Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 5

з дисципліни «Алгоритми та методи обчислень»

на тему «Розв’язання систем лінійних алгебраїчних рівнянь»

ВИКОНАЛА:
студентка 2 курсу
групи ІВ-92
Бабенко В.В.
Залікова - 9201

ПЕРЕВІРИВ:
Доцент кафедри ОТ
Порєв В.М.

Київ - 2021

Хід роботи

Мета: Вивчити алгоритми методів розв'язання систем лінійних алгебраїчних рівнянь на ЕОМ

Завдання:

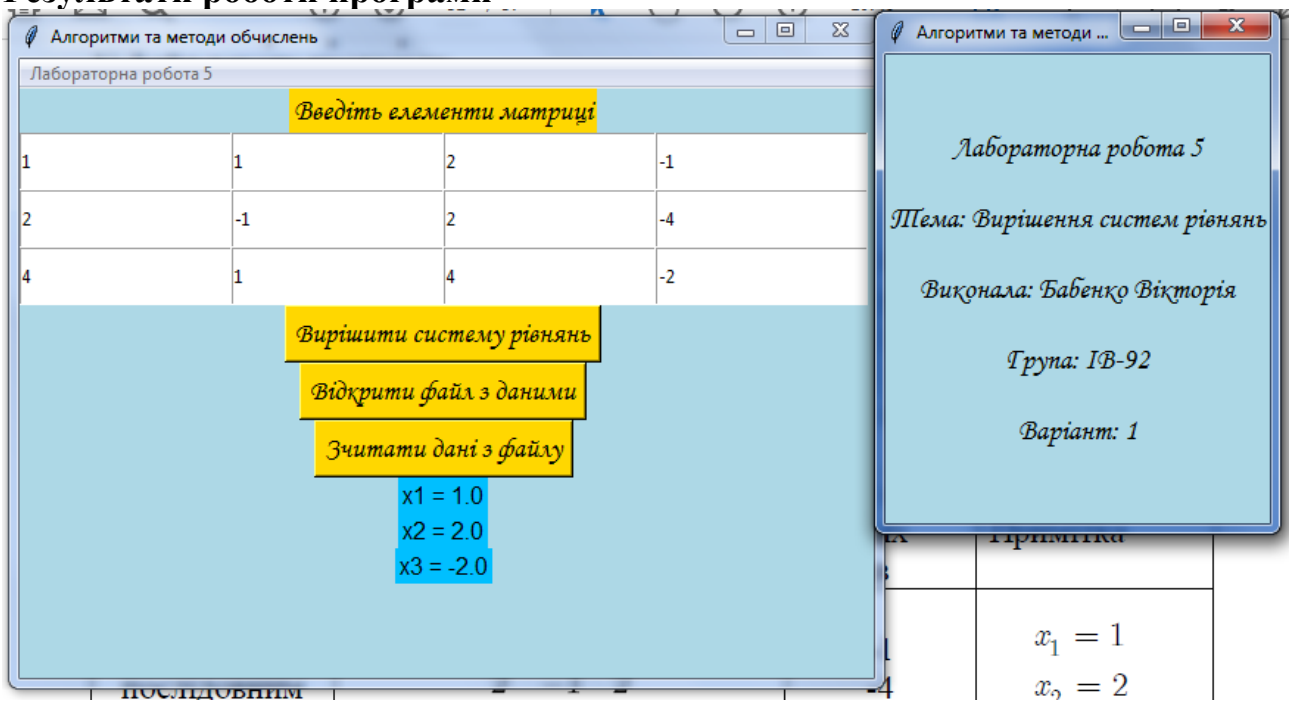
Відповідно до варіанту завдання скласти схему алгоритму розв’язання систем лінійних алгебраїчних рівнянь зазначеним у варіанті методом. Відповідно до блок-схеми скласти програму розв’язання систем лінійних алгебраїчних рівнянь алгоритмічною мовою, узгодженою з викладачем. Розв’язати СЛАР на комп’ютері відповідно до варіанту.

