

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 3**

Выполнил:  
Борцов Богдан Михайлович  
2 курс, группа ИТС-б-о-23-1,  
11.03.02 «Инфокоммуникационные  
технологии и системы связи»,  
направленность (профиль)  
«Инфокоммуникационные системы и  
сети», 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.

Ссылка на репозиторий: <https://github.com/REPONCFU/ai-jlab1>

**Порядок выполнения работы:**

1. Ознакомление с методическими указаниями
2. Выполнение методических указаний

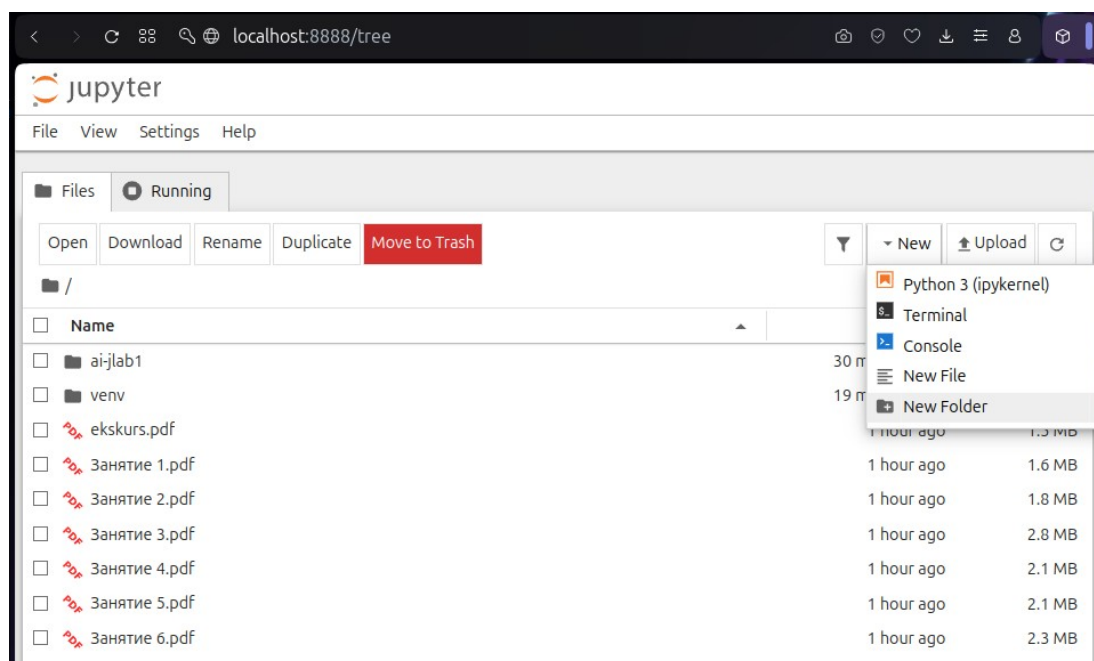


