

**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: «Программирование на Python»

Тема: «Работа с IPython и Jupyter Notebook»

Выполнила: студентка 2 курса,
группы ИВТ-б-о-21-1
Диченко Дина Алексеевна

Ставрополь 2023

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


Практическая часть:

1. Создала общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и выбранный Вами язык программирования.

Create a new repository

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

Owner *

 DinaDichenko ▾

Repository name *

Lab3_1 ✓

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

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 ▾

 You are creating a public repository in your personal account.

Create repository

Рисунок 1. Создание репозитория

2. Выполнила клонирование созданного репозитория на рабочий компьютер.

```

C:\Users\student-09-331\Downloads>git clone https://github.com/DinaDichenko/Lab3_1.git
Cloning into 'Lab3_1'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
C:\Users\student-09-331\Downloads>

```

Рисунок 2. Клонирование репозитория

3. Организовала свой репозиторий в соответствии с моделью ветвления git-flow.

```

C:\Users\student-09-331\Downloads\Lab3_1>git branch develop
C:\Users\student-09-331\Downloads\Lab3_1>git flow init
Which branch should be used for bringing forth production releases?
- develop
- main
branch name for production releases: [main] main
Which branch should be used for integration of the "next release"?
- develop
branch name for "next release" development: [develop] develop
How to name your supporting branch prefixes?
feature branches? [feature/] fea
bugfix branches? [bugfix/] bug
release branches? [release/] rel
hotfix branches? [hotfix/] hot
support branches? [support/] sup
version tag prefix? [] ver
Hooks and filters directory? [C:/Users/student-09-331/Downloads/Lab3_1/.git/hooks] hoo
C:\Users\student-09-331\Downloads\Lab3_1>

```

Рисунок 3. Организация репозитория в соответствии с flow-init

4. Дополнила файл .gitignore необходимыми правилами для выбранного языка программирования, интерактивной оболочки Jupyter notebook и интегрированной среды разработки.

```
.gitignore — Блокнот
Файл  Правка  Формат  Вид  Справка

.ropeproject

# mkdocs documentation
/site

# mypy
.mypy_cache/
.dmypy.json
dmypy.json

# Pyre type checker
.pyre/

# pytype static type analyzer
.pytype/

# Cython debug symbols
cython_debug/

# PyCharm
# JetBrains specific template is maintained in a separate JetBrains.gitignore that can
# be found at https://github.com/github/gitignore/blob/main/Global/JetBrains.gitignore
# and can be added to the global gitignore or merged into this file. For a more nuclear
# option (not recommended) you can uncomment the following to ignore the entire idea folder.
#.idea/

### Python Patch ###
# Poetry local configuration file - https://python-poetry.org/docs/configuration/#local-configuration
poetry.toml

# ruff
.ruff_cache/

# LSP config files
pyrightconfig.json

# End of https://www.toptal.com/developers/gitignore/api/jupyternotebooks,python|
```

Рисунок 4. Изменила файл .gitignore

6. Проработала примеры лабораторной работы.

