#### 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:

```
C Logger.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
5
         // Step 2: Private constructor to prevent instantiation
           private Logger() { System.out.println("Logger initialized."); }
10
           // Step 3: Public method to get the only instance of Logger
           public static Logger getInstance() { 2 usages
13
              if (instance == null) {
                  instance = new Logger(); // Lazy initialization
14
              }
              return instance;
17
18
           // Utility method to simulate logging
19
           public void log(String message) { System.out.println("[LOG]: " + message); }
20
23
```

## Main.java:

```
□ Project ∨
                                     C Logger.java
                                                       \textcircled{6} Main.java \times
                                            package singleton;
      V CALL HANDSON IMPLEMENTAT
밁
                                      2 ▷ public class Main {
        WEEK 1
                                            public static void main(String[] args) {
                                      3 ⊳
          > 🕞 FactoryMethodPatternE
                                                    // Fetch the logger instance

✓ ☐ SingletonPatternExample

                                                     Logger logger1 = Logger.getInstance();
            🗀 .idea
                                                     logger1.log("Starting the application...");
            > 🗀 out
             ∨ 🗀 src
                                                     // Fetch again to check if it's the same instance
               singleton
                                                     Logger logger2 = Logger.getInstance();
                                     9
                    © Logger
                                     10
                                                     {\tt logger2.log("Performing some operation...");}
                   @ Main
                                                     // Check if both logger references point to the same object
                 @ Main
                                                     System.out.println("Are both loggers the same? " + (logger1 == logger2));
               .gitignore
                                     14
      > file External Libraries
                                            }
        Scratches and Consoles
\langle D \rangle
>_
```

#### **OUTPUT:**

```
Run Main ×

"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1.2\lib\idea_r Logger initialized.
[L06]: Starting the application...
[L06]: Performing some operation...
Are both loggers the same? true

Process finished with exit code 0
```