

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. І. Сікорського

Кафедра
інформатики та програмної інженерії
(повна назва кафедри, циклової комісії)

КУРСОВА РОБОТА
з «Основ програмування»
(назва дисципліни)
на тему: «Розв'язання систем нелінійних рівнянь»

Студентки, 1 курсу, групи ІІ-15
Ликової Катерини Олександрівни

Спеціальності 121 «Інженерія програмного забезпечення»

Керівник _____ ст. викл. Головченко М. М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Кількість балів: _____
Національна оцінка _____

Члени комісії

_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)
_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)

Київ- 2022 рік

КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. І. Сікорського

(назва вищого навчального закладу)

Кафедра інформатики та програмної інженерії

Дисципліна Основи програмування

Напрямок "ПЗ"

Курс 1 Група ІІ-15

Семестр 2

ЗАВДАННЯ

на курсову роботу студента

Ликової Катерини Олександрівни

(прізвище, ім'я, по батькові)

1. Тема роботи _____

2. Строк здачі студентом закінченої роботи 12.06.2022

3. Вихідні дані до роботи _____

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Дата видачі завдання 10.02.2022

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів курсової роботи	Термін виконання етапів роботи	Підписи керівника, студента
1.	Отримання теми курсової роботи	10.02.2022	
2.	Підготовка ТЗ	02.05.2022	
3.	Пошук та вивчення літератури з питань курсової роботи	03.05.2022	
4.	Розробка сценарію роботи програми	04.05.2022	
6.	Узгодження сценарію роботи програми з керівником	04.05.2022	
5.	Розробка (вибір) алгоритму рішення задачі	04.05.2022	
6.	Узгодження алгоритму з керівником	04.05.2022	
7.	Узгодження з керівником інтерфейсу користувача	05.05.2022	
8.	Розробка програмного забезпечення	06.05.2022	
9.	Налагодження розрахункової частини програми	10.05.2022	
10.	Розробка та налагодження інтерфейсної частини програми	15.05.2022	
11.	Узгодження з керівником набору тестів для контрольного прикладу	25.05.2022	
12.	Тестування програми	26.05.2022	
13.	Підготовка пояснювальної записки	05.06.2022	
14.	Здача курсової роботи на перевірку	12.06.2022	
15.	Захист курсової роботи	19.06.2022	

Студент _____
(підпис)

Керівник _____
(підпис)

Головченко М. М.
(прізвище, ім'я, по батькові)

"12" квітня 2022 р.

АНОТАЦІЯ

Пояснювальна записка до курсової роботи: 53 сторінки, 31 рисунки, 20 таблиць, 2 посилання.

Об'єкт дослідження: задача пошуку точного розв'язку системи нелінійних рівнянь.

Мета роботи: дослідження методів розв'язання систем нелінійних рівнянь та методів розробки програмного забезпечення.

Вивчено метод розробки програмного забезпечення за принципами об'єктно-орієнтованого програмування. Приведені змістовні постановки задач, їх індивідуальні математичні моделі, а також описано детальний процес розв'язання кожної з них.

Виконана програмна реалізація алгоритмів методу Якобі та методу Гауса-Зейделя розв'язання систем нелінійних рівнянь.

СИСТЕМА НЕЛІНІЙНИХ РІВНЯНЬ, МЕТОД ЯКОБІ, МАТРИЦЯ ЯКОБІ, МЕТОД ГАУСА-ЗЕЙДЕЛЯ, ІТЕРАЦІЙНІ МЕТОДИ, ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ.

ЗМІСТ

ВСТУП	6
1 ПОСТАНОВКА ЗАДАЧІ.....	7
2 ТЕОРЕТИЧНІ ВІДОМОСТІ.....	8
2.1. Метод простих ітерацій (Якобі)	8
2.2. Метод Гауса-Зейделя	9
3 ОПИС АЛГОРИТМІВ.....	11
3.1. Загальний алгоритм	11
3.2. Алгоритм методу Якобі.....	12
3.3. Алгоритм методу Гауса-Зейделя.....	13
4 ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	15
4.1. Діаграма класів програмного забезпечення	15
4.2. Опис методів частин програмного забезпечення	15
4.2.1. Стандартні методи	15
4.2.2. Користувацькі методи	18
5 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	23
5.1. План тестування	23
5.2. Приклади тестування.....	23
6 ІНСТРУКЦІЯ КОРИСТУВАЧА	36
6.1. Робота з програмою	36
6.2. Формат вхідних та вихідних даних	40
6.3. Системні вимоги	40
7 АНАЛІЗ РЕЗУЛЬТАТІВ.....	42
ВИСНОВКИ.....	51
ПЕРЕЛІК ПОСИЛАНЬ	52
ДОДАТОК А ТЕХНІЧНЕ ЗАВДАННЯ.....	53

ДОДАТОК Б ТЕКСТИ ПРОГРАМНОГО КОДУ	56
---	----

ВСТУП

Дана курсова робота створена для вирішення задачі розв'язання систем нелінійних рівнянь наближеними (ітераційними) методами.

Завданням даної курсової роботи є розробка програмного забезпечення для розв'язування систем нелінійних рівнянь методами Якобі (простої ітерації) та Гауса-Зейделя з дотриманням принципів об'єктно-орієнтованого програмування.

Вирішення даного завдання є актуальним, оскільки у сучасних математичних розрахунках точність є надзвичайно важливою характеристикою, а ітераційні методи дозволяють задавати необхідну точність, що дозволяє в результаті отримати розв'язки систем нелінійних рівнянь з мінімальною похибкою, якою можна знехтувати.

Ціль даної роботи полягає у вивченні та застосуванні на практиці парадигми об'єктно-орієнтованого програмування.

1 ПОСТАНОВКА ЗАДАЧІ

Розробити програмне забезпечення, що буде знаходити рішення для заданої системи нелінійних рівнянь наступними методами:

- а) метод простої ітерації (Якобі);
- б) метод Гауса-Зейделя.

Вхідними даними для даної роботи є система нелінійних рівнянь, яка задана в одній з наступних форм (на вибір користувача):

$$\text{а) } \begin{cases} a_{11}x_1^2 + a_{12}x_2^2 + a_{13} = 0, \\ a_{21}x_1^2 + a_{22}x_2^2 + a_{23} = 0; \end{cases}$$

$$\text{б) } \begin{cases} \sin(x_1 + a_{11}) + a_{12}x_2 + a_{13} = 0, \\ a_{21}x_1 + \cos(x_2 + a_{22}) + a_{23} = 0; \end{cases}$$

$$\text{в) } \begin{cases} e^{a_{11}x_1} + e^{a_{12}x_2} + a_{13} = 0, \\ e^{a_{21}x_1} + e^{a_{22}x_2} + a_{23} = 0; \end{cases}$$

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix} - \text{матриця коефіцієнтів обраного рівняння, яку задає}$$

користувач.

Програмне забезпечення повинно обробляти матрицю коефіцієнтів для системи нелінійних рівнянь. Кількість невідомих системи нелінійних рівнянь дорівнює 2.

Вихідними даними для даної роботи являється сукупність дійсних чисел, що є розв'язками даної системи, які виводяться на екран. Програмне забезпечення повинно видавати розв'язок за умови, що для вхідних даних обраний метод сходиться. Якщо це не так, то програма повинна вивести відповідне повідомлення. Якщо система не має розв'язків або їх нескінченна кількість, то програма повинна видати відповідне повідомлення.

ітераціях не стануть малими, тобто в якості критерію завершення ітерацій вибирається одна з умов[1]:

$$|x^{(k)} - x^{(k-1)}| = \sqrt{\sum_{i=1}^n (x_i^{(k)} - x_i^{(k-1)})^2} < \varepsilon, \quad (2.4)$$

$$\max_{1 \leq i \leq n} |x_i^{(k)} - x_i^{(k-1)}| < \varepsilon, \quad (2.5)$$

$$\max_{1 \leq i \leq n} \left| \frac{x_i^{(k)} - x_i^{(k-1)}}{x_i^{(k)}} \right| < \varepsilon, \text{ при } |x_i| \gg 1. \quad (2.6)$$

де $\varepsilon > 0$ – допустима похибка розв’язку.

Тут в першому випадку відмінність векторів $x^{(k)}$ і $x^{(k-1)}$ "на ε " розуміється в сенсі малості модуля їх різниці, в другому – в сенсі малості різниць усіх відповідних компонент векторів, в третьому – в сенсі малості відносних різниць компонент.

Умова збіжності ітераційного процесу така: якщо сума модулів елементів рядків або стовпців менша одиниці, то процес ітерації для даної системи збігається до єдиного розв’язку незалежно від вибору початкового вектора. Отже, умову збіжності можна записати так [2]:

$$\sum_{j=1}^n |a_{ij}| < 1, (i = 1, 2, \dots, n) \text{ або } \sum_{i=1}^n |a_{ij}| < 1, (j = 1, 2, \dots, n). \quad (2.6)$$

2.2. Метод Гауса-Зейделя

Систему (2.2) можна розв’язувати і методом Зейделя, що подібний до методу Гауса-Зейделя розв’язання систем лінійних рівнянь. Значення $x_i^{(k)}$ знаходиться з i -го рівняння системи (2.2) з використанням вже вичислених на поточній ітерації значень невідомих. Таким чином, значення невідомих на k -й ітерації знаходитимуться не з допомогою (2.3), а з допомогою співвідношення[1]:

$$x_i^{(k)} = f_i(x_1^{(k)}, \dots, x_{i-1}^{(k)}, x_i^{(k-1)}, \dots, x_n^{(k-1)}), \quad i = 1, 2, \dots, n. \quad (2.7)$$

В якості критерію завершення ітерацій вибирається одна з умов (2.5)-(2.6) як і в методі простої ітерації.

При використанні методу простої ітерації і методу Зейделя успіх багато в чому визначається вдалим вибором початкових наближень невідомих: вони мають бути досить близькими до істинного розв'язку. Інакше ітераційний процес може не збігтися.

3 ОПИС АЛГОРИТМІВ

Перелік всіх основних змінних та їхнє призначення наведено в таблиці 3.1.

Таблиця 3.1 – Основні змінні та їхні призначення

Змінна	Призначення
eps	Точність з якою необхідно знайти корені розв'язків
table	Двовимірний масив коефіцієнтів рівнянь системи (2×3)
x0	Початкове значення першої невідомої
y0	Початкове значення другої невідомої
max	Максимальна кількість ітерацій обраного методу розв'язку системи нелінійних рівнянь
old_x	Значення першої невідомої на минулій ітерації або початкове значення
old_y	Значення другої невідомої на минулій ітерації(k-1) або початкове значення
new_x	Значення першої невідомої на даній ітерації(k)
new_y	Значення другої невідомої на даній ітерації(k)
dif_x	Різниця між значеннями першої невідомої на даній та попередній ітераціях.
dif_y	Різниця між значеннями другої невідомої на даній та попередній ітераціях.

3.1. Загальний алгоритм

- 1 ПОЧАТОК
- 2 Зчитати двовимірний масив коефіцієнтів рівнянь системи:
 - 2.1 Цикл проходу по всіх рядках двовимірного масиву коефіцієнтів рівнянь системи(a_i – поточна строка):
 - 2.1.1 Цикл проходу всіх стовпців двовимірного масиву коефіцієнтів рівнянь системи(a_{ij} – поточний елемент):
 - 2.1.1.1 ЯКЩО поточний елемент двовимірного масиву – вірно записане число, ТО записати його в

відповідну комірку *table*. ІНАКШЕ вивести повідомлення про некоректність введених даних.

- 3 Зчитати точність, початкові значення невідомих та максимальну кількість ітерацій алгоритму:

- 3.1 Зчитати точність:

- 3.1.1 ЯКЩО точність є додатним числом, ТО записати його у змінну *eps*. ІНАКШЕ вивести повідомлення про некоректність введених даних.

- 3.2 Зчитати початкове значення першої невідомої:

- 3.2.1 ЯКЩО початкове значення першої невідомої - вірно записане число, ТО записати його у змінну *x0*. ІНАКШЕ вивести повідомлення про некоректність введених даних.

- 3.3 Зчитати початкове значення другої невідомої:

- 3.3.1 ЯКЩО початкове значення другої невідомої - вірно записане число, ТО записати його у змінну *y0*. ІНАКШЕ вивести повідомлення про некоректність введених даних.

- 3.4 Зчитати максимальну кількість ітерацій алгоритму:

- 3.4.1 ЯКЩО максимальна кількість ітерацій алгоритму - вірно записане число, ТО записати його у змінну *max*. ІНАКШЕ вивести повідомлення про некоректність введених даних.

- 4 ЯКЩО обраний вид системи нелінійних рівнянь:

- 4.1 ЯКЩО введені дані – коректні згідно алгоритму побудови графіку системи, ТО побудувати та вивести графік системи.

- 4.2 ЯКЩО обрано метод Якобі, ТО обрахувати і вивести рішення системи, записати систему та її рішення у файл.

- 4.3 ЯКЩО обрано метод Гауса-Зейделя, ТО обрахувати і вивести рішення системи, записати систему та її рішення у файл.

3.2. Алгоритм методу Якобі

1 ПОЧАТОК

- 2 Присвоїти змінним `new_x` та `new_y` початкові значення невідомих `x0` та `y0`. Присвоїти змінним `dif_x` та `dif_y` значення точності `eps` збільшене на одиницю. Присвоїти лічильнику ітерацій `num` значення 0.
- 3 ПОКИ змінна `dif_x` або `dif_y` більша за значення точності:
 - 3.1 Збільшувати `num` кожної ітерації на 1.
 - 3.2 ЯКЩО `num > max`, ТО присвоїти змінній `res` повідомлення про перевищену кількість ітерацій та вивести її.
 - 3.3 Присвоїти змінним `old_x` та `old_y` значення змінних `new_x` та `new_y`.
 - 3.4 ЯКЩО алгоритм знаходження першої похідної повертає повідомлення, що немає розв'язків, ТО присвоїти змінній `res` дане повідомлення та вивести її.
 - 3.5 Присвоїти `new_x` результат алгоритму знаходження першої невідомої з використанням `old_y`.
 - 3.6 ЯКЩО алгоритм знаходження другої похідної повертає повідомлення, що немає розв'язків, ТО присвоїти змінній `res` дане повідомлення та вивести її.
 - 3.7 Присвоїти `new_y` результат алгоритму знаходження другої невідомої з використанням `old_x`.
 - 3.8 Присвоїти `dif_x` та `dif_y` значення різниці нових та старих значень першої та другої невідомої (`new_x - old_x` та `new_y - old_y`).
 - 3.9 Додати до значення змінної `res` результати ітерації.
- 4 Додати до значення змінної `res` результати розв'язання системи нелінійних рівнянь та вивести її значення.

3.3. Алгоритм методу Гауса-Зейделя

- а) ПОЧАТОК
- б) Присвоїти змінним `new_x` та `new_y` початкові значення невідомих `x0` та `y0`. Присвоїти змінним `dif_x` та `dif_y` значення точності `eps`

збільшене на одиницю. Присвоїти лічильнику ітерацій `num` значення 0.

в) ПОКИ змінна `dif_x` або `dif_y` більша за значення точності:

3.1 Збільшувати `num` кожної ітерації на 1.

3.2 ЯКЩО `num > max`, ТО присвоїти змінній `res` повідомлення про перевищену кількість ітерацій та вивести її.

3.3 Присвоїти змінним `old_x` та `old_y` значення змінних `new_x` та `new_y`.

3.4 ЯКЩО алгоритм знаходження першої похідної повертає повідомлення, що немає розв'язків, ТО присвоїти змінній `res` дане повідомлення та вивести її.

3.5 Присвоїти `new_x` результат алгоритму знаходження першої невідомої з використанням `old_y`.

3.6 ЯКЩО алгоритм знаходження другої похідної повертає повідомлення, що немає розв'язків, ТО присвоїти змінній `res` дане повідомлення та вивести її.

3.7 Присвоїти `new_y` результат алгоритму знаходження другої невідомої з використанням `new_x`.

3.8 Присвоїти `dif_x` та `dif_y` значення різниці нових та старих значень першої та другої невідомої (`new_x - old_x` та `new_y - old_y`).

3.9 Додати до значення змінної `res` результати ітерації.

г) Додати до значення змінної `res` результати розв'язання системи нелінійних рівнянь та вивести її значення.

4 ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1. Діаграма класів програмного забезпечення

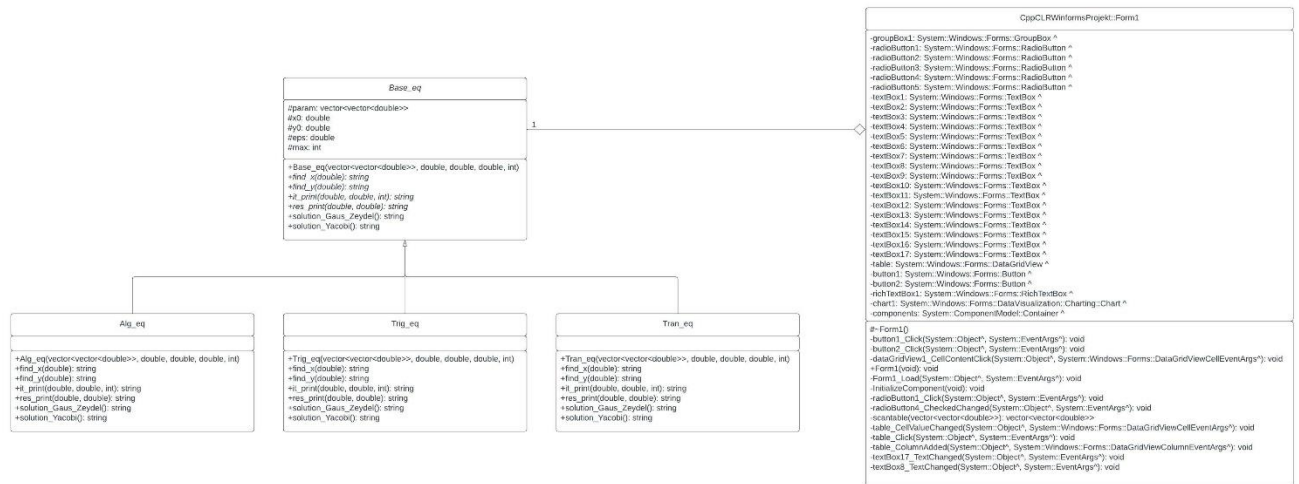


Рисунок 4.1 – Діаграма класів

4.2. Опис методів частин програмного забезпечення

4.2.1. Стандартні методи

У таблиці 4.1 наведено стандартні методи, використані при розробці програмного забезпечення.

