1. Зарегистрируйтесь на сайте service-online.su, ознакомьтесь со страницей генерации ценников https://service-online.su/forms/cenniki/

2. Создайте новый Python unit test. Напишите методы setUp и tearDown для запуска браузера, входа на сайт и открытия страницы генерации ценников при запуске тестов и закрытия браузера по окончании теста. В качетве первого теста напишите проверку появления в меню ссылки «Личный кабинет» после авторизации.

import unittest

from selenium import webdriver

import time

class PriceTestCase(unittest.TestCase):

# подготовка к каждому тесту

def setUp(self):

# запуск Firefox при начале каждого теста

self.driver = webdriver.Firefox()

# открытие страницы при начале каждого теста

self.page = self.driver.get(

'https://service-online.su/forms/cenniki/'

)

# открытие окна авторизации

elem = self.driver.find\_element\_by\_id("enter")

elem.click()

# ввод логина и пароля

elem = self.driver.find\_element\_by\_name("login")

elem.send\_keys("**Ваш логин**")

elem = self.driver.find\_element\_by\_name("password")

elem.send\_keys("**Ваш пароль**")

# нажатие кнопки "Войти"

elem = self.driver.find\_element\_by\_name("test\_enter")

elem.click()

# 5 сек ожидания открытия окна

time.sleep(5)

# закрытие сообщения об успешном входе

elem = self.driver.find\_element\_by\_id("delete")

elem.click()

# окончание каждого теста

def tearDown(self):

# выход с сайта во избежание блокировки

# системой защиты от подбора пароля

ul = self.driver.find\_element\_by\_id("menu")

lis = ul.find\_elements\_by\_tag\_name("li")

li = lis[-1]

li.click()

# закрытие браузера при окончании каждого теста

self.driver.close()

# тест на наличие ссылки "Личный кабинет" в меню

def testAuthorization(self):

elem = self.driver.find\_element\_by\_id("menu")

self.assertIn("Личный\nкабинет", elem.text)

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

3. Запустите тест, убедитесь в его успешном выполнении.

4. Поскольку ценники будут сохраняться в формате PDF, требуется **установить пакет textract в PyCharm, скачать утилиты командной строки XpdfReader для работы с PDF, поместить все \*.exe файлы из архива в папку с интерпретатором Python.exe.**

Ссылка на страницу скачивания консольных утилит (command line tools):

https://www.xpdfreader.com/download.html

5. Подключите необходимые пакеты для работы с файлами, настройте определенную папку в качестве места сохранения файлов с помощью профиля браузера в методе setUp, очищайте эту папку при запуске тестов:

# для работы с сохраненными файлами

import os

# в именах сохраненных файлов ставится дата

import datetime

# класс для обработки PDF

import textract

# для очистки папки с сохраненными файлами после теста

import shutil

class PriceTestCase(unittest.TestCase):

# подготовка к каждому тесту

def setUp(self):

# папка для сохранения файлов

self.savePath = "c:\\testDownloads\\"

# очистка папки с сохраненными файлами

if os.path.isdir(self.savePath):

shutil.rmtree(self.savePath)

# создание папки для сохранения файлов

os.mkdir(self.savePath)

# профиль для сохранения файлов

profile = webdriver.FirefoxProfile()

profile.set\_preference(

"browser.download.dir",

self.savePath

)

profile.set\_preference("browser.download.folderList", 2)

profile.set\_preference(

"browser.helperApps.neverAsk.saveToDisk",

"application/pdf;application/x-pdf"

)

profile.set\_preference(

"browser.helperApps.alwaysAsk.force",

False

)

# запуск Firefox при начале каждого теста

self.driver = webdriver.Firefox(firefox\_profile=profile)

# открытие страницы при начале каждого теста

self.page = self.driver.get(

"https://service-online.su/forms/cenniki/"

)

# открытие окна авторизации

elem = self.driver.find\_element\_by\_id("enter")

elem.click()

# 5 сек ожидания открытия окна

time.sleep(5)

# ввод логина и пароля

elem = self.driver.find\_element\_by\_name("login")

elem.send\_keys("**Ваш логин**")

elem = self.driver.find\_element\_by\_name("password")

elem.send\_keys("**Ваш пароль**")

# нажатие кнопки "Войти"

elem = self.driver.find\_element\_by\_name("test\_enter")

elem.click()

# закрытие сообщения об успешном входе

elem = self.driver.find\_element\_by\_id("delete")

elem.click()

6. Добавьте тест для проверки сохранения одного ценника с полями, заполненными на английском языке:

# тест сохранения одного ценника на английском

def testOnePrice(self):

elem = self.driver.find\_element\_by\_id("comp\_name")

elem.send\_keys("Eurotorg")

# английский формат даты год/месяц/день

elem = self.driver.find\_element\_by\_id("date")

elem.clear()

elem.send\_keys("2020/01/13")

# выбор белорусского рубля в выпадающем списке валют

elem = self.driver.find\_element\_by\_name("valyuta")

elem.click()

options = elem.find\_elements\_by\_tag\_name("option")

for option in options:

if option.text == "Белорусский рубль, 974":

option.click()

break

elem = self.driver.find\_element\_by\_id("tovar\_ed\_default")

elem.send\_keys("kg")

elem = self.driver.find\_element\_by\_id("tovar\_country\_default")

elem.send\_keys("RB")