Варіанти завдання

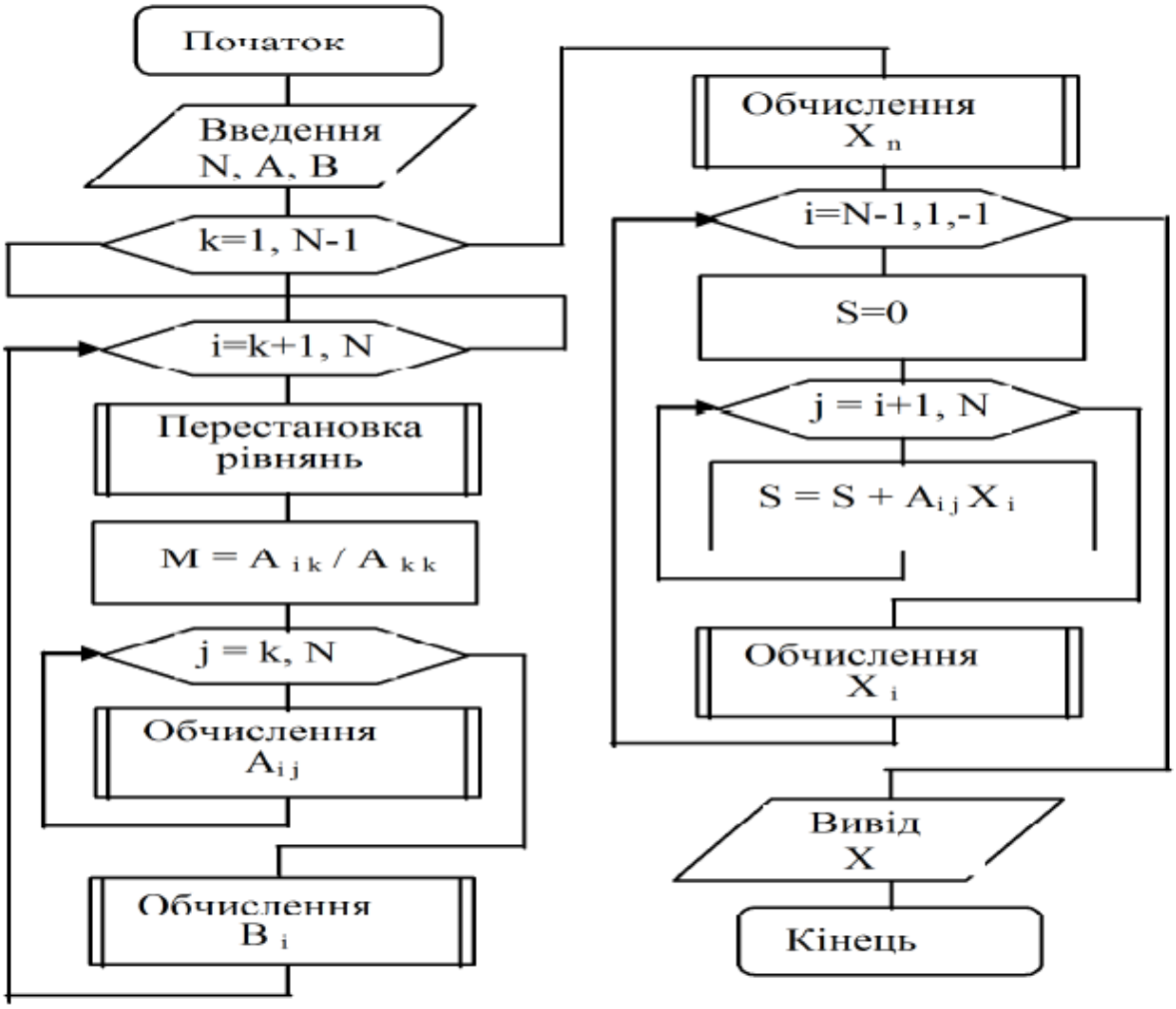
Варіант1

Номер варіанту	Матриця коефіцієнтів системи	Стовпець вільних членів	Примітка
1 Метод Гауса з послідовним виключенням невідомих	$\begin{matrix} 1 & 1 & 2 \\ 2 & -1 & 2 \\ 4 & 1 & 4 \end{matrix}$	-1 -4 -2	$x_1 = 1$ $x_2 = 2$ $x_3 = -2$