Таблиця 4.1 – Стандартні методи

№ п/п	Назва класу	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів	Заголовний файл
1	-	abs	Повертає модуль заданого дійсного числа	x – число типу double	Число y типу double, таке що $y = x \geq 0$ або $y = -x \geq 0$	math.h
2	-	sin	Повертає синус заданого числа	x – число типу double	y - синус числа x $0 \leq y \leq 0$	math.h
3	-	cos	Повертає косинус заданого числа	x – число типу double	y - косинус числа x $0 \leq y \leq 0$	math.h
4	-	stod	Конвертує рядок у число типу double	x – рядок типу string	Число y типу double	-

Продовження таблиці 4.1

№ п/п	Назва класу	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів	Заголовний файл
5	-	to_string	Конвертує величину у рядок типу string	x – число типу double	Рядок у типу string	-
6	-	pow	Повертає число піднесене до певної степені	x – число типу double, n – число типу int	Число $y = x^n$ типу double	math.h
7	-	exp	Повертає число e піднесене до певної степені	x – число типу int	Число $y = e^x$ типу double	math.h
8	-	log	Повертає ступінь, в якій число e дорівнює певному числу	x – число типу double	Число у типу double $e^y = x$	math.h
9	Convert	ToString	Конвертує величину у рядок типу String	x – число типу double	Число у типу String	-
10	Convert	ToInt32	Конвертує величину у число типу int	x – рядок типу String	Число у типу int	-
11	Convert	toDouble	Конвертує величину у число типу double	x – рядок типу String	Число у типу double	-

Продовження таблиці 4.1

№ п/ п	Назва класу	Назва функції	Призначенн я функції	Опис вхідних параметрі в	Опис вихідних параметрі в	Заголовни й файл
12	Int32	TryParse	Перевіряє величину на коректність даних для конвертації у тип int	х – рядок типу String, n – число типу int	Число у > 0 типу int якщо дані коректні, у = 0 якщо ні	-
13	Double	TryParse	Перевіряє величину на коректність даних для конвертації у тип double	х – рядок типу String, n – число типу double	Число у > 0 типу double якщо дані коректні, у = 0 якщо ні	-
14	gcnew	String	Конвертує величину типу string у рядок типу String	х.c_str(), де х – рядок типу string	Рядок у типу String	-
15	DataGridView	AutoResizeColumns	Вирівнює комірки таблиці в залежності від кількості введених цифр	-	-	-

Продовження таблиці 4.1

№ п/п	Назва класу	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів	Заголовний файл
16	AxisScaleView	Zoom	Задає масштаб	х – число типу double, n – число типу double	-	-
17	DataPoint ^>	Clear	Очищує графік	-	-	-
18	DataPointCollection	AddXY	Додає точку на графік	х – число типу double, n – число типу double	-	-

4.2.2. Користувацькі методи

У таблиці 4.2 наведено користувацькі методи, використані при розробці програмного забезпечення.

Таблиця 4.2 – Користувацькі методи

№ п/п	Назва класу	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів	Заголовний файл
1	Base_eq	Base_eq	Задас атрибути класу	x – двовимір- ний масив типу double, n1, n2, n3 – число типу double, n4 – число типу int	y – двовимір- ний масив типу double, n1, n2, n3 – число типу double, n4 – число типу int	base_equations.h
2	Base_eq	find_x	Знаходить значення першої невідомої під час ітерації	x - число типу double	y – рядок типу string	base_equations.h
3	Base_eq	find_y	Знаходить значення другої невідомої під час ітерації	x - число типу double	y – рядок типу string	base_equations.h
4	Base_eq	solution_Yacobi	Знаходить розв’язок системи нелінійних рівнянь методом Якобі	-	y – рядок типу string	base_equations.h

Продовження таблиці 4.2

№ п/ п	Назва класу	Назва функції	Призначенн я функції	Опис вхідних параметрі в	Опис вихідних параметрі в	Заголовний файл
5	Base_e q	solution_Gaus_Zeyd el	Знаходить розв'язок системи нелінійних рівнянь методом Гауса- Зейделя	-	y – рядок типу string	base_equations. h
6	Base_e q	it_print	Задає рядок для виведення результатів ітерації розв'язання	x1, x2 – числа типу double, x3 - число типу int	y – рядок типу string	base_equations. h
7	Base_e q	res_print	Задає рядок для виведення результатів розв'язання системи нелінійних рівнянь	x1, x2 – числа типу double	y – рядок типу string	base_equations. h
8	Form1	scantable	Заповнює двовимірний масив введеними в таблицю значеннями	x – двовимір- ний масив типу double	y – двовимір- ний масив типу double	Form1.h

Продовження таблиці 4.2

№ п/ п	Назва класу	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрі в	Заголов ний файл
9	Form1	button1_Click	Перевіряє дані на коректність, розв'язує обрану систему нелінійних рівнянь обраним методом, виводить результат на екран та зберігає його у текстовий файл	Змінна x типу System::Object ^ та n типу System::EventArgs^	-	Form1.h
10	Form1	table_CellValueChanged	Змінює розмір комірки таблиці в залежності від введених значень після кожної зміни значення	Змінна x типу System::Object ^ та n типу System::Windows::Forms::DataGridViewCellEventArgs^	-	Form1.h

Продовження таблиці 4.2

№ п/ п	Назва класу	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрі в	Заголов ний файл
11	Form1	Form1_Load	Задає розмірність таблиці (2×3), задає розміри комірок таблиці, налагоджує графік (масштаб, можливість користувача збільшувати або зменшувати масштаб)	Змінна x типу System::Object ^ та n типу System::EventArgs	-	Form1.h
12	Form1	button2_Click	Перевіряє дані на коректність, будує графік обраної системи нелінійних рівнянь	Змінна x типу System::Object ^ та n типу System::EventArgs	-	Form1.h
13	-	writeTextFile	Записує результат розв'язання системи нелінійних рівнянь у текстовий файл	Змінна x типу string	-	func.h

5 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

5.1. План тестування

Тестування програмного забезпечення, яке включає тестування основного функціоналу програми та тестування реакцій на виключні ситуації, буде проведено за наступним планом:

- а) Тестування коректності введених значень.
 - 1) Тестування при введенні некоректних символів.
 - 2) Тестування при введенні нульового або від'ємного значення точності.
 - 3) Тестування при введенні нульового або від'ємного значення максимальної кількості ітерацій.
- б) Тестування коректного запуску розв'язання.
 - 1) Тестування при запуску розв'язання без обраного виду системи.
 - 2) Тестування при запуску розв'язання без обраного методу розв'язання.
- в) Тестування коректної роботи при введенні систем, що не мають коренів.
- г) Тестування коректної роботи при введенні наближених початкових значень невідомих, що призводять до розбіжного ітераційного процесу.
- д) Тестування коректності роботи методів Якобі та Гауса-Зейделя.
 - 1) Перевірка коректності роботи методу Якобі.
 - 2) Перевірка коректності роботи методу Гауса-Зейделя.
- е) Тестування побудови графіків.
 - 1) Тестування побудови графіків без обраного виду системи.
 - 2) Тестування коректності побудови графіків.
- ж) Тестування збереження системи та результатів у файл.

5.2. Приклади тестування

Проведемо тестування програмного забезпечення за зазначеним вище планом. Під час тестів будемо фіксувати мету тесту, початковий стан програми,

вхідні дані, схему проведення тесту, очікуваний результат та стан програми після проведення випробувань в окремій таблиці для кожного тесту.

Результати тестування за зазначеним вище планом наведено в таблицях 5.1 – 5.12.

Таблиця 5.1 – Приклад роботи програми при введенні некоректних символів.

Мета тесту	Перевірити можливість введення некоректних даних
Початковий стан програми	Відкрите вікно програми
Вхідні дані	<p>Тест 1 – Коефіцієнти: ; -8 9 -9 2 1</p> <p>Тест 2 – Коефіцієнти: v -9 2 { 3 '}</p> <p>Тест 3 – Коефіцієнти: 2 & -0.1 ! 5 -9</p> <p>Тест 4 – Наближені початкові значення невідомих: * 2</p> <p>Тест 5 – Наближені початкові значення невідомих: 8 8-</p> <p>Тест 6 – Наближені початкові значення невідомих: [0,5</p> <p>Тест 7 – Точність: 0.1</p> <p>Тест 8 – Точність: -p</p> <p>Тест 9 – Точність: <</p> <p>Тест 10 – Максимальна кількість ітерацій: 7m</p> <p>Тест 11 – Максимальна кількість ітерацій: +</p> <p>Тест 12 – Максимальна кількість ітерацій: 0,7</p>
Схема проведення тесту	Поелементне заповнення таблиці коефіцієнтів, ввести значення наближених початкових невідомих, точності, максимальної кількості ітерацій і натиснути «Знайти корені»
Очікуваний результат	Вивід повідомлення про помилку формату даних
Стан програми після проведення випробувань	Вивід «Некоректно введені дані»

Таблиця 5.2 – Приклад роботи програми при введенні нульового або від’ємного значення точності.

Мета тесту	Перевірити можливість введення від’ємного значення точності
Початковий стан програми	Відкрите вікно програми
Вхідні дані	Тест 1 – Точність: -2 Тест 2 – Точність: 0 Тест 3 – Точність: -0,5
Схема проведення тесту	Ввести значення точності і натиснути «Знайти корені»
Очікуваний результат	Вивід повідомлення про помилку формату даних
Стан програми після проведення випробувань	Вивід «Некоректно введені дані»

Таблиця 5.3 – Приклад роботи програми при введенні нульового або від’ємного значення максимальної кількості ітерацій.

Мета тесту	Перевірити можливість введення нульового або від’ємного значення максимальної кількості ітерацій
Початковий стан програми	Відкрите вікно програми
Вхідні дані	Тест 1 – Максимальна кількість ітерацій: -7 Тест 2 – Максимальна кількість ітерацій: 0 Тест 3 – Максимальна кількість ітерацій: -20
Схема проведення тесту	Ввести значення максимальної кількості ітерацій і натиснути «Знайти корені»
Очікуваний результат	Вивід повідомлення про помилку формату даних
Стан програми після проведення випробувань	Вивід «Некоректно введені дані»

Таблиця 5.4 – Приклад роботи програми при запуску розв’язання без обраного виду системи.

Мета тесту	Перевірити можливість запуску розв’язання без обраного виду системи
Початковий стан програми	Відкрите вікно програми

Продовження таблиці 5.4

Вхідні дані	Тест 1 – Коефіцієнти: 1 2 3 -9 5 3 Наближені початкові значення невідомих: 1 2 Точність: 0,001 Максимальна кількість ітерацій: 100 Тест 2 – Коефіцієнти: 1,7 27,5 -3 -3,6 0,5 29 Наближені початкові значення невідомих: 0 0 Точність: 0,0001 Максимальна кількість ітерацій: 150 Тест 3 – Коефіцієнти: -8 27,1 -0,01 1 1 1 Наближені початкові значення невідомих: -2 -4 Точність: 0,00001 Максимальна кількість ітерацій: 200
Схема проведення тесту	Поелементне заповнення таблиці коефіцієнтів, ввести значення наближених початкових невідомих, точності, максимальної кількості ітерацій і натиснути «Знайти корені»
Очікуваний результат	Відсутність розв'язку
Стан програми після проведення випробувань	Вивід «»

Таблиця 5.5 – Приклад роботи програми при запуску розв'язання без обраного методу розв'язання.

Мета тесту	Перевірити можливість запуску розв'язання без обраного методу розв'язання
Початковий стан програми	Відкрите вікно програми

Продовження таблиці 5.5

Вхідні дані	Тест 1 – Коефіцієнти: 1,2 82 -3 -0,9 55 3 Наближені початкові значення невідомих: -1,4 0,3 Точність: 0,01 Максимальна кількість ітерацій: 300 Вид системи: алгебраїчна Тест 2 – Коефіцієнти: -15,7 17,5 -36 -3,66 6,55 5 Наближені початкові значення невідомих: 0 -0,8 Точність: 0,00001 Максимальна кількість ітерацій: 1500 Вид системи: тригонометрична Тест 3 – Коефіцієнти: 78 11,5 -9,01 1 5 7 Наближені початкові значення невідомих: -1 10 Точність: 0,1 Максимальна кількість ітерацій: 2000 Вид системи: трансцендентна
Схема проведення тесту	Поелементне заповнення таблиці коефіцієнтів, ввести значення наближених початкових невідомих, точності, максимальної кількості ітерацій, обрати вид системи і натиснути «Знайти корені»
Очікуваний результат	Відсутність розв'язку
Стан програми після проведення випробувань	Вивід «»

Таблиця 5.6 – Приклад роботи програми при введенні систем, що не мають коренів.

Мета тесту	Перевірити можливість запуску розв'язання при введенні систем, що не мають коренів
Початковий стан програми	Відкрите вікно програми

Продовження таблиці 5.6

Вхідні дані	<p>Тест 1 – Коефіцієнти: 1 1 1 3 2 2</p> <p>Наближені початкові значення невідомих: 0 0</p> <p>Точність: 0,0001</p> <p>Максимальна кількість ітерацій: 300</p> <p>Вид системи: алгебраїчна</p> <p>Метод розв'язання: Якобі</p> <p>Тест 2 – Коефіцієнти: -4 -5 -32 2 4 5</p> <p>Наближені початкові значення невідомих: 0 -0,8</p> <p>Точність: 0,00001</p> <p>Максимальна кількість ітерацій: 1500</p> <p>Вид системи: алгебраїчна</p> <p>Метод розв'язання: Гауса-Зейделя</p> <p>Тест 3 – Коефіцієнти: -7 -6 3,4 1 5 7,11</p> <p>Наближені початкові значення невідомих: -2 4</p> <p>Точність: 0,0001</p> <p>Максимальна кількість ітерацій: 130</p> <p>Вид системи: трансцендентна</p> <p>Метод розв'язання: Якобі</p>
Схема проведення тесту	<p>Поелементне заповнення таблиці коефіцієнтів, ввести значення наближених початкових невідомих, точності, максимальної кількості ітерацій, обрати вид системи, метод розв'язання і натиснути «Знайти корені»</p>
Очікуваний результат	Відсутність розв'язків
Стан програми після проведення випробувань	Вивід «Немає розв'язків»

Таблиця 5.7 – Приклад роботи програми при введенні наближених початкових значень невідомих, що призводять до розбіжного ітераційного процесу.

Мета тесту	Перевірити можливість запуску розв'язання при введенні наближених початкових значень невідомих, що призводять до розбіжного ітераційного процесу
Початковий стан програми	Відкрите вікно програми
Вхідні дані	<p>Тест 1 – Коефіцієнти: 1 1 -10 2 0,9 -10 Наближені початкові значення невідомих: 0 0 Точність: 0,0001 Максимальна кількість ітерацій: 100 Вид системи: алгебраїчна Метод розв'язання: Якобі</p> <p>Тест 2 – Коефіцієнти: 1 1 -10 0,01 0,9 -10 Наближені початкові значення невідомих: 5 2 Точність: 0,01 Максимальна кількість ітерацій: 1000 Вид системи: тригонометрична Метод розв'язання: Гауса-Зейделя</p> <p>Тест 3 – Коефіцієнти: 1 1 -9 5 -1 -1 Наближені початкові значення невідомих: 0,5 1,7 Точність: 0,0001 Максимальна кількість ітерацій: 500 Вид системи: трансцендентна Метод розв'язання: Якобі</p>
Схема проведення тесту	Поелементне заповнення таблиці коефіцієнтів, ввести значення наближених початкових невідомих, точності, максимальної кількості ітерацій, обрати вид системи, метод розв'язання і натиснути «Знайти корені»
Очікуваний результат	Відсутність розв'язків
Стан програми після проведення випробувань	Вивід «Немає розв'язків» або «Кількість ітерацій перевищила максимальну. Немає розв'язків»

Таблиця 5.8 – Приклад роботи програми при обраному методі Якобі.

Мета тесту	Перевірити коректність роботи методу Якобі
Початковий стан програми	Відкрите вікно програми
Вхідні дані	<p>Тест 1 – Коефіцієнти: 2 -2 -1 0,1 1 -10 Наближені початкові значення невідомих: 6,7 7 Точність: 0,001 Максимальна кількість ітерацій: 3000 Вид системи: алгебраїчна Метод розв'язання: Якобі</p> <p>Тест 2 – Коефіцієнти: 3 2 4 7 6 5 Наближені початкові значення невідомих: 0 -2 Точність: 0,00001 Максимальна кількість ітерацій: 1000 Вид системи: тригонометрична Метод розв'язання: Якобі</p> <p>Тест 3 – Коефіцієнти: 2 20 -0,01 0,1 -1 -6 Наближені початкові значення невідомих: -2 -3 Точність: 0,0001 Максимальна кількість ітерацій: 500 Вид системи: трансцендентна Метод розв'язання: Якобі</p>
Схема проведення тесту	Поелементне заповнення таблиці коефіцієнтів, ввести значення наближених початкових невідомих, точності, максимальної кількості ітерацій, обрати вид системи, метод розв'язання і натиснути «Знайти корені»
Очікуваний результат	<p>Тест 1 – $x = \pm 3,09$; $y = \pm 3,008$</p> <p>Тест 2 – $x = -0,591$; $y = -2,334$</p> <p>Тест 3 – $x = -2,302$; $y = -1,65$</p>
Стан програми після проведення випробувань	<p>Виводить ітерації та результат</p> <p>Тест 1 – $x = \pm 3,090218$; $y = \pm 3,007507$ (7 ітерацій)</p> <p>Тест 2 – $x = -0,590607$; $y = -2,334254$ (8 ітерацій)</p> <p>Тест 3 – $x = -2,302585$; $y = -1,649749$ (3 ітерації)</p>

Таблиця 5.9 – Приклад роботи програми при обраному методі Гауса-Зейделя.

Мета тесту	Перевірити коректність роботи методу Гауса-Зейделя
Початковий стан програми	Відкрите вікно програми
Вхідні дані	<p>Тест 1 – Коефіцієнти: 2 -2 -1 0,1 1 -10 Наближені початкові значення невідомих: 6,7 7 Точність: 0,001 Максимальна кількість ітерацій: 3000 Вид системи: алгебраїчна Метод розв'язання: Гауса-Зейделя Тест 2 – Коефіцієнти: 3 2 4 7 6 5 Наближені початкові значення невідомих: 0 -2 Точність: 0,00001 Максимальна кількість ітерацій: 1000 Вид системи: тригонометрична Метод розв'язання: Гауса-Зейделя Тест 3 – Коефіцієнти: 2 20 -0,01 0,1 -1 -6 Наближені початкові значення невідомих: -2 -3 Точність: 0,0001 Максимальна кількість ітерацій: 500 Вид системи: трансцендентна Метод розв'язання: Гауса-Зейделя</p>
Схема проведення тесту	Поелементне заповнення таблиці коефіцієнтів, ввести значення наближених початкових невідомих, точності, максимальної кількості ітерацій, обрати вид системи, метод розв'язання і натиснути «Знайти корені»
Очікуваний результат	<p>Тест 1 – $x = \pm 3,09$; $y = \pm 3,008$ Тест 2 – $x = -0,591$; $y = -2,334$ Тест 3 – $x = -0,921$; $y = -1,792$</p>
Стан програми після проведення випробувань	<p>Тест 1 – $x = \pm 3,089507$; $y = \pm 3,007573$ (6 ітерацій) Тест 2 – $x = -0,590607$; $y = -2,334254$ (5 ітерацій) Тест 3 – $x = -2,302585$; $y = -1,649749$ (2 ітерації)</p>

Таблиця 5.10 – Приклад побудови графіків без обраного виду системи.

