НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

із лабораторної роботи

з дисципліни «АЛГОРИТМИ І СИСТЕМИ КОМП’ЮТЕРНОЇ

МАТЕМАТИКИ 1.МАТЕМАТИЧНІ АЛГОРИТМИ»

на тему

“Розв’язання систем лінійних рівнянь”

|  |  |
| --- | --- |
| Виконав: | Перевірила: |
| студент групи КМ-01 | Асистент кафедри ПМА |
| Романецький М.С. | Ковальчук-Химюк Л. О. |

Київ — 2023

Зміст

[Вступ 2](#_Toc147677830)

[Основна частина 3](#_Toc147677831)

[Варіант 1 3](#_Toc147677832)

[Вимоги до ПЗ 3](#_Toc147677833)

[Алгоритми розв’язання 3](#_Toc147677834)

[Висновки 4](#_Toc147677835)

[Відповіді на контрольні питання 4](#_Toc147677836)

[Використана література 4](#_Toc147677837)

[Додаток А – скріншоти роботи програми 5](#_Toc147677838)

[Додаток Б – Блок-схема роботи програми 6](#_Toc147677839)

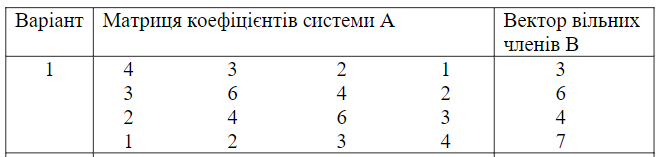
[Додаток В – Код програми 7](#_Toc147677840)

# Вступ

Метою роботи є вивчити правила використання програмних засобів для факторизації матриць і розв’язання систем лінійних алгебраїчних рівнянь, розв’язувати задану систему рівнянь, провести порівняльний аналіз вивчених чисельних методів розв’язання СЛР.

# Основна частина

## Варіант 1



## Вимоги до ПЗ

1. Реалізувати перевірки на некоректний ввід (буква замість числа, порожнє введення, символи замість числа, занадто великі числа).
2. У графічному інтерфейсі повинно бути передбачено можливість гнучкого налагодження розмірності розв’язуваної задачі.
3. Має зберігатись логічно правильне розв’язання (нижня межа інтегрування не має перевищувати верхню, тощо)

## Алгоритми розв’язання

Програма складається з графічного інтерфейсу, та методів розв’язання на мові Python та мові Octave.

* Мовою Python реалізовано розв’язання системи рівнянь LU методом.
* Мовою Octave реалізовано розв’язання системи рівнянь LU методом.

Алгоритм LU працює наступним чином:

1. Розкладання LU: Матриця A розкладається на дві матриці: L (нижньотрикутна матриця) та U (верхньотрикутна матриця). Тобто, якщо A - це наша матриця, то A = L \* U.
2. Передувальний прохід: Множимо матрицю L на вектор b, ми отримуємо новий вектор. Цей крок зводить нашу систему до верхньотрикутної форми.
3. Зворотній прохід: Використовуючи матрицю U та новий вектор, ми отримуємо розв’язок системи.

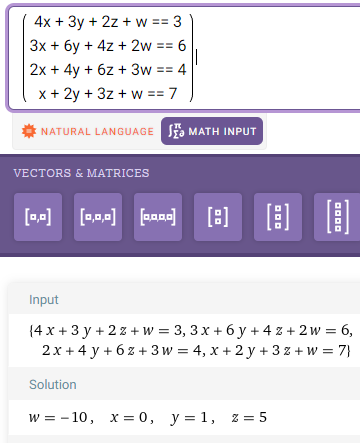
Цей метод ефективний для розв’язання великих систем лінійних рівнянь

# Висновки

На мові Python:

* Час виконання — 0.002 сек
* Похибка — 0
* Результат — [0. 1. 5. -10.]

