**七大设计原则：**

1。单一原则：

2。开闭原则：OCP

3。里氏代换原则：LCP

4。依赖倒置原则：DIP

5。接口隔离原则：ISP

6。合成复用原则：CRP

7。迪米特法则：DP

**创建型模式：**

**用来帮助我们创建对象**

1。单例模式

2。简单工厂

3。工厂方法

4。抽象工厂

5。建造者模式：分离了对象子组件的单独构造和装配

6。原型模式： 通过new产生一个对象需要非常繁琐的数据准备或者访问权限

**结构型模式：**

**关注对象和类的组织**

1。代理模式：为真实的对象提供一个代理，从而控制对真实对象的访问

2。设配器模式： 使原本由于接口不兼容不能一起工作的类一起工作。 2种？！！！

3。桥接模式： 处理多层继承结构，处理多维度变化的场景，将各个维度设计成独立的继承结构，时 各个维度独立的可以在抽象层建立关联

4。组合模式： 将对象组合成树状结构以表示部分和整体的层次结构，使客户可以统一的调用叶子对象和容器对象。

5。装饰模式： 动态的给一个对象添加额外的功能， 比继承灵活

6。外观模式： 给子系统提供统一的调用接口，使得子系统更加容易使用

7。共享模式： 运用共享技术有效的管理大量细粒度的对象，洁身内存，提高效率。

**行为型模式：**

**关注系统中对象之间的相互交互，研究系统在运行时对象之间的相互通信和协作，进一步明确对象的职责。**

1。职责连模式：避免请求接送者和发送者耦合，让多个对象都有可能接受请求，将这些对象连成一条链，并且沿着这条链传递请求， 直到有对象处理为止。

2。命令模式：将一个请求封装成一个对象，从而使得请求的调用者和请求的接受者解耦。

3。解释器模式： 描述如何为语言定义一个文法，如何解析

4。迭代器模式： 提供了一种方法来访问聚合对象

5。中介者模式： 通过一个中介对象来封装一系列的对象交互，使得各对象不需要相互引用

6。备忘录模式： 捕获一个对象的内部状态，并保存之。需要时，可以恢复到保存状态

7。观察者模式： 当一个对象状态发生改变时，其相关依赖对象皆得到通知并被更新

8。状态模式： 允许一个对象在自己内部状态改变时改变其行为

9。策略模式： 定义一系列的算法，并将每个算法封装在一个类中

10。模板方法： 定义一个操作的算法骨架，将某些易变的步骤延迟到子类中实现。

11。访问者模式： 表示一个作用于某对象结构中的各元素的操作，它使的用户可以在不改变各元素的类的前提下定义作用于这些元素的新操作。。。