Мета тесту	Перевірити можливість побудови графіків без обраного виду системи
Початковий стан програми	Відкрите вікно програми
Вхідні дані	Тест 1 – Коефіцієнти: -0,566 56 1,5 30,9 -2 66 Тест 2 – Коефіцієнти: 4,512 -67 2 -11 -3,8 0,1 Тест 3 – Коефіцієнти: 4 7 5 3,1 -25 0,11
Схема проведення тесту	Поелементне заповнення таблиці коефіцієнтів і натиснути «Побудувати графік»
Очікуваний результат	Відсутність побудови графіка
Стан програми після проведення випробувань	Побудова графіка не відбувається

Таблиця 5.11 – Приклад побудови графіків.

Мета тесту	Перевірити коректність побудови графіків
Початковий стан програми	Відкрите вікно програми
Вхідні дані	Тест 1 – Коефіцієнти: 1 -5 -6 2 2 -30 Вид системи: алгебраїчна Тест 2 – Коефіцієнти: -98,7 -7 67 -1 5 -0,65 Вид системи: тригонометрична Тест 3 – Коефіцієнти: 2 0,1 -2 0,5 -0,01 -1 Вид системи: трансцендентна
Схема проведення тесту	Поелементне заповнення таблиці коефіцієнтів, обрати вид системи і натиснути «Побудувати графік»
Очікуваний результат	Побудова графіку

Продовження таблиці 5.11

Стан програми після
проведення випробувань

Тест 1 (Рисунок 5.1):

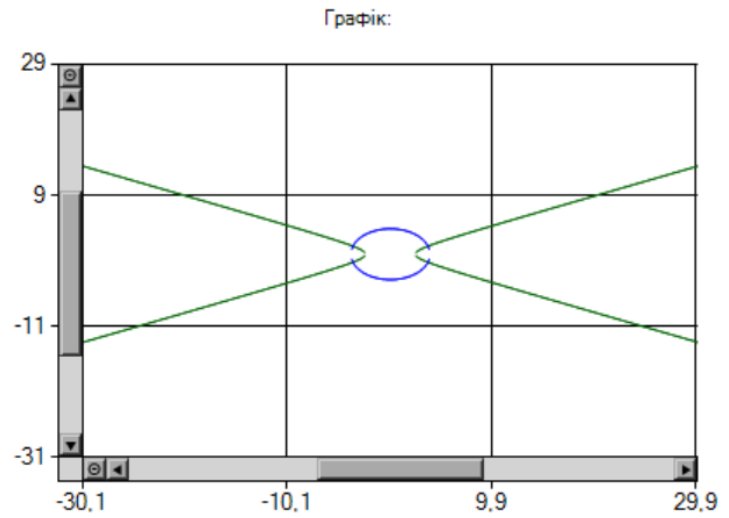


Рисунок 5.1 – Графіки для тесту 1

Тест 2 (Рисунок 5.2):

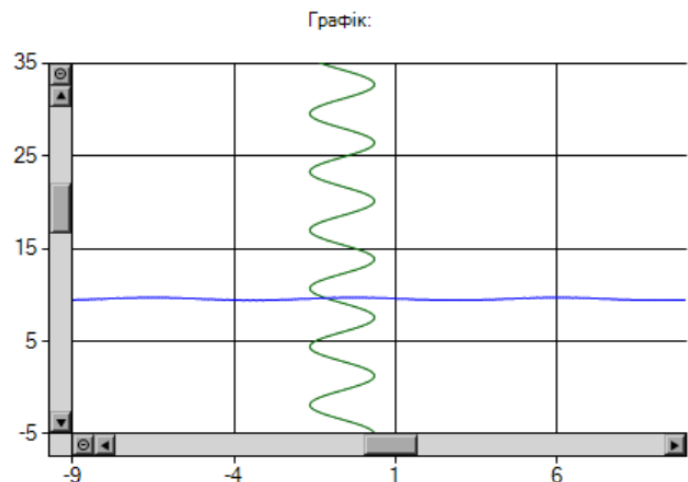


Рисунок 5.2 – Графіки для тесту 2

Тест 3 (Рисунок 5.3):

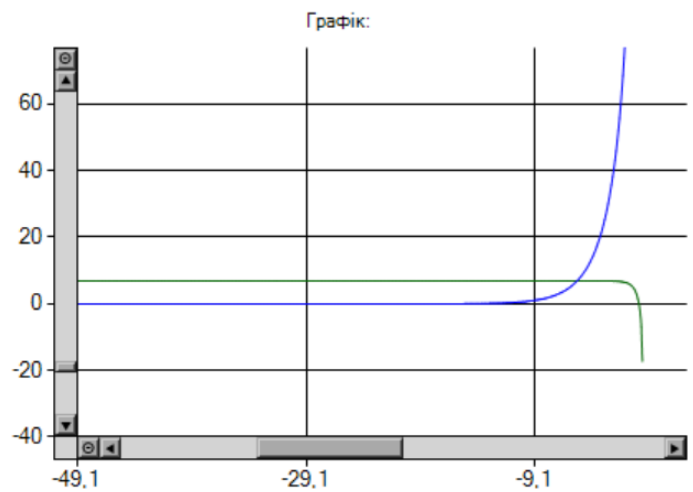
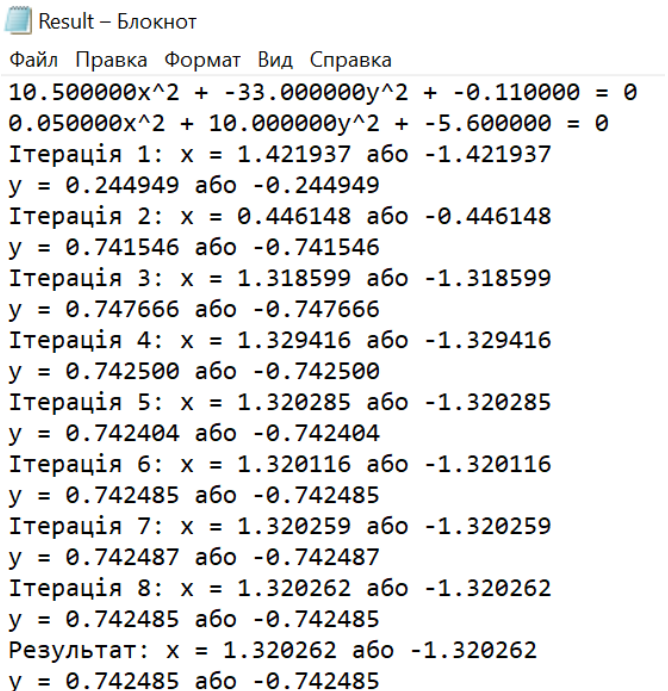
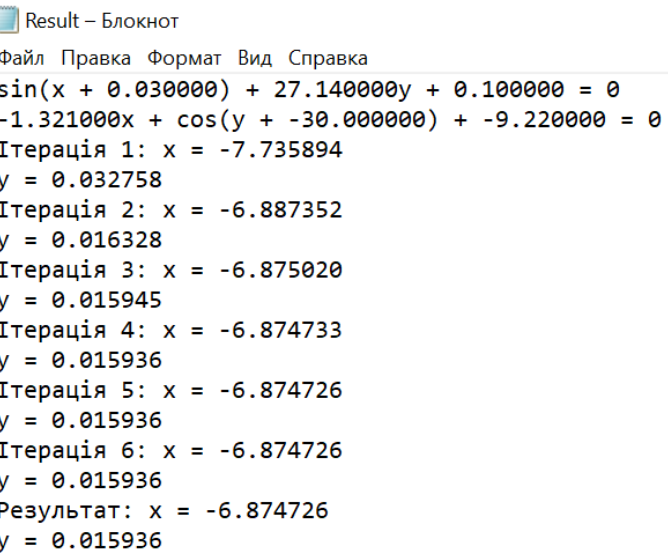
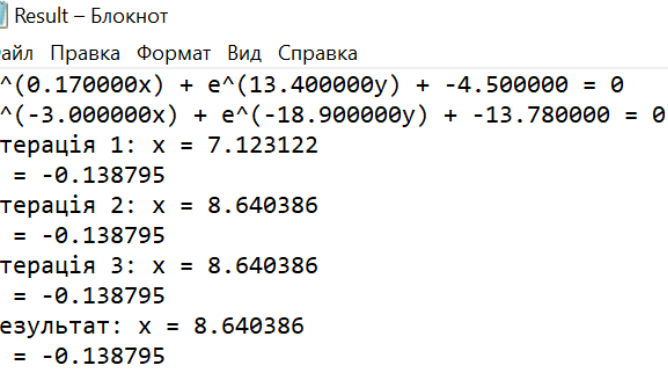


Рисунок 5.3 – Графіки для тесту 3

Таблиця 5.12 – Приклад збереження системи та результатів у файл.

Мета тесту	Перевірити можливість збереження системи та результатів у файл
Початковий стан програми	Відкрите вікно програми
Вхідні дані	<p>Тест 1 – Коефіцієнти: 10,5 -33 -0,11 0,05 10 -5,6 Наближені початкові значення невідомих: 10 0,8 Точність: 0,00001 Максимальна кількість ітерацій: 256 Вид системи: алгебраїчна Метод розв'язання: Якобі Тест 2 – Коефіцієнти: 0,03 27,14 0,1 -1,321 -30 -9,22 Наближені початкові значення невідомих: 185 14,25 Точність: 0,0000001 Максимальна кількість ітерацій: 2897 Вид системи: тригонометрична Метод розв'язання: Гауса-Зейделя Тест 3 – Коефіцієнти: 0,17 13,4 -4,5 -3 -18,9 -13,78 Наближені початкові значення невідомих: 5,2 0,01 Точність: 0,001 Максимальна кількість ітерацій: 1555 Вид системи: трансцендентна Метод розв'язання: Гауса-Зейделя</p>
Схема проведення тесту	Поелементне заповнення таблиці коефіцієнтів, ввести значення наближених початкових невідомих, точності, максимальної кількості ітерацій, обрати вид системи, метод розв'язання і натиснути «Знайти корені». Відкрити файл «Result»
Очікуваний результат	Відображення системи та результатів у файлі

Продовження таблиці 5.12

<p>Стан програми після проведення випробувань</p>	<p>Тест 1 (Рисунок 5.4):</p>  <p>Result – Блокнот</p> <p>Файл Правка Формат Вид Справка</p> $10.500000x^2 + -33.000000y^2 + -0.110000 = 0$ $0.050000x^2 + 10.000000y^2 + -5.600000 = 0$ <p>Ітерація 1: x = 1.421937 або -1.421937 y = 0.244949 або -0.244949</p> <p>Ітерація 2: x = 0.446148 або -0.446148 y = 0.741546 або -0.741546</p> <p>Ітерація 3: x = 1.318599 або -1.318599 y = 0.747666 або -0.747666</p> <p>Ітерація 4: x = 1.329416 або -1.329416 y = 0.742500 або -0.742500</p> <p>Ітерація 5: x = 1.320285 або -1.320285 y = 0.742404 або -0.742404</p> <p>Ітерація 6: x = 1.320116 або -1.320116 y = 0.742485 або -0.742485</p> <p>Ітерація 7: x = 1.320259 або -1.320259 y = 0.742487 або -0.742487</p> <p>Ітерація 8: x = 1.320262 або -1.320262 y = 0.742485 або -0.742485</p> <p>Результат: x = 1.320262 або -1.320262 y = 0.742485 або -0.742485</p> <p>Рисунок 5.4 – Графіки для тесту 1</p> <p>Тест 2 (Рисунок 5.5):</p>  <p>Result – Блокнот</p> <p>Файл Правка Формат Вид Справка</p> $\sin(x + 0.030000) + 27.140000y + 0.100000 = 0$ $-1.321000x + \cos(y + -30.000000) + -9.220000 = 0$ <p>Ітерація 1: x = -7.735894 y = 0.032758</p> <p>Ітерація 2: x = -6.887352 y = 0.016328</p> <p>Ітерація 3: x = -6.875020 y = 0.015945</p> <p>Ітерація 4: x = -6.874733 y = 0.015936</p> <p>Ітерація 5: x = -6.874726 y = 0.015936</p> <p>Ітерація 6: x = -6.874726 y = 0.015936</p> <p>Результат: x = -6.874726 y = 0.015936</p> <p>Рисунок 5.5 – Графіки для тесту 2</p> <p>Тест 3 (Рисунок 5.6):</p>  <p>Result – Блокнот</p> <p>Файл Правка Формат Вид Справка</p> $e^{(0.170000x)} + e^{(13.400000y)} + -4.500000 = 0$ $e^{(-3.000000x)} + e^{(-18.900000y)} + -13.780000 = 0$ <p>Ітерація 1: x = 7.123122 y = -0.138795</p> <p>Ітерація 2: x = 8.640386 y = -0.138795</p> <p>Ітерація 3: x = 8.640386 y = -0.138795</p> <p>Результат: x = 8.640386 y = -0.138795</p> <p>Рисунок 5.6 – Графіки для тесту 3</p>
---	--

6 ІНСТРУКЦІЯ КОРИСТУВАЧА

6.1. Робота з програмою

Після запуску виконавчого файлу з розширенням *.exe, відкривається головне вікно програми (Рисунок 6.1).

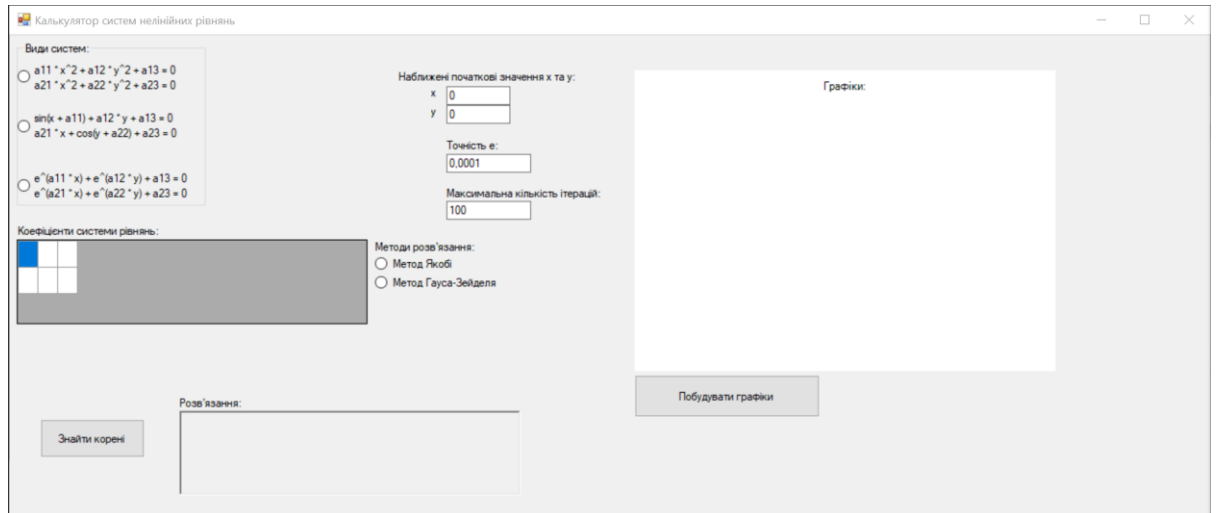


Рисунок 6.1 – Головне вікно програми

Далі серед запропонованого списку видів систем під назвою «Види систем» шляхом натиску на кнопку навпроти системи необхідно обрати вид системи, який буде оброблятися програмою (Рисунок 6.2).

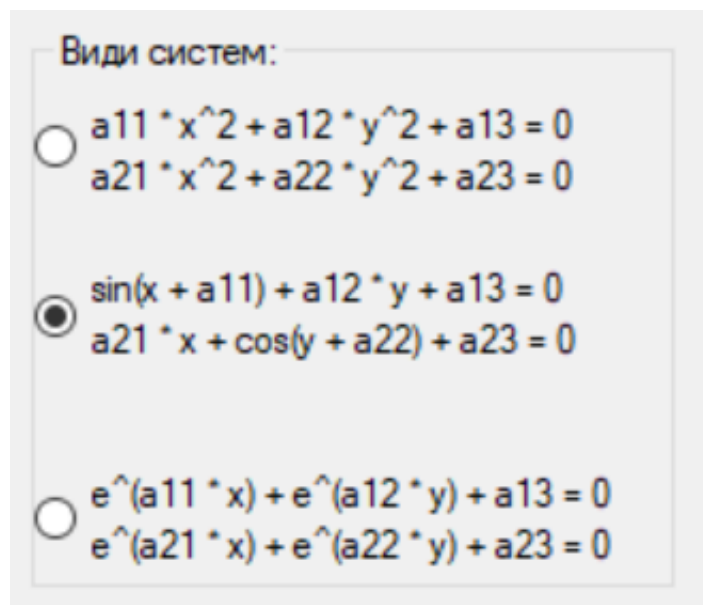


Рисунок 6.2 – Вибір необхідного виду системи

Наступним кроком за допомогою таблиці під назвою «Коефіцієнти системи рівнянь» шляхом натиску на комірки таблиці та введення чисел з клавіатури необхідно задати коефіцієнти системи рівнянь (Рисунок 6.3).

Коефіцієнти системи рівнянь:

1		
	1	

Рисунок 6.3 – Вибір необхідних коефіцієнтів системи рівнянь

Потім серед запропонованого списку методів розв'язання систем під назвою «Методи розв'язання» шляхом натиску на кнопку навпроти методу необхідно обрати метод розв'язання системи, який буде оброблятися програмою (Рисунок 6.4).

Методи розв'язання:

☒ Метод Якобі

☐ Метод Гауса-Зейделя

Рисунок 6.4 – Вибір необхідного методу розв'язання системи

Після цього за допомогою комірок під назвою «Наближені початкові значення x та y » шляхом натиску на комірки та введення чисел з клавіатури необхідно задати наближені початкові значення двох невідомих (Рисунок 6.5).

Наближені початкові значення x та y :

x	15
y	0

Рисунок 6.5 – Вибір необхідних наближених початкових значень двох невідомих

Далі за допомогою комірки під назвою «Точність ϵ » шляхом натиску на комірку та введення чисел з клавіатури необхідно задати точність розрахунків (Рисунок 6.6).

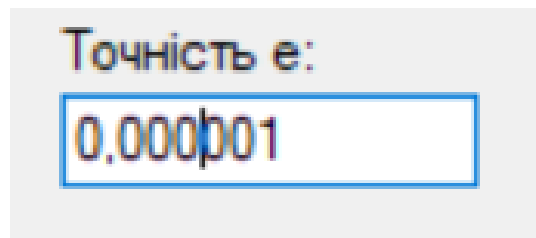


Рисунок 6.6 – Вибір необхідної точності розрахунків

Наступним кроком за допомогою комірки під назвою «Максимальна кількість ітерацій» шляхом натиску на комірку та введення чисел з клавіатури необхідно задати максимальну кількість ітерацій під час розрахунків (Рисунок 6.7).

