# 面向对象设计原则

## 单一责任原先SRP

就一个类而言，应该仅有一个引起它变化的原因。

也就是说一个类应该只负责一件事情。如果一个类中存在方法1和方法2，我们修改方法1的时候会影响方法，也需要修改方法2，因此我们可以把方法1和方法2分离，单独成类。

好处：

降低复杂度，一个只负责一项责任。

提高可读性，和维护性。

修改的时候影响降到最低。

## 开闭原则 OCP

软件中的对象（类，模块，函数等）应该对于扩展是开放的，但是，对于修改是封闭的。

1. 增加需求的时候不修改内部代码
2. 通过继承来实现

## 里氏替换原则 LSP

所有引用基类的地方必须能够透明的使用其子类对象。

简单的说：子类可以去扩展父类的功能，但是比你修改父类原有的功能。

1. 子类可以实现父类的抽象方法，但不能覆盖
2. 子类可以增加自己独有的方法
3. 子类大大重载父类方法的时候，方法的形参要比父类的更加宽松。
4. 子类实现父类的抽象方法时候，方法的返回要比父类更加严格。

## 依赖倒置原则 DIP

特定的解耦形式，使得高层的模块不依赖于低层次模块的实现细节，以来模块被倒置了。

简单的说：

1. 高层次不依赖对方的底层模块，双方都依赖对方的抽象。
2. 抽象不依赖细节
3. 细节应该依赖抽象

## 接口隔离原则 ISP

类间的依赖关系应该建立在最小的接口上。简单的说就是让客户端依赖的接口尽可能的小。

一个接口方法写的太多，很多类是不需要实现这么多类的，因此可以把这个拆分开，分成还几个接口。this is a接口隔离。

## 迪米特原则（最小知识原则）LOD

一个对象应该对其他对象有最少的了解。

简单的说：就是让对象与最直接的朋友进行联系。

每个对象必然有其他类有耦合的关系，两个对象之间互相耦合就为朋友关系，比如：组合、聚合、依赖等。