1. **编程范式：**

范，模范、典范也。范式即模式、方法。常见的编程范式有：函数式编程、程序编程、面向对象编程、指令式编程等。用特定的语法+数据结构+算法组成代码。

1. **面向对象编程：**

所以我们一般认为， 如果你只是写一些简单的脚本，去做一些一次性任务，用面向过程的方式是极好的，但如果你要处理的任务是复杂的，且需要不断迭代和维护 的， 那还是用面向对象最方便了。

1. **面向对象的几个核心对象：**

核心特性：

**Class 类**  
一个类即是对一类拥有相同属性的对象的抽象、蓝图、原型。在类中定义了这些对象的都具备的属性（variables(data)）、共同的方法

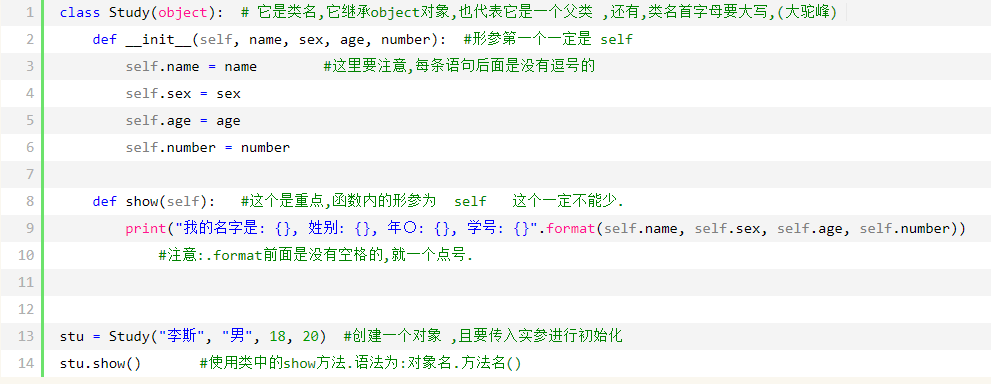
**Object 对象**  
一个对象即是一个类的实例化后实例，一个类必须经过实例化后方可在程序中调用，一个类可以实例化多个对象，每个对象亦可以有不同的属性，就像人类是指所有人，每个人是指具体的对象，人与人之前有共性，亦有不同

**Encapsulation 封装**  
在类中对数据的赋值、内部调用对外部用户是透明的，这使类变成了一个胶囊或容器，里面包含着类的数据和方法

**Inheritance 继承**  
一个类可以派生出子类，在这个父类里定义的属性、方法自动被子类继承

**Polymorphism 多态**  
多态是面向对象的重要特性,简单点说:“一个接口，多种实现”，指一个基类中派生出了不同的子类，且每个子类在继承了同样的方法名的同时又对父类的方法做了不同的实现，这就是同一种事物表现出的多种形态。  
编程其实就是一个将具体世界进行抽象化的过程，多态就是抽象化的一种体现，把一系列具体事物的共同点抽象出来, 再通过这个抽象的事物, 与不同的具体事物进行对话。  
对不同类的对象发出相同的消息将会有不同的行为。比如，你的老板让所有员工在九点钟开始工作, 他只要在九点钟的时候说：“开始工作”即可，而不需要对销售人员说：“开始销售工作”，对技术人员说：“开始技术工作”, 因为“员工”是一个抽象的事物, 只要是员工就可以开始工作，他知道这一点就行了。至于每个员工，当然会各司其职，做各自的工作。  
多态允许将子类的对象当作父类的对象使用，某父类型的引用指向其子类型的对象,调用的方法是该子类型的方法。这里引用和调用方法的代码编译前就已经决定了,而引用所指向的对象可以在运行期间动态绑定

1. **类的语法：**

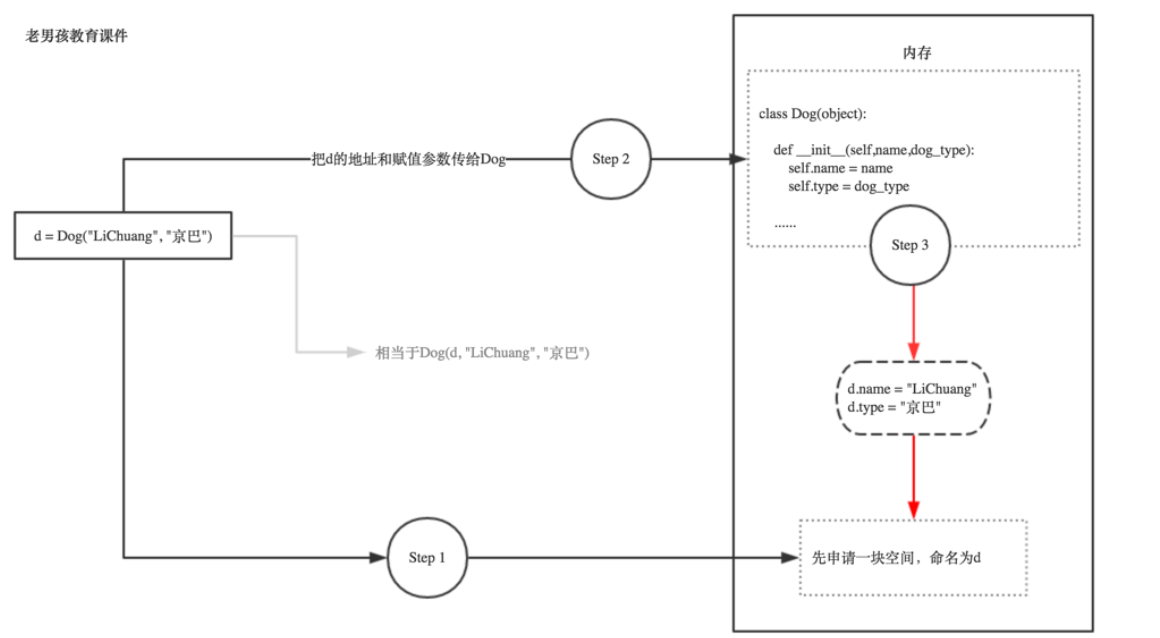


附：def \_\_init\_\_(self,…,…,…,…): 初始化函数，初始化类属性。

下面的def show(self)为类内方法

Stu = Study(“李斯”，“男”，18，20)：创建一个对象，对类的实例化。

1. **类的实例化过程：**



1. **面向对象的特性：**

封装：封装最好理解了。封装是面向对象的特征之一，是对象和类概念的主要特性。

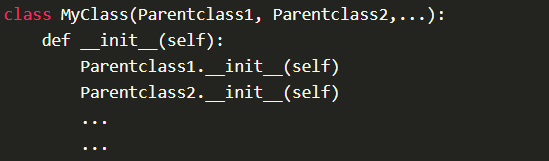
封装，也就是把客观事物封装成抽象的类，并且类可以把自己的数据和方法只让可信的类或者对象操作，对不可信的进行信息隐藏。

继承：继承概念的实现方式有两种：实现继承和接口继承



一个类可以继承自多个类，具有父类的所有变量和方法，语法如下： ...

...



多态：一个接口多种实现，Pyhon 很多语法都是支持多态的，比如 len(),sorted(), 你给len传字符串就返回字符串的长度，传列表就返回列表长度。



领域模型：待学习