Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

# ОТЧЕТ
# ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
**дисциплины**
**«Искусственный интеллект и машинное обучение»**
**Вариант 2**

Выполнила:
Алексеева Софья Алексеевна
2 курс, группа ИВТ-б-о-23-1,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____
(подпись)

Проверил:
доцент департамента цифровых, робототехнических систем и электроники института перспективной инженерии Воронкин Р. А.

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2025 г.

**Тема:** Работа с Jupyter Notebook, JupyterLab и Google Colab.

**Цель:** исследовать базовые возможности интерактивных оболочек Jupyter Notebook, JupyterLab и Google Colab для языка программирования Python.

**Порядок выполнения работы:**

1. Запустили оболочку Jupyter Notebook в браузере.

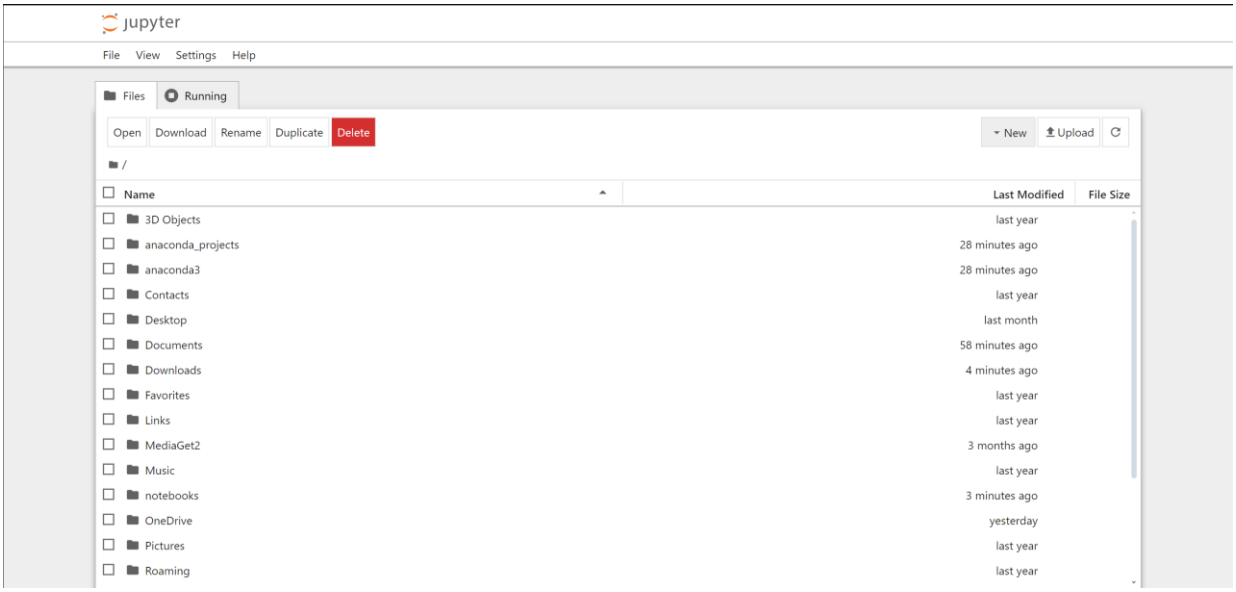


Рисунок 1. Оболочка Jupyter Notebook в браузере

2. Создали первый файл.

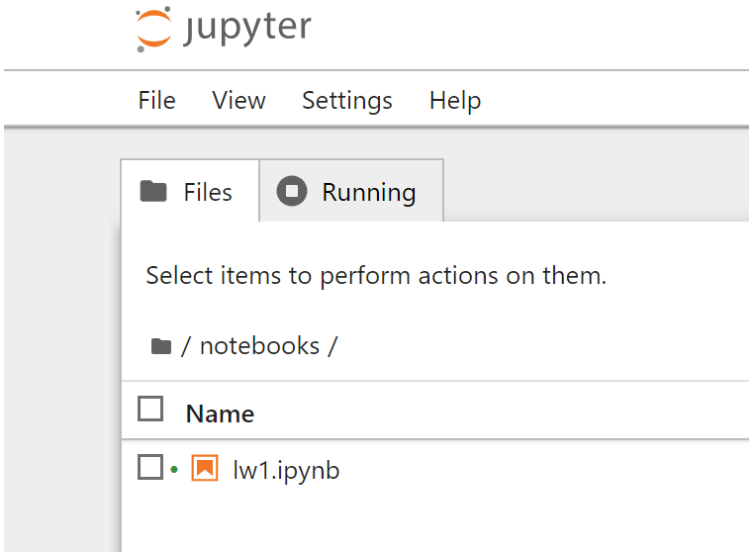


Рисунок 2. Первый файл в Jupyter Notebook

3. Выполнили несколько примеров.

```
[1]: 3 + 2

[1]: 5

[3]: a = 5
     b = 7
     print (a + b)

     12

[5]: n = 7
     for i in range(n):
         print(i*10)

     0
     10
     20
     30
     40
     50
     60

[7]: i = 0
     while True:
         i += 1
         if i > 5:
             break
         print("test while")

     test while
     test while
     test while
     test while
     test while
```

