**1 Задание**

Задание:

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 и граничные значения:

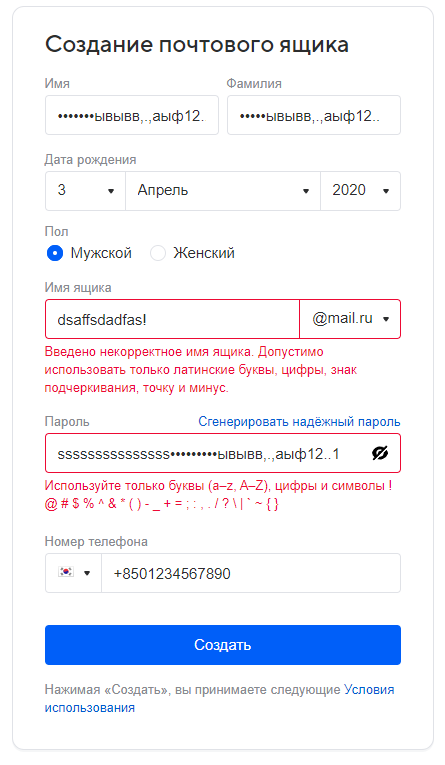
**Имя** - любой текст 40 символов

**Фамилия** - любой текст 40 символов

**Ящик** - латинские буквы, цифры, знак подчеркивания, точку и минус.

**Пароль** - от 8 до 40 символов (только латинские буквы (a–z, A–Z), цифры и символы ! @ # $ % ^ & \* ( ) - \_ + = ; : , . / ? \ | ` ~ { })

**Номер телефона** - только цифры от 11 до 15 символов



**2 Задание**

Задание:

Выбрать ПО и составить 3 сценария тестирования и чек-лист для них.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Проверка функций сервиса mail.ru | | | | | |
|  | Сценарий | Google Chrome | Opera | Firefox | Microsoft Edge |
|  | Регистрация | Успешно | Успешно | Успешно | Успешно |
|  | Авторизация | Успешно | Успешно | Успешно | Успешно |
|  | Просмотр списка входящих сообщений | Успешно | Успешно | Успешно | Успешно |

**3 Задание**

Задание:

Выбрать ПО и составить для него баг-репорт.

Баг репорт для программы Paint

|  |  |
| --- | --- |
| Заголовок | При выборе размера кисти, размер выбирается неправильно |
| Проект | Paint |
| Компонент приложения | Кисть |
| Номер версии | 21H2 |
| Критичность | Значительный |
| Приоритет | Высокий |
| Статус | Новый (баг-репорт только добавлен) |
| Автор | Пупкин В.В. |
| Назначен на | Иванов И.А. |
| Как воспроизвести | Шаги, по которым можно воспроизвести ситуацию, приведшую к ошибке.   1. Запустить Paint 2. Выбрать инструмент “Кисть” 3. Выбрать толщину кисти “8 пкс”   Полученный результат  -Толщина кисти осталось стандартной “3 пкс”  Ожидаемый результат  -Выбрана толщина кисти “8 пкс” |
| Дополнительные сведения | Операционная система: Windows 10 Pro 21H2 |
| Прикрепленный файл | Лога нет |

**4 Задание**

Задание:

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.

По результатам лабораторной работы оформит отчет, исходные код программы выложить на Github.

Код:

main.py: (Содержит тестируемые функции)

# Возвращает результат математической операции "sign" с числами "a" и "b"  
# Входные данные - числа "a" и "b" и знак математической операции "sign"  
# Граничные значения:  
# a, b - числа  
# sign - знак из множества "+, -, \*, /"  
# Возвращает результат математической операции  
def calc(a : float, b: float, sign:str):  
 res = -1  
 if sign in ('+', '-', '\*', '/'):  
 if sign == '+':  
 res = a+b  
 elif sign == '-':  
 res = a-b  
 elif sign == '\*':  
 res = a\*b  
 elif sign == '/':  
 res = a/b  
 return res  
  
# Сортирует список "a" по возрастанию  
# Входные данные - натуральное число "n"  
# Граничные значения:  
# a - список чисел  
# Возвращает отсортированный по возрастанию список "a"  
def bubblesort(a : list):  
 N = len(a)  
 for i in range(N - 1):  
 for j in range(N - i - 1):  
 if a[j] > a[j + 1]:  
 a[j], a[j + 1] = a[j + 1], a[j]  
 return a  
  
# Вывод списка из "n" чисел Фибоначчи  
# Входные данные - натуральное число "n"  
# Граничные значения:  
# a - натуральное число  
# Возвращает "n" чисел Фибоначчи  
def fib(n):  
 if n <= 0:  
 raise IndexError  
 output = []  
 a, b = 0, 1  
 output.append(a)  
 output.append(b)  
 for \_ in range(n-2):  
 a, b = b, a + b  
 output.append(b)  
 return output  
  
def main():  
 print(fib(10))  
 print(bubblesort([1, 5, 3, 2, 4]))  
 print(calc(3, 5, '\*'))  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

test\_calc.py (Тестирует функцию main.calc)

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

test\_fib.py (Тестирует функцию main. fib)

import main  
import pytest  
  
# Тест функции, которая выводит первые "n" чисел Фибоначчи  
class TestFib:  
  
 # Тестируем программу на коррктных данных. Функция возвращает список элементов.  
 def test\_on\_correct\_n(self):  
 assert main.fib(10) == [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]  
  
 # Тестируем программу на некорретных данных. Функция вызывает исключение IndexError.  
 def test\_on\_negative(self):  
 with pytest.raises(IndexError):  
 main.fib(-10)

test\_ bubblesort.py (Тестирует функцию main.bubblesort)

import main  
import pytest  
  
# Тест функции, которая сортирует список  
class TestFib:  
  
 # Тестируем программу на коррктных данных. Функция возвращает отсортированный список элементов.  
 def test\_on\_correct\_n(self):  
 assert main.bubblesort([1, 5, 3, 2, 4]) == [1, 2, 3, 4, 5]  
  
 # Тестируем программу на некорретных данных. Функция вызывает исключение TypeError.  
 def test\_on\_negative(self):  
 with pytest.raises(TypeError):  
 main.bubblesort(-4563)

Все тесты успешно пройдены