Список лабараторних робiт

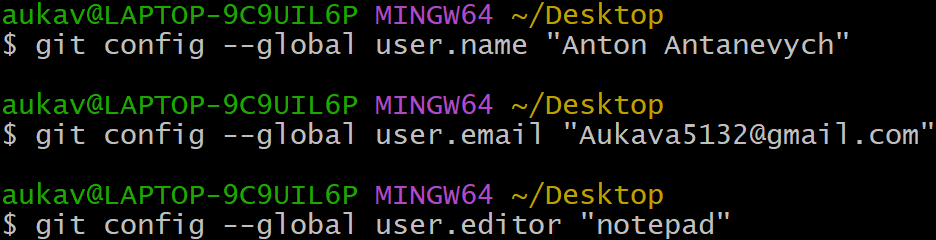
з предмету АСД

студента групи КН-24-1

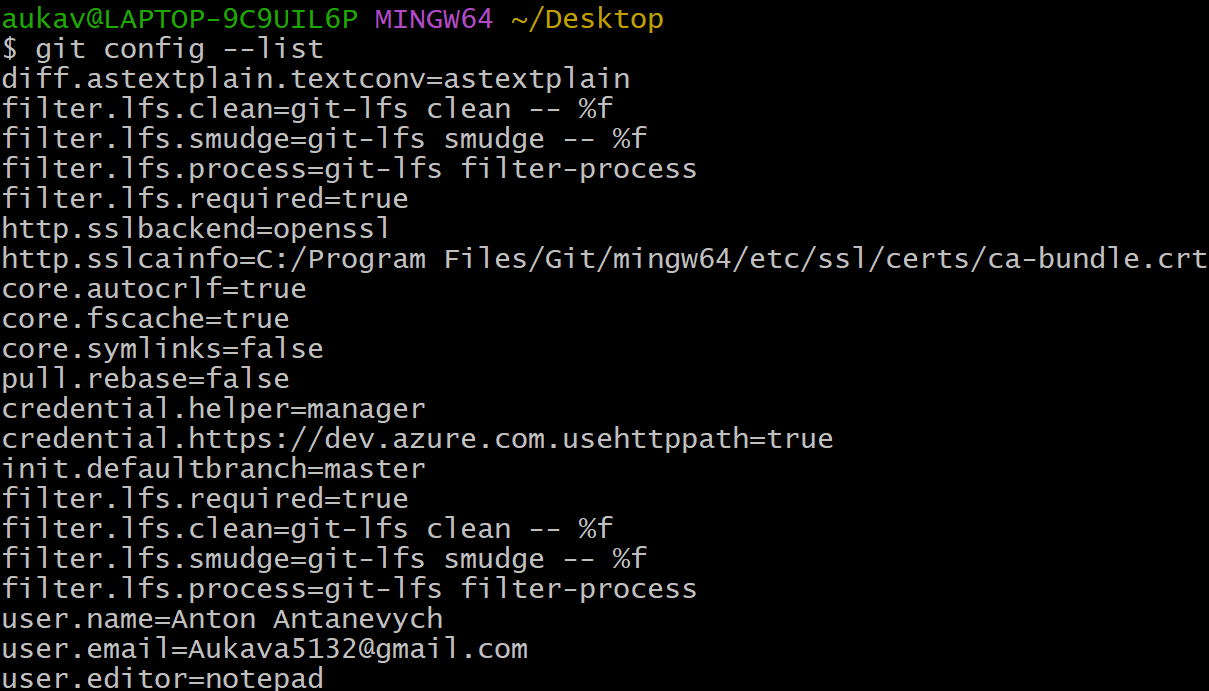
