Assignment-7.4

Task01: Task 1: Debugging a Recursive Calculation Module

Scenario You are maintaining a utility module in a software project that performs mathematical computations. One function is meant to calculate the factorial of a number, but users are reporting crashes or incorrect outputs.

Task Description :You are given a Python function intended to calculate the factorial of a number using recursion, but it contains logical or syntactical errors (such as a missing base condition or incorrect recursive call).

Use GitHub Copilot or Cursor AI to:

- **Analyze the faulty code**
- **Identify the exact cause of the error**
- **Suggest and apply corrections to make the function work**

Correctly Document how the AI detected the issue and what changes were made.

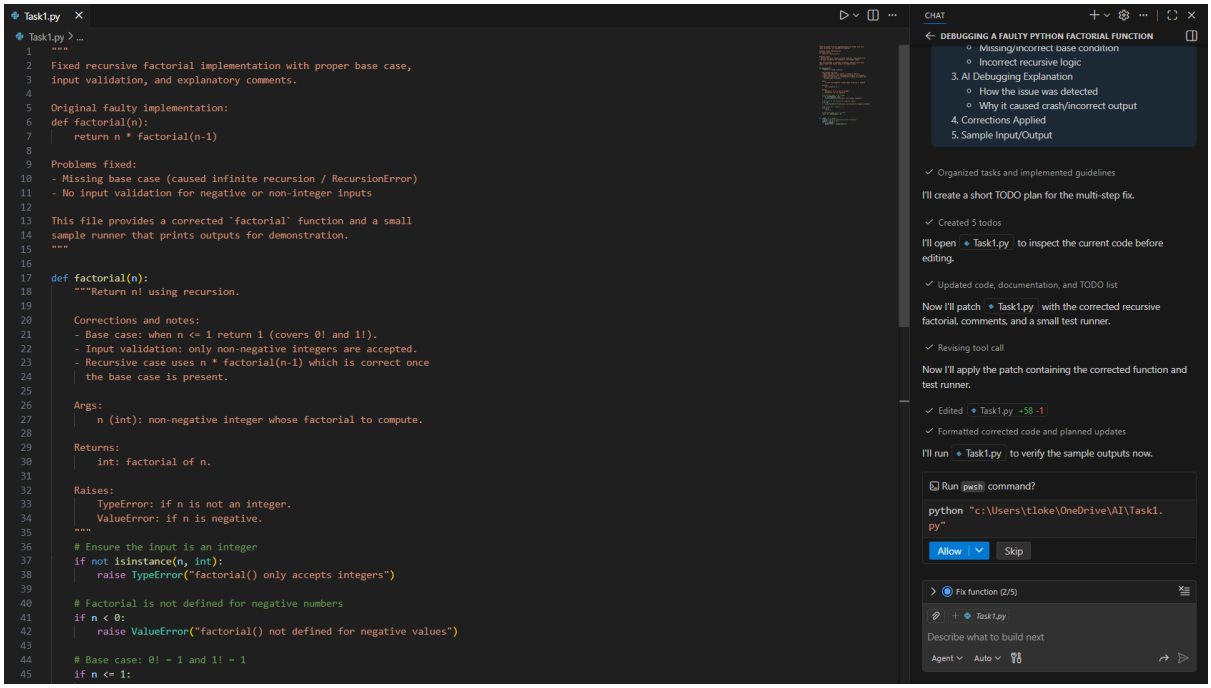
Expected Outcome

- **A corrected recursive factorial function**
- **AI-generated explanation identifying:**

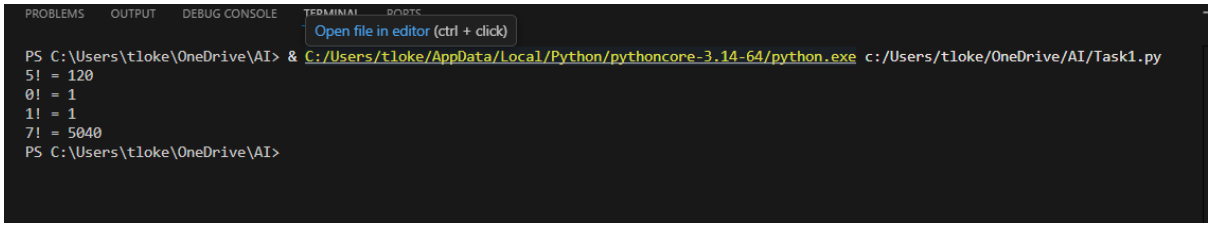
o The missing or incorrect base case

o The corrected recursive logic

Sample input/output demonstrating correct execution



OUTPUT:



TASK-02:

Fixing Data Type Errors in a Sorting Utility

Scenario You are developing a data processing script that sorts user input values. The program crashes when users enter mixed data types.

Task Description

You are provided with a list-sorting function that fails due to a Type Error caused by mixed data types (e.g., integers and strings).Use GitHub Copilot or Cursor AI to:

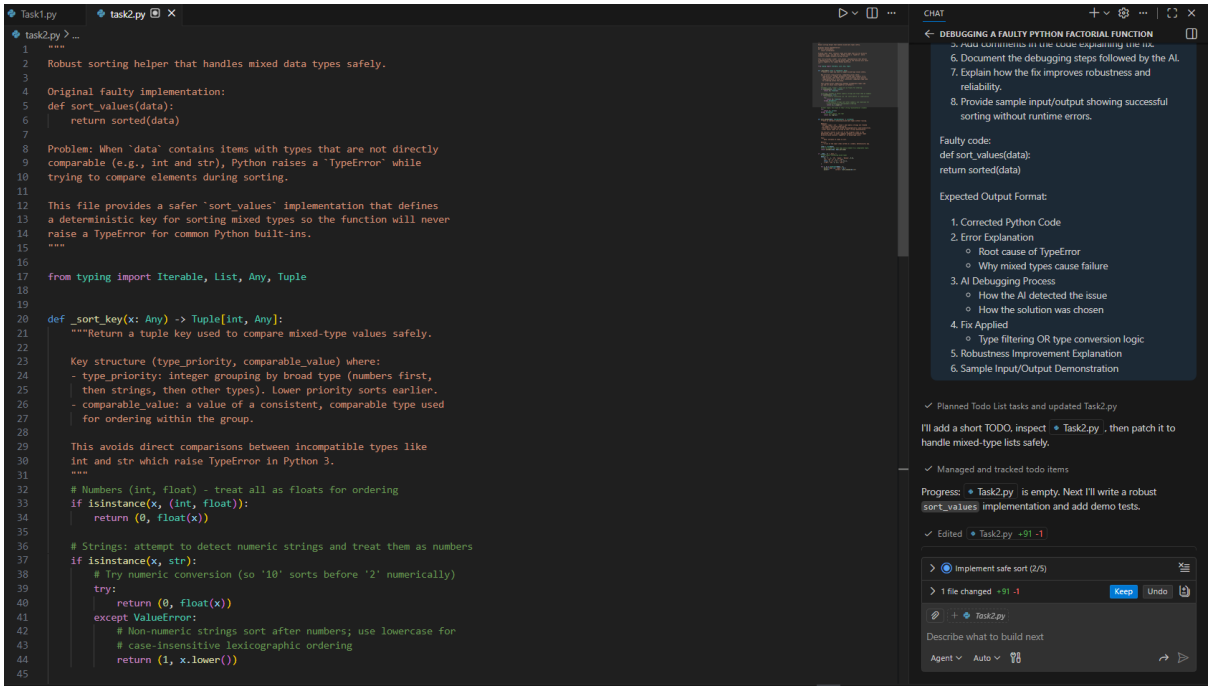
- **Detect the root cause of the runtime error**
- **Modify the code to ensure consistent sorting (by filtering or type conversion)**

- **Prevent the program from crashing**

Explain the debugging steps followed by the AI.