Рисунок 3. Несколько выполненных примеров

4. Вывели изображение на экран.

```
[11]: from matplotlib import pylab as plt
      %matplotlib inline

[12]: x = [i for i in range(50)]
      y = [i**2 for i in range(50)]
      plt.plot(x, y)

[12]: [<matplotlib.lines.Line2D at 0x1f8f9fe8ec0>]
```

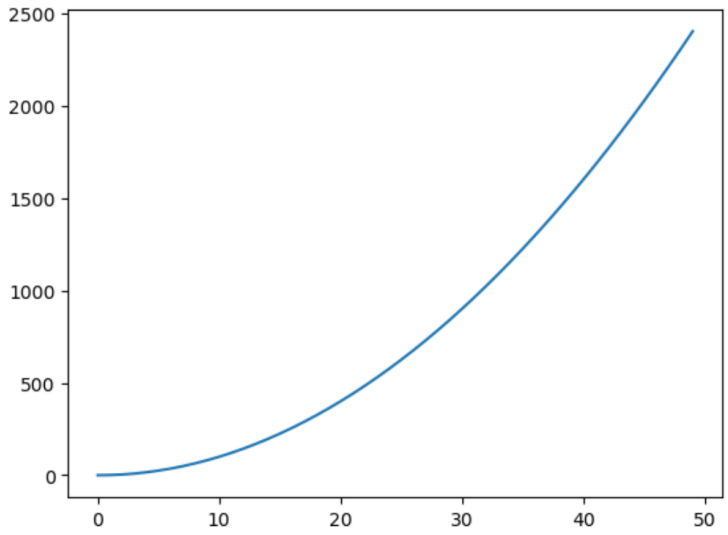


Рисунок 4. Выведение графика в Jupyter Notebook

5. Использовали магические команды. Они были выведены в виде списков.

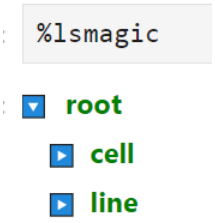


Рисунок 5. Магические команды в Jupyter Notebook

6. Применили команду %env для работы с окружением.

```
%env TEST = 5
```

```
env: TEST=5
```

Рисунок 6. Команда %env

7. Примени команду %run для запуска кода из другого файла.

```
%run ./test.ipynb
```

```
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
```

Рисунок 7. Команда %run

8. Использовали команду %%time. Параметр CPU times выводит время работы процессора, исключая периоды ожидания между ними. Параметр Wall time – общее время работы операции.

```
%%time
import time
for i in range(50):
    time.sleep(0.1)
```

```
CPU times: total: 15.6 ms
Wall time: 5.04 s
```

Рисунок 8. Команда %%time

9. Использовали команду %timeit.

```
%timeit x = [(i**10) for i in range(10)]
```

```
758 ns ± 7.23 ns per loop (mean ± std. dev. of 7 runs, 1,000,000 loops each)
```

Рисунок 9. Команда %timeit

10. Использовали Markdown для форматирования текста и формул.

# Заголвок первого уровня

## Заголовок второго уровня

**Полужирный текст**, *курсив*, `код в строке` Список:

- Пункт 1
- Пункт 2
- Пункт 3 Формула: $y = mx + b$

Рисунок 10. Использование Markdown

11. Использовали линейную магическую команду %timeit и блочную магическую команду %%time в JupyterLab.

```
%timeit sum(range(100))
```

```
697 ns ± 4.59 ns per loop (mean ± std. dev. of 7 runs, 1,000,000 loops each)
```

```
%%time
total = 0
for i in range(10**6):
    total += 1
```

```
CPU times: total: 125 ms
Wall time: 124 ms
```

Рисунок 11. Команды %timeit и %%time

12. Построили график с помощью библиотеки matplotlib.

```python
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 10, 100)
y = np.sin(x)

plt.plot(x, y)
plt.xlabel("X")
plt.ylabel("Y")
plt.title("График синусоиды")
plt.show()
```

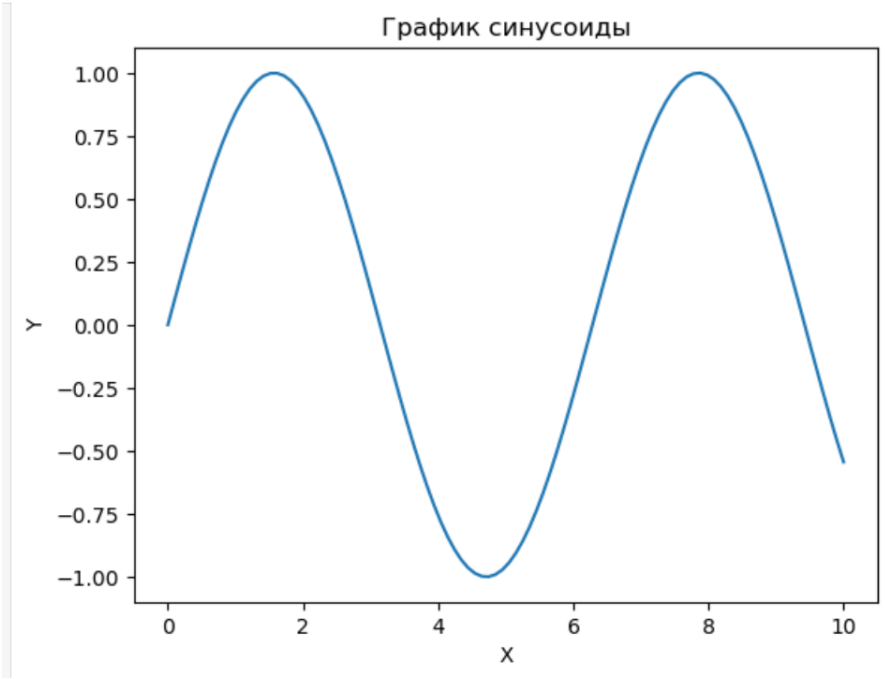


Рисунок 12. Построение графика в JupyterLab

13. Установили с помощью терминала несколько библиотек. Проверили версию Python.

```
PS C:\Users\Sonya> pip install numpy pandas matplotlib
Requirement already satisfied: numpy in c:\users\sonya\anaconda3\lib\site-packages (1.26.4)
Requirement already satisfied: pandas in c:\users\sonya\anaconda3\lib\site-packages (2.2.2)
Requirement already satisfied: matplotlib in c:\users\sonya\anaconda3\lib\site-packages (3.9.2)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\sonya\appdata\roaming\python\python312\sit
e-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\sonya\anaconda3\lib\site-packages (from pandas) (202
4.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\sonya\anaconda3\lib\site-packages (from pandas) (2
023.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\sonya\anaconda3\lib\site-packages (from matplotl
ib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\sonya\anaconda3\lib\site-packages (from matplotlib)
(0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\sonya\anaconda3\lib\site-packages (from matplot
lib) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\sonya\anaconda3\lib\site-packages (from matplot
lib) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\sonya\appdata\roaming\python\python312\site-packa
ges (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in c:\users\sonya\anaconda3\lib\site-packages (from matplotlib) (10
.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\sonya\anaconda3\lib\site-packages (from matplotl
ib) (3.1.2)
Requirement already satisfied: six>=1.5 in c:\users\sonya\appdata\roaming\python\python312\site-packages (fr
om python-dateutil>=2.8.2->pandas) (1.17.0)
PS C:\Users\Sonya> python --version
Python 3.12.7
PS C:\Users\Sonya>
```

Рисунок 13. Использование терминала

14. Создали новый ноутбук в Google Colab. Использовали тип ячейки Markdown.

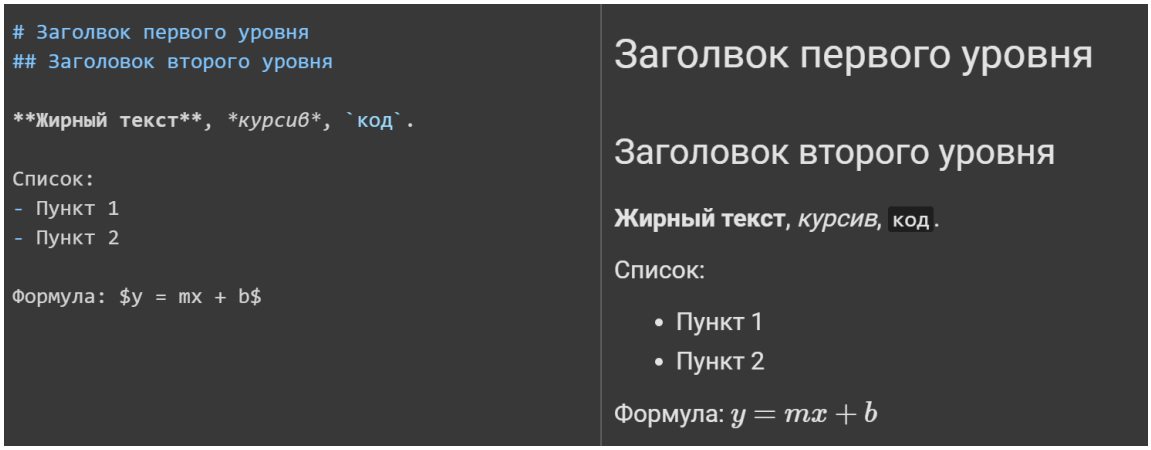


Рисунок 14. Использование типа ячейки Markdown в Google Colab

15. Изменили среду выполнения и проверили наличие GPU.

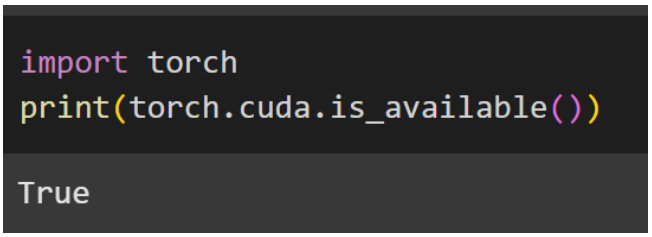


Рисунок 15. Проверка наличия GPU

16. Подключили Google диск. Далее появляется окно с авторизацией и запросом на доступ.

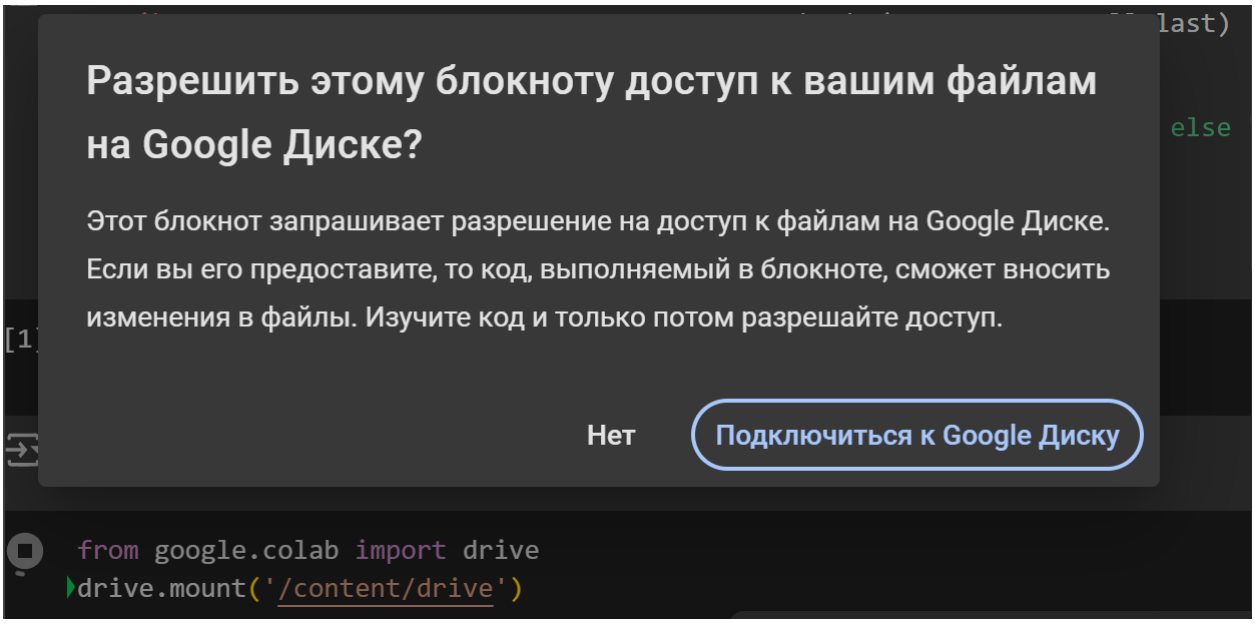


Рисунок 16. Подключение Google диска к Google Colab

17. Выполнили загрузку файлов через код.

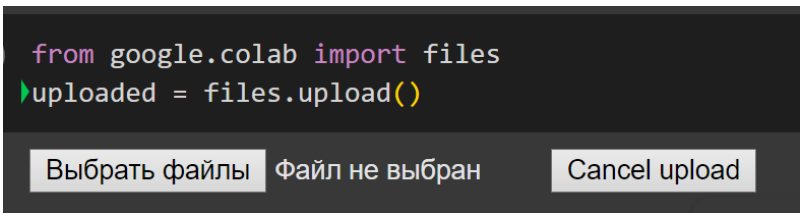


Рисунок 17. Загрузка файлов

18. Выполнили работу с файловой системой: вывели текущую директорию, создали каталог, просмотрели файлы в директории и удалили каталог.

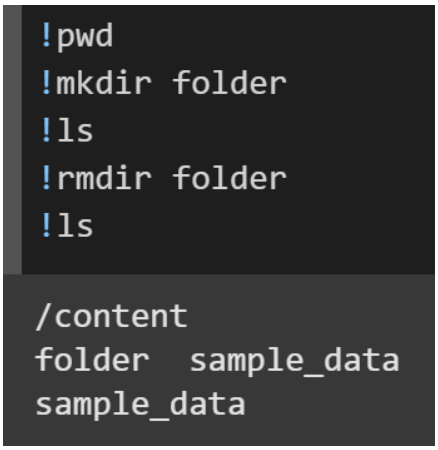


Рисунок 18. Команды для работы с файловой системой

19. Использовали магические команды в Google Colab.

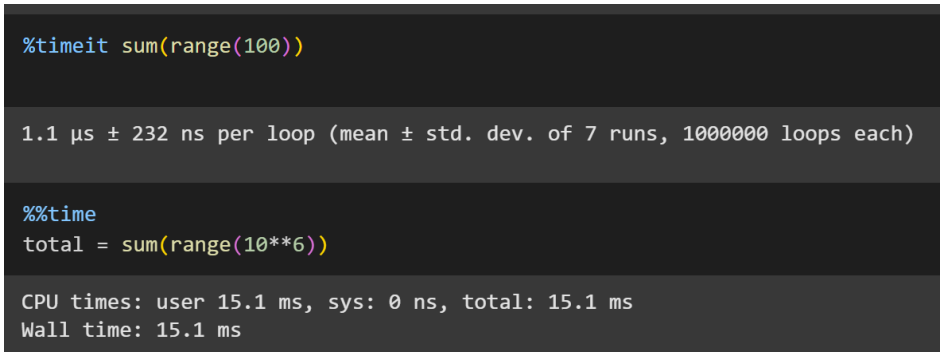


Рисунок 19. Магические команды в Google Colab

20. Построили график в Google Colab.

```python
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 10, 100)
y = np.sin(x)

plt.plot(x, y)
plt.xlabel("x")
plt.ylabel("Y")
plt.title("График сунисоиды")
plt.show()
```

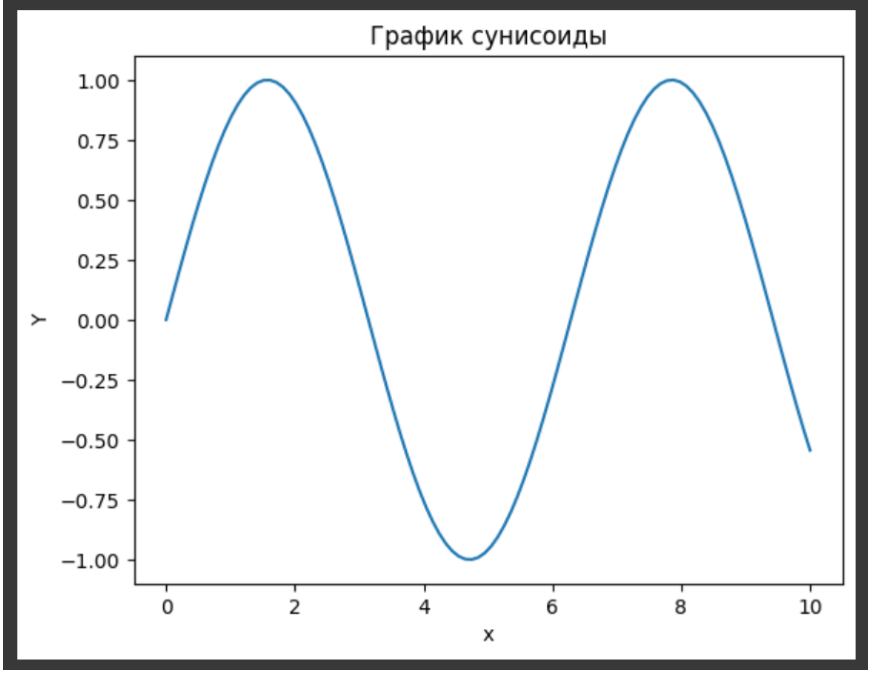


Рисунок 20. Построение графика в Google Colab

21. Установили стандартные библиотеки.

```
!pip install numpy pandas matplotlib

Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (1.26.4)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplo
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matpl
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matpl
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplot
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplo
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateuti
```

Рисунок 21. Установка стандартных библиотек

22. Создать Markdown и:

– написали заголовок "Практическое задание №1";

– добавили жирный и курсивный текст;

– создали нумерованный и маркированный списки;

– вставили согласно индивидуальному заданию;