# таблица с товарами

elem = self.driver.find\_element\_by\_id("tab1")

tbody = elem.find\_element\_by\_tag\_name("tbody")

tr = tbody.find\_element\_by\_tag\_name("tr")

# название товара

td = tr.find\_element\_by\_class\_name("tovar\_name")

field = td.find\_element\_by\_tag\_name("textarea")

field.send\_keys("Candy Southern Night")

# цена товара - английский формат с точкой

td = tr.find\_element\_by\_class\_name("tovar\_cena")

field = td.find\_element\_by\_tag\_name("input")

field.send\_keys("10.55")

# жмем ссылку "Скачать"

elem = self.driver.find\_element\_by\_id("download")

elem.click()

# 10 сек ожидания

# на случай, если Firefox спросит, сохранять файл

time.sleep(10)

# проверяем наличие сохраненного файла по названию

today = datetime.date.today()

fullpath = (self.savePath +

"cenniki-new-" +

today.strftime("%Y-%m-%d") +

".pdf"

)

self.assertEqual(

os.path.isfile(fullpath),

True

)

# получаем текст из сохраненного файла

page = textract.process(fullpath)

# проверяем наличие введенных значений в тексте файла

#print(page)

self.assertIn(b"Eurotorg", page)

self.assertIn(b"2020/01/13", page)

self.assertIn(b"kg", page)

self.assertIn(b"RB", page)

self.assertIn(b"Candy Southern Night", page)

7. Запустите тест, убедитесь в обнаружении несоответствия формата даты.

8. Для работы с русскоязычными PDF-файлами потребуется другой модуль – **pdfminer – установите его**.

9. Добавьте аналогичный тест для данных на русском языке:

# классы для обработки русского текста в PDF

import io

from pdfminer.converter import TextConverter

from pdfminer.pdfinterp import PDFPageInterpreter

from pdfminer.pdfinterp import PDFResourceManager

from pdfminer.pdfpage import PDFPage

# ...

# тест сохранения одного ценника на русском

def testOnePriceRu(self):

elem = self.driver.find\_element\_by\_id("comp\_name")

elem.send\_keys("ООО Евроторг")

elem = self.driver.find\_element\_by\_id("date")

elem.send\_keys("13.01.2020")

# выбор белорусского рубля в выпадающем списке валют

elem = self.driver.find\_element\_by\_name("valyuta")

elem.click()

options = elem.find\_elements\_by\_tag\_name("option")

for option in options:

if option.text == "Белорусский рубль, 974":

option.click()

break

elem = self.driver.find\_element\_by\_id("tovar\_ed\_default")

elem.send\_keys("кг")

elem = self.driver.find\_element\_by\_id("tovar\_country\_default")

elem.send\_keys("РБ")

# таблица с товарами

elem = self.driver.find\_element\_by\_id("tab1")

tbody = elem.find\_element\_by\_tag\_name("tbody")

tr = tbody.find\_element\_by\_tag\_name("tr")

# название товара

td = tr.find\_element\_by\_class\_name("tovar\_name")

field = td.find\_element\_by\_tag\_name("textarea")

field.send\_keys("Конфеты Южная ночь")

# цена товара

td = tr.find\_element\_by\_class\_name("tovar\_cena")

field = td.find\_element\_by\_tag\_name("input")

field.send\_keys("10,55")

# жмем ссылку "Скачать"

elem = self.driver.find\_element\_by\_id("download")

elem.click()

# 10 сек ожидания

# на случай, если Firefox спросит, сохранять файл

time.sleep(10)

# проверяем наличие сохраненного файла по названию

today = datetime.date.today()

fullpath = (self.savePath +

"cenniki-new-" +

today.strftime("%Y-%m-%d") +

".pdf")

self.assertEqual(

os.path.isfile(fullpath),

True

)

# получаем текст из сохраненного файла

# открываем файл

fh = open(fullpath, 'rb')

# открываем первую страницу

page\_obj = PDFPage.get\_pages(fh,

caching=True,

check\_extractable=True

).\_\_next\_\_()

resource\_manager = PDFResourceManager()

# создаем объект для вывода текста

fake\_file\_handle = io.StringIO()

# создаем конвертер для извлечения текста из PDF

converter = TextConverter(resource\_manager, fake\_file\_handle)

# создаем интерпретатор страницы

page\_interpreter = PDFPageInterpreter(resource\_manager, converter)

# извлекаем текст страницы

page\_interpreter.process\_page(page\_obj)

# забираем текст страницы в переменную page

page = fake\_file\_handle.getvalue()

# уничтожаем созданные объекты

converter.close()

fake\_file\_handle.close()

# проверяем наличие введенных значений в тексте файла

print(page)

self.assertIn("ООО Евроторг", page)

self.assertIn("13.01.2020", page)

self.assertIn("кг", page)

self.assertIn("РБ", page)

self.assertIn("Конфеты Южная ночь", page)

# цена

self.assertIn("10", page)

self.assertIn("55", page)

10. Запустите тест, убедитесь в его успешном выполнении.