**Exercise 01.**

1. Implement an Interface called command with a method signature execute().

Java

Public interface command {

Void execute();

}

1. Implement an Interface called Light with method signatures on() and off().

Java

Public interface Light {

Void on();

Void off();

}

1. Create two concrete classes KitchenLight and LivingRoom wich implement Light interface.

Java

Public class Kitchenlight implements Light {

Public void on() {

System.out.println(“Kitchen light is ON”);

}

Public void off() {

System.out.println(“Kitchen light is OFF”);

}

}

Public class LivingRoomLight implements Light {

Public void on() {

System.out.println(“Living room light is ON”);

}

Public void off() {

System.out.println(“Living room light is OFF”);

}

}

1. Implement two concrete classes, LightOnCommand and LightOFFCommand, off the Command interface.

Java

Public class LightOnCommand implements Command {

Private Light light;

Public LightOnCommand(Light light) {

This.light = light

}

Public void execute() {

Light.on();

}

}

Public class LightOffCommand implements Command {

Private Light light;

Public LightOffCommand(Light light) {

This.light = light;

}

Public void execute() {

Light.off();

}

}

1. Now create a test class that checks every on and off command for each light.

Java

Public class CommandPattern Test {

Publc static void main(String[] args) {

Light KitchenLight = New KitchenLight();

Light livingRoomLight = New livingRoomLight();

Command KitchenLightOn = New LightOnCommand(KitchenLight);

Control KitchenLightOff = New LightOffCommand(KitchenLight);

Command livingRoomLightOn = New LightOnCommand(LivingRoomLight);

Command livingRoomLightOff = New LightOffCommand(LivingRoomLight);

KitchenLightOn.execute(); // Kitchen Light is ON

KitchenLightOff.execute(); // Kitchen Light is OFF

livingRoomLightOn.execute(); // Kitchen Light is ON

livingRoomLightOff.execute(); // Kitchen Light is OFF

}

}