**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Казовые компоненты интернет технологий»

Отчет по домашнему заданию

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Руководитель: |
| студент группы ИУ5-33Б: |  | преподаватель каф. ИУ5 |
| Лебедева С.К. |  | Гапанюк Ю.Е. |
|  |  |  |

Москва, 2022 г.

**Цель домашнего задания:**

Разработка комплексного приложения на языке Python.

**Задание:**

Домашнее задание состоит из решения нескольких задач.

**Задание №1:**

С использованием механизма итераторов или генераторов реализуйте с помощью концепции ленивых вычислений одну из последовательностей OEIS. Примером могут являться числа Фибоначчи.

**Задание №2:**

Для реализованной последовательности разработайте 3-5 модульных тестов, которые, в том числе, проверяют то, что последовательность поддерживает ленивые вычисления.

**Задание №3:**

Разработайте веб-сервис с использованием фреймворка Flask, который возвращает N элементов последовательности (параметр N передается в запросе к сервису).

**Задание №4:**

Создайте Jupyter-notebook, который реализует обращение к веб-сервису с использованием библиотеки requests и визуализацию полученных от веб-сервиса данных с использованием библиотеки matplotlib.

**Текст программы:**

**src/fib.py**

def fib():

    '''

    Генераторная функция для чисел Факториал

    '''

    prev, cur = 0, 1

    while True:

        yield cur

        prev, cur = cur, prev + cur

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

    fib\_gen = fib()

    res = [next(fib\_gen) for \_ in range(20)]

    print(res)

**src/get\_numbers.py**

import requests

import matplotlib.pyplot as plt

def make\_url(cnt):

    base\_url = 'http://127.0.0.1:5000/num/'

    res = base\_url + str(cnt)

    return res

def get\_data(cnt):

    url = make\_url(cnt)

    r = requests.get(url)

    return r.json()

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

    y = get\_data(int(input('Сколько чисел Фибоначчи?')))

    print(y)

    print(len(y))

    x = list(range(1, len(y) + 1))

    fig = plt.figure(figsize=(10, 5))

    plt.bar(x, y)

    plt.xlabel('Ось абсцисс')

    plt.ylabel('Ось ординат')

    plt.title('Первые {} чисел последовательности Фибоначчи'.format(len(y)))

    plt.show()

    plt.plot(x, y)

    plt.show()

**app.py**

from flask import Flask

from src.fib import fib

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

    return '<p>Returning the Fibonacci numbers!</p>'

@app.route('/num/<int:cnt>')

def get\_fib(cnt):

    fib\_gen = fib()

    res = [next(fib\_gen) for \_ in range(cnt)]

    return res

**main.py**

from app import app

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

    app.run()

**Tests.py**

import unittest

from src.fib import fib

class test\_fib(unittest.TestCase):

    def test\_1(self):

        fib\_gen = fib()

        res = [next(fib\_gen) for \_ in range(10)]

        self.assertEqual(len(res), 10)

        self.assertEqual(res, [1, 1, 2, 3, 5, 8, 13, 21, 34, 55])

    def test\_2(self):

        fib\_gen = fib()

        res = [next(fib\_gen) for \_ in range(3)]

        self.assertEqual(len(res), 3)

        self.assertEqual(res, [1, 1, 2])

    def test\_3(self):

        fib\_gen = fib()

        res = [next(fib\_gen) for \_ in range(5)]

        self.assertEqual(len(res), 5)

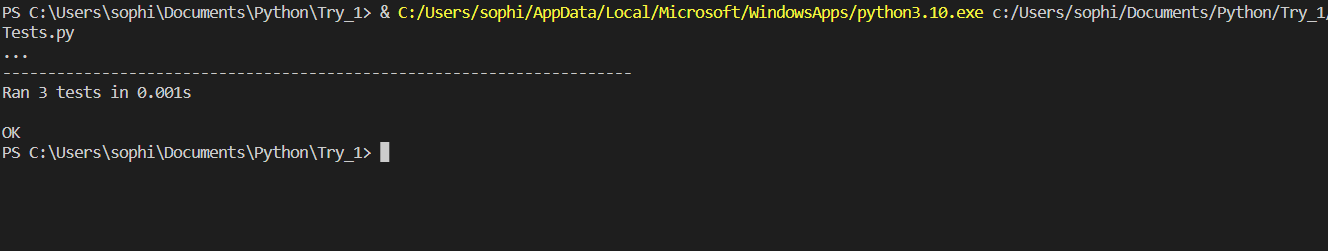
        self.assertEqual(res, [1, 1, 2, 3, 5])

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

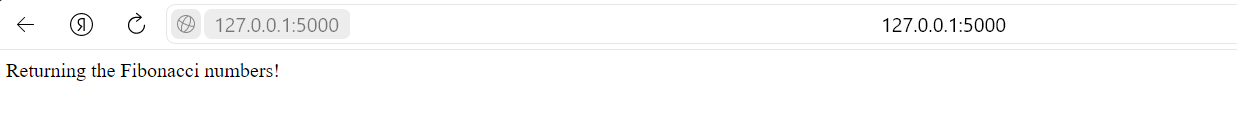
    unittest.main()

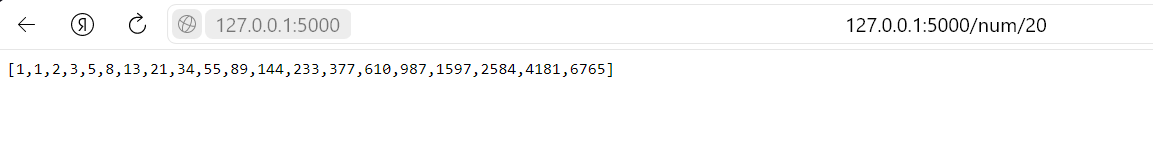
**Примеры выполнения:**

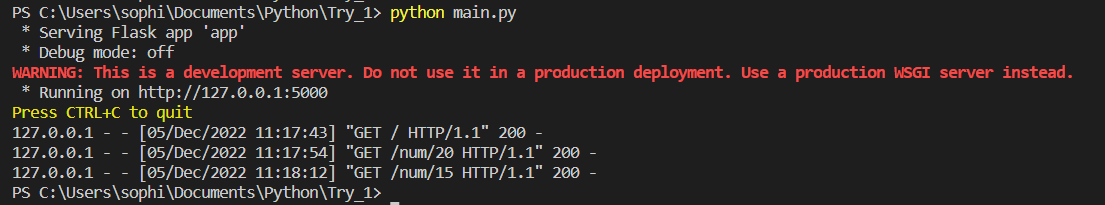
