**Exercise 5: Feature Enhancement with Limited Context**

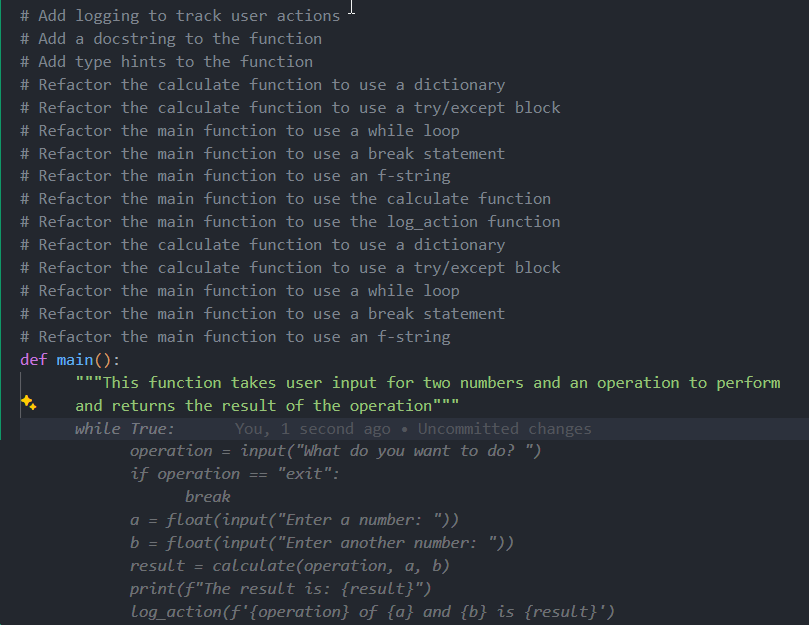
### **Objective:** Enhance your Python project by adding a logging feature to track user actions, even when the code has minimal context or documentation.

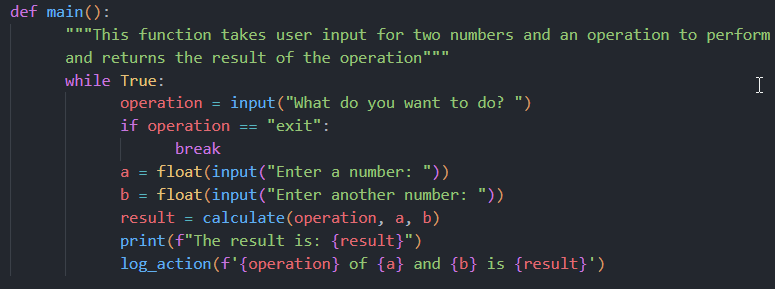
## **Step-by-Step Instructions:**

### **Review the Existing Code**

* 1. Open main.py in VS Code.
  2. Look for a function or section that lacks documentation.
  3. Add a comment like:

| # What does this function do? Let's add logging. |
| --- |





### **Define the New Feature**

* 1. Decide that you want to add a logging function which writes a message to a file named activity.log.

### **Write Tests for the New Feature**

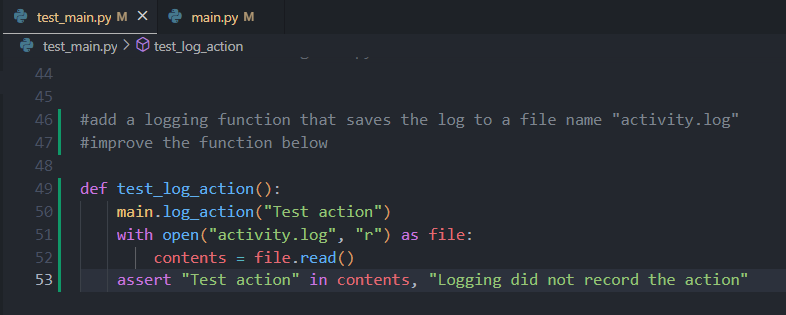
* 1. Open test\_main.py.
  2. Insert a comment:

| # Test for logging feature |
| --- |

* 1. Write a simple test:

| def test\_log\_action():  log\_action("Test action")  with open("activity.log", "r") as file:  contents = file.read()  assert "Test action" in contents, "Logging did not record the action" |
| --- |

* 1. Allow Copilot to suggest improvements if available.



* 1. Save the test file.

### **Implement the Logging Feature**

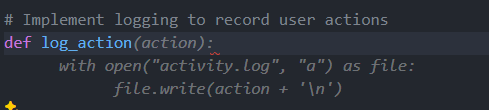
* 1. Return to main.py.
  2. Add a comment:

| # Implement logging to record user actions |
| --- |

* 1. Use Copilot to generate a function. For example:

| def log\_action(action):  with open("activity.log", "a") as file:  file.write(f"{action}\n") |
| --- |

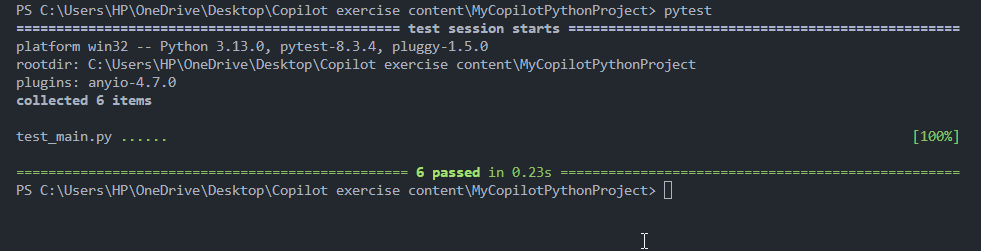
* 1. Accept the suggestion and adjust as necessary.



* 1. Save the file.

### **Run and Verify**

* 1. Run your tests (e.g., with pytest) to ensure the logging feature works.



* 1. Once all tests pass, (optionally) commit your changes:

| git add . git commit -m "Added logging feature in Python with limited context" |
| --- |

Voila!! We have successfully completed this exercise.