

LAB NAME : AI ASSISTED CODING
ROLL NO :2503A51L16
BRANCH : CSE
NAME : K.JASHUVA

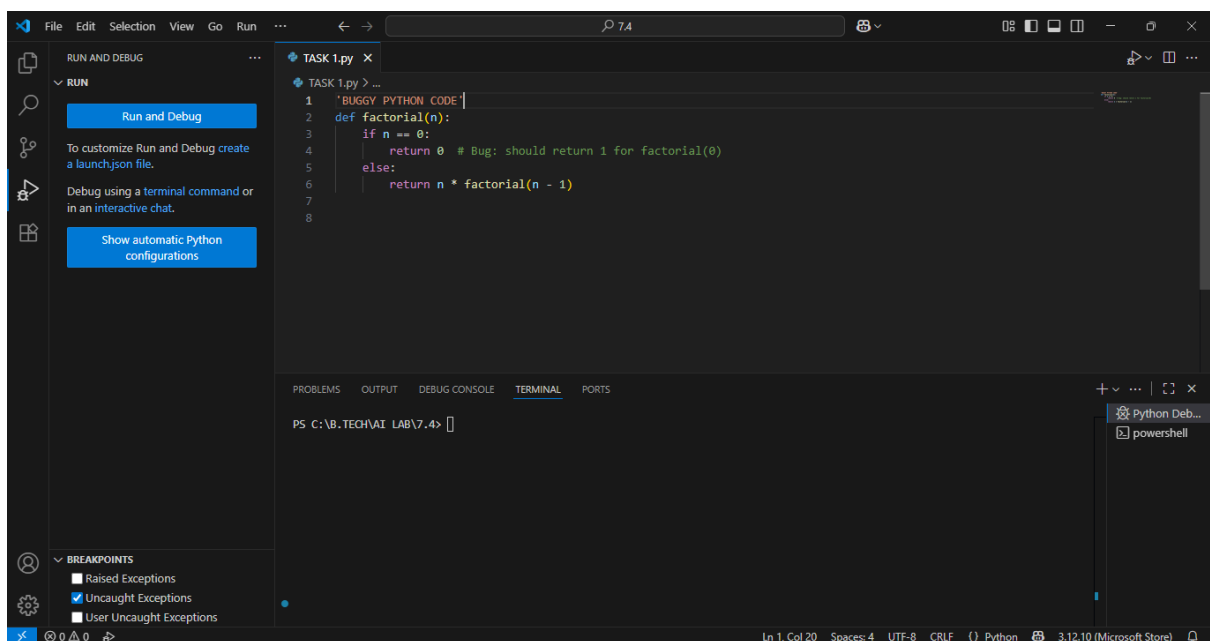
TASK 1

Task Description: Introduce a buggy Python function that calculates the factorial of a number using recursion.

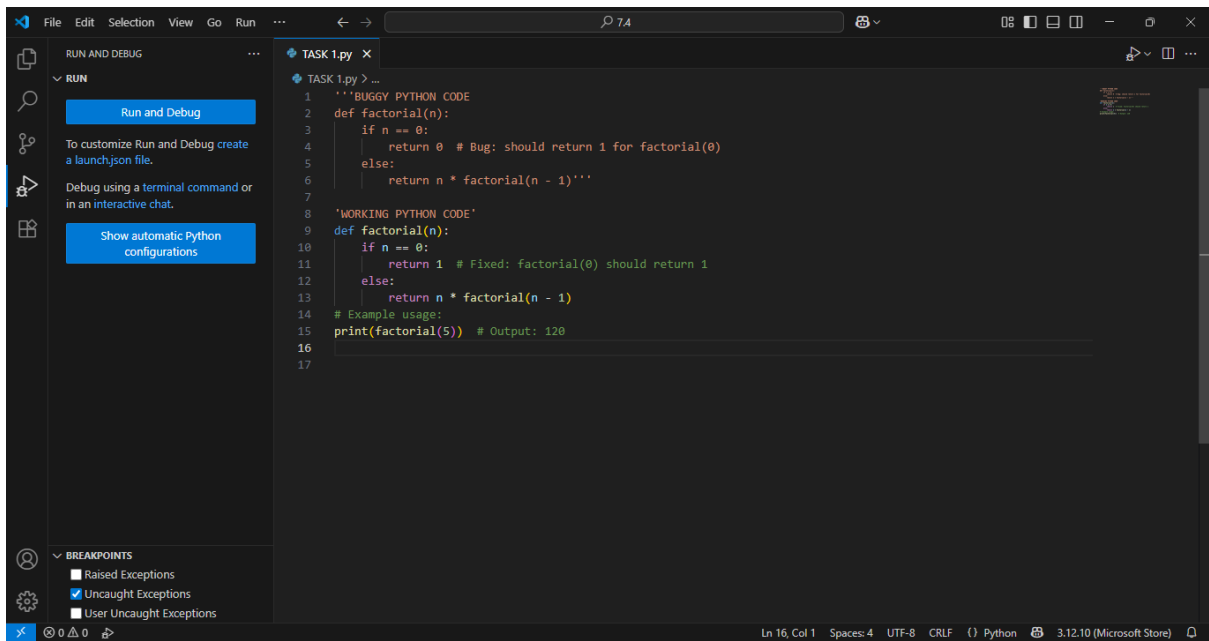
Use Copilot or Cursor AI to detect and fix the logical or syntax errors.

PROMPT: In Above Code there is a bug can you identify and fix it, along with explanation.