Антаневича Антона

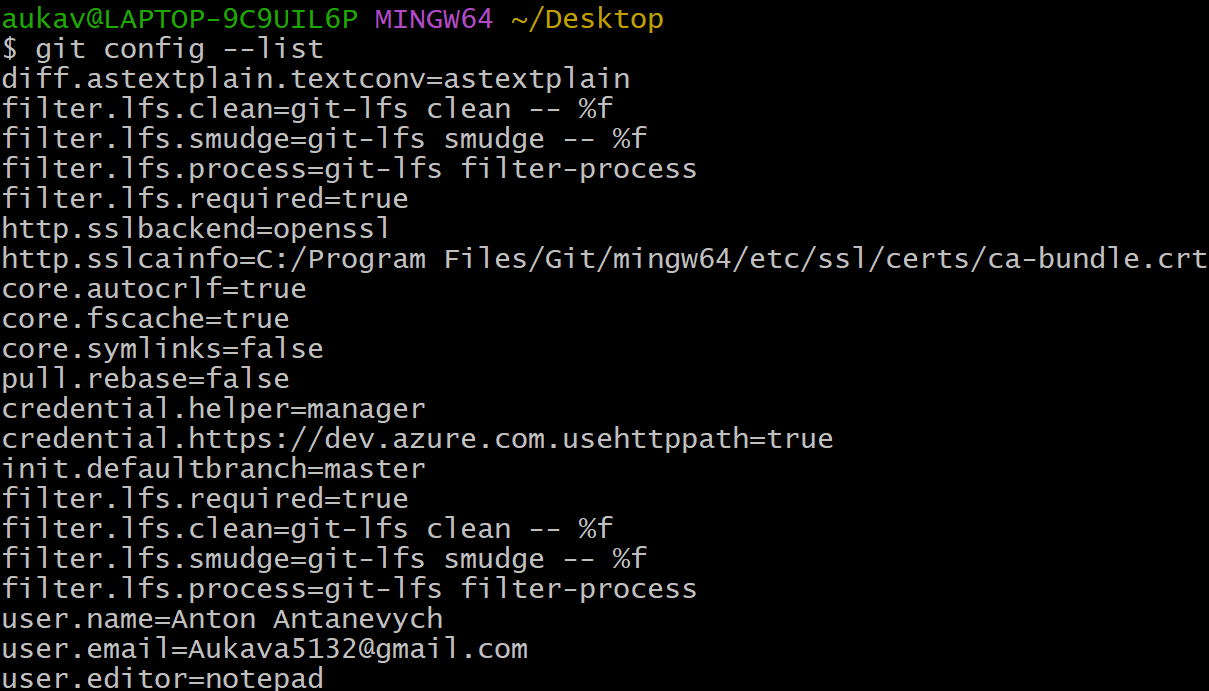
**Лабараторна робота №1**

Налагоджую профiль через Git Bash

