**AI ASSIGNMENT-7.4 NAME:D.ANKITHA**

**ROLL NO:2503A51L09**

**Task Description #1:**

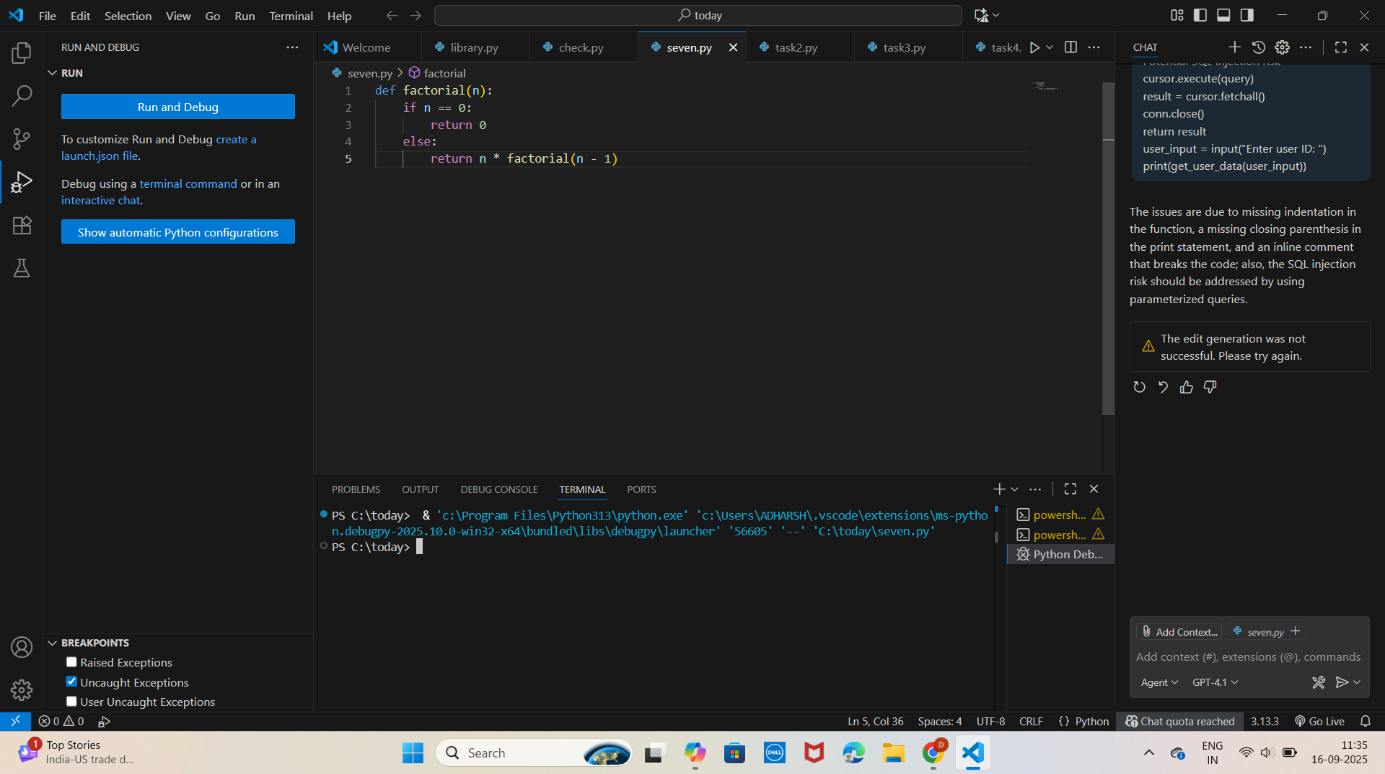
• Introduce a buggy Python function that calculates the factorial of a number using recursion. Use Copilot or Cursor AI to detect and fix the logical or syntax errors.

