## 第二十三章 状态模式

1. **以下关于状态模式的叙述错误的是（D）**
   1. 状态模式允许一个对象在其内部状态改变时改变它的行为。对象看起来似乎修改了它的类。
   2. 状态模式中引入了一个抽象类来专门表示对象的状态，而具体的状态都继承了该类，并实现了不同状态的行为，包括各种状态之间的转换。
   3. 状态模式使得状态的改变更加清晰明了，也容易创建对象的新状态。
   4. 状态模式完全符合开闭原则，增加新的状态类无需对原有类库进行任何修改。
2. **场景(C)不是状态模式的实例。**
   1. 银行账户根据余额不同用于不同的存取款操作
   2. 游戏软件中根据虚拟角色的不同拥有不同的权限
   3. 某软件在不同的操作系统中程序不同的外观
   4. 在会员系统中会员等级操作可以实现不同的行为
3. 分析一下代码：

public class TestXYZ{

private int behaviour;

//Getter and Setter

public void handleAll(){

if (behaviour == 0) {//do something

}else if (behaviour ==1) {//do something

}else if (behaviour ==2) {//do something

}else if (behaviour ==3) {//do something

}...some more esle if ...

}

}

为了提高代码的可扩展性和健壮性，可以使用(D)设计模式进行重构。

* 1. 访问者
  2. 外观
  3. 备忘录
  4. 状态