

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций
Институт цифрового развития**

ОТЧЁТ
по лабораторной работе №3.1
Дисциплина: «Анализ данных»
Тема: «Работа с IPython и Jupyter Notebook»

Выполнил: студент 2 курса
группы ИВТ-б-о-21-1
Кочкаров Умар Ахматович

Ставрополь 2023

Цель работы: исследовать базовые возможности интерактивных оболочек IPython и Jupyter Notebook для языка программирования Python.


Ход работы:

1. Создать общедоступный репозиторий с лицензией MIT и языком программирования Python

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *

 umarkochkarov ▾

Repository name *

/ lb3.1 ✓

Great repository names are short and memorable. Need inspiration? How about [bookish-broccoli?](#)

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.



Add a README file

This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore


Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python ▾

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License ▾

This will set  main as the default branch. Change the default name in your [settings](#).



You are creating a public repository in your personal account.

Create repository

Рисунок 1. Создание репозитория

2. Клонировать репозиторий на ПК:

```
C:\Users\Student>cd C:\Users\Student\Desktop
C:\Users\Student\Desktop>git clone https://github.com/umarkochkarov/lb3.1.git
Cloning into 'lb3.1'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Рисунок 2. Клонирование репозитория на ПК

3. Организовать репозиторий в соответствии с моделью ветвления git-flow:

```
C:\Users\Student\Desktop\lb3.1>git flow init
Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]
How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/Student/Desktop/lb3.1/.git/hooks]
```

Рисунок 3. Организация репозитория в соответствии с git-flow

4. Проработка примеров лабораторной работы:

1)

```
In [1]: 3 + 2
Out[1]: 5

In [2]: a = 5
        b = 7
        print(a + b)
12

In [3]: n = 7
        for i in range(n):
            print(i * 10)
0
10
20
30
40
50
60

In [4]: i = 0
        while True:
            i += 1
            if i > 5:
                break
            print("Test while")
Test while
Test while
Test while
Test while
Test while
```

Рисунок 4. Пример 1

2)

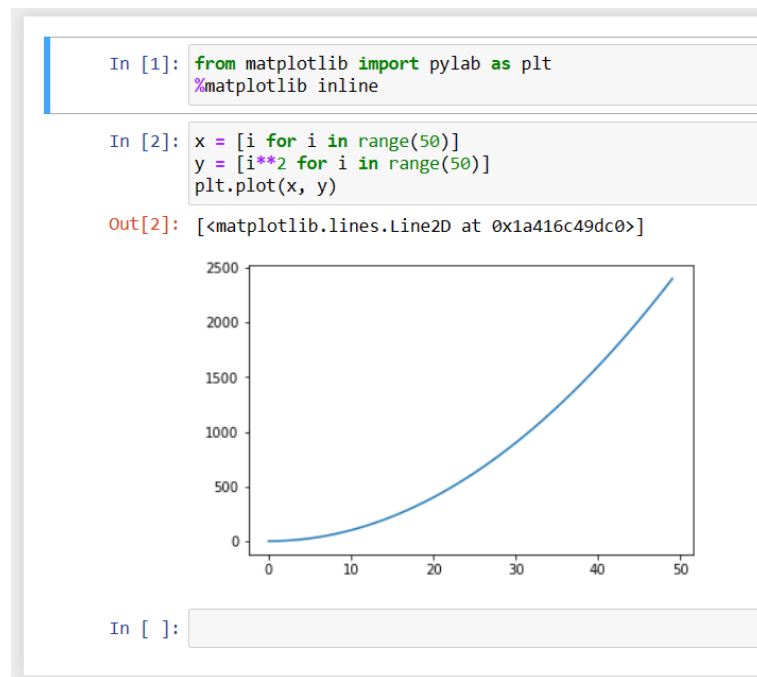


Рисунок 5. Пример 2

3)

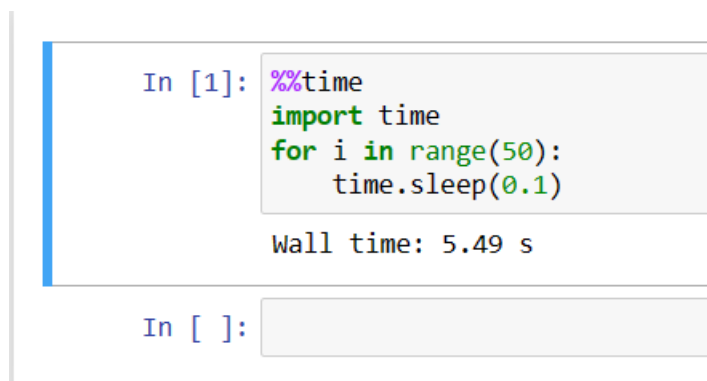


Рисунок 6. Пример 3

4)

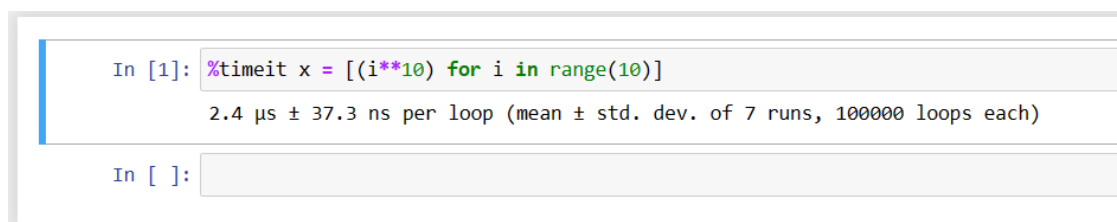


Рисунок 7. Пример 4

5. Индивидуальное задание: Создать ноутбук, в котором выполнить решение вычислительной задачи.

Два автомобиля едут навстречу друг другу с постоянными скоростями 60 км/ч и 80км/ч соответственно. Расстояние между автомобилями на

начальной момент составляет 200 км. На каком расстоянии от начальной точки встретятся автомобили?

