一、TestRunner 测试运行器 执行用例，输出测试结果

1、unittest提供生产测试报告的模块 TextTestRunner 生成文本格式测试报告

2、常见的第三方库结合unittest生产html格式测试报告

HtmlTestRunner

官网下载HtmlTestRunner.py只能支持python2版本，支持Python3,需要做修改

路径：python安装路径/Lib

HtmlTestRunner 的应用

BeatifulReport 的应用

企业测试报告的优化及定制 优化测试报告模板 通过js+html/html5

pytest+allure 生成更加美观的测试报告+优化定制（装饰器）

```
import unittest
from class08.testcase_01 import TestCase01
import HTMLTestRunner
import BeautifulReport
from BeautifulReport import BeautifulReport

# 加载用例
testcases=unittest.TestLoader().loadTestsFromTestCase(TestCase01)
# 执行用例，生产对应的测试报告TextTestRunner
 # with open('./reports/report.txt',"w+") as txtfile:
  #文件写入测试结果
  # unittest.TextTestRunner(stream=txtfile,verbosity=2).run(testcases)

#HTMLTestRunner生成的测试报告
 # with open('./reports/report.html', "wb+") as htmlfile:
 # runner=HTMLTestRunner.HTMLTestRunner(stream=htmlfile,title="码尚教育测试报告",description="码尚教育测试用例详情")
 # runner.run(testcases)
 #BeautifulReport 生成的测试报告
 with open('./reports/report.html', "wb+") as htmlfile:
  BeautifulReport(testcases).report(description="码尚教育测试报告",filename="report_bf",report_dir="reports")
```

二、装饰器 @ unittest.skip 强制跳过&条件跳过

```

class TestCase01(unittest.TestCase):

 @unittest.skip("此用例暂时不启用")
 def test_login(self):
 """
```

```
    登录
    :return:
    """
    print("用例01")
    @unittest.skipIf(3>2,"条件为真，则跳过执行")
    def test_selectgoods(self):
    """
    检索商品
    :return:
    """
    print("用例02")

    @unittest.skipUnless(2>3,"条件：2>3不成立，则跳过执行")
    def test_gointocart(self):
    """
    加入购物车
    :return:
    """
    print("用例03")
```

```
"""
@unittest.skip 强制跳过执行
@unittest.skipIf 符合条件，则跳过执行
@unittest.skipUnless 条件不成立，则跳过执行
"""
import unittest

@unittest.skipUnless(False,"整个模块下的用例强制跳过执行")
class TestSkipModule(unittest.TestCase):

    def test_login(self):
    """
    登录
    :return:
    """
    print("用例01")

    def test_selectgoods(self):
    """
    检索商品
```

```
    :return:
    """
    print("用例02")

    def test_gointocart(self):
    """
    加入购物车
    :return:
    """
    print("用例03")

if __name__ == '__main__':
    unittest.main()
```

三、unittest的常用断言方法

```
常用断言方法
1、assertIn(字符1，字符2) 字符1是否包含在字符2
2、self.assertNotIn(字符1，字符2) 字符1不包含包含在字符2
self.assertEqual(参数1,参数2,"断言失败的描述") 参数1等于参数2
self.assertNotEqual(参数1,参数2,"断言失败的描述")参数不等于参数2
self.assertTrue(True)
self.assertFalse(False)
```

```
from selenium import webdriver

class TestCase01(unittest.TestCase):

    def setUp(self) -> None:
    # 打开chrome浏览器
    self.driver=webdriver.Chrome()

    def test_selectgoods(self):
    """
    检索商品
    :return:
    """
    self.driver.get("http://47.107.116.139/shopnc/shop/")
    # 定位搜索输入
    el_select=self.driver.find_element(By.ID,"keyword")
    el_select.send_keys("手机")
```

```
18  el_button=self.driver.find_element(By.ID,"button")
19  el_button.click()
20  time.sleep(2)
21  #断言：验证测试结果与预期结果是否一致
22  #获取商品列表的标题
23  content=self.driver.find_element(By.XPATH,"//div[@class='goods-name']/
a").text
24  print(content)
25  #判断content是否包含手机字符？
26  #常用断言方法
27  """
28  常用断言方法
29  1、assertIn(字符1，字符2) 字符1是否包含在字符2
30  2、self.assertNotIn(字符1，字符2) 字符1不包含包含在字符2
31  self.assertEqual(参数1,参数2,"断言失败的描述") 参数1等于参数2
32  self.assertNotEqual(参数1,参数2,"断言失败的描述")参数不等于参数2
33  self.assertTrue(True)
34  self.assertFalse(False)
35  """
36  # 标题是否包含手机
37  self.assertIn("手机2222",content,"断言失败的描述")
38  #列表下有多少个商品 返回元素列表得到个数
39  # count=els.count
40  # self.assertEqual(count,1)
```

会员账号： admin msjy123

会员账号2： xingyao mashang

后台账号： admin msjy123

作业：编辑 完成支付功能用例