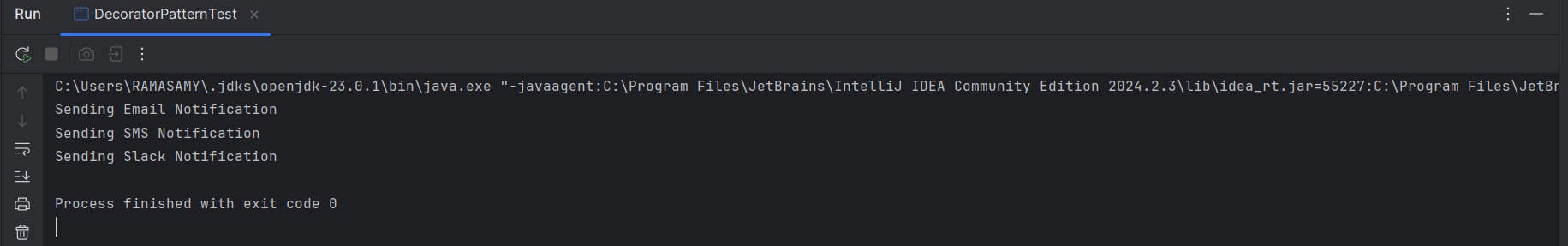
**Exercise 5: Implementing the Decorator Pattern**

**CODE:**

**DecoratorPatternTest.java**

interface Notifier {  
 void send();  
}  
  
class EmailNotifier implements Notifier {  
 public void send() {  
 System.*out*.println("Sending Email Notification");  
 }  
}  
  
abstract class NotifierDecorator implements Notifier {  
 protected Notifier notifier;  
 public NotifierDecorator(Notifier notifier) {  
 this.notifier = notifier;  
 }  
 public void send() {  
 notifier.send();  
 }  
}  
  
class SMSNotifierDecorator extends NotifierDecorator {  
 public SMSNotifierDecorator(Notifier notifier) {  
 super(notifier);  
 }  
 public void send() {  
 super.send();  
 System.*out*.println("Sending SMS Notification");  
 }  
}  
  
class SlackNotifierDecorator extends NotifierDecorator {  
 public SlackNotifierDecorator(Notifier notifier) {  
 super(notifier);  
 }  
 public void send() {  
 super.send();  
 System.*out*.println("Sending Slack Notification");  
 }  
}  
  
class DecoratorPatternTest {  
 public static void main(String[] args) {  
 Notifier notifier = new SlackNotifierDecorator(  
 new SMSNotifierDecorator(  
 new EmailNotifier()));  
 notifier.send();  
 }  
}

**OUTPUT:**

****