## **WEEK 1 Mandatory Hands On**

## Design Patterns and Principles Question:

Exercise 1: Implementing the Singleton Pattern

Scenario:

You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

Steps:

1. Create a New Java Project:

o Create a new Java project named SingletonPatternExample.

2. Define a Singleton Class:

o Create a class named Logger that has a private static instance of itself.

o Ensure the constructor of Logger is private.

o Provide a public static method to get the instance of the Logger class.

3. Implement the Singleton Pattern:

o Write code to ensure that the Logger class follows the Singleton design pattern.

4. Test the Singleton Implementation:

o Create a test class to verify that only one instance of Logger is created and used across the application.

## Solution:

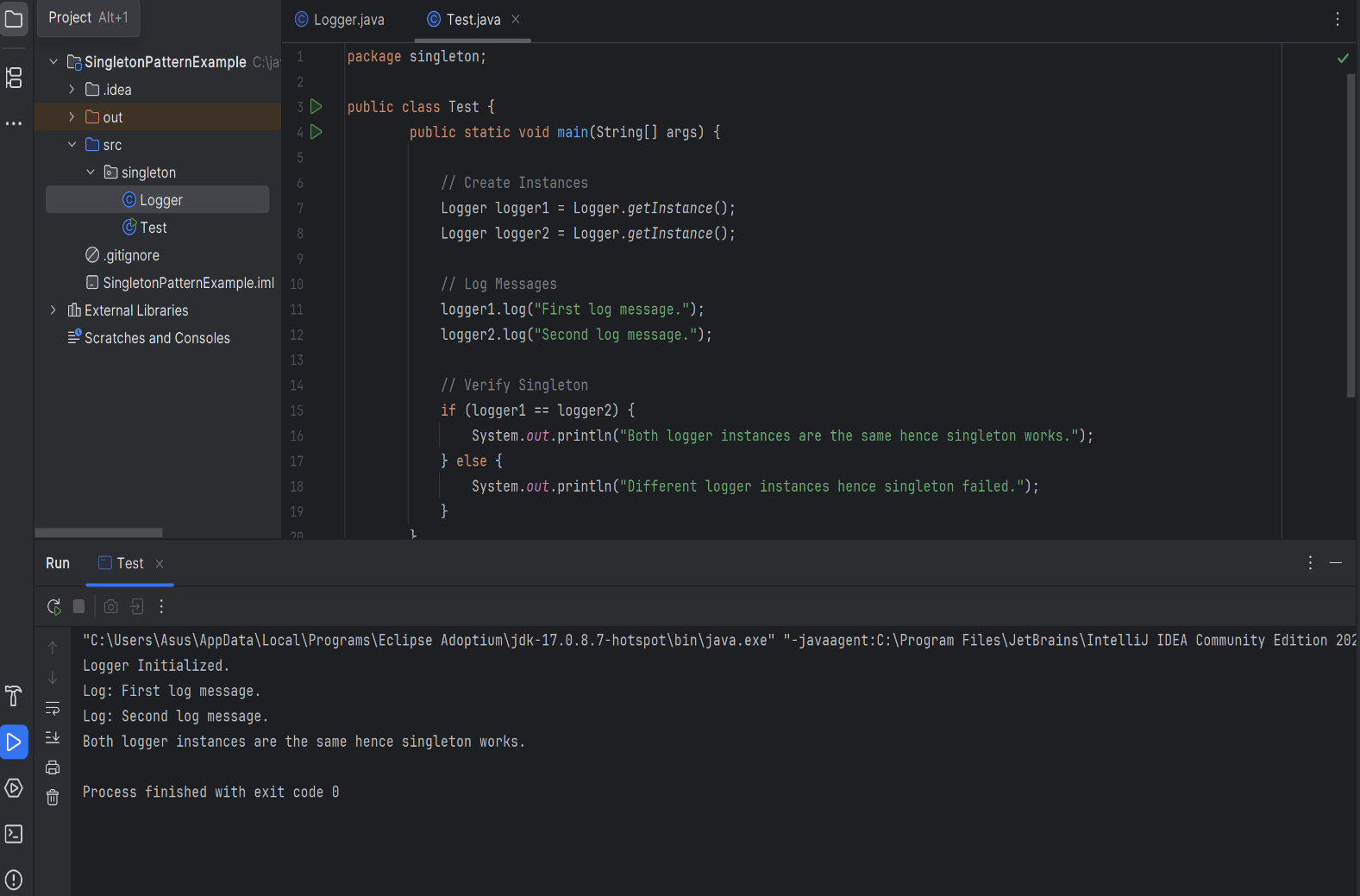
CODE: Logger.java

**package singleton;  
  
 public class Logger {  
  
 private static volatile Logger *singleInstance*;  
  
 private Logger() {  
 System.*out*.println("Logger Initialized.");  
 }  
 public static Logger getInstance() {  
 if (*singleInstance* == null){  
 synchronized (Logger.class){  
 if (*singleInstance* == null) {  
 *singleInstance* = new Logger();  
 }  
 }  
 }  
 return *singleInstance*;  
 }  
  
 public void log(String message) {  
 System.*out*.println("Log: " + message);  
 }  
 }**

CODE: Test.java

**package singleton;  
  
public class Test {  
 public static void main(String[] args) {  
  
 // Create Instances  
 Logger logger1 = Logger.*getInstance*();  
 Logger logger2 = Logger.*getInstance*();  
  
 // Log Messages  
 logger1.log("First log message.");  
 logger2.log("Second log message.");  
  
 // Verify Singleton  
 if (logger1 == logger2) {  
 System.*out*.println("Both logger instances are the same hence singleton works.");  
 } else {  
 System.*out*.println("Different logger instances hence singleton failed.");  
 }  
 }  
 }**

## OUTPUT:

****