随着软件功能的增多，模块繁多，那么测试用例也会越来越多，为了针对特定的执行，这样可以解决执行时间，快速得到测试报告结果。

1. 运行指定页面模块：

pytest.mian(“-v -s xxx.py”)

1. 运行文件夹下批量文件

pytest.main(“-v -s ./)

Pytest.main(“-v -s xxx/”)

1. 运行指定class下的用例

pytest.main("-v -s XX.py::XXClass")

4.运行指定fun下的用例

pytest.main("-v -s XX.py::XXClass::testxx\_spec")

5.运行特定指定的用例

Pytest.mark

（1）根目录下新建的pytest.ini，注册自定义标签

（2）在用例类或函数等前打标签 @pytest.mark.webtest

（3）运行测试用例

例：pytest -m "webtest"

6.conftest.py+fixture相当于unittest 的setup teardown,但功能更强大

(参数scope：session，module，class，function）

所有同目录测试文件运行前都会执行conftest.py文件，

适用场景：

（1）、每个接口需共用到的token

（2）、每个接口需共用到的测试用例数据

（3）、每个接口需共用到的配置信息

6-2:pytest.mark.usefixtures('name')

1.单个fixture：在被测试函数之前执行@pytest.mark.usefixtures('one')

2.多个fixture:按传值先后顺序执行@pytest.mark.usefixtures('two',’one’)

3.叠加fixture: 自下至上执行，离测试函数越近的先被执行

@pytest.mark.usefixtures('one')

@pytest.mark.usefixtures('two')