## 1、装饰器

装饰器（简洁说：封装一个函数，用这样或者那样的方式来修改他的行为。在不改变原函数的内部逻辑和函数调用代码的情况下进行功能扩展。由decorator和嵌套的wrapper组成一个共生体，这个共生体我们叫闭包，闭包满足三个条件：1、函数嵌套函数2、函数返回函数3、函数之间有引用）

通过decorator来传入实际要运行的函数，通过wrapper函数来传入函数参数并调用函数，同时在wrapper函数实现需要的逻辑。

@ a\_new\_decorator

#== a\_function\_requiring\_decoration = a\_new\_decorator(a\_function\_requiring\_decoration)

def a\_function\_requiring\_decoration():

print(a\_function\_requiring\_decoration.\_\_name\_\_) #\_\_name\_\_表示当前模块执行过程的名称

被装饰后的函数其实是另外一个函数了，\_\_name\_\_(变成wrapper)、\_\_doc\_\_(变成decorator的注释)的属性发生变化。python的functools包提供了叫wraps的decorator能保留原函数的名称和函数属性，通过复制操作。

python 返回函数名可以被传递，赋值给别的变量。加括号则会执行