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

Институт перспективной инженерии
Департамент цифровых, роботехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
дисциплины «Искусственный интеллект и машинное обучение»
Вариант 7

Выполнила:
Еремина Татьяна Евгеньевна
2 курс, группа ИВТ-б-о-24-1,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____
(подпись)

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

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2026 г.

Тема: работа Jupyter Notebook, JupyterLab и Google Colab.

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

Практическая часть:

Исходный репозиторий: https://github.com/TL-hash/ML_lab1/tree/main.

Примеры.

Выполним ряд примеров для демонстрации возможностей Jupyter Notebook, JupyterLab и Google Colab:

1. В Jupyter Notebook код необходимо вписывать в пустые ячейки, которые можно добавлять и изменять вид содержимого.

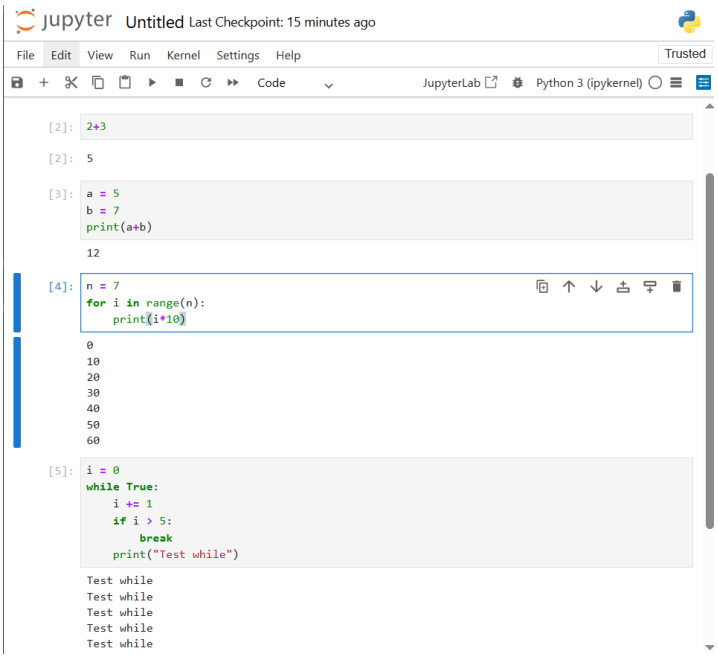

Рисунок 1 - Работа с ячейками в Jupyter Notebook

2. Используя библиотеку matplotlib, можно строить графики.

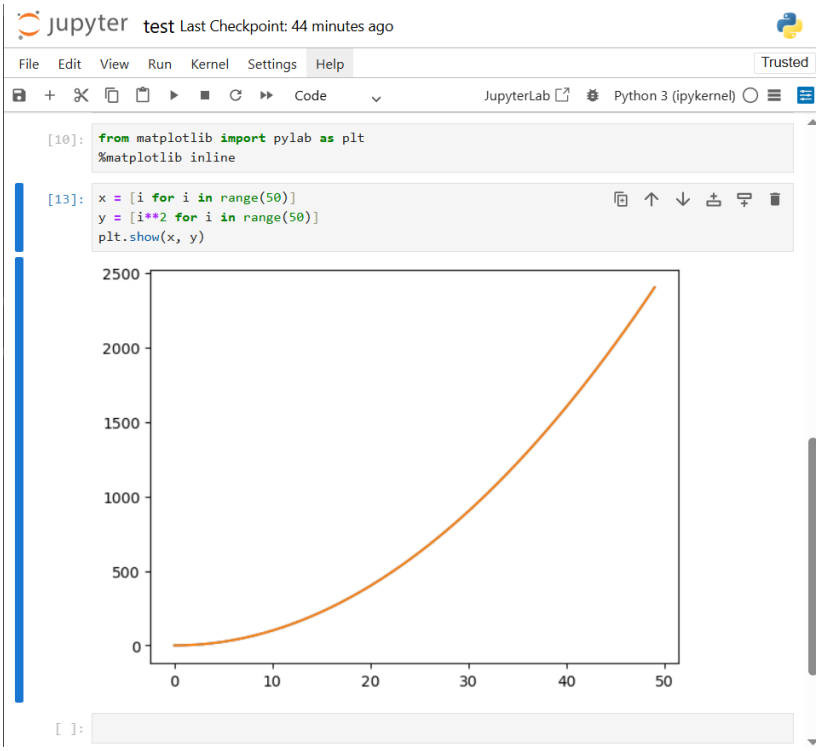

Рисунок 2 - Использование библиотеки matplotlib

3. Для работы с переменными окружения используется магическая команда %env.

```
[15]: %env TEST = 5

env: TEST=5
```

Рисунок 3 - магическая команда %env

4. Запуск .py и .ipynb осуществляется командой %run, получить информацию о времени работы кода в рамках одной ячейки можно с помощью %%time, %timeit запускает переданный код 100000 и выводит информацию о среднем значении трех наиболее быстрых точек.

```
[16]: %run C:/Users/User/OneDrive/Документы/test_py.py

HelloHelloHelloHelloHello
```

