**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчет по домашнему заданию

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-35Б |  | преподаватель каф. ИУ5 |
| Самойлов Константин |  | Нардид А.Н. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2022 г.

**Описание задания**

### Задание:

1. С использованием механизма итераторов или генераторов реализуйте с помощью концепции ленивых вычислений [одну из последовательностей OEIS.](https://ru.wikipedia.org/wiki/%D0%9E%D0%BD%D0%BB%D0%B0%D0%B9%D0%BD-%D1%8D%D0%BD%D1%86%D0%B8%D0%BA%D0%BB%D0%BE%D0%BF%D0%B5%D0%B4%D0%B8%D1%8F_%D1%86%D0%B5%D0%BB%D0%BE%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D1%85_%D0%BF%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D0%B5%D0%B9) Примером могут являться [числа Фибоначчи.](https://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%BB%D0%B0_%D0%A4%D0%B8%D0%B1%D0%BE%D0%BD%D0%B0%D1%87%D1%87%D0%B8)
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](https://requests.readthedocs.io/en/latest/) и визуализацию полученных от веб-сервиса данных с использованием библиотеки [matplotlib.](https://matplotlib.org/)

**Текст программы**

**fibonacci.py**

def fib(n):  
 prev, curr = 0, 1  
 for i in range(n):  
 yield prev  
 prev, curr = curr, prev + curr

**tests.py**

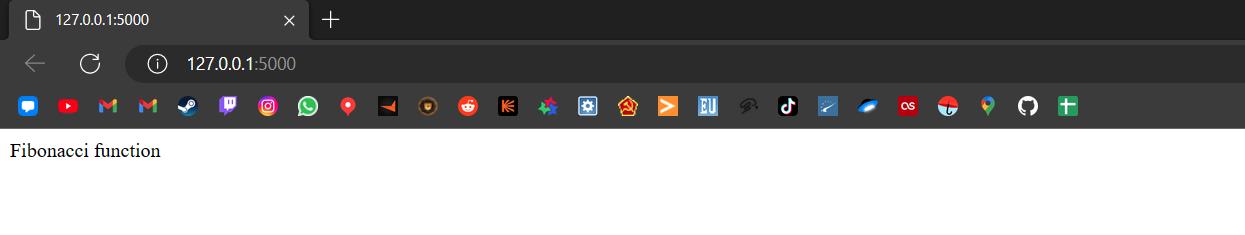
import unittest  
from fibonacci import fib  
from time import time  
  
class fibonacci(unittest.TestCase):  
 def test\_fib5(self):  
 a = [i for i in fib(5)]  
 expected = [0, 1, 1, 2, 3]  
 self.assertEqual(a, expected)  
  
 def test\_fib15(self):  
 a = [i for i in fib(15)]  
 expected = [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377]  
 self.assertEqual(a, expected)  
  
 def test\_fib0(self):  
 a = [i for i in fib(0)]  
 expected = []  
 self.assertEqual(a, expected)  
  
 def test\_fib\_time1(self):  
 start\_time = time()  
 a = fib(100000)  
 end\_time = time() - start\_time  
 self.assertLess(end\_time, 0.5)  
  
 def test\_fib\_time2(self):  
 start\_time = time()  
 a = [i for i in fib(100000)]  
 end\_time = time() - start\_time  
 self.assertLess(end\_time, 0.5)  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 unittest.main()