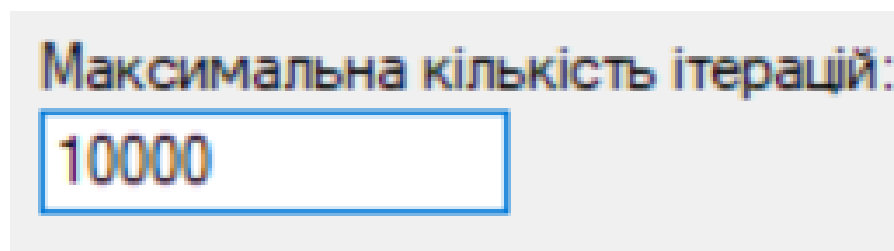


Рисунок 6.7 – Вибір необхідної максимальної кількості ітерацій під час розрахунків

Потім необхідно натиснути на кнопку під назвою «Знайти корені» для того, щоб вивести розв'язання системи у комірці під назвою «Розв'язання» (Рисунок 6.8).

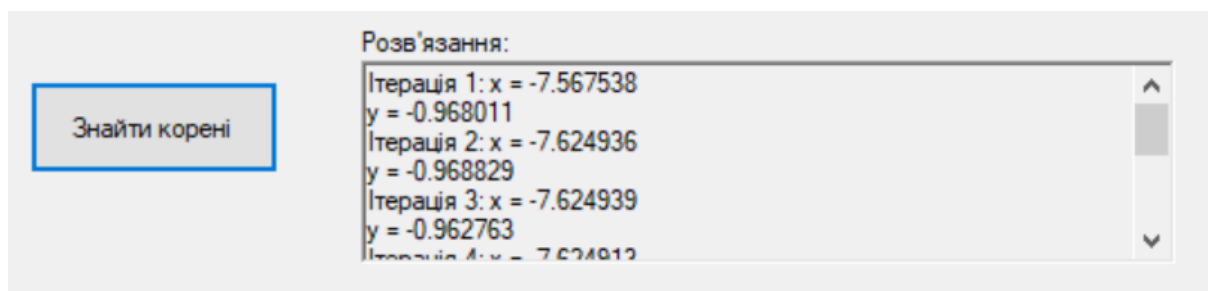


Рисунок 6.8 – Виконання розв'язання, виведення результату у комірці

Після цього необхідно натиснути на кнопку під назвою «Побудувати графіки» для того, щоб побудувати графіки системи у комірці під назвою «Графіки» (Рисунок 6.9).

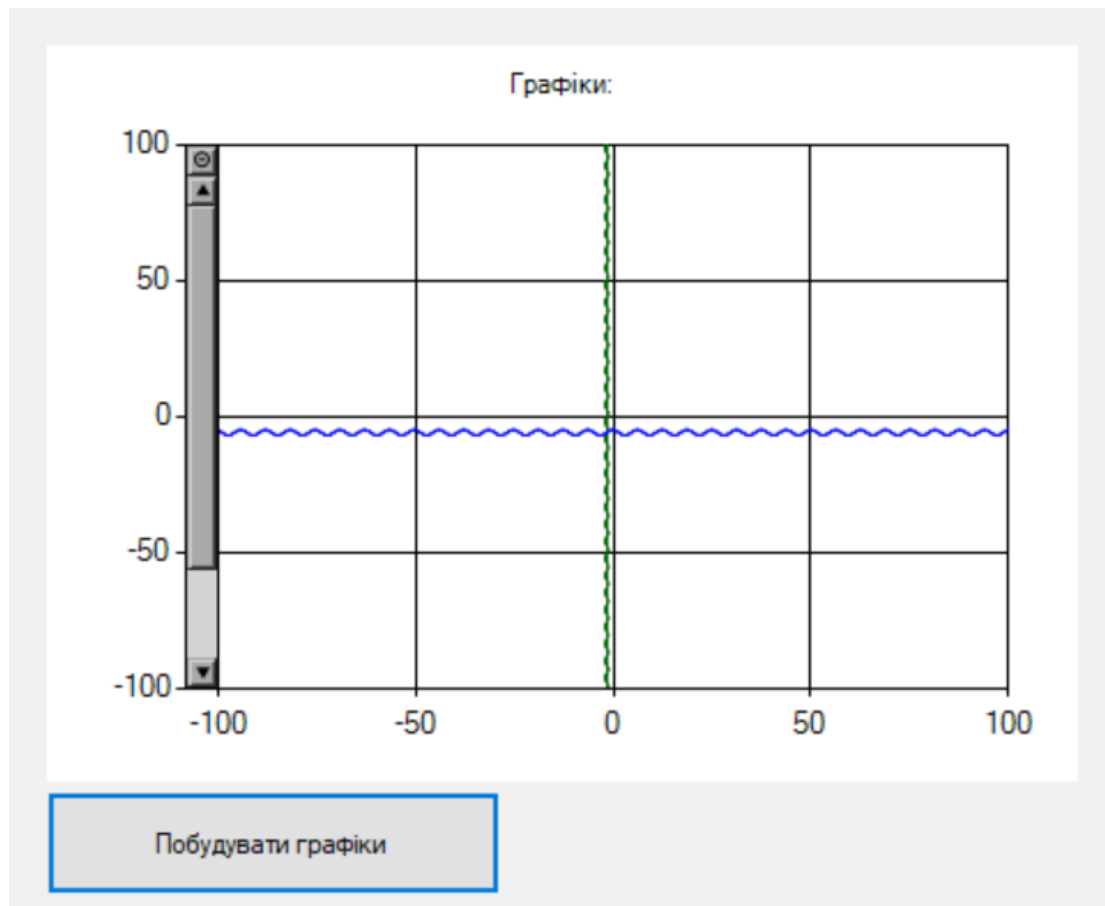


Рисунок 6.9 – Побудова графіків системи рівнянь

Далі за необхідності є можливість виділити комп'ютерною мишею на графіках певну площу задля наближення або віддалення графіків. Також є можливість зменшувати масштаб за допомогою натискання кнопок по осі x та y і промотувати графіки шляхом переміщення повзунків або натискання стрілок по осі x та y (Рисунок 6.10).

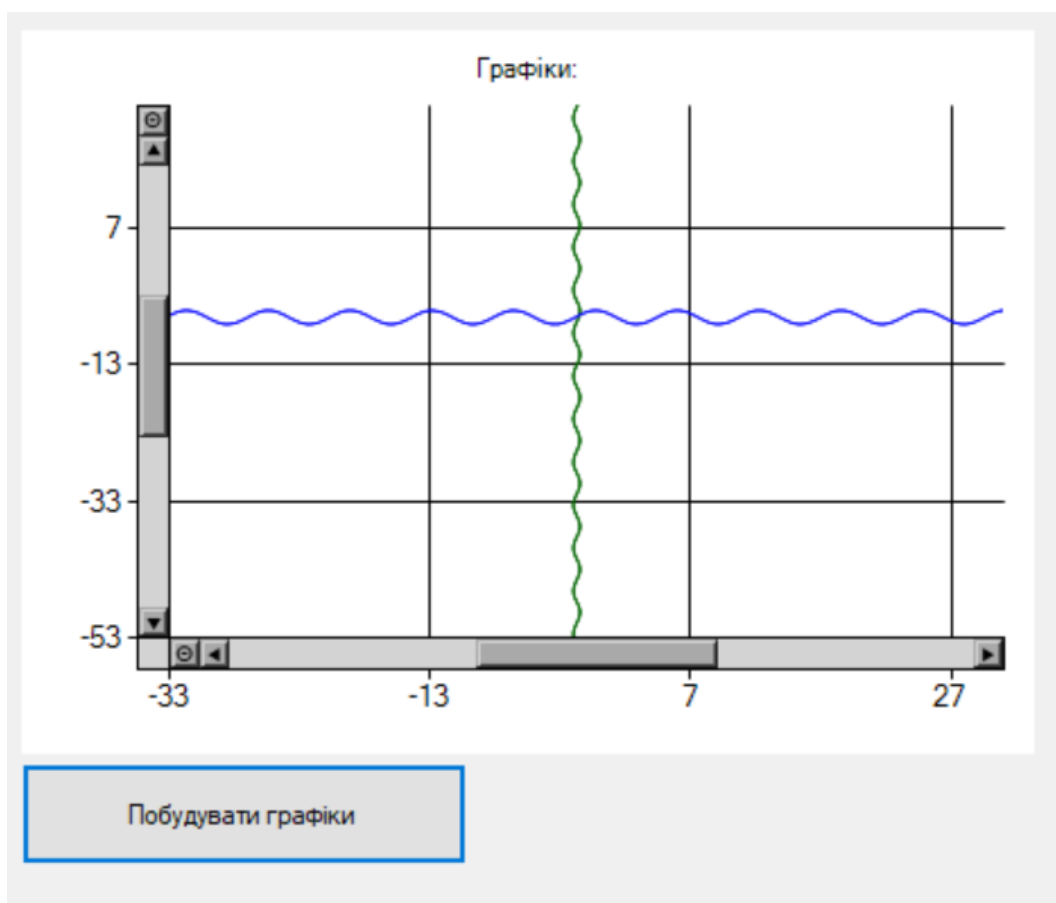


Рисунок 6.10 – Масштабування та прокрутка графіків системи рівнянь

6.2. Формат вхідних та вихідних даних

Користувачем на вхід програми подається система нелінійних рівнянь у вигляді таблиці коефіцієнтів, яка має складатися з дійсних чисел, дробова частина яких відділяється від цілої комою. Також користувачем вводяться наближені початкові значення двох невідомих та точність розрахунків, що є дійсними числами, і максимальну кількість ітерацій під час розв'язання, яка є цілим числом.

Результатом виконання програми є розв'язання даної системи нелінійних рівнянь та вивід результату на екран. Якщо система має розв'язки за обраним методом розв'язання, виводяться результати після кожної ітерації та кінцевий результат. У випадку, коли розв'язків немає, виводиться відповідне повідомлення.

6.3. Системні вимоги

Системні вимоги до програмного забезпечення наведені в таблиці 6.1.

Таблиця 6.1 – Системні вимоги програмного забезпечення

	Мінімальні	Рекомендовані
Операційна система	Windows 10 (з останніми оновленнями)	Windows 10 і вище (з останніми оновленнями)
Процесор	Intel® Pentium® III 1.0 GHz або AMD Athlon™ 1.0 GHz	Intel® Pentium® D або AMD Athlon™ 64 X2
Оперативна пам'ять	1 GB RAM	2 GB RAM
Відеоадаптер	Intel GMA 950 з відеопам'яттю об'ємом не менше 64 МБ (або сумісний аналог)	
Дисплей	800x600	1024x768 або краще
Прилади введення	Клавіатура, комп'ютерна миша	
Додаткове програмне забезпечення	Microsoft .Net Framework 4.5.2 або вище	

7 АНАЛІЗ РЕЗУЛЬТАТІВ

Головною задачею курсової роботи була реалізація програми для розв'язання системи нелінійних рівнянь наступними методами: простої ітерації (Якобі) та Гауса-Зейделя.

Критичні ситуації у роботі програми виявлені не були. Під час тестування було з'ясовано, що більшість помилок виникало під час введення користувачем некоректних даних. Тому всі дані, які вводить користувач, ретельно перевіряються на валідність і лише потім подаються на обробку програмі.

Для перевірки та доведення достовірності результатів роботи програмного забезпечення буде використано MS Excel:

а) Метод Якобі.

1) Система алгебраїчних рівнянь.

Результат виконання методу Якобі для системи алгебраїчних рівнянь наведено на рисунку 7.1:

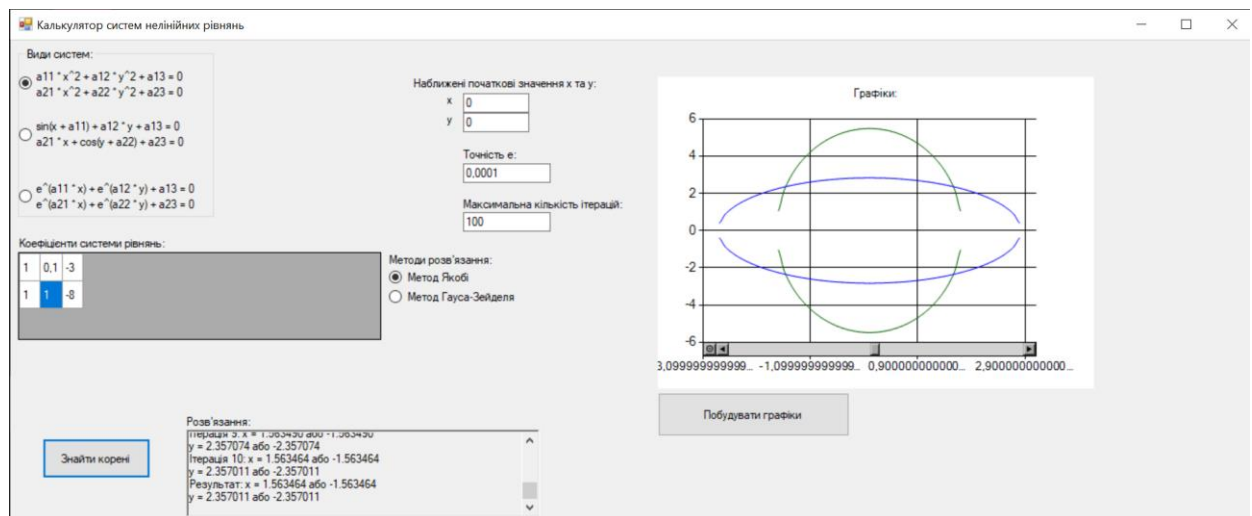


Рисунок 7.1 – Результат виконання методу Якобі для системи алгебраїчних рівнянь

Оскільки результат виконання збігається з результатом в MS Excel (Рисунок 7.2), то розроблений алгоритм методу Якобі для системи алгебраїчних рівнянь працює правильно.

	A	B	C	D	E	F	G
1	Таблица коефіцієнтів			Розв'язок			Точність
2	1	0,1	-3	-0,0000302353			0,0001
3	1	1	-8	-0,0000794666			

Рисунок 7.2 – Перевірка методу Якобі для системи алгебраїчних рівнянь в MS Excel

2) Система тригонометричних рівнянь.

Результат виконання методу Якобі для системи тригонометричних рівнянь наведено на рисунку 7.3:

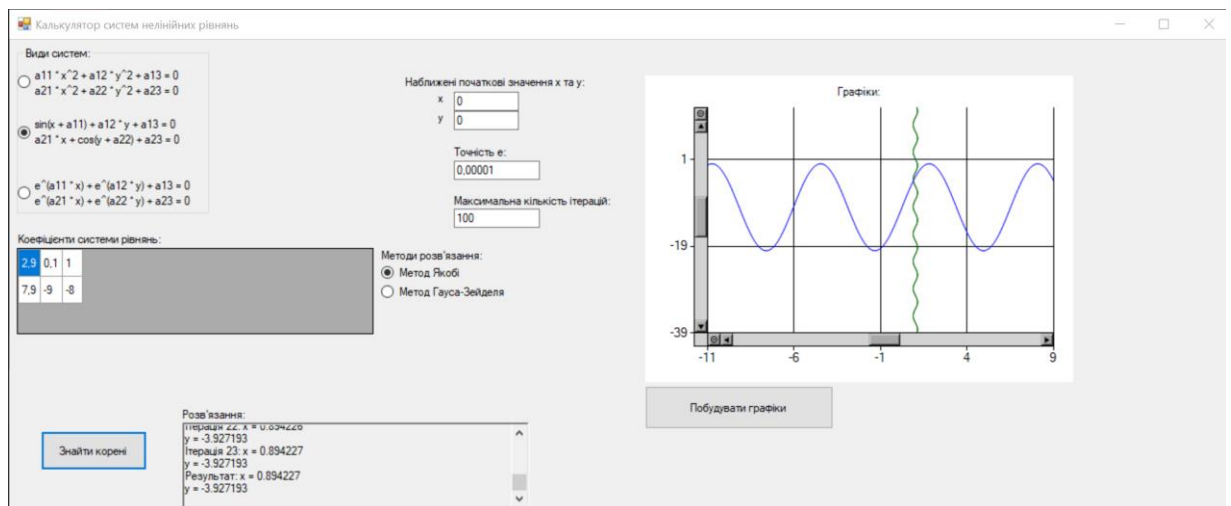
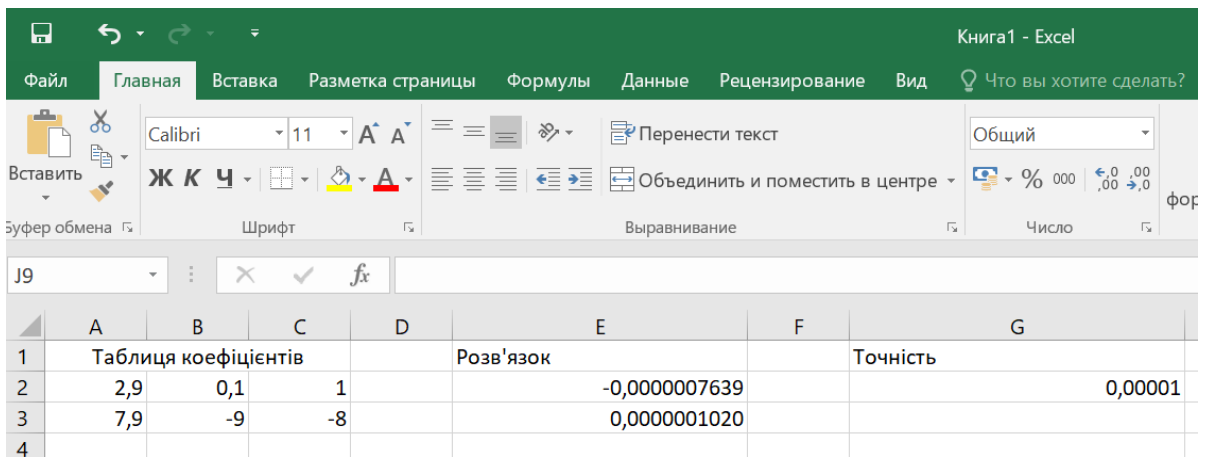


Рисунок 7.3 – Результат виконання методу Якобі для системи тригонометричних рівнянь

Оскільки результат виконання збігається з результатом в MS Excel (Рисунок 7.4), то розроблений алгоритм методу Якобі для системи тригонометричних рівнянь працює правильно.



Книга1 - Excel

Файл Главная Вставка Разметка страницы Формулы Данные Рецензирование Вид Что вы хотите сделать?

Вставить Буфер обмена Шрифт Выравнивание Число

	A	B	C	D	E	F	G
1	Таблица коефіцієнтів			Розв'язок		Точність	
2	2,9	0,1	1		-0,0000007639		0,00001
3	7,9	-9	-8		0,0000001020		
4							

Рисунок 7.4 – Перевірка методу Якобі для системи тригонометричних рівнянь в MS Excel

3) Система трансцендентних рівнянь.