Рисунок 4 - Команда %run

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

CPU times: total: 15.6 ms
Wall time: 5.03 s

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

7.84 µs ± 350 ns per loop (mean ± std. dev. of 7 runs, 100,000 loops each)
```

Рисунок 5 - Команды %%time, %timeit

5. Использование Markdown в JupyterLab для форматирования текста и формул.

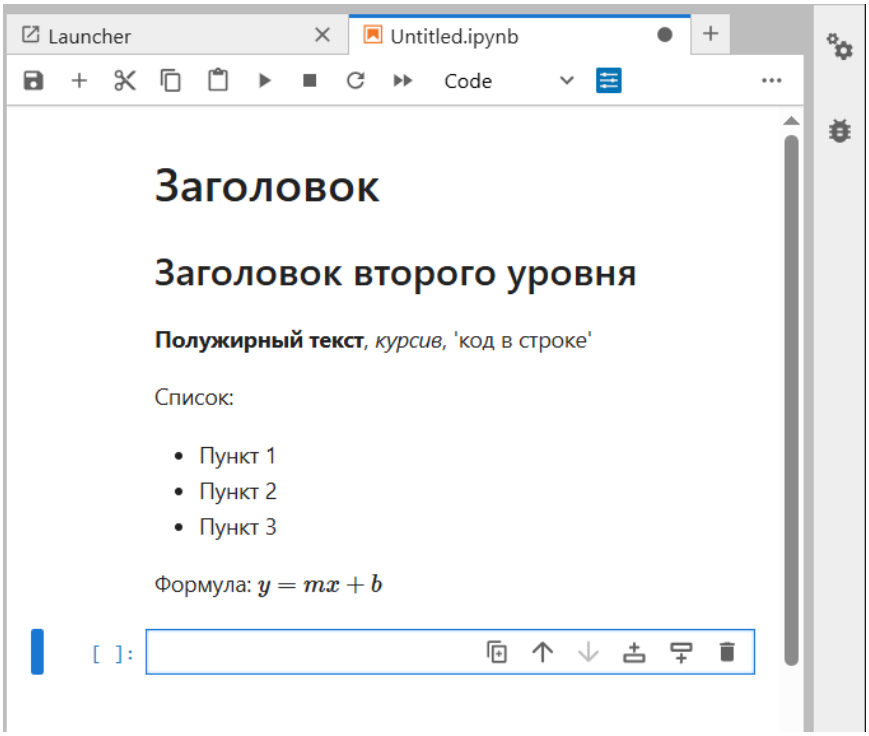

Рисунок 6 - Markdown в JupyterLab

6. Значок магических команд % означает их применение к строке кода, %% - к всей ячейке.

```
[1]: %timeit sum(range(1000))
     68.4 µs ± 2.13 µs per loop (mean ± std. dev. of 7 ru
     ns, 10,000 loops each)
[2]: %pwd
[2]: 'C:\\Users\\User'
[3]: %%time
     total = 0
     for i in range(10**6):
         total += i
     CPU times: total: 844 ms
     Wall time: 841 ms
```

Рисунок 7 - Различие %% и %

7. В JupyterLab также можно строить графики с подключением библиотеки matplotlib.