Результати роботи програми



Блок-схеми алгоритмів



Блок-схема алгоритму методу Гаусса з послідовним виключенням невідомих

Текстовий опис:

У даному алгоритмі на вхідні значення подаються значення (A, B, N). Прямий хід методу Гауса. Перевіримо, щоб принаймні один із коефіцієнтів a11 , a21 , a31 не дорівнював нулю. Обчислюється множник: M2=a21/a11. Перше рівняння множимо на M2 і віднімаємо від другого рівняння, таким чином виключаємо x1. Далі обчислюємо Мв = ав1/а11, множимо цей множник на перше рівняння і віднімаємо від третього рівняння, тоді коефіцієнт при x1 стає нулем. Таким чином, ми позбулися від змінної x1, всі ті ж маніпуляції проводимо до зведення матриці до трикутної матриці коефіцієнтів. Починаємо зворотній хід методу Гауса Знайти розв'язок такої системи просто: із
3го рівняння знайти x3, підставити результат у друге і знайти x2 , підставити x2 і x3 в 1-е рівняння системи і знайти 1 x .

Призначення індексів в схемі алгоритму (рис. 2):

k – номер рівняння, яке віднімається від інших, а також номер невідомого, яке виключається із залишених $к$-рівнянь;

i – номер рівняння, із якого в даний момент виключається невідоме;

j – номер стовпця.

