**Exercise 9: Implementing the Command Pattern**

**Command.java**

public interface Command {

void execute();

}

**Light.java**

public class Light {

public void turnOn() {

System.***out***.println("Light is ON.");

}

public void turnOff() {

System.***out***.println("Light is OFF.");

}

}

**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();

}

}

**RemoteControl.java**

public 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.");

}

}

}

**Main.java**

public class Main {

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("Pressing ON button:");

remote.setCommand(lightOn);

remote.pressButton();

System.***out***.println("\nPressing OFF button:");

remote.setCommand(lightOff);

remote.pressButton();

}

}

Output:

