**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»**

**Домашнее задание**

по предмету

«Базовые компоненты интернет-технологий»

Выполнил:

студент группы ИУ5-31Б

Михалёв Ярослав

Проверил:

Преподаватель кафедры ИУ-5

Гапанюк Юрий

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.

**Код программы**

**fib.py**

*def* fibonacci(*n*):  
 *if* type(*n*) *not in* [int]:  
 *raise* TypeError(f"Коэффициент n должен быть натуральным числом!")  
  
 *if n* <= 0:  
 *raise* ValueError("Коэффициент n должен быть натуральным числом!")  
  
 fib1, fib2 = 0, 1  
 *for* i *in* range(*n*):  
 fib1, fib2 = fib2, fib1 + fib2  
 *yield* fib1  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 print(\*fibonacci(10))

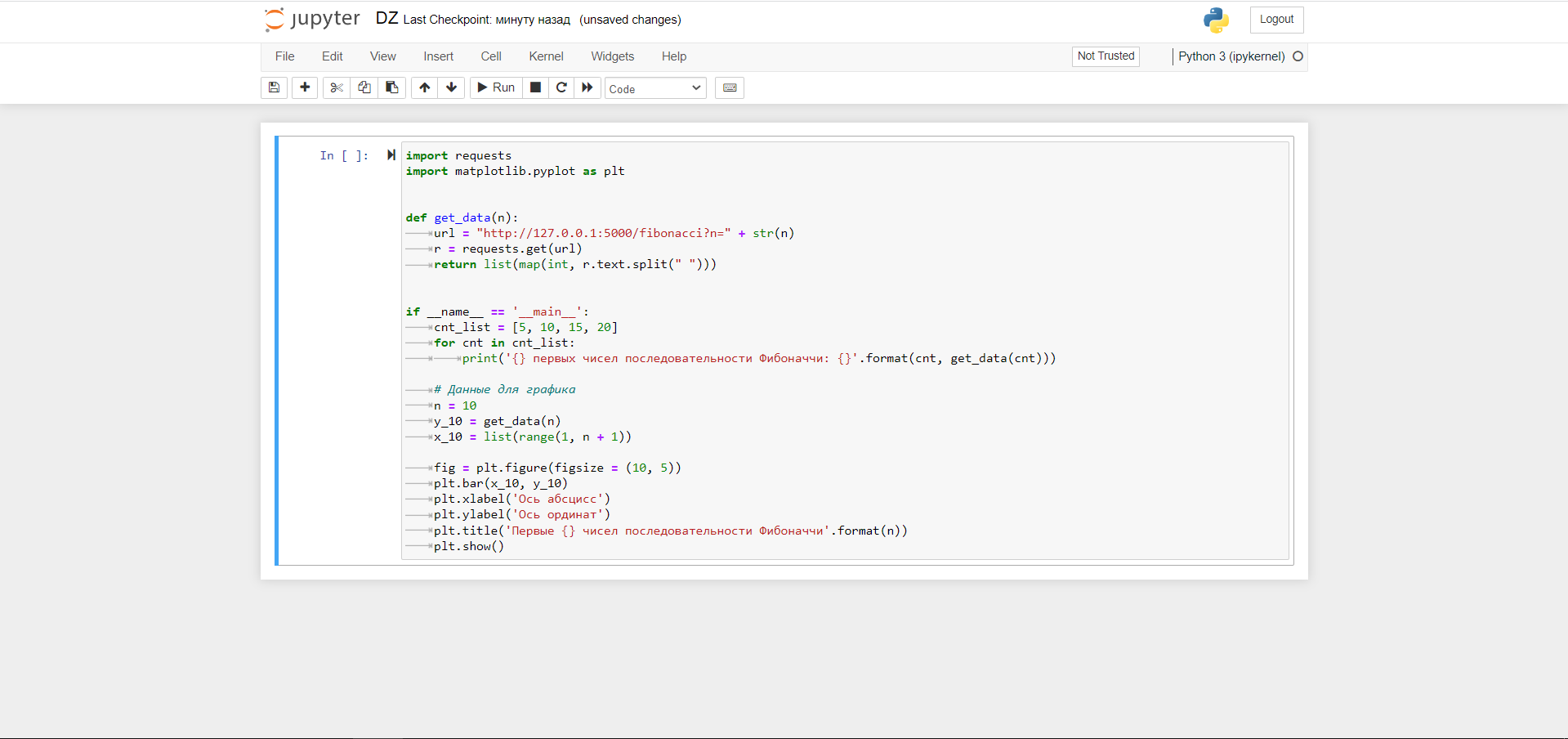
**tests.py**

*import* unittest  
*import* time  
*from* fib *import* fibonacci  
  
  
*class* TestEquation(unittest.TestCase):  
  
