Вариативная лабораторная работа №4

**Тема:** Тестирование

**Задача 4.2**

**Постановка задачи:** Разработать программу для считывания данных из JSON-файла и вывода их в табличном виде на экран и протестировать работоспособность с использованием unittest.

**Код программы:**

"""

Разработать программу для считывания данных JSON-формата из файла и вывод их в табличном виде на экран.

Организовать тестирование работоспособности программы с помощью assert, print.

"""

import json

def json\_table(data):

table = []

sstr = '| {id:^3} | {first\_name:^10} | {last\_name:^15} | {email:^30} | {gender:^6} | {ip\_address:^16} |'

t\_caption = '| {:^3} | {:^10} | {:^15} | {:^30} | {:^6} | {:^16} |'.format('id','first\_name','last\_name','email','gender','ip\_address')

roof = '-'\*len(t\_caption)

table.append(roof)

table.append(t\_caption)

table.append(roof)

for el in range(len(data)):

temp = data[el]

res = sstr.format(\*\*temp)

table.append(res)

table.append(roof)

return(table)

import unittest

class TestFunc(unittest.TestCase):

def setUp(self):

with open('/home/vasya/Документы/Github/prog3/srTest/vsr/MOCKDATA.json') as f:

data\_dict = json.load(f)

self.data = data\_dict

def test\_equal(self):

self.assertEqual(json\_table(self.data)[1],'| id | first\_name | last\_name | email | gender | ip\_address |')

def test\_istuple(self):

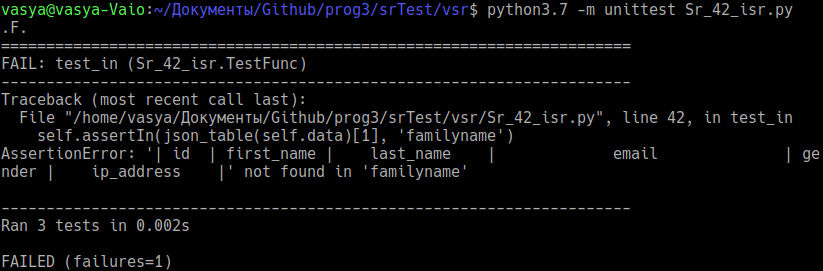
self.assertIs(type(json\_table(self.data)),list)

def test\_in(self):

self.assertIn(json\_table(self.data)[1], 'familyname')

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

**Результат тестирования:**

**Задача 4.2**

**Постановка задачи:** Дополнение программы задания 4.1 (считывание данных JSON-формата) тестами с использованием библиотеки doctest.

**Код программы:**

"""

Разработать программу для считывания данных JSON-формата из файла и вывод их в табличном виде на экран.

Организовать тестирование работоспособности программы с помощью assert, print.

"""

import json

def json\_table():

"""

>>> json\_table()[1]

'| id | first\_name | last\_name | email | gender | ip\_address |'

"""

with open('MOCKDATA.json') as f:

data\_dict = json.load(f)

table = []

sstr = '| {id:^3} | {first\_name:^10} | {last\_name:^15} | {email:^30} | {gender:^6} | {ip\_address:^16} |'

t\_caption = '| {:^3} | {:^10} | {:^15} | {:^30} | {:^6} | {:^16} |'.format('id','first\_name','last\_name','email','gender','ip\_address')

roof = '-'\*len(t\_caption)

table.append(roof)

table.append(t\_caption)

table.append(roof)

for el in range(len(data\_dict)):

temp = data\_dict[el]

res = sstr.format(\*\*temp)

table.append(res)

table.append(roof)

return(table)

def main():

a = json\_table()

"""

for el in table:

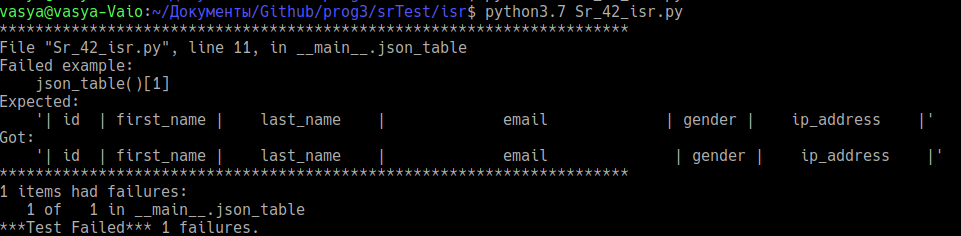
print(el)

"""

if \_\_name\_\_ == "\_\_main\_\_":

import doctest

doctest.testmod()

**Результат тестирования:**