**Exercise 9: Implementing the Command Pattern**

public class CommandPatternExample {

// Step 2: Command Interface

interface Command {

void execute();

}

// Step 5: Receiver Class

static class Light {

public void turnOn() {

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

}

public void turnOff() {

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

}

}

// Step 3: Concrete Command - Light ON

static class LightOnCommand implements Command {

private Light light;

public LightOnCommand(Light light) {

this.light = light;

}

public void execute() {

light.turnOn();

}

}

// Step 3: Concrete Command - Light OFF

static class LightOffCommand implements Command {

private Light light;

public LightOffCommand(Light light) {

this.light = light;

}

public void execute() {

light.turnOff();

}

}

// Step 4: Invoker Class

static class RemoteControl {

private Command command;

public void setCommand(Command command) {

this.command = command;

}

public void pressButton() {

if (command != null) {

command.execute();

} else {

System.out.println("No command set.");

}

}

}

// Step 6: Test

public static void main(String[] args) {

Light livingRoomLight = new Light();

Command lightOn = new LightOnCommand(livingRoomLight);

Command lightOff = new LightOffCommand(livingRoomLight);

RemoteControl remote = new RemoteControl();

System.out.println("Turning the light ON:");

remote.setCommand(lightOn);

remote.pressButton();

System.out.println("\nTurning the light OFF:");

remote.setCommand(lightOff);

remote.pressButton();

}

}

OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.