**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

«разработка комплексного приложения на языке Python»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-34Б |  | преподаватель каф. ИУ5 |
| Угрюмов Михаил |  | Гапанюк Ю. Е. |
|  |  |  |

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.**

Padovan.py

*''' Числа Падована:  
 P(0) = P(1) = P(2) = 1  
 P(n) = P(n - 2) + P(n - 3) '''*def Padovan(n):  
 p = [1, 1, 1]  
 for i in range(min(3, n + 1)):  
 yield p[i]  
 for i in range(3, n + 1):  
 next = p[0] + p[1]  
 p[0] = p[1]  
 p[1] = p[2]  
 p[2] = next  
 yield next

TestPadovan.py

import sys  
import unittest as ut  
from Padovan import Padovan  
  
class TestPadovan(ut.TestCase):  
 def setUp(self):  
 self.answer\_15 = [1, 1, 1, 2, 2, 3, 4, 5, 7, 9, 12, 16, 21, 28, 37, 49]  
 self.alt\_15 = [1, 1, 1, 2, 2]  
 self.answer\_2 = [1, 1, 1]  
 for i in range(5, 16):  
 self.alt\_15.append(self.alt\_15[i - 1] + self.alt\_15[i - 5])  
  
 def test\_Padovan\_2(self):  
 *''' P(n) = P(n - 2) + P(n - 3) '''* self.assertEqual(list(Padovan(2)), self.answer\_2)  
  
 def test\_Padovan\_15(self):  
 *''' P(n) = P(n - 2) + P(n - 3) '''* self.assertEqual(list(Padovan(15)), self.answer\_15)  
  
 def test\_alternative\_recurrent\_relations(self):  
 *''' P(n) = P(n - 1) + P(n - 5) '''* self.assertEqual(list(Padovan(15)), self.alt\_15)  
  
 def test\_lazy\_calculations(self):  
 self.assertEqual(sys.getsizeof(Padovan(10)), sys.getsizeof(Padovan(10000)))  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 ut.main()

WebPadovan.py

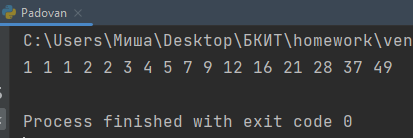
from flask import Flask, request  
from Padovan import Padovan  
  
app = Flask(\_\_name\_\_)  
  
@app.route('/padovan', methods=['GET', 'POST'])  
def form\_example():  
 if request.method == 'POST':  
 n = int(request.form.get('N'))  
 return '''  
 <h1>Num is: {}</h1>  
 <h2>First {} Padovan`s numbers are:</h2>  
 <h3>{}</h3>'''.format(n, n, ', '.join([str(elem) for elem in Padovan(n)]))  
  
 return '''  
 <form method="POST">  
 <div><label>Input N: <input type="text" name="N"></label></div>  
 <input type="submit" value="Submit">  
 </form>'''  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app.run(debug=True)

JupyterGraphic.ipynb

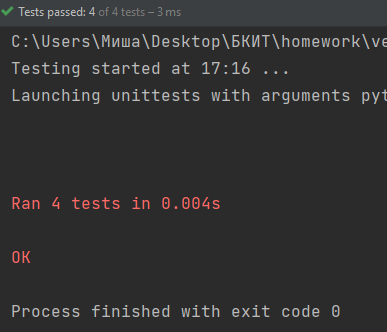
import requests  
import matplotlib.pyplot as plt  
  
def graphic(content, start, finish, n):  
 x = [i for i in range(0, n + 1, 1)]  
 y = [int(elem) for elem in content[start: finish].split(', ')]  
  
 plt.figure(figsize=(12, 8))  
 plt.title('First {} Padovan`s numbers'.format(n))  
 plt.xlabel('index')  
 plt.ylabel('Padovan`s numbers')  
 plt.plot(x, y, 'ro', linewidth=2, color='r')  
 plt.grid(which='major', color='k', linewidth=1)  
 plt.show()  
  
def query(n):  
 url = 'http://127.0.0.1:5000/padovan'  
  
 param = {'N': str(n)}  
 with requests.Session() as s:  
 p = s.post(url, data=param)  
  
 content = str(p.content)  
 start = content.find('<h3>') + 4  
 finish = content.find('</h3>')  
  
 graphic(content, start, finish, n)  
  
def main():  
 n = int(input('Введите n: '))  
 query(n)  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

**Результаты выполнения программы:**

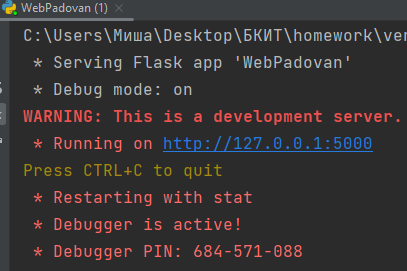
Если в файле Padovan.py дописать в конце print(\*Padovan(15)), получится:

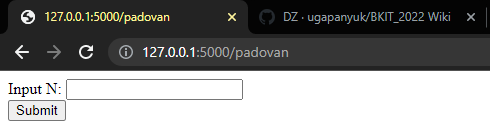


Файл TestPadovan.py:

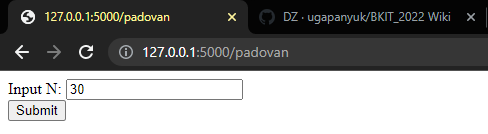


Запуск WebPadovan.py:

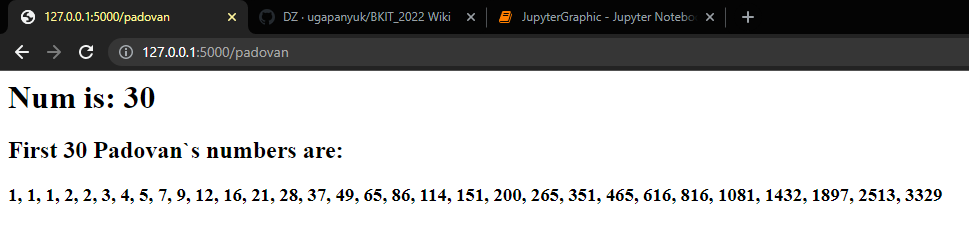




Введём 30:



Нажмём submit:



Построение графика в Jupyter notebook (файл JupyterGraphic.ipynb):