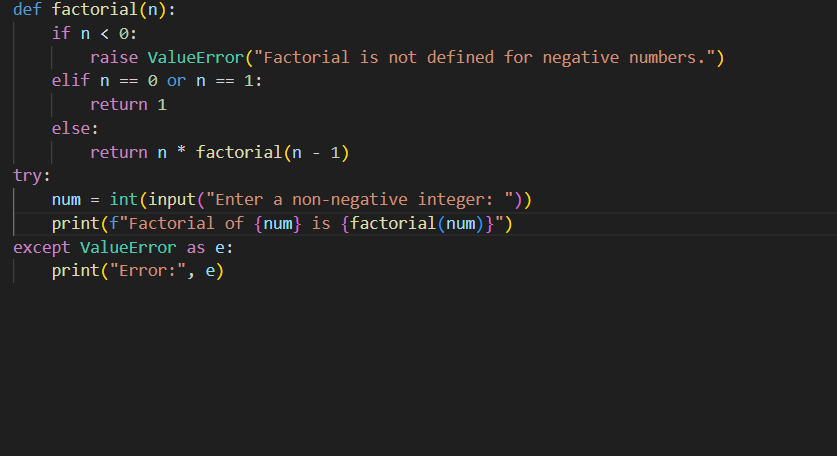
**Expected Outcome #1:**

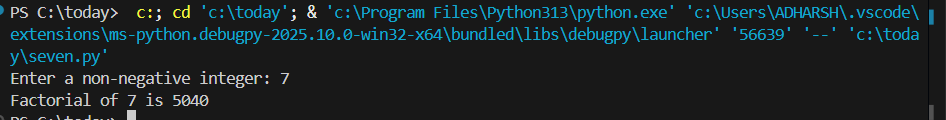
• Copilot or Cursor AI correctly identifies missing base condition or incorrect recursive call and suggests a functional factorial implementation.

**OUTPUT:**

**BUGGY CODE:**

****

**CORRECTED CODE:**

****

**OBSERVATION**: Negative inputs cause infinite recursion and eventually a RecursionError.AI generated a corrected code with error exception which does not allow negative values.

* Returning 0 for n == 0 contradicts the mathematical definition of factorial(0) =1
* **Correct Base Case**: Returns 1 for both 0 and 1, aligning with factorial rules

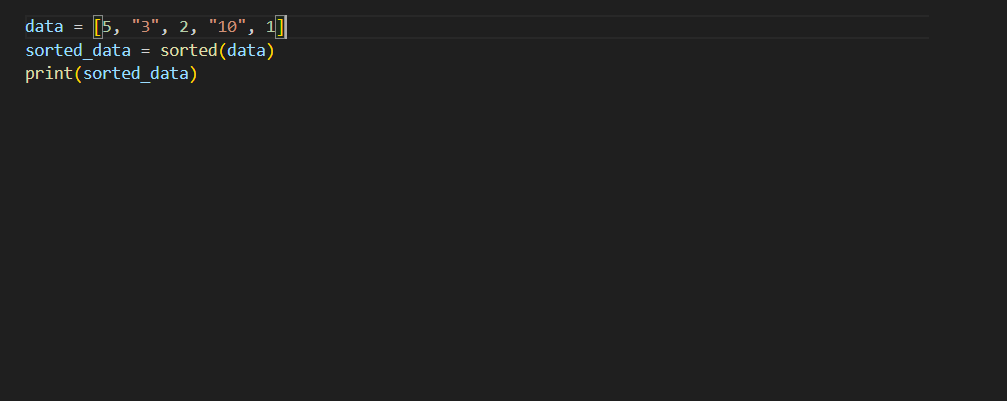
**Task Description #2:**

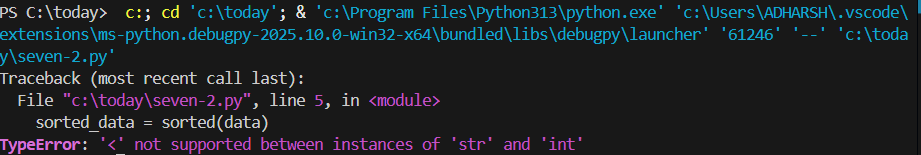
• Provide a list sorting function that fails due to a type error (e.g., sorting list with mixed integers and strings). Prompt AI to detect the issue and fix the code for consistent sorting.

