**Задание №1**. Выбрать форму регистрации сайте, например, gmail.com или yandex.ru. Найти классы эквивалентности и граничные значения для полей. Определить, на каких данных проводить тестирование этой формы.

Классы эквивалентности:

**Имя ящика:** латинские буквы, цифры, знак подчеркивания, точку и минус

**Имя и фамилия:** Любые символы

**Пароль:** (a–z, A–Z), цифры и символы ! @ # $ % ^ & \* ( ) - \_ + = ; : , . / ? \ | ` ~ { }

Граничные значения:

**Имя ящика:** от 5 до 31 символа

**Имя и фамилия:** от 1 до 40 символов

**Пароль:** От 8 до 40 символов

**Задание №2.** Выбрать ПО и составить 3 сценария тестирования и чек-лист для них.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Проверка функций сервиса mail.ru | | | | |
|  | Сценарий | Google Chrome | Firefox Desktop | Microsoft Edge |
| 1 | Регистрация в сервисе | Успешно | Успешно | Успешно |
| 2 | Удаление почтового ящика | Успешно | Успешно | Успешно |
| 3 | Отправка эл. письма | Успешно | Успешно | Успешно |

**Задание №3.** Выбрать ПО и составить для него баг-репорт.

|  |  |
| --- | --- |
| Заголовок | Невозможно подключиться к адаптеру |
| Проект | ASUS USB-BT400 |
| Компонент приложения | Bluetooth |
| Номер версии | 1.0 |
| Критичность | Критический |
| Приоритет | Высокий |
| Статус | Открыт |
| Автор | Создатель баг репорта |
| Назначен на | Имя сотрудника, назначенного на решение проблемы |
| Как воспроизвести | 1. Подключить адаптер к устройству 2. С любого устройства подключиться по Bluetooth   Полученный результат:  -Устройство не подключено  Ожидаемый результат  -Устройство в процессе подключения к адаптеру |
| Дополнительные сведения | Версия программного обеспечения устройства, к которому подключался адаптер: Windows 10.0.19045.2130 |
| Прикрепленный файл | IMG2134214dsfdasdfds.jpg |

**Задание №4.** Реализовать небольшую библиотеку функций. В библиотеку включить следующие функции:

* определения n чисел Фибоначчи – функция принимает n, возвращает список из чисел
* сортировку пузырьком  функция принимает список чисел, возвращает его отсортированную копию
* калькулятор  функция принимает 3 аргумента: число 1, число 2 и знак действия: +, -, \*, / выполняет действие и возвращает результат

Реализовать unit-тесты данных функций при помощи библиотеки pytest. Реализовать тестирование реакции программы на корректные и не корректные входные данные. Определить классы эквивалентности и граничные значения для входных данных. Код сопроводить комментариями.

**Код:**

myLib.py: содержит требуемые функции

import copy  
  
  
#Принимает list содержащий в себе числа  
#Производится сортировка методом пузырьковой сортировки  
#Возвращает отсортированный list  
#Пример: [2, 5, 0, -5, 42] -> [-5, 0, 2, 5, 42]  
#Классы эквивалентности: list - должен быть типом list  
def bubble\_sort(list1 : list):  
 if list1 == []:  
 raise ValueError  
 arr = copy.deepcopy(list1)  
 def swap(i, j):  
 arr[i], arr[j] = arr[j], arr[i]  
 n = len(arr)  
 swapped = True  
 x = -1  
 while swapped:  
 swapped = False  
 x = x + 1  
 for i in range(1, n - x):  
 if arr[i - 1] > arr[i]:  
 swap(i - 1, i)  
 swapped = True  
 return arr  
  
  
#Принимает n  
#Производится вычисление чисел Фибоначчи  
#Возвращает list чисел Фибоначчи  
#Пример: n = 5 -> [1, 1, 2, 3, 5]  
#Классы эквивалентности: n - натуральное число  
#Граничные значения: n > 0  
def fibon(n):  
 if n < 0:  
 raise IndexError  
 i = 0  
 fib1 = 1  
 fib2 = 1  
 list1 = [fib1, fib2]  
 while i < n - 2:  
 fib\_sum = fib1 + fib2  
 fib1 = fib2  
 fib2 = fib\_sum  
 list1.append(fib2)  
 i = i + 1  
 return list1  
  
  
#Принимает 2 числа, с которыми необходимо произвести вычисления и знак действия  
#Возвращает результат вычислений  
#Пример: 5, 1, + -> 6  
#Классы эквивалентности: num1, num2 - числа; symb - символ  
#Граничные значения: symb == ("+" || "-" || "/" || "\*")  
def calc(num1, num2, symb: str):  
 if symb == "+":  
 return num1 + num2  
 if symb == "-":  
 return num1 - num2  
 if symb == "\*":  
 return num1 \* num2  
 if symb == "/":  
 return num1 / num2  
  
print(bubble\_sort([5,3,4,3,-546,555,666666]))

TestBubbleSort.py: тестирует функцию myLib.bubble\_sort

import myLib  
import pytest  
  
  
# Тест функции, которая производит сортировку пузырьковым методом  
class TestBubbleSort:  
  
 # Тестируем программу на коррктных данных.  
 def test\_on\_correct\_sign(self):  
 list1 = [5, 3, 4, 3, -546, 555, 666666]  
 assert myLib.bubble\_sort(list1) == [-546, 3, 3, 4, 5, 555, 666666]  
  
 # Тестируем программу на некорретных данных. Функция вызывает исключение ValueError.  
 def test\_on\_negative\_value(self):  
 with pytest.raises(ValueError):  
 myLib.bubble\_sort([])

TestCalc.py: тестирует функцию myLib.calc

import myLib  
import pytest  
  
  
# Тест функции, которая производит математические операции  
class TestCalc:  
  
 # Тестируем программу на коррктных данных.  
 def test\_on\_correct\_sign(self):  
 assert myLib.calc(3, 5, '\*') == 15  
  
 # Тестируем программу на некорретных данных. Функция вызывает исключение ZeroDivisionError.  
 def test\_on\_div\_by\_zero(self):  
 with pytest.raises(ZeroDivisionError):  
 myLib.calc(3, 0, '/')

TestFib.py: тестирует функцию myLib.fibon

import myLib  
import pytest  
  
  
# Тест функции, которая производит вычисление чисел Фибоначчи  
class TestFib:  
  
 # Тестируем программу на коррктных данных.  
 def test\_on\_correct\_sign(self):  
 assert myLib.fibon(10) == [1, 1, 2, 3, 5, 8, 13, 21, 34, 55]  
  
 # Тестируем программу на некорретных данных. Функция вызывает исключение IndexError.  
 def test\_on\_negative\_value(self):  
 with pytest.raises(IndexError):  
 myLib.fibon(-50)

Все тесты успешно пройдены