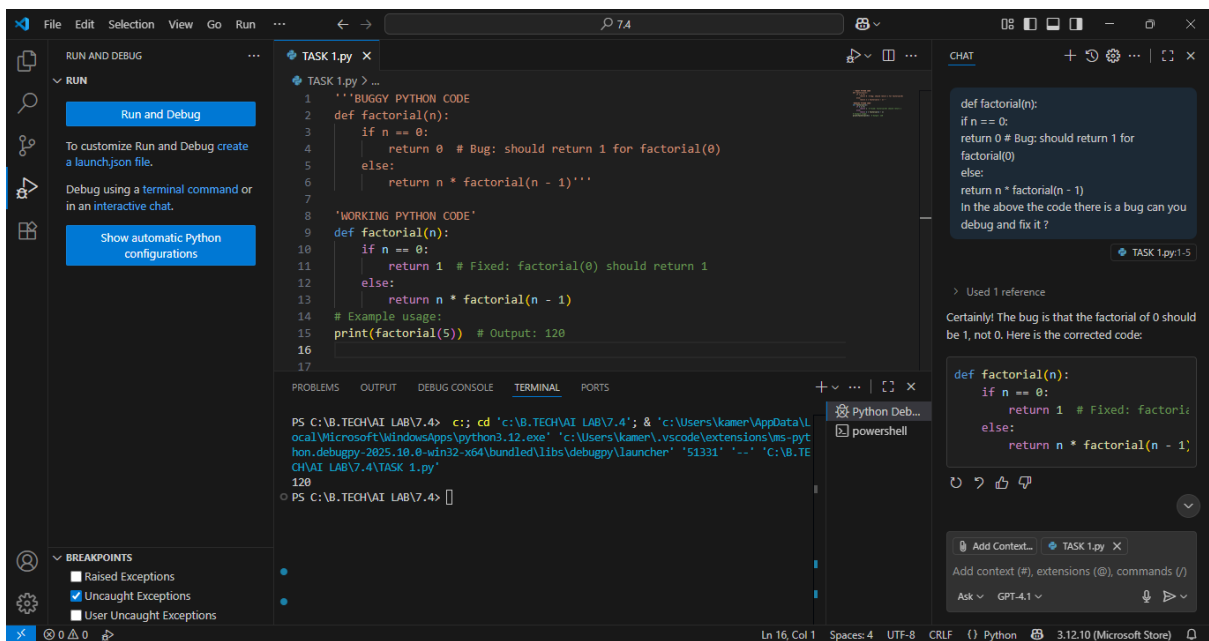
BUGGY CODE:



WORKING CODE:



OUTPUT:



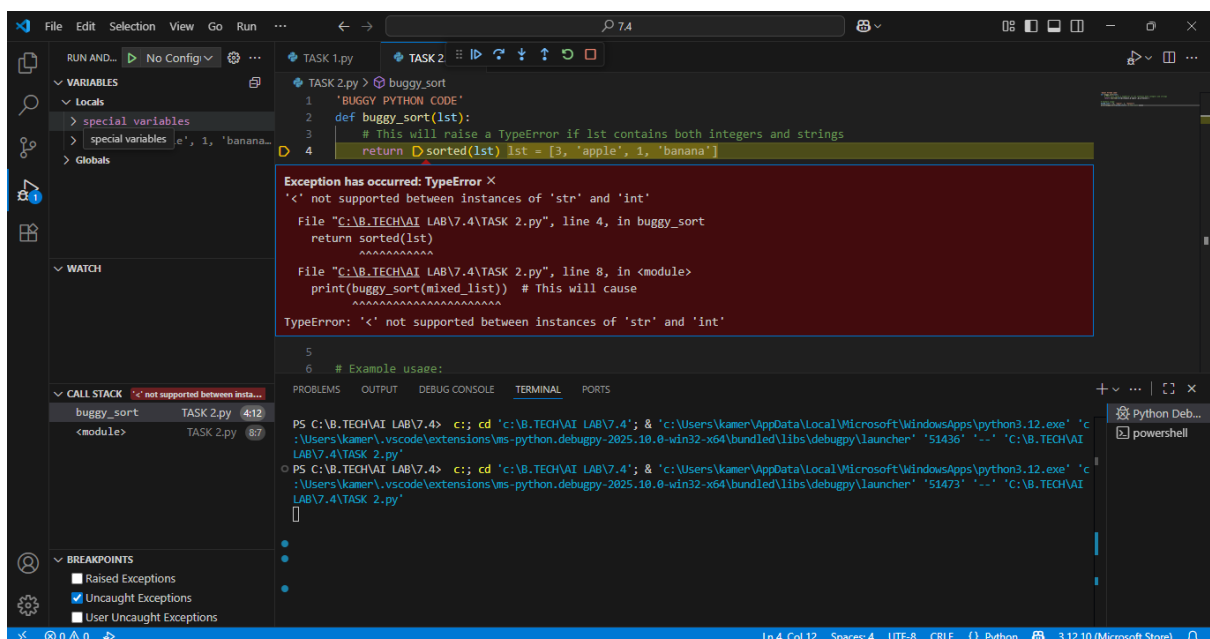
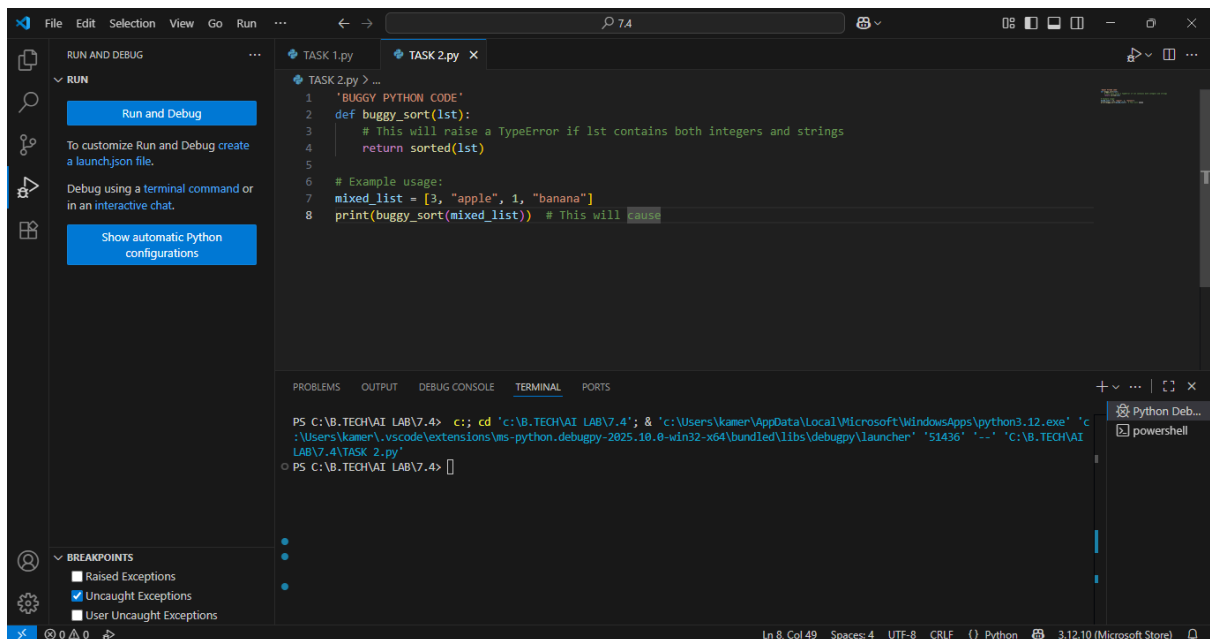
TASK 2

Task Description #2:

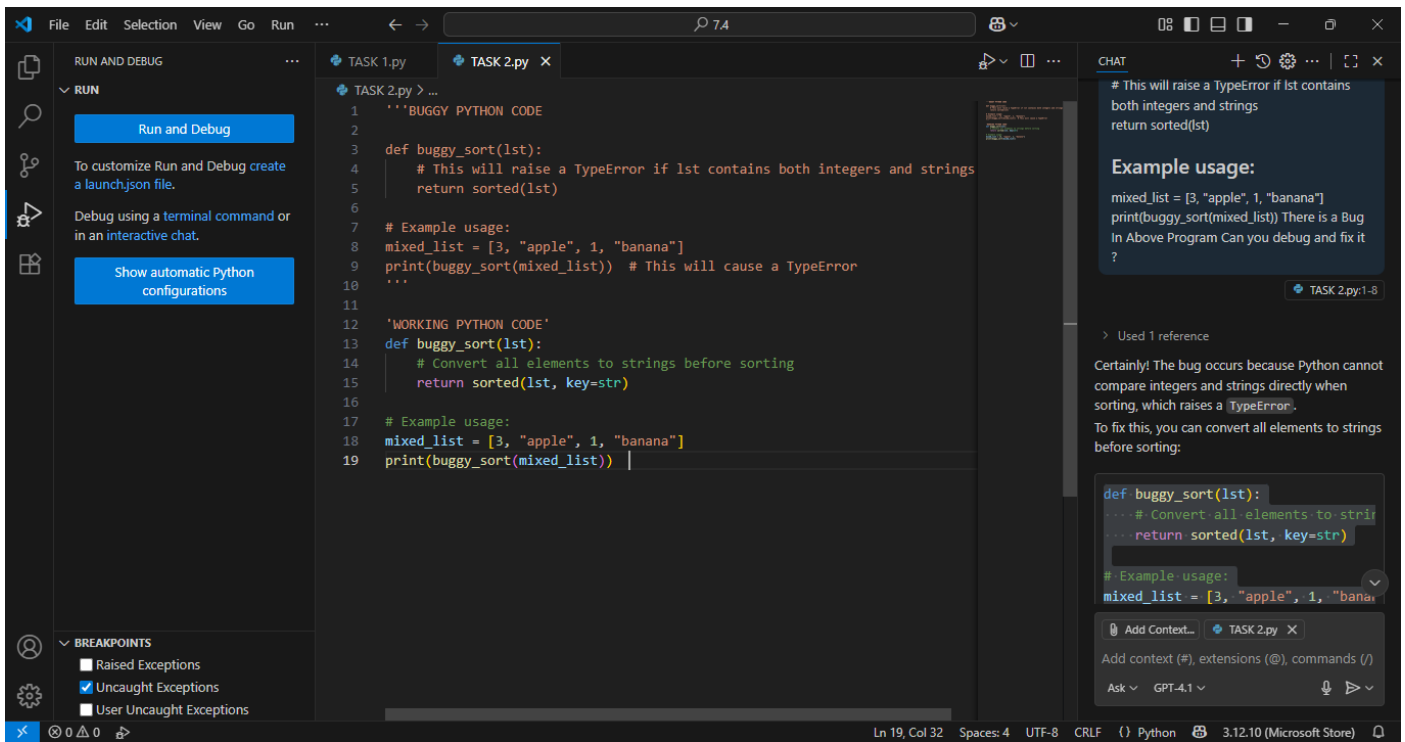
- Provide a list sorting function that fails due to a type error (e.g., sorting list with mixed integers and strings). Prompt AI to detect the issue and fix the code for consistent sorting.

PROMPT: : In Above Code there is a bug can you identify and fix it, along with explanation.

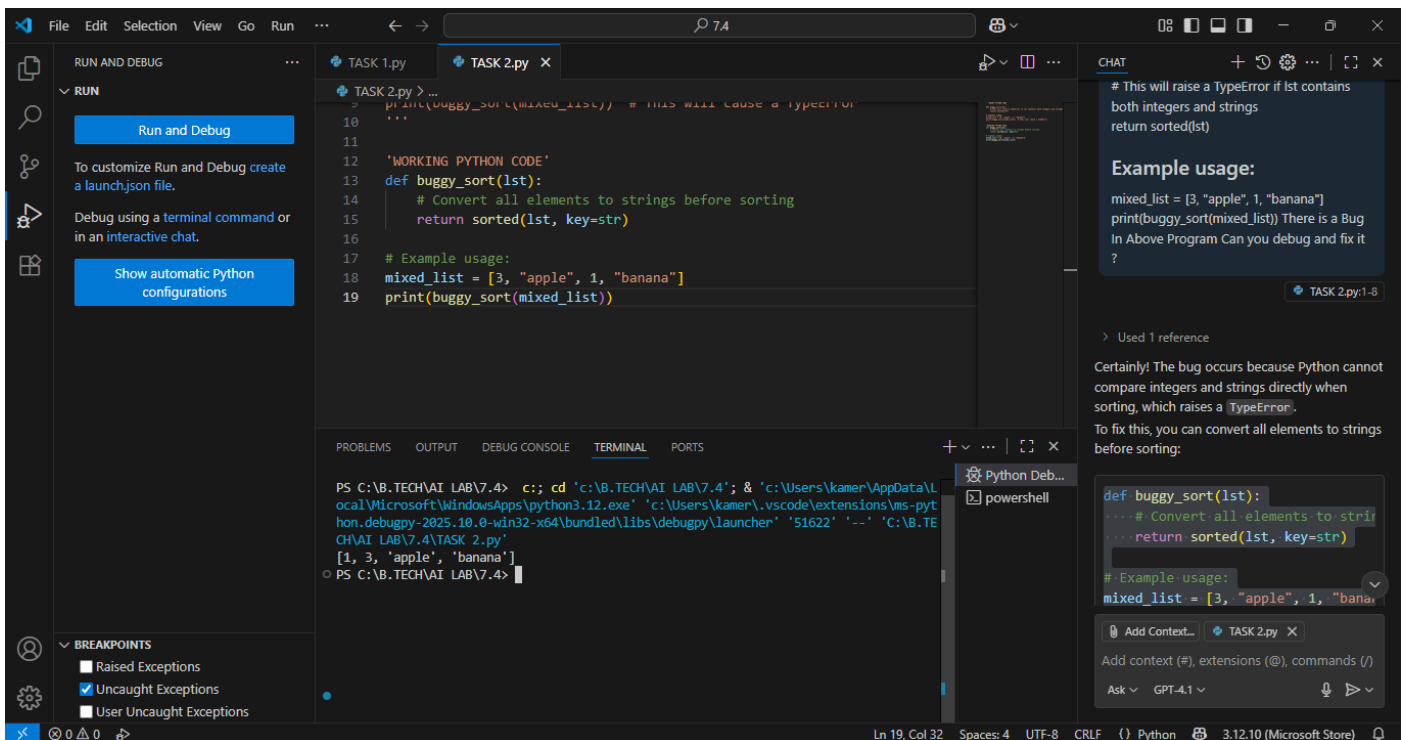
BUGGY CODE:



WORKING CODE:



OUTPUT:



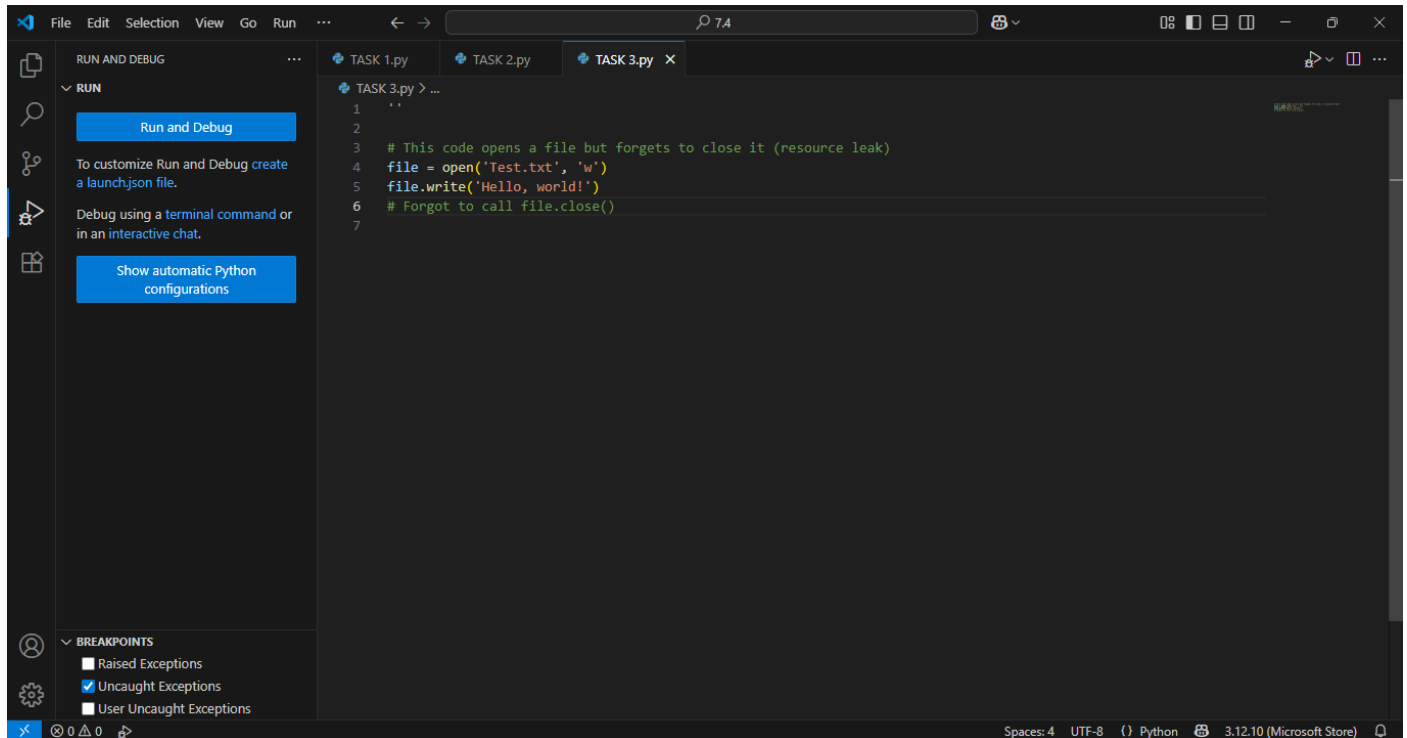
TASK 3

Task Description #3:

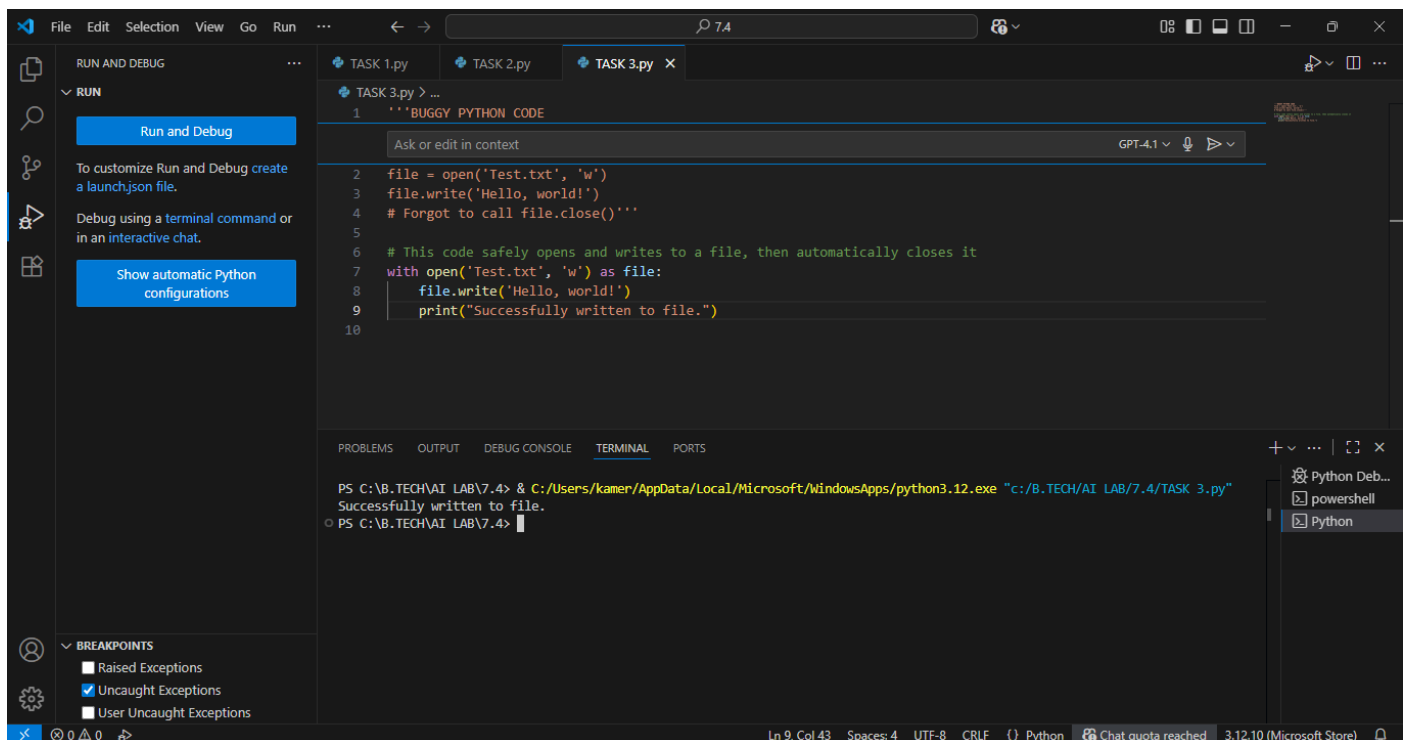
- Write a Python snippet for file handling that opens a file but forgets to close it.Ask Copilot or Cursor AI to improve it using the best practice (e.g., with open() block).