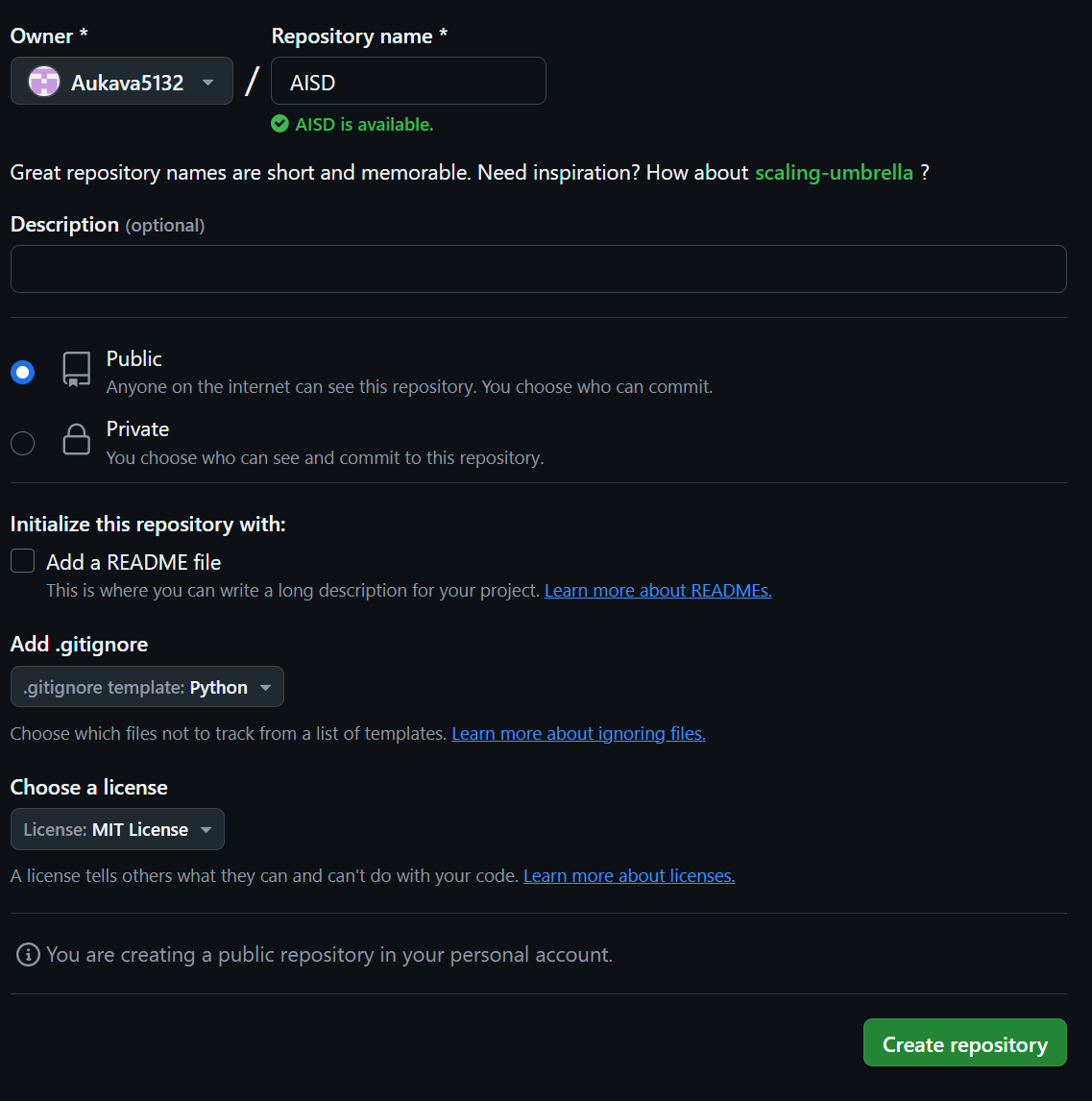
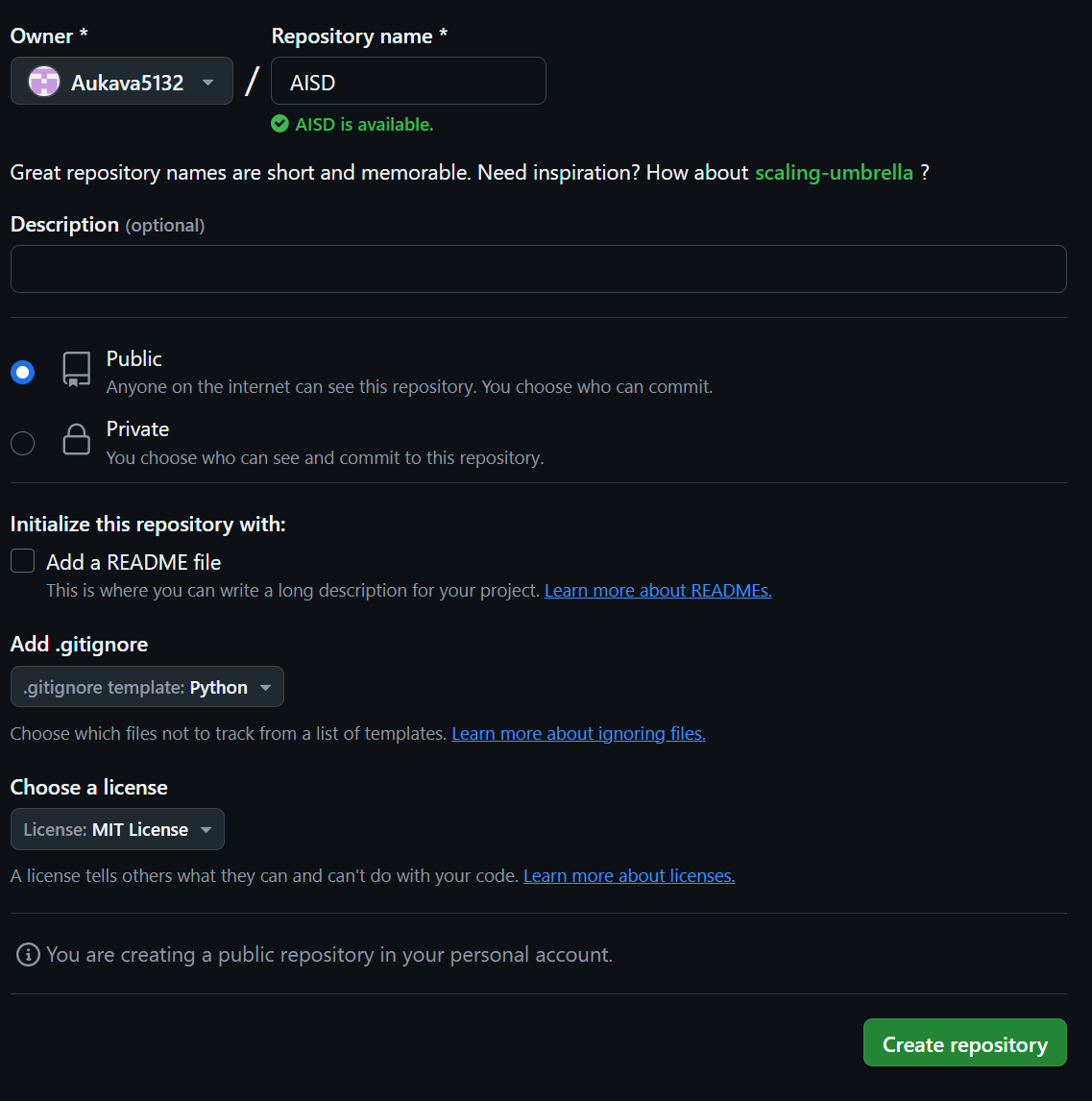
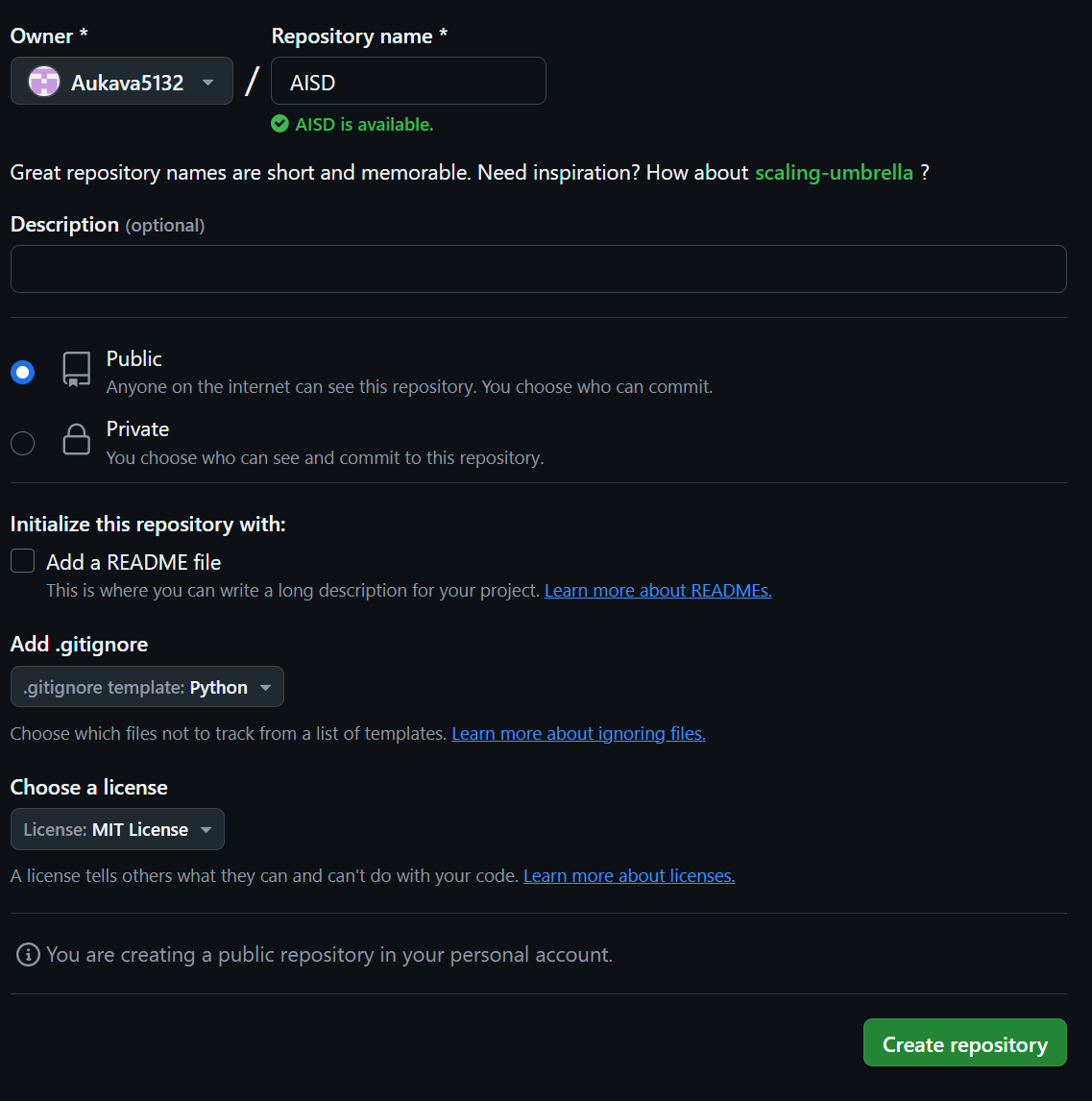