На мові Octave:

* Час виконання — 3.10 сек
* Похибка — 0
* Результат — [0. 1. 5. -10.]

Результати співпали. Час виконання менший при використання Python. Нульова похибка скоріше за все пов'язана з доцільним використанням LU- методу для заданої матриці.

Розв’язавши систему у Wolfram Alpha бачимо, що було отримано точний розв’язок

Скріншоти виконання програми наведені в Додатку А.

# Відповіді на контрольні питання

Відповіді наведені в репозиторії GitHub за [посиланням](https://github.com/TokenRR/ASKM/tree/main/Lab_1)

# Використана література

1. Python to GNU Octave Bridge — <https://oct2py.readthedocs.io/en/latest/>
2. Python GUI Programming With Tkinter — <https://realpython.com/python-gui-tkinter/>

# Додаток А – скріншоти роботи програми

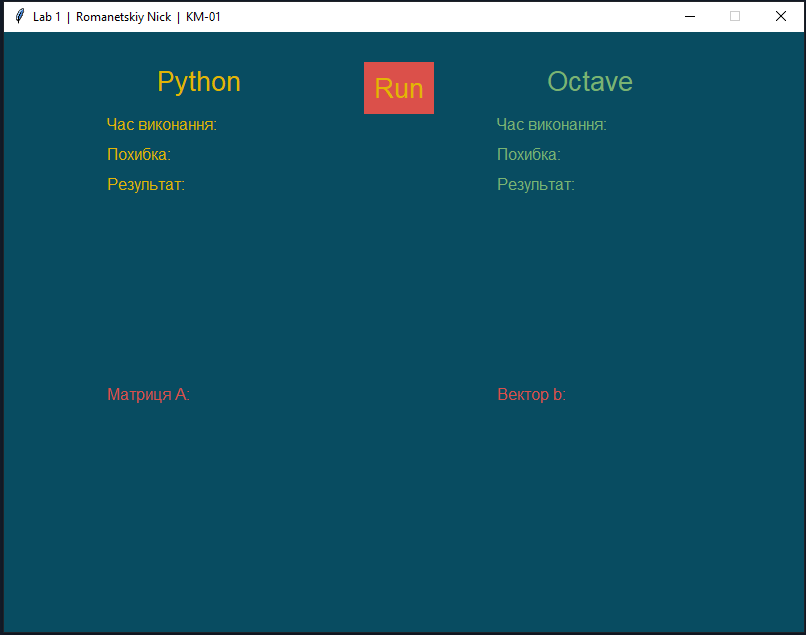


Рис. 1 – Вікно програми (до натискання кнопки ‘Run’)

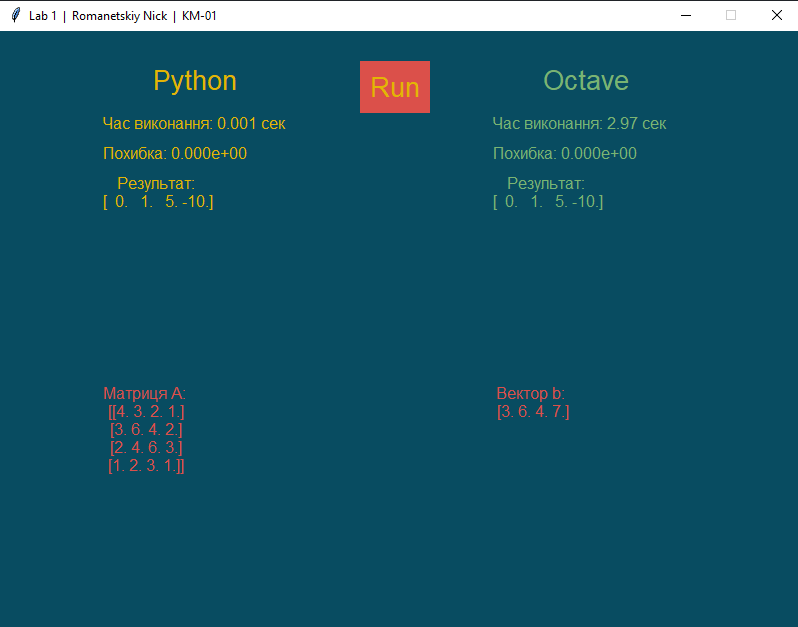


Рис. 2 – Вікно програми (після натискання кнопки ‘Run’)

