**Exercise 1: Implementing the Singleton Pattern**

**Scenario:**

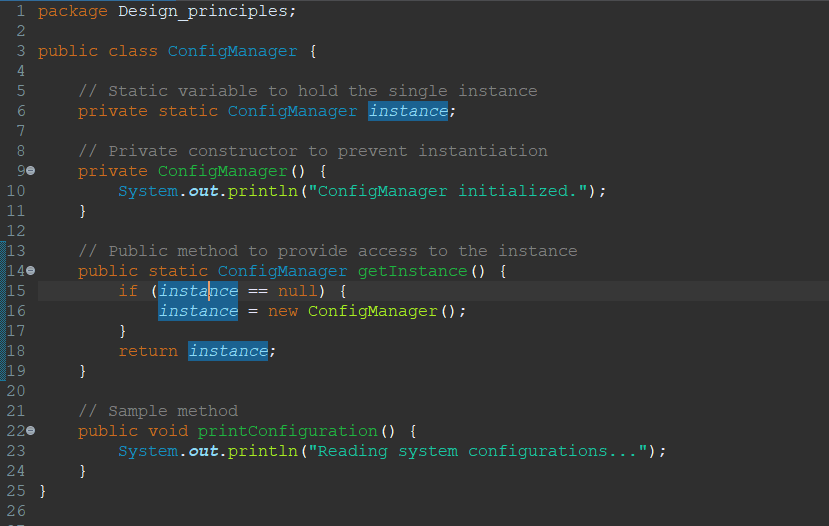
You are developing a configuration manager for your application. This manager should ensure that only one instance exists throughout the application lifecycle to prevent inconsistent configuration settings. You will implement this using the Singleton Design Pattern.

**Steps:**

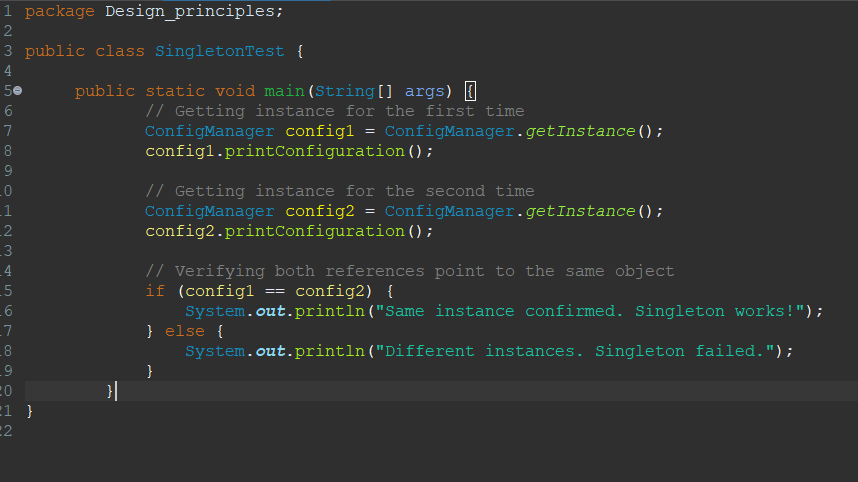
1. Create a new Java project named SingletonPatternExample.
2. Define a class ConfigManager that should follow the Singleton design principles.
3. Ensure:
   * A private static instance of ConfigManager.
   * A private constructor.
   * A public static method getInstance() to get the single instance.
4. Test the Singleton by creating multiple references and verifying they point to the same instance.