Результат виконання методу Якобі для системи трансцендентних рівнянь наведено на рисунку 7.5:

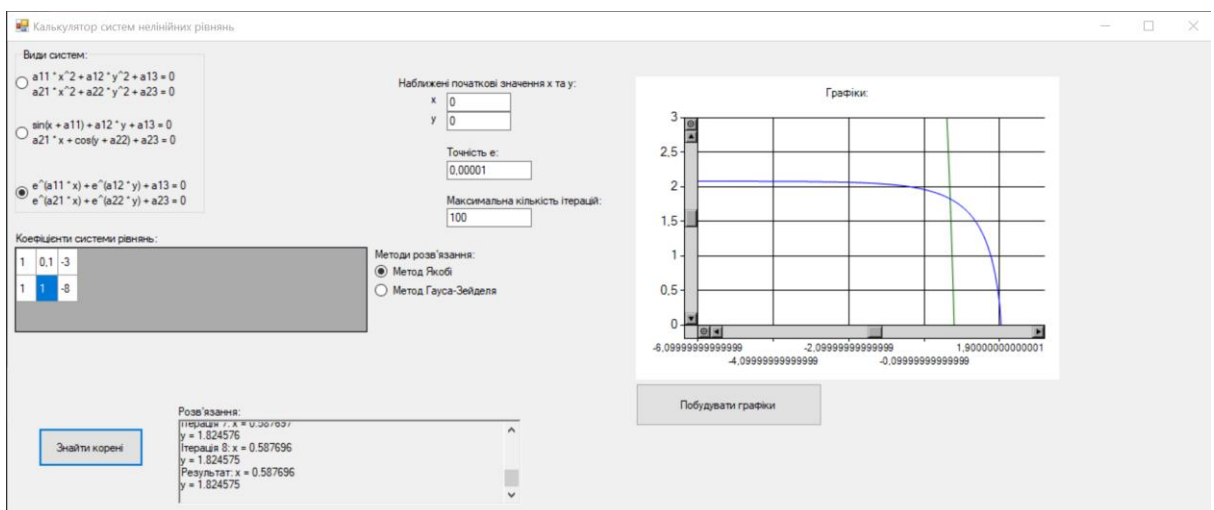


Рисунок 7.5 – Результат виконання методу Якобі для системи трансцендентних рівнянь

Оскільки результат виконання збігається з результатом в MS Excel (Рисунок 7.6), то розроблений алгоритм методу Якобі для системи трансцендентних рівнянь працює правильно.

	A	B	C	D	E	F	G
1	Таблиця коефіцієнтів			Розв'язок		Точність	
2	1	0,1	-3		-0,0000000465		0,00001
3	1	1	-8		-0,0000037981		
4							

Рисунок 7.6 – Перевірка методу Якобі для системи трансцендентних рівнянь в MS Excel

б) Метод Гауса Зейделя.

1) Система алгебраїчних рівнянь.

Результат виконання методу Гауса-Зейделя для системи алгебраїчних рівнянь наведено на рисунку 7.7:

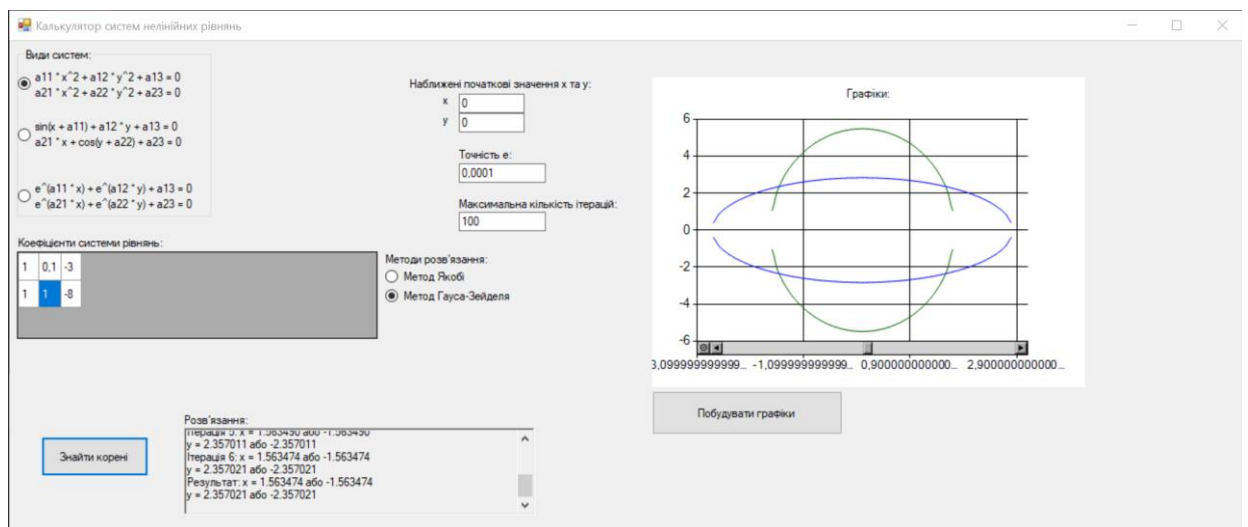
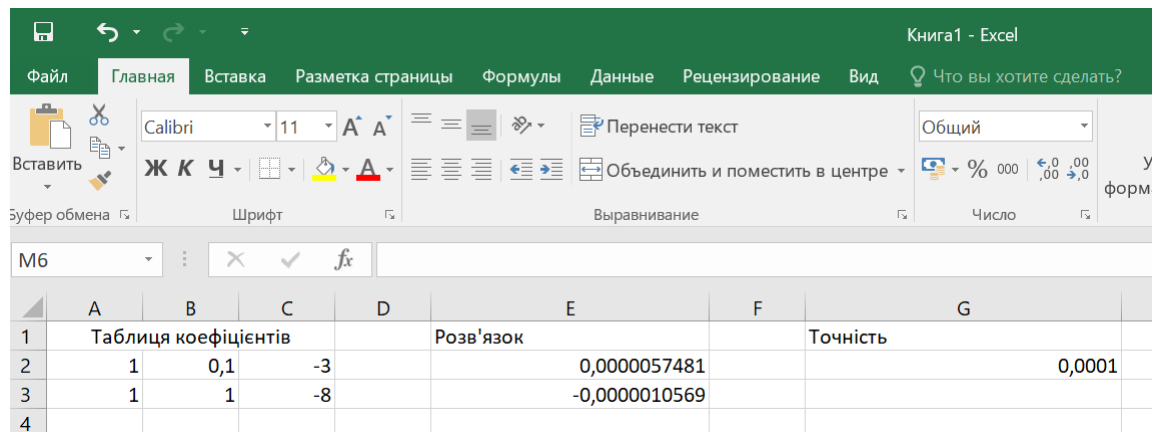


Рисунок 7.7 – Результат виконання методу Гауса-Зейделя для системи алгебраїчних рівнянь

Оскільки результат виконання збігається з результатом в MS Excel (Рисунок 7.8), то розроблений алгоритм методу Гауса-Зейделя для системи алгебраїчних рівнянь працює правильно.



Книга1 - Excel

Файл Главная Вставка Разметка страницы Формулы Данные Рецензирование Вид Что вы хотите сделать?

Вставить Шрифт Выравнивание Число

	A	B	C	D	E	F	G
1	Таблица коефіцієнтів			Розв'язок		Точність	
2	1	0,1	-3		0,0000057481		0,0001
3	1	1	-8		-0,0000010569		
4							

Рисунок 7.8 – Перевірка методу Гауса-Зейделя для системи алгебраїчних рівнянь в MS Excel

2) Система тригонометричних рівнянь.

Результат виконання методу Гауса-Зейделя для системи тригонометричних рівнянь наведено на рисунку 7.9:

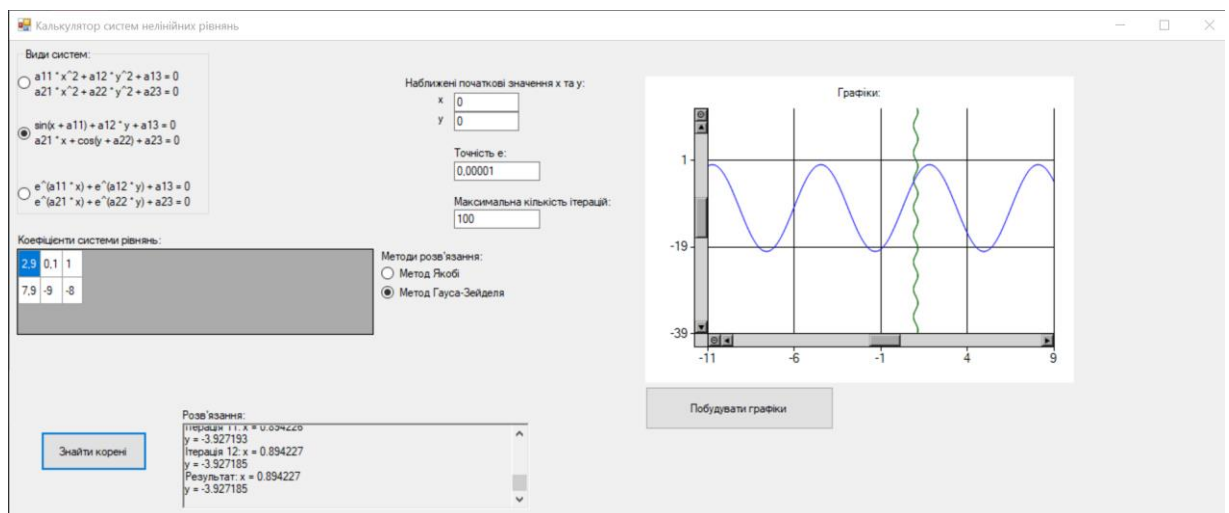
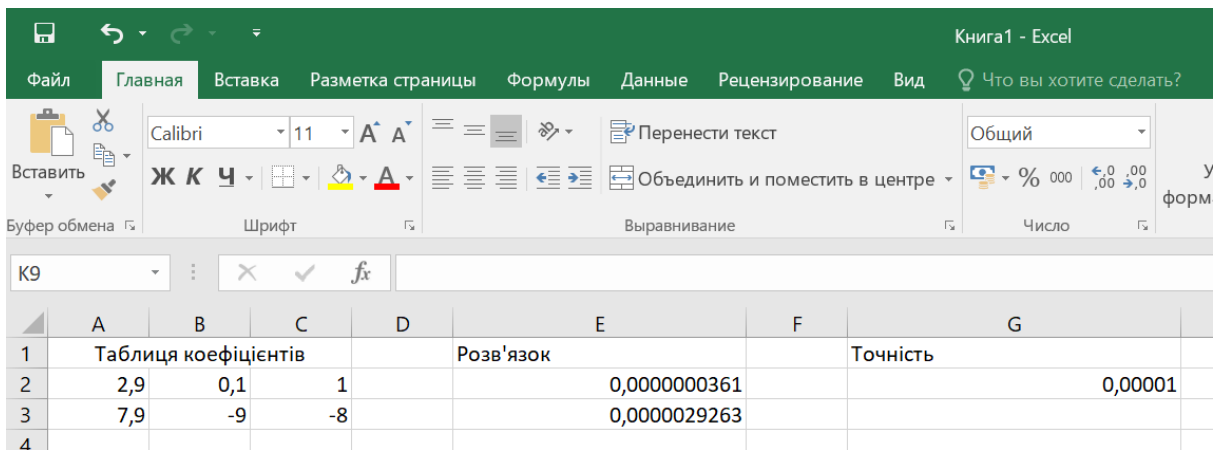


Рисунок 7.9 – Результат виконання методу Гауса-Зейделя для системи тригонометричних рівнянь

Оскільки результат виконання збігається з результатом в MS Excel (Рисунок 7.10), то розроблений алгоритм методу Гауса-Зейделя для системи тригонометричних рівнянь працює правильно.



	A	B	C	D	E	F	G
1	Таблиця коефіцієнтів			Розв'язок			Точність
2	2,9	0,1	1		0,0000000361		0,00001
3	7,9	-9	-8		0,0000029263		
4							

Рисунок 7.10 – Перевірка методу Гауса-Зейделя для системи тригонометричних рівнянь в MS Excel

3) Система трансцендентних рівнянь.

Результат виконання методу Гауса-Зейделя для системи трансцендентних рівнянь наведено на рисунку 7.11:

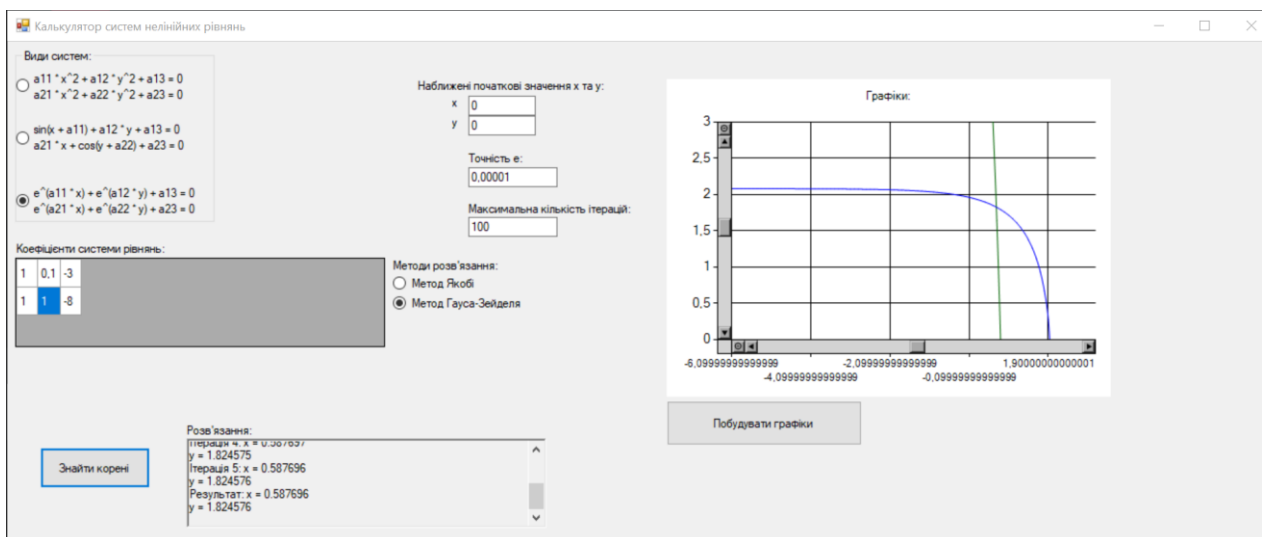


Рисунок 7.11 – Результат виконання методу Гауса-Зейделя для системи трансцендентних рівнянь

Оскільки результат виконання збігається з результатом в MS Excel (Рисунок 7.12), то розроблений алгоритм методу Гауса-Зейделя для системи трансцендентних рівнянь працює правильно.

	A	B	C	D	E	F	G
1	1	0,1	-3		0,0000000735		0,00001
2	1	1	-8		0,0000024021		
3							

Рисунок 7.12 – Перевірка методу Гауса-Зейделя для системи трансцендентних рівнянь в MS Excel

Результати тестування ефективності алгоритмів розв'язання систем нелінійних рівнянь наведено у таблицях 7.1 та 7.2. Для точності проведення аналізу алгоритми тестуються на двох різних наборах вхідних даних, причому точність є змінною величиною, зо якою буде зроблено аналіз.

Таблиця 7.1 – Тестування ефективності методів на першому наборі вхідних даних.

Точність	Параметри тестування	Метод	
		Якобі	Гауса-Зейделя
0,1	Кількість ітерацій	18	14
0,01	Кількість ітерацій	52	32
0,001	Кількість ітерацій	88	51
0,0001	Кількість ітерацій	126	69
0,00001	Кількість ітерацій	160	87
0,000001	Кількість ітерацій	195	107
0,0000001	Кількість ітерацій	213	107

Візуалізація результатів таблиці 7.1 наведена на рисунку 7.13. Для більшої наочності на графіку показана залежність кількості ітерацій від логарифму з основою 0,1 значення точності, замість використання самої точності безпосередньо. Таким чином при точності 0,01 логарифм буде дорівнювати 2, при 0,001 – 3 і т. д.

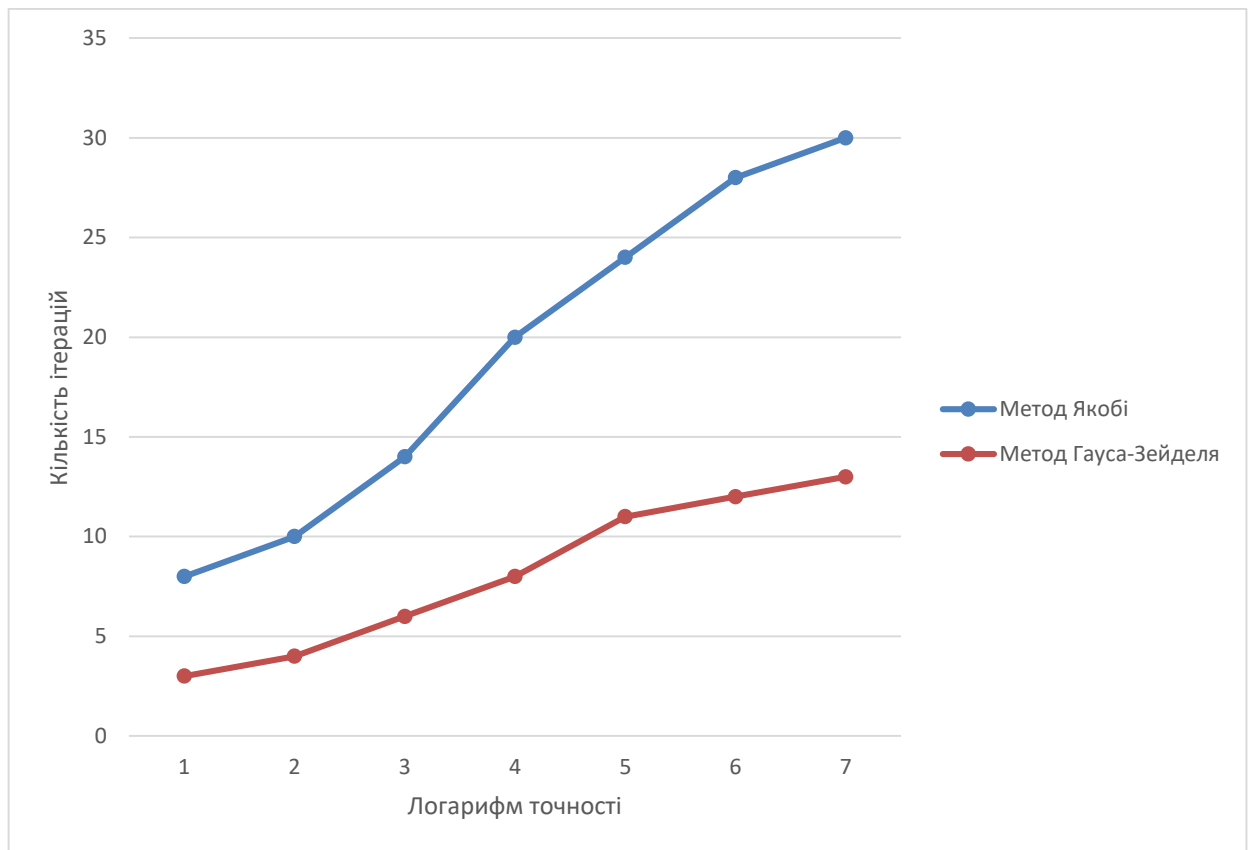


Рисунок 7.13 – Графік залежності кількості ітерацій методу від точності обчислень на першому наборі даних

Таблиця 7.2 – Тестування ефективності методів на другому наборі вхідних даних.