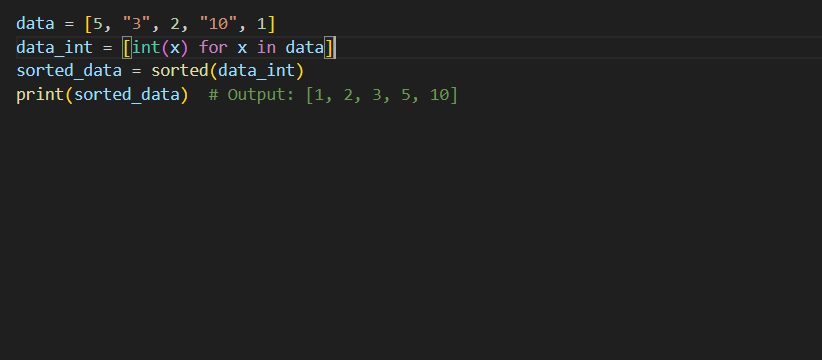
**Expected Outcome #2:**

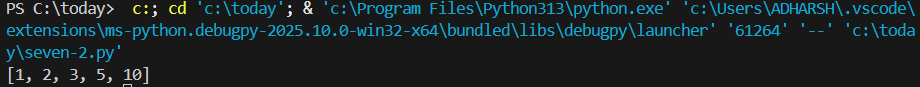
• AI detects the type inconsistency and either filters or converts list elements, ensuring successful sorting without a crash.

