

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

Факультет «Радиотехнический»

Кафедра ИУ5. Курс «Базовые компоненты интернет-технологий»

Отчет по Домашнему заданию
« Разработка комплексного приложения на языке Python »

Выполнил:

студент группы РТ5-31Б
Фруктин А.Е.

Проверил:

преподаватель каф. ИУ5
Гапанюк Ю.Е.

дата: 21.12.2022

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.

Задание 1

Файл cpn.py

```
def cpn_gen():
    num, prev = 1, 1
    while True:
        yield prev
        prev = prev + num
        num += 1
# gen = cpn_gen()
# a = [next(gen) for _ in range(6)]
# print(a)
```

Задание 2

Файл test.py

```
import unittest
from cpn import cpn_gen
from collections.abc import Generator
from functools import reduce

class TestCPN(unittest.TestCase):
    def test_generator(self):
        sequence = cpn_gen()
        self.assertIsInstance(sequence, Generator)

        self.assertEqual(next(sequence), 1)
        self.assertEqual(next(sequence), 2)
        self.assertEqual(next(sequence), 4)

    def test_sequence(self):
        gen = cpn_gen()
```

```

sequence = [next(gen) for _ in range(8)]
self.assertEqual(len(sequence), 8)
self.assertEqual(sequence, [1, 2, 4, 7, 11, 16, 22, 29])

exp = [37, 46]
for ind, val in enumerate(gen):
    if ind > 1:
        break
    self.assertEqual(val, exp[ind])

def test_func(self):
    gen = cpn_gen()
    sequence = list(zip(range(6), gen))
    self.assertEqual(len(sequence), 6)
    self.assertEqual(sequence, [(0, 1), (1, 2), (2, 4), (3, 7), (4, 11),
(5, 16)])

    sequence = list(zip(range(6), gen))
    self.assertEqual(len(sequence), 6)
    self.assertEqual(sequence, [(0, 22), (1, 29), (2, 37), (3, 46), (4,
56), (5, 67)])

if __name__ == "__main__":
    unittest.main()
# результат выполнения программы
[Running] python -u "c:\BKIT\homework\tempCodeRunnerFile.py"
...
-----
Ran 3 tests in 0.001s

OK

[Done] exited with code=0 in 0.608 seconds

```

Задание 3

Файл web.py

```

from flask import Flask
from cpn import cpn_gen

app = Flask(__name__)

@app.route("/") # декоратор для декораций, начинается с собачки
def main_page():
    return "<h3> central polygonal numbers sequence generator</h3>"

```

```
@app.route("/cpn/<int:num>")
def get_cpn(num):
    cpn = cpn_gen()
    return [next(cpn) for _ in range(num)]

# if __name__ == "__main__":
#     app.run(host="127.0.0.1", port=5000)
```

Результат выполнения

(venv) PS C:\BKIT\homework> flask --app web run

* Serving Flask app 'web'

* Debug mode: off

WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.

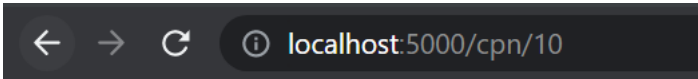
* Running on <http://127.0.0.1:5000>

Press CTRL+C to quit

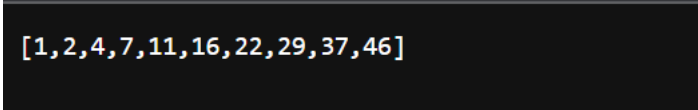


← → ↻ ⓘ localhost:5000

central polygonal numbers sequence generator



← → ↻ ⓘ localhost:5000/cpn/10



[1, 2, 4, 7, 11, 16, 22, 29, 37, 46]

Задание 4

Результат выполнения

(venv) PS C:\BKIT> jupyter notebook

[I 00:38:23.071 NotebookApp] Serving notebooks from local directory:
C:\BKIT

[I 00:38:23.071 NotebookApp] Jupyter Notebook 6.5.2 is running at:

[I 00:38:23.071 NotebookApp]

<http://localhost:8888/?token=9154c45a6e14957617d6e0603a6cad918ce3f0a65c1838ed>

[I 00:38:23.071 NotebookApp] or

<http://127.0.0.1:8888/?token=9154c45a6e14957617d6e0603a6cad918ce3f0a65c1838ed>

[I 00:38:23.072 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).