**Code:**

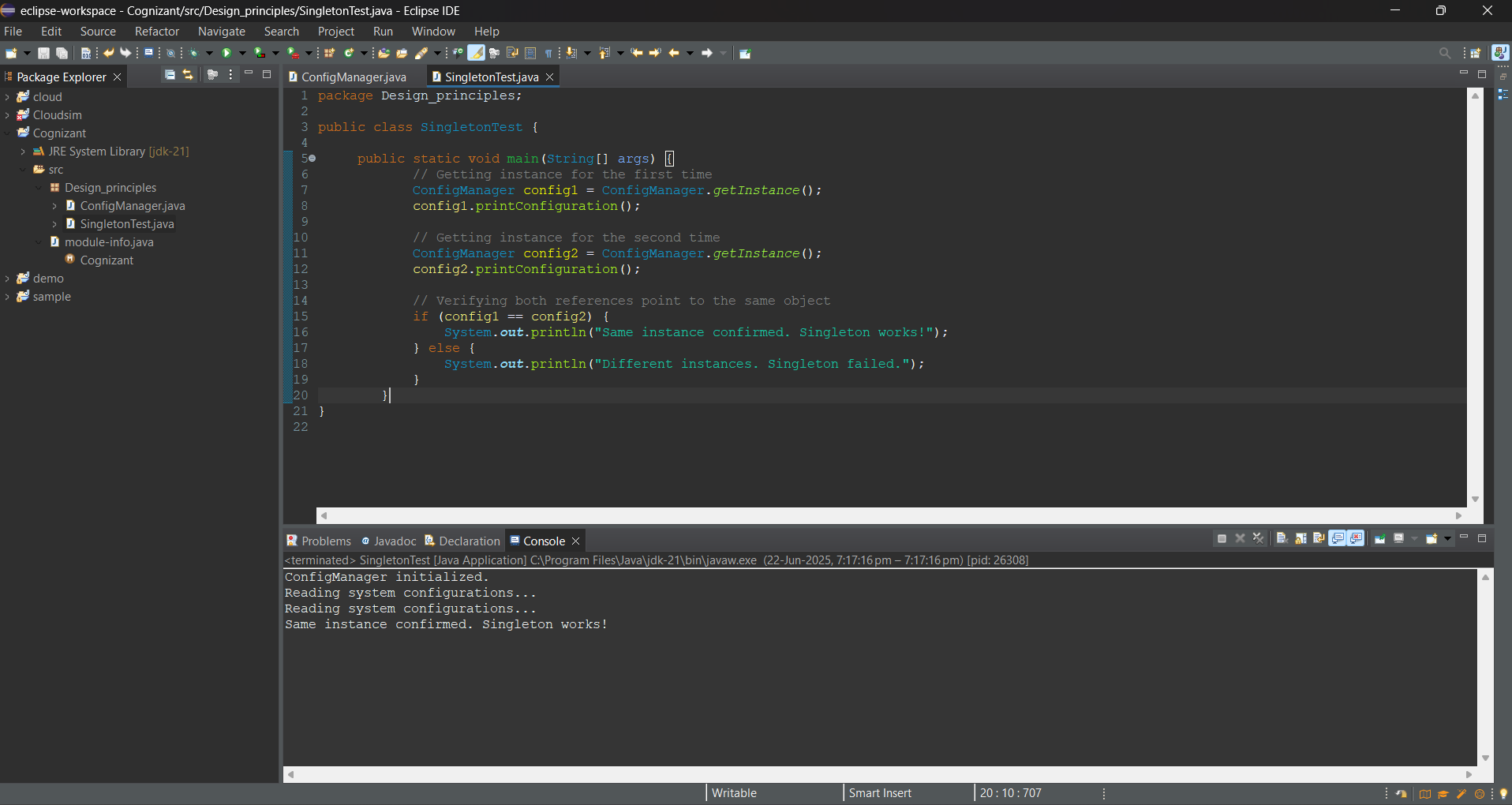
**ConfigManager Class**



**SingletonTest Class:**

****

**Output Screenshot:**



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

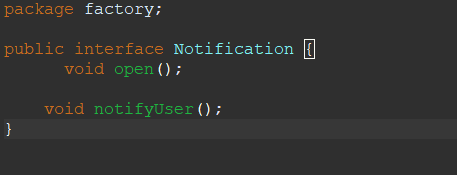
**Scenario:**

You are building a notification system that can send notifications via different channels such as Email, SMS, and Push Notifications. You want to instantiate the correct type of notification sender based on user preferences using the Factory Method Pattern.

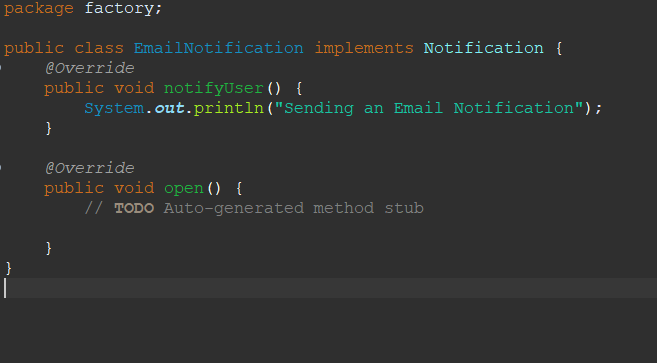
**Steps:**

1. Create a new Java project named FactoryMethodPatternExample.
2. Define an interface Notification with a method notifyUser().
3. Create concrete classes for different types of notifications (Email, SMS, Push).
4. Create an abstract class NotificationFactory with an abstract method createNotification().
5. Implement concrete factory classes to produce the correct notification object.
6. Create a main class to demonstrate the pattern.

