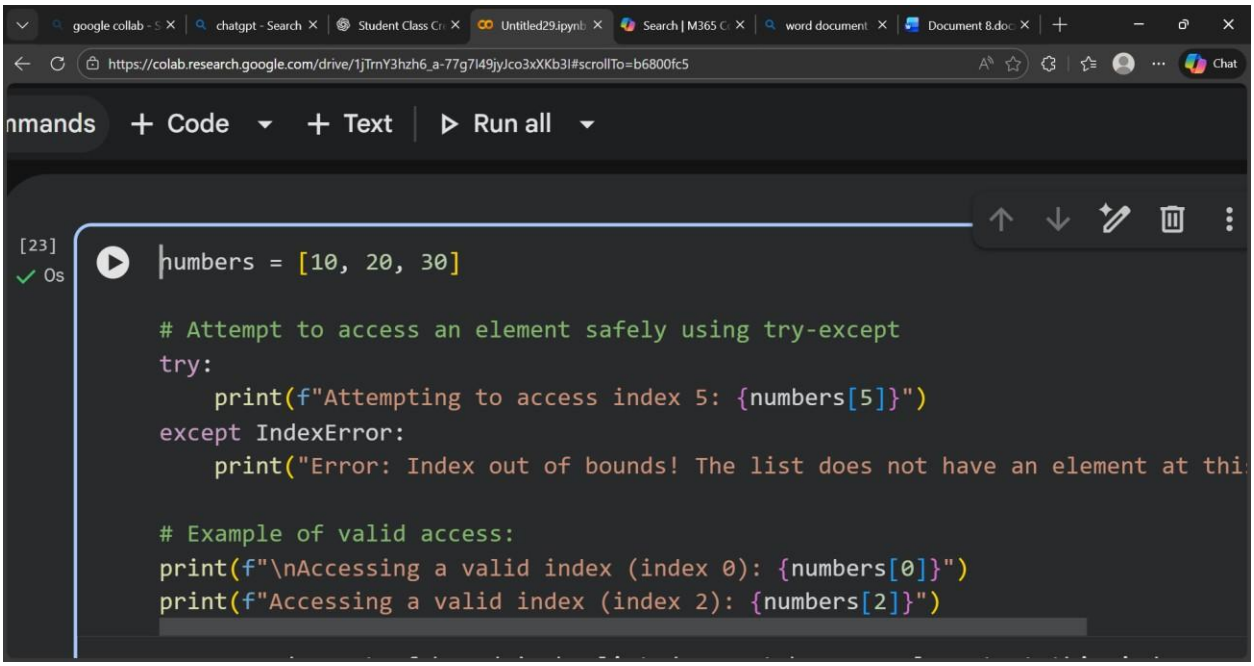
numbers = [10, 20, 30]

(numbers[5])

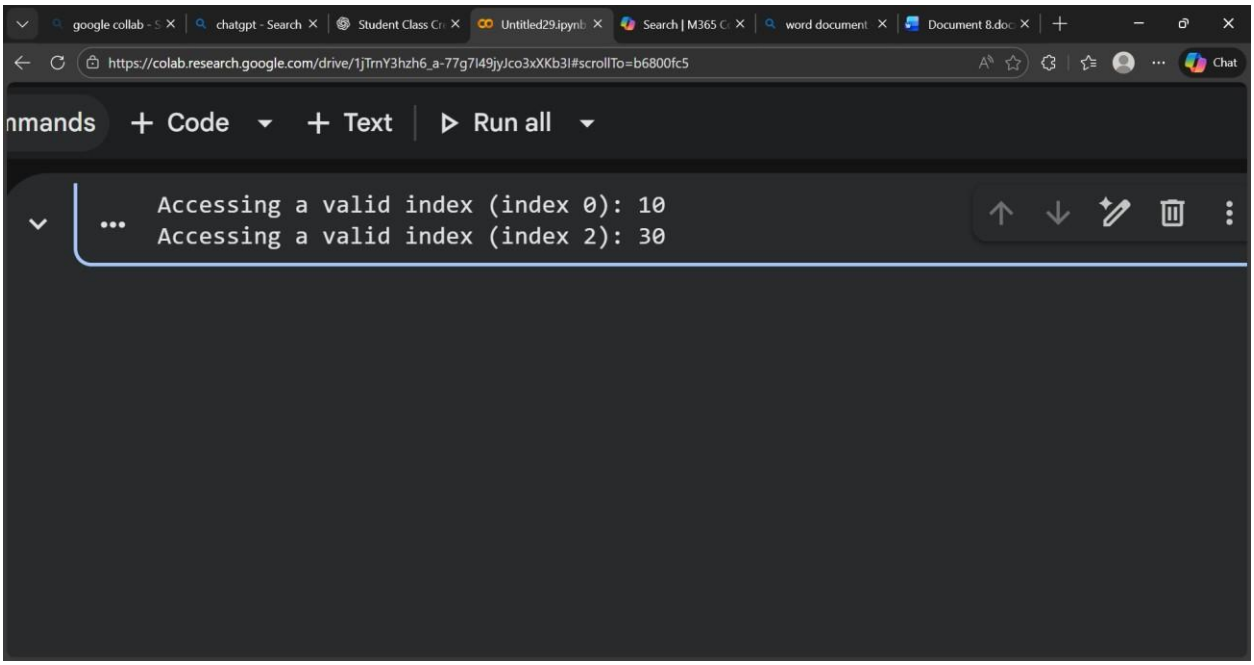
Input:Bug code



Corrected Code:



Output:



Explanation: The program tried to access an index that does not exist in the list, causing an `IndexError`.

Using `len()` to check bounds prevents the program from crashing.