

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:  
Жоелилала Максим Милахарисонович  
2 курс, группа ИТС-б-о-23-1,  
11.03.02 «Инфокоммуникационные  
технологии и системы связи», очная  
форма обучения

---

(подпись)

Проверил:  
Доцент департамента цифровых,  
робототехнических систем и  
электроники 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. Ознакомились с теоретическим материалом
2. Создание репозитория

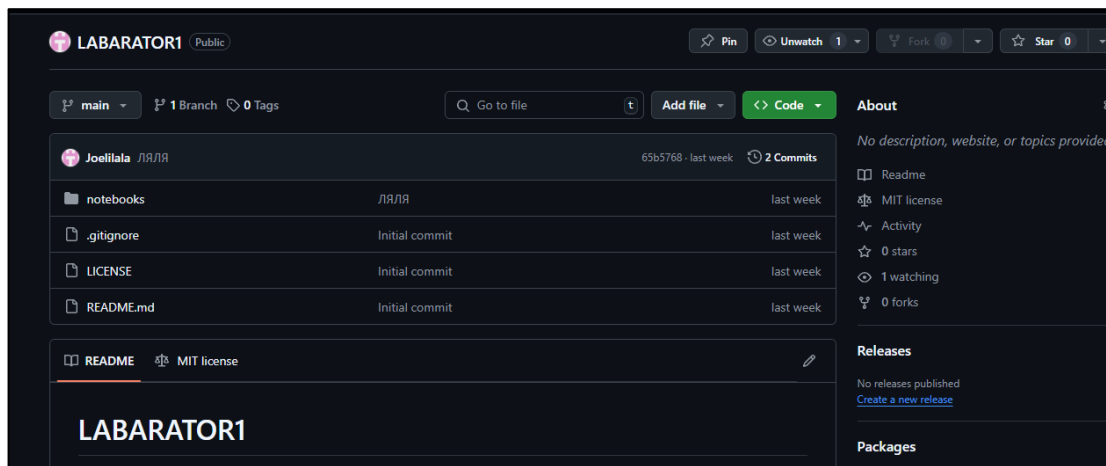


Рисунок 1. Репозиторий

3. Выполнено клонирование репозитория

```
C:\Users\Home>git clone https://github.com/Joelilala/LABARATOR1.git
Cloning into 'LABARATOR1'...
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.

C:\Users\Home>
```

Рисунок 2. Клонирование

4. Проработан примеры работы с Jupyter

```
[1]: 3+2

[1]: 5

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

12

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

0
10
20
30
40
50
60

[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 3. Проработка примеров

## 5. Проработан пример работы с JupyterLab

### Заголовок первого уровня

### Заголовок второго уровня

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

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