Точність	Параметри тестування	Метод	
		Якобі	Гауса-Зейделя
0,1	Кількість ітерацій	8	3
0,01	Кількість ітерацій	10	4
0,001	Кількість ітерацій	14	6
0,0001	Кількість ітерацій	20	8
0,00001	Кількість ітерацій	24	11
0,000001	Кількість ітерацій	28	12
0,0000001	Кількість ітерацій	30	13

Візуалізація результатів таблиці 7.2 наведена на рисунку 7.14.

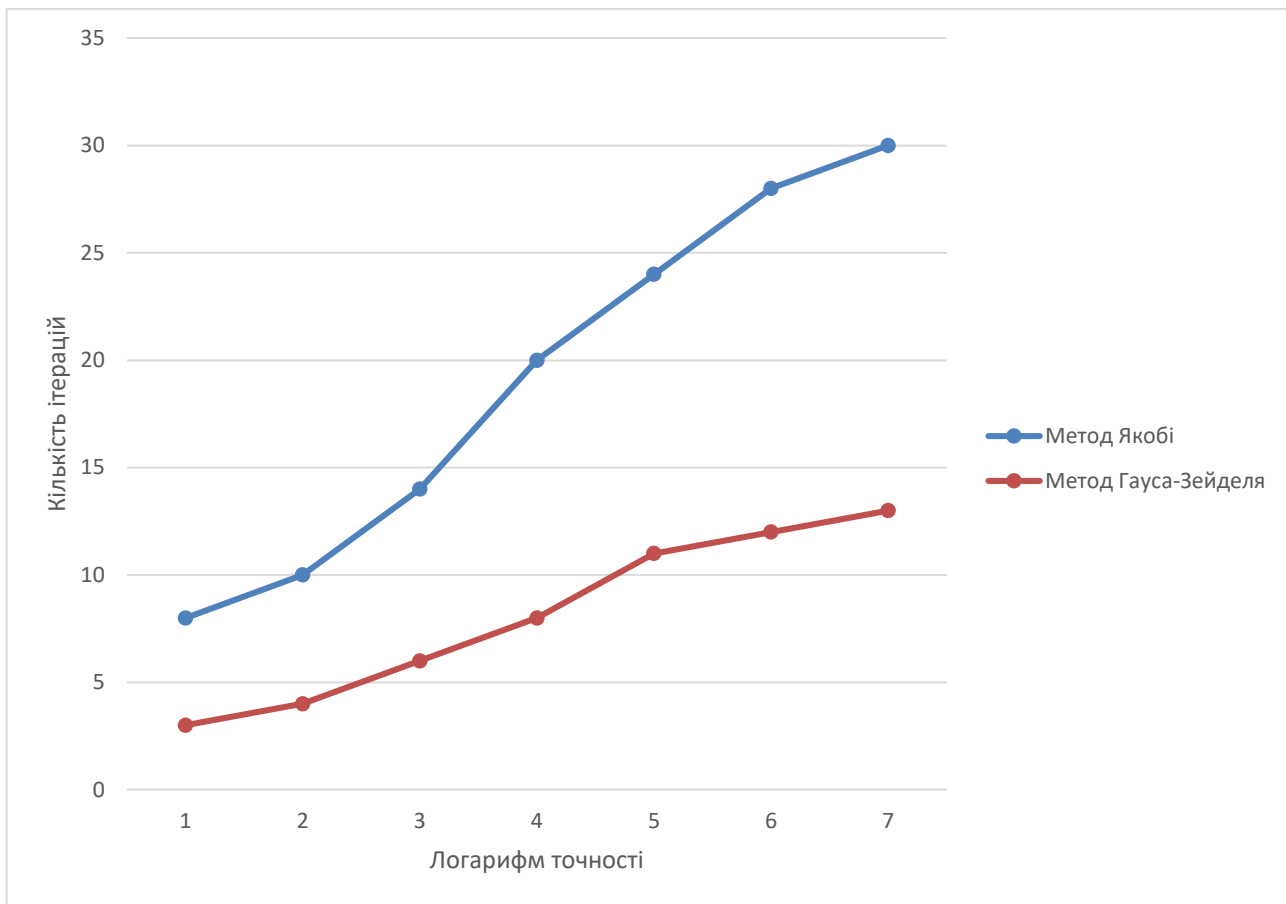


Рисунок 7.14 – Графік залежності кількості ітерацій методу від точності обчислень на другому наборі даних

За результатами тестування можна зробити такі висновки:

- а) Для розглянутих методів розв’язання системи нелінійних рівнянь кількість ітерацій сильно відрізняється. Також на це мають вплив різні набори вхідних даних. До того ж існує пряма залежність між кількістю ітерацій та обраною точністю розрахунків.
- б) Всі розглянуті алгоритми мають логарифмічну асимптотичну складність, тобто $\theta(\log n)$, де n – точність обчислення розв’язання системи.
- в) З-поміж досліджуваних методів найбільш ефективним та доречним для практичного використання є метод Гауса-Зейделя, тому що на всіх наборах вхідних даних він виконується за меншу кількість ітерацій, тобто працює швидше.

ВИСНОВКИ

Під час виконання даної курсової роботи було досліджено ітераційні методи розв'язання систем алгебраїчних, тригонометричних та трансцендентних рівнянь. Було створене програмне забезпечення з реалізацією алгоритмів досліджуваних методів з дотриманням принципів ООП мовою C++. Реалізація графічного інтерфейсу користувача була виконана за допомогою засобу C++/CLI Windows Forms. Були набуті навички застосування парадигми об'єктно-орієнтованого програмування.

У даній курсовій роботі було досліджено теоретичну базу поставленої проблеми, була побудована діаграма класів створеного програмного забезпечення, було сплановано та проведено тестування програмного застосунку, було сформовано інструкцію користувача з описом всього функціоналу програми та було проаналізовано та оцінено достовірність і ефективність роботи реалізованих алгоритмів.

Поставлена задача, що полягала у розробці програмного забезпечення, що знаходить розв'язок заданої системи нелінійних рівнянь методом Якобі або методом Гауса-Зейделя, була виконана.

ПЕРЕЛІК ПОСИЛАНЬ

1. URL (дата звернення: 20.06.2022):
<https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjStoDV-Kj4AhXjsYsKHbueD0UQFnoECCsQAQ&url=https%3A%2F%2Flearn.ztu.edu.ua%2Fmod%2Fresource%2Fview.php%3Fid%3D48793&usg=AOvVaw0FQsLVjZnHD64UWVXs203V>
2. ЧИСЕЛЬНІ МЕТОДИ // Міністерство освіти і науки України. Вінницький національний технічний університет. О. К. Колесницький, І. Р. Арсенюк, В. І. Месюра. Навчальний посібник. Вінниця ВНТУ 2017. С. 77-81. URL (дата звернення: 20.06.2022):
http://pdf.lib.vntu.edu.ua/books/IRVC/2021/Kolesnitskij_2017_130.pdf

ДОДАТОК А ТЕХНІЧНЕ ЗАВДАННЯ

КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. І. Сікорського

Кафедра
інформатики та програмної інженерії

Затвердив

Керівник Головченко Максим Миколайович

«12» квітня 2022 р.

Виконавець:

Студент Ликова Катерина Олександрівна

«12» квітня 2022 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання курсової роботи

на тему: «Розв'язання систем нелінійних рівнянь»

з дисципліни:

«Основи програмування»

1. *Мета:* Метою курсової роботи є розробка ефективного програмного забезпечення для розв'язання систем нелінійних рівнянь.
2. *Дата початку роботи:* «12» квітня 2022 р.
3. *Дата закінчення роботи:* «12» червня 2022 р.
4. *Вимоги до програмного забезпечення.*

1) Функціональні вимоги:

- Можливість задавати загальний вигляд системи нелінійних рівнянь (алгебраїчна, тригонометрична або трансцендентна)
- Можливість задавати значення коефіцієнтів рівнянь системи
- Можливість обирати метод розв'язання системи (Якобі або Гауса-Зейделя)
- Можливість перевірки на коректність введення даних та виведення відповідних повідомлень у разі помилки
- Можливість розв'язувати системи нелінійних рівнянь обраним методом
- Можливість графічного розв'язку системи нелінійних рівнянь;
- Можливість збереження розв'язання у файл
- Можливість відображення статистичних та/або аналітичних даних для подальшого аналізу ефективності алгоритму.

2) Нефункціональні вимоги:

- Можливість запуску програмного забезпечення на комп'ютерах із операційною системою Windows 10 і вище;
- Все програмне забезпечення та супроводжуюча технічна документація повинні задовольняти наступним ДЕСТам:

ГОСТ 29.401 - 78 - Текст програми. Вимоги до змісту та оформлення.

ГОСТ 19.106 - 78 - Вимоги до програмної документації.

ГОСТ 7.1 - 84 та ДСТУ 3008 - 2015 - Розробка технічної документації.

5. *Стадії та етапи розробки:*

- 1) Об'єктно-орієнтований аналіз предметної області задачі (до 04.05.2022 р.)

- 2) Об'єктно-орієнтоване проектування архітектури програмної системи (до 05.05.2022 р.)
- 3) Розробка програмного забезпечення (до 06.05.2022 р.)
- 4) Тестування розробленої програми (до 26.05.2022 р.)
- 5) Розробка пояснювальної записки (до 05.06.2022 р.).
- 6) Захист курсової роботи (до 19.06.2022 р.).

6. Порядок контролю та приймання. Поточні результати роботи над КР регулярно демонструються викладачу. Своєчасність виконання основних етапів графіку підготовки роботи впливає на оцінку за КР відповідно до критеріїв оцінювання.

ДОДАТОК Б ТЕКСТИ ПРОГРАМНОГО КОДУ

*Тексти програмного коду програмного забезпечення
вирішення задачі розв'язання систем нелінійних рівнянь*

(Найменування програми (документа))

Електронний носій

(Вид носія даних)

42 арк, 275 Кб

(Обсяг програми (документа), арк., Кб)

студентки групи ІП-15 І курсу

Ликової К. О.

Файл «algebraic_equations.cpp»

```
#include "pch.h"
```

```
#include "algebraic_equations.h"
```

```
Alg_eq::Alg_eq(vector<vector<double>> table, double x, double y, double e, int  
num) :Base_eq(table, x, y, e, num) {};
```

```
string Alg_eq::solution_Yacobi()
```

```
{
```

```
    return Base_eq::solution_Yacobi();
```

```
}
```

```
string Alg_eq::solution_Gaus_Zeydel()
```

```
{
```

```
    return Base_eq::solution_Gaus_Zeydel();
```

```
}
```

```
string Alg_eq::find_x(double y)
```

```
{
```

```
    string res = "";
```

```
    if (((-param[0][1] * pow(y, 2) - param[0][2]) / param[0][0]) < 0 || param[0][0]  
== 0)
```

```
{
```

```
        res = "Немає розв'язків";
```

```
}
```

```
else
```

```
{
```

```
        res = to_string(pow((( -param[0][1] * pow(y, 2) - param[0][2]) /  
param[0][0]), 0.5));
```

```
}
```

```
    return res;
```

```
}
```

```
string Alg_eq::find_y(double x)
```

```
{
```

```
    string res = "";
```

```

        if (((-param[1][0] * pow(x, 2) - param[1][2]) / param[1][1]) < 0 || param[1][1]
== 0)
        {
            res = "Немає розв'язків";
        }
        else
        {
            res = to_string(pow((-param[1][0] * pow(x, 2) - param[1][2]) /
param[1][1], 0.5));
        }
        return res;
    }
string Alg_eq::it_print(double x, double y, int num)
{
    string res = "";
    res += "Ітерація " + to_string(num) + ": x = " + to_string(x) + " або -" +
to_string(x) + "\n";
    res += "y = " + to_string(y) + " або -" + to_string(y) + "\n";
    return res;
}
string Alg_eq::res_print(double x, double y)
{
    string res = "";
    res += "Результат: x = " + to_string(x) + " або -" + to_string(x) + "\n";
    res += "y = " + to_string(y) + " або -" + to_string(y) + "\n";
    return res;
}

```

Файл «algebraic_equations.h»

```
#include "base_equations.h"
```

```
/// <summary>
```

```
/// Клас для заданої системи нелінійних алгебраїчних рівнянь.
```

```

/// </summary>
class Alg_eq: private Base_eq
{
public:
    Alg_eq(vector<vector<double>>, double, double, double, int);
    string find_x(double);
    string find_y(double);
    string solution_Yacobi();
    string solution_Gaus_Zeydel();
    string it_print(double, double, int);
    string res_print(double, double);
};
Файл «AssemblyInfo.cpp»
#include "pch.h"

using namespace System;
using namespace System::Reflection;
using namespace System::Runtime::CompilerServices;
using namespace System::Runtime::InteropServices;
using namespace System::Security::Permissions;

[assembly:AssemblyTitleAttribute(L"CppCLRWinformsProjekt")];
[assembly:AssemblyDescriptionAttribute(L"")];
[assembly:AssemblyConfigurationAttribute(L"")];
[assembly:AssemblyCompanyAttribute(L"")];
[assembly:AssemblyProductAttribute(L"CppCLRWinformsProjekt")];
[assembly:AssemblyCopyrightAttribute(L"Copyright (c) 2019")];
[assembly:AssemblyTrademarkAttribute(L"")];
[assembly:AssemblyCultureAttribute(L"")];

[assembly:AssemblyVersionAttribute("1.0.*")];

```

```
[assembly:ComVisible(false)];
```

```
[assembly:CLSCompliantAttribute(true)];
```

```
Файл «base_equations.cpp»
```

```
#include "pch.h"
```

```
#include "base_equations.h"
```

```
Base_eq::Base_eq(vector<vector<double>>table, double x, double y, double e, int  
num)
```

```
{
```

```
    param.resize(2, vector<double>(3));
```

```
    for (int i = 0; i < 2; i++)
```

```
    {
```

```
        for (int j = 0; j < 3; j++)
```

```
        {
```

```
            param[i][j] = table[i][j];
```

```
        }
```

```
    }
```

```
    x0 = x;
```

```
    y0 = y;
```

```
    eps = e;
```

```
    max = num;
```

```
}
```

```
string Base_eq::solution_Yacobi()
```

```
{
```

```
    double new_x = x0;
```

```
    double new_y = y0;
```

```
    double old_x;
```

```
    double old_y;
```

```
    double dif_x = eps + 1;
```

```
    double dif_y = eps + 1;
```

```

int num = 0;
string res = "";
while (dif_x > eps || dif_y > eps)
{
    num++;
    if (num > max)
    {
        res = "Кількість ітерацій перевищила максимальну. Немає
розв'язків";
        return res;
    }
    old_x = new_x;
    old_y = new_y;
    if (find_x(old_y) == "Немає розв'язків")
    {
        res = find_x(old_y);
        return res;
    }
    new_x = stod(find_x(old_y));
    if (find_y(old_x) == "Немає розв'язків")
    {
        res = find_y(old_x);
        return res;
    }
    new_y = stod(find_y(old_x));
    dif_x = abs(new_x - old_x);
    dif_y = abs(new_y - old_y);
    res += it_print(new_x, new_y, num);
}
res += res_print(new_x, new_y);
return res;

```

```

}
string Base_eq::solution_Gaus_Zeydel()
{
    double new_x = x0;
    double new_y = y0;
    double old_x;
    double old_y;
    double dif_x = 1;
    double dif_y = 1;
    int num = 0;
    string res = "";
    while (dif_x > eps || dif_y > eps)
    {
        num++;
        if (num > max)
        {
            res = "Кількість ітерацій перевищила максимальну. Немає
розв'язків";
            return res;
        }
        old_x = new_x;
        old_y = new_y;
        if (find_x(old_y) == "Немає розв'язків")
        {
            res = find_x(old_y);
            return res;
        }
        new_x = stod(find_x(old_y));
        if (find_y(new_x) == "Немає розв'язків")
        {
            res = find_y(new_x);

```

```

        return res;
    }
    new_y = stod(find_y(new_x));
    dif_x = abs(new_x - old_x);
    dif_y = abs(new_y - old_y);
    res += it_print(new_x, new_y, num);
}
res += res_print(new_x, new_y);
return res;
}
Файл «base_equations.h»
#pragma once
#include <vector>
#include <math.h>
#include <string>
using namespace std;
/// <summary>
/// Базовий клас для трьох заданих систем нелінійних рівнянь.
/// </summary>
class Base_eq
{
public:
    /// <summary>
    /// Задає атрибути класу.
    /// </summary>
    Base_eq(vector<vector<double>>, double, double, double, int);
    /// <summary>
    /// Рахує значення першої невідомої.
    /// </summary>
    virtual string find_x(double) = 0;
    /// <summary>

```



```

/// Рахує значення другої невідомої.
/// </summary>
virtual string find_y(double) = 0;
/// <summary>
/// Розв'язання системи нелінійних рівнянь методом Якобі.
/// </summary>
string solution_Yacobi();
/// <summary>
/// Розв'язання системи нелінійних рівнянь методом Гауса-Зейделя.
/// </summary>
string solution_Gaus_Zeydel();
/// <summary>
/// Формує рядок для виведення результатів розрахунків за одну ітерацію.
/// </summary>
virtual string it_print(double, double, int) = 0;
/// <summary>
/// Формує рядок для виведення результатів розрахунків вцілому.
/// </summary>
virtual string res_print(double, double) = 0;

```

protected:

```

/// <summary>
/// Таблиця значень коефіцієнтів системи нелінійних рівнянь.
/// </summary>
vector<vector<double>>> param;
/// <summary>
/// Початкове значення першої похідної.
/// </summary>
double x0;
/// <summary>
/// Початкове значення другої похідної.
/// </summary>

```

```

double y0;
/// <summary>
/// Значення точності розрахунків.
/// </summary>
double eps;
/// <summary>
/// Значення максимальної кількості ітерацій.
/// </summary>
int max;
};
Файл «CppCLR_WinformsProjekt.cpp»
#include "pch.h"

using namespace System;

//int main(array<System::String ^> ^args)
//{
//    return 0;
//}

#include "Form1.h"
using namespace System::Windows::Forms;

[STAThread]
int main() {
    Application::EnableVisualStyles();
    Application::SetCompatibleTextRenderingDefault(false);
    Application::Run(gcnew CppCLRWinformsProjekt::Form1());
    return 0;
}
Файл «Form1.cpp»

```

```

#include "pch.h"
//#include "Form1.h"
Файл «Form1.h»
#pragma once
#include "algebraic_equations.h"
#include "trigonometric_equations.h"
#include "transcendental_equations.h"
#include "func.h"
namespace CppCLRWinformsProjekt {

    using namespace System;
    using namespace System::ComponentModel;
    using namespace System::Collections;
    using namespace System::Windows::Forms;
    using namespace System::Data;
    using namespace System::Drawing;

    /// <summary>
    /// Клас для відображення форми, що містить засоби для розв'язання систем
    нелінійних рівнянь.
    /// </summary>
    public ref class Form1 : public System::Windows::Forms::Form
    {
    public:
        /// <summary>
        /// Створює форму Form1 та ініціалізує її компоненти.
        /// </summary>
        Form1(void)
        {
            InitializeComponent();
        }
    }
}