Рисунок 1. Интерфейс Jupyter Notebook

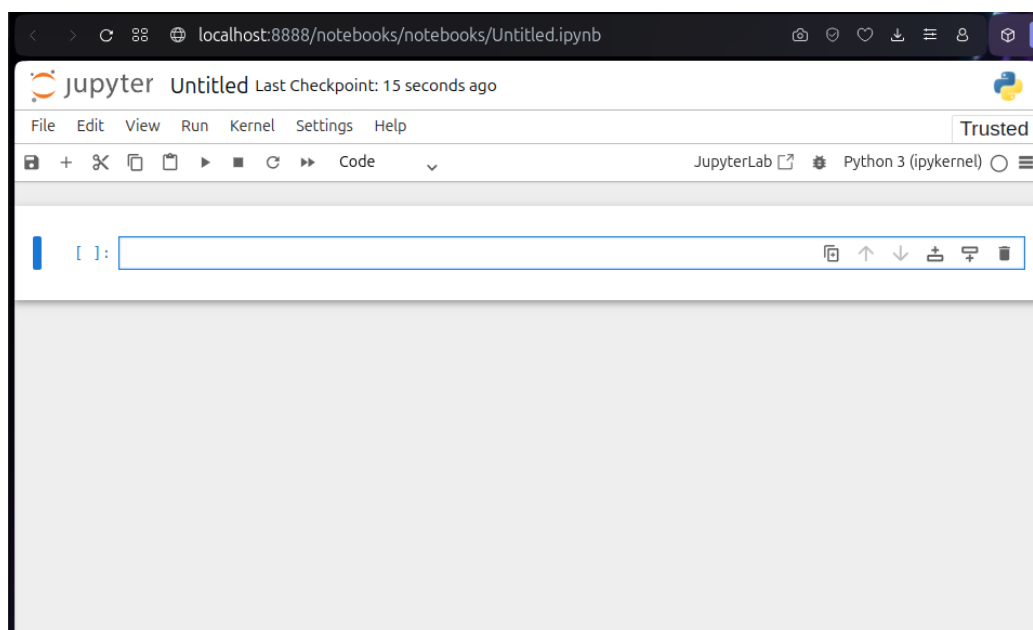


Рисунок 2. Первый notebook

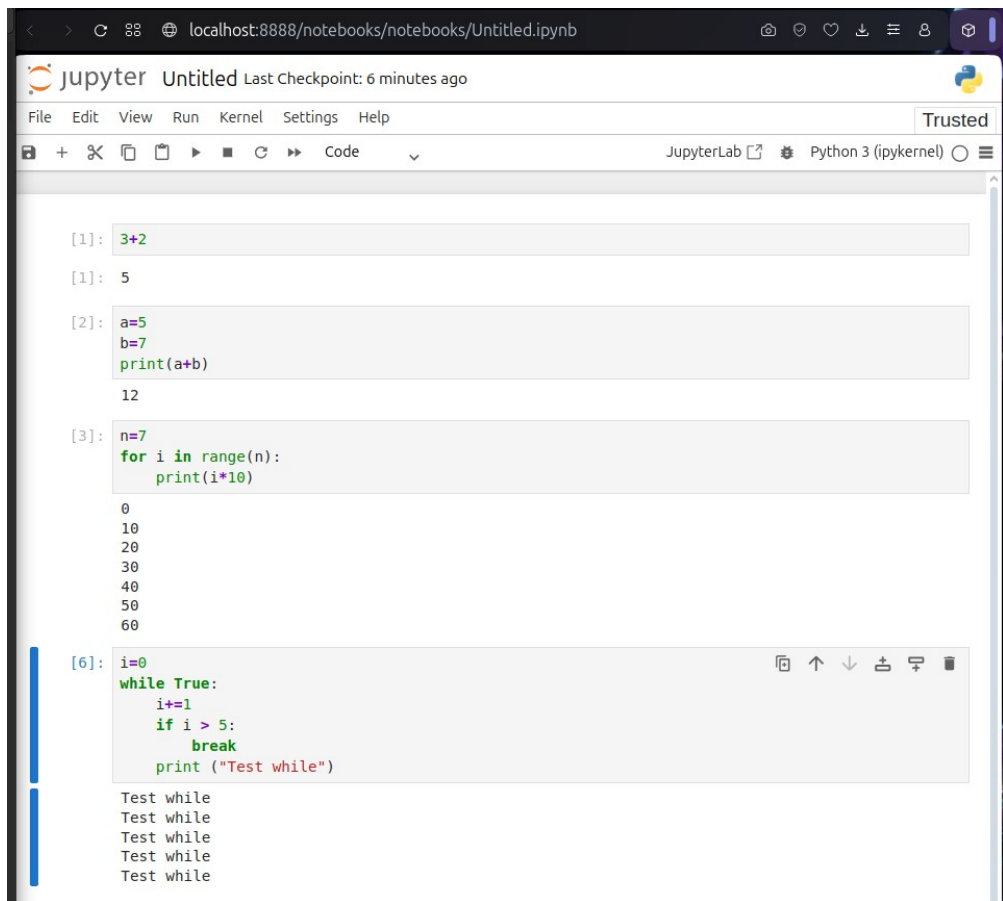


Рисунок 3. Работа с code-блоками

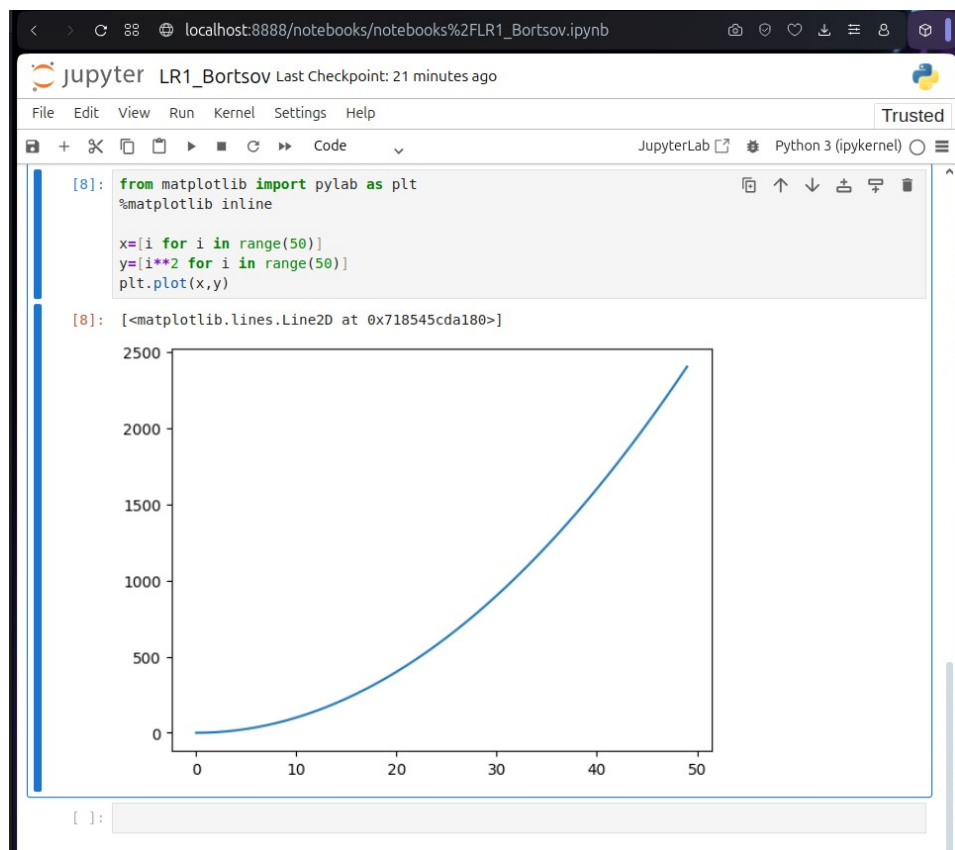
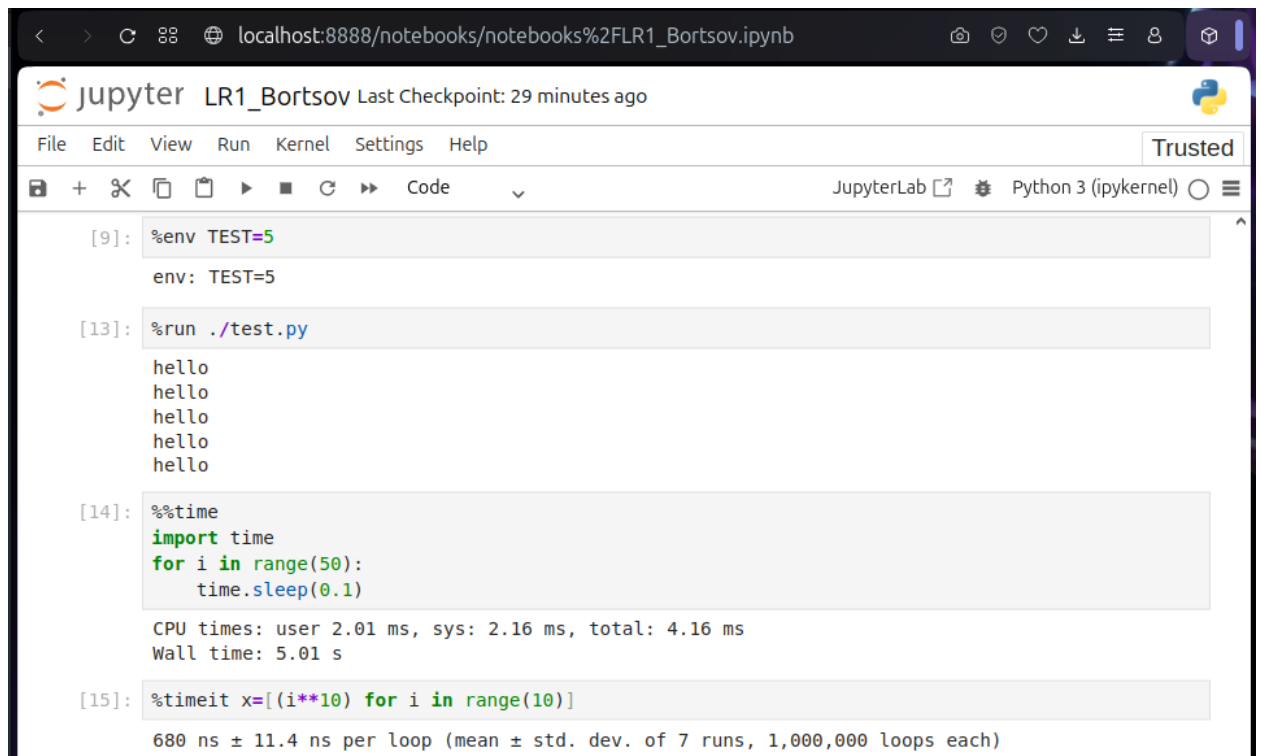


Рисунок 4. Вывод изображений и расчетов



The screenshot shows a JupyterLab interface with a notebook titled 'LR1\_Bortsov'. The browser address bar shows 'localhost:8888/notebooks/notebooks%2FLR1\_Bortsov.ipynb'. The notebook has four code cells:

- Cell [9]: `%env TEST=5` followed by the output `env: TEST=5`.
- Cell [13]: `%run ./test.py` followed by the output `hello` repeated five times.
- Cell [14]: `%%time` followed by a Python loop: `import time; for i in range(50): time.sleep(0.1)`. The output shows CPU times (user: 2.01 ms, sys: 2.16 ms, total: 4.16 ms) and wall time (5.01 s).
- Cell [15]: `%timeit x=[(i**10) for i in range(10)]` followed by the output `680 ns ± 11.4 ns per loop (mean ± std. dev. of 7 runs, 1,000,000 loops each)`.

