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

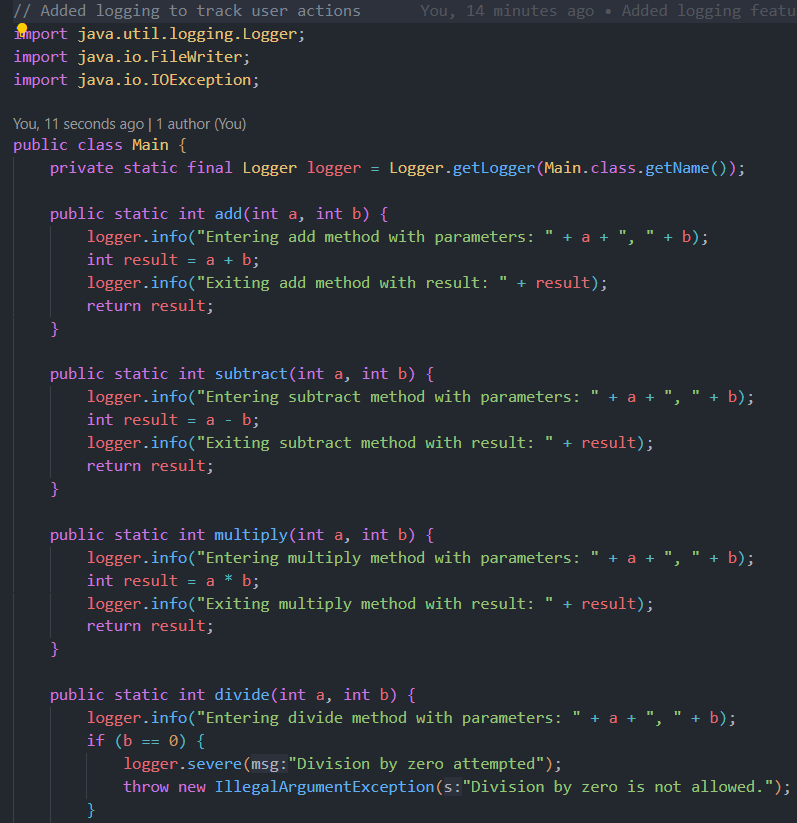
### **Objective:** Enhance your Java project by adding a logging feature to track user actions, even when the code has minimal context or documentation.

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

### **Review the Existing Code**

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

| // Need to add logging here to track user actions |
| --- |



### **Define the New Feature**

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

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

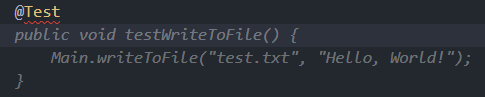
* 1. Open MainTest.java.
  2. Insert a comment:

| // Test for logging feature |
| --- |

* 1. Write a simple test:

| @Test public void testLogAction() {  Logger.logAction("Test action");  // Read the activity.log file to assert the message exists (using helper methods or libraries)  // For simplicity, assume the test passes if no exception is thrown. } |
| --- |

* 1. Allow Copilot to suggest improvements if available.



* 1. Save the test file.

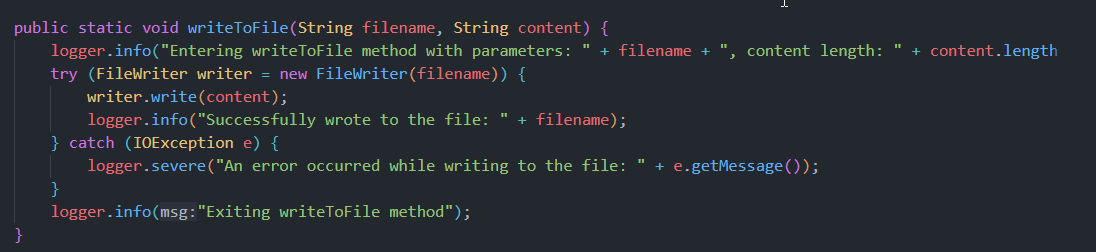
### **Implement the Logging Feature**

* 1. Return to Main.java.
  2. Add a comment:

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

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

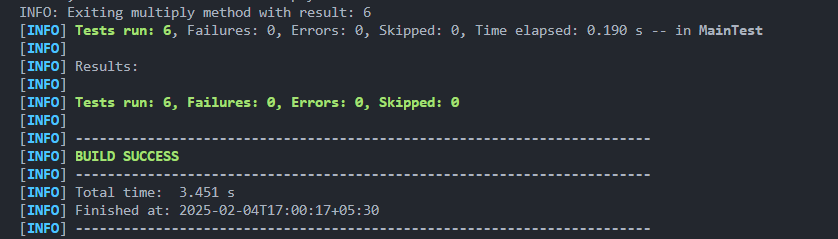
| import java.io.FileWriter; import java.io.IOException;  public class Logger {  public static void logAction(String action) {  try (FileWriter fw = new FileWriter("activity.log", true)) {  fw.write(action + "\n");  } catch (IOException e) {  e.printStackTrace();  }  } } |
| --- |



* 1. Accept the suggestion and adjust as necessary.
  2. Save the file.

### **Run and Verify**

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



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

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

Voila!! We have successfully completed this exercise.