– вставили изображение через ![Описание](URL).

```
# Практическое задание №1
**Это жирное выделение текста**.

*А это курсивное*.

Моё расписание:
1.   Проснулась
2.   Улыбнулась
3.   Уснула

Список дел на день:
*   Сделать домашнее задание.
*   Помыть пол.
*   Лечь не позже 12.

$$ \frac{d}{dx} \ln(x) = \frac{1}{x} $$

![Русская борзая](https://avatars.mds.yandex.net/i?
id=8f422ac622d4f0dece9f3f3e119dfd39_l-8491909-images-thumbs&n=13)
```

Рисунок 22. Код ячейки с типом Markdown


Рисунок 23. Код программы

23. Создали ячейку Python-кода и:

– запросили у пользователя его имя с помощью input();

– вывели приветствие: «Привет, <имя>! Добро пожаловать в Google Colab!»;

– запустили ячейку (Shift + Enter).

```
name = input()
print(f"Привет, {name}! Добро пожаловать в Google Colab!")
```

```
Софья
Привет, Софья! Добро пожаловать в Google Colab!
```

Рисунок 24. Вывод приветствия с помощью input()

24. Выполнили список указаний:

– создали и сохранили текстовый файл с помощью open();

– записали в него несколько строк текста;

– закрыли и затем снова открыли его, считалв содержимое и выведя на экран;

– проверили, существует ли файл, используя os.path.exists();

– удалили файл с помощью модуля os.

```
import os

with open("example.txt", "w") as f:
  f.write("Первая строка\n")
  f.write("Вторая строка\n")
  print("Содержимое файла:\n", content)

with open("example.txt", "r") as f:
  content = f.read()
  print("Содержимое файла:\n", content)

print("Файл существует:", os.path.exists("example.txt"))

os.remove("example.txt")
```

```
Содержимое файла:
 <built-in method read of _io.TextIOWrapper object at 0x789a5dffdb10>
Содержимое файла:
 Первая строка
Вторая строка

Файл существует: True
```

Рисунок 25. Выполнение работ с файлами

25. Вывели список всех доступных магических команд.

```
%lsmagic
```

```
Available line magics:
%alias  %alias_magic  %autoawait  %autocall  %automagic  %autosave  %bookmark
%cat  %cd  %clear  %colors  %conda  %config  %connect_info  %cp  %debug  %dhis
%dirs  %doctest_mode  %ed  %edit  %env  %gui  %hist  %history  %killbgscripts
%ldir  %less  %lf  %lk  %ll  %load  %load_ext  %loadpy  %logoff  %logon
%logstart  %logstate  %logstop  %ls  %lsmagic  %lx  %macro  %magic  %man
%matplotlib  %mkdir  %more  %mv  %notebook  %page  %pastebin  %pdb  %pdef  %pd
%pfile  %pinfo  %pinfo2  %pip  %popd  %pprint  %precision  %prun  %psearch
%psource  %pushd  %pwd  %pycat  %pylab  %qtconsole  %quickref  %recall  %rehas
%reload_ext  %rep  %rerun  %reset  %reset_selective  %rm  %rmdir  %run  %save
%sc  %set_env  %shell  %store  %sx  %system  %tb  %tensorflow_version  %time
%timeit  %unalias  %unload_ext  %who  %who_ls  %whos  %xdel  %xmode

Available cell magics:
%%!  %%HTML  %%SVG  %%bash  %%bigquery  %%bqsql  %%capture  %%debug  %%file
%%html  %%javascript  %%js  %%latex  %%markdown  %%perl  %%prun  %%pypy  %%pyt
%%python2  %%python3  %%ruby  %%script  %%sh  %%shell  %%spanner_graph  %%svg
%%sx  %%system  %%time  %%timeit  %%writefile

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

Рисунок 26. Список магических команд в Google Colab

26. Выполнили список указаний:

– использовали %time для измерения врмени выполнения кода;

– создали python-скрипт (%%writefile script.py) и выполнили его через !python script.py;

– вывели список файлов в текущей директории с помощью %ls;

– использовали %history для просмотра истории команд.

```
%%writefile test_script.py
for i in range(3):
  print(f"Итерация {i}")
```

```
Overwriting test_script.py
```

```
!python test_script.py

%time sum(range(200))
%ls
%history
```

```
Итерация 0
Итерация 1
Итерация 2
CPU times: user 11 µs, sys: 1 µs, total: 12 µs
Wall time: 17.4 µs
sample_data/  test_script.py
name = input()
print("Привет, ",name,"!Добро пожаловать в Google Colab!")
name = input()
print("Привет, ",name,"! Добро пожаловать в Google Colab!")
name = input()
```

Рисунок 27. Выполненные указания в Google Colab

27. Выполнили список указаний по командам в терминале:

– вывели список файлов в текущей директории с помощью !ls;
– проверили, какой Python используется;
– создали папку test и убедитесь, что она появилась;
– удалили папку;
– очистили вывод в ячейке.

