**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»**

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

## Отчёт по домашнему заданию

Выполнил:

студент группы ИУ5-33Б Кузнецов В. А.

подпись: , дата:

Проверил:

лектор Гапанюк Ю. Е.

подпись: , дата:

2022 г.

**Задание:**

1. С использованием механизма итераторов или генераторов реализуйте с помощью концепции ленивых вычислений [одну из последовательностей OEIS.](https://ru.wikipedia.org/wiki/%D0%9E%D0%BD%D0%BB%D0%B0%D0%B9%D0%BD-%D1%8D%D0%BD%D1%86%D0%B8%D0%BA%D0%BB%D0%BE%D0%BF%D0%B5%D0%B4%D0%B8%D1%8F_%D1%86%D0%B5%D0%BB%D0%BE%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D1%85_%D0%BF%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D0%B5%D0%B9) Примером могут являться [числа Фибоначчи.](https://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%BB%D0%B0_%D0%A4%D0%B8%D0%B1%D0%BE%D0%BD%D0%B0%D1%87%D1%87%D0%B8)
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](https://requests.readthedocs.io/en/latest/) и визуализацию полученных от веб-сервиса данных с использованием библиотеки [matplotlib.](https://matplotlib.org/)

**Текст программы**

lucky\_numbers.py

**from** itertools **import** count, islice  
  
  
**def** lucky\_numbers():  
 *"""  
 Генератор счастливых чисел\n  
 https://oeis.org/A000959  
 """* **yield** 1  
 sequence = []  
 **for** i **in** count(3, 2):  
 **for** divider **in** sequence:  
 divider[1] += 1  
 **if** divider[1] % divider[0] == 0:  
 **break  
 else**:  
 **yield** i  
 sequence.append([i, len(sequence) + 2])

tests.py

**import** unittest  
**from** types **import** GeneratorType  
**from** itertools **import** islice  
**from** lucky\_numbers **import** lucky\_numbers  
  
  
**class** TestLuckyNumbers(unittest.TestCase):  
 **def** test\_is\_generator(self):  
 self.assertIsInstance(lucky\_numbers(), GeneratorType)  
  
 **def** test\_values(self):  
 result = [1, 3, 7, 9, 13, 15, 21, 25, 31, 33,  
 37, 43, 49, 51, 63, 67, 69, 73, 75, 79,  
 87, 93, 99, 105, 111, 115, 127, 129, 133,  
 135, 141, 151, 159, 163, 169, 171, 189, 193,  
 195, 201, 205, 211, 219, 223, 231, 235, 237,  
 241, 259, 261, 267, 273, 283, 285, 289, 297,  
 303, 307, 319, 321, 327, 331, 339, 349, 357,  
 361, 367, 385, 391, 393, 399, 409, 415, 421,  
 427, 429, 433, 451, 463, 475, 477, 483, 487,  
 489, 495, 511, 517, 519, 529, 535, 537, 541,  
 553, 559, 577, 579, 583, 591, 601, 613, 615,  
 619, 621, 631, 639, 643, 645, 651, 655, 673,  
 679, 685, 693, 699, 717, 723, 727, 729, 735,  
 739, 741, 745, 769, 777]  
 answer = list(islice(lucky\_numbers(), len(result)))  
 self.assertEqual(result, answer)  
  
 **def** test\_first\_ten(self):  
 result = [1, 3, 7, 9, 13, 15, 21, 25, 31, 33]  
 numbers = lucky\_numbers()  
 **for** i **in** range(10):  
 self.assertEqual(result[i], numbers.\_\_next\_\_())  
  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 unittest.main()

**from** flask **import** Flask  
**from** lucky\_numbers **import** lucky\_numbers  
  
  
app = Flask(\_\_name\_\_)  
  
  
@app.route(**"/"**)  
**def** hello\_world():  
 html = **"""<h1>Returning the lucky numbers!</h1>  
 <a href="https://oeis.org/A000959">**

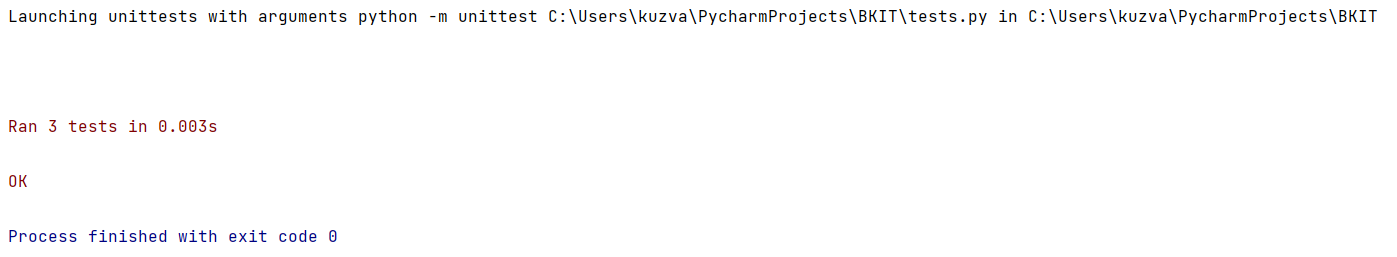
**The sequence in the OEIS<br>**

**</a>  
 <a href="https://ru.wikipedia.org/wiki/Счастливое\_число\_(lucky\_number)">**

**Wikipedia**

**</a>"""  
 return** html  
  
  
@app.route(**'/num/<int:cnt>'**)  
**def** get\_fib(cnt):  
 numbers = lucky\_numbers()  
 res = [next(numbers) **for** \_ **in** range(cnt)]  
 **return** res

**Пример выполнения**

****