Expected Outcome

- **A corrected sorting function**
- **AI-generated solution handling type inconsistencies**
- **Successful sorting without runtime errors**
- **Explanation of how the fix improves robustness**



OUTPUT:

```
Demo 1: input = [3, '1', 2, '10', 'apple', 'Banana', 0.5]
        sorted = [0.5, '1', 2, 3, '10', 'apple', 'Banana']

Demo 2: input = ['a', 'B', 'c', 2, 1, '10']
        sorted = [1, 2, '10', 'a', 'B', 'c']

Demo 3: input = [None, 5, '3', {'k': 'v'}, (2,)]
        sorted = ['3', 5, (2,), None, {'k': 'v'}]

Demo 4: input = ['100', '20', 3, 4.5, 'abc']
        sorted = [3, 4.5, '20', '100', 'abc']

PS C:\Users\tloke\OneDrive\AI>
```

Task-03: Task 3: Improving File Handling Reliability

Scenario: A backend script reads data from files regularly. Over time, the system shows performance issues due to improper resource management.

Task Description

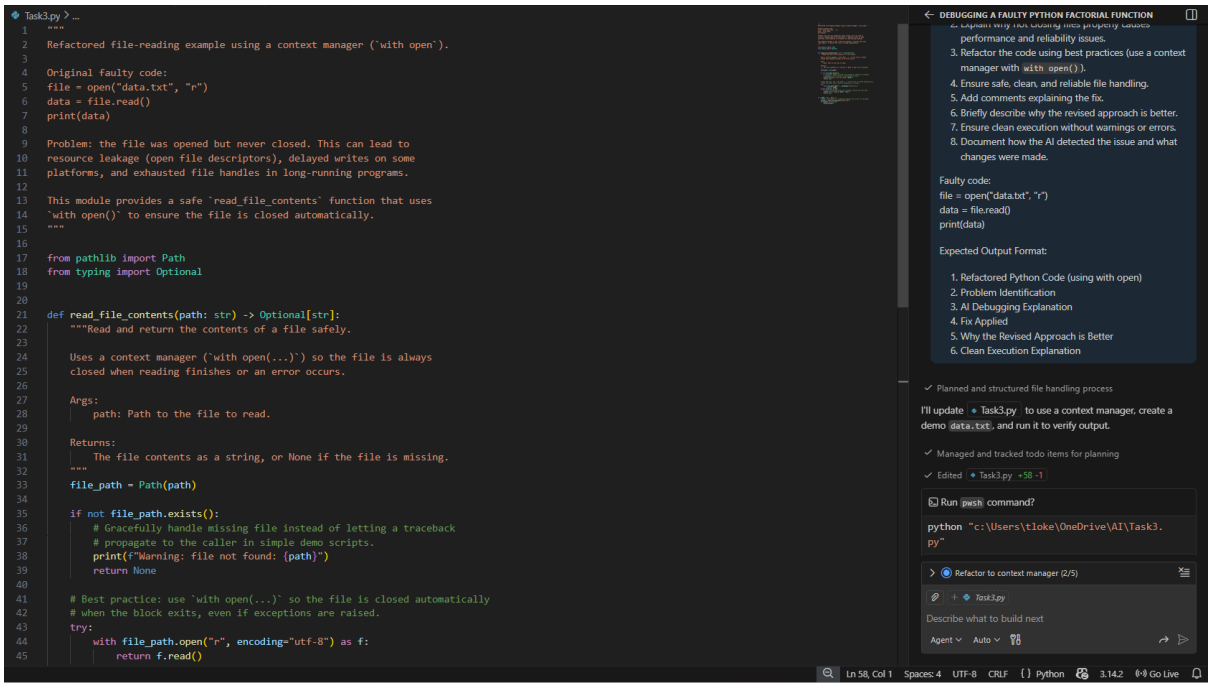
You are given a Python file-handling snippet that opens a file but does not explicitly close it. Use GitHub Copilot or Cursor AI to:

- **Identify the potential problem in the code**
- **Refactor it using best practices (such as a context manager)**
- **Ensure safe and reliable file handling**

Briefly describe why the revised approach is better.

