Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный** **исследовательский политехнический университет»**

Факультет: Прикладной математики и механики

Кафедра: Вычислительной математики, механики и биомеханики

Направление магистратуры: 09.03.02 Информационные системы и технологии

Профиль: «Информационные системы и технологии»

**О Т Ч Е Т**

по лабораторной работе

по дисциплине

**«Информатика в приложении к отрасли»**

Лабораторная работа № **2**

Тема работы: «**Тестирование программного обеспечения**»

**Выполнил** студент гр. ЦТУ-20-1б

**Мясников Артем Евгеньевич**

**Принял**

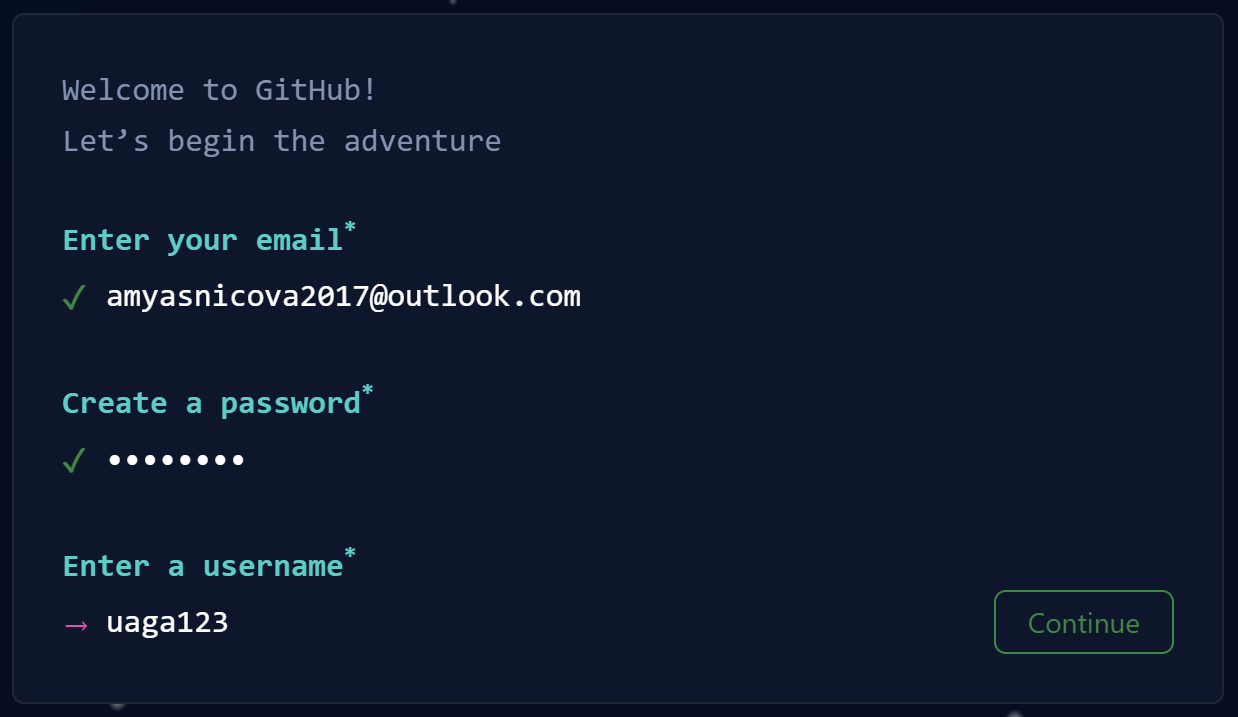
Ведущий инженер каф.ВММБ Нетбай Георгий Владимирович

**Пермь 2023**

**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.

Форма регистрации GitHub.com:



Классы эквивалентности:

1. Email – класс, имеющий шаблон: логин@сервер.домен
2. Имя и фамилия пользователя – класс, принимающий любые значения на латинице не начинающихся с цифры
3. Пароль – класс, значения которого скрыто при вводе

Граничные значения:

1. Email – должен быть похожим на шаблон
2. Пароль – произвольная граница на сложность пароля
3. Имя и фамилия пользователя – не должен начинаться с цифры и используются только латинский алфавит и цифры