**flask1.py**

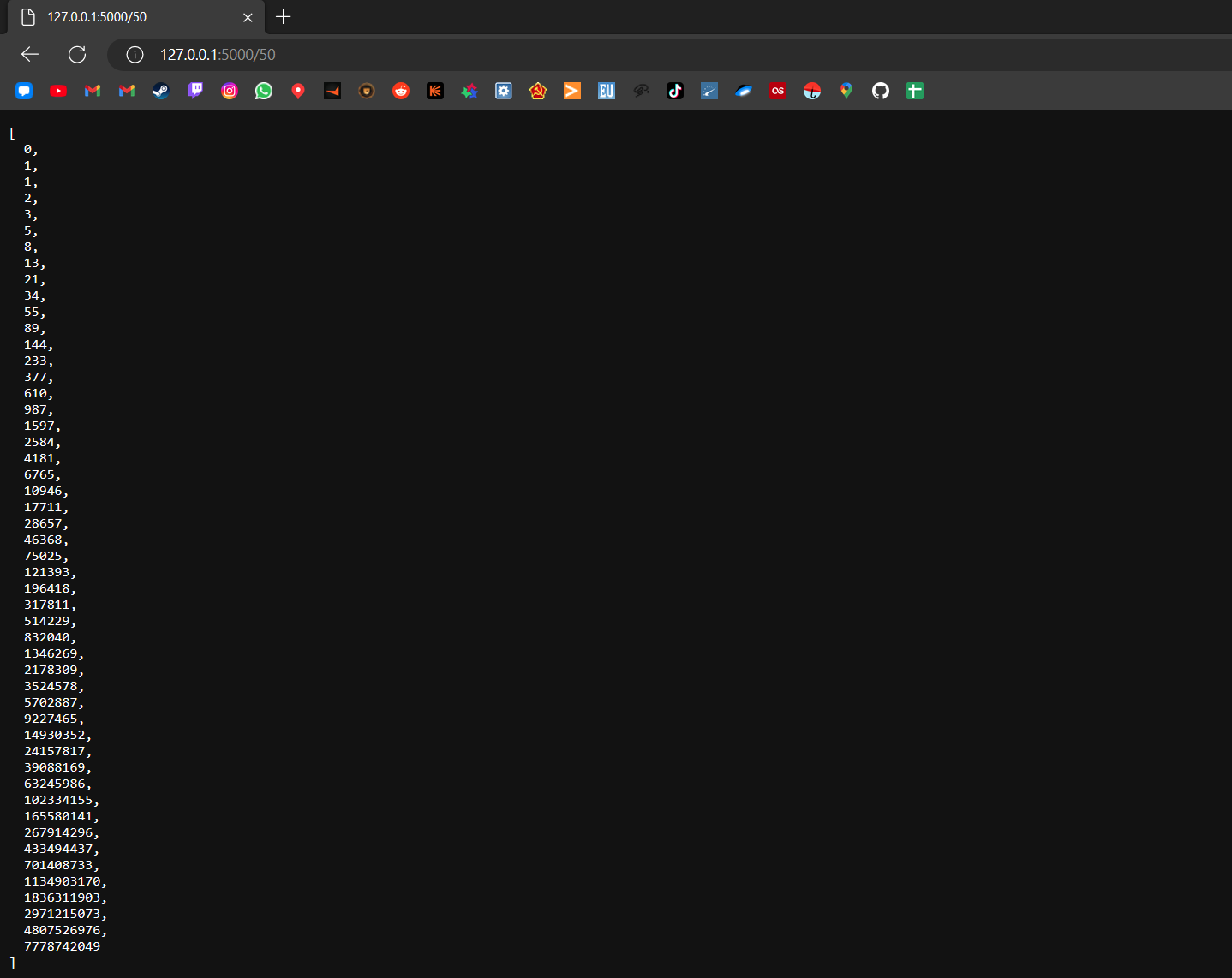
from flask import Flask  
from fibonacci import fib  
  
app = Flask(\_\_name\_\_)  
  
@app.route('/')  
def index():  
 return "<p>Fibonacci function<p>"  
  
@app.route('/<int:cnt>')  
def number(cnt):  
 fib\_gen = fib(cnt)  
 res = [next(fib\_gen) for i in range(cnt)]  
 return res  
  
@app.errorhandler(404)  
def not\_found\_error(error):  
 return "Error, try to enter an int number"  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app.run(debug = True)

**Экранные формы**

**Главное окно запущенного веб-сервиса:**

****

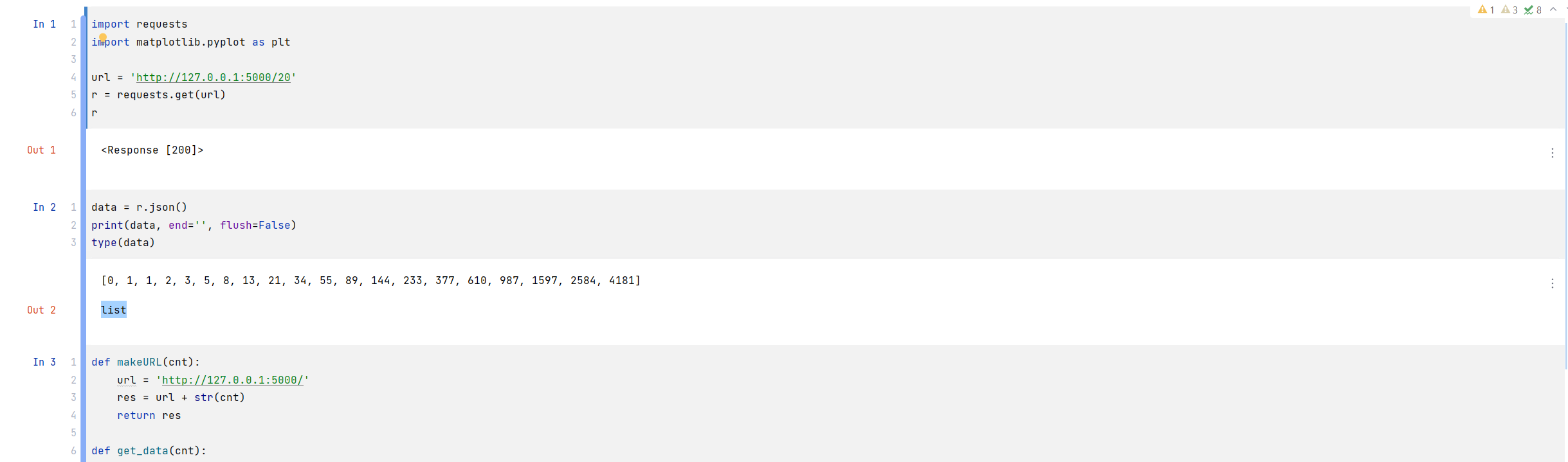
**Окно веб-сервиса с выданными на запрос 50 первыми числами Фиббоначи:**