**BUGGY CODE:**

****

****

**CORRECTED CODE:**

****

**OBSERVATION:Mixed datatypes are given as input.AI has converted them into one datatype and sorted the data.**

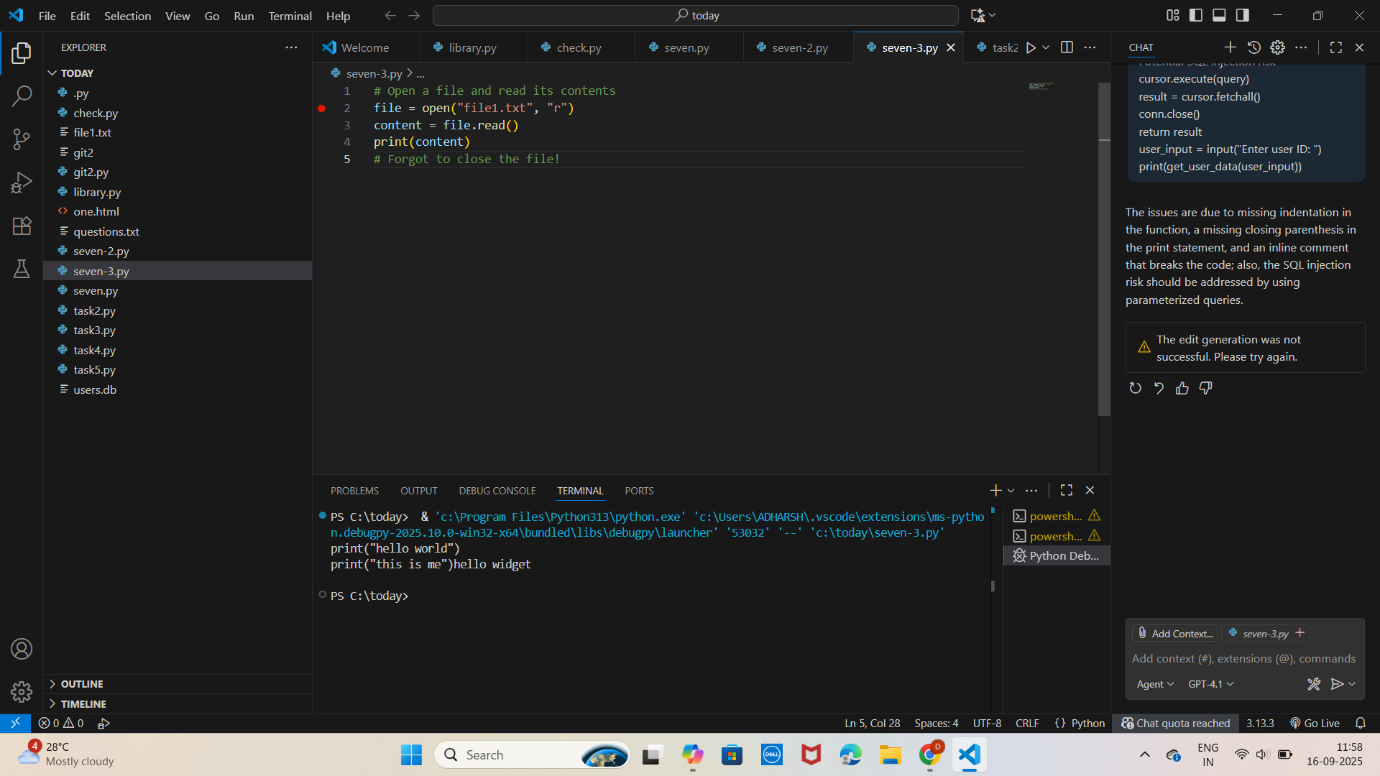
**Task Description #3:**

• Write a Python snippet for file handling that opens a file but forgets to close it. Ask Copilot or Cursor AI to improve it using the best practice (e.g., with open() block).

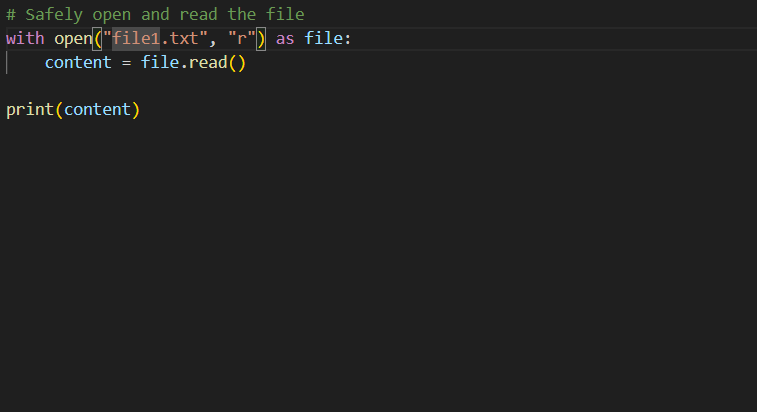
**Expected Outcome #3:**

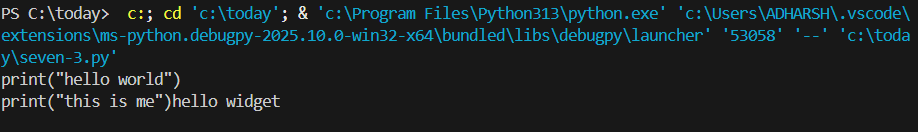
• AI refactors the code to use a context manager, preventing resource leakage and runtime warnings.

**BUGGY CODE:**

****

**CORRECTED CODE:**

****

****

**OBSERVATION:**

The file.close() is missing so, file remains open after reading, which can:

Leak system resources,Lock the file (especially on Windows),

Trigger runtime warnings or errors in larger applications

**No Exception Handling:** If the file doesn't exist or can't be read, the code will crash without a fallback.

**Using a Context Manager (with open)**:

Automatically closes the file when the block exits—even if an error occurs

Prevents resource leakage and improves reliability.