Перевiряю збереження iнформацiї

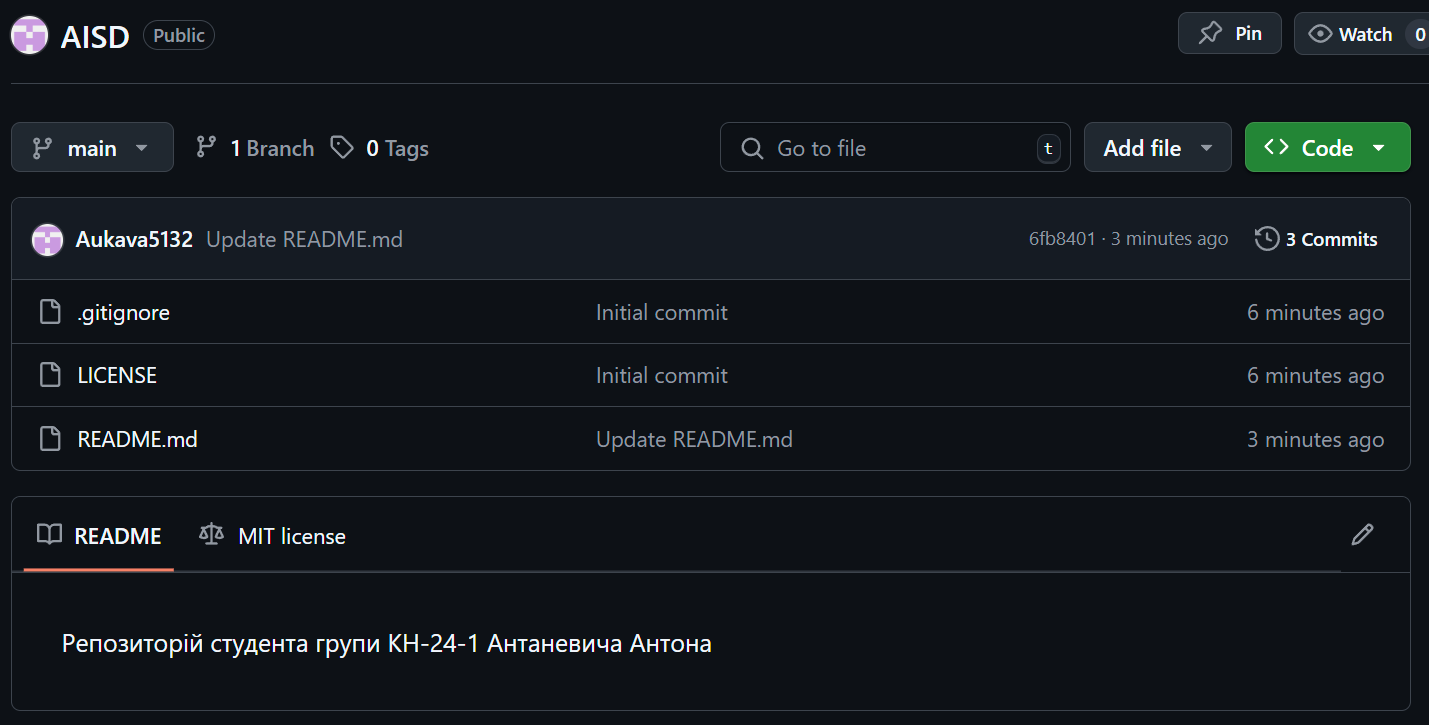




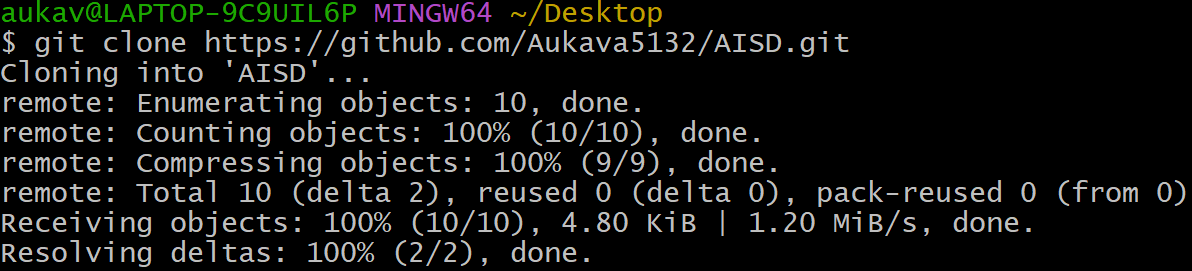
Створюю репозиторiй на GitHub



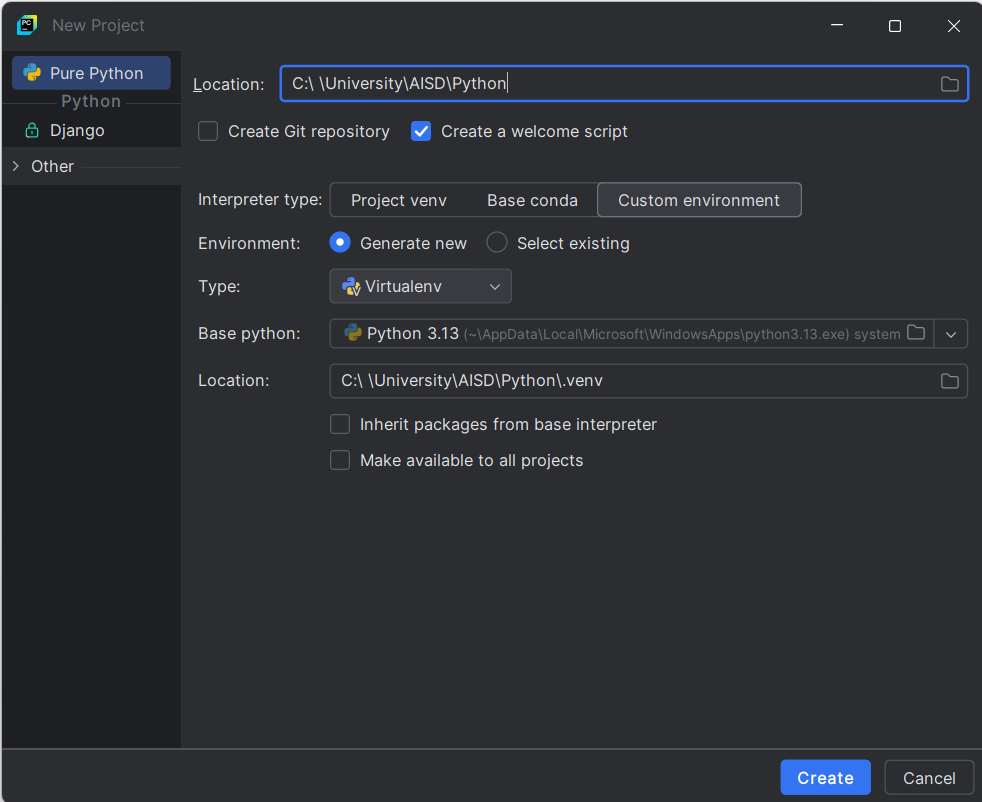
Створений репозиторiй

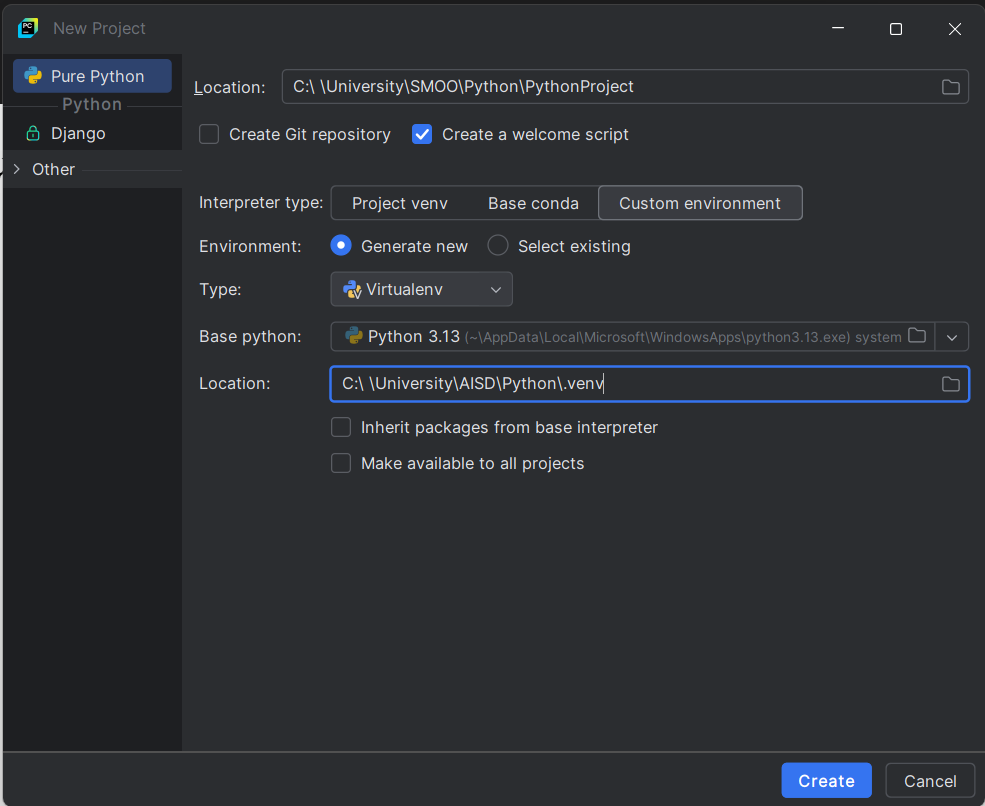


Клоную