```
[4]: 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()
```

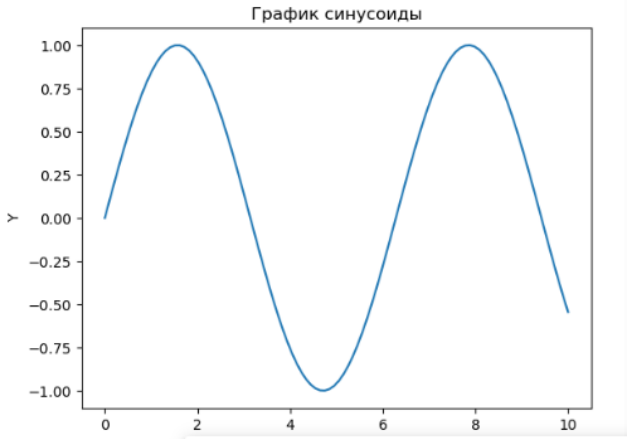

Рисунок 8 - Библиотека matplotlib

8. Google Colab также представляет ряд возможностей, например, форматирование текста в Markdown.

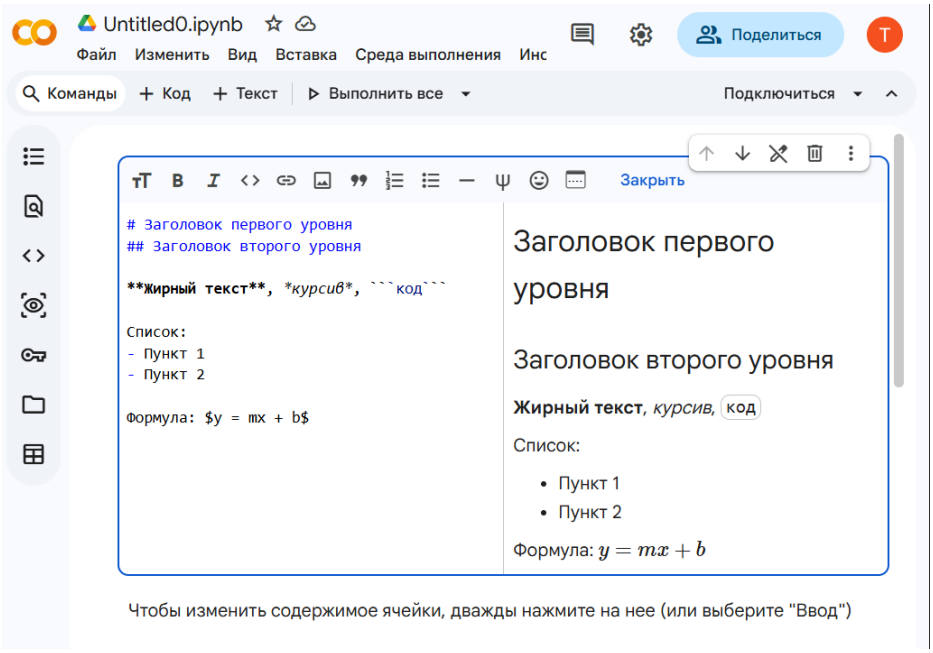

Рисунок 9 - Markdown в Google Colab

9. В Google Colab возможна работа с файловой системой.

Рисунок 10 - Работа с файловой системой

10. Доступны магические команды.

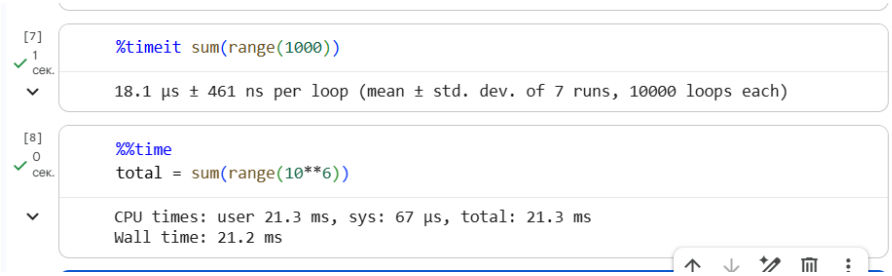

Рисунок 11 - Магические команды

11. А также библиотеки, например, matplotlib.

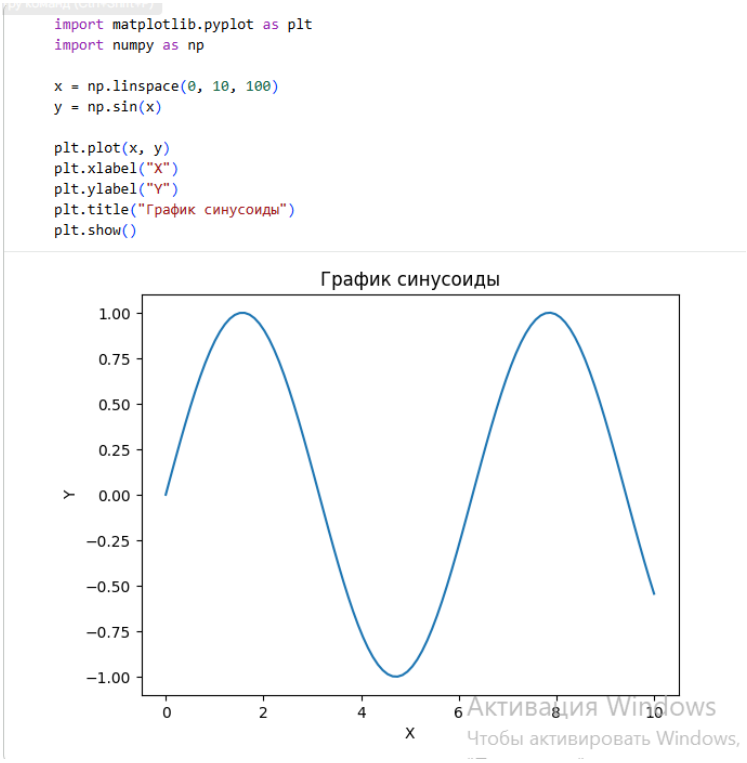

Рисунок 12 - Библиотека matplotlib в Google Colab

Задания практической работы.

Задание 1. Работа с ячейками и Markdown.

Создайте новую Markdown-ячейку и:

- Напишите заголовок "Практическое задание "1"
- Добавьте жирный и курсивный текст.

- Создайте нумерованный и маркированный списки.
- Вставьте, согласно индивидуальному заданию (ряды Тейлора для функции экспоненты, вариант 7).
- Вставьте изображение через ![Описание](URL)

Создайте ячейку Python-кода и:

- Запросите у пользователя его имя с помощью input() .
- Выведите приветствие: "Привет, <имя>! Добро пожаловать в JupyterLab / Google Colab!"
- Запустите ячейку (Shift + Enter).

Решение:

1. В Jupyter Notebook создаем новый блокнот, выбираем ядро (Python по умолчанию).
2. В панеле меню выбираем изменить тип ячейки на Markdown.
3. Записываем текст согласно условию задания: # - выделяет заголовки разного уровня, * - делает текст жирным, ** - курсив, $ - выделяет формулу, ![Описание](URL) – вставляет картинку по ссылке из открытого источника.

Листинг ячейки:

```
# Практическое задание №1
**Ряды Тейлора для функции *экспоненты***

## Ряды Тейлора существуют для следующих функций:
- экспоненты
- синуса
- косинуса
- натурального логарифма

## План выполнения:
1. Написать функцию вычисления ряда Тейлора для $e^x$
2. Вставить изображение разложения функции экспоненты по ряду Тейлора
```

Формула ряда Тейлора

$$ e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots $$

![График разложения функции экспоненты по ряду Тейлора](https://studfile.net/html/2706/180/html_3F8gg3iRbE.XORp/img-mScTp7.png)

Практическое задание №1

Ряды Тейлора для функции *экспоненты*

Ряды Тейлора существуют для следующих функций:

- экспоненты
- синуса
- косинуса
- натурального логарифма

План выполнения:

1. Написать функцию вычисления ряда Тейлора для e^x
2. Вставить изображение разложения функции экспоненты по ряду Тейлора

Формула ряда Тейлора

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

MATHCAD

Приближение функций рядом Тейлора

Используется оператор series из меню кнопки 'Символические операторы'

$f1(x) = \sin(x)\ \text{series},x,8 \rightarrow x - \frac{1}{6}\cdot x^3 + \frac{1}{120}\cdot x^5 - \frac{1}{5040}\cdot x^7$

$f2(x) = \sin(x)\ \text{series},x,16 \rightarrow x - \frac{1}{6}\cdot x^3 + \frac{1}{120}\cdot x^5 - \dots$

$y(x) = \exp(-0.3\cdot x^2)$

