МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ЗВІТ ПРО ВИКОНАННЯ

ІНДИВІДУАЛЬНОГО ДОМАШНЬОГО ЗАВДАННЯ

із дисципліни: «Програмування мовою Python»

на тему «Робота з бібліотеками мови Python. Використання бібліотеки NumPy для розв’язання задач лінійної алгебри»

Виконав студент групи ІН-21/2

Іващев Родіон Олександрович

Варіант №3

Перевірили Парфененко Ю.В.,

Холявка Є.П.

Суми 2023

**ЗМІСТ**

[ВСТУП 3](#_Toc153659333)

[1. АНАЛІЗ ЗАСТОСУВАННЯ БІБЛІОТЕК NUMPY ТА SCIPY ДЛЯ РОЗВ’ЯЗАННЯ ЗАДАЧ ЛІНІЙНОЇ АЛГЕБРИ 4](#_Toc153659334)

[2. РОЗВ’ЯЗАННЯ ЗАДАЧ ЛІНІЙНОЇ АЛГЕБРИ З ВИКОРИСТАННЯМ БІБЛІОТЕКИ NUMPY 6](#_Toc153659335)

[2.1 Постановка задачі 6](#_Toc153659336)

[2.2 Алгоритм розв’язання задачі 6](#_Toc153659337)

[2.3 Програмна реалізація 6](#_Toc153659338)

[Текст програми 8](#_Toc153659339)

[Контрольний приклад 11](#_Toc153659340)

[ВИСНОВОК 13](#_Toc153659341)

[СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 14](#_Toc153659342)

# ВСТУП

Python, з беззаперечною репутацією однієї з найпопулярніших мов програмування, став необхідним інструментом для розв’язання різноманітних завдань, від простих до складних. Його простий синтаксис та велика екосистема бібліотек роблять його ідеальним вибором для вивчення програмування та вирішення завдань з області науки про дані, обробки зображень, штучного інтелекту та багатьох інших.

# АНАЛІЗ ЗАСТОСУВАННЯ БІБЛІОТЕК NUMPY ТА SCIPY ДЛЯ РОЗВ’ЯЗАННЯ ЗАДАЧ ЛІНІЙНОЇ АЛГЕБРИ

NumPy (Numerical Python)[[2](#_СПИСОК_ВИКОРИСТАНОЇ_ЛІТЕРАТУРИ)] - це основна бібліотека для обчислень з використанням масивів та матриць в мові програмування Python. Основне завдання NumPy - надання швидких та ефективних операцій над числовими масивами. Основні функції бібліотеки:

* Масиви та операції

NumPy надає об’єкт `ndarray`, який представляє собою n-вимірний масив та дозволяє виконувати над ним різні операції.

* Базові функції

Математичні операції, статистика, сортування, індексація та інші базові функції реалізовані у бібліотеці NumPy.

* Броадкастінг

Механізм броадкастінгу дозволяє виконувати операції над масивами різного розміру, що спрощує код та робить його більш зрозумілим.

* Лінійна алгебра

Підмодуль `numpy.linalg` містить функції для лінійної алгебри, такі як обчислення власних значень, розв’язання лінійних систем, тощо.

* Випадкові числа

Генерація випадкових чисел реалізована через `numpy.random`.

SciPy[[3](#_СПИСОК_ВИКОРИСТАНОЇ_ЛІТЕРАТУРИ)] - це бібліотека для наукових обчислень, яка будується на основі NumPy. Вона надає додаткові можливості та алгоритми для різних областей наукових обчислень. Основні функції SciPy:

* Оптимізація

`scipy.optimize` містить різні методи оптимізації, такі як мінімізація функцій.

* Інтеграція та диференціальні рівняння

Модуль `scipy.integrate` надає функції для числового інтегрування та розв'язання диференціальних рівнянь.

* Обробка сигналів та обробка зображень

Модулі `scipy.signal` та `scipy.ndimage` дозволяють працювати з сигналами та зображеннями відповідно.

* Лінійна алгебра

Модуль `scipy.linalg` розширює функціонал лінійної алгебри, додаючи деякі додаткові функції порівняно з NumPy.

* Статистика та розподіли

`scipy.stats` містить додаткові статистичні функції та розподіли.

Таблиця 1.1 – Порівняльна таблиця модулів NumPy та SciPy

|  |  |  |
| --- | --- | --- |
| Критерій | NumPy | SciPy |
| Масштабованість | Висока | Висока |
| Функціональність | Базовий набір функцій лінійної алгебри | Додаткові можливості, такі як розширений функціонал лінійної алгебри |
| Швидкодія | Максимальна швидкість завдяки використанню низькорівневих операцій та оптимізаціям | Зазвичай трошки повільніший порівняно з NumPy, але надає додатковий функціонал |
| Документація | Добре задокументована | Добре задокументована |
| Зручність використання | Легко вивчається та використовується | Зручний інтерфейс, але із деяким додатковим навчанням для повного використання |

# РОЗВ’ЯЗАННЯ ЗАДАЧ ЛІНІЙНОЇ АЛГЕБРИ З ВИКОРИСТАННЯМ БІБЛІОТЕКИ NUMPY

## Постановка задачі

Необхідно написати програму мовою Python, яка буде розв’язувати задачі лінійної алгебри з використанням бібліотеки NumPy.

