**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский государственный технический университет**

**имени Н.Э. Баумана (национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

**Факультет «Информатика и системы управления»**

**Кафедра «Системы обработки информации и управления»**

**Курс «Базовые компоненты интернет-технологий»**

Отчёт по домашнему заданию.

Выполнил:

студент группы ИУ5-33Б

Александр Саргсян

Проверил:

к.т.н., доц., Ю. Е. Гапанюк

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:**

*lazy\_fibonacci.py*

def fibonacci(n):  
 a, b = 0, 1  
 for \_ in range(n):  
 yield a  
 a, b = b, a + b

*fibonacci\_test.py*

import unittest  
from lazy\_fibonacci import fibonacci  
  
class fibonacci\_test(unittest.TestCase):  
 def test\_fib1(self):  
 self.assertEqual(len(list(fibonacci(10))), 10)  
 self.assertEqual(list(fibonacci(10)), [0, 1, 1, 2, 3, 5, 8, 13, 21, 34])  
  
 def test\_fib2(self):  
 res = fibonacci(4)  
 self.assertEqual(next(res), 0)  
 self.assertEqual(next(res), 1)  
 self.assertEqual(next(res), 1)  
 self.assertEqual(next(res), 2)  
  
 def test\_fib3(self):  
 self.assertEqual(len(list(fibonacci(7))), 7)  
 self.assertEqual(list(fibonacci(7)), [0, 1, 1, 2, 3, 5, 8])

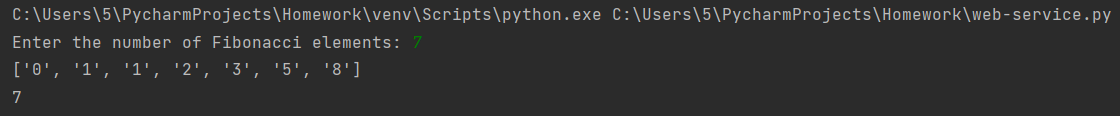
*web-service.py*

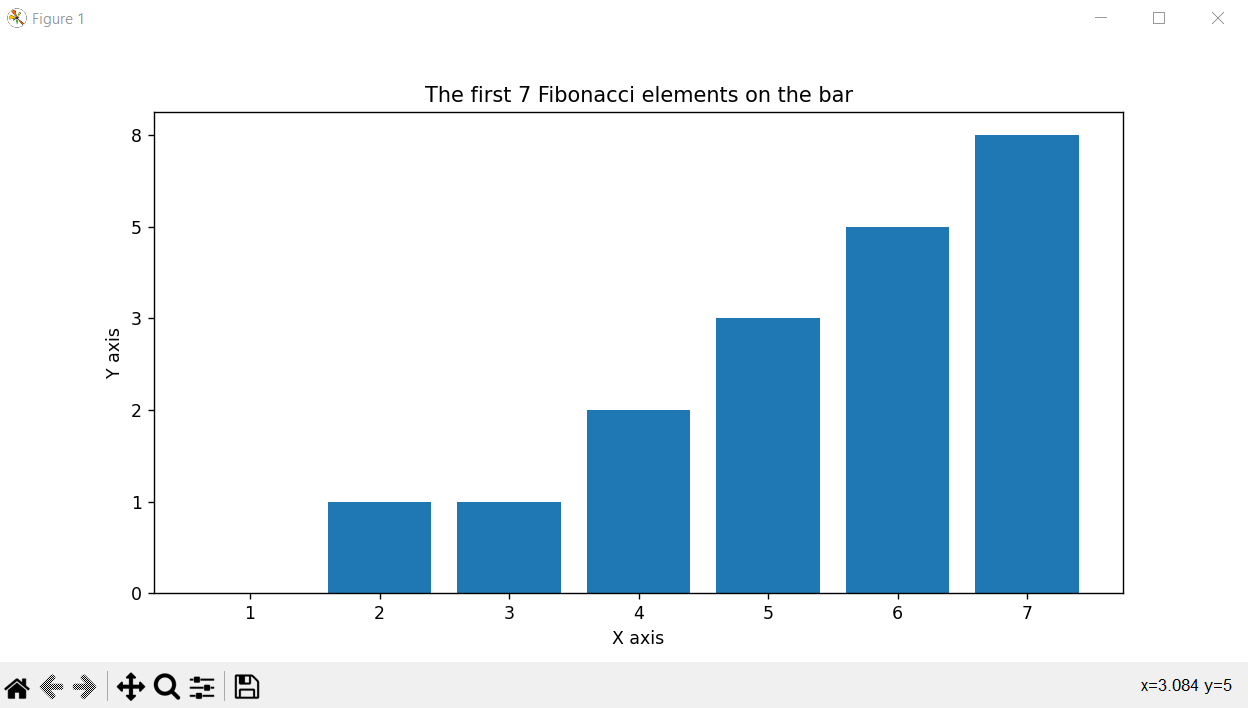
import requests  
import matplotlib.pyplot as plt  
  