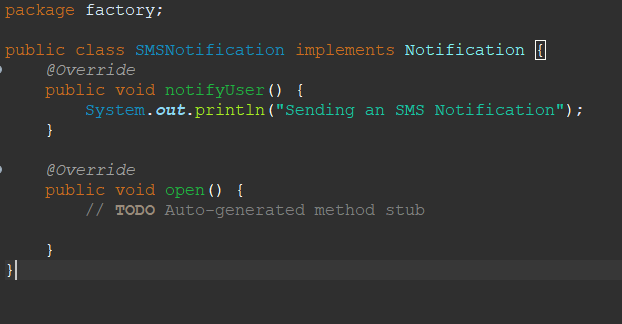
**Code**

**Notification Interface**

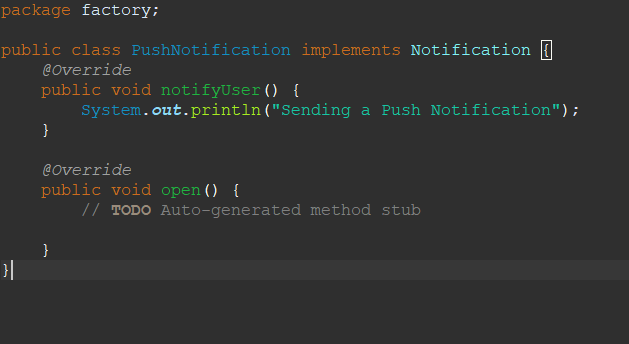
**EmailNotification Class**

****

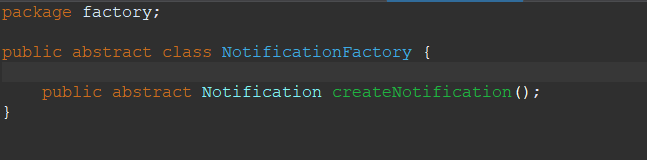
**SMSNotification Class**

****

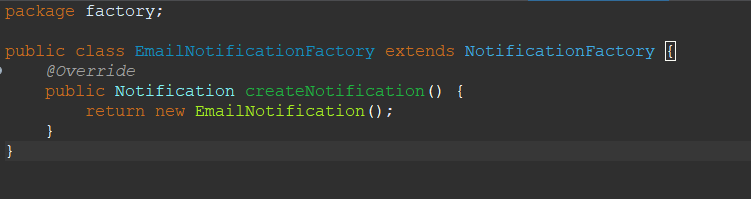
**PushNotification Class**

****

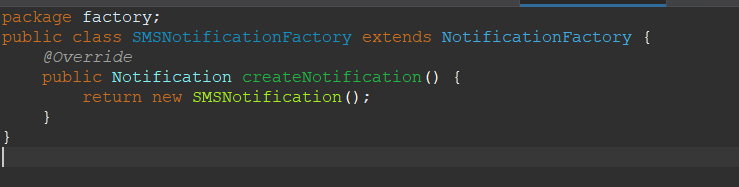