Роздруківка тексту програми

```
from tkinter import *
from tkinter import messagebox
import subprocess
import file1
root = Tk()
root.geometry("580x400")
root.title("Алгоритми та методи обчислень")
root.config(bg="lightblue")

label1 = Label(root, text="Введіть елементи матриці", font=('Monotype Corsiva',
15))
label1.config(bg="gold")
a11_entry = Entry(root)
a12_entry = Entry(root)
a13_entry = Entry(root)
a14_entry = Entry(root)
a21_entry = Entry(root)
a22_entry = Entry(root)
a23_entry = Entry(root)
a24_entry = Entry(root)
a31_entry = Entry(root)
a32_entry = Entry(root)
a33_entry = Entry(root)
a34_entry = Entry(root)


def solve_task():
    global a11_entry
    global a12_entry
    global a13_entry
    global a14_entry
    global a21_entry
    global a22_entry
    global a23_entry
    global a24_entry
    global a31_entry
    global a32_entry
    global a33_entry
    global a34_entry


    def bubble_max_row(m, col):
        #    Replace matrix[col] row with the one of the underlying rows with
the modulo greatest first element.
        #    :param matrix: matrix (list of lists)
        #    :param col: index of the column/row from which underlying search
will be launched
        #    :return: None. Function changes the matrix structure.
        max_element = m[col][col]
        max_row = col
        for i in range(col + 1, len(m)):
            if abs(m[i][col]) > abs(max_element):
                max_element = m[i][col]
                max_row = i
```

```
            if max_row != col:
                m[col], m[max_row] = m[max_row], m[col]

        def display_results(x):
            x1_label["text"] = "x1 = " + str(x[0])
            x2_label["text"] = "x2 = " + str(x[1])
            x3_label["text"] = "x3 = " + str(x[2])
            for i in x:
                print(i)

        def solve_gauss(m):
            """Solve linear equations system with gaussian method.
            :param m: matrix (list of lists)
            :return: None
            """
            n = len(m)
            # forward trace
            for k in range(n - 1):
                bubble_max_row(m, k)
                for i in range(k + 1, n):
                    div = m[i][k] / m[k][k]
                    m[i][-1] -= div * m[k][-1]
                    for j in range(k, n):
                        m[i][j] -= div * m[k][j]

            # check modified system for nonsingularity
            if is_singular(m):
                print('The system has infinite number of answers...')
                return

            # backward trace
            x = [0 for i in range(n)]
            for k in range(n - 1, -1, -1):
                x[k] = (m[k][-1] - sum([m[k][j] * x[j] for j in range(k + 1, n)])) /
m[k][k]

            # Display results
            display_results(x)

        def is_singular(m):
            """Check matrix for nonsingularity.
            :param m: matrix (list of lists)
            :return: True if system is nonsingular
            """
            for i in range(len(m)):
                if not m[i][i]:
                    return True
            return False

        matrix = [[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
        try:
            matrix[0][0] = float(a11_entry.get())
            matrix[0][1] = float(a12_entry.get())
            matrix[0][2] = float(a13_entry.get())
            matrix[0][3] = float(a14_entry.get())

            matrix[1][0] = float(a21_entry.get())
            matrix[1][1] = float(a22_entry.get())
            matrix[1][2] = float(a23_entry.get())
            matrix[1][3] = float(a24_entry.get())

            matrix[2][0] = float(a22_entry.get())
            matrix[2][1] = float(a22_entry.get())
            matrix[2][2] = float(a22_entry.get())
```

```
            matrix[2][3] = float(a22_entry.get())
        except:
            messagebox.showinfo("Error", "Введіть значення!!!")



        solve_gauss(matrix)

    def open_file():
        subprocess.call(["notepad.exe", "file1.py"])

    def read_file():
        def bubble_max_row(m, col):
            #    Replace matrix[col] row with the one of the underlying rows with
    the modulo greatest first element.
            #    :param matrix: matrix (list of lists)
            #    :param col: index of the column/row from which underlying search
    will be launched
            #    :return: None. Function changes the matrix structure.
            max_element = m[col][col]
            max_row = col
            for i in range(col + 1, len(m)):
                if abs(m[i][col]) > abs(max_element):
                    max_element = m[i][col]
                    max_row = i
            if max_row != col:
                m[col], m[max_row] = m[max_row], m[col]

        def display_results(x):
            x1_label["text"] = "x1 = " + str(x[0])
            x2_label["text"] = "x2 = " + str(x[1])
            x3_label["text"] = "x3 = " + str(x[2])
            for i in x:
                print(i)

        def solve_gauss(m):
            """Solve linear equations system with gaussian method.
            :param m: matrix (list of lists)
            :return: None
            """
            n = len(m)
            # forward trace
            for k in range(n - 1):
                bubble_max_row(m, k)
                for i in range(k + 1, n):
                    div = m[i][k] / m[k][k]
                    m[i][-1] -= div * m[k][-1]
                    for j in range(k, n):
                        m[i][j] -= div * m[k][j]

            # check modified system for nonsingularity
            if is_singular(m):
                x1_label["text"] = "Система має нескінченну кількість розв'язків"
                x2_label["text"] = ""
                x3_label["text"] = ""
                print('The system has infinite number of answers...')
                return

            # backward trace
            x = [0 for i in range(n)]
            for k in range(n - 1, -1, -1):
                x[k] = (m[k][-1] - sum([m[k][j] * x[j] for j in range(k + 1, n)])) /
    m[k][k]
```

```
        # Display results
        display_results(x)

    def is_singular(m):
        """Check matrix for nonsingularity.
        :param m: matrix (list of lists)
        :return: True if system is nonsingular
        """
        for i in range(len(m)):
            if not m[i][i]:
                return True
        return False

    matrix = file1.matrix

    solve_gauss(matrix)


solve_button = Button(root, text="Вирішити систему рівнянь", font=('Monotype
Corsiva', 14), command=solve_task)
open_button = Button(root, text="Відкрити файл з даними", font=('Monotype
Corsiva', 14), command=open_file)
read_button = Button(root, text="Зчитати дані з файлу", font=('Monotype
Corsiva', 14), command=read_file)

solve_button.config(bg="gold")
open_button.config(bg="gold")
read_button.config(bg="gold")

x1_label = Label(root, text="x1 =", font=10)
x2_label = Label(root, text="x2 =", font=10)
x3_label = Label(root, text="x3 =", font=10)
x1_label.config(bg="deep skyblue")
x2_label.config(bg="deep skyblue")
x3_label.config(bg="deep skyblue")
label1.grid(row=0, column=1, columnspan=4)

a11_entry.grid(row=1, column=1, ipadx=10, ipady=10)
a12_entry.grid(row=1, column=2, ipadx=10, ipady=10)
a13_entry.grid(row=1, column=3, ipadx=10, ipady=10)
a14_entry.grid(row=1, column=4, ipadx=10, ipady=10)

a21_entry.grid(row=2, column=1, ipadx=10, ipady=10)
a22_entry.grid(row=2, column=2, ipadx=10, ipady=10)
a23_entry.grid(row=2, column=3, ipadx=10, ipady=10)
a24_entry.grid(row=2, column=4, ipadx=10, ipady=10)

a31_entry.grid(row=3, column=1, ipadx=10, ipady=10)
a32_entry.grid(row=3, column=2, ipadx=10, ipady=10)
a33_entry.grid(row=3, column=3, ipadx=10, ipady=10)
a34_entry.grid(row=3, column=4, ipadx=10, ipady=10)

solve_button.grid(row=4, column=1, columnspan=4)
open_button.grid(row=5, column=1, columnspan=4)
read_button.grid(row=6, column=1, columnspan=4)


x1_label.grid(row=7, column=1, columnspan=4)
x2_label.grid(row=8, column=1, columnspan=4)
x3_label.grid(row=9, column=1, columnspan=4)
```

```
def hello():
    global win1
    if win1 == 0:
        win1 = Toplevel(root)

        label = Label(win1, text="\n\nЛабораторна робота 5\n\nТема: Вирішення
систем рівнянь\n"
                                 "\nВиконала: Бабенко Вікторія\n\nГрупа: ІВ-
92\n\nВаріант: 1\n\n",
                      font=('Monotype Corsiva', 15), bg="lightblue")
        label.pack()
        win1.config(bg="gold")
    else:
        win1.destroy()
        win1 = 0

mainmenu = Menu(root)
root.config(menu=mainmenu)


lab5 = Menu(mainmenu, tearoff=0)

lab5.add_command(label="Автор", command=hello)

mainmenu.add_cascade(label="Лабораторна робота 5", menu=lab5)

win1 = 0


root.mainloop()
```

