Отчет

Лабораторная работа с регистрацией:  
Тестов реализовано: 23

Пройдено тестов: 23

#valid inputs

        self.assertEqual(check\_if\_login\_is\_valid("login@mail.ru"),("Success","")) – Проверка логина на валид

        self.assertEqual(check\_if\_login\_is\_valid("+7-999-999-9999"),("Success",""))– Проверка логина на валид

        self.assertEqual(check\_if\_login\_is\_valid("5simvolov"),("Success",""))– Проверка логина на валид

        # invalid inputs

        self.assertEqual(check\_if\_login\_is\_valid("edg1337"),("Failure","User with this login already exists")) – Проверка на уже существующий логин

        self.assertEqual(check\_if\_login\_is\_valid("+7-999-999-999"),("Failure","Phone number is in incorrect format, must be (+x-xxx-xxx-xxxx)")) – Проверка на тип логина

        self.assertEqual(check\_if\_login\_is\_valid("edg@mailru"),("Failure", "Email must be in a format of (xxx@xxx.xx)")) – Проверка на тип логина

        self.assertEqual(check\_if\_login\_is\_valid("aaa"),("Failure", "Login must be at least 5 characters long"))– Проверка на тип логина

        self.assertEqual(check\_if\_login\_is\_valid("somelogin\*\*\*"),("Failure", "Login can only contain latin characters, numbers from 0 to 9, and \_ symbol")) – проверка на валид

        self.assertEqual(check\_if\_login\_is\_valid(""),("Failure", "Login can not be empty")) – проверка на валид

self.assertEqual(validate\_password("jeka","jeka"),("Password must be at least 7 characters long")) - проверка пароля на валид

        self.assertEqual(validate\_password("jeka","jeka2"),("Passwords does not match")) – Проверка на совпадение паролей

        self.assertEqual(validate\_password("ааааааааааааааа","ааааааааааааааа"),("Password must contain at least 1 special symbol")) ")) - проверка пароля на валид

        self.assertEqual(validate\_password("латиницавпароле+","латиницавпароле+"),("Password must contain at least 1 uppercase letter"))")) - проверка пароля на валид

        self.assertEqual(validate\_password("12345678","12345678"),("Password must contain at least 1 uppercase letter"))")) - проверка пароля на валид

        self.assertEqual(validate\_password("coolpasswordxd1","coolpasswordxd1"),("Password can not contain any latin characters"))")) - проверка пароля на валид

        self.assertEqual(validate\_password("Пожалуйстапомогите+","Пожалуйстапомогите+"),("Password must contain at least 1 number"))")) - проверка пароля на валид

        self.assertEqual(validate\_password("ПАРОЛЬКАПСОМ+1","ПАРОЛЬКАПСОМ+1"),("Password must contain at least 1 lowercasse letter"))")) - проверка пароля на валид

        #valid

        self.assertEqual(validate\_password("Подходящийпароль1+1","Подходящийпароль1+1"),("All good")) ")) - проверка пароля на валид

        self.assertEqual(validate\_password("Какойтопароль+2","Какойтопароль+2"),("All good"))")) - проверка пароля на валид

        self.assertEqual(validate\_password("Последнийпарольнатест))2","Последнийпарольнатест))2"),("All good"))")) - проверка пароля на валид

       self.assertEqual(check\_login\_type("login@mail.ru"),(2)) – Проверка на тип логина

        self.assertEqual(check\_login\_type("+7-999-999-9999"),(1)) - Проверка на тип логина

        self.assertEqual(check\_login\_type("shlyapa"),(3)) -Проверка на тип логина

Треугольники:

Всего тестов: 20

Пройдено тестов: 20

        self.assertEqual(classifyTriangle(1,90,1),(4)) – Проверка классификации

        self.assertEqual(classifyTriangle(1,1,1),(1)) – Проверка классификации

        self.assertEqual(classifyTriangle(5,5,3),(2)) – Проверка классификации

        self.assertEqual(classifyTriangle(3,4,5),(3)) – Проверка классификации

#invalid

        self.assertEqual(inputDataValidationAndCalculation("a",90,1),('')) – проверка на валидность введенных данных

        self.assertEqual(inputDataValidationAndCalculation(-2,90,1),('Arguments must be in a float type')) – проверка на валидность введенных данных

        self.assertEqual(inputDataValidationAndCalculation((1,2,3),90,1),('Arguments must be in a float type')) – проверка на валидность введенных данных

        self.assertEqual(inputDataValidationAndCalculation([1,2],90,1),('Arguments must be in a float type')) – проверка на валидность введенных данных

        self.assertEqual(inputDataValidationAndCalculation({1:2},90,1),('Arguments must be in a float type')) – проверка на валидность введенных данных

        self.assertEqual(inputDataValidationAndCalculation({1,2},90,1),('Arguments must be in a float type')) – проверка на валидность введенных данных

        self.assertEqual(inputDataValidationAndCalculation(False,90,1),('Arguments must be in a float type')) – проверка на валидность введенных данных

        #valid

        self.assertEqual(inputDataValidationAndCalculation(5.0,5.0,3.0),('Isosceles')) – проверка на валидность введенных данных

        self.assertEqual(inputDataValidationAndCalculation(2.0,2.0,2.0),('Equilateral') – проверка на валидность введенных данных

        self.assertEqual(inputDataValidationAndCalculation(3.0,5.0,4.0),('Scalene')) – проверка на валидность введенных данных

        self.assertEqual(getTriangleCoordinates(2.0, 2.0, 2.0), [[0, 0], [2, 0], [0, 1]]) – Проверка вывода

        self.assertEqual(getTriangleCoordinates(2.0,2.0,2.0),[[0, 0], [2, 0], [0, 1]]) – Проверка вывода

        self.assertEqual(getTriangleCoordinates(3.0,5.0,4.0),[[0, 0], [3, 0], [1, -3]]) – Проверка вывода

        self.assertEqual(getTriangleCoordinates(300000.0,5.0,4.0),([-1, -1], [-1, -1], [-1, -1])) – Проверка вывода

        self.assertEqual(getTriangleCoordinates(3.0,4.0,5.0),[[0, 0], [3, 0], [-3, -3]]) – Проверка вывода

        self.assertEqual(getTriangleCoordinates(5.0,5.0,5.0),[[0, 0], [5, 0], [1, -4]]) – Проверка вывода