**NotificationFactory (Abstract)**

****

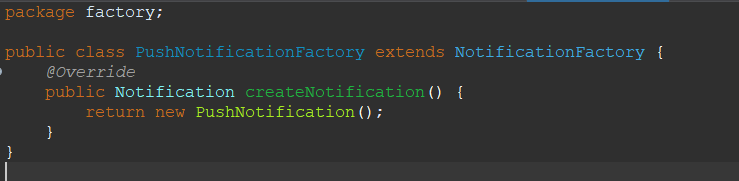
**EmailNotificationFactory class**

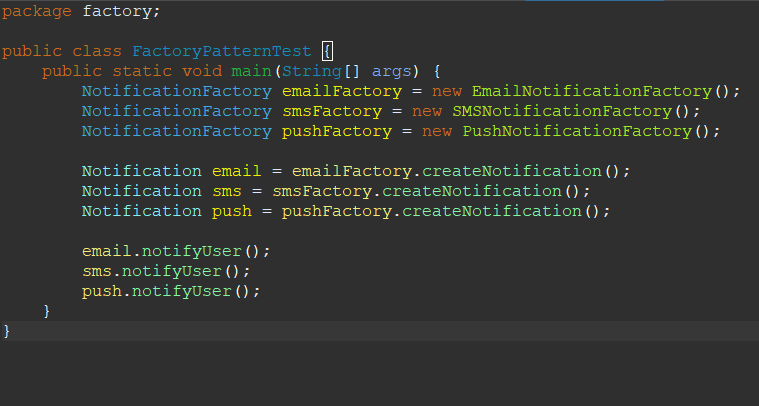
****

**SMSNotificationFactory Class**

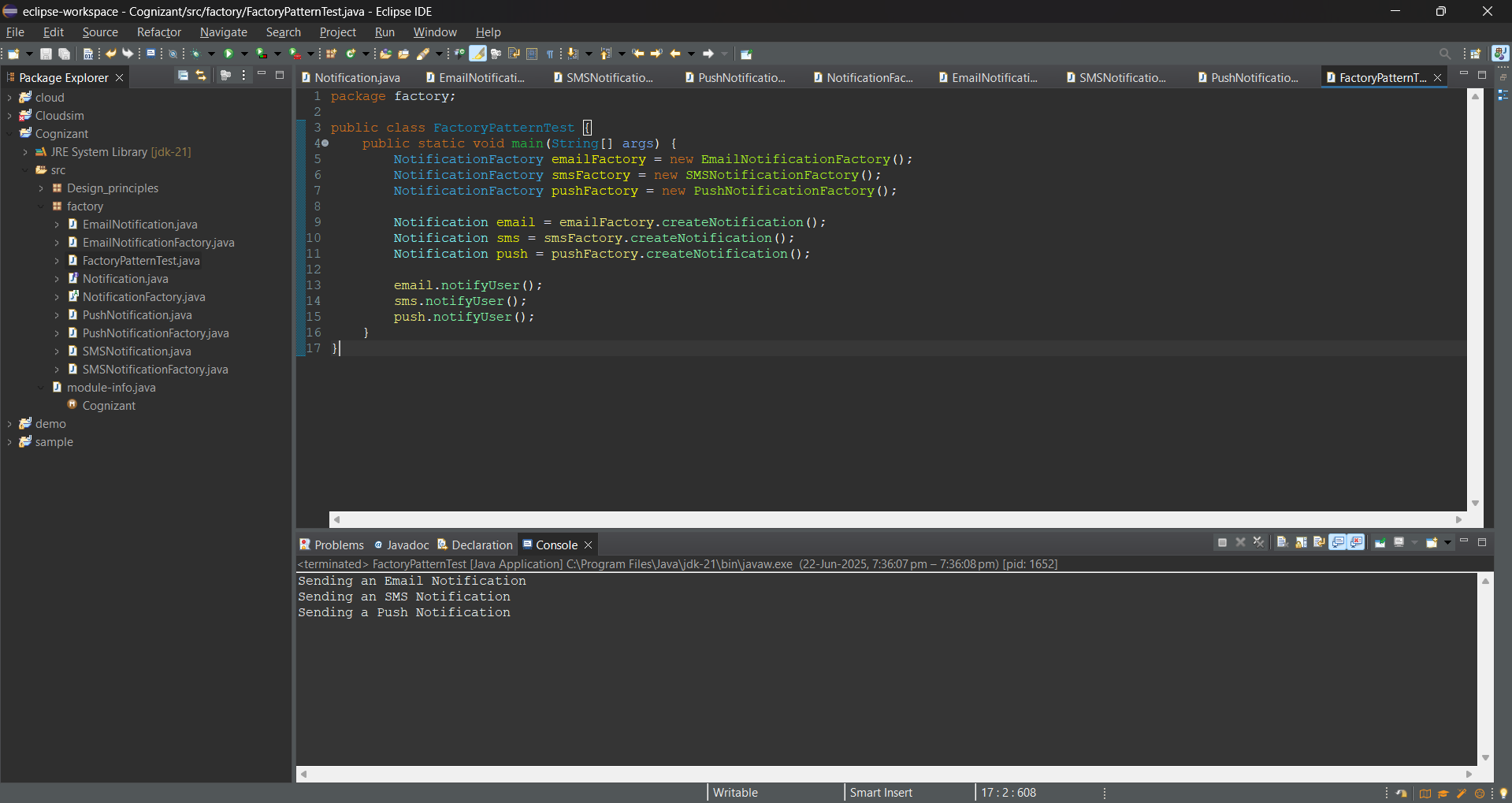
****

**PushNotificationFactory Class**

****

**FactoryPatternTest Class**

**Output Screenshot**

****