# Додаток Б – Блок-схема роботи програми

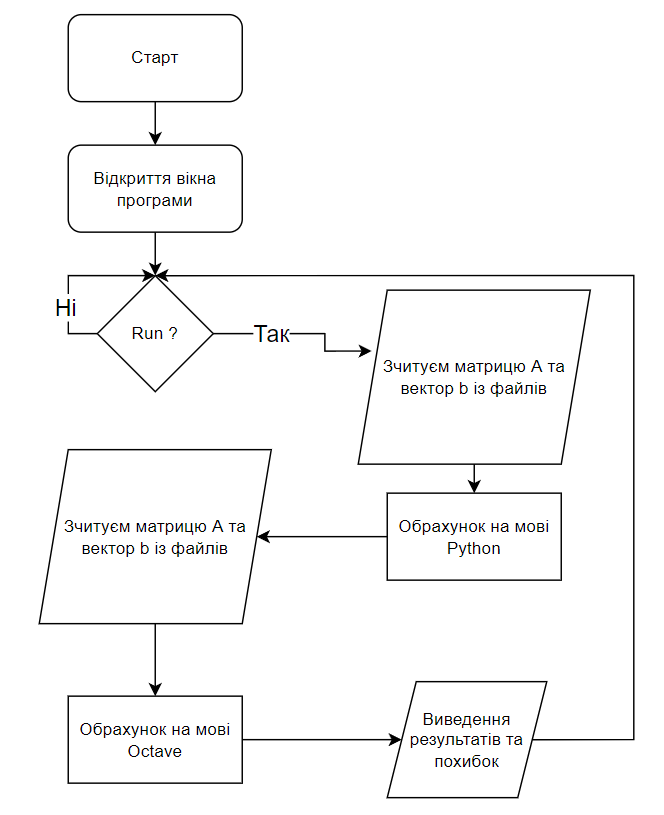


Рис. 3 – Блок-схема роботи програми

# Додаток В – Код програми

Вміст файлу Matrix\_A.txt :

4 3 2 1

3 6 4 2

2 4 6 3

1 2 3 1

Вміст файлу Vector\_b.txt :

3

6

4

7

Вміст файлу main.py :

from window import \*

import os

if \_\_name\_\_ == '\_\_main\_\_':

    os.system('cls')  *#  Очищення консолі*

    root.mainloop()  *#  Запуск головного циклу програми*

*#            ТЕСТИ*

*# 1. Матриця А - буква           |  +*

*# 2. Матриця А - замало чисел    |  +*

*# 3. Матриця А - забагато чисел  |  +*

*# 4. Матриця А - не квадратна    |  +*

*# 5. Вектор b  - буква           |  +*

*# 6. Вектор b  - замало чисел    |  +*

*# 7. Вектор b  - забагато чисел  |  +*

*# 8. Вектор b  - більше 1 числа  |  +*

Вміст файлу window.py :

from octave\_ import run\_octave

from python\_ import run\_python

import time

import os

import tkinter as tk

CL\_BLUE = '#084C61'

CL\_RED = '#DB504A'

CL\_YELLOW = '#E3B505'

CL\_GRAY = '#4F6D7A'

CL\_OCEAN = '#56A3A6'

CL\_GREEN = '#79B473'

