МГТУ им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления»

Домашнее задание

«Базовые компоненты интернет-технологий»

Студент группы ИУ5-31Б: Преподаватель кафедры ИУ5:

Строганов Георгий Константинович Гапанюк Юрий Евгеньевич

Москва, 2022

**Условия ДЗ**

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.

**Листинг программы**

**Flask\_web.py**

from flask import Flask

from generator import fib

app = Flask(\_\_name\_\_)

@app.route("/")

def hello\_world():

return "<p>Hello, World!</p>"

@app.route("/fibonachi")

def fibonachi\_start():

return "Write after URL '/number!'"

@app.route("/fibonachi/<int:n>")

def fibonachi\_number(n):

return list(fib(n))

@app.errorhandler(404)

def page\_not\_found(e):

return "Oops! Try to enter a '/fibonachi/number!'"

**generator.py**

def fib(n):

a, b = 0, 1

for i in range(n):

yield a

a, b = b, a + b

**test\_TDD.py**

# -\*- coding: cp1251 -\*-

import pytest

from time import time

from generator import fib

class cm\_timer:

def \_\_enter\_\_(self):

self.\_\_time\_begin = time()

def \_\_exit\_\_(self, type, value, traceback):

print(time() - self.\_\_time\_begin)

c = 100000

# тестирование результата выполнения

def test\_fib\_1():

assert [i for i in fib(5)] == [0, 1, 1, 2, 3]

def test\_fib\_2():

assert [i for i in fib(10)] == [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]

def test\_fib\_3():

assert [i for i in fib(0)] == []

# тестирование на ленивые выражения (запускать не через pytest чтобы увидеть время выполнения)

def test\_fib\_time\_1():

print("Время выполнения с ленивыми вычислениями")

with cm\_timer():

temp = fib(c)

assert list(fib(c)) == [i for i in fib(c)]

def test\_fib\_time\_2():

print("Время выполнения с обычными вычислениями")

with cm\_timer():

temp = [i for i in fib(c)]

assert [i for i in fib(c)] == list(fib(c))

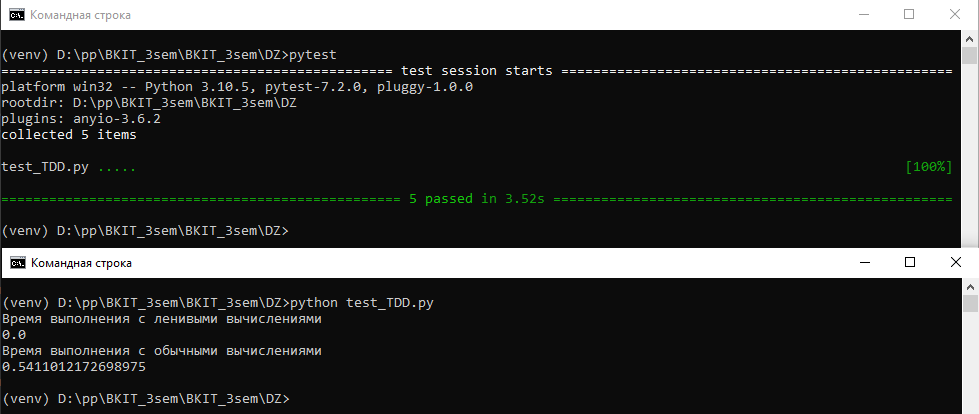
if \_\_name\_\_ == "\_\_main\_\_":

test\_fib\_time\_1()

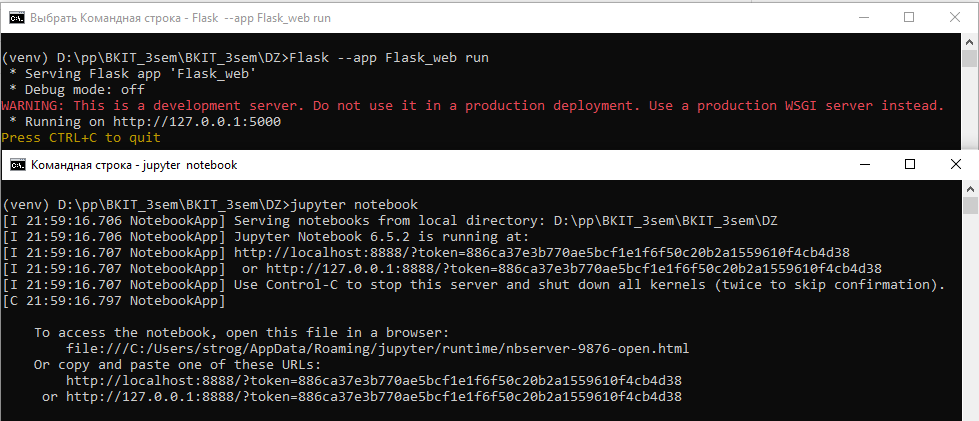
test\_fib\_time\_2()

**Запуск файла с тестированием**

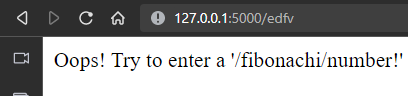
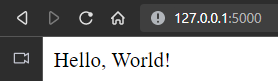
При запуске **test\_TDD.py** через **pytest** производится тестирование на правильность работы файла **generator.py**, а при запуске просто как исполняемый файл выводится информация о скорости вычисления с обычными и с ленивыми вычислениями.

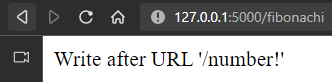


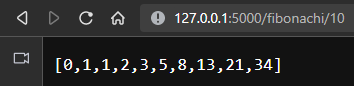
**Запуск Flask и Jupyter Notebook**

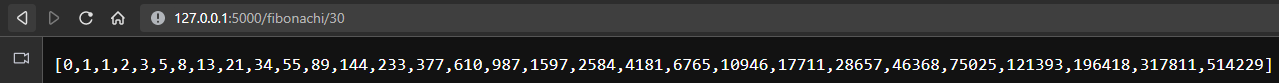


**Переход по http://127.0.0.1:5000/ и т.п.**









**Jupyter Notebook**

Смотреть в разделе: **BKIT\_3sem/DZ/Pylounge.ipynb**