1.continue关键字：跳出当前循环，执行下一次的循环

break关键字：跳出全部循环

2.在要测试文件的路径下打开python解释器，是为了保证python在寻找文件时先去解释器的主目录里去找文件时能找到。这里的知识点是python解释器在寻找文件时的四个目录。

3.open()函数返回一个文件对象，所以使用的时候要将其赋给一个变量。

4.用了try except语句

5.关于with as语句

6.关于类方法，实例方法，静态方法。类和实例。

类方法可以通过类方法装饰器定义。调用的时候可以是类.方法，也可以是实例.方法，也可以是直接调用。

静态方法和类方法基本相同。一般通过静态方法装饰器进行定义。除了可以用类和实例来调用，也可以直接拿来用。

而实例方法只能通过实例.方法的方式进行调用，即使是用类本身直接调用也是不可以的。构造函数属于实例方法。但是子类不会自动调用父类的构造函数，可以通过在子类的构造函数里使用super（）实现此机制。

7.对于import的路径，pycharm和appliance上是不一样的。Pycharm的路径除了python内置库路径，自定义的模块好像只能用同一个文件夹下的？Appliance上的路径在SPEC文件下找。

8.以”\_\_”开头的变量是类的私有变量，只能内部使用.

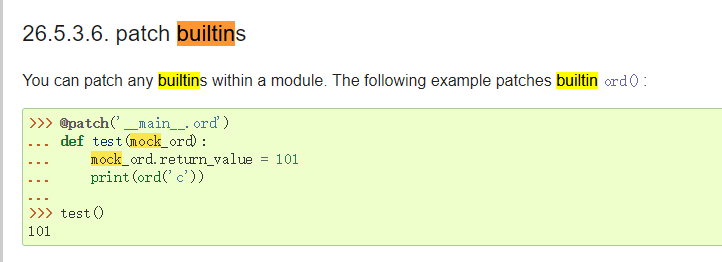
9.关于写UT

1. 数据流重定向(stringIO模块)
2. 生成XML文件报告(xmlrunner模块)
3. Mock模块：assert的使用方法
4. setup()和teardown()函数的必要性作用
5. assert\_called\_with()只能对最近（后）一次调用的参数进行判断，而assert\_any\_call()可以对整个过程中调用的参数进行判断。assert\_called\_once\_with()是对最近一次的参数的参数判断其在整个过程中是否只被调用过一次。
6. @patch.object('\_\_builtin\_\_', "open", mock\_open(read\_data = 'texx'))

@patch('\_\_builtin\_\_.open', mock\_open(read\_data='xxx'))

为什么前者的时候报错\_\_builtin\_\_ not define.

解：关于这个问题查阅了官方文档，只有下图的一个说明，并没有说为什么。就这么记着吧。



10.官方文档中说os模块是This module provides a portable way of using operating system dependent functionality.

而sys模块是This module provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter. It is always available.

所以说os是跟操作系统有关，而sys是跟编译器有关。

\_\_builtin\_\_.py模块是This module is not normally accessed explicitly by most applications，也就是说不通的应用其访问路径也是不同的。

在pycharm里查看os.py sys.py \_\_builtin\_\_.py 三者的存放路径也能看出：os.py存放在python的库里，而后两者都存放于pycharm的库里。

11.关于单下划线和双下划线的变量（函数）名。

两者都是被定义为‘私有’的，但是单下划线开头的可以被类对象本身和其子类对象访问，双下划线只能被类对象本身调用。

12.