Рисунок 5. Работа с магией

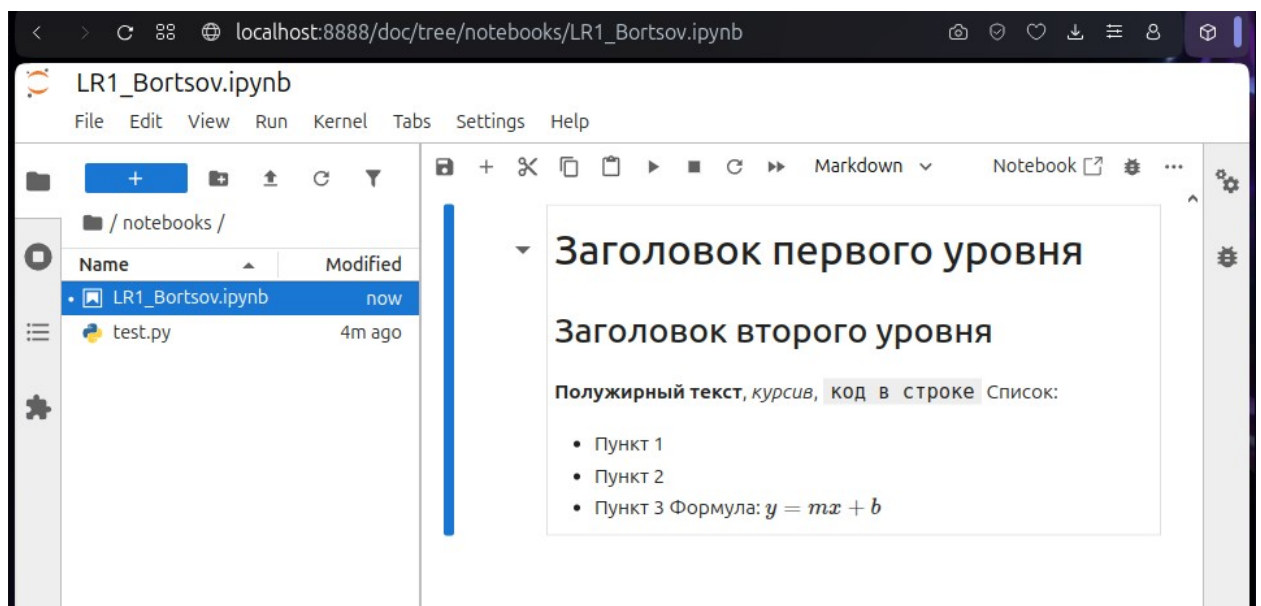


Рисунок 6. Работа с markdown-блоком в Lab

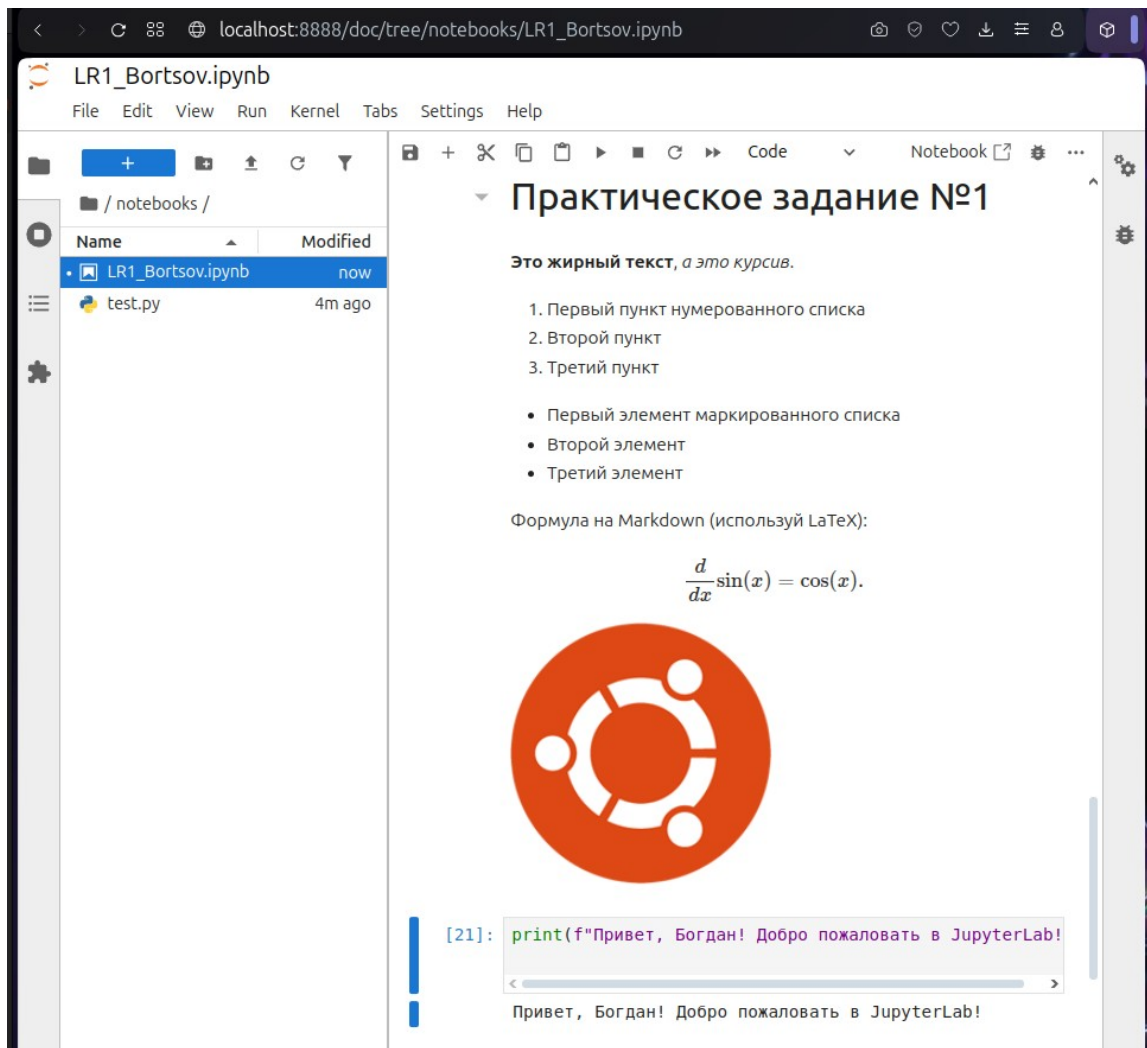


Рисунок 7. Первое практическое задание

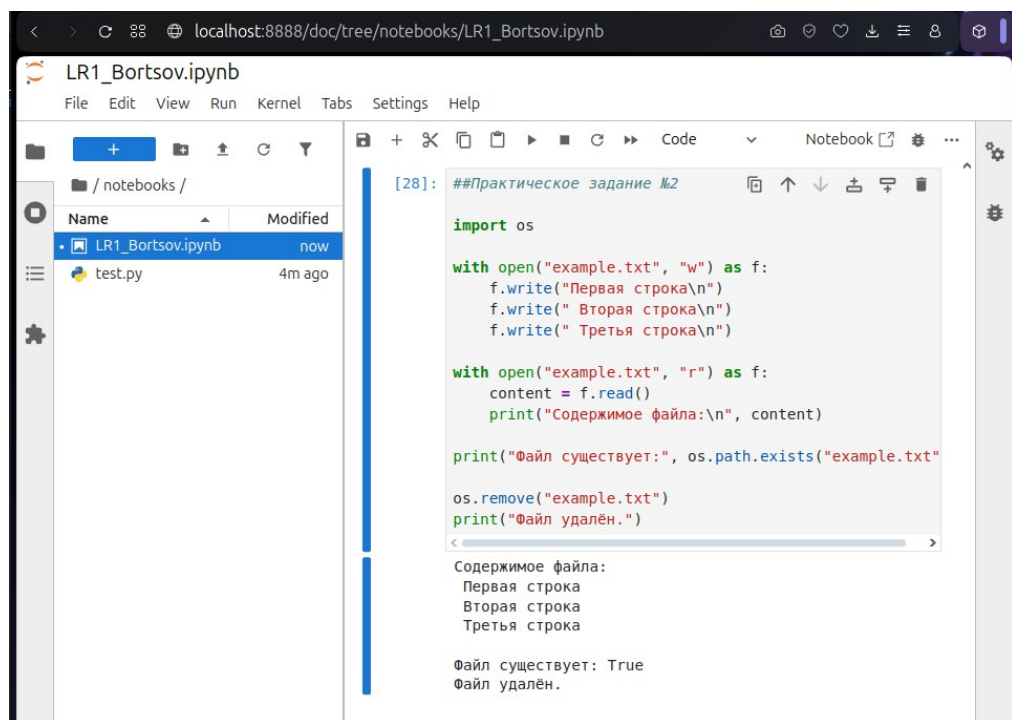


Рисунок 8. Второе практическая задание

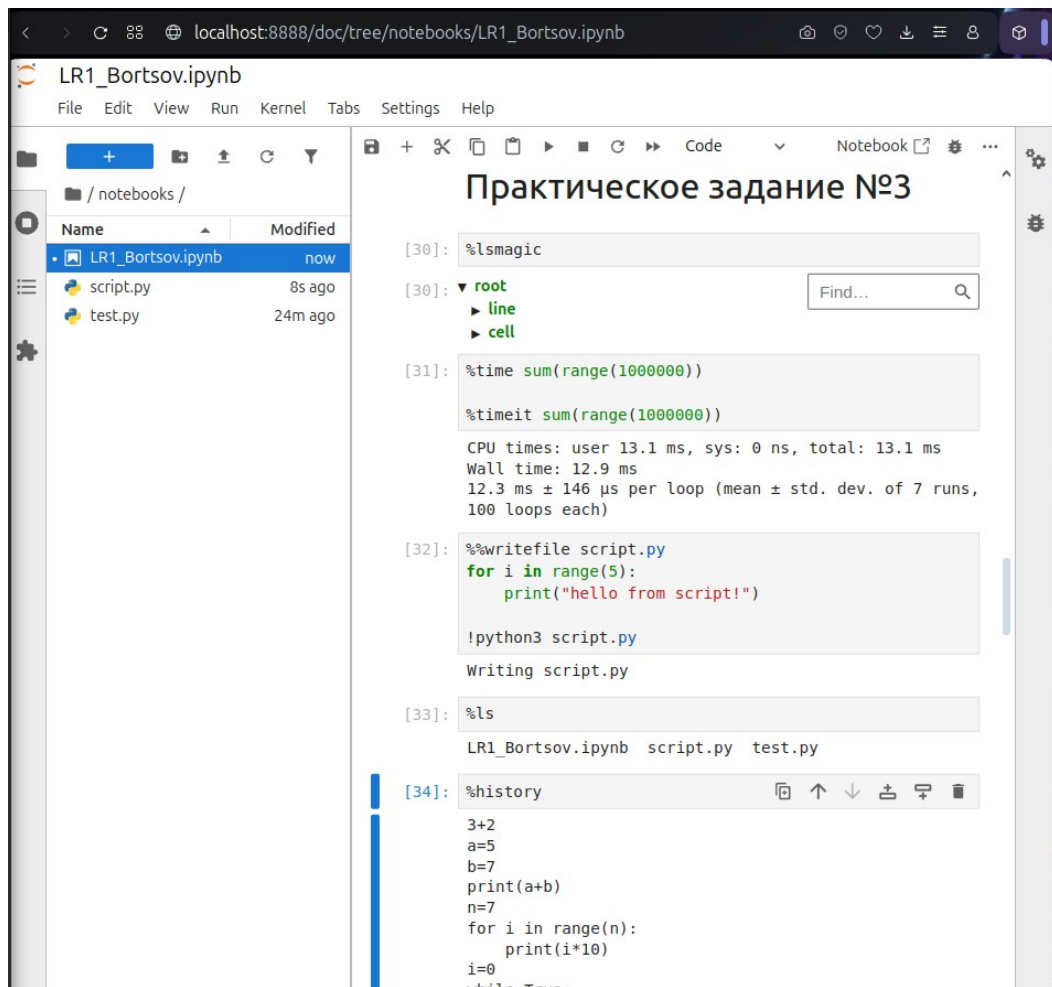


Рисунок 9. Третье практическая задание

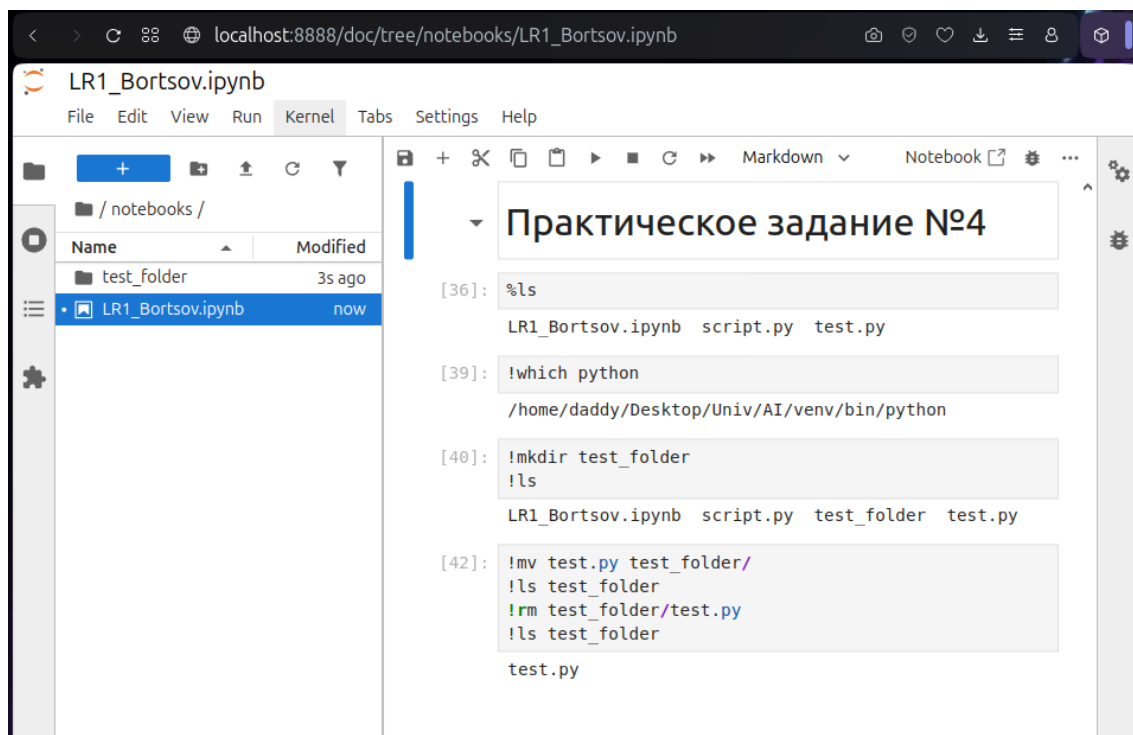


Рисунок 10. Четвертое практическая задание

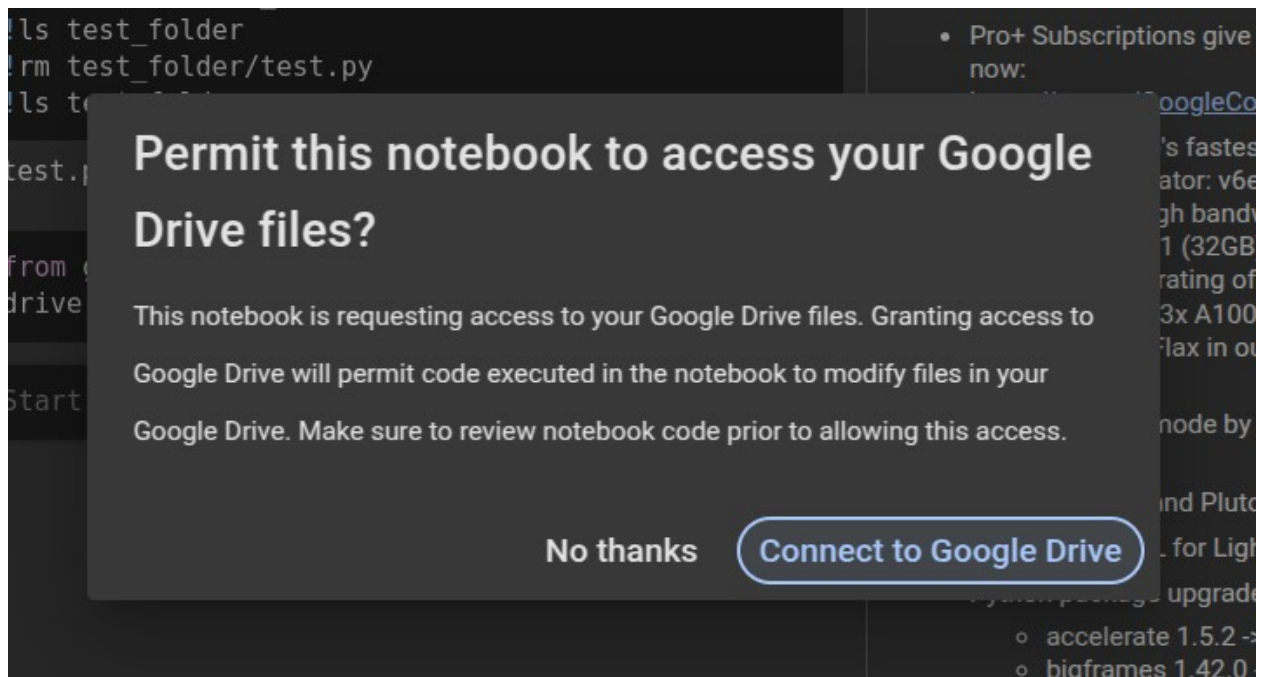


Рисунок 11. Подключаем Google drive к Colab

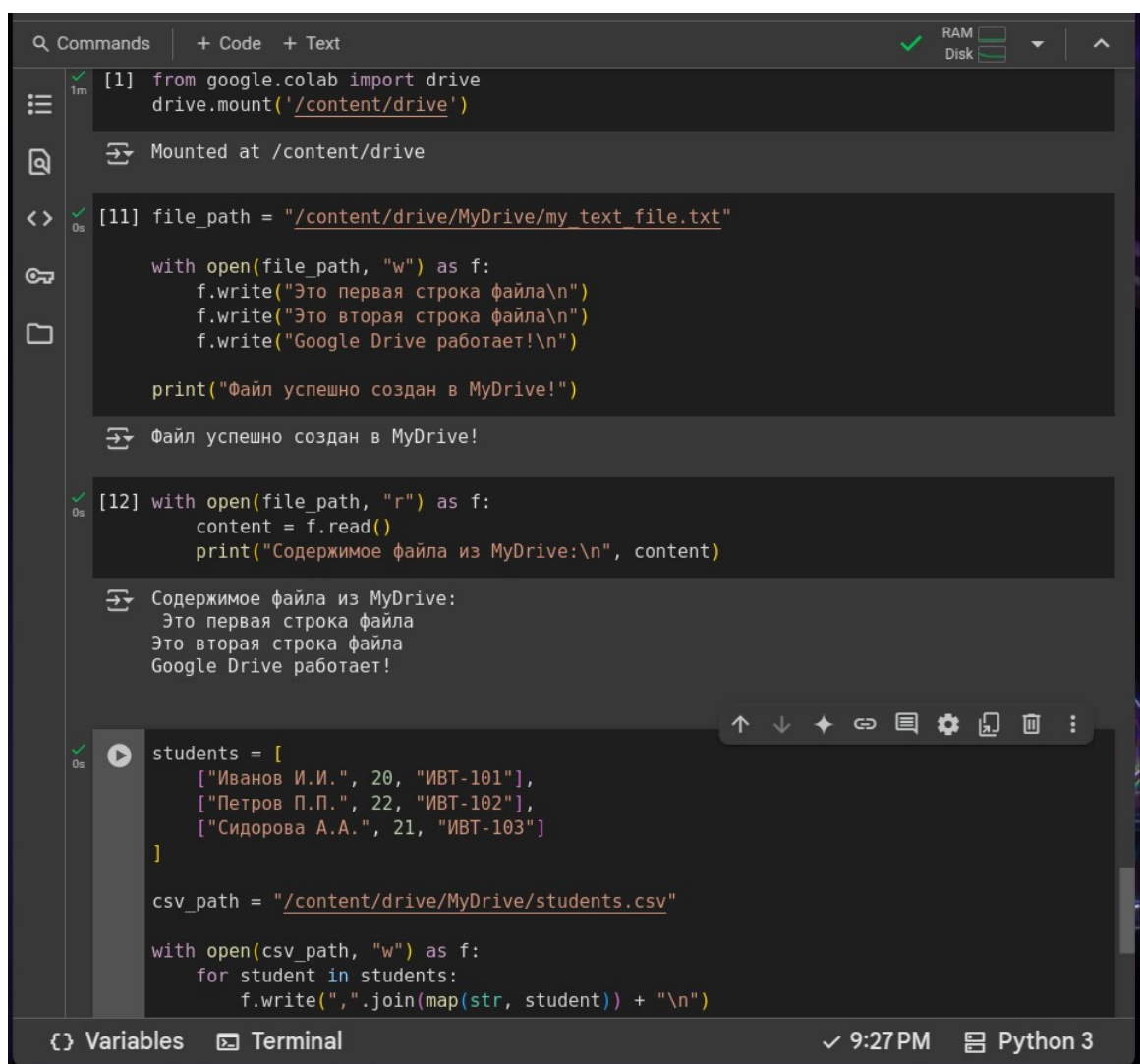


Рисунок 12. Пятая практическая работа

## **Ответы на контрольные вопросы:**

### **1. Основные отличия JupyterLab от Jupyter Notebook**

JupyterLab — это современная интегрированная среда разработки, которая по сравнению с классическим Jupyter Notebook даёт значительно больше возможностей для организации работы. В JupyterLab пользователь может работать одновременно с несколькими файлами, открывая их во вкладках или размещая окна рядом друг с другом. Помимо тетрадей, JupyterLab поддерживает работу с текстовыми файлами, изображениями, терминалами, а также позволяет управлять структурой каталогов через встроенный файловый менеджер. В классическом Jupyter Notebook каждая тетрадь открывается на отдельной вкладке браузера, и отсутствует развитая навигация по проекту.