```
!ls
!which python
!mkdir test
!ls
!rmdir test
!ls
!clear
```

```
sample_data  test_folder
/usr/local/bin/python
sample_data  test  test_folder
sample_data  test_folder
H
```

Рисунок 28. Выполнили команды терминала в Google Colab

28. Выполнили список указаний по командам в терминале:

– подключили Google Drive;
– создали и сохранили текстовый файл в Google Drive;
– прочитали файл из Google Drive;
– создали и сохранили csv-файл вручную.

```
from google.colab import drive
drive.mount("/content/drive")
```

```
Mounted at /content/drive
```

```
!ls /content/drive/MyDrive
```

```
'Colab Notebooks'   анализ.gdoc   экзы
```

```
file_path = "/content/drive/MyDrive/my_text_file.txt"
with open(file_path, "w") as f:
  f.write("Тестовый файл в  Google Drive.")
  f.write("Второй абзац.")
print("Файл сохранен.")
with open(file_path, "r") as f:
  content = f.read()
  print("Содержимое файла:\n", content)
cats = [
    ["Лили", 2, "Рэгдолл"],
    ["Песто", 0.8, "Шотландская золотая шиншила"],
    ["Ктулху", 1, "Корниш-рекс"]
]
csv_path = "/content/drive/MyDrive/cats.csv"
with open(csv_path, "w") as f:
  for cats in cats:
    f.write(",".join(map(str, cats)) + "\n")
print("Файл cats.csv сохранен.")
```