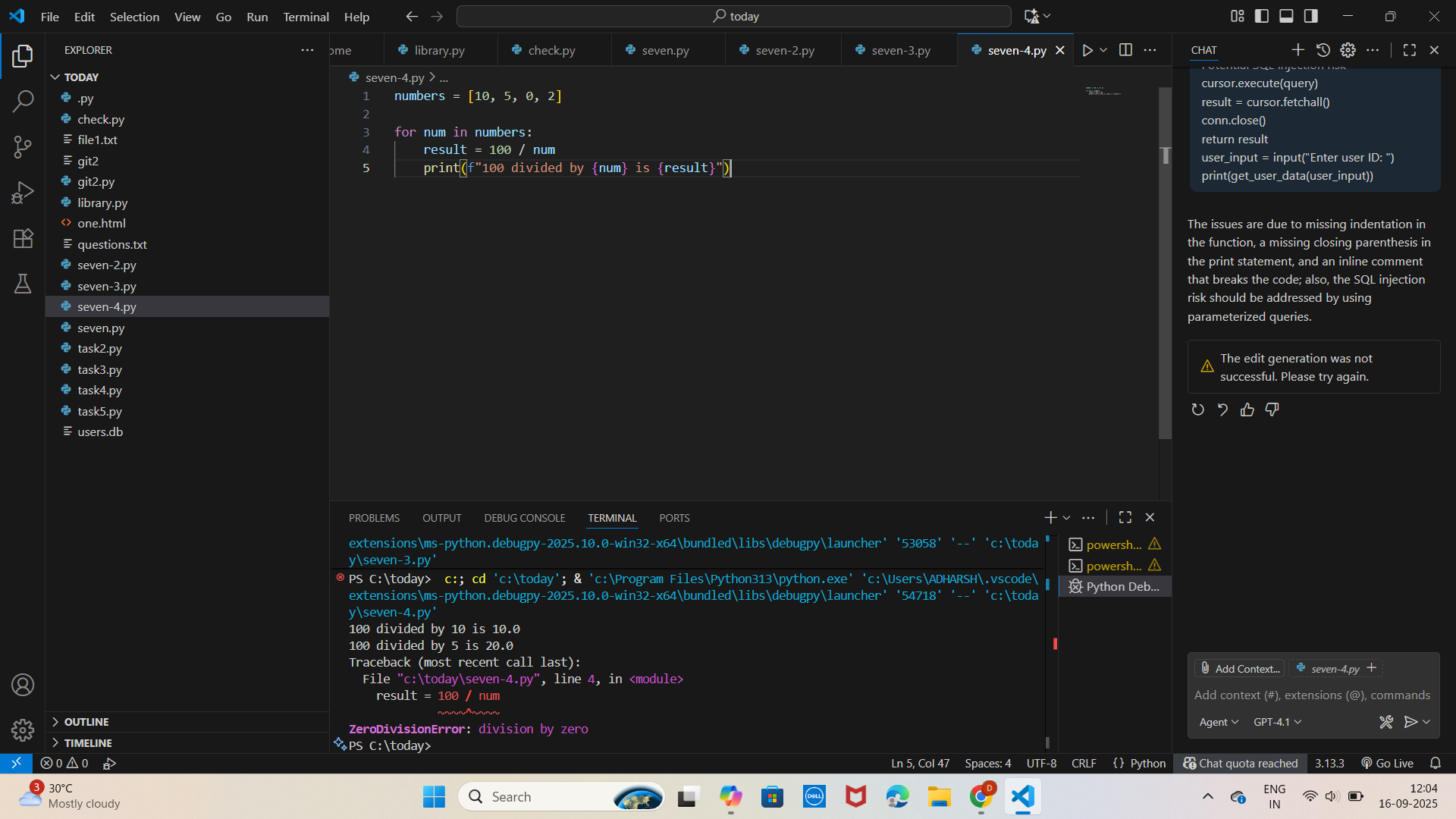
**Task Description #4:**

**•** Provide a piece of code with a ZeroDivisionError inside a loop. Ask AI to add error handling using try-except and continue execution safely.