```
[ ]: |
```

Рисунок 4. Проработка примера

## 6. Проработан пример с построением графика

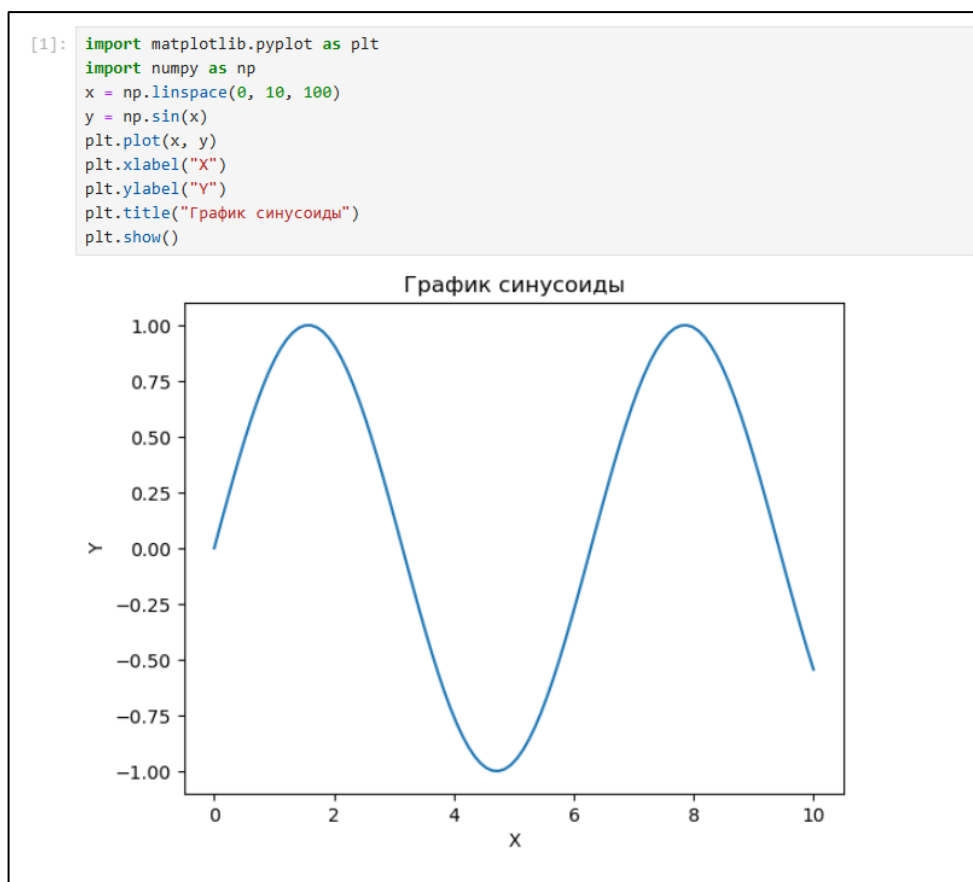


Рисунок 5. Пример с графиком

## 7. Пример работы с Google Colab

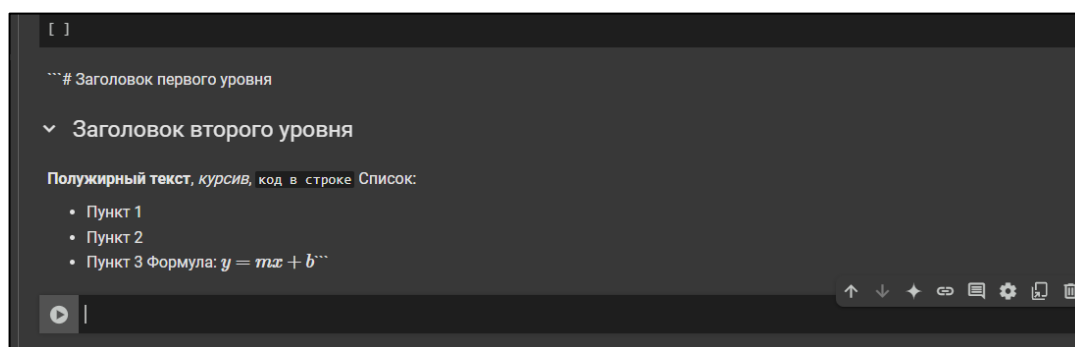


Рисунок 6. Google Colab

## 8. Выполнено практическое задание с Markdown-ячейкой

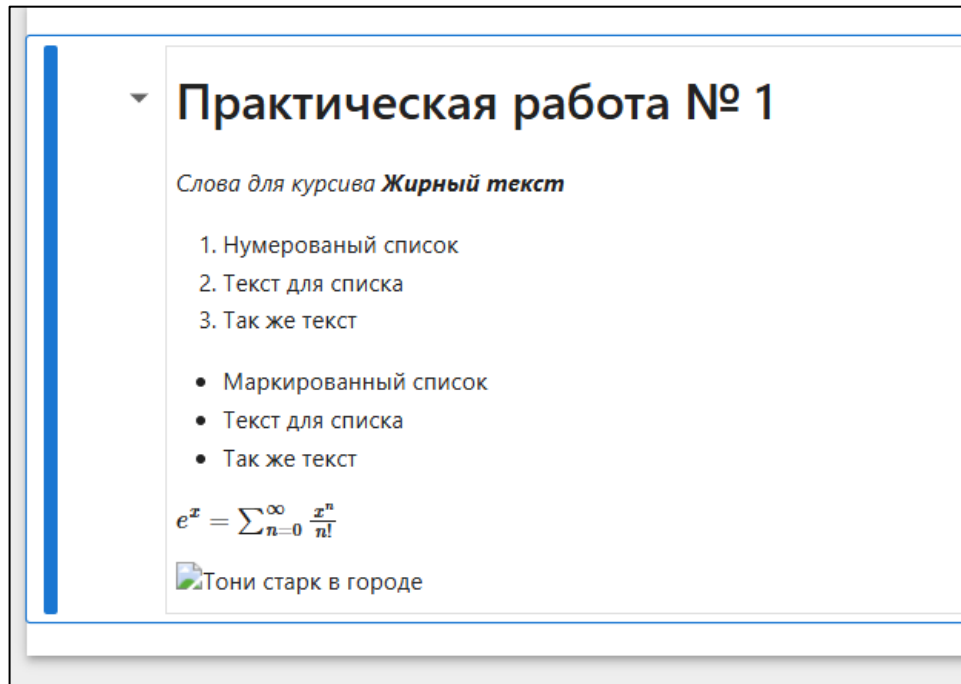


Рисунок 7. Markdown-ячейка

9. Выполнено практическое задание с использованием ячейки Python-кода



Рисунок 8. Python-код

10. Решена простенькая задача

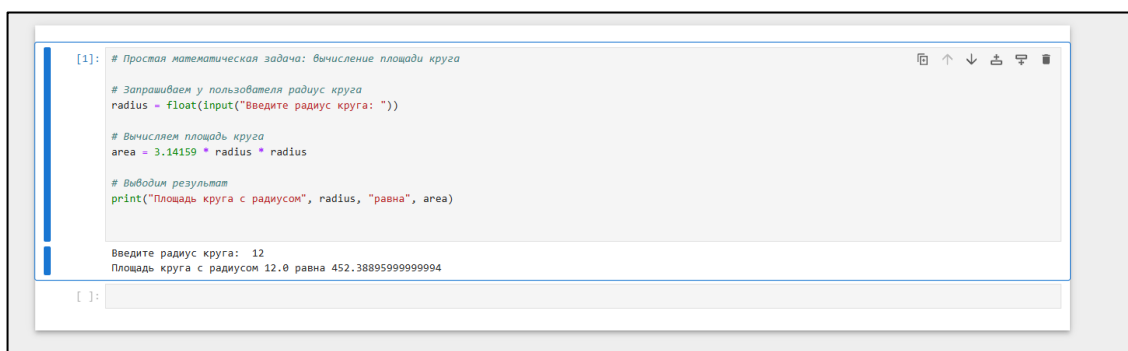


Рисунок 9. Задача

11. Выполнено задание с открытием файла

```
[1]: 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(f"\nСодержимое файла:")
        print(content)

    # 4. Проверьте, существует ли файл, используя os.path.exists()
    if os.path.exists(filename):
        print(f"Файл '{filename}' существует.")
    else:
        print(f"Файл '{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. Задание с JupyterLab

## 12. Работа с магическими командами

```
[1]: %lsmagic

[1]: root
line
cell

[2]: 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

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

Writing script.py

[4]: !python script.py

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

Рисунок 11. Магические команды

```
4
[5]: %ls

      *~ 9 r6aa*06ajr' c f E-ffa -fueE.
      'faEen0 ~-fa ue~ : DEBF-AE32

      *0Hfa'E~0f I IEE C:\Users\K~r\LABRATOR1\note books

      18.02.2025 01:28 <DIR>      .
      18.02.2025 01:28 <DIR>      ..
      18.02.2025 01:28 <DIR>      .ipynb_checkpoints
      18.02.2025 01:28      90 script.py
      14.02.2025 12:41      1s187 *9f~E~ = = 9 1.ipynb
      18.02.2025 01:21      2s187 *9f~E~ = IaE~fal.ipynb
      18.02.2025 01:27      3s524 $ H E~ 2.ipynb
      18.02.2025 01:26      1s661 $ H з .ipynb
      18.02.2025 01:27      72 H 3Es e~ ~ Hn.ipynb
      18.02.2025 01:25      1s501 UaEjy~06ajf~f~.ipynb
      18.02.2025 01:21      1s076 UaE~fa 2.ipynb
      18.02.2025 01:24      40s268 UaE~fa 3.ipynb
      9 д 0~*9y      51s486 9 0e
      3 I I~e 121s172s582s400 9 0e 6y~9ue~

[6]: %history

%ismagic
import time

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

%timeit
time.sleep(0.01)
%%writefile script.py
print("Пример из скрипта script.py!")
for i in range(5):
    print(i)
!python script.py
%ls
%history
```

Рисунок 12. Магические команды

### 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\furzik\labal\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 Задание.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      389 690 056 192 байт свободно

C:\Users\furzik\anaconda3\python.exe
C:\Users\furzik\AppData\Local\Programs\Python\Python313\python.exe
C:\Users\furzik\AppData\Local\Microsoft\WindowsApps\python.exe

[ ]:
```

Рисунок 13. Команды в JupyterLab

### 14. Работа с Google Drive

```
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
1647796279147.jpg
```

Рисунок 14. Код для работы с Google Drive

15. Зафиксированы изменения на репозитории и отправлены на сервер GitHub

```
VID_20230121_213105.mp4
VID_20230121_213210.mp4
VID_20230128_232848_493.mp4.mov
VID_20230308_135908.mp4
YekOK2s-L6CgrFk6ljwMT1akczHcoWaJQZQDJ1QN8gphQsQxFxwMAMZOkMNAFyd2puYVghcFocvdaKqw9CdZJKuS.jpg
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 15. 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. Откроется 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 интерфейсе JupyterLab?**

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.

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

**2.Использование files.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 измерения времени 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 гитхаб: <https://github.com/Joelilala/LABARATOR1>