$f3(x) = y(x)\ \text{series},x,8 \rightarrow 1 - .3\cdot x^2 + 4.5\cdot 10^{-2}\cdot x^4 - 4.5\cdot 10^{-3}\cdot x^6$

$f4(x) = y(x)\ \text{series},x,16 \rightarrow 1 - .3\cdot x^2 + 4.5\cdot 10^{-2}\cdot x^4 - \dots$

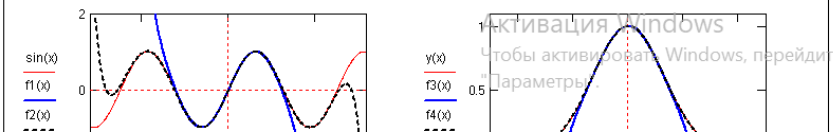

Рисунок 13 - Результат выполнения

Задание 2. Работа с файлами.

- Создайте и сохраните текстовый файл с помощью open()
- Запишите в него несколько строк текста.
- Закройте файл и затем откройте его снова, считав содержимое и выведя на экран.
- Проверьте, существует ли файл, используя os. path.exists()
- Удалите файл с помощью модуля os.

Решение:

1. Для выполнения работы воспользуемся JupyterLab.

2. Импортируем модуль os, который предоставляет функции для взаимодействия с операционной системой.

3. open("example.txt", "w") открывает файл example.txt в режиме записи ("w"), with – менеджер контекста, который автоматически закрывает файл после завершения блока, f.write() записывает строки в файл.

4. open("example.txt", "r") открывает файл в режиме чтения ("r"), f.read() считывает всё содержимое файла в переменную content.

5. os.path.exists("example.txt") проверяет, существует ли файл.

6. os.remove("example.txt") удаляет файл из файловой системы.

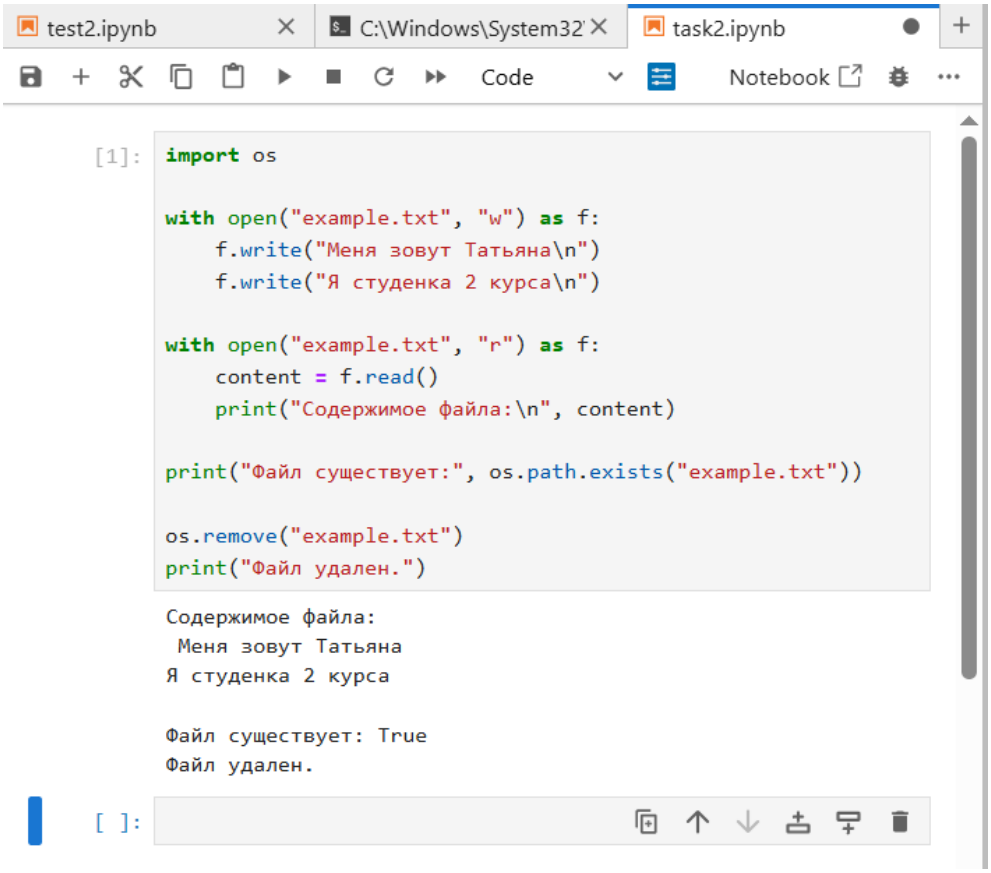

Рисунок 14 - Результат выполнения

Задание 3. Магические команды Jupyter.

- Выведите список всех доступных магических команд (%lsmagic).
- Используйте *time и **timeit для измерения времени выполнения кода.

- Создайте Python-скрипт в Jupyter (%%writefile script.py) и выполните его через !python script.py.
- Выведите список файлов в текущей директории с помощью %ls.
- Используйте *history для просмотра истории команд.

Решение:

1. С помощью %lsmagic выводим список магических команд.
2. Используем магическую команду %%time для приостановки выполнения программы на 2 секунды. Функция time.sleep(2) принимает в качестве аргумента количество секунд, на которое нужно приостановить выполнение.
3. numpy.random.normal(size=100) генерирует массив из 100 случайных чисел, взятых из нормального распределения, %timeit автоматически запускает этот код много раз и выводит статистику о времени выполнения.
4. %%writefile записывает код в файл, !python выполняет его.
5. %ls выводит список файлов в текущей директории.
6. %history используется для просмотра истории команд.

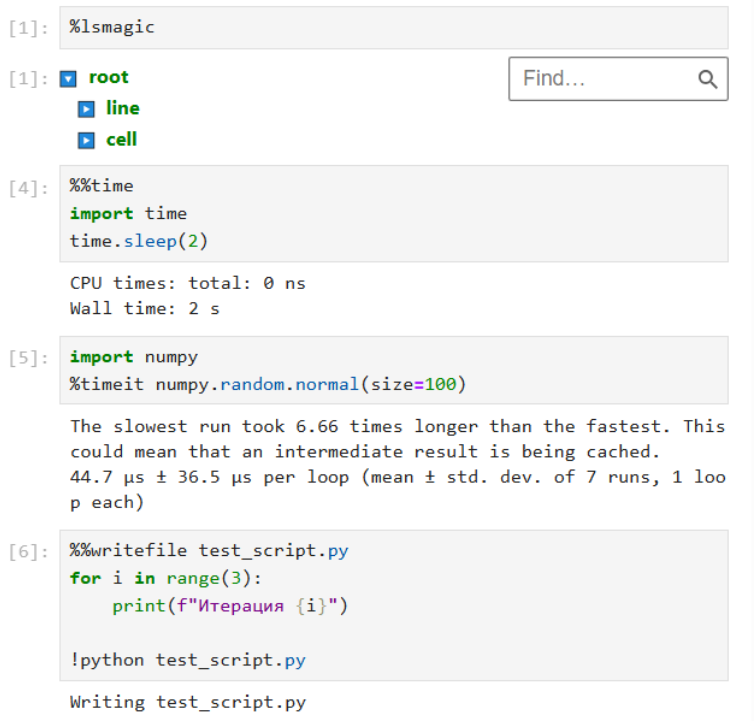

Рисунок 15 - Результат выполнения

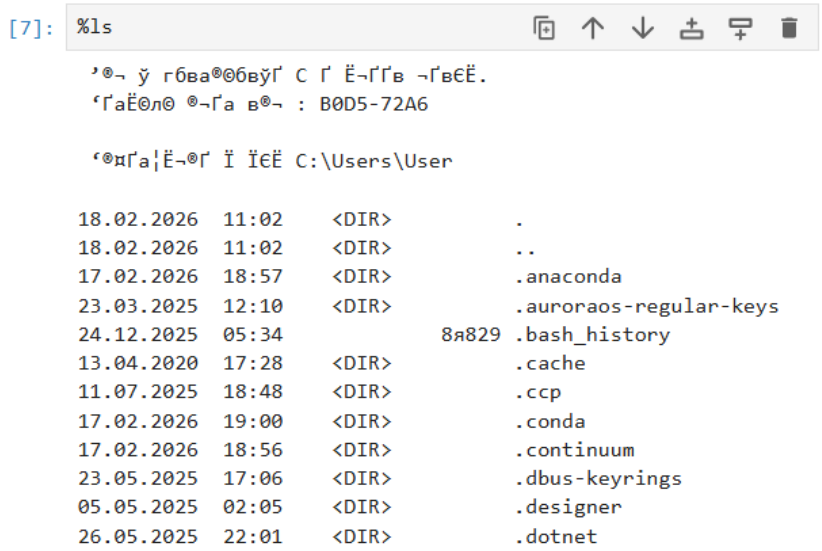

Рисунок 16 - Команда %ls

Рисунок 17 - Команда %history

Задание 4. Взаимодействие с оболочкой системы.

- Выведите список файлов в текущей директории с помощью !ls .
- Проверьте, какой Python используется (!which python).
- Создайте папку test_folder (!mkdir test_folder) и убедитесь, что она появилась.