**Задание №2**

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

Тест-кейсы:

|  |  |
| --- | --- |
| Идентификатор | Тестирование регистрации №1 |
| Краткое описание | Тестирование регистрации на сайте GitHub.com |
| Предусловия | На сайте пользователь ещё не авторизован |
| Шаги | 1. Корректно заполнить поля регистрации 2. Нажать на кнопку регистрации |
| Ожидаемый результат | Сайт зарегистрирует пользователя |

|  |  |
| --- | --- |
| Идентификатор | Тестирование регистрации №2 |
| Краткое описание | Тестирование регистрации на сайте GitHub.com |
| Предусловия | На сайте пользователь ещё не авторизован |
| Шаги | 1. В полях регистрации не вписать почту, а остальные поля заполнить корректно 2. Нажать на кнопку регистрации |
| Ожидаемый результат | Сайт не зарегистрирует пользователя, кнопка не сработает |

|  |  |
| --- | --- |
| Идентификатор | Тестирование регистрации №3 |
| Краткое описание | Тестирование регистрации на сайте GitHub.com |
| Предусловия | На сайте пользователь ещё не авторизован |
| Шаги | 1. Ввести несуществующую почту (Пример: OLOLO234@WER.com)в поле для почты. Остальные поля заполнить корректно 2. Нажать на кнопку регистрации |
| Ожидаемый результат | Сайт не зарегистрирует пользователя, но кнопка сработает |

Чек-лист:

|  |  |
| --- | --- |
| Этап | Сценарий |
| Сценарий 1 | Успешно |
| Сценарий 2 | Успешно |
| Сценарий 3 | Сайт завис |

**Задача №3**

Выбрать ПО и составить для него баг-репорт.

|  |  |
| --- | --- |
| Заголовок | Ошибка регистрации при вводе некорректной почты |
| Проект | Регистрация на сайте GitHub.com |
| Компонент приложения | Регистрация на сайте |
| Номер версии | Актуальная версия сайта Desmos на 04.06.2023 |
| Критичность | Незначительный |
| Приоритет | Низкий |
| Статус | Новый |
| Автор | Мясников Артем |
| Назначен на | Администратор сайта GitHub.com |
| Как воспроизвести | 1. В поле email ввести почту со знаком # в логине (пример: ###@gmail.com) 2. Нажать на кнопку регистрации |
| Дополнительные сведения | Ошибка была обнаружена в браузере Yandex 23.5.1.714 (64-bit) |
| Прикрепленный файл | Рис.1 |

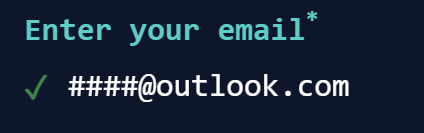


Рис. 1Корректность email с ###

**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.

Классы эквивалентности:

Функция *fibonacci* содержит один класс эквивалентности, который представляет собой целые числа.

Функция *sort* содержит один класс эквивалентности, представляющий из себя массив чисел.

Функция *calc* содержит два класса эквивалентности: первый – число, второй – символ, обозначающий действие, которое должен совершить калькулятор между числами.

Граничные значения:

В функции *fibonacci* принимаются лишь значения целых чисел, другие значения не обрабатываются.

В функции *sort* принимается множество чисел разделённые пробелами, другие значения не обрабатываются.

В функции *calc* принимаются только числа и символ математической операции, другие значения не обрабатываются.

Код:

Функции:

def fibonacci(n):

if not(int(n) == n):

return []

array = []

for i in range(n):

if i == 0: array.append(1)

if i == 1: array.append(1)

if i > 1: array.append(array[i - 1] + array[i - 2])

return array

def sort(nums: str):

array = []

for num in nums.split():

array.append(int(num))

for i in range(len(array) - 1):

for j in range(len(array) - 1):

if array[j] > array[j + 1]:

array[j], array[j + 1] = array[j + 1], array[j]

return array

def calc(a, b, operator):

return {

operator == "+": a + b,

operator == "-": a - b,

operator == "/": a / b,

operator == "\*": a \* b

}[True]

Тесты:

import pytest

import function

class TestFibonacci:

def test\_1(self):

assert function.fibonacci(10) == [1, 1, 2, 3, 5, 8, 13, 21, 34, 55]

def test\_2(self):

assert len(function.fibonacci(-2)) == 0

def test\_3(self):

assert function.fibonacci(1) == [0]

def test\_4(self):

assert function.fibonacci(1.9) == []

class TestSort:

def test\_1(self):

assert function.sort("9 8 7 6 5 4 3 2 1") == [1, 2, 3, 4, 5, 6, 7, 8, 9]

def test\_2(self):

assert function.sort([0, -1]) == [-1, 0]

def test\_3(self):

assert function.sort([-1, 1]) == [1, -1]

class TestCalc:

def test\_1(self):

assert function.calc(1, 2, '+') == 3

def test\_2(self):

assert function.calc(5, 2, '-') == 3

def test\_3(self):

assert function.calc(6, 3, '\*') == 18

def test\_4(self):

assert function.calc(6, 3, '/') == 2

def test\_5(self):

with pytest.raises(AssertionError):

assert function.calc(3, '+', 4)

def test\_6(self):

assert function.calc(6, "aab", 3) is None

Результаты тестов: 