修饰器

1. 是一种著名的设计模式，常常用于有切面需求的场景（AOP 面向切面编程）
2. 可以抽出大量的函数中的功能，和雷同的代码，并继续重用
3. 修饰器的作用就是为已存在的对象，添加额外的功能

关键字：@
1. 修饰器修饰的是函或者是方法，不能修饰一个类
2. 修饰器必须出现在函数或方法的前一行
3. 修饰器本身就是一个函数，将被修饰的函数名作为参数，传递给修饰器，执行修饰器中，返回传递进来的函数对象

```
# 基本形式
    def A(a):  #fun
        print('A')
        return a

    @A
    def fun():
        print('fun')

    fun()
# 利用函数嵌套
    def A(a):  #fun
            def c():
                print('c')
            c()
            return a

        @A
        def fun():
            print('fun')

    fun()
#  利用闭包
    def A(a):  #a=fun
        def c():
            print('c')
            return a()  #fun

        return c

    @A
    def fun():
        print('fun')

    fun()
# 如果原函数有参数，使用闭包的参数要和原函数的参数一致（收参）
```

```
        def A(a):  #a=fun
            def c(n,n1):
                print(n,n1)
                return a(n,n1)
            return c

        @A
        def fun(name,name1):
            print('fun')

        fun('辉哥','逗哥')
```

 # 原函数有返回值

同上，打印一下即可

 # 有参数的修饰器，参数的原函数，使用闭包接收原函数的函数名

因为实在闭包中收到的函数名，所以会直接调用原函数

```
        def A(a):  #a=fun
            print(a)
            def c(fun_name):
                print(fun_name)
                return fun_name     #会无差别调用，第一次收到原函数对象
            return c

        @A(30)
        def fun():
            print('fun')

        fun()
```