//Facade Pattern

class ClassA {

    public void operationA(){

        System.out.println("Performing methodA");

    }

}

class ClassB {

    public void operationB(){

        System.out.println("Performing methodB");

    }

}

class ClassC {

    public void operationC(){

        System.out.println("Performing methodC");

    }

}

class Facade {

    private ClassA objectA;

    private ClassB objectB;

    private ClassC objectC;

    Facade(){

        objectA = new ClassA();

        objectB = new ClassB();

        objectC = new ClassC();

    }

    void doTask(){

        System.out.println("Performing all subsystem methods");

        objectA.operationA();

        objectB.operationB();

        objectC.operationC();

    }

}

public class Main {

    public static void main(String args[]){

        Facade f = new Facade();

        f.doTask();

    }

}

Exercise:

Create a class called lights  
2 methods turnon turnoff ()  
  
fan class  
2 methods fanstart fanstop (fan stated, fan stopped)  
  
airconditioner class  
2 methods settemp turnoffac (air con temp set to , air conditioner turned off)  
  
  
Home facade  
2 methods leave home returnhome (turnoff all, turnon all)

//HomeFacade

class Light {

    public void turnOn() {

        System.out.println("All lights on..");

    }

    public void turnOff() {

        System.out.println("All lights off!");

    }

}

class Fan {

    public void fanStart() {

        System.out.println("Fan started..");

    }

    public void fanStop() {

        System.out.println("Fan stopped!");

    }

}

class AC {

    public void setTemp(int temp) {

        System.out.println("AC Turned on.. \nTemp: " + temp);

    }

    public void stopAc() {

        System.out.println("AC stopped!");

    }

}

class Home {

    private Fan fan;

    private Light light;

    private AC ac;

    Home() {

        fan = new Fan();

        light = new Light();

        ac = new AC();

    }

    public void turnOnAll(int setTemp) {

        fan.fanStart();

        ac.setTemp(setTemp);

        light.turnOn();

    }

    public void turnOffAll() {

        fan.fanStop();

        ac.stopAc();

        light.turnOff();

    }

}

public class Main {

    public static void main(String[] args) {

        Home h = new Home();

        h.turnOnAll(22);

        System.out.println();

        h.turnOffAll();

    }

}

//Adapter Pattern

interface Target {

    void request();

}

class Adaptee {

    public void specificRequest(){

        System.out.println("Specific Request");

    }

}

class Adapter implements Target {

    private Adaptee adaptee;

    public Adapter(){

        this.adaptee = new Adaptee();

    }

    public void request(){

        adaptee.specificRequest();

    }

}

public class Main {

    public static void main(String args[]){

        Adapter adapter = new Adapter();

        adapter.request();

    }

}

//Adapter Pattern

interface Target {

    void request();

}

class Adaptee {

    public void specificRequest(){

        System.out.println("Specific Request");

    }

}

class Adapter implements Target {

    private Adaptee adaptee;

    public Adapter(Adaptee adaptee){

        this.adaptee = adaptee;

    }

    public void request(){

        adaptee.specificRequest();

    }

}

public class Main {

    public static void main(String args[]){

        Adaptee adaptee = new Adaptee();

        Adapter adapter = new Adapter(adaptee);

        adapter.request();

    }

}

Define a interface for European power outlet EPO

Method plugin

Adaptee class US Power Outlet USPO

Method insertPlug(US power plug inserted)

Client class …

//Adapter Pattern

interface EPO {

    void plugin();

}

class USPO {

    public void insertPlug(){

        System.out.println("US power plug inserted..");

    }

}

class Adapter implements EPO {

    private USPO uspo;

    public Adapter(){

        uspo = new USPO();

    }

    public void plugin(){

        uspo.insertPlug();

    }

}

public class Main {

    public static void main(String args[]){

        Adapter adapter = new Adapter();

        adapter.plugin();

    }

}

//Adapter Pattern Real World Example

interface EuropeanPowerOutlet {

    void plugIn();

}

class USPowerPlug {

    void insertPlug(){

        System.out.println("US Plug inserted");

    }

}

class PowerAdapter implements EuropeanPowerOutlet {

    private USPowerPlug usPlug;

    PowerAdapter(){

        usPlug = new USPowerPlug();

    }

    public void plugIn(){

        usPlug.insertPlug();

        System.out.println("Us plug connected to european outlet using a power adapter");

    }

}

class ElectronicDevice {

    private EuropeanPowerOutlet european;

    ElectronicDevice(){

        european = new PowerAdapter();

    }

    public void powerOn(){

        european.plugIn();

    }

}

public class Main {

    public static void main(String args[]){

        ElectronicDevice ed = new ElectronicDevice();

        ed.powerOn();

    }

}

//pub-sub

import java.util.ArrayList;

import java.util.List;

class Publisher {

    private List<Subscriber> subscribers = new ArrayList<>();

    private int state;

    public void subscribe(Subscriber subscriber){

        subscribers.add(subscriber);

    }

    public void unsubscribe(Subscriber subscriber){

        subscribers.remove(subscriber);

    }

    private void notifySubscriber(){

        for(Subscriber subscriber: subscribers){

            subscriber.update();

        }

    }

    public int getState(){

        return state;

    }

    public void setState(int state){

        this.state = state;

        notifySubscriber();

    }

    public void showSubscribers() {

        System.out.println(subscribers.toString());

    }

}

class Subscriber {

    private Publisher publisher;

    private String subscriber;

    Subscriber(Publisher publisher, String subscriber){

        this.publisher = publisher;

        this.subscriber = subscriber;