Expected Outcome

- **Refactored code using the with open() statement**
- **AI explanation highlighting prevention of resource leaks**
- **Clean execution without warnings or errors**



data.txt

```
1   Sample data file for Task3 demo.
2   Line 2: Hello, world!
3
```

OUTPUT:

```
PS C:\Users\tloke\OneDrive\AI> & C:/Users/tloke/AppData/Local/Python/pythoncore-3.14-64/python.exe c:/Users/tloke/OneDrive/AI/Task3.py
Sample data file for Task3 demo.
Line 2: Hello, world!
                    Open file in editor (ctrl + click)
PS C:\Users\tloke\OneDrive\AI> & C:/Users/tloke/AppData/Local/Python/pythoncore-3.14-64/python.exe c:/Users/tloke/OneDrive/AI/Task3.py
Sample data file for Task3 demo.
Line 2: Hello, world!
```

Task 4: Handling Runtime Errors Gracefully in Loops

Scenario: You are working on a data analysis script that processes a list of values. Some values cause runtime errors, but the program should continue processing remaining data.

Task Description

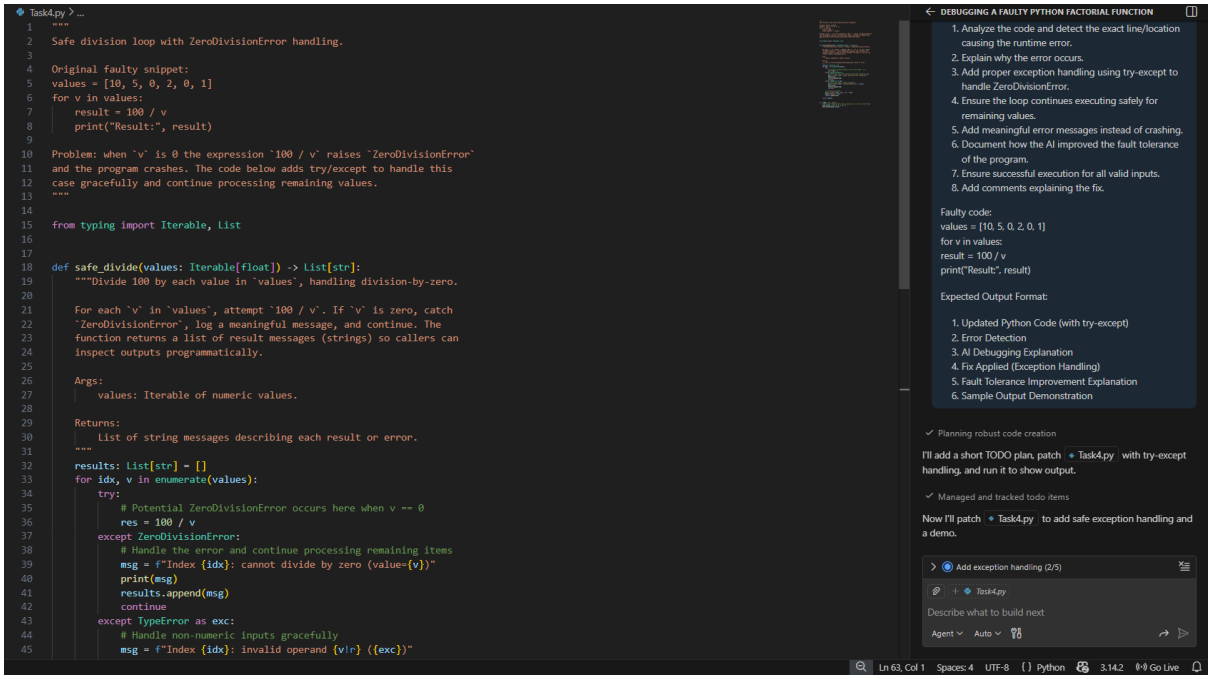
You are provided with a code snippet containing a ZeroDivisionError inside a loop. Use GitHub Copilot or Cursor AI to:

- **Detect the exact location of the error**
- **Add appropriate exception handling using try-except**
- **Ensure the loop continues executing safely**

Document how AI improved the fault tolerance of the program.

Expected Outcome

- **Updated code with proper exception handling**
- **Meaningful error messages instead of program crashes**
- **Successful execution for all valid inputs**



OUTPUT:

```
PS C:\Users\tloke\OneDrive\AI> & C:/Users/tloke/AppData/Local/Python/pythoncore-3.14-64/python.exe c:/Users/tloke/OneDrive/AI/Task4.py
Result: 10.0
Result: 20.0
Index 2: cannot divide by zero (value=0)
Result: 50.0
Index 4: cannot divide by zero (value=0)
Result: 100.0
PS C:\Users\tloke\OneDrive\AI>
```

Task 5: Debugging Class Initialization Errors

Scenario: A class written by a junior developer is throwing unexpected errors when objects are created or attributes are accessed.