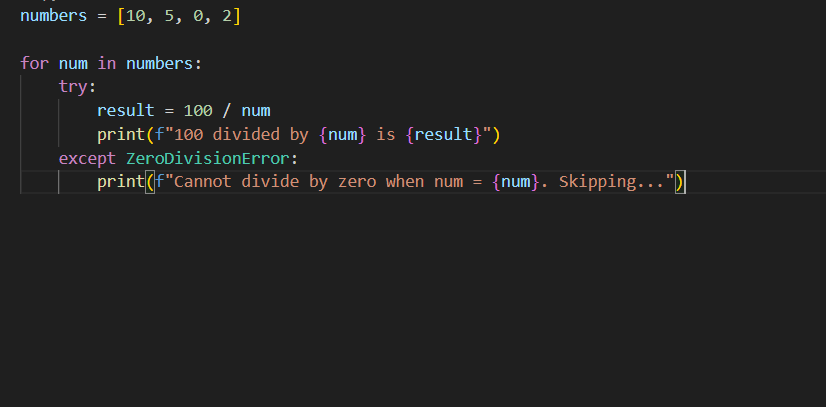
**Expected Outcome #4:**

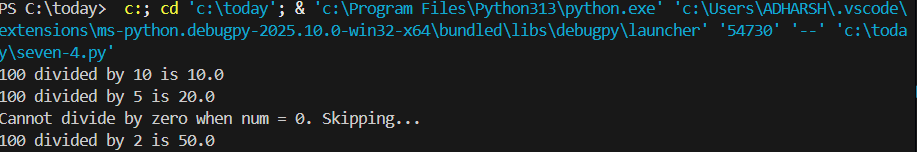
**•** Copilot adds a try-except block around the risky operation, preventing crashes and printing a meaningful error message.

**BUGGY CODE:**

****

**CORRECTED CODE:**

****

****

**OBSERVATION:**

**There is no Error Handling.**

**Poor User Feedback**: No indication of what went wrong or which value caused the issue.

**Try Except block** **added:**isolates risky operation and catches the specific error.

**Improved Feedback**: Prints a clear message when an error occurs, aiding debugging and user understanding.

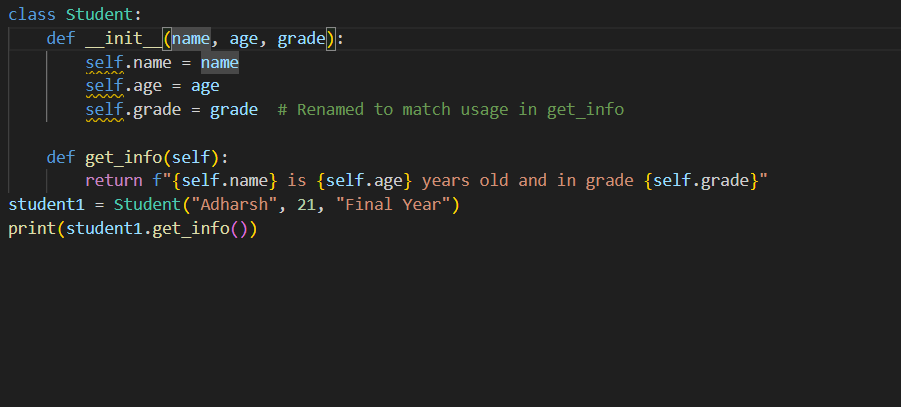
**Task Description #5:**

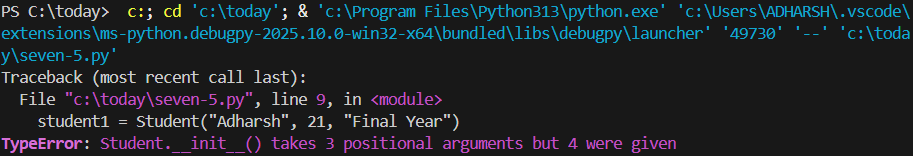
**•** Include a buggy class definition with incorrect \_\_init\_\_ parameters or attribute references. Ask AI to analyze and correct the constructor and attribute usage.