```
Файл сохранен.
Содержимое файла:
 Тестовый файл в  Google Drive.Второй абзац.
Файл cats.csv сохранен.
```

Рисунок 29. Работа с Google Drive в Google Colab

## Ответы на контрольные вопросы:

1. Какие основные отличия JupyterLab от Jupyter Notebook?

| Функции | Jupyter Notebook | JupyterLab |
|---|---|---|
| Интерфейс и организации работы | Отдельные веб-страницы с линейной последовательностью ячеек. Окружение ограничено одной тетрадью. | Интегрированная среда (IDE) с вкладками, панелями, файловыми менеджерами, терминалом и редактором кода. Редактирование нескольких файлов одновременно. |
| Работа с файлами | .ipynb. | .ipynb, .py, .csv, .md, .json, .yaml, .txt. |
| Многозадачность | Требуется несколько вкладов браузера для работы с разными тетрадями. | Система вкладок и разделения экрана позволяет работать с несколькими файлами в одном окне. |
| Поддержка расширений | Поддерживает, сложная настройка. | Встроенный менеджер расширений, удобное добавление новых функций. |

| | | |
|---|---|---|
| Гибкость интерфейса | Фиксированный интерфейс. | Гибкий интерфейс: настройка панелей, вида и работа в нескольких окнах. |
| Поддержка терминала | Требуется ручное подключение и настройка. | Встроенный терминал. |
| Производительность | Высокая. | Потребляет много ресурсов при нескольких вкладках. |

Таблица 1. Основные отличия Jupyter Notebook и JupyterLab

2. Как создать новую рабочую среду (ноутбук) в JupyterLab?

Необходимо в меню выбрать File, затем в выпадающим меню выбрать New и Notebook. Новая тетрадь сразу же откроется.

3. Какие типы ячеек поддерживаются в JupyterLab и как их переключать?

Код (Code) – для написания и выполнения программного кода.

Текст (Markdown) – используется для оформления пояснений, форматированного текста и математических формул на основе LaTeX.

Вывод (Raw) – предназначен для хранения необработанного текста, например, для экспорта в другие форматы.

4. Как выполнить код в ячейке и какие горячие клавиши для этого используются?

Запустить код в ячейке можно с помощью кнопки перевернутого треугольника. Выполнение кода с помощью горячих клавиш запускается сочетанием Shift + Enter.

5. Как запустить терминал или текстовый редактор внутри JupyterLab?

Терминал запускается через меню File, затем пункты New и Terminal.

Текстовой редактор запускается при изменении типа ячейки на Markdown.

6. Какие инструменты JupyterLab позволяют работать с файлами и структурами каталогов?

JupyterLab предоставляет удобную систему взаимодействия с файлами благодаря файловому браузеру в левой боковой панели.

Сама среда позволяет открывать и редактировать файлы разных типов, включая. ipynb, .py, .txt, .json, .csv, и другие.

7. Как можно управлять ядрами (kernels) в JupyterLab?

Управление ядрами осуществляется несколькими способами. Первый способ – выбрать в верхнем меню кнопку Kermel.

Второй способ – в левом боковом меню выбрать круглую иконку.

8. Каковы основные возможности системы вкладок и окон в интерфейсе JupyterLab?

JupyterLab имеет гибкий интерфейс, позволяющий пользователю настраивать его под себя.

Программа имеет многооконный режим. Он позволяет работать сразу с несколькими файлами. Так же рабочее пространство можно разделить на несколько панелей.

В левом боковом меню расположена удобная система работы с вкладками, позволяющая взаимодействовать с файловой системой.

В правом боковом меню находятся несколько функций: сохранение и использование макетов кода, хранение переменных, использованных ресурсов, точки остановы и встроенный искусственный интеллект.

9. Какие магические команды можно использовать в JupyterLab для измерения времени выполнения кода? Приведите примеры.

Для измерения времени выполнения команды используется %timeit.

Для измерения времени выполнения всей ячейки используется %%time.

10. Какие магические команды позволяют запускать код на других языках программирования в JupyterLab?

Некоторые языки программирования можно запустить с помощью магических команд. Например, код Python можно запустить с помощью команды %run. Прочие языки запускать с помощью команд %%bash, %sql, %%js и т.д.

11. Какие основные отличия Google Colab от JupyterLab?

| Функции | Google Colab | JupyterLab |
|---|---|---|

| | | |
|---|---|---|
| Интерфейс и организации работы | Облачный сервис, работающий в браузере. Интерфейс с вкладками, но ограниченный по настройке. | Интегрированная среда (IDE) с вкладками, панелями, файловыми менеджерами, терминалом и редактором кода. Редактирование нескольких файлов одновременно. |
| Работа с файлами | Виртуальная файловая система, доступ через Google Drive. | .ipynb, .py, .csv, .md, .json, .yaml, .txt. |
| Многозадачность | Открытие нескольких вкладок браузера возможно, но ограничено сессиями. | Система вкладок и разделения экрана позволяет работать с несколькими файлами в одном окне. |
| Поддержка расширений | Ограничена, нельзя устанавливать пользовательские расширения. | Встроенный менеджер расширений, удобное добавление новых функций. |
| Гибкость интерфейса | Фиксированный интерфейс. | Гибкий интерфейс: настройка панелей, вида и работа в нескольких окнах. |
| Поддержка терминала | Доступ к терминалу через !. | Встроенный терминал. |
| Производительность | Требует подключения к серверу. | Потребляет много ресурсов при нескольких вкладках. |

Таблица 2. Основные отличия Google Colab и JupyterLab

12. Как создать новый ноутбук в Google Colab?

В верхнем меню необходимо нажать Файл, а затем «Создать новый блокнот».

13. Какие типы ячеек доступны в Google Colab, и как их переключать?

В Google Colab доступно 2 вида ячеек: код и текст.

Переключаться между ними можно несколькими способами. Первый – горячие клавиши Ctrl + M M для смены на Markdown и Ctrl + M Y для смены на код.

Второй способ – нажать на кнопки +Код и + Текст под верхним меню.

Третий способ – при наведении мышью под текущую ячейку появится выбор типа новой ячейки.

14. Как выполнить код в ячейке Google Colab и какие горячие клавиши для этого используются?

Выполнить код в ячейке можно двумя способами.

Первый способ – навести на область слева от ячейки.

Второй способ – нажать комбинацию горячих клавиш Shift + Enter.

15. Какие способы загрузки и сохранения файлов поддерживает Google Colab?

Загрузка файлов с компьютера осуществляется с помощью следующих команд:

from google.colab import files

uploaded = files.upload().

Так же файлы можно загрузить из хранилища Google Drive:

from google.colab import drive

drive.mount('/content/drive').

16. Как можно подключить Google Drive к Google Colab и работать с файлами?

Подключение Google Drive происходит следующим образом:

from google.colab import drive

drive.mount('/content/drive').

17. Какие команды используются для загрузки файлов в Google Colab из локального компьютера?

from google.colab import files

uploaded = files.upload().

18. Как посмотреть список файлов, хранящихся в среде Google Colab?

Необходимо использовать команду !ls.

19. Какие магические команды можно использовать в Google Colab для измерения времени выполнения кода? Приведите примеры.

Для измерения времени выполнения команды используется %timeit.

Для измерения времени выполнения всей ячейки используется %%time.

20. Как можно изменить аппаратные ресурсы в Google Colab (например, переключиться на GPU)?

Для этого необходимо в верхнем меню выбрать пункт Среда выполнения. В выпадающем меню выбрать «сменить среду выполнения». В открывшимся окне выбрать необходимую среду.

**Вывод:** в ходе лабораторной работы были исследованы базовые возможности интерактивных оболочек Jupyter Notebook, JupyterLab и Google Colab для языка программирования Python, выполнены практические задания.