设计模式是前辈们对代码开发经验的总结，是解决特定问题的一系列套路。它不是语法规定，而是一套用来提高代码可复用性、可维护性、可读性、稳健性以及安全性的解决方案。

设计模式的六大原则：

1. 开闭原则（总原则）对扩展开放，对修改关闭
   1. 在程序进行扩展的时候，不能去修改原有的代码，要扩展现有的代码
   2. 想达到这样的效果就需要用到接口和抽象类
2. 单一职责原则
   1. 每个类应该实现单一的功能，否则就应该把类拆分
3. 里氏替换原则
   1. 继承关系中，任何基类可以出现的地方，都可以用子类替换。比如 在 子类1:抽象类/接口 和 子类2：抽象类/接口 的关系中， 在方法中只需要实例化基类，且等号右边装入子类，就可以使用子类1和2的方法，实现不同的功能。接口也是一样的道理。 只需要实例化一个基类就行。
4. 依赖倒置原则
   1. 上层模块和底层模块 都依赖于抽象
   2. 上层调用抽象，底层根据抽象的规范定义实现
5. 接口隔离原则
   1. 每个接口不应该存在子类用不到却必须实现的方法，如果不然，那就将接口拆分。
6. 迪米特法则
   1. 一个类对自己依赖的类知道的越少越好。无论被依赖的类多么复杂，都应该将逻辑封装在方法的内部，通过public方法提供给外部。这样当被依赖的类变化时，才能最小的影响该类。
7. 合成复用原则
   1. 合成或聚合可以将已有对象纳入到新对象中，使之成为新对象的一部分，因此新对象可以调用已有对象的功能。