**EXERCISE – 9**

**STRATEGY PATTERN EXAMPLE**

// CommandPatternExample.java  
  
interface Command {  
 void execute();  
}  
  
class Light {  
 public void turnOn() {  
 System.*out*.println("The light is on");  
 }  
  
 public void turnOff() {  
 System.*out*.println("The light is off");  
 }  
}  
  
class LightOnCommand implements Command {  
 private Light light;  
  
 public LightOnCommand(Light light) {  
 this.light = light;  
 }  
  
 @Override  
 public void execute() {  
 light.turnOn();  
 }  
}  
  
class LightOffCommand implements Command {  
 private Light light;  
  
 public LightOffCommand(Light light) {  
 this.light = light;  
 }  
  
 @Override  
 public void execute() {  
 light.turnOff();  
 }  
}  
  
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");  
 }  
 }  
}  
  
public class CommandPatternExample {  
 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();  
 }  
}