CL\_PURPLE = '#414073'

root = tk.Tk()  *#  Створення головного вікна програми*

root.title('Lab 1  |  Romanetskiy Nick  |  KM-01')  *#  Задання назви вікну*

root.geometry('800x600')  *#  Задаємо розмір вікна*

root.resizable(width=False, height=False)  *#  Заборона зміни розмірів вікна*

root.configure(bg=CL\_BLUE)  *#  Задаємо колір фону*

def run():

    os.system('cls')  *#  Очищення консолі*

*# - - - - - #*

    try:

        start\_python\_time = time.time()

        matrix\_A, vector\_b, python\_answer, python\_accuracy = run\_python()

        python\_run\_time = time.time() - start\_python\_time

        python\_time\_label.config(text=f'Час виконання: {round(python\_run\_time, 4)} сек')

        python\_accuracy\_label.config(text=f'Похибка: {"{:.3e}".format(python\_accuracy)}')

        python\_result\_label.config(text=f'Результат: \n{python\_answer}')

*# - - - - - #*

        start\_octave\_time = time.time()

        octave\_answer, octave\_accuracy = run\_octave()

        octave\_run\_time = time.time() - start\_octave\_time

        octave\_time\_label.config(text=f'Час виконання: {round(octave\_run\_time, 2)} сек')

        octave\_accuracy\_label.config(text=f'Похибка: {"{:.3e}".format(octave\_accuracy)}')

        octave\_result\_label.config(text=f'Результат: \n{octave\_answer}')

*# - - - - - #*

        matrix\_A\_label.config(text=f'Матриця А:\n {matrix\_A}')

        vector\_b\_label.config(text=f'Вектор b:\n {vector\_b}')

    except:

        pass

python\_name\_label = tk.Label(root,

                        text='Python',

                        bg=CL\_BLUE,

                        fg=CL\_YELLOW,

                        font=('Helvetica', 20),

                        ).place(x=150, y=30)

python\_time\_label = tk.Label(root,

                        text='Час виконання: ',

                        bg=CL\_BLUE,

                        fg=CL\_YELLOW,

                        font=('Helvetica', 12),

                        )

python\_time\_label.place(x=100, y=80)

python\_accuracy\_label = tk.Label(root,

                        text='Похибка: ',

                        bg=CL\_BLUE,

                        fg=CL\_YELLOW,

                        font=('Helvetica', 12),

                        )

python\_accuracy\_label.place(x=100, y=110)

python\_result\_label = tk.Label(root,

                        text='Результат: ',

                        bg=CL\_BLUE,

                        fg=CL\_YELLOW,

                        font=('Helvetica', 12),

                        )

python\_result\_label.place(x=100, y=140)

*# ----------------------------------------------- #*

octave\_name\_label = tk.Label(root,

                        text='Octave',

                        bg=CL\_BLUE,

                        fg=CL\_GREEN,

                        font=('Helvetica', 20),

                        ).place(x=540, y=30)

octave\_time\_label = tk.Label(root,

                        text='Час виконання: ',

                        bg=CL\_BLUE,

                        fg=CL\_GREEN,

                        font=('Helvetica', 12),

                        )

octave\_time\_label.place(x=490, y=80)

octave\_accuracy\_label = tk.Label(root,

                        text='Похибка: ',

                        bg=CL\_BLUE,

                        fg=CL\_GREEN,

                        font=('Helvetica', 12),

                        )

octave\_accuracy\_label.place(x=490, y=110)

octave\_result\_label = tk.Label(root,

                        text='Результат: ',

                        bg=CL\_BLUE,

                        fg=CL\_GREEN,

                        font=('Helvetica', 12),

                        )

octave\_result\_label.place(x=490, y=140)

*# ----------------------------------------------- #*

matrix\_A\_label = tk.Label(root,

                        text='Матриця А: ',

                        bg=CL\_BLUE,

                        fg=CL\_RED,

                        font=('Helvetica', 12),

                        )

matrix\_A\_label.place(x=100, y=350)

vector\_b\_label = tk.Label(root,

                        text='Вектор b: ',

                        bg=CL\_BLUE,

                        fg=CL\_RED,

                        font=('Helvetica', 12),

                        )

vector\_b\_label.place(x=490, y=350)