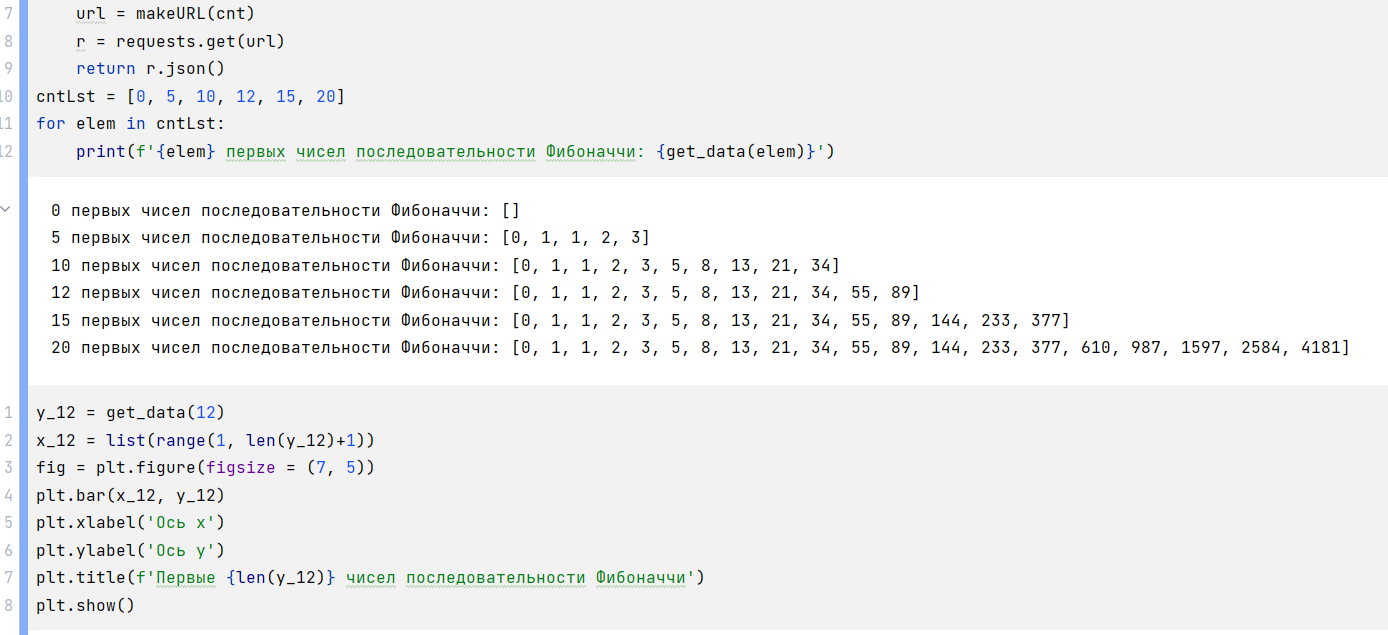
****

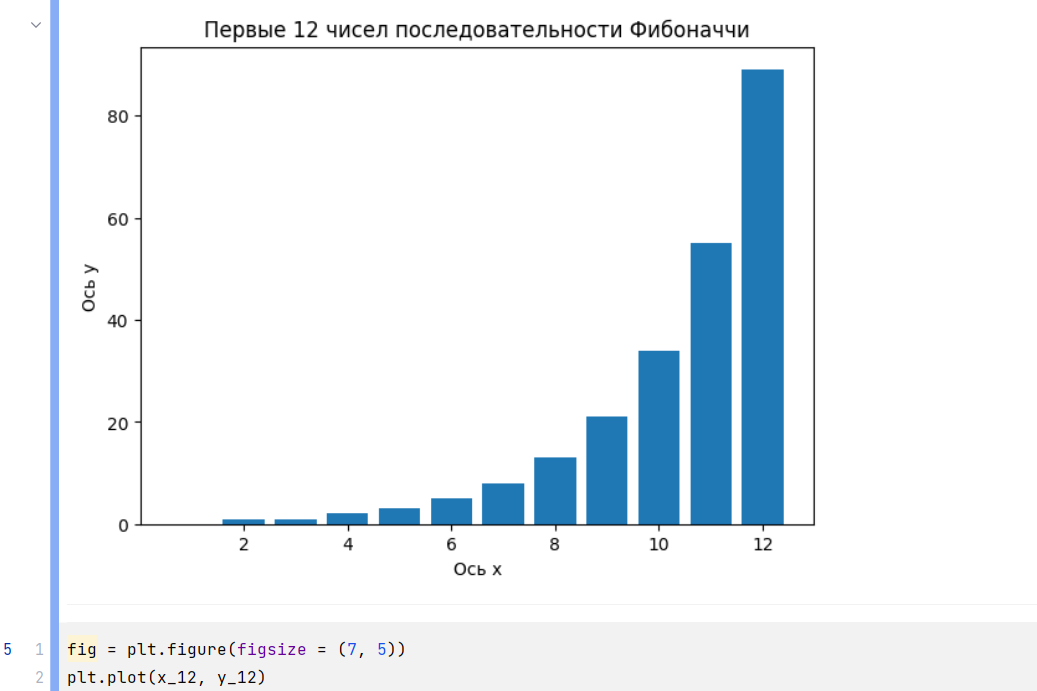
**Запуск веб-сервиса из терминала:**

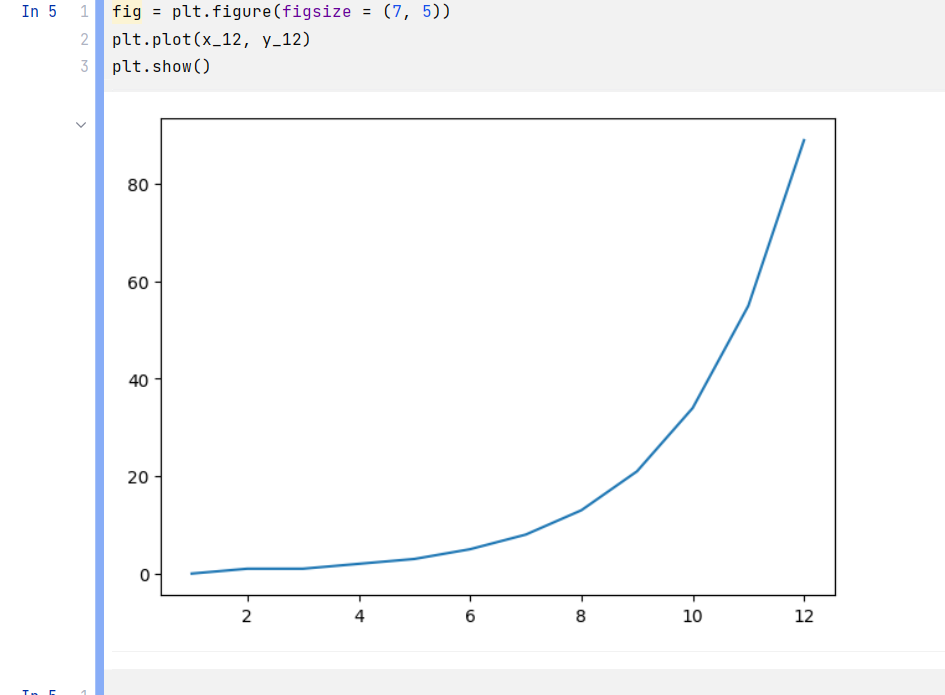
****

**Созданный Jupiter-notebook:**

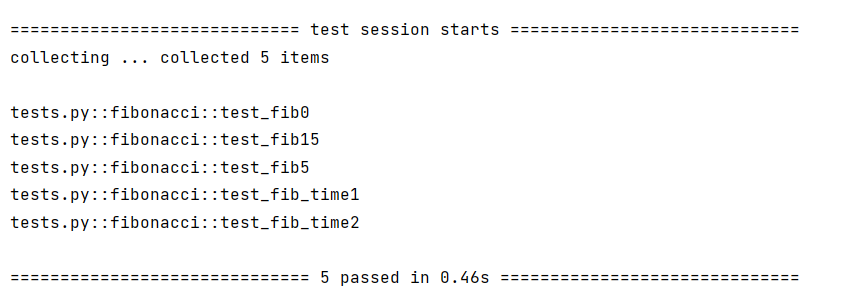
****

****

****

****

**TDD-тестирование используемого генератора:**

****