- Переместите файл в новую папку и удалите его.
- Очистите вывод в ячейке (!clear).

Решение:

1. Для работы воспользуемся Google Colab.

2. Выведем список файлов в текущей директории с помощью !ls.

3. Проверим, какой Python используется с помощью !which python.

4. Создадим папку test_folder, внутри разместим текстовый файл командой !torch.

5. Переместим файл в новую папку и удалим старую.

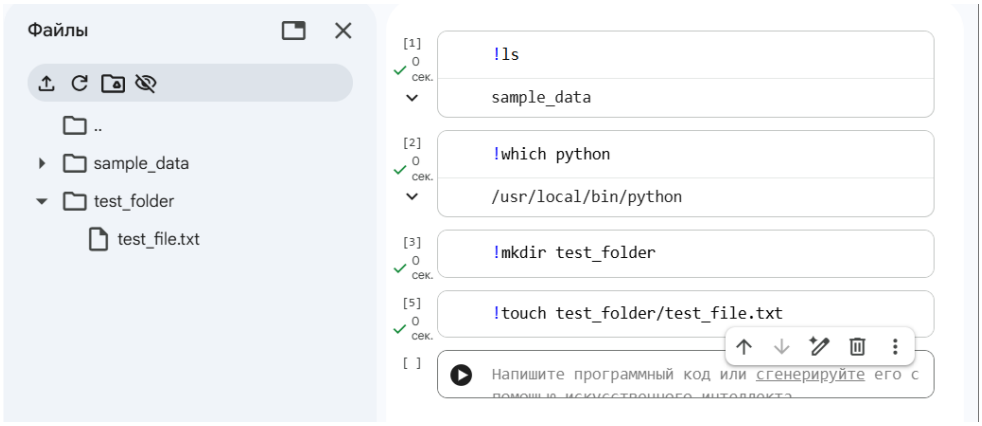

Рисунок 18 - Создание папки с файлом

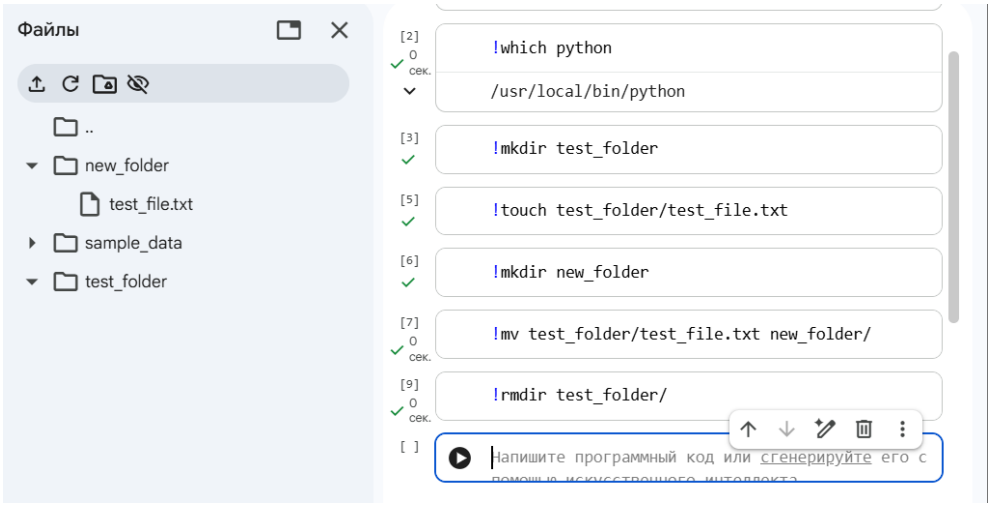

Рисунок 19 - Перемещение файла и удаление папки

Задание 5. Работа с Google Drive в Google Colab.

1. Подключим Google Drive к Colab с помощью команды: from google.colab import drive.

2. Проверим, что диск успешно подключился, используя !ls /content/drive/MyDrive.

3. Создайте и сохраните текстовый файл в Google Drive: откроем файл my_text_file.txt в режиме записи, запишем в него несколько строк текста, закроем файл и убедимся, что он появился в папке MyDrive.

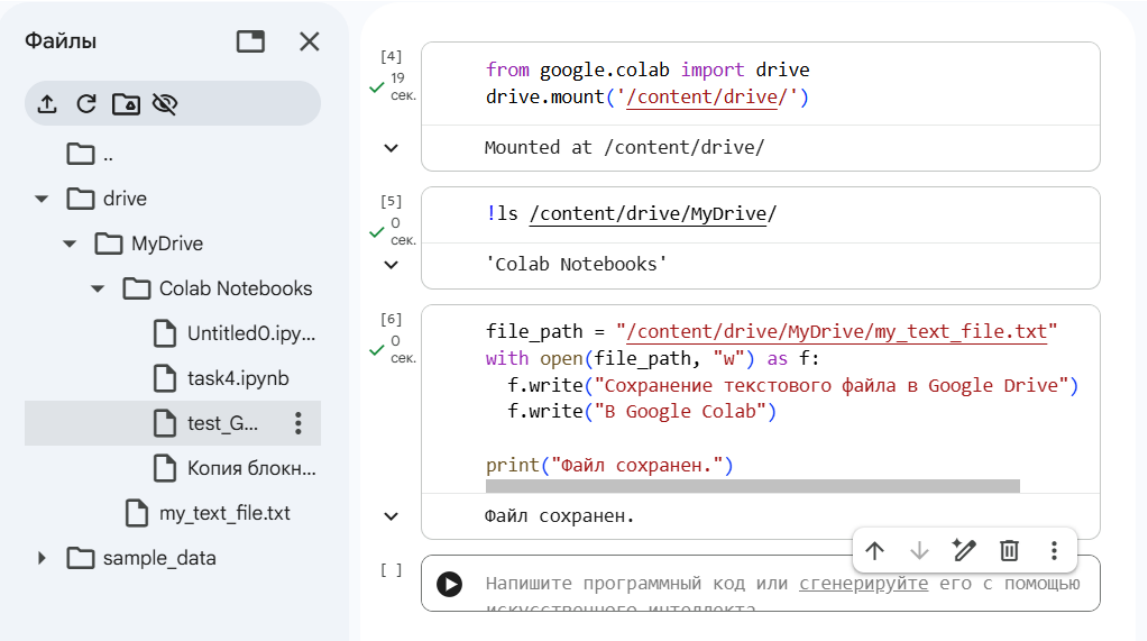

Рисунок 20 - Подключение Google Drive и запись информации в файл

4. Прочитаем файл из Google Drive: откроем ранее созданный файл, считаем его содержимое и выведем в ячейке.

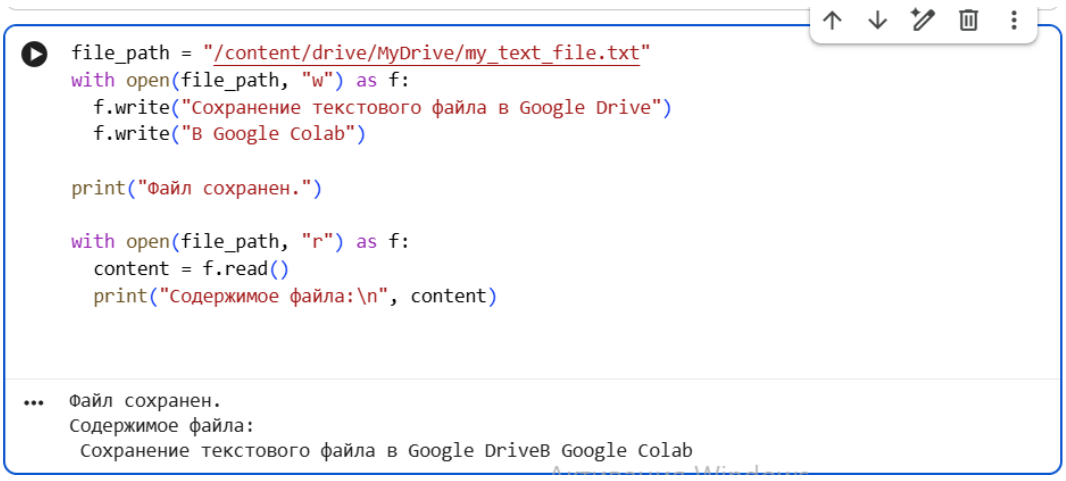

Рисунок 21 - Вывод текста из файла

5. Создадим и сохраним CSV-файл вручную: запишем список со студентами в CSV-файл в Google Drive.