    }

    public void update(){

        int newState = publisher.getState();

        System.out.println("New notification from subscriber, New Value : "+newState);

        System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

    }

    public String toString() {

        return this.subscriber;

    }

}

public class Client {

    public static void main(String args[]){

        Publisher publisher = new Publisher();

        Subscriber sub1 = new Subscriber(publisher, "sub1");

        Subscriber sub2 = new Subscriber(publisher, "sub2");

        publisher.subscribe(sub1);

        publisher.subscribe(sub2);

        publisher.showSubscribers();

        // publisher.setState(10);

        // publisher.unsubscribe(sub2);

        // publisher.setState(20);

    }

}

//News Agency

import java.util.ArrayList;

import java.util.List;

class News {

    private String heading;

    private String body;

    public News(String heading, String body) {

        this.heading = heading;

        this.body = body;

    }

    public String getHeading() {

        return heading;

    }

    public String getBody() {

        return body;

    }

}

interface Publisher {

    void subscribe(Subscriber subscriber);

    void unsubscribe(Subscriber subscriber);

    void notifySubscribers(News news);

}

class NewsAgency implements Publisher {

    private List<Subscriber> subscribers = new ArrayList<>();

    public void subscribe(Subscriber subscriber) {

        subscribers.add(subscriber);

    }

    public void unsubscribe(Subscriber subscriber) {

        subscribers.remove(subscriber);

    }

    public void notifySubscribers(News news) {

        for (Subscriber subscriber : subscribers) {

            subscriber.receiveNews(news);

        }

    }

    public void publishNews(String heading, String body) {

        News news = new News(heading, body);

        notifySubscribers(news);

    }

}

interface Subscriber {

    void receiveNews(News news);

}

class NewsSubscriber implements Subscriber {

    private String name;

    public NewsSubscriber(String name) {

        this.name = name;

    }

    public void receiveNews(News news) {

        System.out.println(name + " received news: " + news.getHeading() + " - " + news.getBody());

    }

}

public class Main {

    public static void main(String[] args) {

        NewsAgency newsAgency = new NewsAgency();

        Subscriber subscriber1 = new NewsSubscriber("Subscriber 1");

        Subscriber subscriber2 = new NewsSubscriber("Subscriber 2");

        newsAgency.subscribe(subscriber1);

        newsAgency.subscribe(subscriber2);

        newsAgency.publishNews("Breaking News", "Something important happened!");

        newsAgency.unsubscribe(subscriber1);

        newsAgency.publishNews("Exclusive News", "Only for subscriber 2!");

        newsAgency.unsubscribe(subscriber2);

        newsAgency.publishNews("Latest Update", "No subscribers, no notifications.");

    }

}

//MVC

//Model

class Student {

    private String name;

    private int age;

    public String getName(){

        return this.name;

    }

    public void setName(String name){

        this.name = name;

    }

    public int getAge(){

        return this.age;

    }

    public void setAge(int age){

        this.age = age;

    }

}

class StudentView {

    public void displayStudentDetails(String name, int age){

        System.out.println("Student Details : ");

        System.out.println("Name : "+name);

        System.out.println("Age : "+age);

    }

}

class StudentController {

    private Student model;

    private StudentView view;

    StudentController(Student model, StudentView view){

        this.model = model;

        this.view = view;

    }

    public void setName(String name){

        model.setName(name);

    }

    public void setAge(int age){

        model.setAge(age);

    }

    public void updateView(){

        view.displayStudentDetails(model.getName(),model.getAge());

    }

}

public class MVC {

    public static void main(String args[]){

        Student studentModel = new Student();

        StudentView studentView = new StudentView();

        StudentController studentController = new StudentController(studentModel,studentView);

        studentController.setName("Saswath");

        studentController.setAge(26);

        studentController.updateView();

    }

}

//MVC

//Model

class Student {

    private String name;

    private int age;

    private String email;

    public String getName(){

        return this.name;

    }

    public void setName(String name){

        this.name = name;

    }

    public int getAge(){

        return this.age;

    }

    public void setAge(int age){

        this.age = age;

    }

    public String getEmail(){

        return this.email;

    }

    public void setEmail(String email){

        this.email = email;

    }

}

class StudentView {

    public void displayStudentDetails(String name, int age, String email){

        System.out.println("Student Details : ");

        System.out.println("Name : "+name);

        System.out.println("Age : "+age);

        System.out.println("Email: " + email);

    }

}

class StudentController {

    private Student model;

    private StudentView view;

    StudentController(Student model, StudentView view){

        this.model = model;

        this.view = view;

    }

    public void setName(String name){

        model.setName(name);

    }

    public void setAge(int age){

        model.setAge(age);

    }

    public void setEmail(String email) {

        if (email.contains("@")) {

            model.setEmail(email);

        } else {

            model.setEmail(new String("Not valid email!!!"));

        }

    }

    public void updateView(){

        view.displayStudentDetails(model.getName(),model.getAge(), model.getEmail());

    }

}

public class MVC {

    public static void main(String args[]){

        Student studentModel = new Student();

        StudentView studentView = new StudentView();

        StudentController studentController1 = new StudentController(studentModel,studentView);

        studentController1.setName("Rohith");

        studentController1.setAge(26);

        studentController1.setEmail("rohith@gmail.com");

        studentController1.updateView();

        System.out.println();

        StudentController studentController2 = new StudentController(studentModel,studentView);

        studentController2.setName("Bharath");

        studentController2.setAge(27);

        studentController2.setEmail("bharathgmail.com");

        studentController2.updateView();

    }

}