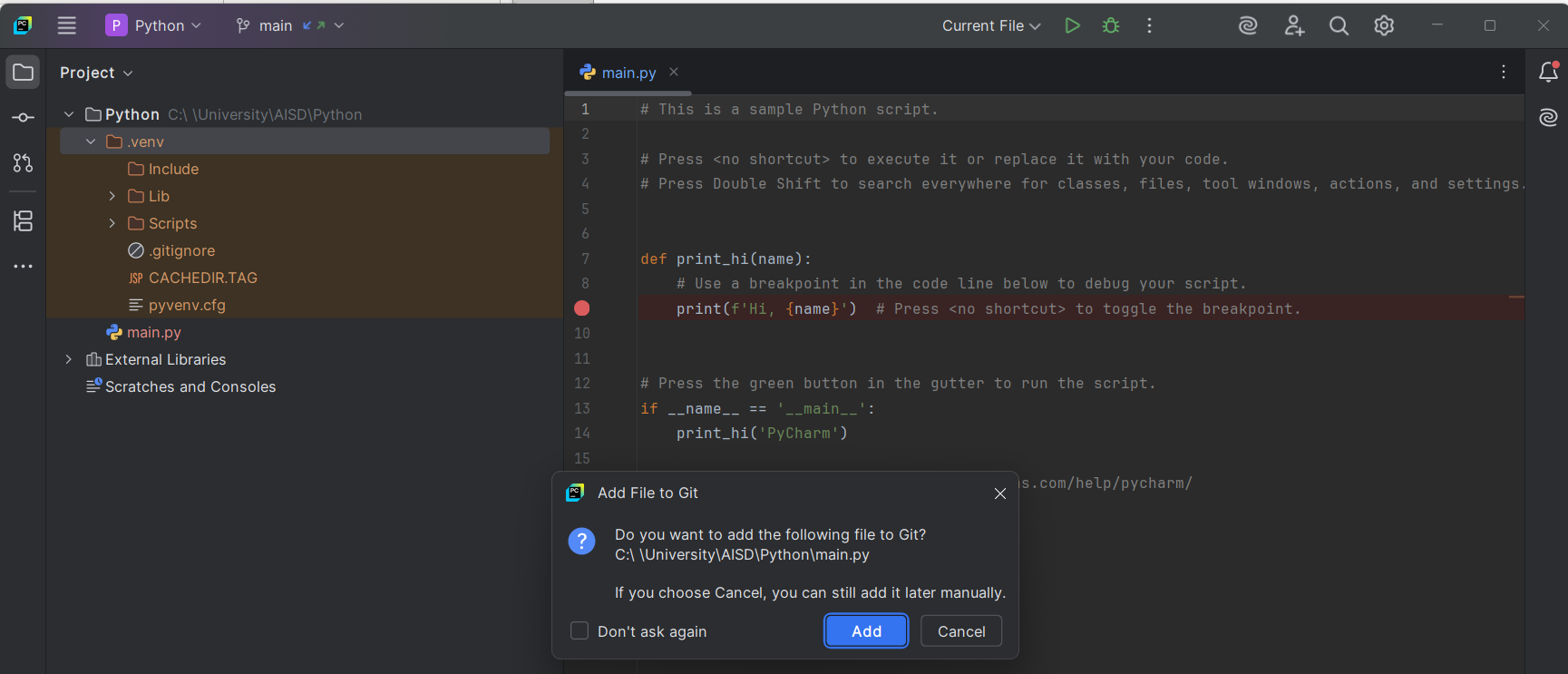


Створюю проект





Створений проект

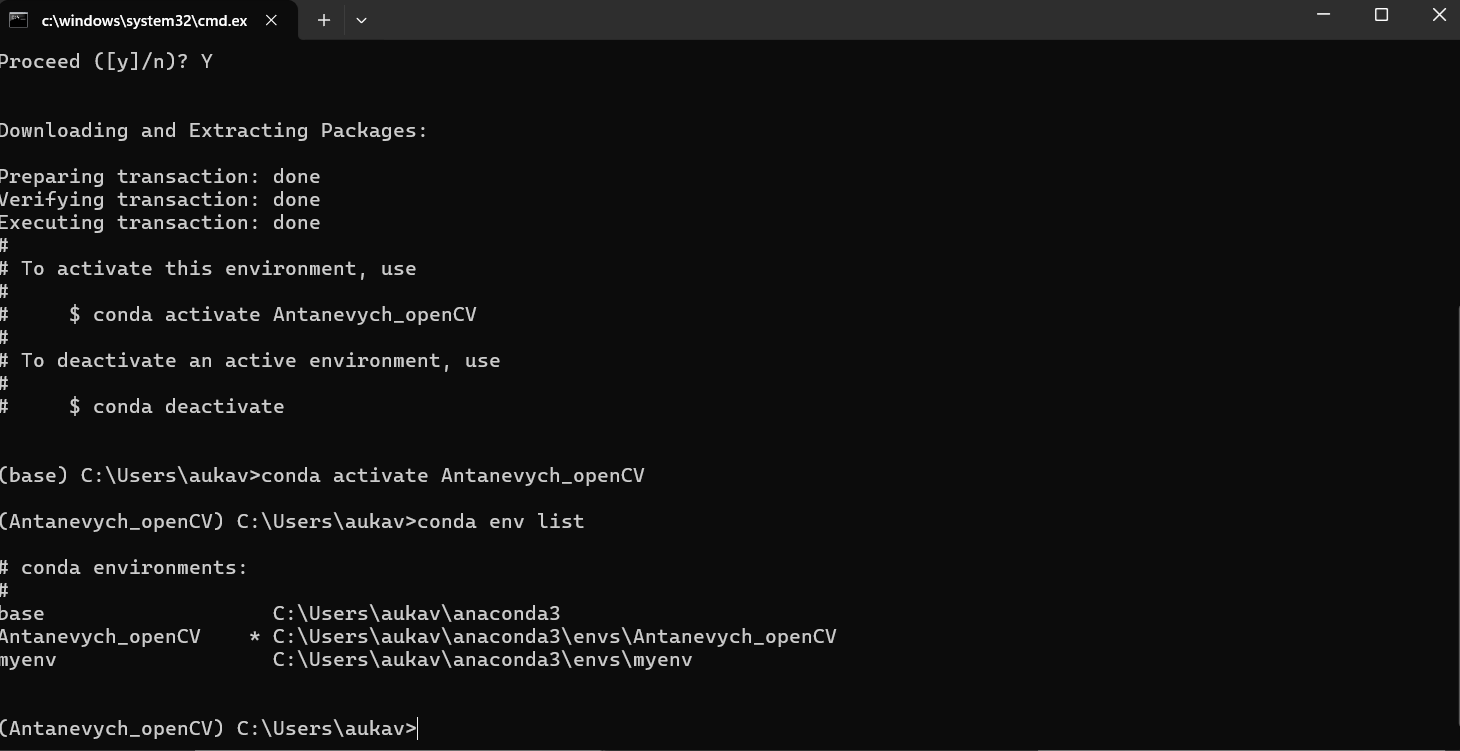


Оновлений репозиторiй

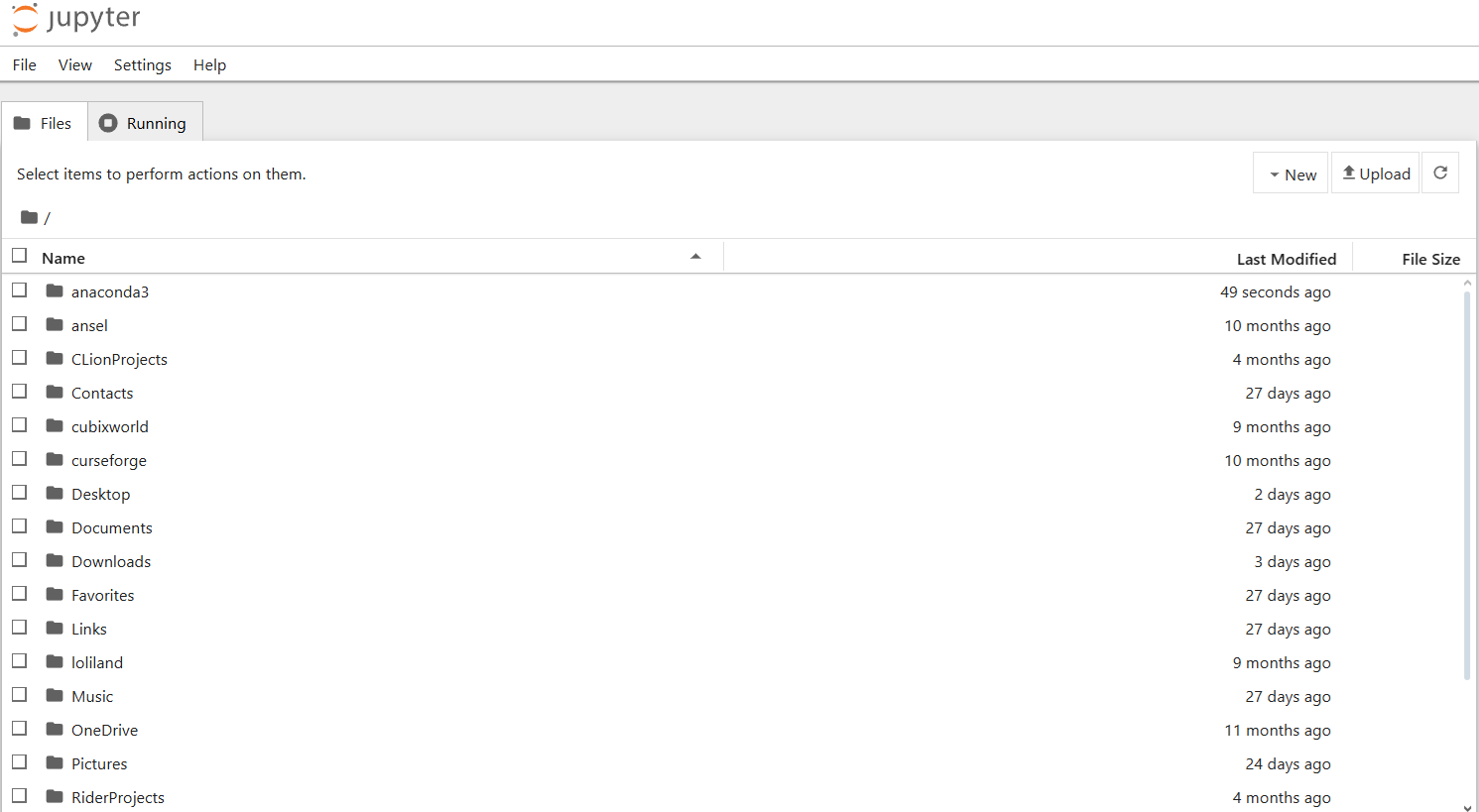


**Лабараторна робота №2**

Налаштування Conda



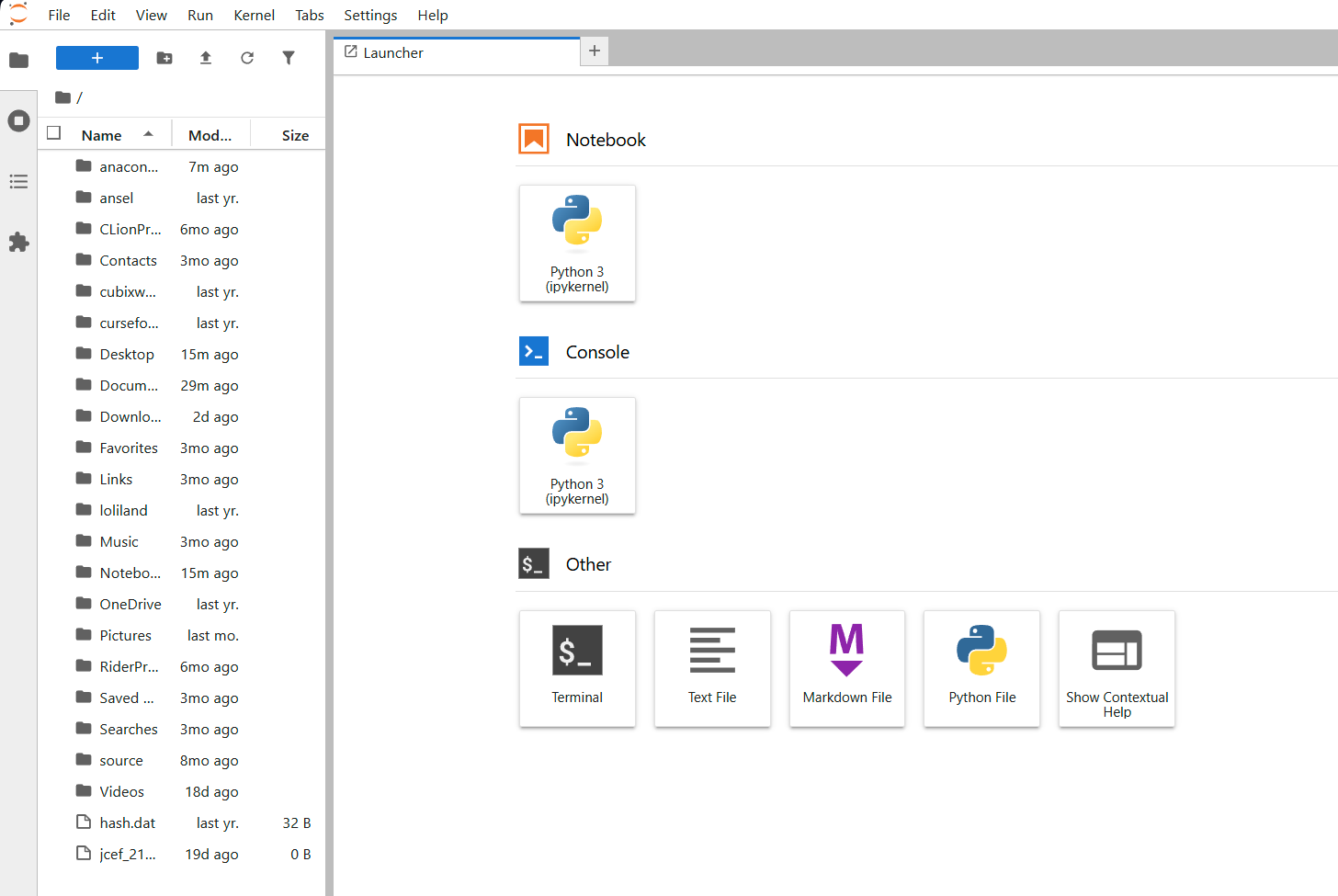