Було вирішено розробити програму, яка буде зчитувати дві матриці з файлу та виконувати над ними операції додавання, віднімання та множення.

## Алгоритм розв’язання задачі

Алгоритм розв’язання задачі у програмі виглядає наступним чином:

1. Зчитування матриць з файлу: Програма використовує функцію read\_matrices\_from\_json, яка використовує модуль json для зчитування даних з файлу та конвертації їх у список матриць NumPy.
2. Вибір операції: Користувач вводить операцію (додавання, віднімання, множення) через введення з клавіатури.
3. Виконання операції з використанням NumPy: В залежності від вибраної операції, програма використовує відповідний метод NumPy (наприклад, np.add, np.subtract, np.dot) для виконання відповідної лінійної алгебричної операції.
4. Виведення результату: Результат операції виводиться на екран, дозволяючи користувачеві переглядати результат.

## 2.3 Програмна реалізація

Таблиця 2.1 – Таблиця компонентів програми

|  |  |  |  |
| --- | --- | --- | --- |
| Назва компоненту | Тип | Опис | Призначення |
| NumPy | Бібліотека | Бібліотека для виконання операцій лінійної алгебри | Використовується для роботи з матрицями та векторами в програмі на мові Python. |
| json | Модуль | Вбудований модуль Python для роботи з форматом JSON | Надає функціонал для роботи з JSON-даними |
| read\_matrices\_from\_json | Функція | Зчитує матриці з файлу JSON і повертає список них | Забезпечення можливості зчитування матриць з файлу JSON |
| main | Функція | Основна частина програми, викликається для виконання основного коду | Керує введенням матриць з файлу, вибором операції та виведенням результату. |

Таблиця 2.2 – Таблиця символьних імен

|  |  |  |
| --- | --- | --- |
| Назва | Тип даних | Призначення |
| file\_name | Рядок | Зберігає ім'я файлу для зчитування матриць. |
| matrices | Список NumPy масивів | Список матриць, зчитаних з файлу |
| operation | Рядок | Зберігає обрану користувачем операцію. |
| result\_matrix | NumPy масив | Зберігає результат виконаної операції. |

## Текст програми

import numpy as np

import json

def read\_matrices\_from\_json(filename):

"""

Зчитує матриці з файлу JSON та повертає їх у вигляді списку NumPy масивів.

Parameters:

filename (str): Ім'я файлу, з якого зчитується матриці у форматі JSON.

Returns:

list of np.ndarray: Список матриць у форматі NumPy або None, якщо виникла помилка.

"""

try:

with open(filename, 'r') as file:

# Завантаження JSON-даних з файлу

data = json.load(file)

# Створення списку матриць

matrices = [np.array(matrix) for matrix in data['matrices']]

return matrices

except FileNotFoundError:

print(f"Файл {filename} не знайдено.")

return None

except Exception as e:

print(f"Помилка при зчитуванні матриць з файлу: {e}")

return None

def write\_result\_to\_file(result\_matrix, output\_filename="out.txt"):

"""

Записує результат обчислення до файлу.

Parameters:

result\_matrix (np.ndarray): Результат обчислення у форматі NumPy масиву.

output\_filename (str): Ім'я вихідного файлу.

Returns:

None

"""

try:

with open(output\_filename, 'w') as file:

# Запис результату у файл

file.write("Результат обчислення:\n")

file.write(str(result\_matrix))

print(f"Результат записано у файл {output\_filename}")

except Exception as e:

print(f"Помилка при записі результату у файл: {e}")

def main():

# Ім'я файлу з матрицями у форматі JSON

file\_name = "matrix.json"

# Зчитування матриць з файлу

matrices = read\_matrices\_from\_json(file\_name)

if matrices is None:

return

if not matrices:

print("Файл не містить матриць для обробки.")

return

# Вибір операції користувачем

operation = input(

"\nВиберіть операцію (додавання - 'add', віднімання - 'subtract', множення - 'multiply'): "

)

try:

# Виконання вибраної операції

if operation == 'add':

result\_matrix = np.add(matrices[0], matrices[1])

elif operation == 'subtract':

result\_matrix = np.subtract(matrices[0], matrices[1])

elif operation == 'multiply':

result\_matrix = np.dot(matrices[0], matrices[1])

else:

print("Невірна операція. Завершення програми.")

return

# Запис результату у файл

write\_result\_to\_file(result\_matrix)

except ValueError as ve:

print(f"Помилка: {ve}")

if \_\_name\_\_ == "\_\_main\_\_":

main()

