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:
студент группы ИУ5-35Б
Самойлов Константин
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Нардид А.Н.
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. 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.

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()
```

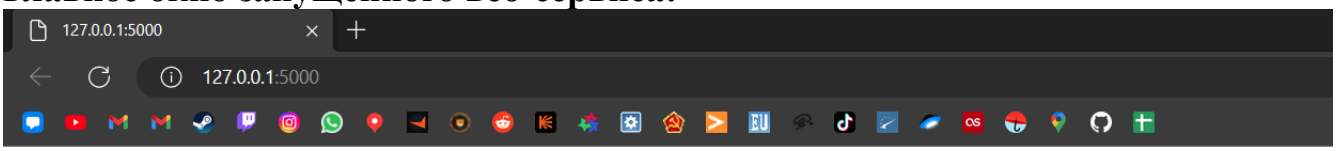
flask1.py

```
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)
```

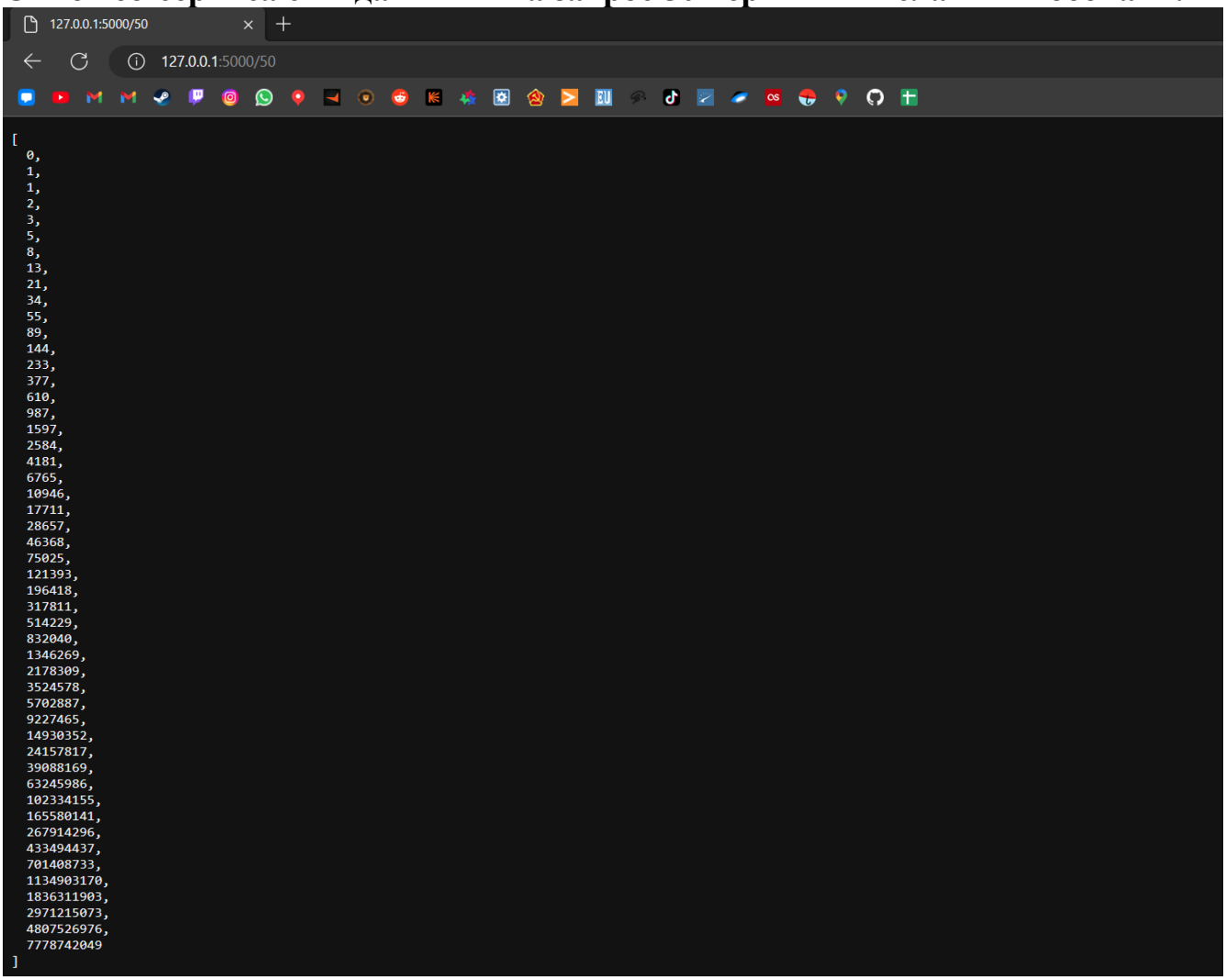
Экранные формы

Главное окно запущенного веб-сервиса:



Fibonacci function

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




```
[
  0,
  1,
  1,
  2,
  3,
  5,
  8,
  13,
  21,
  34,
  55,
  89,
  144,
  233,
  377,
  610,
  987,
  1597,
  2584,
  4181,
  6765,
  10946,
  17711,
  28657,
  46368,
  75025,
  121393,
  196418,
  317811,
  514229,
  832040,
  1346269,
  2178309,
  3524578,
  5702887,
  9227465,
  14930352,
  24157817,
  39088169,
  63245986,
  102334155,
  165580141,
  267914296,
  433494437,
  701408733,
  1134903170,
  1836311903,
  2971215073,
  4807526976,
  7778742049
]
```

Запуск веб-сервиса из терминала:

```
Run
C:\Users\kroll\PycharmProjects\BKIT\dz\venv\Scripts\python.exe C:\Users\kroll\PycharmProjects\BKIT\dz\flask1.py
 * Serving Flask app 'flask1'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 971-263-986
127.0.0.1 - - [29/Dec/2022 12:58:26] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:27] "GET /favicon.ico HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:27] "GET /20 HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:27] "GET /0 HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:27] "GET /5 HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:27] "GET /10 HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:27] "GET /12 HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:27] "GET /15 HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:27] "GET /20 HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:27] "GET /12 HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:45] "GET /50 HTTP/1.1" 200 -
127.0.0.1 - - [29/Dec/2022 12:58:45] "GET /favicon.ico HTTP/1.1" 200 -
```

Созданный Jupiter-notebook:

In 1
```
import requests
import matplotlib.pyplot as plt

url = 'http://127.0.0.1:5000/20'
r = requests.get(url)
r
```

Out 1
```
<Response [200]>
```

In 2
```
data = r.json()
print(data, end='', flush=False)
type(data)
```

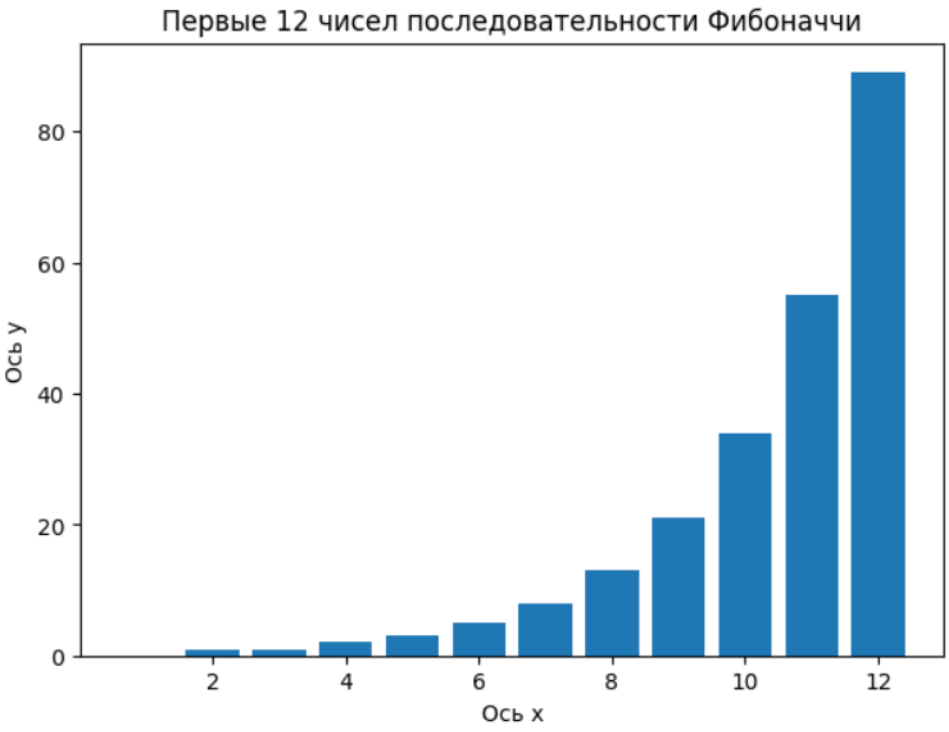
```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181]
```

Out 2
```
list
```

In 3
```
def makeURL(cnt):
    url = 'http://127.0.0.1:5000/'
    res = url + str(cnt)
    return res

def get_data(cnt):
    url = makeURL(cnt)
    r = requests.get(url)
    return r.json()
cntLst = [0, 5, 10, 12, 15, 20]
for elem in cntLst:
    print(f'{elem} первых чисел последовательности Фибоначчи: {get_data(elem)}')
```

```
0 первых чисел последовательности Фибоначчи: []
5 первых чисел последовательности Фибоначчи: [0, 1, 1, 2, 3]
10 первых чисел последовательности Фибоначчи: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
12 первых чисел последовательности Фибоначчи: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
15 первых чисел последовательности Фибоначчи: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377]
20 первых чисел последовательности Фибоначчи: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181]
```

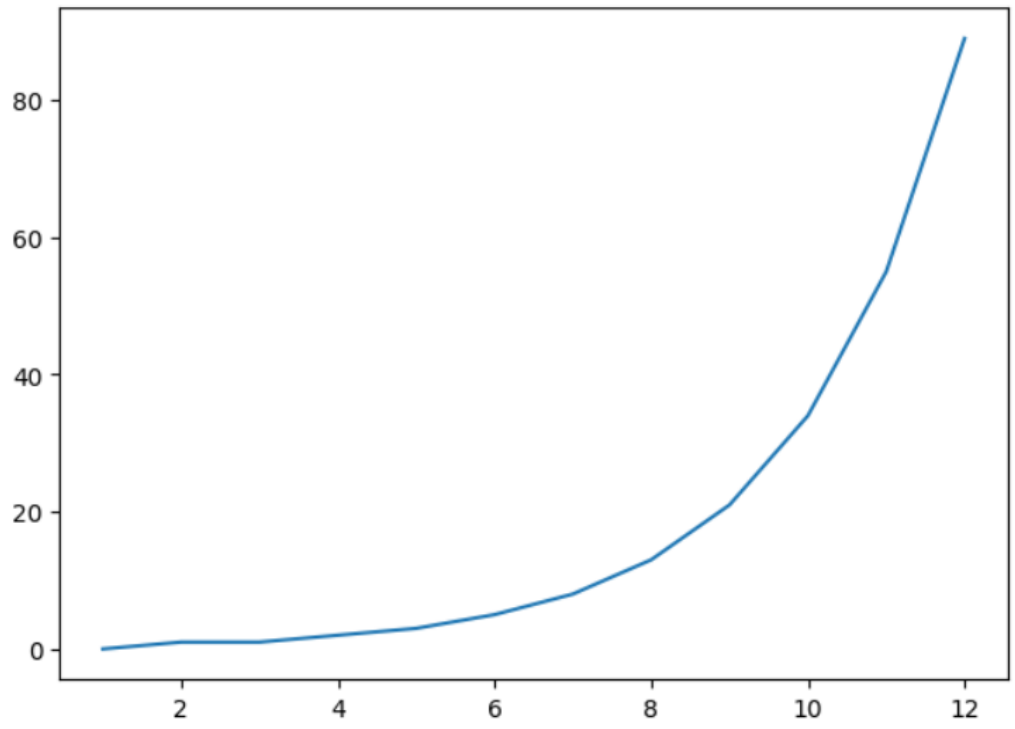
```
y_12 = get_data(12)
x_12 = list(range(1, len(y_12)+1))
fig = plt.figure(figsize = (7, 5))
plt.bar(x_12, y_12)
plt.xlabel('Ось x')
plt.ylabel('Ось y')
plt.title(f'Первые {len(y_12)} чисел последовательности Фибоначчи')
plt.show()
```




```
5  1  fig = plt.figure(figsize = (7, 5))
   2  plt.plot(x_12, y_12)
```

In 5

```
1 fig = plt.figure(figsize = (7, 5))
2 plt.plot(x_12, y_12)
3 plt.show()
```



TDD-тестирование используемого генератора:

```
============================= test session starts ==============================
collecting ... collected 5 items

tests.py::fibonacci::test_fib0
tests.py::fibonacci::test_fib15
tests.py::fibonacci::test_fib5
tests.py::fibonacci::test_fib_time1
tests.py::fibonacci::test_fib_time2

============================== 5 passed in 0.46s ===============================
```