```
In [8]: import numpy as np
import matplotlib.pyplot as plt

In [9]: speed1 = 60
speed2 = 80
distance = 100
speed1m = speed1 / 60
speed2m = speed2 / 60
time = distance / (speed1m + speed2m)

In [7]: print("Время встречи:", round(time,2), "минут")
Время встречи: 42.86 минут

In [10]: time_arr = np.linspace(0, time, 1000)
distance_arr = distance - (speed1m + speed2m) * time_arr

plt.plot(time_arr, distance_arr)
plt.xlabel("Время")
plt.ylabel("Расстояние")
plt.title("Зависимость расстояния от времени")
plt.show()
```

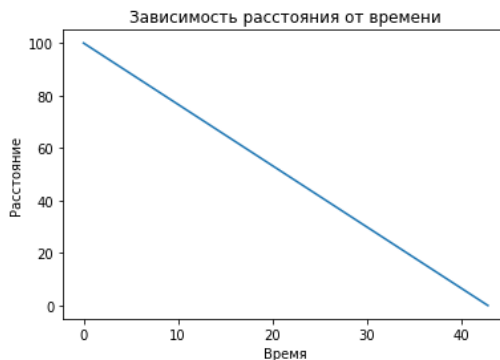


Рисунок 8. Индивидуальное задание

Контрольные вопросы:

1. Как осуществляется запуск Jupyter notebook?

Jupyter Notebook входит в состав Anaconda. Для запуска Jupyter Notebook перейдите в папку Scripts (она находится внутри каталога, в котором установлена Anaconda) и в командной строке наберите:> ipython notebook

В результате будет запущена оболочка в браузере.

2. Какие существуют типы ячеек в Jupyter notebook?

Если это код Python, то на панели инструментов нужно выставить свойство “Code”.

Если это Markdown текст – выставить “Markdown”.

3. Как осуществляется работа с ячейками в Jupyter notebook?

Если ваша программа зависла, то можно прервать ее выполнение выбрав на панели меню пункт

Kernel -> Interrupt.

Для добавления новой ячейки используйте Insert->Insert Cell Above и Insert->Insert Cell Below.

Для запуска ячейки используете команды из меню Cell, либо следующие сочетания клавиш:

Ctrl+Enter – выполнить содержимое ячейки.

Shift+Enter – выполнить содержимое ячейки и перейти на ячейку ниже.

Alt+Enter – выполнить содержимое ячейки и вставить новую ячейку ниже.

4. Что такое "магические" команды Jupyter notebook? Какие "магические" команды Вы знаете?

Важной частью функционала Jupyter Notebook является поддержка магии. Под магией в IPython понимаются дополнительные команды, выполняемые в рамках оболочки, которые облегчают процесс разработки и расширяют ваши возможности.

Для работы с переменными окружения используется команда %env.

Запуск Python кода из “.py” файлов, а также из других ноутбуков – файлов с расширением “.ipynb”, осуществляется с помощью команды %run.

%%time позволяет получить информацию о времени работы кода в рамках одной ячейки.

%timeit запускает переданный ей код 100000 раз (по умолчанию) и выводит информацию о среднем значении трех наиболее быстрых прогонов.

5. Самостоятельно изучите работу с Jupyter notebook и IDE PyCharm и Visual Studio Code.

Приведите основные этапы работы с Jupyter notebook в IDE PyCharm и Visual Studio Code.

PyCharm

1. Сначала вы должны создать новый проект.

2. В этом проекте создайте новый файл `ipy_nb`, выбрав `File> New...> Jupyter Notebook`. Это должно открыть новый файл записной книжки.

3. Если у вас не установлен пакет Jupyter Notebook, над вновь открытым файлом `ipy_nb` появится сообщение об ошибке. Сообщение об ошибке гласит: «Пакет Jupyter не установлен», и у вас будет опция «Установить пакет jupyter» рядом с ним.

4. Нажмите «Установить пакет jupyter». Это запустит процесс установки, который вы можете просмотреть, щелкнув запущенные процессы в правом нижнем углу окна PyCharm.

5. Чтобы начать изучение Jupyter Notebook в PyCharm, создайте ячейки кода и выполните их.

6. Выполните ячейку кода, чтобы запустить сервер Jupyter. По умолчанию сервер Jupyter использует порт 8888 по умолчанию на локальном хосте. Эти конфигурации доступны в окне инструментов сервера. После запуска вы можете просмотреть сервер над окном исходного кода, а рядом с ним вы можете просмотреть ядро, созданное как «Python 2» или «Python 3».

7. Теперь вы можете получить доступ к вкладке переменных в PyCharm, чтобы увидеть, как значения ваших переменных меняются при выполнении ячеек кода. Это помогает при отладке. Вы также можете установить точки останова в строках кода, а затем щелкнуть значок

«Выполнить» и выбрать «Debug Cell» (или использовать сочетание клавиш `Alt+Shift+Enter`), чтобы начать отладку.

Visual Studio Code

Если у вас еще нет существующего файла Jupyter Notebook, откройте VS Code Command Palette с помощью сочетания клавиш `CTRL+SHIFT+P` (Windows) или `Command+SHIFT+P` (macOS) и запустите команду «Python: Create Blank New Jupyter Notebook».

Если у вас уже есть файл Jupyter Notebook, это так же просто, как просто открыть этот файл в VS Code. Он автоматически откроется с новым нативным редактором Jupyter.

Вывод: исследованы базовые возможности интерактивных оболочек IPython и Jupyter Notebook для языка программирования Python.