AI Assistant Coding

Name: CH.VenuGopal

HTNO: 2303A51844

Batch-13

Date: 05-02-2026

Assignment - 7.4

Problem statements with solution:

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

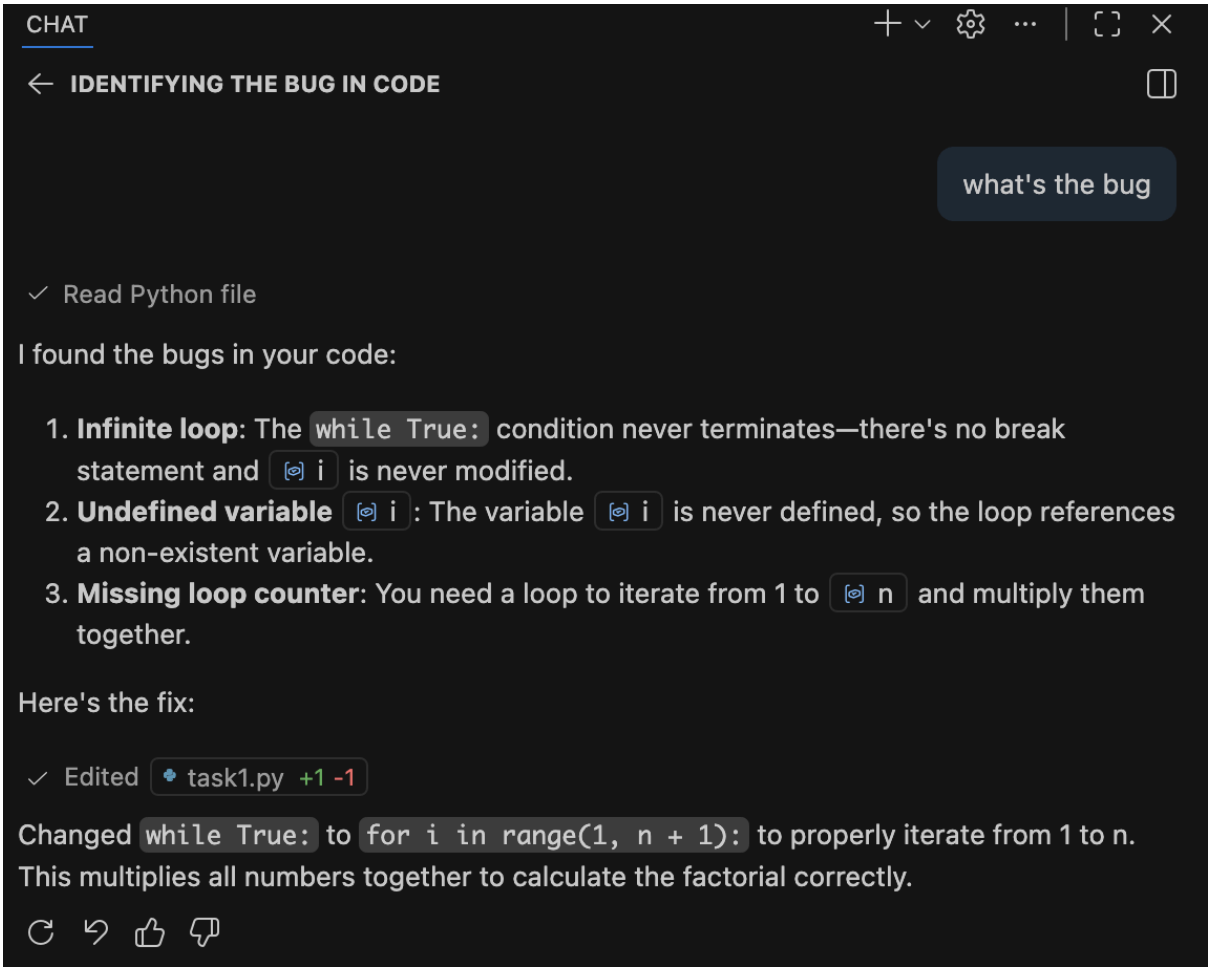
• Identify the exact cause of the error

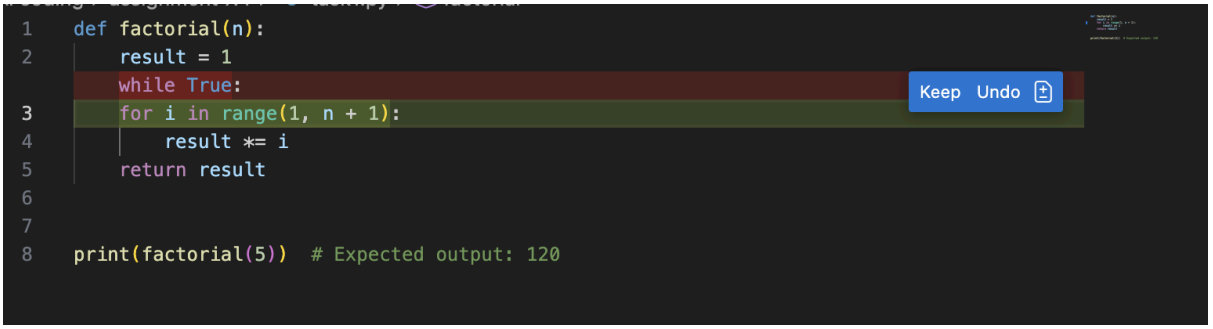
• Suggest and apply corrections to make the function work correctly

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

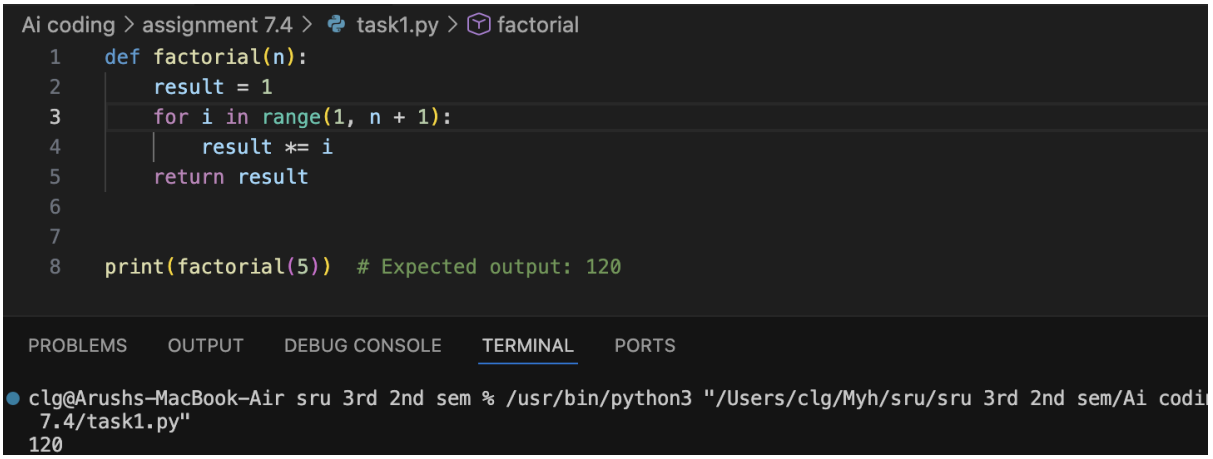
```python
def factorial(n):
    result = 1
    while True:
        result *= i
    return result


print(factorial(5))  # Expected output: 120
```

```
clg@Arushs-MacBook-Air sru 3rd 2nd sem % /usr/bin/python3 "/Users/clg/Myh/sru/sru 3rd 2nd sem/A
i coding/assignment 7.4/task1.py"
Traceback (most recent call last):
  File "/Users/clg/Myh/sru/sru 3rd 2nd sem/Ai coding/assignment 7.4/task1.py", line 8, in <module>
    print(factorial(5))  # Expected output: 120
  File "/Users/clg/Myh/sru/sru 3rd 2nd sem/Ai coding/assignment 7.4/task1.py", line 4, in factorial
    result *= i
NameError: name 'i' is not defined
```





correct execution:



Task 2: 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

TypeError 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.

```python
def sort_user_values(values):
    return sorted(values)


user_input = [10, "20", 5, "3"]

sorted_values = sort_user_values(user_input)
print(sorted_values)
```

```
clg@Arushs-MacBook-Air sru 3rd 2nd sem % /usr/bin/python3 "/Users/clg/Myh/sru/sru 3rd 2nd sem/Ai coding/assignment 7.4/task2.py"
Traceback (most recent call last):
  File "/Users/clg/Myh/sru/sru 3rd 2nd sem/Ai coding/assignment 7.4/task2.py", line 7, in <module>
    sorted_values = sort_user_values(user_input)
  File "/Users/clg/Myh/sru/sru 3rd 2nd sem/Ai coding/assignment 7.4/task2.py", line 2, in sort_user_values
    return sorted(values)
TypeError: '<' not supported between instances of 'str' and 'int'
```

```python
def sort_user_values(values):
    return sorted(values)


user_input = [10, "20", 5, "3"]
user_input = [10, 20, 5, 3]

sorted_values = sort_user_values(user_input)
print(sorted_values)
```

Keep Undo 1 of 1

correct execution:

Ai coding > assignment 7.4 > task2.py > sorted_values

```python
def sort_user_values(values):
    return sorted(values)


user_input = [10, 20, 5, 3]

sorted_values = sort_user_values(user_input)
print(sorted_values)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
clg@Arushs-MacBook-Air sru 3rd 2nd sem % /usr/bin/python3 "/Users/clg/Myh/sru/sru 3rd 2nd sem/Ai coding/assignment 7.4/task2.py"
[3, 5, 10, 20]
clg@Arushs-MacBook-Air sru 3rd 2nd sem %
```

The fix improves robustness in several ways:

1. Eliminates type errors: Mixed types cause Python to crash when trying to compare incompatible objects. By ensuring all values are integers, the sorted() function works reliably.
2. Type consistency: The code now handles data predictably. Every element can be compared with every other element using the same logic.

3. Deterministic behavior: With consistent types, sorting produces expected results every time. The output is deterministic rather than error-prone.
4. Better maintainability: Code dealing with homogeneous data types is easier to understand and debug.

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.

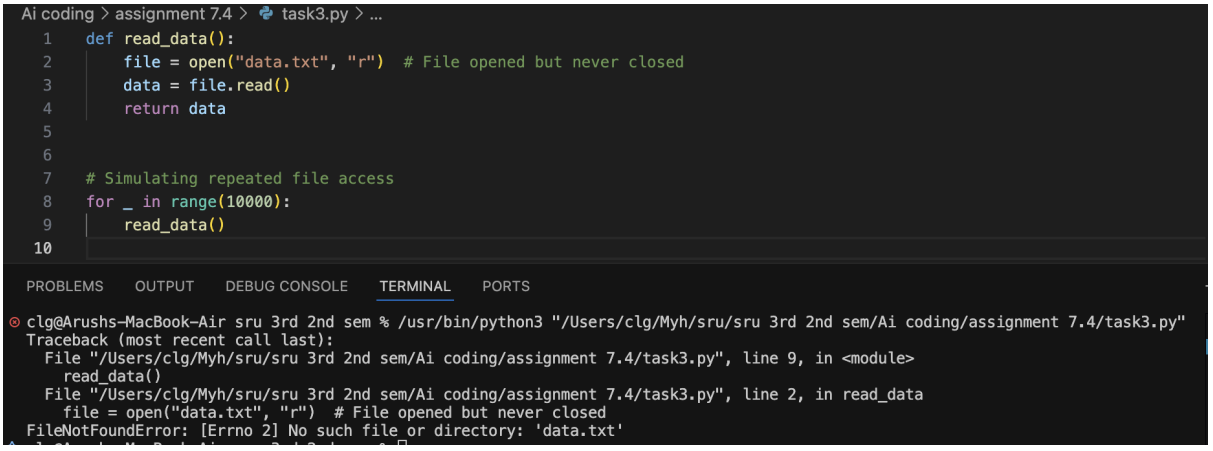
Use GitHub Copilot or Cursor AI to:

• Identify the potential problem in the code

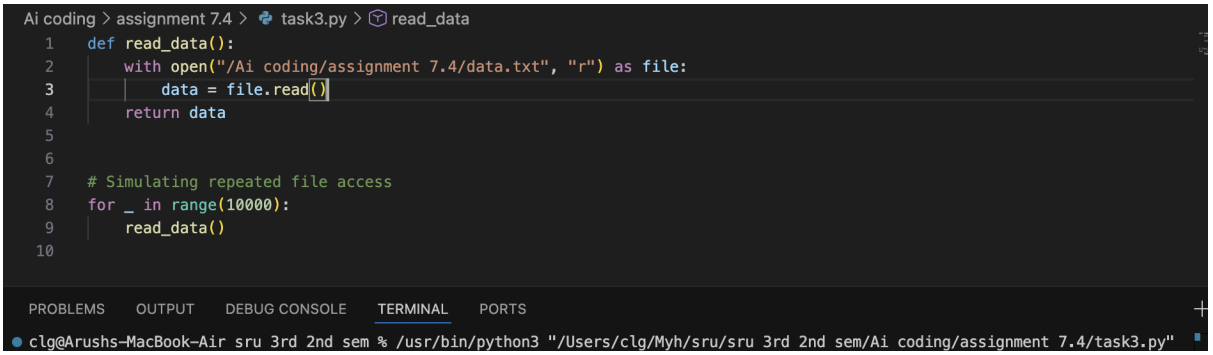
• Refactor it using best practices (such as a context manager)

• Ensure safe and reliable file handling

Briefly describe why the revised approach is better.



correct execution:



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.

```python
def process_values(values):
    results = []
    for v in values:
        result = 100 / v   # 💥 crashes when v == 0
        results.append(result)
    return results


data = [10, 5, 0, 20, 4]

# Program stops at v == 0
output = process_values(data)
print(output)

```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
clg@Arushs-MacBook-Air sru 3rd 2nd sem % /usr/bin/python3 "/Users/clg/Myh/sru/sru 3rd 2
nd sem/Ai coding/assignment 7.4/task3.py"
clg@Arushs-MacBook-Air sru 3rd 2nd sem % clear
clg@Arushs-MacBook-Air sru 3rd 2nd sem % /usr/bin/python3 "/Users/clg/Myh/sru/sru 3rd 2nd sem/Ai coding/assignment 7.4/task3.py"
clg@Arushs-MacBook-Air sru 3rd 2nd sem % /usr/bin/python3 "/Users/clg/Myh/sru/sru 3rd 2
nd sem/Ai coding/assignment 7.4/task4.py"
Traceback (most recent call last):
  File "/Users/clg/Myh/sru/sru 3rd 2nd sem/Ai coding/assignment 7.4/task4.py", line 12, in <module>
    output = process_values(data)
  File "/Users/clg/Myh/sru/sru 3rd 2nd sem/Ai coding/assignment 7.4/task4.py", line 4, in process_values
    result = 100 / v   # 💥 crashes when v == 0
ZeroDivisionError: division by zero
```

Ai coding > assignment 7.4 > task4.py > ...

```python
def process_values(values):
    results = []
    for v in values:
        result = 100 / v   # 💥 crashes when v == 0
        results.append(result)
        try:
            result = 100 / v
            results.append(result)
        except ZeroDivisionError:
            results.append(None)  # or 0, or skip it
        except (TypeError, ValueError):
            results.append(None)  # Handle invalid types
    return results


data = [10, 5, 0, 20, 4]

# Program stops at v == 0
output = process_values(data)
print(output)

```

Keep Undo | 1 of 1

correct execution:

Ai coding > assignment 7.4 > task4.py > ...

```python
def process_values(values):
    results = []
    for v in values:
        try:
            result = 100 / v
            results.append(result)
        except ZeroDivisionError:
            results.append(None)  # or 0, or skip it
        except (TypeError, ValueError):
            results.append(None)  # Handle invalid types
    return results


data = [10, 5, 0, 20, 4]

# Program stops at v == 0
output = process_values(data)
print(output)

```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
clg@Arushs-MacBook-Air sru 3rd 2nd sem % /usr/bin/python3 "/Users/clg/Myh/sru/sru 3rd 2nd sem/Ai coding/assignment 7.4/task4.py"
[10.0, 20.0, None, 5.0, 25.0]
```

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)

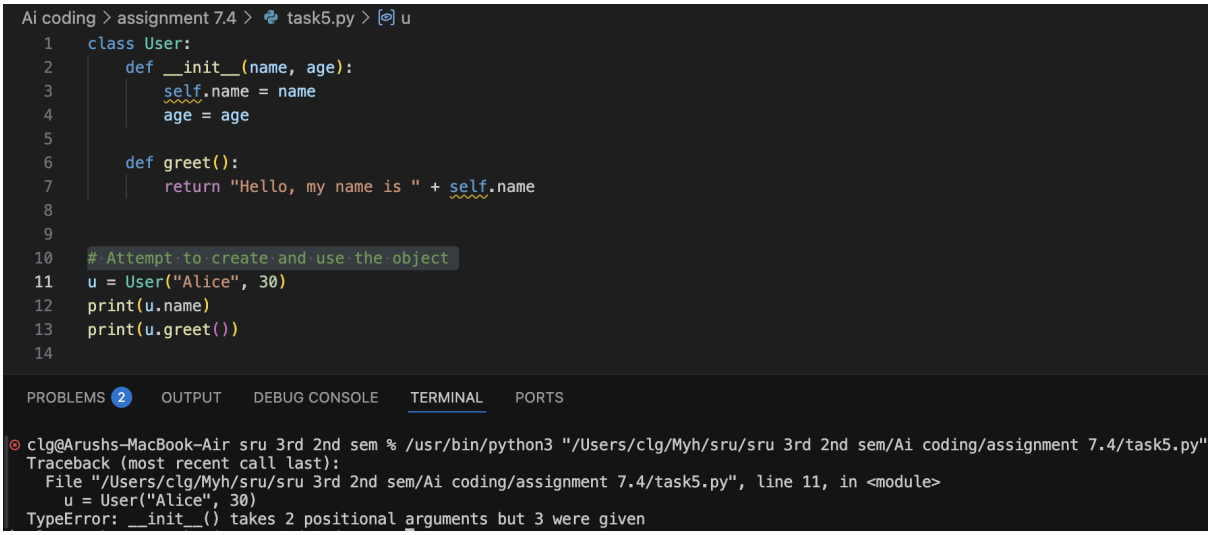
Use GitHub Copilot or Cursor AI to:

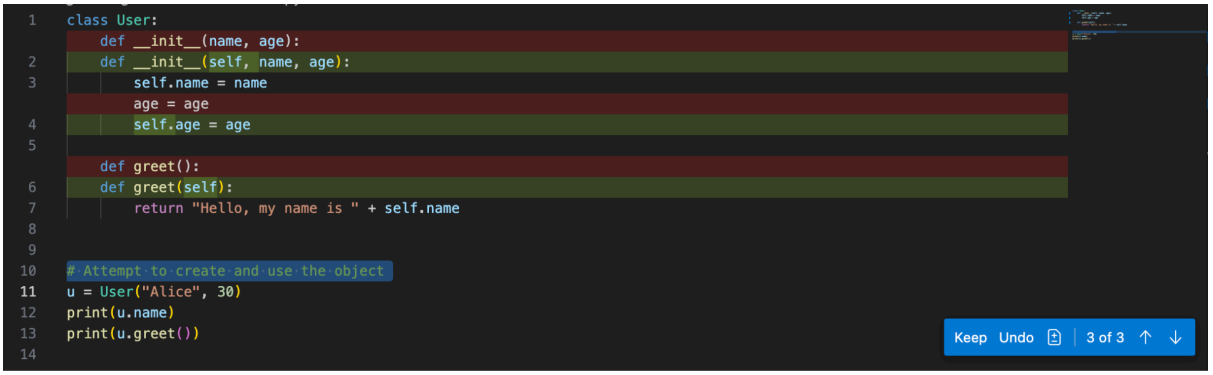
• Analyze the class definition

• Identify constructor and attribute issues

• Correct the class so objects initialize and behave correctly

Explain the corrections suggested by the AI.





Problem:

1. First parameter must be self to reference the instance
2. Creates a local variable, doesn't store it on the object
3. Method has no self parameter, can't access instance variables

Fix:

1. def __init__(self, name, age):
2. self.age = age to store on instance
3. def greet(self): to enable self.name access

self represents the current object instance. All instance methods must have self as the first parameter.

self.name and self.age store data on the object, while local variables (age = age) are discarded after the function ends

correct execution: