Московский государственный технический
Университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»
Отчет по домашнему заданию

Выполнила:
студентка группы
ИУ5-31Б
Гапеева Олеся

Проверил:
преподаватель
каф. ИУ5
Гапанюк Ю.Е.

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.

Текст программы

fib.py

```
def fib(n):
    a, b = 0, 1
    for i in range(n):
        yield a
        a, b = b, a + b
```

tests.py

```
import unittest
from fib 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(0.5, end_time)

if __name__ == '__main__':
    unittest.main()
```

fl.py

```
from flask import Flask
from fib 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)
```

juipiter.ipynb

```
import requests
import json
import matplotlib.pyplot as plt

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

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

def make_url(cnt):
    base_url = 'http://127.0.0.1:5000/'
    res = base_url + str(cnt)
```

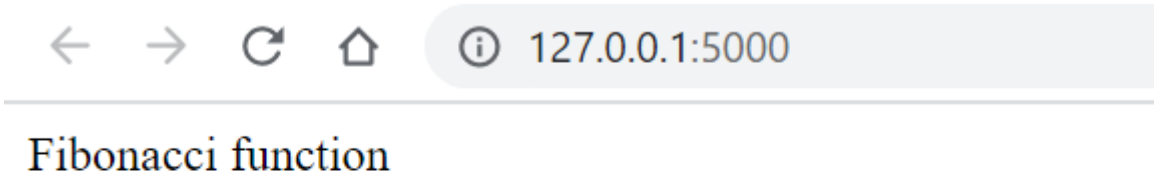
```
    return res

def get_data(cnt):
    url = make_url(cnt)
    r = requests.get(url)
    return r.json()


cnt_list = [0, 5, 10, 12, 15, 20]
for cnt in cnt_list:
    print('{} первых чисел последовательности Фибоначчи: {}'.format(cnt,
get_data(cnt)))


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 абсцисс')
plt.ylabel('Ось ординат')
plt.title('Первые {} чисел последовательности Фибоначчи'.format(len(y_12)))
plt.show()


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

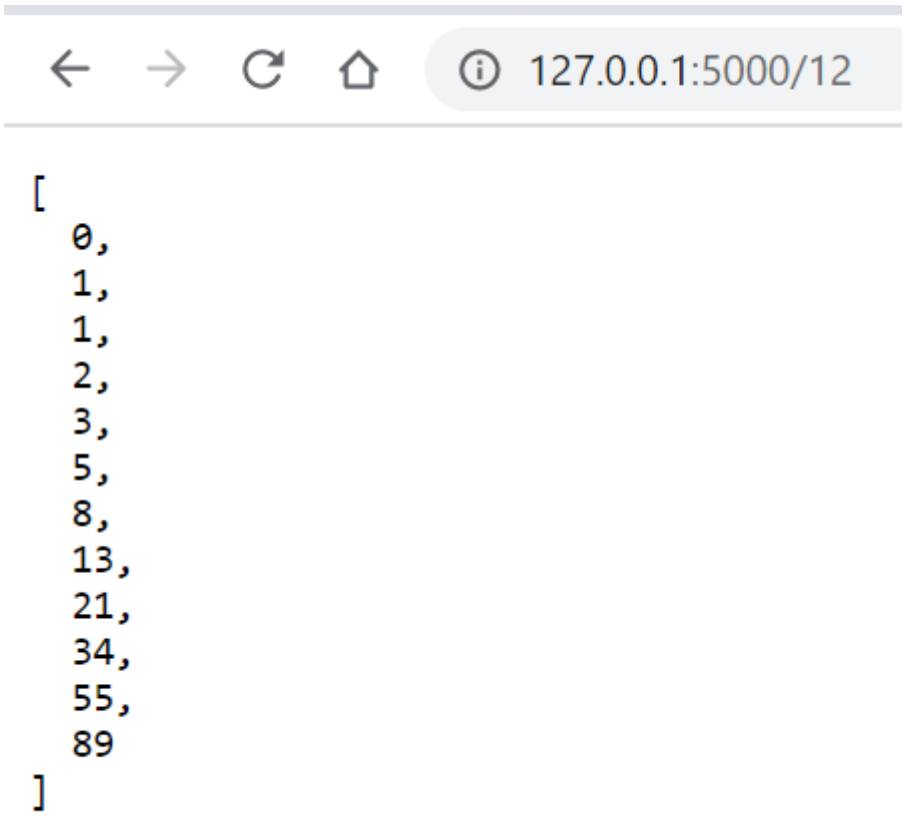
Экранные формы с примерами выполнения программы

tests.py

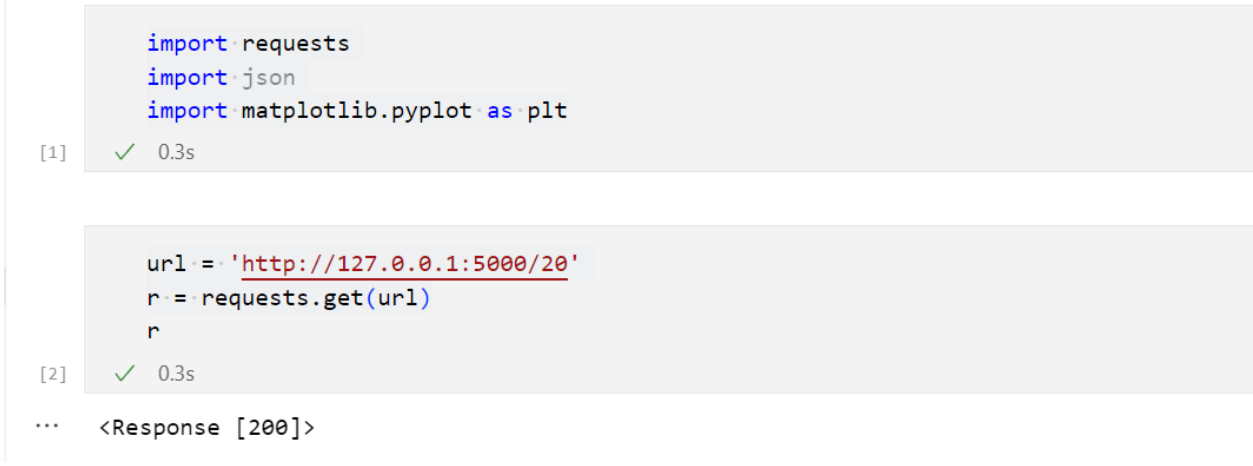
```
(env) PS C:\Users\olgap\Documents\BKIT\dz> & c:/Users/olgap/Documents/BKIT/dz/env
lgap/Documents/BKIT/dz/tests.py
....F
======================================================================
FAIL: test_fib_time2 (__main__.fibonacci)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "c:\Users\olgap\Documents\BKIT\dz\tests.py", line 28, in test_fib_time2
    self.assertLess(0.5, end_time)
AssertionError: 0.5 not less than 0.1950981616973877

