**Module 1 : DESIGN PATTERNS AND PRINCIPLES**

**Exercise 1: Implementing the Singleton Pattern**

**Code:**

**AppLogger.java — Singleton Implementation :**

public class AppMain {

public static void main(String[] args) {

AppLogger firstLogger = AppLogger.getLoggerInstance();

AppLogger secondLogger = AppLogger.getLoggerInstance();

firstLogger.writeLog("This is the first log message.");

secondLogger.writeLog("This is the second log message.");

if (firstLogger == secondLogger) {

System.out.println("Both firstLogger and secondLogger are the same instance. Singleton works!");

} else {

System.out.println("Different instances exist. Singleton failed.");

}

}

}

class AppLogger {

private static volatile AppLogger loggerInstance = null;

private AppLogger() {

System.out.println("AppLogger has been created.");

}

public static AppLogger getLoggerInstance() {

if (loggerInstance == null) {

synchronized (AppLogger.class) {

if (loggerInstance == null) {

loggerInstance = new AppLogger();

}

}

}

return loggerInstance;

}

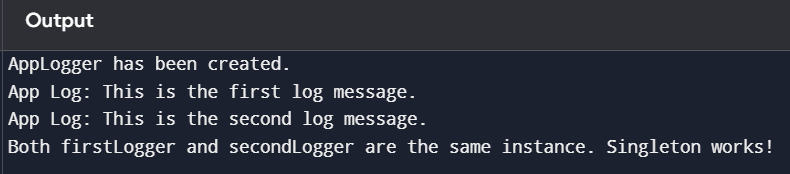
public void writeLog(String logMessage) {

System.out.println("App Log: " + logMessage);

}

}

**Output:**

****

**Exercise 2: Implementing the Factory Method Pattern**

**Code:**

**FactoryMethodPattern.java:**

public class AppMain {

public static void main(String[] args) {

Notification notification1 = NotificationFactory.createNotification("EMAIL");

notification1.notifyUser();

Notification notification2 = NotificationFactory.createNotification("SMS");

notification2.notifyUser();

Notification notification3 = NotificationFactory.createNotification("PUSH");

notification3.notifyUser();

}

}

interface Notification {

void notifyUser();

}

class EmailNotification implements Notification {

public void notifyUser() {

System.out.println("Sending an Email Notification");

}

}

class SMSNotification implements Notification {

public void notifyUser() {

System.out.println("Sending an SMS Notification");

}

}

class PushNotification implements Notification {

public void notifyUser() {

System.out.println("Sending a Push Notification");

}

}

class NotificationFactory {

public static Notification createNotification(String type) {

if (type == null || type.isEmpty()) {

return null;

}

if (type.equalsIgnoreCase("EMAIL")) {

return new EmailNotification();

} else if (type.equalsIgnoreCase("SMS")) {

return new SMSNotification();

} else if (type.equalsIgnoreCase("PUSH")) {

return new PushNotification();

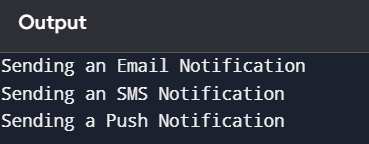
}

return null;

}

}

**Output:**

****