 *def* test\_numbers(self):  
 self.assertEqual(len(list(fibonacci(10))), 10)  
 self.assertEqual(list(fibonacci(5)), [1, 1, 2, 3, 5])  
  
 *def* test\_iteration(self):  
 res = fibonacci(2)  
 self.assertEqual(next(res), 1)  
 self.assertEqual(next(res), 1)  
  
 *def* test\_value(self):  
 *with* self.assertRaises(ValueError) *as* e:  
 list(fibonacci(-10))  
  
 *def* test\_type(self):  
 *with* (self.assertRaises(TypeError)) *as* e:  
 list(fibonacci("B"))  
  
 *def* test\_lazy(self):  
 start\_time = time.time()  
 a = fibonacci(1000000)  
 end\_time = time.time() - start\_time  
 self.assertLess(end\_time, 1)  
  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 unittest.main()

**start.py**

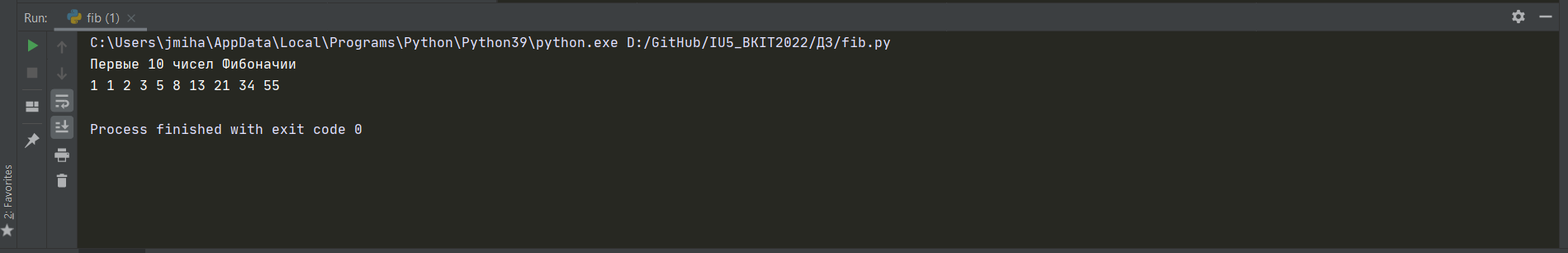
*from* flask *import* Flask, request  
*from* fib *import* fibonacci  
  
  
app = Flask(\_\_name\_\_)  
  
  
@app.route('/')  
*def* hello\_world():  
 *return* "Hello, world"  
  
@app.route('/fibonacci')  
*def* print\_fibonacci1():  
 n = request.args.get("n")  
 *return* " ".join(map(str, fibonacci(int(n))))  
  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 app.run()

**DZ.ipynb**

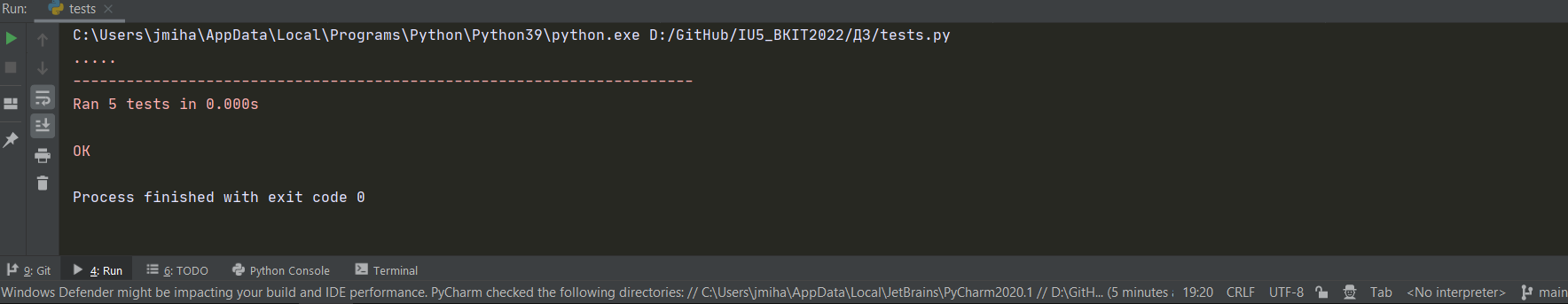


**Результаты выполнения**

**fib.py**



**tests.py**



**DZ.ipynb**

