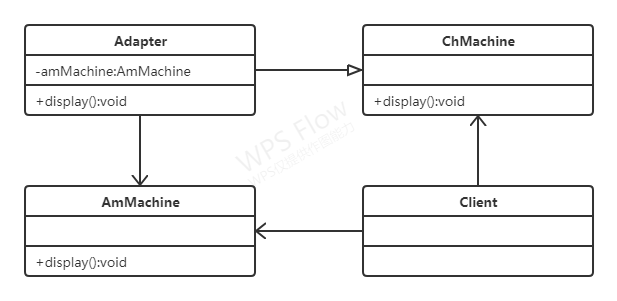
**实验六-课堂作业**

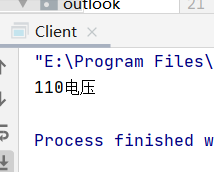
1. 变电站将发电站输出的10KV电压换成220V电压，变压器把一种电压变换成另一种电压。美国的生活用电电压是110V，而中国的电压是220V。如果要在中国使用美国电器，就必须有一个能把220V电压转换成110V电压的变压器。使用对象适配器模式设计类图，并给出代码。

类图：



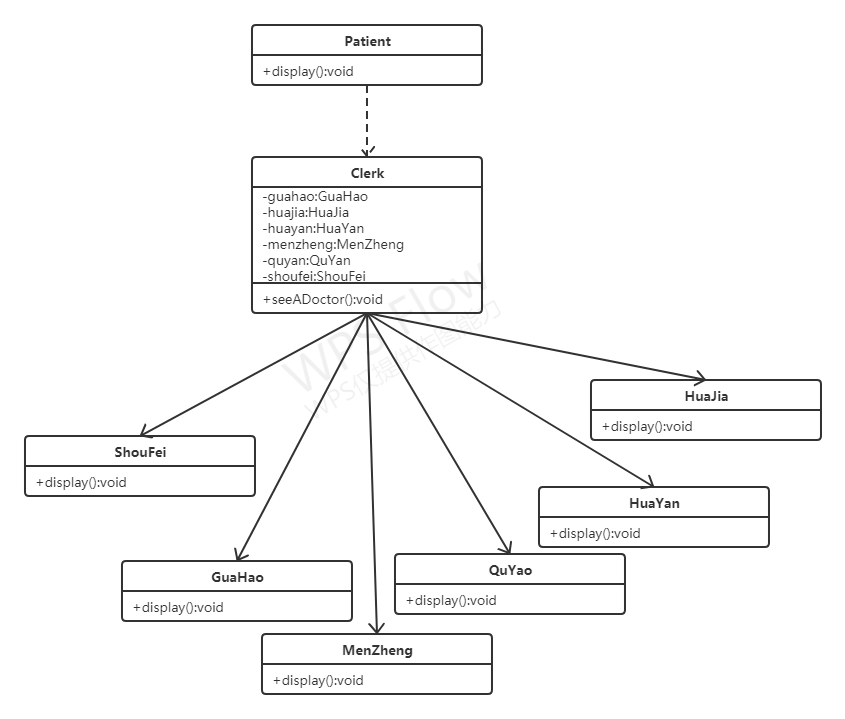
代码：

package experiment6.adapter;  
  
public class Adapter extends ChMachine{  
 private AmMachine amMachine;  
 public Adapter(AmMachine amMachine){  
 this.amMachine = amMachine;  
 }  
 @Override  
 public void display(){  
 amMachine.display();  
 }  
  
}  
  
class ChMachine{  
 public void display(){  
 System.*out*.println("220电压");  
 }  
}  
  
class Client{  
 public static void main(String[] args) {  
 AmMachine amMachine = new AmMachine();  
 ChMachine chMachine = new Adapter(amMachine);  
 chMachine.display();  
 }  
}  
  
class AmMachine{  
 public void display(){  
 System.*out*.println("110电压");  
 }  
}

结果：

1. 如果把医院作为一个子系统，按照部门职能，这个系统可以划分为挂号、门诊、划价、化验、收费、取药等。看病的病人要与这些部门打交道，就如同一个子系统的客户端与一个子系统的各个类打交道一样，不是一件容易的事情。首先病人必须先挂号，然后门诊。如果医生要求化验，病人必须首先划价，然后缴费，才能到化验部门做化验。化验后，再回到门诊室。为了解决这个问题，可以设置一个接待员的位置，由接待员负责代为挂号、划价、缴费、取药等，这个接待员就是外观模式的体现，病人只接触接待员，由接待员负责与医院的各个部门打交道。使用外观模式设计类图，并给出代码。