[C 00:38:23.171 NotebookApp]

To access the notebook, open this file in a browser:

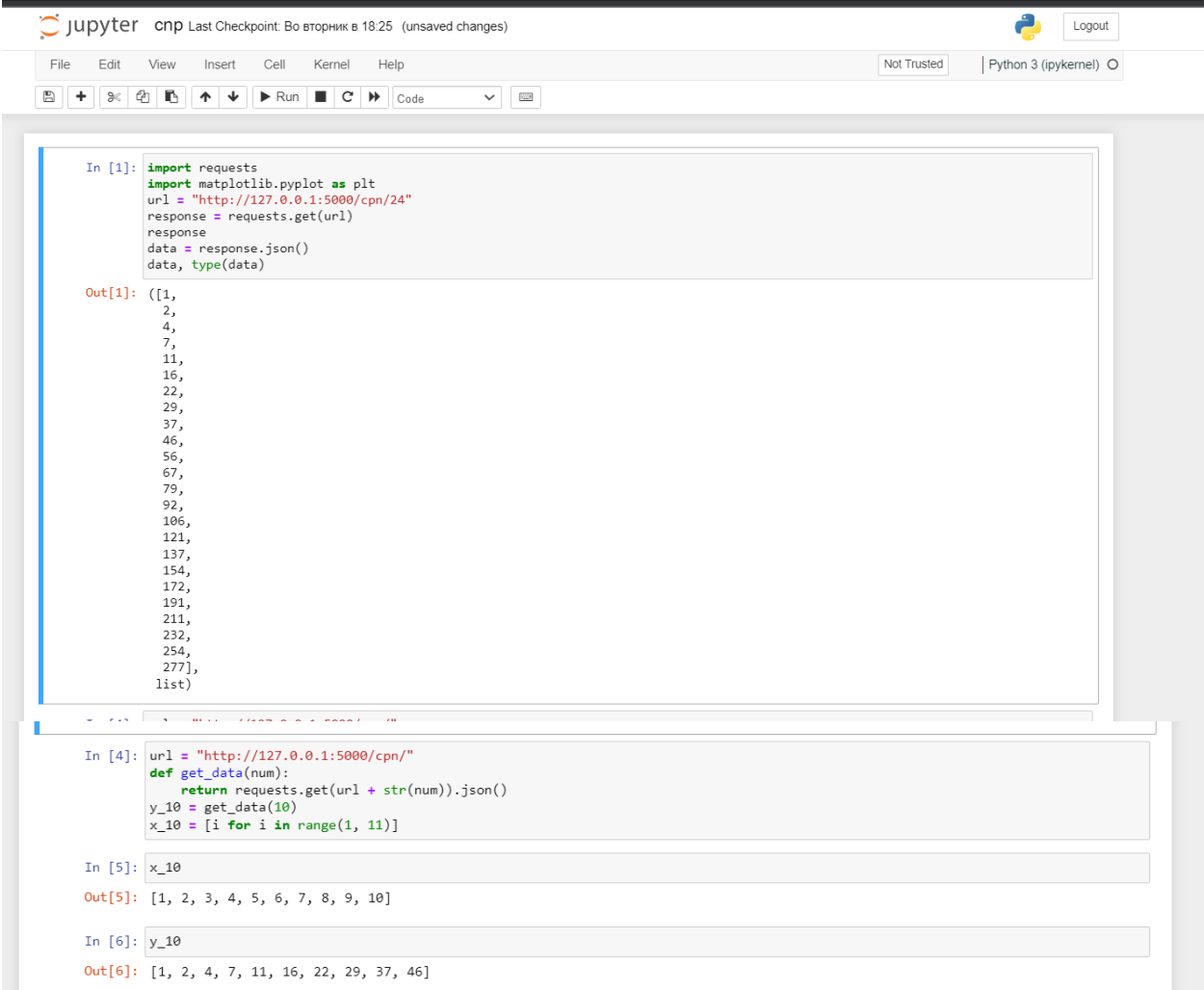
file:///C:/Users/%D0%90%D0%B4%D0%BC%D0%B8%D0%BD/AppData/Roaming/jupyter/runtime/nbserver-25856-open.html

