# With语法

with 工作原理

（１）紧跟with后面的语句被求值后，返回对象的“ \_\_enter\_\_()”方法被调用，这个方法的返回值将被赋值给as后面的变量；

（２）当with后面的代码块全部被执行完之后，将调用前面返回对象的“ \_\_exit\_\_()”方法。

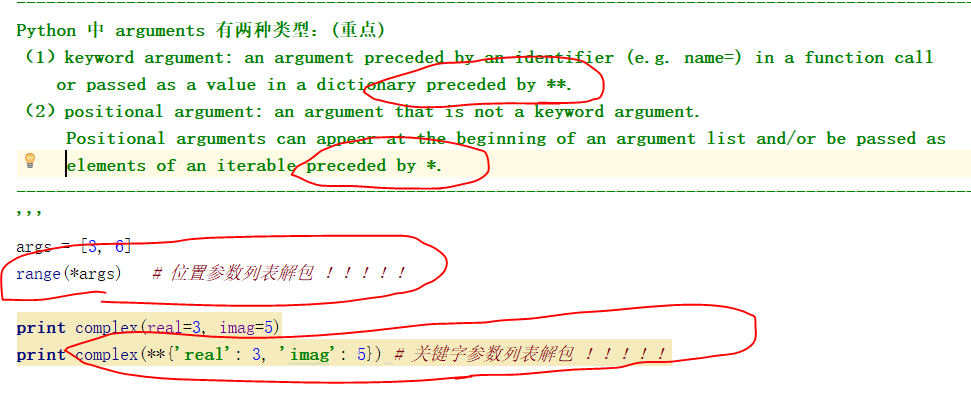
with 语句适用于对资源进行访问的场合，确保不管使用过程中是否发生异常都会执行必要的“清理”操作，释放资源，比如文件使用后自动关闭／线程中锁的自动获取和释放等。

# List 中 extend 和 append 区别

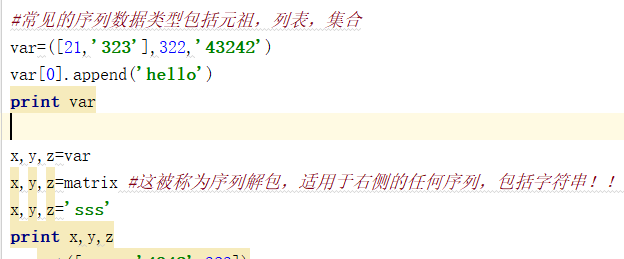
 Lists 的两个方法 extend 和 append 看起来类似，但实际上完全不同。

* extend 接受一个参数，这个参数总是一个 list，并且把这个 list 中的每个元素添加到原 list 中。
* append 接受一个参数，这个参数可以是任何数据类型，并且简单地追加到 list 的尾部。

# 3. 函数的默认参数以及参数类型

**def是一条可执行语句，Python 解释器执行 def 语句时，就会在内存中就创建了一个函数对象（此时，函数里面的代码逻辑并不会执行，因为还没调用嘛），在全局命名空间，有一个函数名（变量叫 func）会指向该函数对象，记住，至始至终，不管该函数调用多少次，函数对象只有一个，就是function object，不会因为调用多次而出现多个函数对象。 函数对象生成之后，它的属性：名字和默认参数列表都将初始化完成。因此，不能用可变对象作为默认参数的****值！！！**

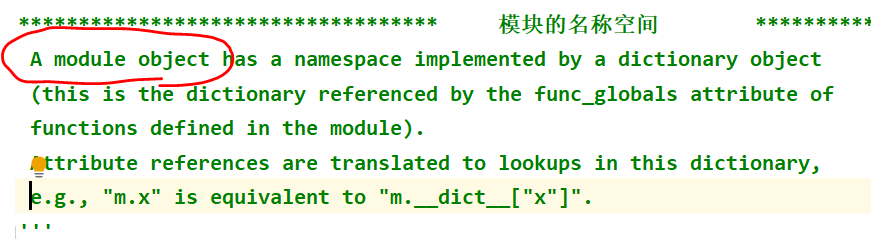
# 4. 专有名词记录

* 序列解包
* 列表推导
* 字典推导
* 位置参数列表解包
* 关键字参数列表解包

# 5.条件判断

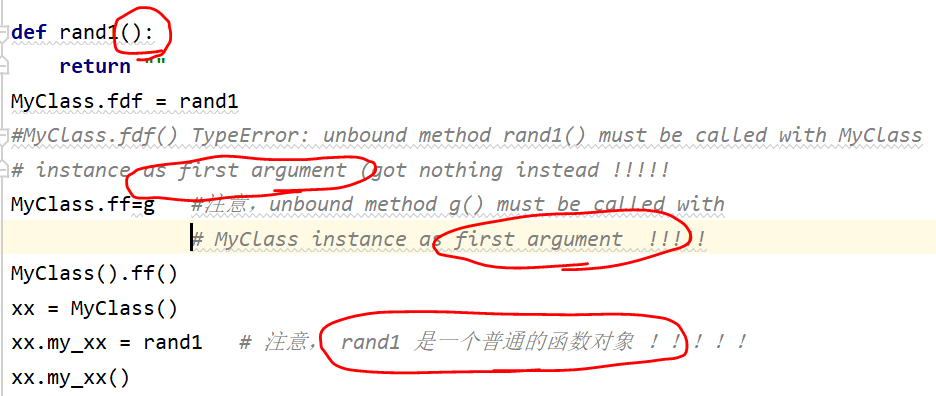
在Python中如果condition为 ''，()，[]，{}，None，set()，那么该条件为Flase，否则为True。