*# ----------------------------------------------- #*

button\_run = tk.Button(root,

                       text='Run',

                       command=run,

                       bg=CL\_RED,

                       fg=CL\_YELLOW,

                       borderwidth=0,

                       font=('Helvetica', 20),

                       ).place(x=360, y=30)

Вміст файлу python\_.py :

import numpy as np

import tkinter.messagebox as messagebox

def run\_python():

    def read\_matrix(filename):

        """Функція для зчитування матриці з файлу

        """

        try:

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

                lines = file.readlines()

                matrix = []

                for line in lines:

                    row = [float(x) for x in line.strip().split()]

                    matrix.append(row)

                return np.array(matrix)

        except FileNotFoundError:

            messagebox.showerror('ERROR', f'Помилка: Файл `{filename}` не знайдено.')

        except ValueError:

            messagebox.showerror('ERROR', f'Помилка: Матриця в файлі `{filename}` містить неправильні значення.')

        except Exception as e:

            messagebox.showerror('ERROR', f'Помилка: Виникла невідома помилка при зчитуванні матриці: {str(e)}')

    def read\_vector(filename):

        """Функція для зчитування вектора з файлу

        """

        try:

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

                lines = file.readlines()

                vector = [float(line.strip()) for line in lines]

                return np.array(vector)

        except FileNotFoundError:

            messagebox.showerror('ERROR', f'Помилка: Файл `{filename}` не знайдено.')

        except ValueError:

            messagebox.showerror('ERROR', f'Помилка: Вектор в файлі `{filename}` містить неправильні значення.')

        except Exception as e:

            messagebox.showerror('ERROR', f'Помилка: Виникла невідома помилка при зчитуванні вектора: {str(e)}')

*# Зчитування матриці A та вектора b з файлів*

    matrix\_A = read\_matrix('Labs\Lab\_1\Matrix\_A.txt')

    vector\_b = read\_vector('Labs\Lab\_1\Vector\_b.txt')

    if len(vector\_b) != len(matrix\_A):

        messagebox.showerror('ERROR', f'Помилка: Вектор b не відповідає розмірності матриці А.')

*# Розв'язання СЛАР Ax = b*

    try:

        solution = np.linalg.solve(matrix\_A, vector\_b)

*# Обчислення похибки*

        accuracy = np.mean((vector\_b - np.dot(matrix\_A, solution)) \*\* 2)

*# print(accuracy)*

*# accuracy = np.max(np.abs(vector\_b - np.dot(matrix\_A, solution)))*

*# print(accuracy)*

        return [matrix\_A, vector\_b, solution, accuracy]

    except:

        pass

if \_\_name\_\_ == '\_\_main\_\_':

    a, b, x, e = run\_python()

    print(f'Matrix:\n{a}')

    print(f'Vector: {b}')

    print(f'Solution: {x}')

    print(f'Accuracy: {e}')

Вміст файлу octave\_.py :

from oct2py import octave

def run\_octave():

    """Використання Octave для завантаження даних та розв'язання системи рівнянь

    """

    octave.eval("A = load('Labs\Lab\_1\Matrix\_A.txt');")

    octave.eval("b = load('Labs\Lab\_1\Vector\_b.txt');")

    octave.eval("x = A \\ b;")

*# Отримання результату з Octave*

    solution = octave.pull('x')

    solution = solution.flatten()

*# Обчислення похибки*

    octave.eval("accuracy = norm(A \* x - b);")

    accuracy = octave.pull('accuracy')

    return [solution, accuracy]

if \_\_name\_\_ == '\_\_main\_\_':

    x, e = run\_octave()

    print(f'Solution: {x}')

    print(f'Accuracy: {e}')