Or copy and paste one of these URLs:

<http://localhost:8888/?token=9154c45a6e14957617d6e0603a6cad918ce3f0a65c1838ed>

or

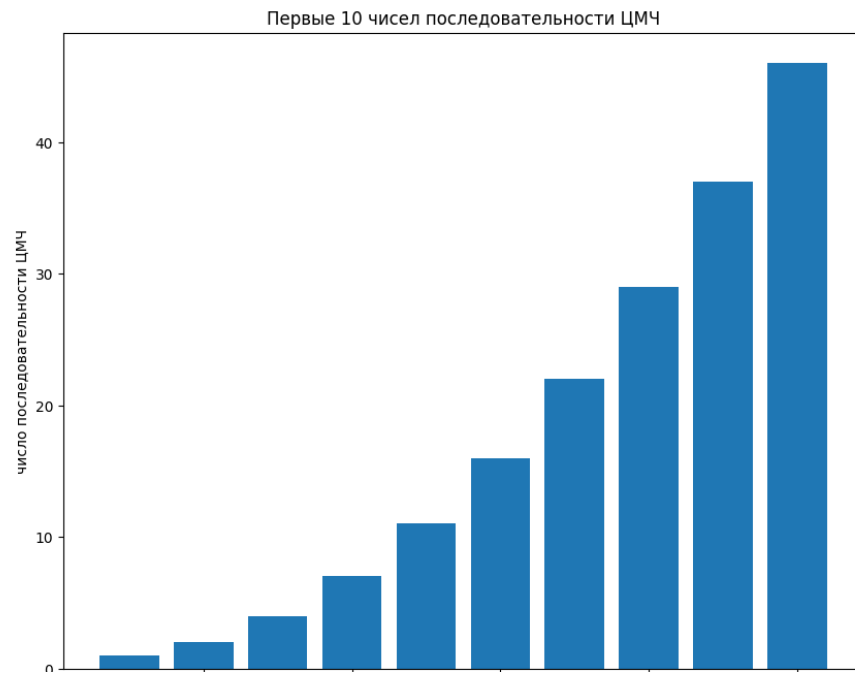
<http://127.0.0.1:8888/?token=9154c45a6e14957617d6e0603a6cad918ce3f0a65c1838ed>



The screenshot displays a Jupyter Notebook interface. At the top, the header shows the Jupyter logo, the name 'cnp', and the last checkpoint information: 'Last Checkpoint: Во вторник в 18:25 (unsaved changes)'. There is a 'Logout' button and a 'Not Trusted' warning. The menu bar includes 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', and 'Help'. Below the menu is a toolbar with icons for file operations, running, and code execution. The notebook contains several code cells:

- In [1]:** A code cell that imports 'requests' and 'matplotlib.pyplot as plt', sets a URL to 'http://127.0.0.1:5000/cpn/24', makes a GET request, and prints the response data. The output shows a list of numbers: [1, 2, 4, 7, 11, 16, 22, 29, 37, 46, 56, 67, 79, 92, 106, 121, 137, 154, 172, 191, 211, 232, 254, 277].
- In [4]:** A code cell that defines a function 'get_data(num)' which returns the JSON response from a GET request to 'http://127.0.0.1:5000/cpn/' + str(num). It then calls 'get_data(10)' and creates a list 'x_10' containing values from 1 to 11.
- In [5]:** A code cell that prints 'x_10', resulting in the output: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].
- In [6]:** A code cell that prints 'y_10', resulting in the output: [1, 2, 4, 7, 11, 16, 22, 29, 37, 46].

```
In [10]: fig = plt.figure(figsize = (10, 8))
plt.bar(x_10, y_10)
plt.xlabel('порядковый номер')
plt.ylabel('число последовательности ЦМЧ')
plt.title('Первые {} чисел последовательности ЦМЧ'.format(len(y_10)))
plt.show()
```



```
In [11]: fig = plt.figure(figsize = (10, 8))
plt.plot(x_10, y_10)
plt.show()
```