PROMPT: : In Above Code there is a bug can you identify and fix it, along with explanation.

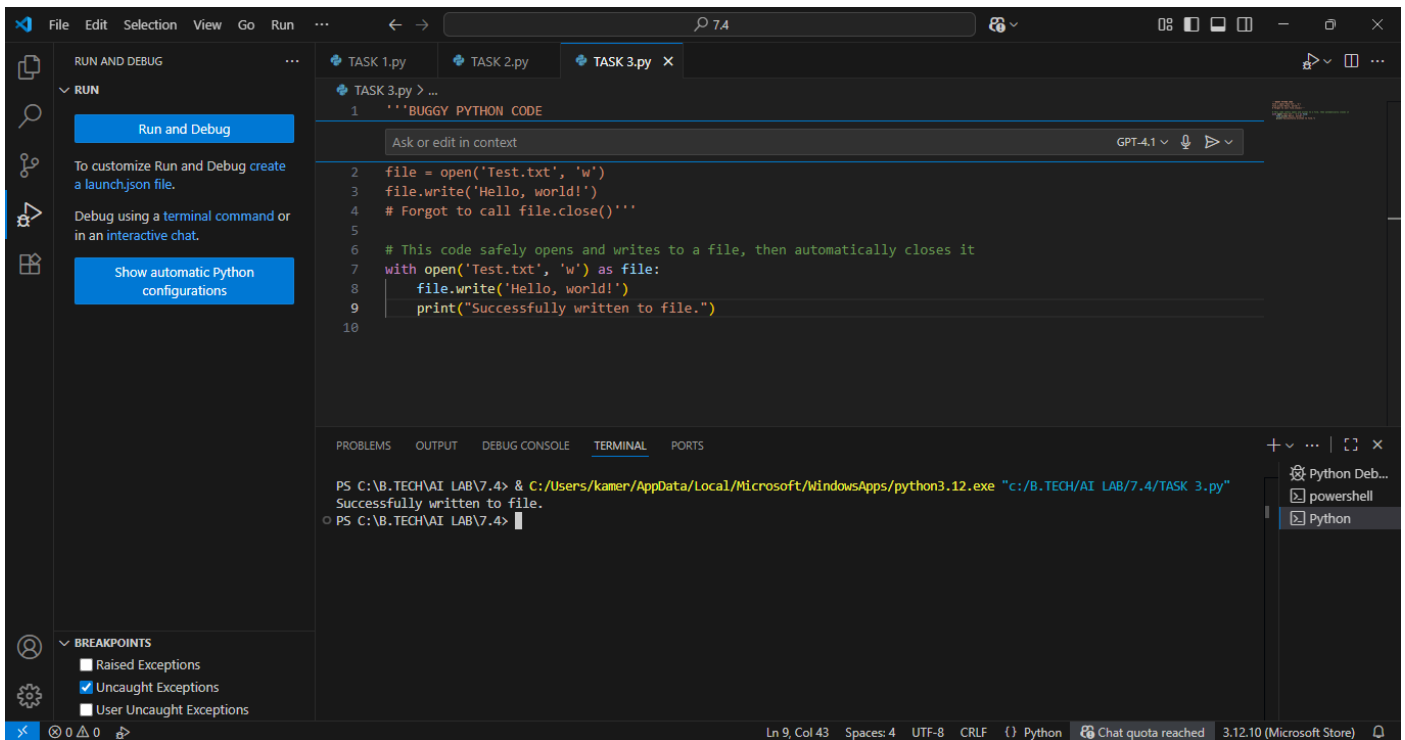
BUGGY CODE:



WORKING CODE:



OUTPUT:



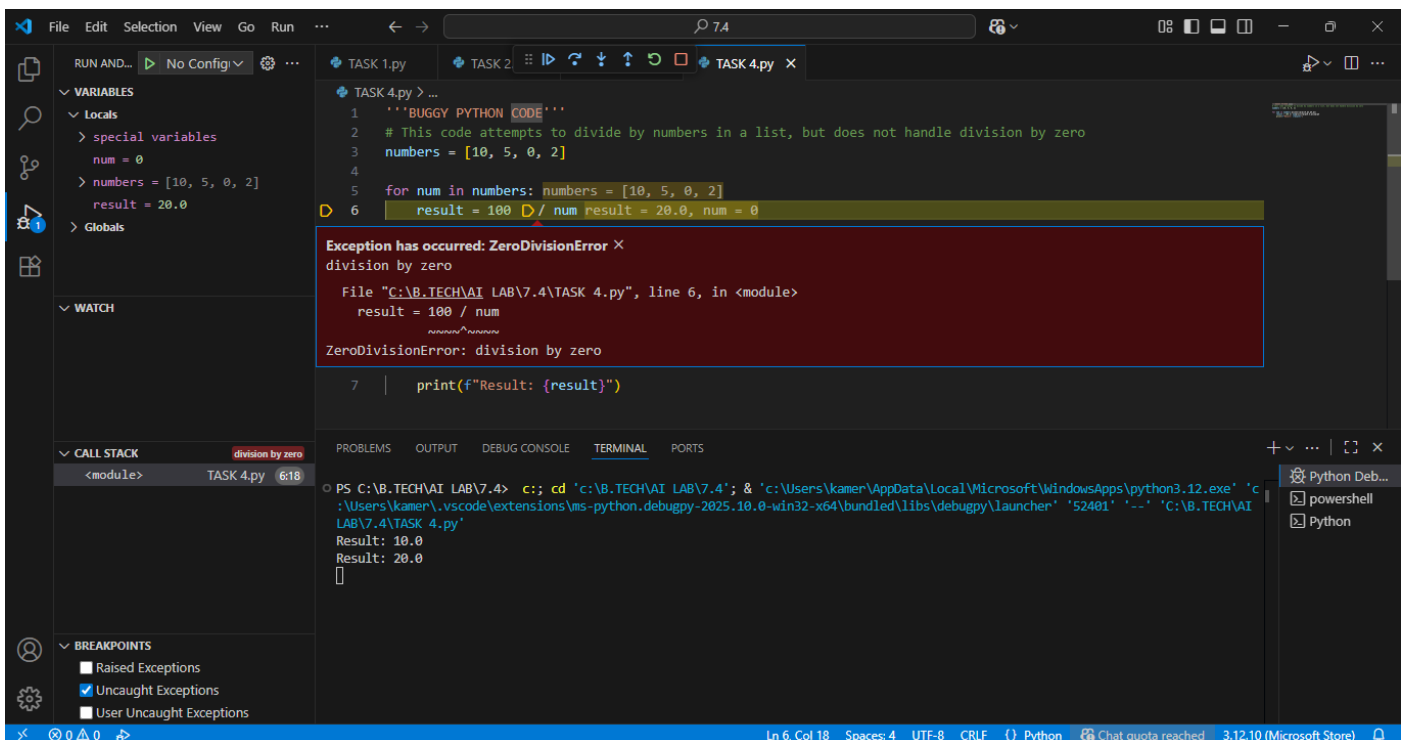
TASK 4

Task Description #4:

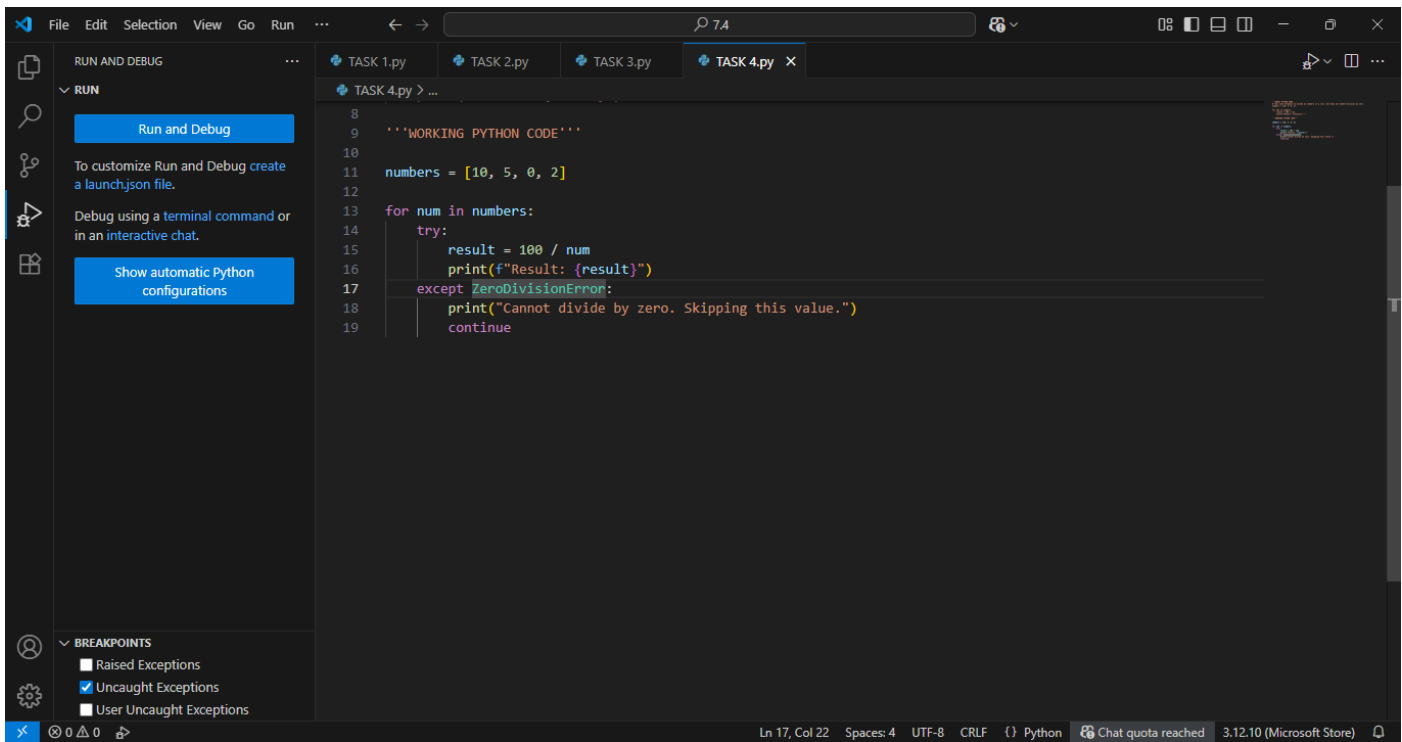
- Provide a piece of code with a ZeroDivisionError inside a loop. Ask AI to add error handling using try-except and continue execution safely.

PROMPT: : In Above Code there is a bug can you identify and fix it, along with explanation.

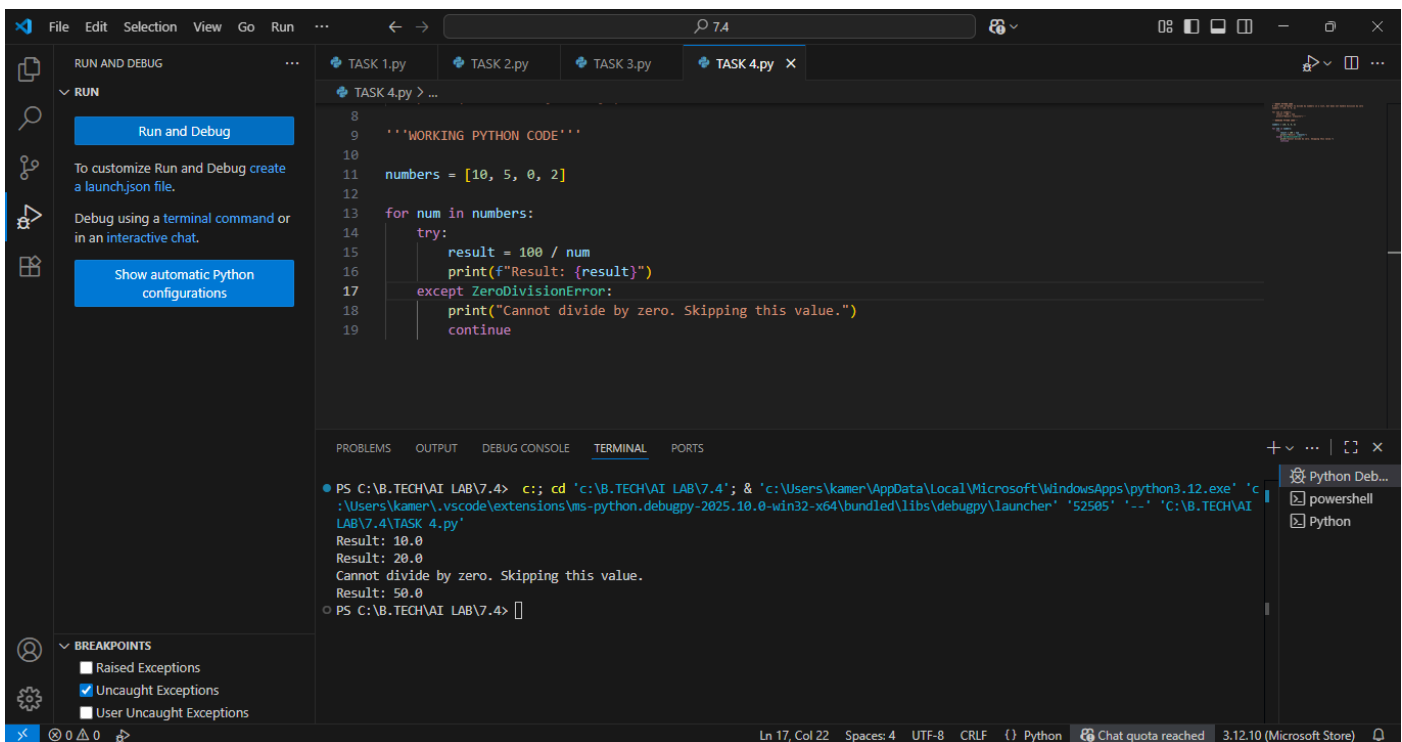
BUGGY CODE:



WORKING CODE:



OUTPUT:



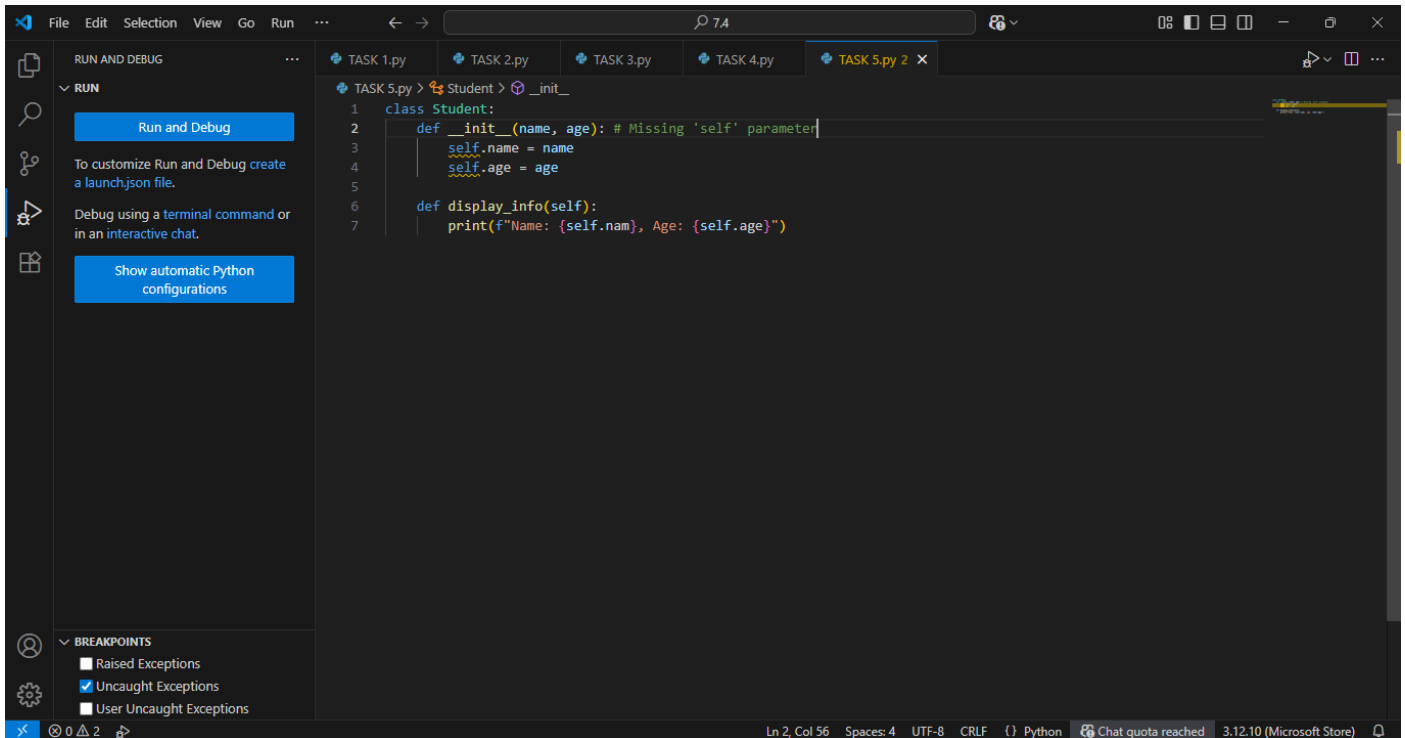
TASK 5

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

PROMPT: : In Above Code there is a bug can you identify and fix it, along with explanation.