```

protected:

```
/// <summary>
```

```
/// Очищує використані формою Form1 компоненти.
```

```
/// </summary>
```

```
~Form1()
```

```
{
```

```
    if (components)
```

```
    {
```

```
        delete components;
```

```
    }
```

```
}
```

```
private: System::Windows::Forms::GroupBox^ groupBox1;
```

protected:

```
private: System::Windows::Forms::RadioButton^ radioButton3;
```

```
private: System::Windows::Forms::RadioButton^ radioButton2;
```

```
private: System::Windows::Forms::RadioButton^ radioButton1;
```

```
private: System::Windows::Forms::TextBox^ textBox6;
```

```
private: System::Windows::Forms::TextBox^ textBox5;
```

```
private: System::Windows::Forms::TextBox^ textBox4;
```

```
private: System::Windows::Forms::TextBox^ textBox3;
```

```
private: System::Windows::Forms::TextBox^ textBox2;
```

```
private: System::Windows::Forms::TextBox^ textBox1;
```

```
private: System::Windows::Forms::DataGridView^ table;
```

```
private: System::Windows::Forms::Button^ button1;
```

```
private: System::Windows::Forms::RichTextBox^ richTextBox1;
```

```
private: System::Windows::Forms::TextBox^ textBox7;
```

```
private: System::Windows::Forms::DataVisualization::Charting::Chart^ chart1;
```

```

private: System::Windows::Forms::Button^ button2;
private: System::Windows::Forms::TextBox^ textBox9;
private: System::Windows::Forms::TextBox^ textBox10;
private: System::Windows::Forms::TextBox^ textBox11;
private: System::Windows::Forms::TextBox^ textBox12;
private: System::Windows::Forms::TextBox^ textBox13;
private: System::Windows::Forms::TextBox^ textBox14;
private: System::Windows::Forms::TextBox^ textBox15;
private: System::Windows::Forms::RadioButton^ radioButton4;
private: System::Windows::Forms::RadioButton^ radioButton5;
private: System::Windows::Forms::TextBox^ textBox16;
private: System::Windows::Forms::TextBox^ textBox17;
private: System::Windows::Forms::TextBox^ textBox18;
private: System::Windows::Forms::TextBox^ textBox19;

```

```
private:
```

```
    /// <summary>
```

```
    /// Обов'язкова змінна конструктора.
```

```
    /// </summary>
```

```
    System::ComponentModel::Container ^components;
```

```
#pragma region Windows Form Designer generated code
```

```
    /// <summary>
```

```
    /// Необхідний метод для підтримки дизайнера.
```

```
    /// Вміст методу не можна змінювати за допомогою редактора коду.
```

```
    /// </summary>
```

```
    void InitializeComponent(void)
```

```
    {
```

```
        System::Windows::Forms::DataVisualization::Charting::ChartArea^
```

```

chartArea1                                =                                (gcnew
System::Windows::Forms::DataVisualization::Charting::ChartArea());

        System::Windows::Forms::DataVisualization::Charting::Series^
series1 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());
        System::Windows::Forms::DataVisualization::Charting::Series^
series2 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());
        System::Windows::Forms::DataVisualization::Charting::Series^
series3 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());
        System::Windows::Forms::DataVisualization::Charting::Series^
series4 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());
        System::Windows::Forms::DataVisualization::Charting::Title^
title1 = (gcnew System::Windows::Forms::DataVisualization::Charting::Title());

        this->groupBox1                                =                                (gcnew
System::Windows::Forms::GroupBox());

        this->radioButton3                                =                                (gcnew
System::Windows::Forms::RadioButton());

        this->radioButton2                                =                                (gcnew
System::Windows::Forms::RadioButton());

        this->radioButton1                                =                                (gcnew
System::Windows::Forms::RadioButton());

        this->textBox6 = (gcnew System::Windows::Forms::TextBox());
        this->textBox5 = (gcnew System::Windows::Forms::TextBox());
        this->textBox4 = (gcnew System::Windows::Forms::TextBox());
        this->textBox3 = (gcnew System::Windows::Forms::TextBox());
        this->textBox2 = (gcnew System::Windows::Forms::TextBox());
        this->textBox1 = (gcnew System::Windows::Forms::TextBox());
        this->table = (gcnew System::Windows::Forms::DataGridView());
        this->button1 = (gcnew System::Windows::Forms::Button());
        this->richTextBox1                                =                                (gcnew
System::Windows::Forms::RichTextBox());

        this->textBox7 = (gcnew System::Windows::Forms::TextBox());

```

```

        this->chart1                                =                                (gcnew
System::Windows::Forms::DataVisualization::Charting::Chart());

        this->button2 = (gcnew System::Windows::Forms::Button());
        this->textBox9 = (gcnew System::Windows::Forms::TextBox());
        this->textBox10 = (gcnew System::Windows::Forms::TextBox());
        this->textBox11 = (gcnew System::Windows::Forms::TextBox());
        this->textBox12 = (gcnew System::Windows::Forms::TextBox());
        this->textBox13 = (gcnew System::Windows::Forms::TextBox());
        this->textBox14 = (gcnew System::Windows::Forms::TextBox());
        this->textBox15 = (gcnew System::Windows::Forms::TextBox());
        this->radioButton4                                =                                (gcnew
System::Windows::Forms::RadioButton());

        this->radioButton5                                =                                (gcnew
System::Windows::Forms::RadioButton());

        this->textBox16 = (gcnew System::Windows::Forms::TextBox());
        this->textBox17 = (gcnew System::Windows::Forms::TextBox());
        this->textBox18 = (gcnew System::Windows::Forms::TextBox());
        this->textBox19 = (gcnew System::Windows::Forms::TextBox());
        this->groupBox1->SuspendLayout();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this->table))-
>BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this->chart1))-
>BeginInit();

        this->SuspendLayout();
        //
        // groupBox1
        //
        this->groupBox1->Controls->Add(this->radioButton3);
        this->groupBox1->Controls->Add(this->radioButton2);

```

```

this->groupBox1->Controls->Add(this->radioButton1);
this->groupBox1->Controls->Add(this->textBox6);
this->groupBox1->Controls->Add(this->textBox5);
this->groupBox1->Controls->Add(this->textBox4);
this->groupBox1->Controls->Add(this->textBox3);
this->groupBox1->Controls->Add(this->textBox2);
this->groupBox1->Controls->Add(this->textBox1);
this->groupBox1->Location = System::Drawing::Point(12, 12);
this->groupBox1->Name = L"groupBox1";
this->groupBox1->Size = System::Drawing::Size(301, 270);
this->groupBox1->TabIndex = 0;
this->groupBox1->TabStop = false;
this->groupBox1->Text = L"Види систем:";
//
// radioButton3
//
this->radioButton3->AutoSize = true;
this->radioButton3->Location = System::Drawing::Point(2, 225);
this->radioButton3->Name = L"radioButton3";
this->radioButton3->Size = System::Drawing::Size(21, 20);
this->radioButton3->TabIndex = 8;
this->radioButton3->TabStop = true;
this->radioButton3->UseVisualStyleBackColor = true;
//
// radioButton2
//
this->radioButton2->AutoSize = true;
this->radioButton2->Location = System::Drawing::Point(2, 128);
this->radioButton2->Name = L"radioButton2";
this->radioButton2->Size = System::Drawing::Size(21, 20);
this->radioButton2->TabIndex = 7;

```



```

this->radioButton2->TabStop = true;
this->radioButton2->UseVisualStyleBackColor = true;
//
// radioButton1
//
this->radioButton1->AutoSize = true;
this->radioButton1->Location = System::Drawing::Point(2, 46);
this->radioButton1->Name = L"radioButton1";
this->radioButton1->Size = System::Drawing::Size(21, 20);
this->radioButton1->TabIndex = 6;
this->radioButton1->TabStop = true;
this->radioButton1->UseVisualStyleBackColor = true;
this->radioButton1->Click += gcnew System::EventHandler(this,
&Form1::radioButton1_Click);
//
// textBox6
//
this->textBox6->BackColor =
System::Drawing::SystemColors::Control;
this->textBox6->BorderStyle =
System::Windows::Forms::BorderStyle::None;
this->textBox6->Location = System::Drawing::Point(29, 237);
this->textBox6->Name = L"textBox6";
this->textBox6->ReadOnly = true;
this->textBox6->Size = System::Drawing::Size(247, 19);
this->textBox6->TabIndex = 5;
this->textBox6->Text = L" $e^{a_{21} * x} + e^{a_{22} * y} + a_{23} = 0$ ";
//
// textBox5
//

```

```

        this->textBox5->BackColor =
System::Drawing::SystemColors::Control;

        this->textBox5->BorderStyle =
System::Windows::Forms::BorderStyle::None;

        this->textBox5->Location = System::Drawing::Point(29, 212);
        this->textBox5->Name = L"textBox5";
        this->textBox5->ReadOnly = true;
        this->textBox5->Size = System::Drawing::Size(247, 19);
        this->textBox5->TabIndex = 4;
        this->textBox5->Text = L" $e^{(a_{11} * x) + e^{(a_{12} * y) + a_{13} = 0}}$ ";
        //
        // textBox4
        //
        this->textBox4->BackColor =
System::Drawing::SystemColors::Control;

        this->textBox4->BorderStyle =
System::Windows::Forms::BorderStyle::None;

        this->textBox4->Location = System::Drawing::Point(29, 139);
        this->textBox4->Name = L"textBox4";
        this->textBox4->ReadOnly = true;
        this->textBox4->Size = System::Drawing::Size(229, 19);
        this->textBox4->TabIndex = 3;
        this->textBox4->Text = L" $a_{21} * x + \cos(y + a_{22}) + a_{23} = 0$ ";
        //
        // textBox3
        //
        this->textBox3->BackColor =
System::Drawing::SystemColors::Control;

        this->textBox3->BorderStyle =
System::Windows::Forms::BorderStyle::None;

        this->textBox3->Location = System::Drawing::Point(29, 114);

```

```

this->textBox3->Name = L"textBox3";
this->textBox3->ReadOnly = true;
this->textBox3->Size = System::Drawing::Size(229, 19);
this->textBox3->TabIndex = 2;
this->textBox3->Text = L"sin(x + a11) + a12 * y + a13 = 0";
//
// textBox2
//
this->textBox2->BackColor =
System::Drawing::SystemColors::Control;
this->textBox2->BorderStyle =
System::Windows::Forms::BorderStyle::None;
this->textBox2->Location = System::Drawing::Point(29, 60);
this->textBox2->Name = L"textBox2";
this->textBox2->ReadOnly = true;
this->textBox2->Size = System::Drawing::Size(229, 19);
this->textBox2->TabIndex = 1;
this->textBox2->Text = L"a21 * x^2 + a22 * y^2 + a23 = 0";
//
// textBox1
//
this->textBox1->BackColor =
System::Drawing::SystemColors::Control;
this->textBox1->BorderStyle =
System::Windows::Forms::BorderStyle::None;
this->textBox1->Location = System::Drawing::Point(29, 35);
this->textBox1->Name = L"textBox1";
this->textBox1->ReadOnly = true;
this->textBox1->Size = System::Drawing::Size(229, 19);
this->textBox1->TabIndex = 0;
this->textBox1->Text = L"a11 * x^2 + a12 * y^2 + a13 = 0";

```

```

//
// table
//

this->table->AllowUserToAddRows = false;
this->table->AllowUserToDeleteRows = false;
this->table->AllowUserToResizeColumns = false;
this->table->AllowUserToResizeRows = false;
this->table->ColumnHeadersHeightSizeMode =
System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->table->ColumnHeadersVisible = false;
this->table->EditMode =
System::Windows::Forms::DataGridViewEditMode::EditOnKeystroke;

this->table->Location = System::Drawing::Point(12, 336);
this->table->MultiSelect = false;
this->table->Name = L"table";
this->table->RowHeadersVisible = false;
this->table->RowHeadersWidth = 62;
this->table->RowTemplate->Height = 28;
this->table->Size = System::Drawing::Size(559, 139);
this->table->TabIndex = 1;
this->table->CellContentClick += gcnew
System::Windows::Forms::DataGridViewCellEventHandler(this,
&Form1::dataGridView1_CellContentClick);

this->table->CellValueChanged += gcnew
System::Windows::Forms::DataGridViewCellEventHandler(this,
&Form1::table_CellValueChanged);

this->table->ColumnAdded += gcnew
System::Windows::Forms::DataGridViewColumnEventHandler(this,
&Form1::table_ColumnAdded);

```

```

        this->table->Click += gcnew System::EventHandler(this,
&Form1::table_Click);

        //
        // button1
        //
        this->button1->Location = System::Drawing::Point(50, 630);
        this->button1->Name = L"button1";
        this->button1->Size = System::Drawing::Size(166, 63);
        this->button1->TabIndex = 2;
        this->button1->Text = L"Знайти корені";
        this->button1->UseVisualStyleBackColor = true;
        this->button1->Click += gcnew System::EventHandler(this,
&Form1::button1_Click);

        //
        // richTextBox1
        //
        this->richTextBox1->Location = System::Drawing::Point(272,
616);

        this->richTextBox1->Name = L"richTextBox1";
        this->richTextBox1->ReadOnly = true;
        this->richTextBox1->Size = System::Drawing::Size(542, 136);
        this->richTextBox1->TabIndex = 3;
        this->richTextBox1->Text = L"";

        //
        // textBox7
        //
        this->textBox7->BackColor =
System::Drawing::SystemColors::Control;
        this->textBox7->BorderStyle =
System::Windows::Forms::BorderStyle::None;
        this->textBox7->Location = System::Drawing::Point(272, 593);

```

```

this->textBox7->Name = L"textBox7";
this->textBox7->ReadOnly = true;
this->textBox7->Size = System::Drawing::Size(158, 19);
this->textBox7->TabIndex = 4;
this->textBox7->Text = L"Розв'язання:";
//
// chart1
//
chartArea1->Name = L"ChartArea1";
this->chart1->ChartAreas->Add(chartArea1);
this->chart1->Location = System::Drawing::Point(998, 58);
this->chart1->Name = L"chart1";
series1->ChartArea = L"ChartArea1";
series1->ChartType =
System::Windows::Forms::DataVisualization::Charting::SeriesChartType::Spline;
series1->Color = System::Drawing::Color::DarkGreen;
series1->Name = L"Series1";
series2->ChartArea = L"ChartArea1";
series2->ChartType =
System::Windows::Forms::DataVisualization::Charting::SeriesChartType::Spline;
series2->Color = System::Drawing::Color::DarkGreen;
series2->Name = L"Series2";
series3->ChartArea = L"ChartArea1";
series3->ChartType =
System::Windows::Forms::DataVisualization::Charting::SeriesChartType::Spline;
series3->Color = System::Drawing::Color::Blue;
series3->Name = L"Series3";
series4->ChartArea = L"ChartArea1";
series4->ChartType =
System::Windows::Forms::DataVisualization::Charting::SeriesChartType::Spline;
series4->Color = System::Drawing::Color::Blue;

```

```

series4->Name = L"Series4";
this->chart1->Series->Add(series1);
this->chart1->Series->Add(series2);
this->chart1->Series->Add(series3);
this->chart1->Series->Add(series4);
this->chart1->Size = System::Drawing::Size(672, 493);
this->chart1->TabIndex = 5;
this->chart1->Text = L"chart1";
title1->Name = L"title";
title1->Text = L"Графіки:";
this->chart1->Titles->Add(title1);
//
// button2
//
this->button2->Location = System::Drawing::Point(998, 557);
this->button2->Name = L"button2";
this->button2->Size = System::Drawing::Size(295, 69);
this->button2->TabIndex = 7;
this->button2->Text = L"Побудувати графіки";
this->button2->UseVisualStyleBackColor = true;
this->button2->Click += gcnew System::EventHandler(this,
&Form1::button2_Click);
//
// textBox9
//
this->textBox9->Location = System::Drawing::Point(697, 84);
this->textBox9->Name = L"textBox9";
this->textBox9->Size = System::Drawing::Size(100, 26);
this->textBox9->TabIndex = 8;
this->textBox9->Text = L"0";
//

```

```

// textBox10
//
this->textBox10->Location = System::Drawing::Point(697, 116);
this->textBox10->Name = L"textBox10";
this->textBox10->Size = System::Drawing::Size(100, 26);
this->textBox10->TabIndex = 9;
this->textBox10->Text = L"0";
//
// textBox11
//
this->textBox11->Location = System::Drawing::Point(697, 196);
this->textBox11->Name = L"textBox11";
this->textBox11->Size = System::Drawing::Size(133, 26);
this->textBox11->TabIndex = 10;
this->textBox11->Text = L"0,0001";
//
// textBox12
//
this->textBox12->BackColor =
System::Drawing::SystemColors::Control;
this->textBox12->BorderStyle =
System::Windows::Forms::BorderStyle::None;
this->textBox12->Location = System::Drawing::Point(673, 84);
this->textBox12->Name = L"textBox12";
this->textBox12->ReadOnly = true;
this->textBox12->Size = System::Drawing::Size(18, 19);
this->textBox12->TabIndex = 11;
this->textBox12->Text = L"x";
//
// textBox13
//

```



```

        this->textBox13->BackColor =
System::Drawing::SystemColors::Control;
        this->textBox13->BorderStyle =
System::Windows::Forms::BorderStyle::None;
        this->textBox13->Location = System::Drawing::Point(673, 116);
        this->textBox13->Name = L"textBox13";
        this->textBox13->ReadOnly = true;
        this->textBox13->Size = System::Drawing::Size(14, 19);
        this->textBox13->TabIndex = 12;
        this->textBox13->Text = L"y";
        //
        // textBox14
        //
        this->textBox14->BackColor =
System::Drawing::SystemColors::Control;
        this->textBox14->BorderStyle =
System::Windows::Forms::BorderStyle::None;
        this->textBox14->Location = System::Drawing::Point(697, 171);
        this->textBox14->Name = L"textBox14";
        this->textBox14->ReadOnly = true;
        this->textBox14->Size = System::Drawing::Size(100, 19);
        this->textBox14->TabIndex = 13;
        this->textBox14->Text = L"Точність є:";
        //
        // textBox15
        //
        this->textBox15->BackColor =
System::Drawing::SystemColors::Control;
        this->textBox15->BorderStyle =
System::Windows::Forms::BorderStyle::None;
        this->textBox15->Location = System::Drawing::Point(621, 58);

```

```

this->textBox15->Name = L"textBox15";
this->textBox15->ReadOnly = true;
this->textBox15->Size = System::Drawing::Size(318, 19);
this->textBox15->TabIndex = 14;
this->textBox15->Text = L"Наближені початкові значення x та
y:";

//
// radioButton4
//
this->radioButton4->AutoSize = true;
this->radioButton4->Location = System::Drawing::Point(584,
362);

this->radioButton4->Name = L"radioButton4";
this->radioButton4->Size = System::Drawing::Size(130, 24);
this->radioButton4->TabIndex = 15;
this->radioButton4->TabStop = true;
this->radioButton4->Text = L"Метод Якобі";
this->radioButton4->UseVisualStyleBackColor = true;
this->radioButton4->CheckedChanged += gcnew
System::EventHandler(this, &Form1::radioButton4_CheckedChanged);
//
// radioButton5
//
this->radioButton5->AutoSize = true;
this->radioButton5->Location = System::Drawing::Point(584,
392);

this->radioButton5->Name = L"radioButton5";
this->radioButton5->Size = System::Drawing::Size(204, 24);
this->radioButton5->TabIndex = 16;
this->radioButton5->TabStop = true;
this->radioButton5->Text = L"Метод Гауса-Зейделя";