```
students = [
    ["Еремина Т.Е.", 19, "ИВТ-б-о-24-1"],
    ["Петров", 20, "ИВТ-б-о"]
]

csv_path = "/content/drive/MyDrive/student.csv"
with open(csv_path, "w") as f:
  for student in students:
    f.write(",".join(map(str, student)) + "\n")

print("Файл сохранен.")
```

```
Файл сохранен.
Содержимое файла:
 Сохранение текстового файла в Google DriveB Google Colab
Файл сохранен.
```

Рисунок 22 - Запись CSV-файла в Google Drive

6. Загрузим из Google Drive в Colab файл с помощью file.upload().

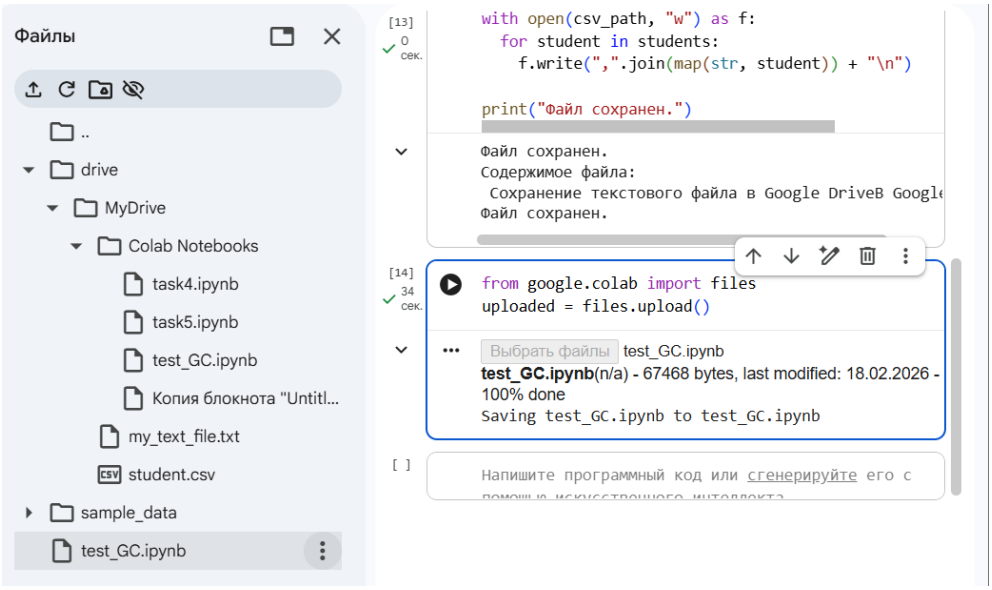

Рисунок 23 - Загрузка файла file.upload()

Контрольные вопросы:

1. Какие основные отличия JupyterLab от Jupyter Notebook?

JupyterLab – это более современная и расширенная среда разработки, тогда как Jupyter Notebook – это более старая, упрощенная версия. Основные отличия:

- JupyterLab предоставляет полноценную IDE с поддержкой нескольких вкладок, панелей и окон
- JupyterLab позволяет работать одновременно с ноутбуками, текстовыми файлами, терминалом и другими инструментами

- JupyterLab имеет более гибкую систему макетов (можно перетаскивать и изменять размер панелей)
- JupyterLab поддерживает больше языков программирования "из коробки"

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

Нажать на иконку "New Launcher" (круглый значок с плюсом) в левом меню, затем выбрать "Notebook" и указать язык программирования.

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

В JupyterLab поддерживаются:

- Code (код)
- Markdown (текст)
- Raw NBConvert (сырой текст)

Переключение: выделить ячейку и нажать:

- Y – для кода
- M – для Markdown
- R – для Raw

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

Shift + Enter – выполнить текущую ячейку и перейти к следующей

Ctrl + Enter – выполнить текущую ячейку и остаться в ней

Alt + Enter – выполнить текущую ячейку и создать новую после нее

5. Как запустить терминал или текстовый редактор внутри JupyterLab?

Нажать на иконку "New Launcher" в левом меню, затем выбрать "Terminal" для терминала или "Text File" для текстового редактора.

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

В левой части интерфейса JupyterLab есть панель навигации по файловой системе, где можно:

- Создавать, переименовывать, удалять файлы и папки
- Перетаскивать файлы между папками
- Просматривать содержимое файлов
- Использовать терминал для сложных операций

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

В правом верхнем углу есть индикатор ядра. Нажав на него, можно:

- Выбрать другое ядро
- Перезапустить ядро
- Остановить ядро
- Перезапустить с очисткой переменных

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

JupyterLab позволяет:

- Работать с несколькими вкладками одновременно
- Перетаскивать вкладки для создания панелей
- Разделять окно на несколько частей (например, ноутбук и терминал)
- Сохранять макет окон для последующего использования

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

%time – измеряет время выполнения одной строки

%%time – измеряет время выполнения всей ячейки

%timeit – измеряет время выполнения несколько раз и дает среднее значение

%%timeit – то же, что и %timeit, но для всей ячейки

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

%load_ext – загружает расширение для поддержки другого языка

%sql – для работы с SQL

%R – для работы с R

%bash – для выполнения команд bash

%javascript – для выполнения JavaScript

11. Какие основные отличия Google Colab от JupyterLab?

Google Colab работает в облаке, не требуя установки на локальном компьютере. Предоставляет бесплатный доступ к GPU и TPU, интегрирован с Google Drive для хранения файлов, имеет более простой интерфейс, автоматически сохраняет изменения.

12. Как создать новый ноутбук в Google Colab?

Нужно перейти на colab.research.google.com и нажать "Файл" – "Создать ноутбук".

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

Доступны два типа ячеек: Code (код) и Text (текст)

Переключение: выделить ячейку и нажать "Code" в выпадающем меню (сверху) или Ctrl + M, затем M для текста или Y для кода.

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

Shift + Enter – выполнить текущую ячейку и перейти к следующей

Ctrl + Enter – выполнить текущую ячейку и остаться в ней

Alt + Enter – выполнить текущую ячейку и создать новую после нее

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

Загрузка:

- Через "Файл" - "Загрузить файлы"
- Через код: from google.colab import files; files.upload()

Сохранение:

- Через "Файл" - "Скачать ноутбук"
- Через код: files.download('filename')
- Автоматическое сохранение в Google Drive

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

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?

Через команду !ls.

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

%time – измеряет время выполнения одной строки

%%time – измеряет время выполнения всей ячейки

%timeit – измеряет время выполнения несколько раз и дает среднее значение

%%timeit – то же, что и %timeit, но для всей ячейки

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

Перейти в "Редактировать" - "Настроить среду выполнения", в открывшемся окне выбрать "Тип оборудования" и установить "GPU" или "TPU", нажать "Сохранить".

Проверка использования GPU:

import tensorflow as tf

print("GPU доступен:", tf.test.is_gpu_available())

Вывод: были исследованы базовые возможности интерактивных оболочек Jupyter Notebook, JupyterLab и Google Colab для языка

программирования Python. В процессе работы освоены основные операции с ячейками кода и текста, изучены методы форматирования с помощью языка Markdown, освоены принципы работы с файловой системой и магические команды для измерения времени выполнения кода, получены навыки создания и редактирования ноутбуков, работы с переменными окружения, выполнения внешних скриптов, а также взаимодействия с облачным хранилищем Google Drive.