Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Основы информатики»

Отчет по домашнему заданию

«Разработка комплексного приложения на языке Python»

Выполнил:
студент группы ИУ5-35Б
Кулешова Ирина
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Нардид А. Н.
Подпись и дата:

Москва, 2022 г.

1. Описание задания:

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.

2. Текст программы:

main.py

```
import math
import start

def createF(n):
    fi = (1 + math.sqrt(5)) / 2
    Fn = int((pow(fi, n) - pow(0 - fi, 0 - n)) / (2 * fi - 1))
    return Fn

def use_generator(n):
    if n > 0:
        return (createF(x) for x in range(n))
    elif n == 0:
        return [0]
    else:
        return (createF(x) for x in range(0, n, -1))

def main():
    n = int(input())
    #print(list(use_generator(n)))
    app.run(n)
```

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

tdd-tests.py

```
import unittest
import math
import time
from itertools import islice
from main import createF, use_generator

class TestFib(unittest.TestCase):
    def test_createF(self):
        self.assertEqual(createF(0), 0)
        self.assertEqual(createF(-4), -3)
        self.assertEqual(createF(10), 55)

    def test_use_generator(self):
        self.assertEqual(list(use_generator(0)), [0])
        self.assertEqual(list(use_generator(-10)), [0, 1, -1, 2, -3, 5, -8, 13, -21, 34])
        self.assertEqual(list(use_generator(10)), [0, 1, 1, 2, 3, 5, 8, 13, 21, 34])
        time1 = time.time()
        a = list(islice(use_generator(1000), 500))
        time2 = time.time()
        b = list(islice([createF(x) for x in range(1000)], 500))
        time3 = time.time()
        self.assertLess(time2 - time1, time3 - time2, "Это не ленивые вычисления.")


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

start.py

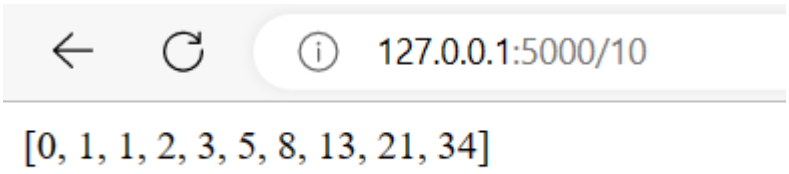
```
from flask import Flask, render_template
import jupyter
import math

app = Flask(__name__)

def createF(n):
    fi = (1 + math.sqrt(5)) / 2
    Fn = int((pow(fi, n) - pow(0 - fi, 0 - n)) / (2 * fi - 1))
    return Fn
```

```
def use_generator(n):
    if n > 0:
        return (createF(x) for x in range(n))
    elif n == 0:
        return [0]
    else:
        return (createF(x) for x in range(0, n, -1))

@app.route('/<int:n>')
def index(n):
    if n >= 0:
        return str(list(use_generator(n)))

    else:
        return str(list(use_generator(n)))



if __name__ == '__main__':
    app.run(debug=True)
```

jupyter.ipynb

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

r = requests.get('http://localhost:5000/12')
F = json.loads(r.text)
print(F)
x = F
y = F
plt.bar(x, y)
plt.show()
```

3. Экранные формы с примерами выполнения программы:



127.0.0.1:5000/20

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

[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]