به نام خدا

پروژه آزمون نرم افزار

پروژه شماره 15

نام و نام خانوادگی: امیر محمد مغربی شماره دانشجویی: 9812762510

نام و نام خانوادگی: سید امیرعلی موسوی شماره دانشجویی: 9822762244

در فایل test_15.py، تست های مربوط به پروژه شماره 15 انجام شد و در آن دو testcase طراحی گردید

یک تست درباره جابجایی هر کارت که در لیست ها موجود بودند و تست دیگر درباره جابجایی خود لیست ها هستند

در هرکدام از این تست ها با استفاده از ChainAction موجود در کتابخانه Selenium، سعی کردیم کار های مربوط به موس را handle کنیم از این بابت بایستی می گفتیم که click-and-hold را فرخوانی می کردیم و در قسمتی که مطلوبمان بود release میکردیم

بعد از رها شدن، location آن تغییر کرده که از طریق آن می توانستیم assertion های خود را انجام دهیم

در فایل test_calculated.py، تست های مربوط پروژه calculator انجام شد

در آن 4 تست کیس فراخوانی شد که برای تست جمع، تفریق به علاوه مقدار های متفاوت انجام شد

در هر تست کیس، 100 حالت بررسی می شود

و سپس مقدار expected را محاسبه و سپس و با مقداری که از ماشین حساب بدست آمده بود مقایسه می گردید

لینک گیتهاب: https://github.com/amm5949/SoftwareTesting

```python
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.firefox.options import Options
import time


class TestCase(unittest.TestCase):
    def setUp(self):
        options = Options()
        options.binary_location = r'C:\Program Files\Mozilla Firefox\firefox.exe'
        self.driver = webdriver.Firefox(
            executable_path=r'C:\webdrivers\geckodriver.exe', options=options)

        self.driver.maximize_window()

        self.driver.delete_all_cookies()

        self.driver.get("http://localhost:3000/")
        print("we are all set!")
        time.sleep(20)

    def testCard(self):
        src1 = '/html/body/div/main/div/div[1]'
        src2 = '/html/body/div/main/div/div[2]'

        card1 = self.driver.find_element(By.XPATH, src1)

        card2 = self.driver.find_element(By.XPATH, src2)

        location2 = card2.location
        # time.sleep(20)
        action = ActionChains(self.driver)
        action.click_and_hold(card1)
        action.release(card2)
        action.perform()

        time.sleep(10)

        location1_after_movement = card1.location
        self.assertEqual(location2, location1_after_movement)

    def testList(self):
        src1 = '/html/body/div/main/div/div[1]'
        src2 = '/html/body/div/main/div/div[2]'

        list1 = self.driver.find_element(By.XPATH, src1)

        list2 = self.driver.find_element(By.XPATH, src2)

        location2 = list2.location

        action = ActionChains(self.driver)
        action.click_and_hold(list1)
        action.release(list2)
        action.perform()

        time.sleep(10)

        location1_after_movement = list1.location
        self.assertEqual(location2, location1_after_movement)

    def tearDown(self):
        self.driver.close()


if __name__ == "__main__":
    unittest.main()

# driver = webdriver.Firefox()

# # driver.maximize_window()

# print(driver._mobile)

# driver.get("https://www.google.com")
```

```python
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.firefox.options import Options
import time


class TestCase(unittest.TestCase):
    def setUp(self):
        options = Options()
        options.binary_location = r'C:\Program Files\Mozilla Firefox\firefox.exe'
        self.driver = webdriver.Firefox(
            executable_path=r'C:\webdrivers\geckodriver.exe', options=options)

        self.driver.maximize_window()

        self.driver.delete_all_cookies()

        self.driver.get("http://localhost:3001/")
        print("we are all set!")
        time.sleep(20)
        zero = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[5]/div[1]/button')
        one = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[4]/div[1]/button')
        two = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[4]/div[2]/button')
        three = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[4]/div[3]/button')
        four = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[3]/div[1]/button')
        five = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[3]/div[2]/button')
        six = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[3]/div[3]/button')
        seven = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[2]/div[1]/button')
        eight = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[2]/div[2]/button')
        nine = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[2]/div[3]/button')

        self.result = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[1]/div')

        self.negative_sign = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[1]/div[2]/button')

        self.subtraction_sign = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[3]/div[4]/button')

        self.plus_sign = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[4]/div[4]/button')
        self.equal_sign = self.driver.find_element(
            By.XPATH, '/html/body/div/div/div[2]/div[5]/div[3]/button')
        self.testArr = [zero, one, two, three, four, five, six, seven, eight, nine]
        # print("Hi")

    def testAdd(self):
        for i in range(10):
            for j in range(10):
                number1 = i
                number2 = j
                expected_value = number1 + number2

                self.testArr[i].click()
                self.plus_sign.click()
                self.testArr[j].click()
                self.equal_sign.click()
                result_1 = int(self.result.text)
                self.assertEqual(result_1, expected_value)

    def test_negative_number_in_add(self):
        for i in range(10):
            for j in range(10):
                number1 = -(i)
                number2 = j
                expected_value = number1 + number2

                self.testArr[i].click()
                self.negative_sign.click()
                self.plus_sign.click()
                self.testArr[j].click()
                self.equal_sign.click()
                result_1 = int(self.result.text)
                self.assertEqual(result_1, expected_value)

    def test_subtraction(self):
        for i in range(10):
            for j in range(10):
                number1 = i
                number2 = j
                expected_value = number1 - number2

                self.testArr[i].click()
                self.subtraction_sign.click()
                self.testArr[j].click()
                self.equal_sign.click()
                result_1 = int(self.result.text)

                self.assertEqual(result_1, expected_value)

    def test_subtraction_with_negative_number_in_first_place(self):
        for i in range(10):
            for j in range(10):
                number1 = -(i)
                number2 = j
                expected_value = number1 - number2

                self.testArr[i].click()
                self.negative_sign.click()
                self.subtraction_sign.click()
                self.testArr[j].click()
                self.equal_sign.click()
                result_1 = int(self.result.text)
                # print(f'result_1 = {result_1}')
                self.assertEqual(result_1, expected_value)

    def tearDown(self):
        self.driver.close()


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