Task Description

You are given a Python class with:

- **Incorrect __init__ parameters**
- **Missing or incorrect attribute references (e.g., missing self)**

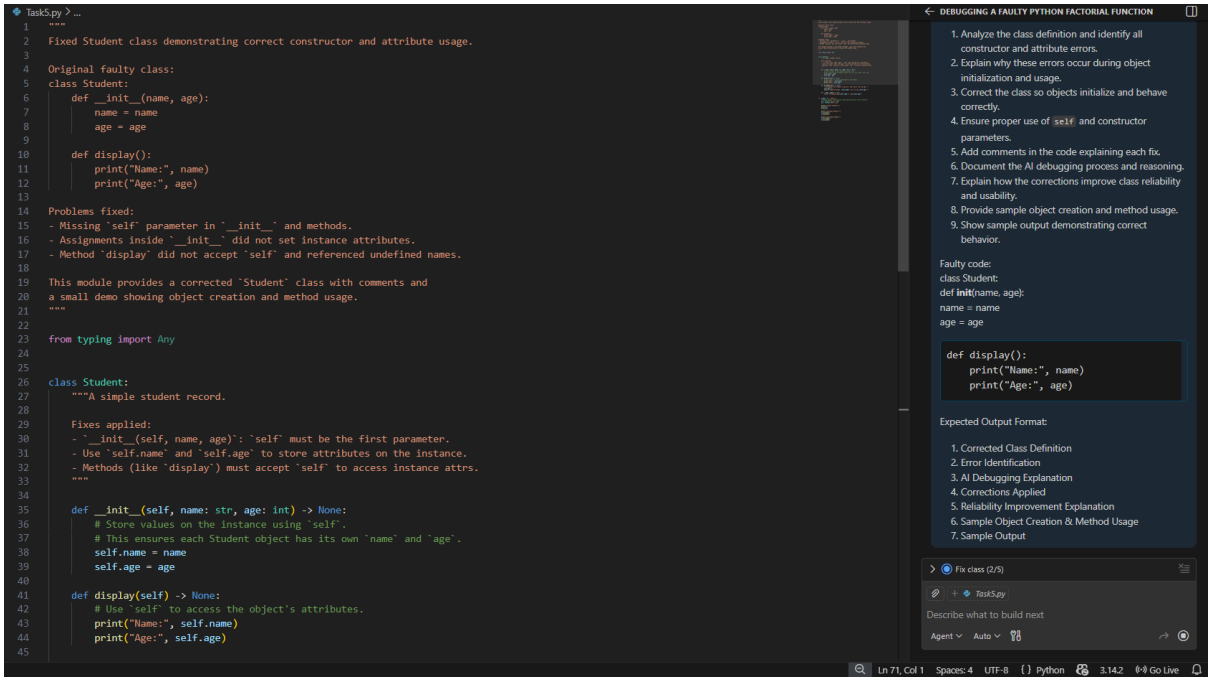
Use GitHub Copilot or Cursor AI to:

- **Analyze the class definition**
- **Identify constructor and attribute issues**
- **Correct the class so objects initialize and behave correctly**

Explain the corrections suggested by the AI.

Expected Outcome

- **A corrected class definition**
- **Proper use of self and constructor parameters**
- **AI-assisted explanation of the original errors and fixes**
- **Sample object creation and method usage**



OUTPUT:

```
PS C:\Users\tloke\OneDrive\AI> & C:/Users/tloke/AppData/Local/Python/pythoncore-3.14-64/python.exe c:/Users/tloke/OneDrive/AI/Task5.py
Created students:
Student(name='Alice', age=20)
Student(name='Bob', age=22)

Display details:
Name: Alice
Age: 20
Name: Bob
Age: 22

Birthday update:
Happy birthday, Alice! You are now 21.
Name: Alice
Age: 21
PS C:\Users\tloke\OneDrive\AI>
```