## Контрольний приклад

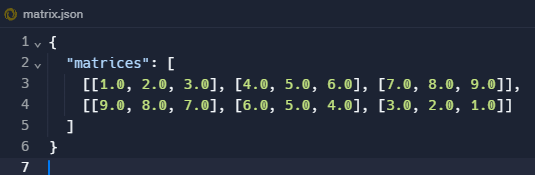


Рисунок 2.1 – вміст файлу matrix.json

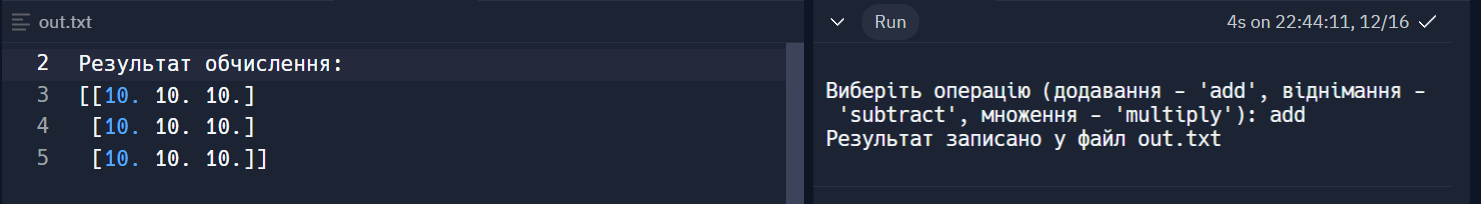


Рисунок 2.2 – результат роботи програми після вибору операції `add` (додавання)

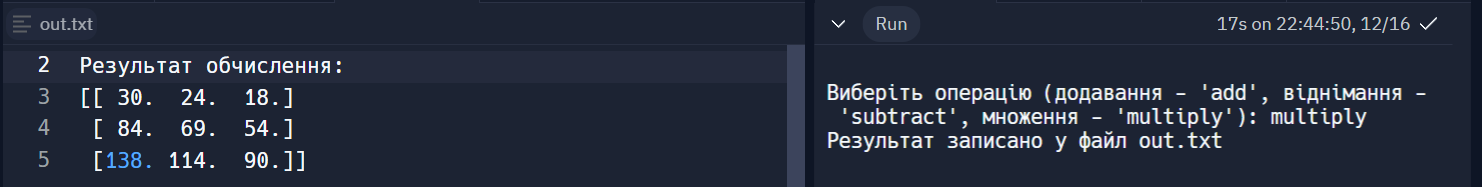


Рисунок 2.3 – результат роботи програми після вибору операції `multiply` (множення)

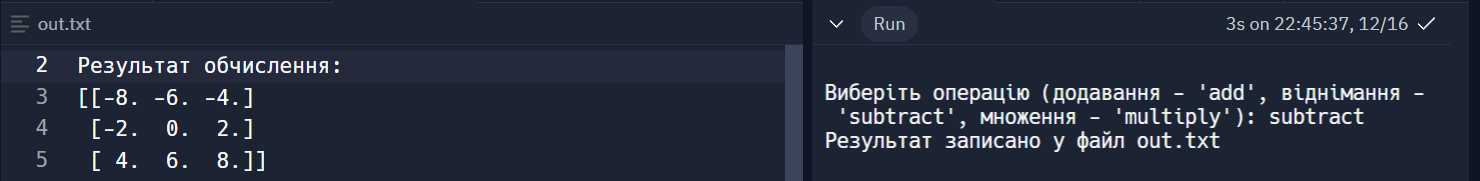


Рисунок 2.4 – результат роботи програми після вибору операції `subtract` (віднімання)

# ВИСНОВОК

У результаті виконання індивідуального завдання, було виконано порівняльний аналіз застосування бібліотек NumPy та SciPy для розв’язання задач лінійної алгебри. На його основі виявлено, що бібліотека SciPy є більш досконалою та надає нові можливості порівняно з NumPy.

Наступним етапом, було запропоновано створити власну програму, яка буде розв’язувати деяку задачу лінійної алгебри. Було вирішено написати код, у результаті виконання якого, з файлу типу .json зчитуються дві матриці, після чого, користувачу пропонується обрати одну з трьох операцій над ними – додавання, віднімання та множення. Результатом роботи є програма, яка приймає у себе матриці розміром 3x3 з файлу matrix.json, після чого, виконує над ними певні розрахунки, та виводить результати до файлу out.txt.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Методичні вказівки до виконання індивідуального домашнього завдання. MIX learning. URL: [<https://mix.sumdu.edu.ua/textbooks/66748/1204111/Методичн__вказ_вки__ДЗ.pdf>]
2. NumPy user guide. URL:  
   [<https://numpy.org/doc/stable/user/>]
3. SciPy v.1.11.4 Manual. SciPy User Guide. URL:  
   [<https://docs.scipy.org/doc/scipy/tutorial/>]