----------------------------------------------------------------------
Ran 5 tests in 0.245s

FAILED (failures=1)
```

fl.py





juipiter.ipynb



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

[3] ✓ 0.3s

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

list
```

```
def make_url(cnt):
    base_url = 'http://127.0.0.1:5000/'
    res = base_url + str(cnt)
    return res

def get_data(cnt):
    url = make_url(cnt)
    r = requests.get(url)
    return r.json()
```

[4] ✓ 0.3s

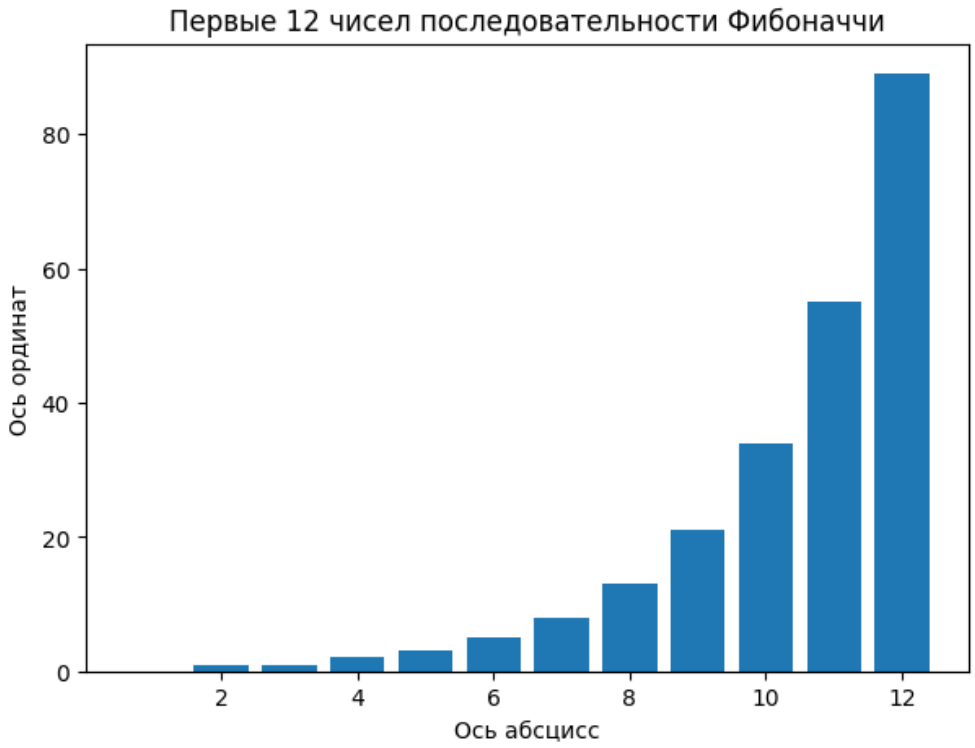
```
cnt_list = [0, 5, 10, 12, 15, 20]
for cnt in cnt_list:
    print('{} первых чисел последовательности Фибоначчи: {}'.format(cnt, get_data(cnt)))
```

[5] ✓ 0.4s Python

```
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 абсцисс')
plt.ylabel('Ось ординат')
plt.title('Первые {} чисел последовательности Фибоначчи'.format(len(y_12)))
plt.show()
```

[6] ✓ 0.1s




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

[7] ✓ 0.9s