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»

Выполнил:
Хохлачев Вадим Александрович
2 курс, группа ИТС-б-о-23-1,
11.03.02«Инфокоммуникационные технологии и системы связи», очная форма обучения

_____
(подпись)

Проверил:
Ассистеннт департамента цифровых, робототехнических систем и электроники Хацукова А.И.

_____
(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. Ознакомились с теоретическим материалом

2. Создание репозитория

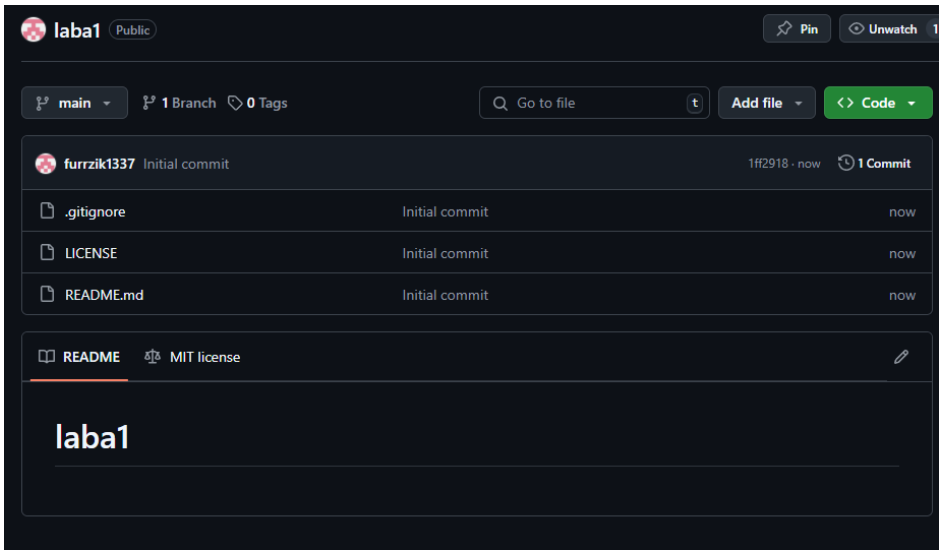


Рисунок 1. Репозиторий

3. Выполнено клонирование репозитория

```
C:\Users\furrzik>git clone https://github.com/furrzik1337/laba1.git
Cloning into 'laba1'...
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 (from 0)
Receiving objects: 100% (5/5), done.
```

Рисунок 2. Клонирование

4. Проработан примеры работы с Jupyter

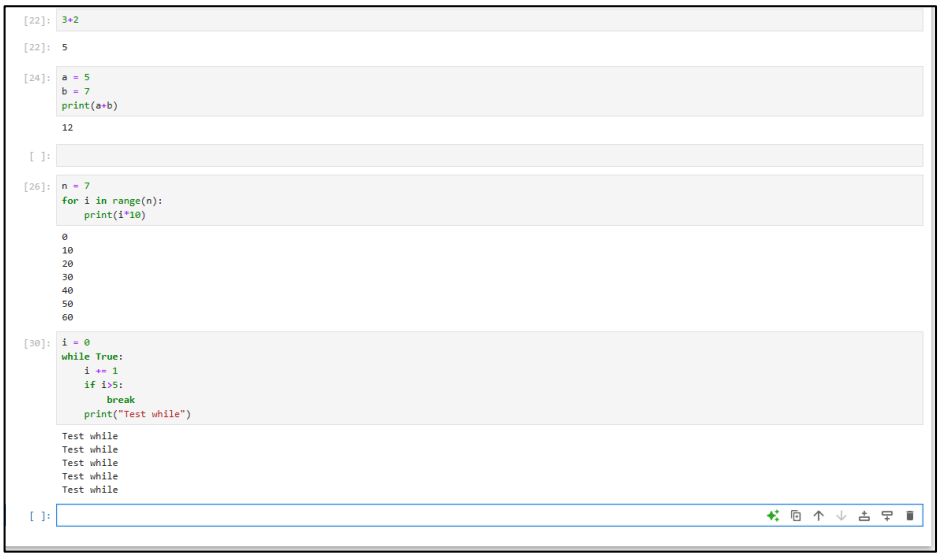


Рисунок 3. Проработка примеров

5. Проработан пример работы с JupyterLab

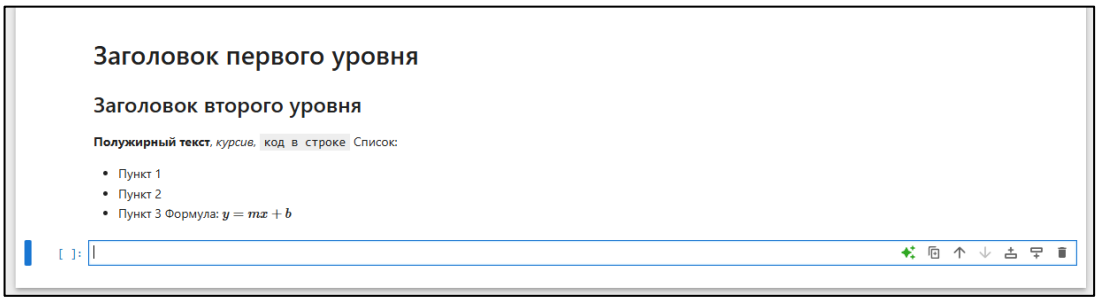


Рисунок 4. Проработка примера

6. Проработан пример с построением графика

```
[32]: 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()
```

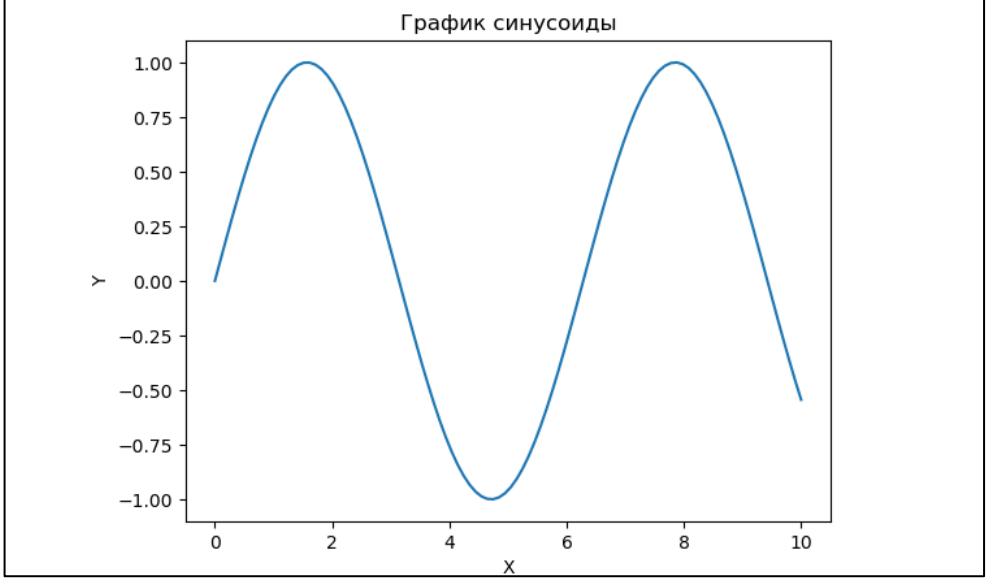


Рисунок 5. Пример с графиком

7. Пример работы с Google Colab

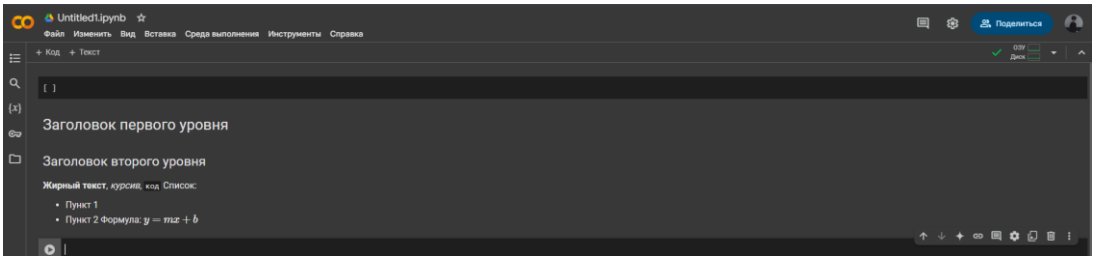


Рисунок 6. Google Colab

8. Выполнено практическое задание с Markdown-ячейкой

"Практическое задание №1"

Жирный текст

Текст с курсивом

Нумерованный список:

1. Первый элемент
2. Второй элемент
3. Третий элемент

Маркированный список

- Элемент 1
- Элемент 2
- Элемент 3

Формула: $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$

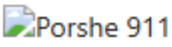


Рисунок 7. Markdown-ячейка

9. Выполнено практическое задание с использованием ячейки Python-кода

```
[2]: name = input("Пожалуйста, введите ваше имя: ")
     print("Привет, {}! Добро пожаловать в JupyterLab / Google Colab!".format(name))

     Пожалуйста, введите ваше имя:  Вадим
     Привет, Вадим! Добро пожаловать в JupyterLab / Google Colab!
```

Рисунок 8. Python-код

10. Решена простенькая задача

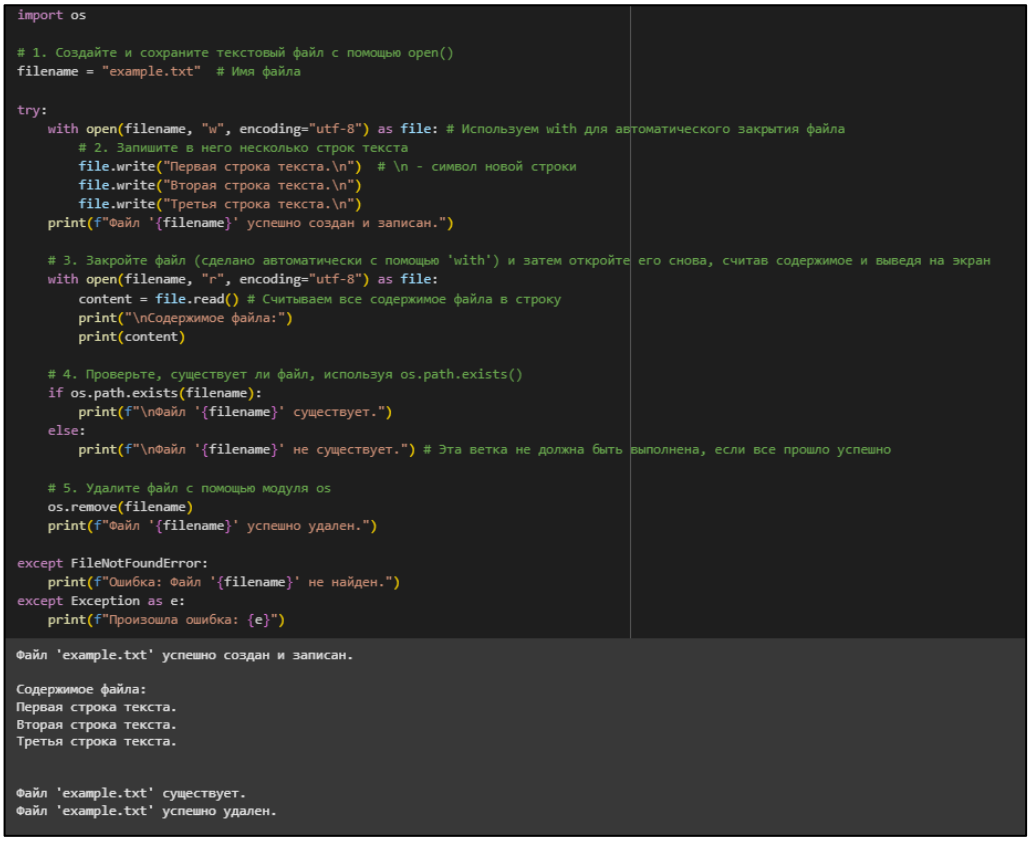
```
[6]: # Входные данные:
     a = 5
     b = 3

     # Вычисление площади:
     s = a * b

     # Вывод результата:
     print("Площадь прямоугольника:", s)

     Площадь прямоугольника: 15

[ ]:
```

Рисунок 9. Задачка

11. Выполнено задание с открытием файла

```
[2]: import os

# 1. Создайте и сохраните текстовый файл с помощью open()
filename = "example.txt"  # Имя файла

try:
    with open(filename, "w", encoding="utf-8") as file: # Используем with для автоматического закрытия файла
        # 2. Запишите в него несколько строк текста
        file.write("Первая строка текста.\n")  # \n - символ новой строки
        file.write("Вторая строка текста.\n")
        file.write("Третья строка текста.\n")
    print(f"Файл '{filename}' успешно создан и записан.")

    # 3. Закройте файл (сделано автоматически с помощью 'with') и затем откройте его снова, считав содержимое и выведя на экран
    with open(filename, "r", encoding="utf-8") as file:
        content = file.read() # Считываем все содержимое файла в строку
        print("\nСодержимое файла:")
        print(content)

    # 4. Проверьте, существует ли файл, используя os.path.exists()
    if os.path.exists(filename):
        print(f"\nФайл '{filename}' существует.")
    else:
        print(f"\nФайл '{filename}' не существует.") # Эта ветка не должна быть выполнена, если все прошло успешно

    # 5. Удалите файл с помощью модуля os
    os.remove(filename)
    print(f"Файл '{filename}' успешно удален.")

except FileNotFoundError:
    print(f"Ошибка: Файл '{filename}' не найден.")
except Exception as e:
    print(f"Произошла ошибка: {e}")

Файл 'example.txt' успешно создан и записан.

Содержимое файла:
Первая строка текста.
Вторая строка текста.
Третья строка текста.


Файл 'example.txt' существует.
Файл 'example.txt' успешно удален.
```

Рисунок 10. Работа с файлами в JupyterLab



```
import os

# 1. Создайте и сохраните текстовый файл с помощью open()
filename = "example.txt"  # Имя файла

try:
    with open(filename, "w", encoding="utf-8") as file: # Используем with для автоматического закрытия файла
        # 2. Запишите в него несколько строк текста
        file.write("Первая строка текста.\n")  # \n - символ новой строки
        file.write("Вторая строка текста.\n")
        file.write("Третья строка текста.\n")
    print(f"Файл '{filename}' успешно создан и записан.")

    # 3. Закройте файл (сделано автоматически с помощью 'with') и затем откройте его снова, считав содержимое и выведя на экран
    with open(filename, "r", encoding="utf-8") as file:
        content = file.read() # Считываем все содержимое файла в строку
        print("\nСодержимое файла:")
        print(content)

    # 4. Проверьте, существует ли файл, используя os.path.exists()
    if os.path.exists(filename):
        print(f"\nФайл '{filename}' существует.")
    else:
        print(f"\nФайл '{filename}' не существует.") # Эта ветка не должна быть выполнена, если все прошло успешно

    # 5. Удалите файл с помощью модуля os
    os.remove(filename)
    print(f"Файл '{filename}' успешно удален.")

except FileNotFoundError:
    print(f"Ошибка: Файл '{filename}' не найден.")
except Exception as e:
    print(f"Произошла ошибка: {e}")

Файл 'example.txt' успешно создан и записан.

Содержимое файла:
Первая строка текста.
Вторая строка текста.
Третья строка текста.


Файл 'example.txt' существует.
Файл 'example.txt' успешно удален.
```

Рисунок 11. Работа с файлами в Google Colab

12. Работа с магическими командами

```
[2]: %lsmagic

[2]: root
     line
     cell
     Find...

[4]: import time

     %time
     time.sleep(2) # Имитация длительной операции
     print("Операция завершена")

     %timeit
     time.sleep(0.01)

CPU times: total: 0 ns
Wall time: 0 ns
Операция завершена

[6]: %%writefile script.py
     print("Привет из скрипта script.py!")
     for i in range(5):
         print(i)

Writing script.py

[8]: !python script.py

Привет из скрипта script.py!
0
1
2
3
4
```

Рисунок 12. Магические команды в Jupyter

```
[10]: %ls

 Том в устройстве C не имеет метки.
 Серийный номер тома: 7813-5486

 Содержимое папки C:\Users\furrzik\laba1\notebooks

17.02.2025  23:46    <DIR>          .
16.02.2025  21:26    <DIR>          ..
17.02.2025  23:43    <DIR>          .ipynb_checkpoints
17.02.2025  23:46                90 script.py
17.02.2025  23:45                72 Untitled.ipynb
16.02.2025  23:19             4 709 Задание 1.ipynb
17.02.2025  23:45             3 537 Задание 2 с jupyterlab.ipynb
16.02.2025  21:17             1 336 Задание 2.ipynb
16.02.2025  21:23             2 998 Задачка.ipynb
16.02.2025  21:17            44 633 Пример 1.ipynb
16.02.2025  15:15             1 087 Пример 2.ipynb
               8 файлов         58 462 байт
               3 папок  389 695 152 128 байт свободно

[12]: %history

import json
import getpass
import hashlib

def import_pandas_safely():
    try:
        return __import__('pandas')
    except ImportError:
        return False


__pandas = import_pandas_safely()


def is_data_frame(v: str):
    obj = eval(v)
    if isinstance(obj, __pandas.core.frame.DataFrame) or isinstance(obj, __pandas.core.series.Series):
```

Рисунок 13. Магические команды в Jupyter

```python
import os

# 1. Создайте и сохраните текстовый файл с помощью open()
filename = "example.txt"  # Имя файла

try:
    with open(filename, "w", encoding="utf-8") as file: # Используем with для автоматического закрытия файла
        # 2. Запишите в него несколько строк текста
        file.write("Первая строка текста.\n")  # \n - символ новой строки
        file.write("Вторая строка текста.\n")
        file.write("Третья строка текста.\n")
    print(f"Файл '{filename}' успешно создан и записан.")

    # 3. Закройте файл (сделано автоматически с помощью 'with') и затем откройте его снова, считав содержимое и выведя на экран
    with open(filename, "r", encoding="utf-8") as file:
        content = file.read() # Считываем все содержимое файла в строку
        print("\nСодержимое файла:")
        print(content)

    # 4. Проверьте, существует ли файл, используя os.path.exists()
    if os.path.exists(filename):
        print(f"\nФайл '{filename}' существует.")
    else:
        print(f"\nФайл '{filename}' не существует.") # Эта ветка не должна быть выполнена, если все прошло успешно

    # 5. Удалите файл с помощью модуля os
    os.remove(filename)
    print(f"Файл '{filename}' успешно удален.")

except FileNotFoundError:
    print(f"Ошибка: Файл '{filename}' не найден.")
except Exception as e:
    print(f"Произошла ошибка: {e}")
```

```
Файл 'example.txt' успешно создан и записан.

Содержимое файла:
Первая строка текста.
Вторая строка текста.
Третья строка текста.


Файл 'example.txt' существует.
Файл 'example.txt' успешно удален.
```

Рисунок 14. Магические команды в Google Colab

```
%lsmagic
```

```
Available line magics:
%alias  %alias_magic  %autoawait  %autocall  %automagic  %autosave  %bookmark  %cat  %cd  %clear  %colors  %conda  %conf
%ll  %load  %load_ext  %loadpy  %logoff  %logon  %logstart  %logstate  %logstop  %ls  %lsmagic  %lx  %macro  %magic  %ma
%precision  %prun  %psearch  %psource  %pushd  %pwd  %pycat  %pylab  %qtconsole  %quickref  %recall  %rehashx  %reload_e
%time  %timeit  %unalias  %unload_ext  %who  %who_ls  %whos  %xdel  %xmode

Available cell magics:
%%!  %%HTML  %%SVG  %%bash  %%bigquery  %%capture  %%debug  %%file  %%html  %%javascript  %%js  %%latex  %%markdown  %%p
%%timeit  %%writefile

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

```
[7] import time

    %%time
    time.sleep(2)  # Имитация длительной операции
    print("Операция завершена")

    import time #Импортируем time еще раз, на случай если в ячейке выше его не было.

    %timeit time.sleep(0.01)
```

```
UsageError: Line magic function `%%time` not found.
```

```
[8] %%writefile script.py
        print("Привет из скрипта script.py!")
        for i in range(5):
            print(i)
```

```
Writing script.py
```

```
[9] !python script.py
```

```
  File "/content/script.py", line 1
    print("Привет из скрипта script.py!")
IndentationError: unexpected indent
```

Рисунок 15. Магические команды в Google Colab

```
!python script.py

  File "/content/script.py", line 1
    print("Привет из скрипта script.py!")
IndentationError: unexpected indent
```

```
[10] %ls

sample_data/  script.py
```

```
[11] %history

    # 5. Удалите файл с помощью модуля os
    os.remove(filename)
    print(f"Файл '{filename}' успешно удален.")

except FileNotFoundError:
    print(f"Ошибка: Файл '{filename}' не найден.")
except Exception as e:
    print(f"Произошла ошибка: {e}")
%lsmagic
 import time

    %time
    time.sleep(2) # Имитация длительной операции
    print("Операция завершена")

    %timeit
    time.sleep(0.01)
import time

start_time = time.time()  # Записываем время начала

time.sleep(2)  # Имитация длительной операции
print("Операция завершена")

end_time = time.time()  # Записываем время окончания
```

Рисунок 16. Магические команды в Google Colab

13. Работа с оболочкой системы

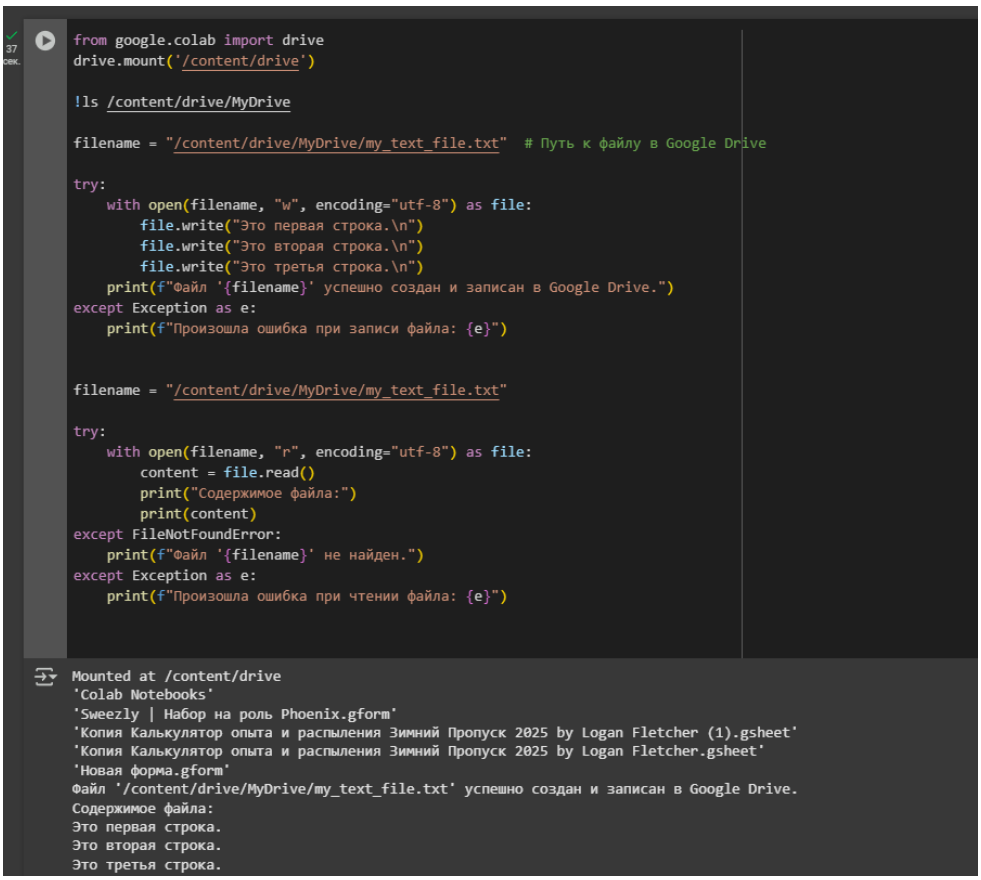
```
[21]: !mkdir test_folder
      !dir
      !where python
      !rmdir test_folder

 Том в устройстве C не имеет метки.
 Серийный номер тома: 7813-5486

 Содержимое папки C:\Users\furrzik\laba1\notebooks

17.02.2025  23:55    <DIR>          .
16.02.2025  21:26    <DIR>          ..
17.02.2025  23:50    <DIR>          .ipynb_checkpoints
17.02.2025  23:46                90 script.py
17.02.2025  23:55    <DIR>          test_folder
16.02.2025  23:19             4 709 Задание 1.ipynb
17.02.2025  23:49             3 537 Задание 2 с jupyterlab.ipynb
16.02.2025  21:17             1 336 Задание 2.ipynb
16.02.2025  21:23             2 998 Задачка.ipynb
17.02.2025  23:49            23 204 Магические команды.ipynb
16.02.2025  21:17            44 633 Пример 1.ipynb
16.02.2025  15:15             1 087 Пример 2.ipynb
17.02.2025  23:54             5 298 работа с оболочкой системы.ipynb
               9 файлов         86 892 байт
               4 папок  389 698 056 192 байт свободно
C:\Users\furrzik\anaconda3\python.exe
C:\Users\furrzik\AppData\Local\Programs\Python\Python313\python.exe
C:\Users\furrzik\AppData\Local\Microsoft\WindowsApps\python.exe

[ ]:
```

Рисунок 17. Команды в JupyterLab

```
13] !mkdir test_folder  # Создание каталога
   !ls  # Просмотр содержимого каталога (эквивалент dir в Linux)
   !which python  # Поиск исполняемого файла Python (эквивалент where в Linux)
   !rm -r test_folder  # Удаление каталога (эквивалент rmdir в Linux).  Обратите внимание на `-r`

sample_data  script.py  test_folder
/usr/local/bin/python
```

Рисунок 18. Команды в Google Colab

14.Работа с Google Drive



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

!ls /content/drive/MyDrive

filename = "/content/drive/MyDrive/my_text_file.txt"  # Путь к файлу в Google Drive

try:
    with open(filename, "w", encoding="utf-8") as file:
        file.write("Это первая строка.\n")
        file.write("Это вторая строка.\n")
        file.write("Это третья строка.\n")
    print(f"Файл '{filename}' успешно создан и записан в Google Drive.")
except Exception as e:
    print(f"Произошла ошибка при записи файла: {e}")


filename = "/content/drive/MyDrive/my_text_file.txt"

try:
    with open(filename, "r", encoding="utf-8") as file:
        content = file.read()
        print("Содержимое файла:")
        print(content)
except FileNotFoundError:
    print(f"Файл '{filename}' не найден.")
except Exception as e:
    print(f"Произошла ошибка при чтении файла: {e}")
```

```
Mounted at /content/drive
'Colab Notebooks'
'Sweezly | Набор на роль Phoenix.gform'
'Копия Калькулятор опыта и распыления Зимний Пропуск 2025 by Logan Fletcher (1).gsheet'
'Копия Калькулятор опыта и распыления Зимний Пропуск 2025 by Logan Fletcher.gsheet'
'Новая форма.gform'
Файл '/content/drive/MyDrive/my_text_file.txt' успешно создан и записан в Google Drive.
Содержимое файла:
Это первая строка.
Это вторая строка.
Это третья строка.
```

Рисунок 19. Код для работы с Google Drive

15. Зафиксированы изменения на репозитории и отправлены на сервер GitHub

```
C:\Users\furrzik\laba1>git add notebooks

C:\Users\furrzik\laba1>git add .

C:\Users\furrzik\laba1>git commit -m "Jupyter"
[main ef941aa] Jupyter
 5 files changed, 650 insertions(+)
 create mode 100644 "notebooks/\320\227\320\260\320\264\320\260\320\275\320\270\320\265 1.ipynb
 create mode 100644 "notebooks/\320\227\320\260\320\264\320\260\320\275\320\270\320\265 2.ipynb
 create mode 100644 "notebooks/\320\227\320\260\320\264\320\260\321\207\320\272\320\260.ipynb"
 create mode 100644 "notebooks/\320\237\321\200\320\270\320\274\320\265\321\200 1.ipynb"
 create mode 100644 "notebooks/\320\237\321\200\320\270\320\274\320\265\321\200 2.ipynb"

C:\Users\furrzik\laba1>git pull
Already up to date.

C:\Users\furrzik\laba1>git push
info: please complete authentication in your browser...
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 12 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 33.01 KiB | 16.50 MiB/s, done.
Total 8 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 1 local object.
To https://github.com/furrzik1337/laba1.git
   1ff2918..ef941aa  main -> main
```

Рисунок 20. Отправка на сервер GitHub

Ответы на контрольные вопросы:

1. Какие основные отличия JupyterLab от Jupyter Notebook?

JupyterLab – это мощная интерактивная среда для разработки, анализа данных и документирования исследований. Она объединяет функциональность Jupyter

Notebook, текстового редактора, терминала и файлового менеджера в одном интерфейсе, что делает ее удобным инструментом для научной и образовательной деятельности.

Jupyter Notebook: представляет собой отдельные веб-страницы с линейной последовательностью ячеек. Окружение ограничено одной тетрадью.

JupyterLab: это полноценная интегрированная среда (IDE) с вкладками, панелями, файловым менеджером, терминалом и редактором кода. Можно открывать и редактировать несколько файлов одновременно

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

1. В меню выберите File → New → Notebook .

2. Выберите доступное ядро (обычно Python).

3. Отроется новая тетрадь, состоящая из ячеек (Cells).

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.

Для того, чтобы изменить тип ячейки на Markdown нужно нажать **M**. Чтобы переключить на тип ячейки код- Y.

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

Запустить ячейку- Shift + Enter

Добавить новую ячейку ниже- B

Удалить текущую ячейку-D D

Изменить тип ячейки на Markdown- M

Изменить тип ячейки на код-Y

Переключение между режимами (редактирование/командный)- Enter/ Esc

5. Как запустить терминал или текстовый редактор внутри JupyterLab?

JupyterLab позволяет открывать терминал (File → New → Terminal) и выполнять команды оболочки.

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

В Jupyter Notebook можно работать только с .ipynb -файлами.

В JupyterLab можно работать с разными типами файлов: .ipynb , .py , .csv , .md и даже .json , .yaml и .txt.

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

1. **Просмотр установленных ядер**. Их можно посмотреть на странице Launcher или через терминал, запустив команду jupyter kernelspec list.

2. **Управление запущенными ядрами**. Для этого нужно использовать вкладку «Запущенные терминалы и ядра» (Running Terminals and Kernels). На ней можно закрыть или выключить открытые вкладки, запущенные ядра и терминалы по отдельности, наведя курсор на правую сторону вкладки и нажав на появившуюся кнопку X.

3. **Использование кнопки меню Kernel**. Она предлагает набор опций для управления ядрами: перезапустить, выключить и изменить ядра.

8. **Каковы основные возможности системы вкладок и окон в интерфейсеJupyterLab?**

Основные возможности системы вкладок и окон в интерфейсе JupyterLab**:**

Основная рабочая область. Позволяет группировать документы (блокноты, текстовые файлы и пр.) и другие инструменты (терминалы,

консоли и т. д.) в виде панелей с вкладками, размер и расположение которых можно изменить перетаскиванием.

Вкладка Tabs в боковой панели. Показывает список открытых документов и инструментов в рабочей области с возможностью переключения.

Режим работы с отдельным документом. Позволяет сфокусироваться на отдельном документе и инструменте без того, чтобы закрывать все остальные вкладки в рабочей области. Его можно запустить из панели View («Single-Document Mode») или воспользоваться сочетанием горячих клавиш (по умолчанию Ctrl+Shift+Enter).

Настройка рабочего пространства. В JupyterLab есть возможность разделить окна по горизонтали и вертикали.

Кроме того, для навигации и запуска инструментов в JupyterLab можно использовать горячие клавиши, которые можно настроить

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

%timeit my_function() Измеряет время выполнения команды.

%timeit. Запускает команду несколько раз и вычисляет среднее время выполнения. Пример использования:

%timeit sum (range (100))

%%time. Даёт информацию о единичном запуске кода в ячейке.

Например,

```
import time
start_time = time.time()
# измеряемый код
sum (range (100))
end_time = time.time()
elapsed_time = end_time - start_time
print(f'Время запуска: {elapsed_time:.2f} секунд')
```

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

Некоторые магические команды, которые позволяют запускать код на других языках программирования в JupyterLab:

%%python2**,** %%python3**,** %%R**,** %%bash**.** Обозначают начало ячейки с кодом на определённом языке программирования (например, Python, R, Bash).

%%latex**.** Позволяют получать отрисовку ячеек с кодом в LaTeX.

%%html**,** %%javascript (или **%%js)**, %%markdown**,** %%ruby**,** %%sh**.** Аналогично служат команды для других языков программирования, например, Ruby, Pearl, JavaScript.

11. Какие основные отличия Google Colab от JupyterLab?

Google Colab (или Google Colaboratory) – это облачная среда для работы с Jupyter Notebook, предоставляемая Google. Она позволяет запускать код на удаленных серверах, что особенно полезно для задач машинного обучения и анализа данных. Google Colab предоставляет доступ к GPU и TPU бесплатно (с ограничениями), а также интегрируется с Google Диском.

12. Как создать новый ноутбук в Google Colab?

1. Перейдите в Файл → Новый ноутбук.

2. Откроется рабочая область с первой ячейкой.

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

Код (Code) – для написания и выполнения Python-кода.

Текст (Markdown) – используется для оформления документации, пояснений и формул (LaTeX).

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

Запустить ячейку- Shift + Enter

Добавить новую ячейку ниже- Ctrl+ M B

Удалить текущую ячейку- Ctrl + M D

Изменить тип ячейки на Markdown- Ctrl + M M

Изменить тип ячейки на код- Ctrl + M Y

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

Google Colab поддерживает загрузку и сохранение файлов через локальный компьютер, Google Диск, а также с помощью URL-ссылок и API.

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

Google Colab позволяет работать с файлами на Google Диске и загружать файлы в локальное окружение.

Подлючение Google Диска:

from google.colab import drive

drive.mount('/content/drive')

После выполнения появится ссылка, по которой нужно авторизоваться.

!ls- Просмотр списка файлов в текущей директории

!pwd- Вывод текущей директории

!rm filename- Удаление файла

!mkdir new_folder- Создание папки

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

В Google Colab для загрузки файлов из локального компьютера можно использовать следующие команды:

Использование модуля files **из библиотеки** google.colab

Uploaded =f iles.upload() После выполнения этой команды появится кнопка для выбора файлов на локальном компьютере. Загруженные файлы будут доступны в переменной uploaded.

2.**Использование fles.download** Чтобы скачать файл из Colab на локальный компьютер, можно использовать данную команду.

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

Чтобы посмотреть список файлов, хранящихся в среде Google Colab, используется команда! ls.

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

1.%time: измеряет время одной строки (%time sum (range (1000))

2. %timeit: выполняет строку несколько раз до точности (%timeit sum (range (10000)).

3.%%time: измеряет время всего блока кода.

Пример:

%%time

total = sum (range(1000));

4.%%timeit: выполняет блок кода несколько раз.

Пример:

%%timeit

total = sum (range(10000));

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

В Google Colab выберите Среда выполнения> Изменить среду выполнения, затем выберите GPU и нажмите Сохранить.

Вывод: в ходе лабораторной работы были приобретены навыки работы с Jupyter Notebook, JupyterLab и Google Colab

Ссылка на GitHub: https://github.com/furrzik1337/laba1