Jupiter Notebook



Запуск JupiterLab

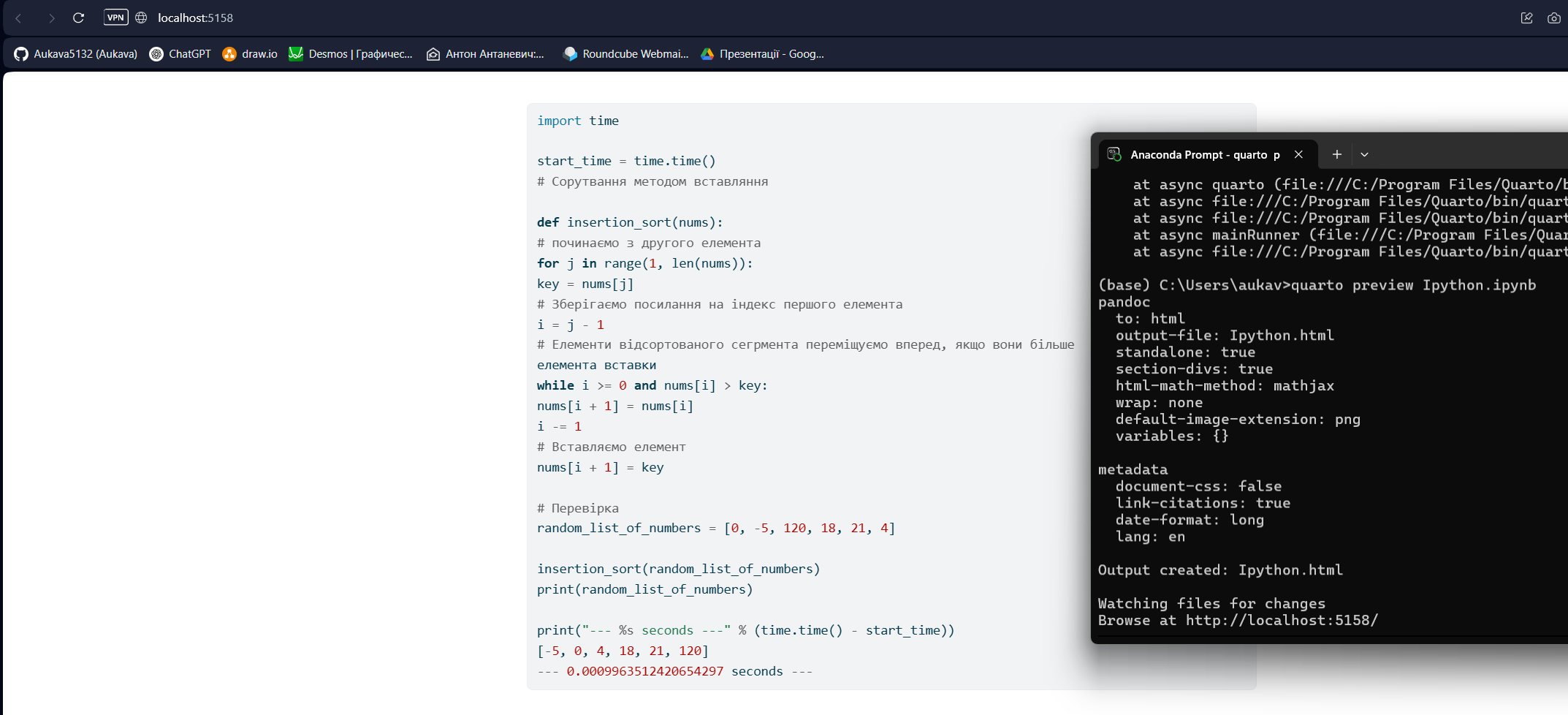
conda activate myenv

jupyter lab



Робота з Quardo

quarto preview ipython.ipynb



**Лабараторна робота №3**

Визначення асимптотичної складності алгоритмів сортування:

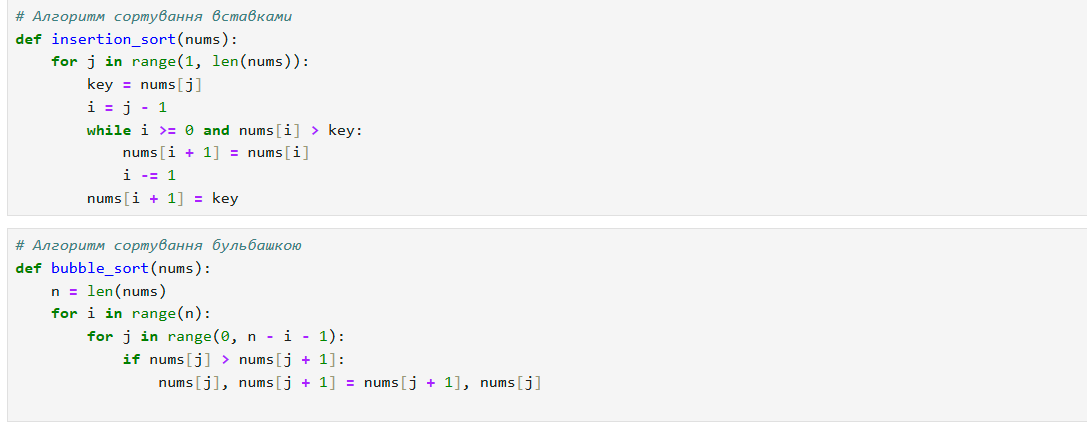
- Сортування вставками (Insertion Sort): O(n2)

- Сортування бульбашкою (Bubble Sort): O(n2)

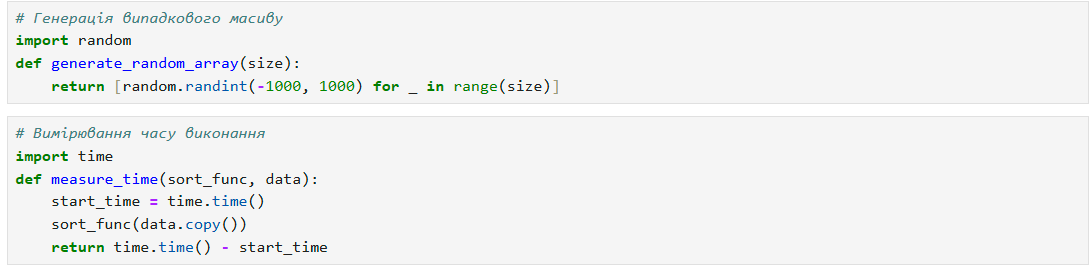
Це означає, що зі збільшенням розміру вхідних даних (n*2*) у 2 рази, час виконання зростає приблизно в 4 рази.

