**Exercise 9: Implementing the Command Pattern**

**Command.java**

package com.mycompany.commandpattern;

public interface Command {

void execute();

}

**Fan.java**

package com.mycompany.commandpattern;

public class Fan {

public void turnOn() {

System.out.println("Fan is ON");

}

public void turnOff() {

System.out.println("Fan is OFF");

}

}

**FanOffCommand.java**

package com.mycompany.commandpattern;

public class FanOffCommand implements Command {

private Fan fan;

public FanOffCommand(Fan fan) {

this.fan = fan;

}

public void execute() {

fan.turnOff();

}

}

**FanOnCommand.java**

package com.mycompany.commandpattern;

public class FanOnCommand implements Command {

private Fan fan;

public FanOnCommand(Fan fan) {

this.fan = fan;

}

public void execute() {

fan.turnOn();

}

}

**Light.java**

package com.mycompany.commandpattern;

public class Light {

public void turnOn() {

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

}

public void turnOff() {

System.out.println("ight is OFF");

}

}

**LightOffCommand.java**

package com.mycompany.commandpattern;

public class LightOffCommand implements Command {

private Light light;

public LightOffCommand(Light light) {

this.light = light;

}

public void execute() {

light.turnOff();

}

}

**LightOnCommand.java**

package com.mycompany.commandpattern;

public class LightOnCommand implements Command {

private Light light;

public LightOnCommand(Light light) {

this.light = light;

}

public void execute() {

light.turnOn();

}

}

**RemoteControl.java**

package com.mycompany.commandpattern;

public class RemoteControl {

private Command command;

public void setCommand(Command command) {

this.command = command;

}

public void pressButton() {

command.execute();

}

}

**CommandPattern.java**

package com.mycompany.commandpattern;

import java.util.Scanner;

public class CommandPattern {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

RemoteControl remote = new RemoteControl();

Light light = new Light();

Fan fan = new Fan();

Command lightOn = new LightOnCommand(light);

Command lightOff = new LightOffCommand(light);

Command fanOn = new FanOnCommand(fan);

Command fanOff = new FanOffCommand(fan);

while (true) {

System.out.println("\n--- Home Automation ---");

System.out.println("1. Turn ON Light");

System.out.println("2. Turn OFF Light");

System.out.println("3. Turn ON Fan");

System.out.println("4. Turn OFF Fan");

System.out.println("5. Exit");

System.out.print("Choose: ");

int choice = sc.nextInt();

switch (choice) {

case 1:

remote.setCommand(lightOn);

break;

case 2:

remote.setCommand(lightOff);

break;

case 3:

remote.setCommand(fanOn);

break;

case 4:

remote.setCommand(fanOff);

break;

case 5:

System.out.println("System Shut Down.");

sc.close();

return;

default:

System.out.println("Invalid choice.");

continue;

}

remote.pressButton();

}

}

}

**Output**