WEEK 1 : Design patterns and principles Handson

Exercise 1 – Singleton Pattern

📌 **Objective:**
Implement a Singleton design pattern to ensure that only one instance of a logging utility class is created and used throughout the application.

CODE :

Logger.java :

Logger.java

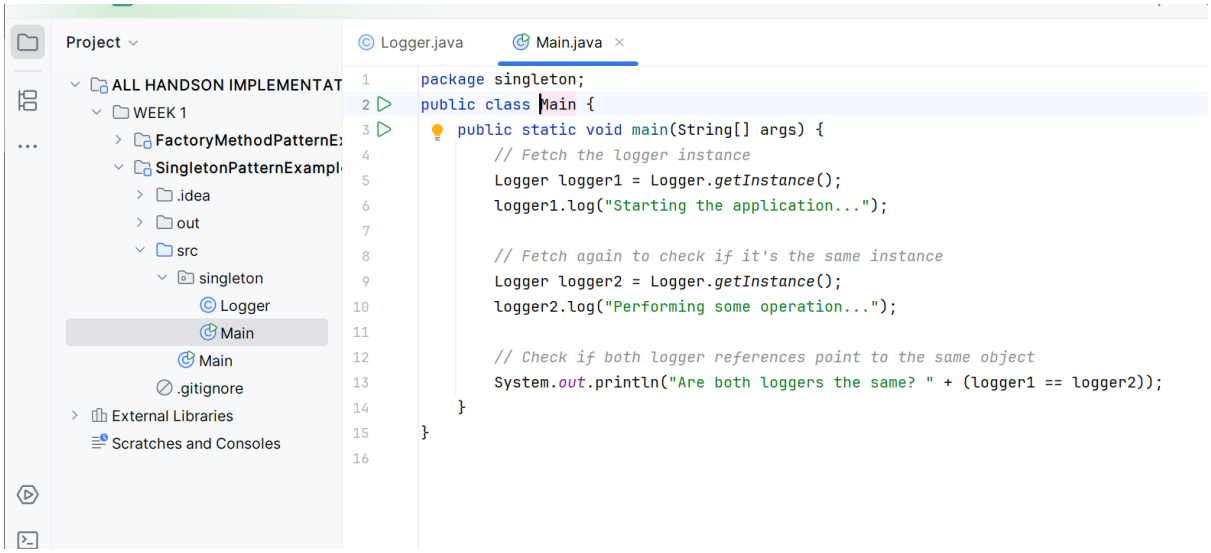
```java
package singleton;
public class Logger {  7 usages
    // Step 1: Create a private static instance of Logger (the single instance)
    private static Logger instance;  3 usages

    // Step 2: Private constructor to prevent instantiation
    private Logger() { System.out.println("Logger initialized."); }

    // Step 3: Public method to get the only instance of Logger
    public static Logger getInstance() {  2 usages
        if (instance == null) {
            instance = new Logger(); // Lazy initialization
        }
        return instance;
    }

    // Utility method to simulate logging
    public void log(String message) { System.out.println("[LOG]: " + message); }
}
```

Main.java :



OUTPUT :