UI 自动化
首先我会倒入一些模块:
1.from selenium import webdriver--将它实例化--driver=webdriver.Chrome()
2.from selenium.webdriver.support.ui import Select--#导入查询下拉选项的类
select_1=driver.find_element_by_css_selector()#定位下拉框位置
select1=Select(select_1)#实例化
3.import time
python+selenium+unittest 的基础上封装的一个自动化框架，这个框架有几个包通过 po 模式来设计的
通过 po 模式把数据、业务和逻辑分开，对代码进行解耦，将一些元素定位方法，系统所需数据如登录等，输出报告的
方法，数据库方法，省成报的方法，存放用例和执行用例等等方法分开管理，这样我们可以很好的维护和管理用例和这些方法，
降低代码的耦合度，降低成本，提升效率
第一步先登录--driver.get(url)-driver.maximize_window()#最大化浏览器-driver implicitly_wait(10)-隐式等待-然后
进入 iframe 框--driver.switch_to_frame
(框的位置--定位包括--id、name、xpath、父级路径、css、link 等)--根据这些定位方法选其一定位到输入用户名框如:
driver.find_element_by_id('id 属性对应的值').send_keys('用户名')--再定位到输入密码框-同上方法--然后定位到
登录按钮 driver.find_element_by_id('id 属性对应的值').click-点击登录--介于浏览器和服务器的响应通常我都会设置
大概 2 秒左右的休眠时间如：time.sleep(2)--接着跳出 iframe 框--driver.switch_to.default_content()。完成后即完成登录
进入到后台管理系统--进行一个 title 断言--设置一个变量名如 title1=driver.title--assert title1=='wms 仓储管理系统'
第二步出库单是由上游推送过来的，点击新增出库按钮--（driver.find_element_by_id('id 属性对应的值').click）
--将出库单导入系统--出现弹窗-是否手动处理订单（driver.find_element_by_id('alert').click()--time.sleep(2)--driver.switch_to.alert.dismiss()）订单处理分为人工处理和自动处理--选择否即是 dismiss-按照系统自动处理订单-选择
确定即是 accept--进入人工处理订单-包括一些信息的录入如:
客户的名称、商品的名称、仓库的名称、数量、时间、下家的物流名称等信息-这些消息的录入需要先定位到指定位置
输入框的定位好位置后.send_keys('信息')，按钮或者其他需要点击的--定位指定位置.click（）即可，还有一些下拉框选项
如选择 A/B/C 仓--select_1=driver.find_element_by_css_selector()#定位下拉框位置--select1=Select(select_1)#实例化
select_elm1.select_by_index(2)-还可以通过 value/lable(文本选择)等。系统会判断库存是否满足，如果不满足，
点击缺货按钮-driver.find_element_by_id('alert').click()--time.sleep(2)将进行补货流程。
3.库存满足的话将进行组建批次的操作，订单信息也从待处理转变成待组批次（断

言）--点击组建批次按钮系统进行

自动组建批次，此时可以进行一个包括清单和面单打印的前置打印操作，订单信息状态由待组批转变成待下发（断言）
。4.点击进入批次下发，批次的下发受到播种口的限制，先下发先占用播种口，批次下发完成后，订单信息由待下发转变
成拣货（断言）。5.点击拣货按钮--拣货分为 PDA 逐件拣货和 PSB 机器人拣货--如果遇到货物不足点击缺货按钮--有件货任务
的订单号会显示红色底色，完成后红色底色消失，完成拣货操作后，订单信息由待拣货转变成待播种（断言）。点击播种口
换箱按钮，将货物录入播种口，满了就换箱子，直至全部完成，并生成一个转运单据，单据类型主要分为以下三种（下拉框
-2c 单件单、2c 多件单、2b 单）订单信息由待播种转变成待转运（断言）。2c 单件单的处理先点击单件转运按钮，订单信息由
待转运转变成待复核后进行复核包装（进行一个清单和面单的后置打印），库存将会进行相应的扣减（断言），订单信息由
待复核转变成待发货。点击包裹的分拣按钮，进行称重，选择物流商家后点击发货，后将信息回传给上游。2c 多件单的处理
先点击集包转运（一个批次占用一个集包口），完成后，点击分拣转运 ，完成后进行二次分拣，此时需要二分人员进行一个
包括清单和面单的中置打印操作，二次分拣完成后进入复核包装，库存将会进行相应的扣减（断言），复核包装完成后进行
清单和面单那的后置打印操作订单信息由待复核转变成待发货。点击包裹的分拣按钮，进行称重，选择物流商家后点击发货，
后将信息回传给上游。2b 单的话点击集包转运，完成后点击调拨转运，完成后进入复核包装，库存将会进行相应的扣减（断言）
复核包装完成后进行清单和面单那的后置打印操作订单信息由待复核转变成待发货，选择物流商家后点击发货，后将信息回传
给上游。最后返回出库列表断言出库单号、仓库名、商家信息、物流名称、时间等信息是否一致。

```
text_value_2=driver.find_element_by_xpath().text
assert text_value_2 =='' #断言判断完成后点击出库-订单信息由待出库变成出库，点击关闭退出系统和浏览器
--driver.quit()
```

我使用 PO 设计模式、和单例模式 结合 Python 中的 selenium 及 unittest 框架进行 UI 自动化开发。
PO 设计模式将页面元素、定位方法、测试用例封装起来，实现了测试用例的分层管理，体现了面向对象的基本思想，实现了松耦合。PO 框架由存放页面元素、文件读取等的配置包、封装基本 API 方法、
元素定位方法的公共包以及封装测试用例的用例包，和存放报告、数据等的数据包、报告包。
单例模式---把登录时打开的浏览器对象，设置为全局唯一的对象，供后面的方法调用（定位元素，输入值，切换句柄）
保证我们登录的唯一性

下面我大概说一下如何实现：首先登录系统，
调用 find_element_by_id 方法进行用户名和密码输入框定位，
然后使用 send_keys()方法输入用户名和密码，定位并 click()登录按钮后转入登录后界面，
assert_equal 进行页面标题断言，判断是否登录成功。进行线索管理时，
click 点击线索管理按钮，新开页面跳转至线索管理首页，switch_window_handle 切换句柄后
，使用 id 进行元素定位并点击相应功能，然后跳转入对应界面，比如线索添加，
id 定位元素并 send_keys 进行输入，最后 click 提交即可。
提交后执行 quit()方法关闭线索管理界面，回到销售专员端首页，
id 定位并 click 点击商机管理按钮，新开页面跳转入商机管理界面，
切换句柄后即可对商机管理界面进行操作。其他功能大概也是这个样子。