Python语法知识小结

引用、浅拷贝、深拷贝辨析：

A=[1,2,3,[‘a’,’b’]]

B=A

C=copy.copy(A)

D=copy.deepcopy(A)

引用：A变，B也变；B变，A也变；

浅拷贝：只拷贝父对象，不会拷贝父对象的子对象；也就是A的子对象变，C也变；A的父对象变，C不变；

也就是说，A的父对象增加子对象，那么C不变；A的父对象的子对象增加子对象就要变了；

深拷贝：相互独立，互不影响；

函数参数：在Python中定义函数，可以用必选参数、默认参数、可变参数、关键字参数和命名关键字参数，这5种参数都可以组合使用。但是请注意，参数定义的顺序必须是：必选参数、默认参数、可变参数、命名关键字参数和关键字参数。

尾递归：解决递归调用栈溢出的方法是通过**尾递归**优化，事实上尾递归和循环的效果是一样的，所以，把循环看成是一种特殊的尾递归函数也是可以的。

尾递归是指，在函数返回的时候，调用自身本身，并且，return语句不能包含表达式。这样，编译器或者解释器就可以把尾递归做优化，使递归本身无论调用多少次，都只占用一个栈帧，不会出现栈溢出的情况。

列表生成式：指直接在中括号中利用循环，生成一个列表；

生成器：利用yield 关键字创建生成器

可迭代的（iterable）：凡是可以用于for循环的对象都是可迭代的对象；可迭代的对象包含\_\_iter\_\_()方法；

迭代器（iterator）：凡是可作用于next()的函数的对象都是迭代器类型；迭代器类型中包含\_\_iter\_\_和\_\_next\_\_()方法；

\_\_iter\_\_()方法：必须要返回一个迭代器对象

高级函数：

map(函数，可迭代的对象)：迭代调用函数作用于可迭代对象的每个元素，并返回一个iterator（python3中返回iterator，python2中返回列表）；

reduce(函数，可迭代对象)：将函数递归调用；reduce(f, [x1, x2, x3, x4]) = f(f(f(x1, x2), x3), x4)，返回一个累积计算结果；

filter(函数,可迭代对象)：filter()把传入的函数依次作用于每个元素，然后根据返回值是True还是False决定保留还是丢弃该元素。filter()函数返回的是一个Iterator（python3中返回iterator，python2中返回列表）；

闭包：我们**在函数lazy\_sum中又定义了函数sum**，并且，内部函数sum可以**引用**外部函数lazy\_sum的参数和局部变量，当lazy\_sum返回函数sum时，相关参数和变量都保存在返回的函数中，这种称为“闭包（Closure）”的程序结构拥有极大的威力。

装饰器：在不修改原函数，源代码基础上，在代码运行器件动态增加功能的方式称为装饰器；本质是一个返回函数的高阶函数；

模块functools：包含许多特别的函数

面向对象编程：继承、重写、多态、