def create\_url(main\_url: str, number\_amount: int) -> str:  
 base\_url = main\_url  
 final\_url = f"{base\_url}/num/{number\_amount}"  
 return final\_url  
  
def get\_info(\*, main\_url: str, number\_amount: int) -> str:  
 final\_url = create\_url(main\_url, number\_amount)  
 res = requests.get(final\_url)  
 return res.text  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 url = 'http://127.0.0.1:5000'  
 y = get\_info(main\_url=url, number\_amount=int(input('Enter the number of Fibonacci elements: '))).split(', ')  
 print(y)  
 print(len(y))  
 x = list(range(1, len(y) + 1))  
 fig = plt.figure(figsize=(10, 5))  
 plt.bar(x, y)  
 plt.xlabel('X axis')  
 plt.ylabel('Y axis')  
 plt.title('The first {} Fibonacci elements on the bar'.format(len(y)))  
 plt.show()  
 plt.plot(x, y)  
 plt.scatter(x, y)  
 plt.xlabel('X axis')  
 plt.ylabel('Y axis')  
 plt.title('The first {} Fibonacci elements on the graph'.format(len(y)))  
 plt.show()

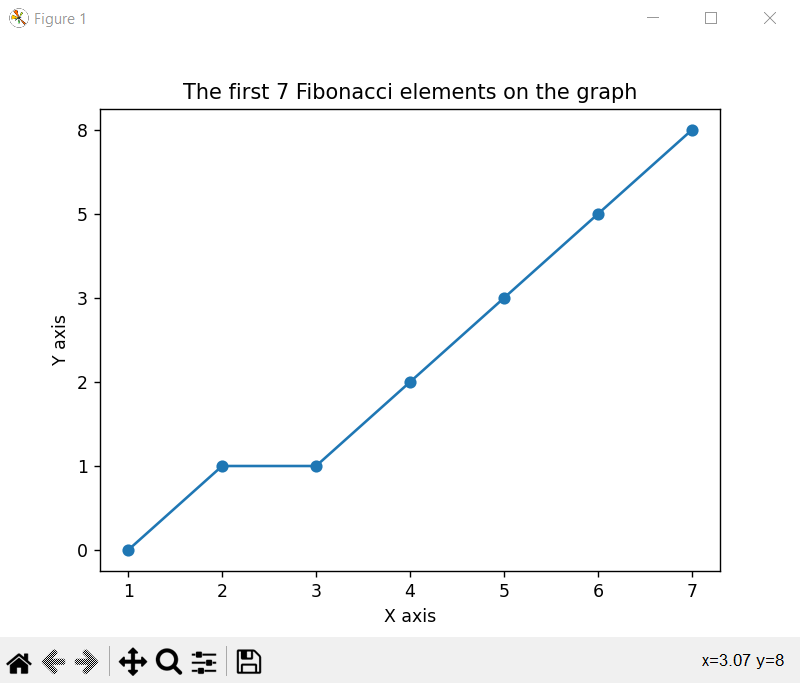
*app.py*

from flask import Flask  
from lazy\_fibonacci import fibonacci  
  
app = Flask(\_\_name\_\_)  
@app.route('/num/<number>')  
def num(number):  
 return str(list(fibonacci(int(number))))  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app.run()

**Пример выполнения программы:**

****

****

****