**Module-1**

**Design patterns and principles**

**Exercise 1: Implementing the Singleton Pattern:**

**Logger.java:**

public class Logger {

private static Logger instance;

private Logger() {

System.out.println("Logger initialized");

}

public static Logger getInstance() {

if (instance == null) {

System.out.println("Creating new Logger instance");

instance = new Logger();

} else {

System.out.println("Returning existing Logger instance");

}

return instance;

}

public void log(String message) {

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

}

}

**Main.java:**

public class Main {

public static void main(String[] args) {

Logger logger1 = Logger.getInstance();

logger1.log("System started.");

Logger logger2 = Logger.getInstance();

logger2.log("Loading modules");

Logger logger3 = Logger.getInstance();

logger3.log("Application running");

if (logger1 == logger2 && logger2 == logger3) {

System.out.println("All logger references are the same instance");

} else {

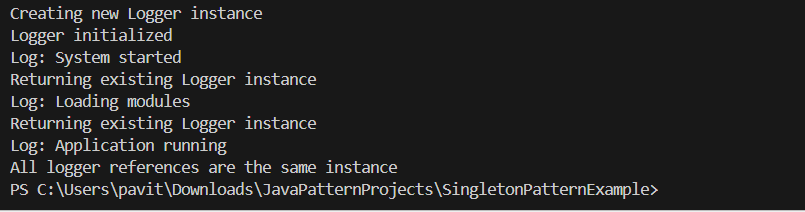
System.out.println("Different logger instances found ");

}

}

}

**Output:**

****