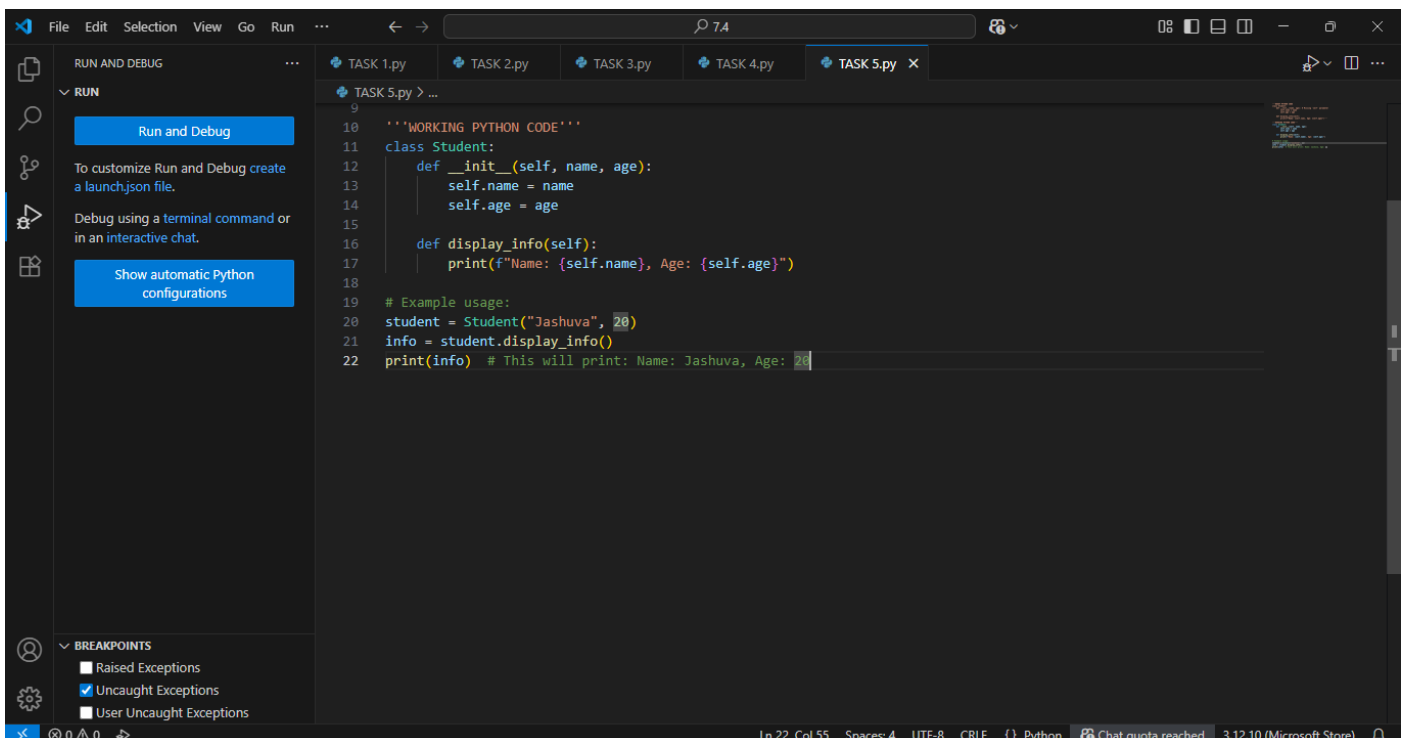
BUGGY CODE:



The screenshot shows the VS Code editor with a file named `TASK 5.py`. The code defines a `Student` class. The `__init__` method has parameters `(name, age)` but is missing the `self` parameter. The `display_info` method uses `self.name` and `self.age` attributes. The status bar at the bottom indicates the cursor is at line 2, column 56.

```
1 class Student:
2     def __init__(name, age): # Missing 'self' parameter
3         self.name = name
4         self.age = age
5
6     def display_info(self):
7         print(f"Name: {self.name}, Age: {self.age}")
```

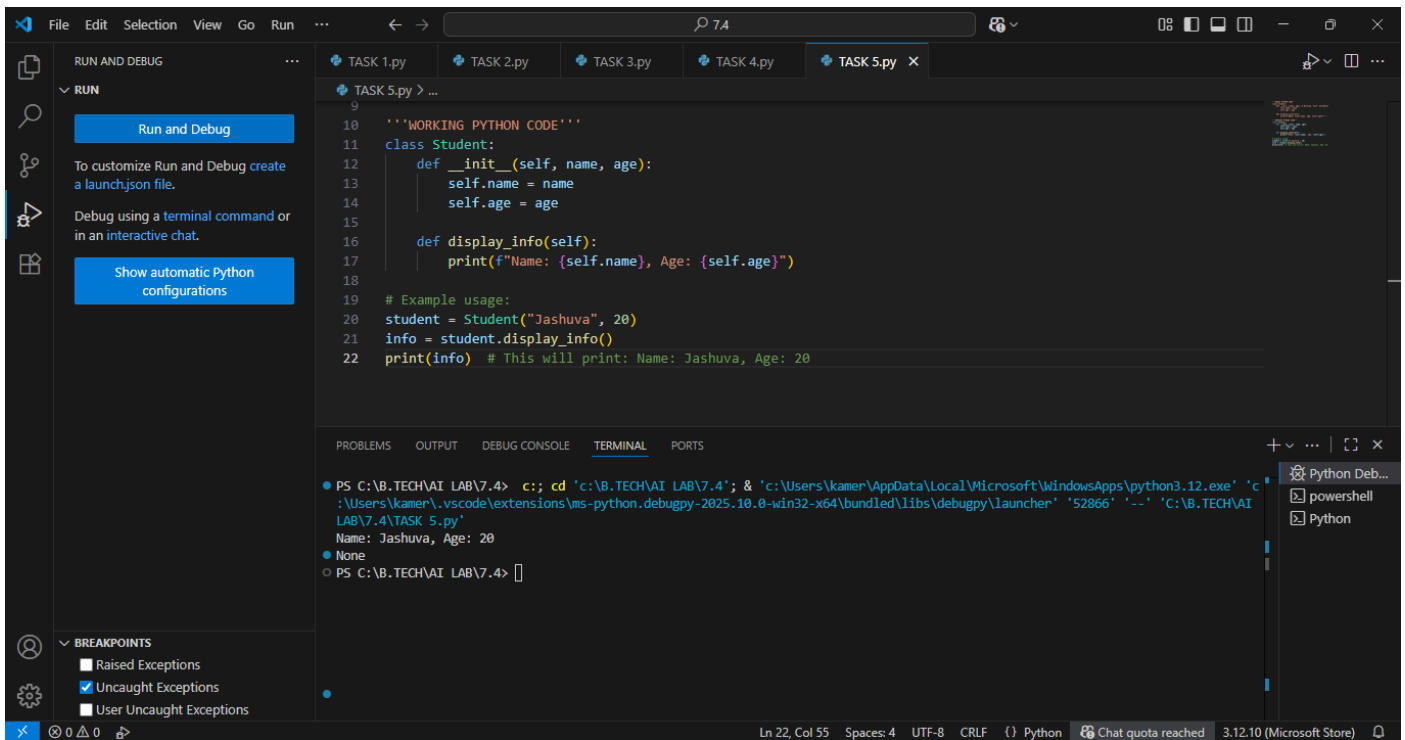
WORKING CODE:



The screenshot shows the VS Code editor with the same file `TASK 5.py`. The code is now corrected. The `__init__` method includes `self` as the first parameter. Below the class definition, there is an example usage showing the creation of a `Student` object and the call to `display_info`. The status bar at the bottom indicates the cursor is at line 22, column 55.

```
9 '''WORKING PYTHON CODE'''
10
11 class Student:
12     def __init__(self, name, age):
13         self.name = name
14         self.age = age
15
16     def display_info(self):
17         print(f"Name: {self.name}, Age: {self.age}")
18
19 # Example usage:
20 student = Student("Jashuva", 20)
21 info = student.display_info()
22 print(info) # This will print: Name: Jashuva, Age: 20
```

OUTPUT:



OBSERVATION:

I Observed that GitHub AI Copilot can easily generate the correct programs and identifying errors or Bugs in given program and it can easily debugging along with explanations. GitHub Copilot is a fascinating tool to observe—especially in how it transforms the developer experience.