面向对象

•一切皆为对象 ，万物可分类

•对象 –—-类的实体\一个叫张三的男人

•类——人\动物\机器

•方法—–人会走,会思考\狗会叫,会咬人\定义一个类的各个功能

•消息传递—–狗叫了,人听见了,就叫通信

•继承——狗都四条腿走路

•封装——人不能引用狗的特性,比如四条腿走路

•多态性—–一个叫的功能,可能是低吼,也可是大声叫

•抽象性——简化复杂的现实问题的途径，它可以为具体问题找到最恰当的类

类：具有相同属性和行为事物的统称。  
对象：对象是类的具体实例

ooa 分析

类与类行为不同，对象与对象数据不同。

先分析对象，然后抽象升华到类

老师可以是一个类 ，可以具有性别，年龄等数据

可以做饭，洗衣等事情 --- 行为

对象：张三老师 女 25 做饭

**类与类的关系**

**泛化：**子类与父类的关系，概念的复用，***耦合度最高***；　　　　（由子到父　就是泛化）

B类泛化A类，意味B类是A类的一种；

做法：B类继承A类

**关联(聚合/组合)：**部分与整体的关系，功能的复用，变化影响一个类；　**耦合度比较低**

A与B关联，意味着B是A的一部分；

做法：在A类中包含B类型成员。

**依赖：**合作关系，一种相对松散的协作，变化影响一个方法；　　　　**耦合度最低**

A类依赖B类，意味A类的某些功能靠B类实现；

做法：B类型作为A类中方法的参数，并不是A的成员。

ood 设计 oop 编程

**三大特征**  **封装 继承 多态**

**封装**

1. 分而治之

类

类

类

类

-- 将一个大的需求分解为许多类，每个类处理一个独立的功能。

-- 拆分好处：便于分工，便于复用，可扩展性强。

(2) 封装变化

-- 变化的地方独立封装，避免影响其他类。

类

方法

方法

(3) 高 内 聚

-- 类中各个方法都在完成一项任务(单一职责的类)。

类

类

类

(4) 低 耦 合

-- 类与类的关联性与依赖度要低(每个类独立)，让一个类的改变，尽少影响其他类。

**继承**

父子类： 子类拥有父类的所有成员。

子类如果没有构造函数，将自动执行父类的，但如果有构造函数将覆盖父类的

**多态** ***１．调用了父类，２．执行了子的不同行为　叫多态***

**原则：**

**开闭原则**：允许扩展新的，但不允许对原有的进行修改

**单一原则**：一个类有且只有一个改变它的原因。

**依赖倒置**：客户端调用的类尽量使用抽象(父类)，不使用具体(子类)。父类稳定，子类多变。

**组合复用：**仅仅为了代码复用优先选择组合复用，而非继承复用

**里式替换：**继承后的重写，指导继承的设计

父类出现的地方可以被子类替换，在替换后依然保持原功能。

子类要拥有父类的所有功能。

子类在重写父类方法时，尽量选择扩展重写，防止改变了功能。

**迪米特法则： 低耦合**