*Прохождение тестов:*

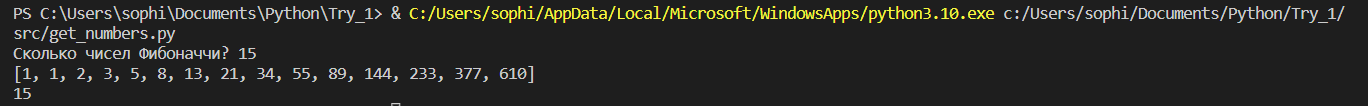
****

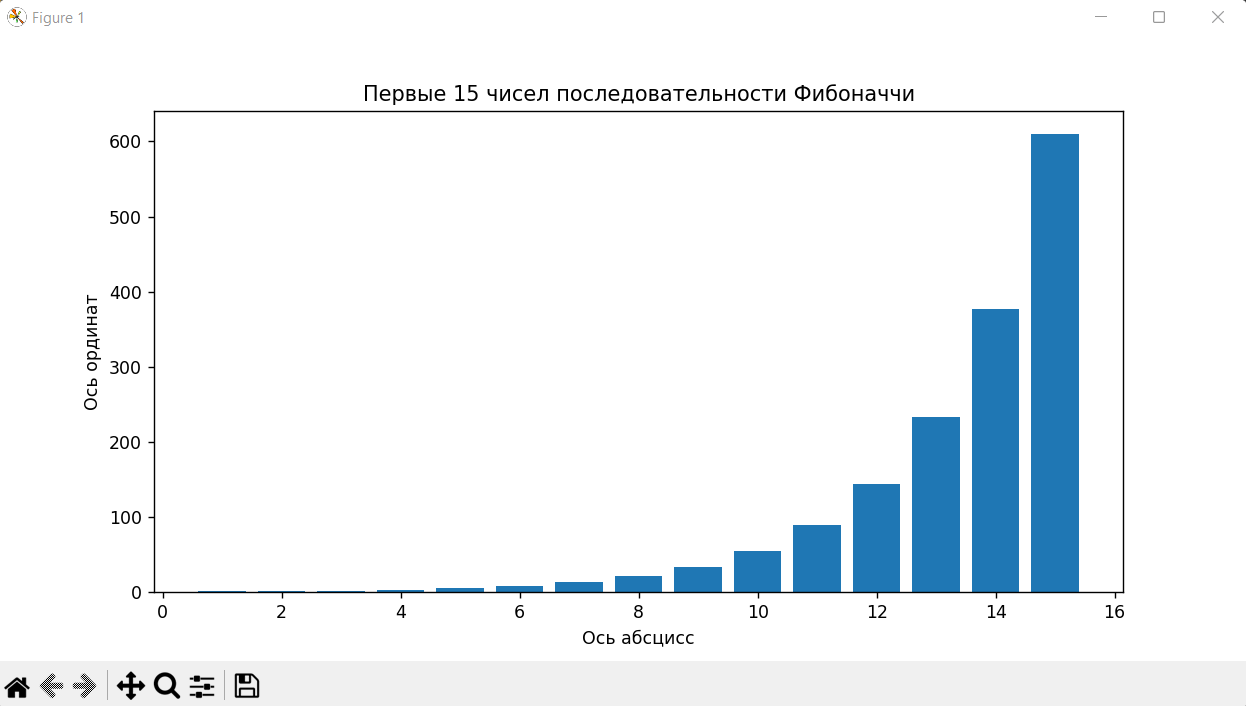
*Выполнение основной части задания:*

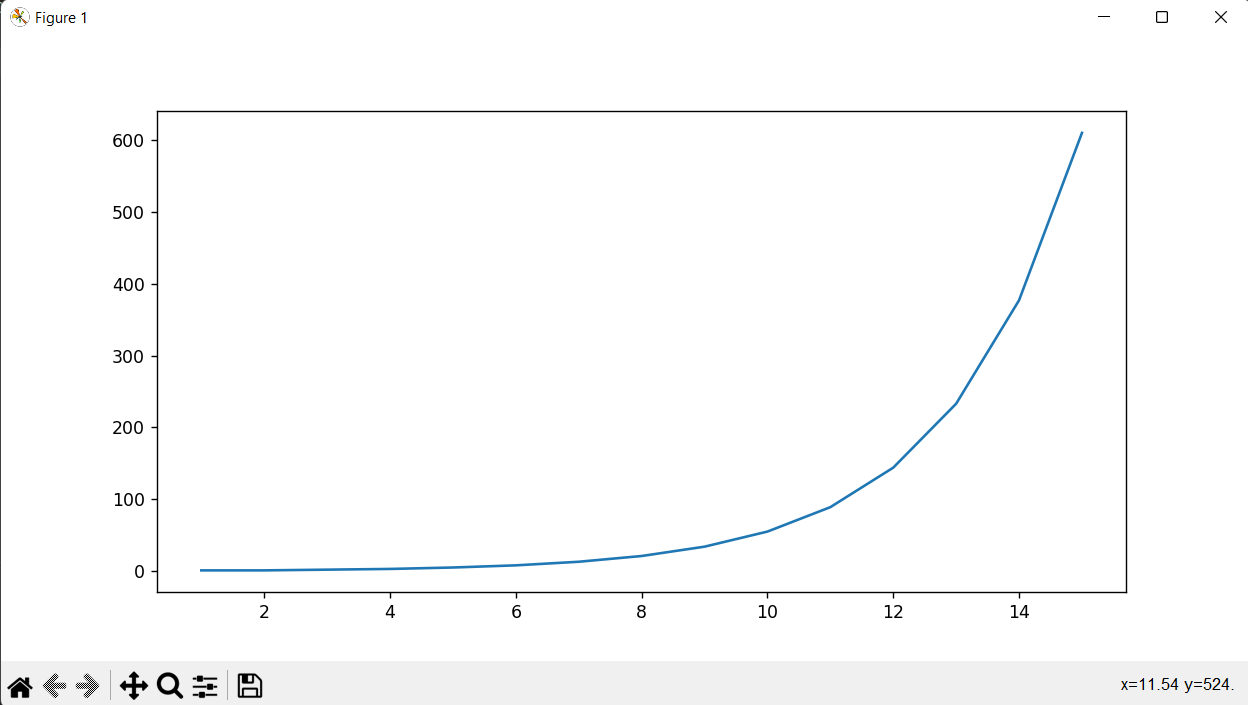
****

****

****

****

****

****