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:

DZ числа Фибоначчи

In [1]:
```
import requests
import json
r = requests.get('http://localhost:5000/10').json()
print(r)
```

```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

In [2]:
```
# Функции для выполнения запроса к сервису

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()
```

In [3]:
```
cnt_list = [5, 10, 15, 20]
for cnt in cnt_list:
    print('{} первых чисел последовательности Фибоначчи: {}'.format(cnt, get_data(cnt)))
```

```
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]
15 первых чисел последовательности Фибоначчи: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377]
```

Построение графиков

In [4]:
```
# Данные для построения
y_10 = get_data(10)
x_10 = list(range(1, len(y_10)+1))
```

In [7]:
```
!pip install matplotlib
import numpy as np
from matplotlib import pyplot as plt

plt.bar(x_10, y_10)
plt.xlabel('Ось абсцисс')
plt.ylabel('Ось ординат')
plt.title('Первые {} чисел последовательности Фибоначчи'.format(len(y_10)))
plt.show()
```