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

Для создания новой тетради в JupyterLab достаточно нажать на иконку запуска (Launcher) в левой части окна или выбрать пункт меню File → New → Notebook. После этого нужно выбрать подходящее ядро (чаще всего Python 3), и сразу откроется новая рабочая тетрадь, готовая к работе.

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

В JupyterLab доступны три типа ячеек: кодовая (Code), ячейка разметки Markdown и необработанная (Raw). Переключать тип ячейки можно через меню или горячими клавишами: — Esc+Y для кода, — Esc+M для Markdown, — Esc+R для Raw. Такой подход позволяет гибко структурировать тетрадь, чередуя программный код, текстовые пояснения и вставки произвольного текста.

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

Чтобы выполнить код в выбранной ячейке JupyterLab, обычно используется сочетание Shift+Enter — оно запускает код и сразу перемещает курсор к следующей ячейке. Ctrl+Enter выполняет ячейку без перехода, а Alt+Enter дополнительно создаёт новую ячейку после выполнения текущей.

### **5. Как запустить терминал или текстовый редактор внутри JupyterLab?**

В JupyterLab терминал можно открыть через меню File → New → Terminal или кнопкой “+” (Launcher). Текстовый редактор доступен аналогично: File → New → Text File, что позволяет создавать и редактировать любые текстовые документы прямо в рабочей среде.



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

Встроенный файловый менеджер отображается в левой панели JupyterLab — с его помощью можно создавать, переименовывать, перемещать и удалять файлы и папки, а также открывать их в различных приложениях среды.

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

Управление ядрами реализовано через верхнее меню: пункт Kernel позволяет перезапускать, останавливать и выбирать другие ядра. В JupyterLab также есть “Диспетчер сессий” (Running Terminals and Kernels), где можно завершить ненужные процессы.

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

JupyterLab позволяет размещать разные документы — например, несколько тетрадей, редактор кода, терминал, файлы — во вкладках и отдельных областях одного окна, что удобно для одновременной работы над разными частями проекта.

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

Для оценки производительности кода используются магические команды — `%time` для разового измерения времени работы команды и `%timeit` для получения усреднённых результатов по множественным прогонам. Например, команда `%time sum(range(1000000))` покажет, сколько времени занимает вычисление суммы, а `%timeit sum(range(1000000))` выведет среднее время выполнения этого кода при повторных запусках.

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

Для запуска кода на других языках (например, `bash` или `R`) используются магические команды — `%bash` для одной строки или `%%bash` для всей ячейки, аналогично для `%%script` или `%perl`, `%ruby` и других поддерживаемых языков.

## **11. Основные отличия Google Colab от JupyterLab**

Google Colab — это облачный сервис, который позволяет запускать Jupyter-ноутбуки прямо в браузере без установки на компьютер. В Colab есть встроенная поддержка бесплатных GPU и TPU, тесная интеграция с Google Drive для хранения и совместной работы с файлами. JupyterLab, в отличие от

Colab, обычно устанавливается локально и требует настройки окружения пользователем.

## **12. Как создать новый ноутбук в Google Colab?**

Для этого надо перейти на сайт [Google Colab](https://colab.research.google.com/), выбрать пункт “Файл” — “Создать тетрадь” (или “New notebook”), после чего сразу откроется новая облачная тетрадь.

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

Colab поддерживает два основных типа ячеек: код (Code) и разметка (Text/Markdown). Переключаться между ними можно с помощью панели инструментов или сочетаний клавиш — например, — Ctrl+M+M для Markdown, — Ctrl+M+Y для Code.

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

Для запуска кода в ячейке используется Shift+Enter — результат выводится прямо под ячейкой, а курсор автоматически переходит ниже. Также можно использовать Ctrl+Enter для выполнения без перехода.

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

Colab позволяет работать с файлами, хранящимися на Google Drive, а также поддерживает загрузку файлов с локального компьютера через модуль files (например, files.upload()) или напрямую через меню файлового менеджера.

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

Для этого нужно выполнить команды: — `from google.colab import drive` — `drive.mount('/content/drive')` После чего Google Drive станет доступен по пути `/content/drive/MyDrive`, где можно создавать, читать и редактировать файлы.

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

Используется модуль `google.colab.files`, в частности команда `files.upload()`, которая открывает диалоговое окно для выбора файлов на компьютере, а затем они становятся доступны в среде Colab.

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

С помощью команды — `!ls` — можно вывести список файлов в текущей рабочей директории; также доступна навигация по папкам Google Drive через файловый менеджер слева.

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

В Colab полностью поддерживаются стандартные магические команды Jupyter — `%time` и `%timeit`, например: — `%time sum(range(10000))` — `%timeit sum(range(10000))`

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

Для этого необходимо выбрать в верхнем меню пункт “Среда выполнения” (Runtime) — “Сменить среду выполнения” (Change runtime type), и в появившемся окне выбрать GPU или TPU вместо стандартного процессора, после чего подтвердить изменения.

**Вывод:** В ходе лабораторной работы были освоены основные приёмы работы в JupyterLab и Google Colab: создание и настройка ноутбуков, оформление ячеек, применение магических команд, а также базовые операции с файлами и подключение Google Drive. Выполнение заданий позволило научиться эффективно организовывать вычисления, работать с данными и использовать современные инструменты для анализа и автоматизации. Полученные навыки обеспечивают уверенное применение Jupyter и Colab для дальнейших учебных и профессиональных задач.