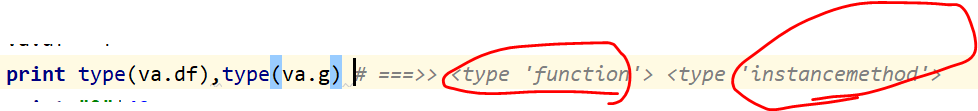
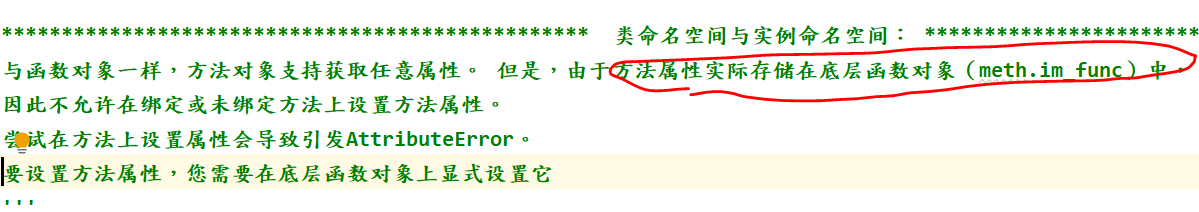
# 6. import 和 from … import 的区别



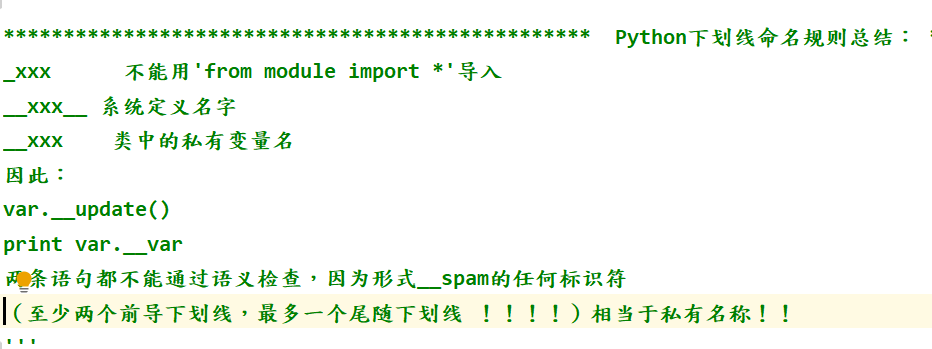
# 7. Python中名称空间

# 8. Python中的类对象和实例对象

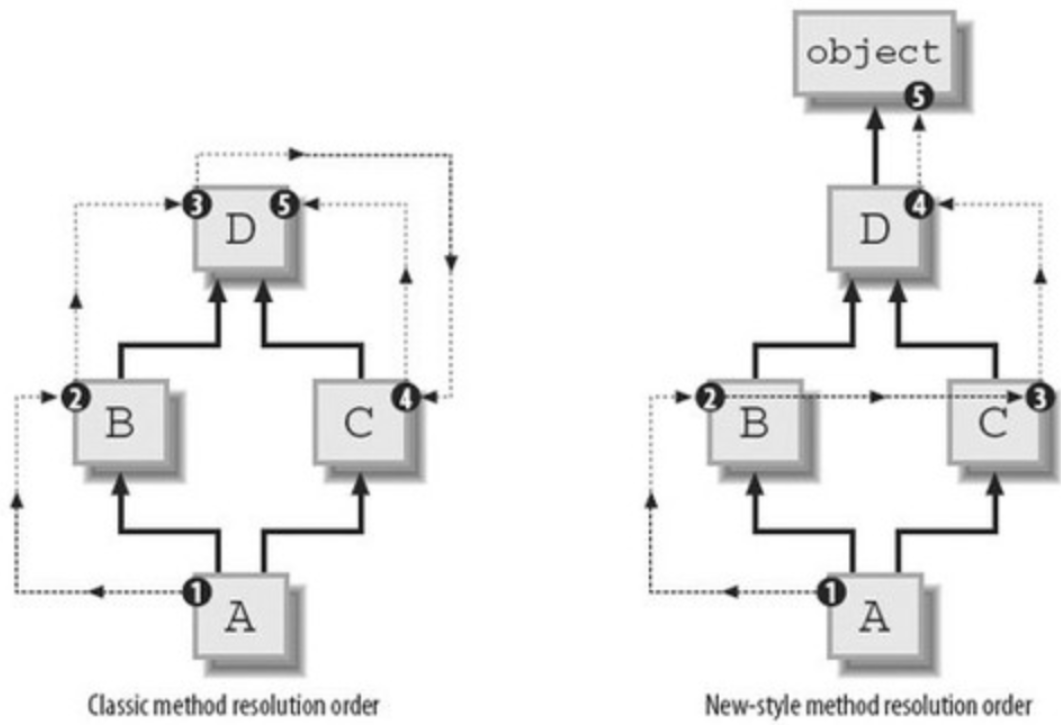
**注意，类对象的属性不能绑定 函数对象，只能绑定方法，而 实例对象的属性只能绑定普通的函数对象 ！！！！！！！！！！！！**

**注意，方法和函数是两个不同的****类型：**

# 9. Python下划线命名规则



# 10.Python 类继承的问题

MRO即方法解析顺序(method resolution order)，用于判断子类调用的属性来自于哪个父类。在Python2.3之前，MRO是基于**深度优先算法**的，自2.3开始使用**广度优先搜索**，定义类时需要继承object，这样的类称为新式类，否则为旧式类

# 11.Python构造器和yield语句