1. python面向对象编程：

(1)创建类：

 class Person(object): #(object)表示该类是从哪个类继承下来的，类名以大写字母开头

    example....

(3)创建实例：

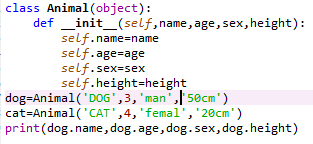
xiaoMing=Person()

1. 创建实例属性

实例被创建后，可以直接添加属性，并给其属性赋值：xiaoMing.name='小明'

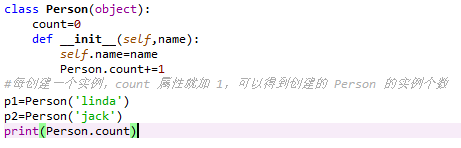
1. 初始化实例属性：创建类时，添加一个特殊的\_\_init\_\_()方法，当创建实例时，\_\_init\_\_()方法被自动调用，我们就能在此为每个实例都统一加上多个属性

示例：



1. python访问限制：Python对属性权限的控制是通过属性名来实现的，如果一个属性由双下划线开头(\_\_)，该属性就无法被外部访问；如果一个属性以"\_\_xxx\_\_"的形式定义，那它又可以被外部访问了，以"\_\_xxx\_\_"定义的属性在Python的类中被称为特殊属性，通常我们不使用；以单下划线开头的属性"\_xxx"虽然也可以被外部访问，但是，按照习惯，他们不应该被外部访问
2. 创建类属性：使例属性每个实例各自拥有，互相独立，而类属性有且只有一份。因为类属性是直接绑定在类上的，所以，访问类属性不需要创建实例，就可以直接访问。对一个实例调用类的属性也是可以访问的，所有实例都可以访问到它所属的类的属性。由于Python是动态语言，类属性也是可以动态添加和修改的

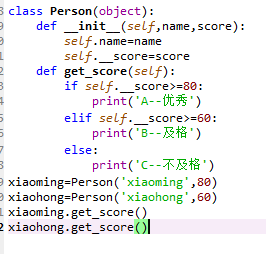
示例：



1. 类属性名与实例属性名冲突：当实例属性和类属性重名时，实例属性优先级高，它将屏蔽掉对类属性的访问。
2. 定义实例方法：实例的方法就是在类中定义的函数，它的第一个参数永远是 self，指向调用该方法的实例本身，其他参数和一个普通函数是完全一样的

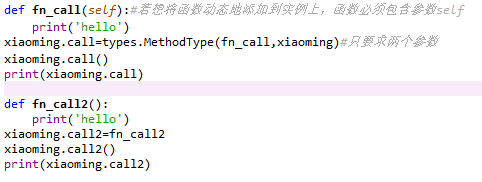
调用实例方法必须在实例上调用,在实例方法内部，可以访问所有实例属性，这样，如果外部需要访问私有属性，可以通过方法调用获得，这种数据封装的形式除了能保护内部数据一致性外，还可以简化外部调用的难度

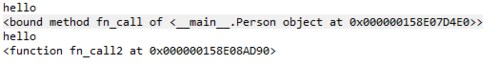
示例：



1. python中方法也是属性：在 class 中定义的实例方法其实也是属性，它实际上是一个函数对象。因为方法也是一个属性，所以，它也可以动态地添加到实例上，只是需要用 types.MethodType() 把一个函数变为一个方法

示例：





1. 定义类方法：类中的方法分为实例方法和类方法：在class中定义的全部是实例方法，实例方法第一个参数 self 是实例本身。要在class中定义类方法，需要通过标记一个 @classmethod，将方法将绑定到 类上，而非类的实例。类方法的第一个参数将传入类本身，通常将参数名命名为 cls
2. 