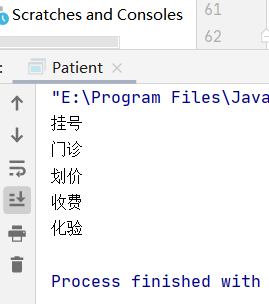
类图：



代码：

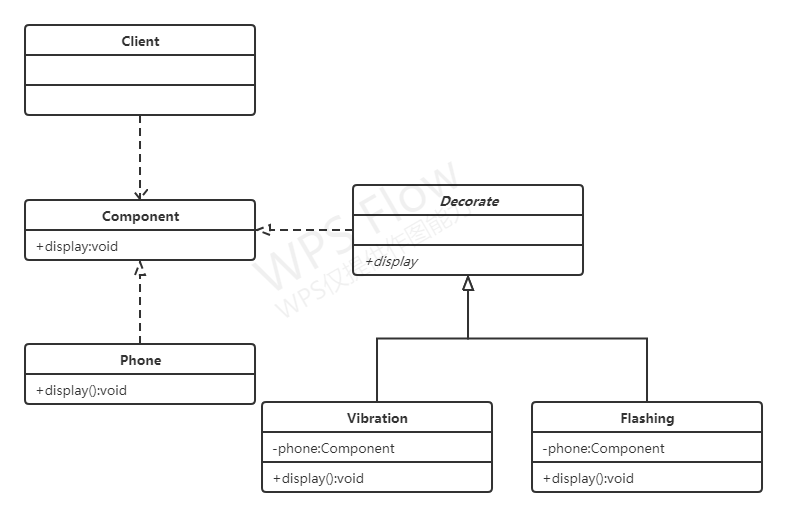
package experiment6.outlook;  
  
public class Clerk {  
 private GuaHao guaHao = new GuaHao();  
 private MenZheng menZheng = new MenZheng();  
 private HuaJia huaJia = new HuaJia();  
 private HuaYan huaYan = new HuaYan();  
 private ShouFei shouFei = new ShouFei();  
 private QuYao quYao = new QuYao();  
  
 public void SeeADoctor(){  
 guaHao.display();  
 menZheng.display();  
 huaJia.display();  
 shouFei.display();  
 huaYan.display();  
  
 }  
}  
  
class GuaHao{  
 public void display(){  
 System.*out*.println("挂号");  
 }  
}  
  
class MenZheng{  
 public void display(){  
 System.*out*.println("门诊");  
 }  
}  
  
class HuaJia{  
 public void display(){  
 System.*out*.println("划价");  
 }  
}  
  
class HuaYan{  
 public void display(){  
 System.*out*.println("化验");  
 }  
}  
  
class ShouFei{  
 public void display(){  
 System.*out*.println("收费");  
 }  
}  
  
class QuYao{  
 public void display(){  
 System.*out*.println("取药");  
 }  
}  
  
class Patient{  
  
 public static void main(String[] args) {  
 Clerk clerk = new Clerk();  
 clerk.SeeADoctor();  
 }  
}

结果：



1. 一部手机，在其接收到来电的时候，会发出声音来提醒主人，而现在我们需要为该手机添加一项功能，在接收来电的时候，产生振动，而另一部更加高级，不仅发声，而且产生振动和灯光闪烁。使用装饰模式设计类图，并给出代码。

类图：



代码：

package experiment6.decorate;  
  
public interface Component {  
 public void display();  
}  
  
class Phone implements Component {  
  
 @Override  
 public void display() {  
 System.*out*.println("发出声音");  
 }  
}  
  
abstract class Decorate implements Component {  
 public abstract void display();  
}  
  
class Vibration extends Decorate {  
  
 private Component phone;  
  
 public Vibration(Component phone) {  
 this.phone = phone;  
 }  
  
 @Override  
 public void display() {  
 phone.display();  
 System.*out*.println("产生振动");  
  
 }  
}  
  
class Flashing extends Decorate {  
  
 private Component phone;  
  
 public Flashing(Component phone) {  
 this.phone = phone;  
 }  
  
 @Override  
 public void display() {  
 phone.display();  
 System.*out*.println("闪烁");  
  
 }  
}  
  
class Client{  
 public static void main(String[] args) {  
 Component phone = new Phone();  
 System.*out*.println("\n基础款：");  
 phone.display();  
  
 phone = new Vibration(phone);  
 System.*out*.println("\n带震动款：");  
 phone.display();  
  
 phone = new Flashing(phone);  
 System.*out*.println("\n带闪烁款：");  
 phone.display();  
  
 }  
}

结果：