Аналіз результатів

Реалізація алгоритму методу Гаусса з послідовним виключенням невідомих. Для матриць обмеженого розміру метод є менш трудомістким в порівнянні з іншими методами. Метод дозволяє однозначно встановити чи сумісна система, у випадку сумісності, знаходимо рішення. Метод Гаусса дозволяє знайти максимальне число лінійно-незалежних рівнянь(ранг матриці). Вище було перераховано переваги методу Гауса. Щодо недоліків методу: метод Гауса не буде оптимальним по швидкості для СЛАР з великою розмірністю, у 1969 році Штрассен довів, що матриці можна перемножити за час O(n^log2(7))=O(n^2.81). Саме з цього випливає, що перетворення матриць і рішення СЛАР можна здійснювати алгоритмами асимптотично більш швидкими по порядку ніж метод Гаусса. Рішено систему рівнянь методо Гаусса, це показано на скріншотах результату.

Висновки

В ході виконання цієї роботи було закрілено навички програмного розв’язання систем лінійних рівнянь. Програмно був реалізований метод Гаусса з послідовним виключенням невідомих. Окрім того були закріплені основні навички роботи з графічними бібліотекою tkinter. Всі методи в основному було реалізовано вручну. Отримані результати виконання програми є правильними, це перевірено на тестовому прикладі. Кінцевої мети досягнуто.