**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ**

**Московский государственный технический университет**

**им. Н.Э. Баумана**

**(МГТУ им. Н.Э. Баумана)**

**Кафедра «Системы обработки информации и управления» (ИУ5)**

Отчёт по домашнему заданию

По курсу: «Базовые компоненты интернет-технологий»

Выполнил: Никулин Данила Дмитриевич

студент группы ИУ5-31Б.

Проверил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_.\_\_\_.2022г.

Подпись: \_\_\_\_\_\_\_\_\_\_\_.

г. Москва 2022 г.

**Задание:**

1. С использованием механизма итераторов или генераторов реализуйте с помощью концепции ленивых вычислений одну из последовательностей OEIS. Примером могут являться числа Фибоначчи.
2. Для реализованной последовательности разработайте 3-5 модульных тестов, которые, в том числе, проверяют то, что последовательность поддерживает ленивые вычисления.
3. Разработайте веб-сервис с использованием фреймворка Flask, который возвращает N элементов последовательности (параметр N передается в запросе к сервису).
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.

**Приложение 1. Текст программы:**

**generator.py**

def fib(n):

a, b = 0, 1

for i in range(n):

yield a

a, b = b, a + b

**unittest.py**

from unittest import TestCase, main

from generator import fib

import time

class fib\_test(TestCase):

def test\_\_fib\_1(self):

arr = [i for i in fib(10)]

true\_arr = [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]

self.assertEqual(arr, true\_arr)

def test\_fib\_2(self):

arr = [i for i in fib(0)]

true\_arr = []

self.assertEqual(arr, true\_arr)

def test\_fib\_3(self):

arr = [i for i in fib(1)]

true\_arr = [0]

self.assertEqual(arr, true\_arr)

def test\_time\_fib\_1(self): #lazy evaluation

begin = time.time()

a = fib(1000000)

end = time.time() - begin

self.assertLess(end, 1)

def test\_time\_fib\_2(self): #calculation on demand

begin = time.time()

a = [i for i in fib(1000000)]

end = time.time() - begin

self.assertLess(1, end)

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

main()

**flask\_app.py**

from flask import Flask

import generator

app = Flask('fibonacci sequences')

@app.route('/')

def main\_page():

return "<h1>Educational project flask app!</h1>"