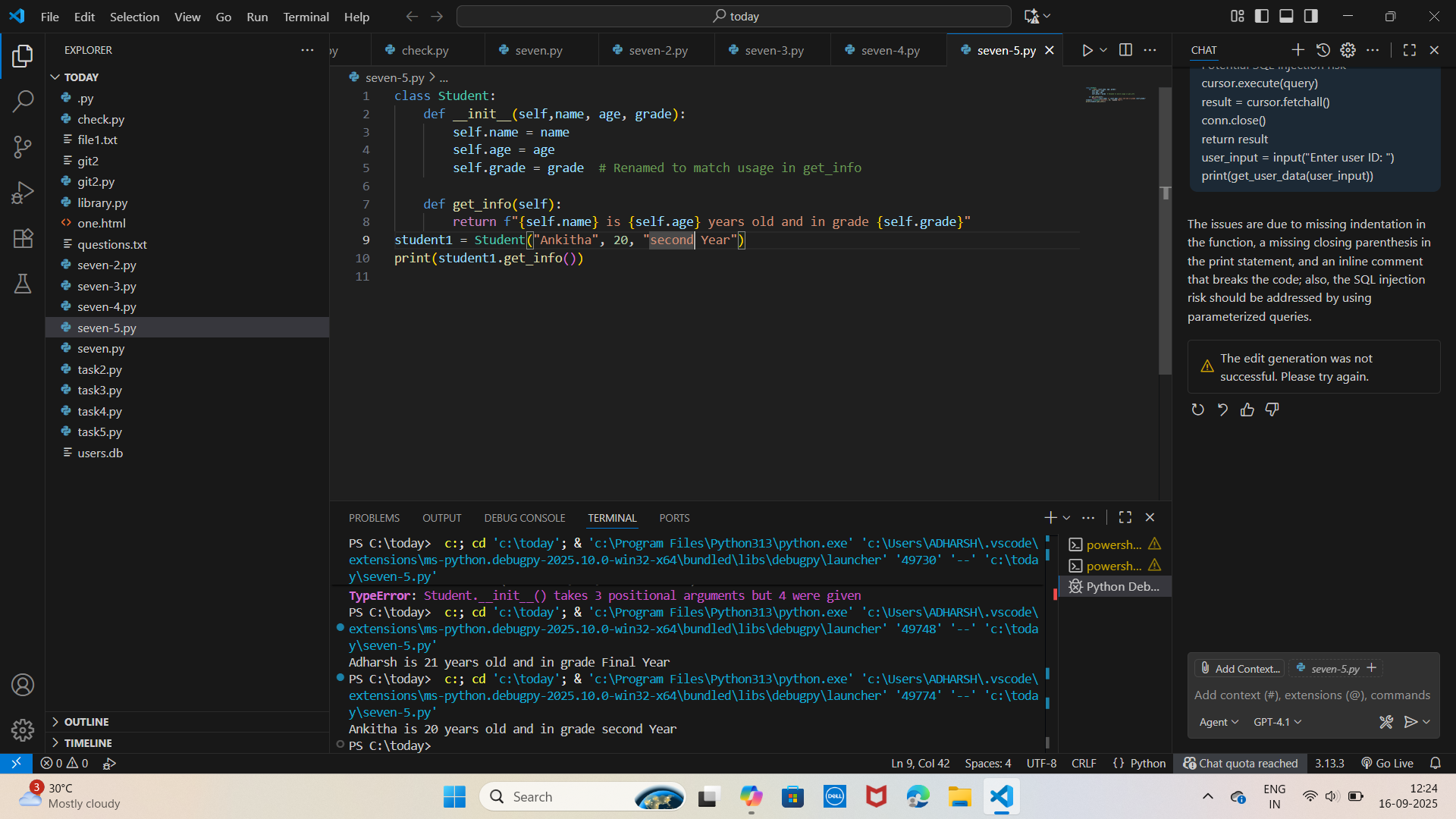
**Expected Outcome #5:**

**•** Copilot identifies mismatched parameters or missing self references and rewrites the class with accurate initialization and usage.

**BUGGY CODE:**

****

****

****

**OBSERVATION:**

* Missing self in \_\_init\_\_ parameters: Python requires self as the first argument in instance methods to refer to the object itself.
* self is the reference to the current instance—required in all instance methods.