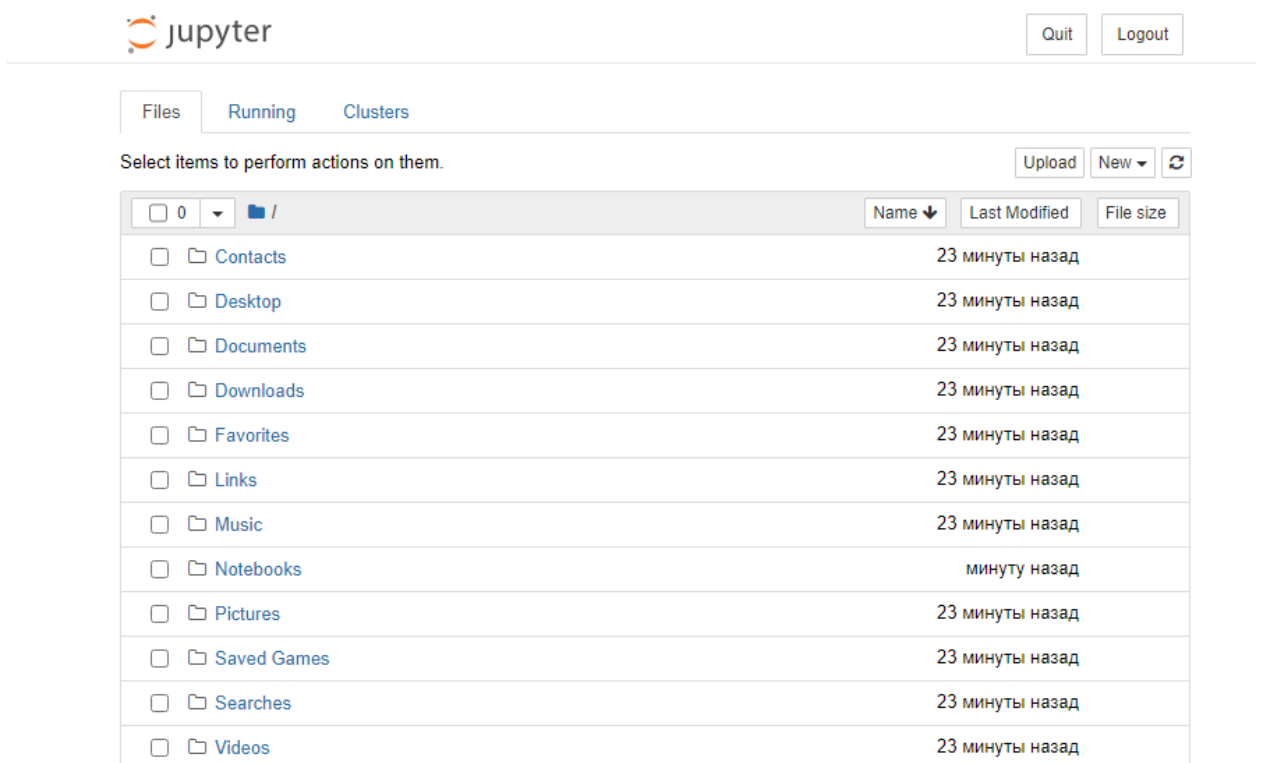


Рисунок 5. Открыла Jupyter Notebook

☐ ☐ Notebooks 1 минуту назад

Рисунок 6. Создала папку для работ

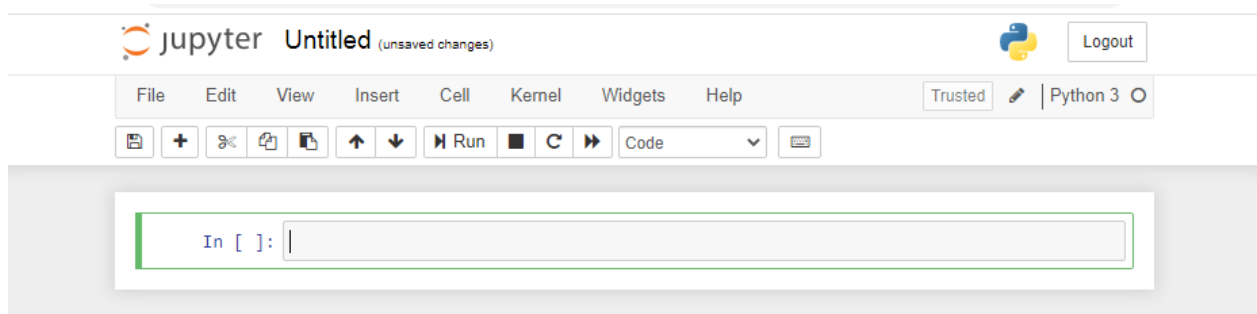


Рисунок 7. Создала ноутбук

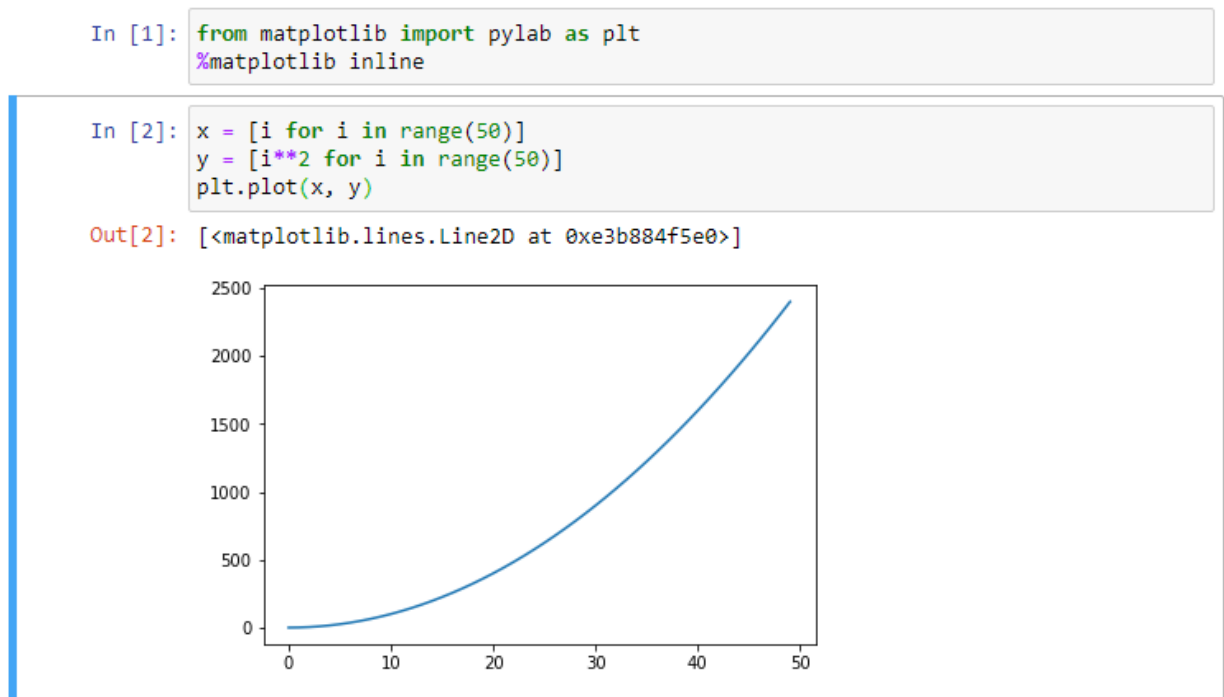


Рисунок 8. Проработка примера 1

```
In [2]: %lsmagic
```

```
Out[2]: Available line magics:
%alias %alias_magic %autoawait %autocall %automagic %autosave %bookmark
%cd %clear %cls %colors %conda %config %connect_info %copy %ddir %debu
g %dhist %dirs %doctest_mode %echo %ed %edit %env %gui %hist %history
%killbgscripts %ldir %less %load %load_ext %loadpy %logoff %logon %logs
tart %logstate %logstop %ls %lsmagic %macro %magic %matplotlib %mkdir
%more %notebook %page %pastebin %pdb %pdef %pdoc %pfile %pinfo %pinfo2
%pip %popd %pprint %precision %prun %psearch %psource %pushd %pwd %pyc
at %pylab %qtconsole %quickref %recall %rehashx %reload_ext %ren %rep
%rerun %reset %reset_selective %rmdir %run %save %sc %set_env %store %
sx %system %tb %time %timeit %unalias %unload_ext %who %who_ls %whos
%xdel %xmode

Available cell magics:
%%! %%HTML %%SVG %%bash %%capture %%cmd %%debug %%file %%html %%javasc
ript %%js %%latex %%markdown %%perl %%prun %%ppyy %%python %%python2 %
%python3 %%ruby %script %sh %svg %sx %system %time %timeit %wr
itefile

Automagic is ON, % prefix IS NOT needed for line magics.
```

Рисунок 9. Список доступных магических команд

```
In [1]: %env TEST = 5
env: TEST=5
```

Рисунок 10. Работа с переменными окружения

```
In [3]: %%time
import time
for i in range(50):
    time.sleep(0.1)

Wall time: 5.35 s
```

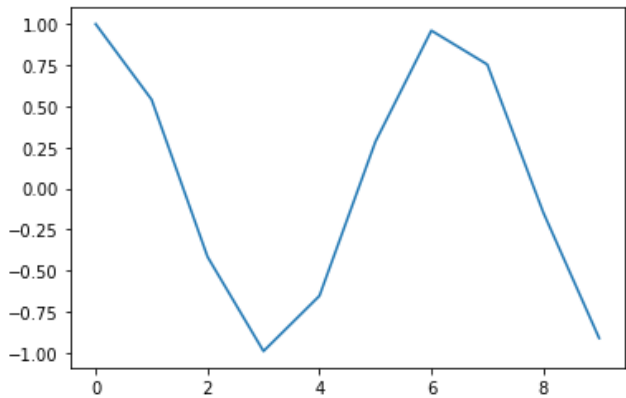
Рисунок 11. Информация о времени работы кода

8. Создала ноутбук, в котором выполнила решение вычислительной задачи (например, задачи из области физики, экономики, математики, статистики и т. д.), условие которой предварительно необходимо согласовать с преподавателем.

```
In [24]: from matplotlib import pylab as plt
import math
%matplotlib inline

In [27]: x = [i for i in range(10)]
y=[math.cos(i) for i in range(10)]
plt.plot(x,y)

Out[27]: [<matplotlib.lines.Line2D at 0xf95fe27d60>]
```



x	y
0	1.00
1	0.54
2	-0.42
3	-0.99
4	-0.76
5	0.28
6	0.96
7	0.75
8	-0.14
9	-0.96

Рисунок 12. Индивидуальное задание

Контрольные вопросы:

1. Как осуществляется запуск Jupyter notebook?

Для запуска Jupyter Notebook перейдите в папку Scripts (она находится внутри каталога, в котором установлена Anaconda) и в командной строке наберите:

>ipython notebook

2. Какие существуют типы ячеек в Jupyter notebook?

- Ячейка кода содержит код, который должен быть выполнен в ядре, и отображает его вывод ниже.
- Ячейка Markdown содержит текст, отформатированный с использованием Markdown, и отображает его вывод на месте при запуске.

2. Как осуществляется работа с ячейками в Jupyter notebook?

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.

3. 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. Список доступных магических команд можно получить с помощью команды `%lsmagic`.

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`.

Для измерения времени работы кода используйте `%%time` и `%timeit`.

`%%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.

4. Самостоятельно изучите работу с Jupyter notebook и IDE PyCharm и Visual Studio Code.
5. Приведите основные этапы работы с Jupyter notebook в IDE PyCharm и Visual Studio Code.

Вывод: в результате выполнения работы были исследованы базовые возможности интерактивных оболочек IPython и Jupyter Notebook для языка программирования Python.