## Exercise 9: Implementing the Command Pattern

### Step 1: Create a New Java Project

Create a new Java project named CommandPatternExample.

### Step 2: Define Command Interface

Create an interface Command with a method execute().

// Command.java

public interface Command {

void execute();

}

### Step 3: Concrete Commands

Create classes LightOnCommand and LightOffCommand that implement Command.

// LightOnCommand.java

public class LightOnCommand implements Command {

private Light light;

public LightOnCommand(Light light) {

this.light = light;

}

@Override

public void execute() {

light.turnOn();

}

}

// LightOffCommand.java

public class LightOffCommand implements Command {

private Light light;

public LightOffCommand(Light light) {

this.light = light;

}

@Override

public void execute() {

light.turnOff();

}

}

### Step 4: Implement Invoker Class

Create a class RemoteControl that holds a reference to a Command and a method to execute the command.

// RemoteControl.java

public class RemoteControl {

private Command command;

public void setCommand(Command command) {

this.command = command;

}

public void pressButton() {

command.execute();

}

}

### Step 5: Implement Receiver Class

Create a class Light with methods to turn on and off.

// Light.java

public class Light {

public void turnOn() {

System.out.println("The light is on.");

}

public void turnOff() {

System.out.println("The light is off.");

}

}

### Step 6: Test the Command Implementation

Create a test class to demonstrate issuing commands using the RemoteControl.

// CommandPatternTest.java

public class CommandPatternTest {

public static void main(String[] args) {

Light light = new Light();

Command lightOn = new LightOnCommand(light);

Command lightOff = new LightOffCommand(light);

RemoteControl remote = new RemoteControl();

// Turn the light on

remote.setCommand(lightOn);

remote.pressButton();

// Turn the light off

remote.setCommand(lightOff);

remote.pressButton();

}

}

This example demonstrates the use of the Command Pattern to control a light in a home automation system. The RemoteControl class acts as the invoker, Light as the receiver, and LightOnCommand and LightOffCommand as concrete command classes. The CommandPatternTest class tests the implementation by issuing commands to turn the light on and off.

Run the CommandPatternTest class to see the output:

The light is on.

The light is off.

This shows that the command pattern has been successfully implemented to control the light.