Реалізація алгоритмів:

Реалiзацiя на Python:



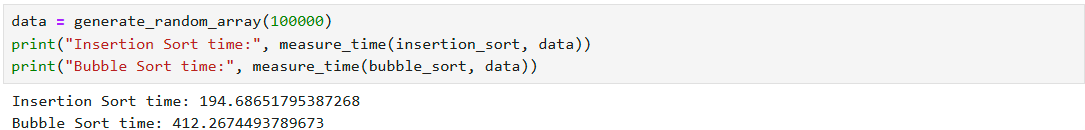
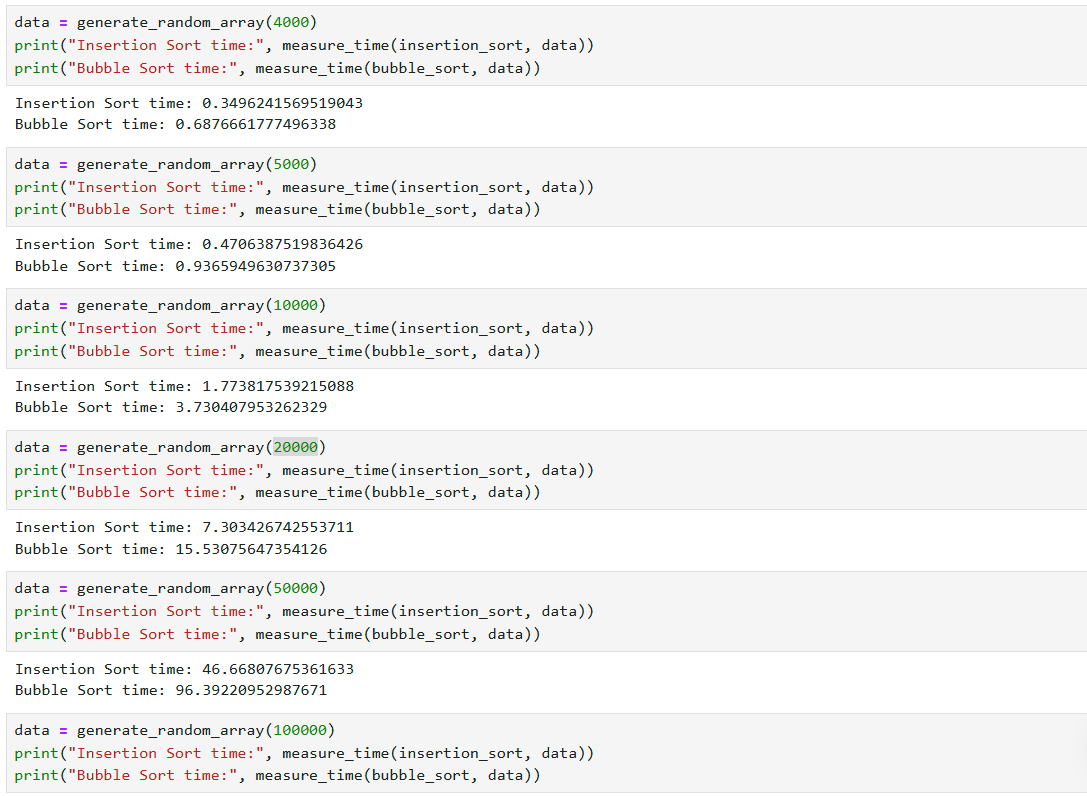
Вимірювання часу виконання



Виміряв час виконання кожного алгоритму за допомогою модуля time:



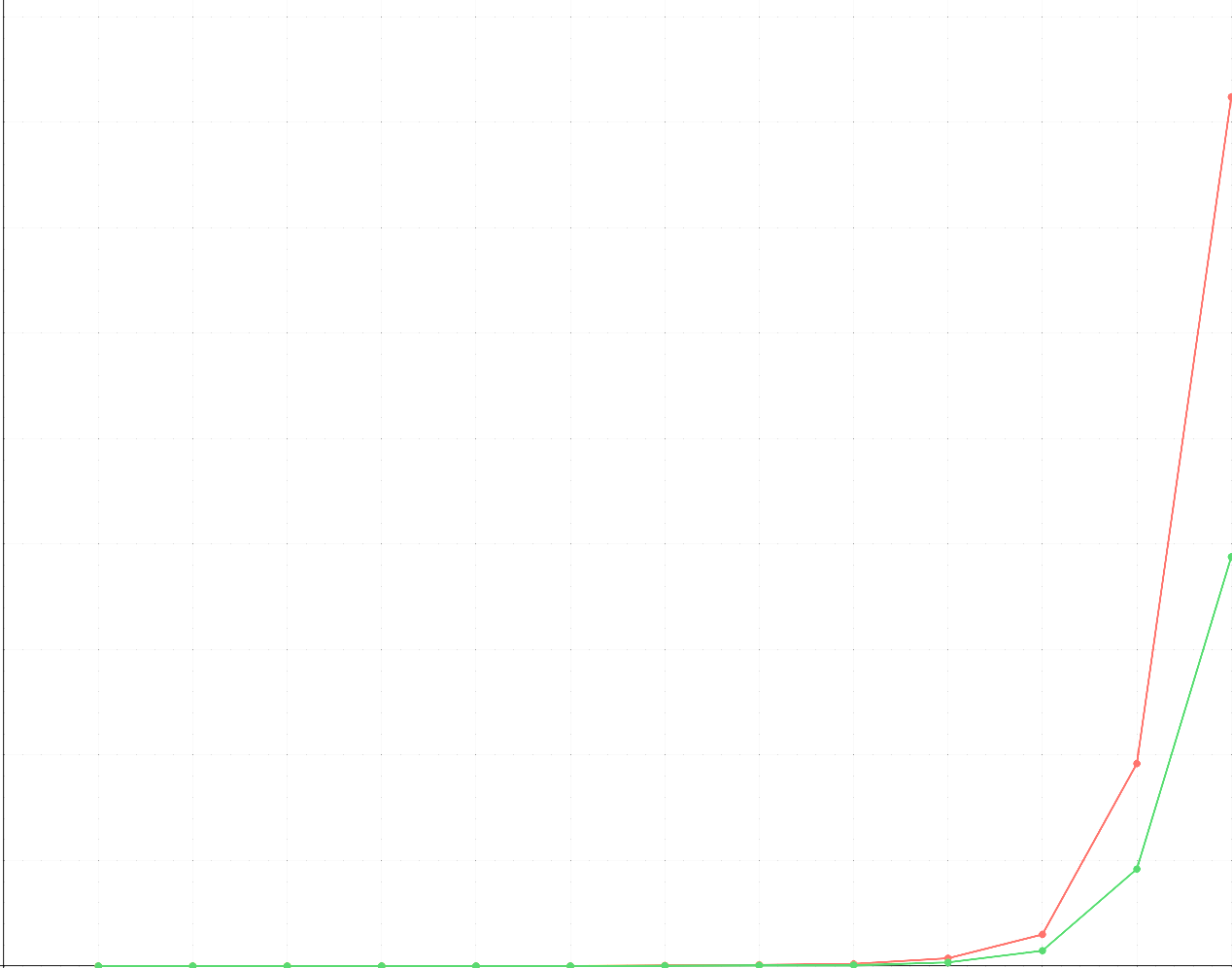




Результати та графіки

Після проведення експериментів я отримав наступні дані:

* Для малих масивів (n<1000*n*<1000) обидва алгоритми працюють швидко, але сортування вставками виявилось ефективнішим.
* Для n=5000*n*=5000 час виконання зріс до 0.5 секунд для insertion\_sort і 1 секунди для bubble\_sort.

Графік залежності часу від розміру масиву:  


46

412

194

92

15

Секунди

Кiлькiсть

5 10 50 100 1000 2000 4000 5000 10000 20000 50000 100000

Висновки:

1. Обидва алгоритми мають квадратичну складність, що підтверджується експериментально.
2. Сортування вставками працює швидше за бульбашкове через меншу кількість операцій обміну