**Exercise 1: Implementing the Singleton Pattern**

**Scenario:**

I ensure that a logging utility class in my application has only one instance throughout the application lifecycle to ensure consistent logging.

**Solution:**

**Logger.java**

public class Logger {

private static Logger instance;

private Logger() {

System.out.println("Logger instance created.");

}

public static Logger getInstance() {

if (instance == null) {

instance = new Logger(); // Create the instance if not already created

}

return instance;

}

public void log(String message) {

System.out.println("Log message: " + message);

}

}

**Main.java**

public class Main {

public static void main(String[] args) {

Logger log1 = Logger.getInstance();

Logger log2 = Logger.getInstance();

log1.log("First message");

log2.log("Second message");

if (log1 == log2) {

System.out.println("Both logger1 and logger2 refer to the same instance.")

} else {

System.out.println("Different instances of Logger exist.");

}

}

}

**Output:**