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

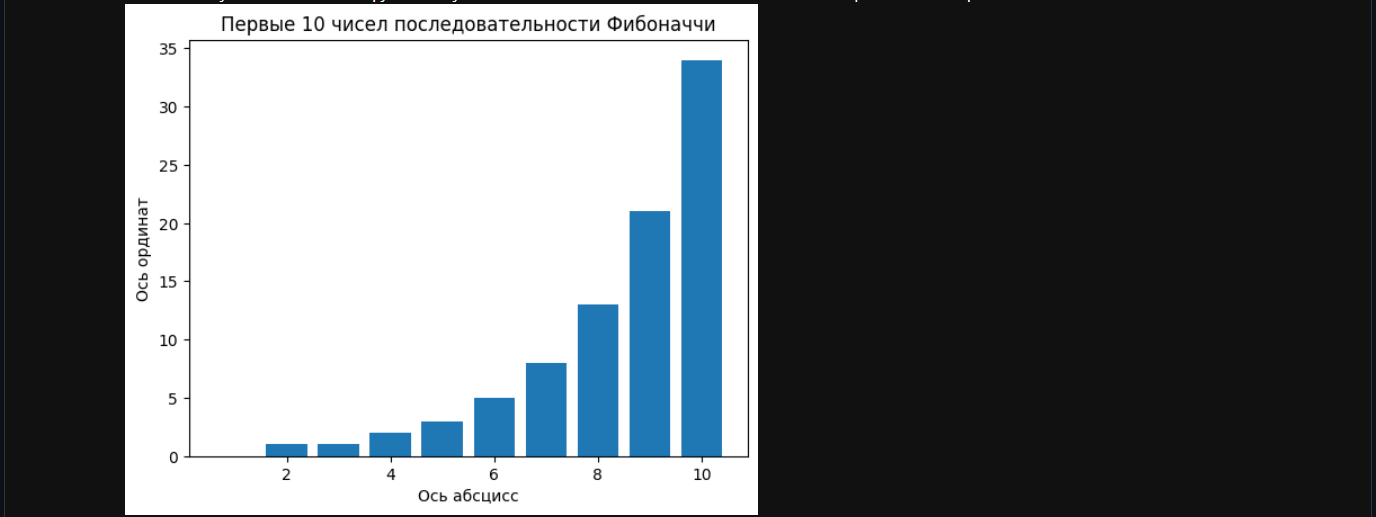
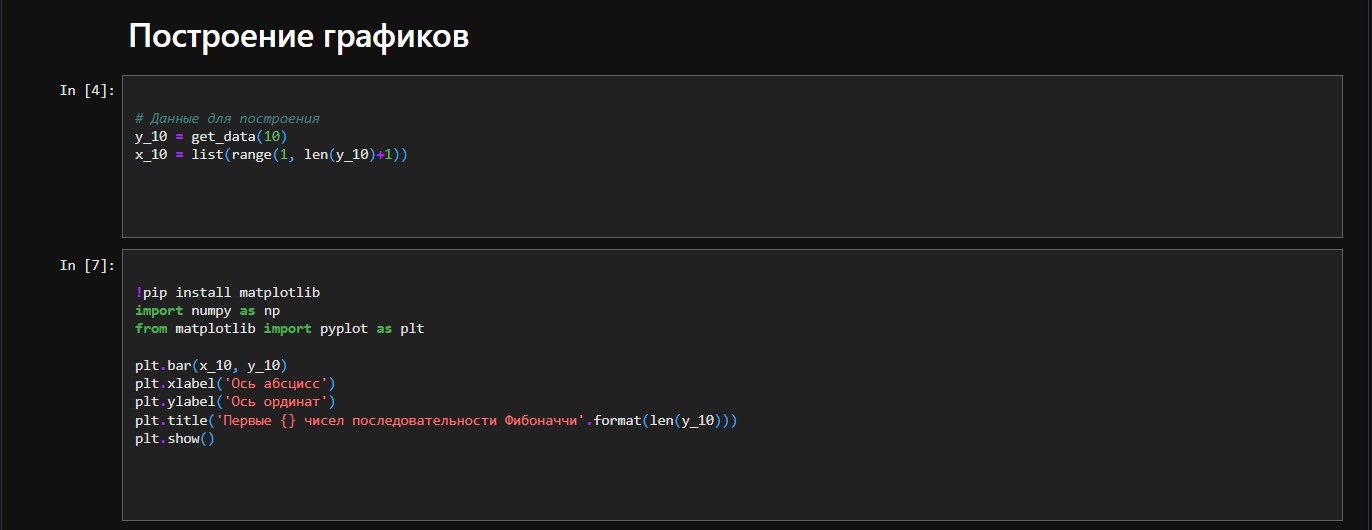
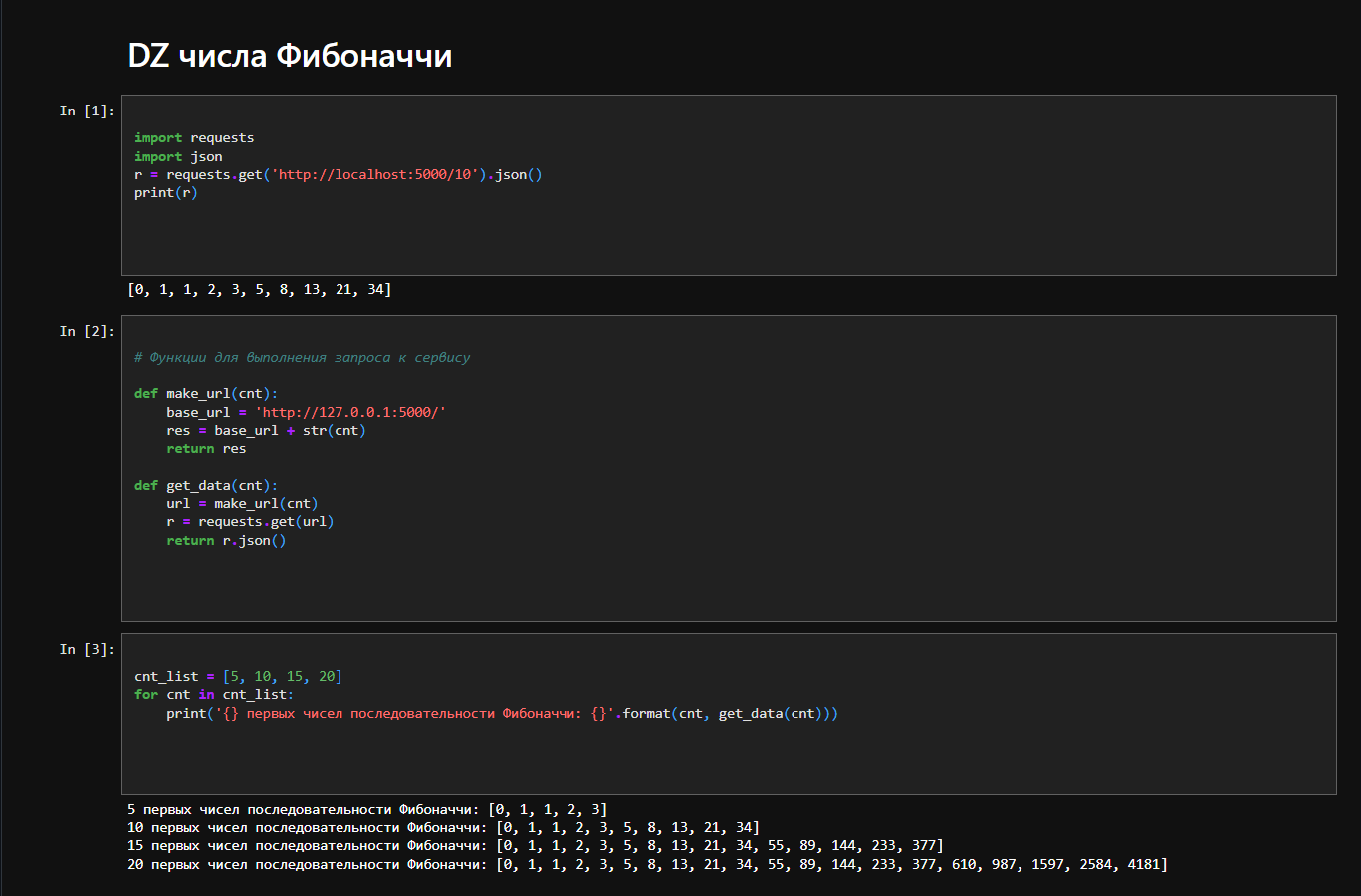
def get\_sequence(n):

return list(generator.fib(n))

@app.errorhandler(404)

def page\_not\_found(e):

return "<h1>Oops! Try to write after URL /your\_number</h1>"

******Приложение 2. Результаты тестирования:**