```

```

this->radioButton5->UseVisualStyleBackColor = true;
//
// textBox16
//
this->textBox16->Location = System::Drawing::Point(697, 272);
this->textBox16->Name = L"textBox16";
this->textBox16->Size = System::Drawing::Size(133, 26);
this->textBox16->TabIndex = 17;
this->textBox16->Text = L"100";
//
// textBox17
//
this->textBox17->BackColor                                     =
System::Drawing::SystemColors::Control;
this->textBox17->BorderStyle                                   =
System::Windows::Forms::BorderStyle::None;
this->textBox17->Location = System::Drawing::Point(697, 249);
this->textBox17->Name = L"textBox17";
this->textBox17->ReadOnly = true;
this->textBox17->Size = System::Drawing::Size(259, 19);
this->textBox17->TabIndex = 18;
this->textBox17->Text = L"Максимальна кількість ітерацій:";
this->textBox17->TextChanged                                 +=          gcnew
System::EventHandler(this, &Form1::textBox17_TextChanged);
//
// textBox18
//
this->textBox18->BackColor                                     =
System::Drawing::SystemColors::Control;
this->textBox18->BorderStyle                                   =
System::Windows::Forms::BorderStyle::None;

```

```

this->textBox18->Location = System::Drawing::Point(14, 311);
this->textBox18->Name = L"textBox18";
this->textBox18->ReadOnly = true;
this->textBox18->Size = System::Drawing::Size(246, 19);
this->textBox18->TabIndex = 19;
this->textBox18->Text = L"Коефіцієнти системи рівнянь.";
//
// textBox19
//
this->textBox19->BackColor =
System::Drawing::SystemColors::Control;
this->textBox19->BorderStyle =
System::Windows::Forms::BorderStyle::None;
this->textBox19->Location = System::Drawing::Point(584, 337);
this->textBox19->Name = L"textBox19";
this->textBox19->ReadOnly = true;
this->textBox19->Size = System::Drawing::Size(246, 19);
this->textBox19->TabIndex = 20;
this->textBox19->Text = L"Методи розв'язання.";
this->textBox19->TextChanged += gcnew
System::EventHandler(this, &Form1::textBox19_TextChanged);
//
// Form1
//
this->AutoScaleDimensions = System::Drawing::SizeF(9, 20);
this->AutoScaleMode =
System::Windows::Forms::AutoScaleMode::Font;
this->ClientSize = System::Drawing::Size(1678, 764);
this->Controls->Add(this->textBox19);
this->Controls->Add(this->textBox18);
this->Controls->Add(this->textBox17);

```

```

this->Controls->Add(this->textBox16);
this->Controls->Add(this->radioButton5);
this->Controls->Add(this->radioButton4);
this->Controls->Add(this->textBox15);
this->Controls->Add(this->textBox14);
this->Controls->Add(this->textBox13);
this->Controls->Add(this->textBox12);
this->Controls->Add(this->textBox11);
this->Controls->Add(this->textBox10);
this->Controls->Add(this->textBox9);
this->Controls->Add(this->button2);
this->Controls->Add(this->chart1);
this->Controls->Add(this->textBox7);
this->Controls->Add(this->richTextBox1);
this->Controls->Add(this->button1);
this->Controls->Add(this->table);
this->Controls->Add(this->groupBox1);
this->Name = L"Form1";
this->Text = L"Калькулятор систем нелінійних рівнянь";
this->Load += gcnew System::EventHandler(this,
&Form1::Form1_Load);
this->groupBox1->ResumeLayout(false);
this->groupBox1->PerformLayout();

(cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this->table))-
>EndInit();

(cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this->chart1))-
>EndInit();

this->ResumeLayout(false);
this->PerformLayout();

```

```

    }
#pragma endregion

    /// <summary>
    /// Перевіряє дані таблиці на коректність.
    /// Записує дані у двовимірний масив.
    /// </summary>
    private: vector<vector<double>> scantable(vector<vector<double>> table1)
    {
        string res;
        double value;
        for (int i = 0; i < 2; i++)
        {
            for (int j = 0; j < 3; j++)
            {
                if (!Double::TryParse(Convert::ToString(table->Rows[i]-
>Cells[j]->Value), value))
                {
                    res = "Некоректно введені дані";
                    richTextBox1->Text = gcnew String(res.c_str());
                    return table1;
                }
                table1[i][j] = Convert::ToDouble(table->Rows[i]->Cells[j]-
>Value);
            }
        }
        return table1;
    }

    private: System::Void dataGridView1_CellContentClick(System::Object^
sender, System::Windows::Forms::DataGridViewCellEventArgs^ e) {

```

```

    }
    /// <summary>
    /// Перевіряє введені дані на коректність.
    /// Розв'язує обрану систему нелінійних рівнянь обраним методом.
    /// Виводить результат.
    /// Зберігає результат.
    /// </summary>
private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e)
{
    string text = "";
    richTextBox1->Text = gcnew String(text.c_str());
    vector<vector<double>> table1(2, vector<double>(3));
    text = "Некоректно введені дані";
    table1 = scantable(table1);
    double value;
    int value2;
    string str_eq = "";
    if (!Double::TryParse(textBox11->Text, value) || !Double::TryParse(textBox9-
>Text, value) || !Double::TryParse(textBox10->Text, value)
|| !Int32::TryParse(textBox16->Text, value2) ||
        Convert::ToDouble(textBox11->Text) <= 0 ||
Convert::ToInt32(textBox16->Text) <= 0)
    {
        richTextBox1->Text = gcnew String(text.c_str());
    }
    if (richTextBox1->Text != gcnew String(text.c_str()))
    {
        double e = Convert::ToDouble(textBox11->Text);
        double x1 = Convert::ToDouble(textBox9->Text);
        double y1 = Convert::ToDouble(textBox10->Text);
        int num = Convert::ToInt32(textBox16->Text);

```

```

if (radioButton1->Checked)
{
    str_eq = to_string(table1[0][0]) + "x^2 + " + to_string(table1[0][1])
+ "y^2 + " + to_string(table1[0][2]) + " = 0 \n" +
    to_string(table1[1][0]) + "x^2 + " + to_string(table1[1][1]) +
"y^2 + " + to_string(table1[1][2]) + " = 0 \n";
    Alg_eq obj = Alg_eq(table1, x1, y1, e, num);
    if (radioButton4->Checked)
    {
        text = obj.solution_Yacobi();
        richTextBox1->Text = gcnew String(text.c_str());
        str_eq += text;
        writeTextFile(str_eq);

    }
    else if (radioButton5->Checked)
    {
        text = obj.solution_Gaus_Zeydel();
        richTextBox1->Text = gcnew String(text.c_str());
        str_eq += text;
        writeTextFile(str_eq);

    }

}
else if (radioButton2->Checked)
{
    str_eq = "sin(x + " + to_string(table1[0][0]) + ") + " +
to_string(table1[0][1]) + "y + " + to_string(table1[0][2]) + " = 0 \n" +
    to_string(table1[1][0]) + "x + cos(y + " +
to_string(table1[1][1]) + ") + " + to_string(table1[1][2]) + " = 0 \n";
    Trig_eq obj = Trig_eq(table1, x1, y1, e, num);

```



```

if (radioButton4->Checked)
{
    text = obj.solution_Yacobi();
    richTextBox1->Text = gcnew String(text.c_str());
    str_eq += text;
    writeTextFile(str_eq);
}
else if (radioButton5->Checked)
{
    text = obj.solution_Gaus_Zeydel();
    richTextBox1->Text = gcnew String(text.c_str());
    str_eq += text;
    writeTextFile(str_eq);
}

}
else if (radioButton3->Checked)
{
    str_eq = "e^(" + to_string(table1[0][0]) + "x) + e^(" +
to_string(table1[0][1]) + "y) + " + to_string(table1[0][2]) + "= 0 \ne^(" +
to_string(table1[1][0]) + "x) + e^(" + to_string(table1[1][1])
+ "y) + " + to_string(table1[1][2]) + "= 0\n";
    Tran_eq obj = Tran_eq(table1, x1, y1, e, num);
    if (radioButton4->Checked)
    {
        text = obj.solution_Yacobi();
        richTextBox1->Text = gcnew String(text.c_str());
        str_eq += text;
        writeTextFile(str_eq);
    }
    else if (radioButton5->Checked)

```

```

        {
            text = obj.solution_Gaus_Zeydel();
            richTextBox1->Text = gcnew String(text.c_str());
            str_eq += text;
            writeTextFile(str_eq);
        }
    }
}

private: System::Void table_CellValueChanged(System::Object^ sender,
System::Windows::Forms::DataGridViewCellEventArgs^ e) {
    table->AutoSizeColumns();
}

private: System::Void table_ColumnAdded(System::Object^ sender,
System::Windows::Forms::DataGridViewColumnEventArgs^ e) {
}

private: System::Void Form1_Load(System::Object^ sender, System::EventArgs^ e) {
    table->ColumnCount = 3;
    table->RowCount = 2;
    table->AutoSizeColumns();
    chart1->ChartAreas[0]->AxisX->ScaleView->Zoom(-100,100);
    chart1->ChartAreas[0]->CursorX->IsUserEnabled = true;
    chart1->ChartAreas[0]->CursorX->IsUserSelectionEnabled = true;
    chart1->ChartAreas[0]->AxisX->ScaleView->Zoomable = true;
    chart1->ChartAreas[0]->AxisX->ScrollBar->IsPositionedInside = true;
    chart1->ChartAreas[0]->AxisY->ScaleView->Zoom(-100, 100);
    chart1->ChartAreas[0]->CursorY->IsUserEnabled = true;
    chart1->ChartAreas[0]->CursorY->IsUserSelectionEnabled = true;
    chart1->ChartAreas[0]->AxisY->ScaleView->Zoomable = true;
    chart1->ChartAreas[0]->AxisY->ScrollBar->IsPositionedInside = true;

```

```

}
private: System::Void table_Click(System::Object^ sender, System::EventArgs^ e) {
}
private:      System::Void      radioButton1_Click(System::Object^      sender,
System::EventArgs^ e) {
}
private:      System::Void      textBox8_TextChanged(System::Object^      sender,
System::EventArgs^ e) {
}
/// <summary>
/// Перевіряє дані таблиці на коректність.
/// Будує графік обраної системи нелінійних рівнянь.
/// </summary>
private: System::Void button2_Click(System::Object^ sender, System::EventArgs^ e)
{
    chart1->Series[0]->Points->Clear();
    chart1->Series[1]->Points->Clear();
    chart1->Series[2]->Points->Clear();
    chart1->Series[3]->Points->Clear();

    string text1 = "Некоректно введені дані";
    string text = "";
    richTextBox1->Text = gcnew String(text.c_str());
    vector<vector<double>> table1(2, vector<double>(3));
    table1 = scantable(table1);
    if (richTextBox1->Text != gcnew String(text1.c_str()))
    {

        if (radioButton1->Checked)
        {

```

```

        if (!(table1[0][0] > 0 && table1[0][1] > 0 && table1[0][2] > 0) ||
            (table1[1][0] > 0 && table1[1][1] > 0 && table1[1][2] > 0) ||
                (table1[0][0] < 0 && table1[0][1] < 0 && table1[0][2] < 0) ||
            (table1[1][0] < 0 && table1[1][1] < 0 && table1[1][2] < 0) || table1[0][1] == 0 ||
            table1[1][1] == 0))
        {
            for (double x = -100; x < 100; x += 0.1)
            {
                chart1->Series[0]->Points->AddXY(x,      pow(((table1[0][0] * pow(x, 2) - table1[0][2]) / table1[0][1]), 0.5));
                chart1->Series[1]->Points->AddXY(x,      -pow(((table1[0][0] * pow(x, 2) - table1[0][2]) / table1[0][1]), 0.5));
                chart1->Series[2]->Points->AddXY(x,      pow(((table1[1][0] * pow(x, 2) - table1[1][2]) / table1[1][1]), 0.5));
                chart1->Series[3]->Points->AddXY(x,      -pow(((table1[1][0] * pow(x, 2) - table1[1][2]) / table1[1][1]), 0.5));
            }
        }
    }
    else if (radioButton2->Checked)
    {
        if (!(table1[0][1] == 0 || table1[1][0] == 0))
        {
            for (double x = -100; x < 100; x += 0.1)
            {
                chart1->Series[2]->Points->AddXY(x,      (-sin(x      +
table1[0][0]) - table1[0][2]) / table1[0][1]);
            }
            for (double y = -100; y < 100; y += 0.1)
            {

```

```

        chart1->Series[0]->Points->AddXY((-cos(y
        +
table1[1][1]) - table1[1][2]) / table1[1][0], y);
    }
}
}
else if (radioButton3->Checked)
{
    if (!(table1[0][2] >= 0 || table1[1][2] >= 0 || table1[0][1] == 0 ||
table1[1][1] == 0))
    {
        for (double x = -100; x < 100; x += 0.1)
        {
            chart1->Series[0]->Points->AddXY(x,      log(-
exp(table1[0][0] * x) - table1[0][2]) / table1[0][1]);
            chart1->Series[2]->Points->AddXY(x,      log(-
exp(table1[1][0] * x) - table1[1][2]) / table1[1][1]);
        }
    }
}
}

}

private: System::Void radioButton4_CheckedChanged(System::Object^ sender,
System::EventArgs^ e) {
}

private: System::Void textBox17_TextChanged(System::Object^ sender,
System::EventArgs^ e) {
}

private: System::Void textBox19_TextChanged(System::Object^ sender,
System::EventArgs^ e) {
}

```

```

}
};
}

```

Файл «func.cpp»

```

#include "pch.h"
#include "func.h"
void writeTextFile(string text)
{
    ofstream textFile("Result");
    textFile << text << endl;
    textFile.close();
}

```

Файл «func.h»

```

#pragma once
#include <fstream>
#include <string>
using namespace std;
/// <summary>
/// Записує результат системи нелінійних рівнянь та її розв'язання у текстовий
файл.
/// </summary>
void writeTextFile(string);

```

Файл «pch.cpp»

// pch.cpp: вихідний файл, що відповідає попередньо скомпільованому заголовку

```

#include "pch.h"

```

// При використанні попередньо скомпільованих заголовків цей вихідний файл потрібен для успішної компіляції.

Файл «pch.h»

// pch.h: це попередньо скомпільований заголовний файл.

// Файли, наведені нижче, компілюються лише один раз, щоб покращити продуктивність збірки для майбутніх збірок.

// Це також впливає на продуктивність IntelliSense, включаючи завершення коду та багато функцій перегляду коду.

// Однак перераховані тут файли ВСІ будуть перекомпільовані, якщо хоча б один з них буде оновлено між збірками.

// Не додавайте сюди файли, які будуть часто оновлюватися, оскільки це призведе до зниження продуктивності.

```
#ifndef PCH_H
```

```
#define PCH_H
```

// Додайте сюди заголовки для попередньої компіляції.

```
#endif //PCH_H
```

Файл «Resource.h»

```
//{ {NO_DEPENDENCIES} }
```

// Включити файл, згенерований Microsoft Visual C++.

// Використовується app.rc

Файл «transcendental_equations.cpp»

```
#include "pch.h"
```

```
#include "transcendental_equations.h"
```

```
Tran_eq::Tran_eq(vector<vector<double>> table, double x, double y, double e, int num) :Base_eq(table, x, y, e, num) { };
```

```
string Tran_eq::solution_Yacobi()
```

```
{
```

```
    return Base_eq::solution_Yacobi();
```

```
}
```

```
string Tran_eq::solution_Gaus_Zeydel()
```

```
{
```

```
    return Base_eq::solution_Gaus_Zeydel();
```

```

}
string Tran_eq::find_x(double y)
{
    string res = "";
    if ((-exp(param[0][1] * y) - param[0][2]) <= 0 || param[0][0] == 0)
    {
        res = "Немає розв'язків";
    }
    else
    {
        res = to_string(log(-exp(param[0][1] * y) - param[0][2]) / param[0][0]);
    }
    return res;
}
string Tran_eq::find_y(double x)
{
    string res = "";
    if ((-exp(param[1][0] * x) - param[1][2]) <= 0 || param[1][1] == 0)
    {
        res = "Немає розв'язків";
    }
    else
    {
        res = to_string(log(-exp(param[1][0] * x) - param[1][2]) / param[1][1]);
    }
    return res;
}
string Tran_eq::it_print(double x, double y, int num)
{
    string res = "";
    res += "Ітерація " + to_string(num) + ": x = " + to_string(x) + "\n";
}

```



```

        res += "y = " + to_string(y) + "\n";
        return res;
    }
string Tran_eq::res_print(double x, double y)
{
    string res = "";
    res += "Результат: x = " + to_string(x) + "\n";
    res += "y = " + to_string(y) + "\n";
    return res;
}

```

Файл «transcendental_equations.h»

```
#pragma once
```

```
#include "base_equations.h"
```

```
/// <summary>
```

```
/// Клас для заданої системи нелінійних трансцендентних рівнянь.
```

```
/// </summary>
```

```
class Tran_eq: private Base_eq
```

```
{
```

```
public:
```

```
    Tran_eq(vector<vector<double>>, double, double, double, int);
```

```
    string find_x(double);
```

```
    string find_y(double);
```

```
    string solution_Yacobi();
```

```
    string solution_Gaus_Zeydel();
```

```
    string it_print(double, double, int);
```

```
    string res_print(double, double);
```

```
};
```

Файл «trigonometric_equations.cpp»

```
#include "pch.h"
```

```
#include "trigonometric_equations.h"
```

```

Trig_eq::Trig_eq(vector<vector<double>> table, double x, double y, double e, int
num) :Base_eq(table, x, y, e, num) {};
string Trig_eq::solution_Yacobi()
{
    return Base_eq::solution_Yacobi();
}
string Trig_eq::solution_Gaus_Zeydel()
{
    return Base_eq::solution_Gaus_Zeydel();
}
string Trig_eq::find_x(double y)
{
    string res = "";
    if (param[1][0] == 0)
    {
        res = "Немає розв'язків";
    }
    else
    {
        res = to_string((-cos(y + param[1][1]) - param[1][2]) / param[1][0]);
    }
    return res;
}
string Trig_eq::find_y(double x)
{
    string res = "";
    if (param[0][1] == 0)
    {
        res = "Немає розв'язків";
    }
    else

```

```

    {
        res = to_string((-sin(x + param[0][0]) - param[0][2]) / param[0][1]);
    }
    return res;
}

string Trig_eq::it_print(double x, double y, int num)
{
    string res = "";
    res += "Ітерація " + to_string(num) + ": x = " + to_string(x) + "\n";
    res += "y = " + to_string(y) + "\n";
    return res;
}

string Trig_eq::res_print(double x, double y)
{
    string res = "";
    res += "Результат: x = " + to_string(x) + "\n";
    res += "y = " + to_string(y) + "\n";
    return res;
}

```

Файл «trigonometric_equations.h»

```
#pragma once
```

```
#include "base_equations.h"
```

```
/// <summary>
```

```
/// Клас для заданої системи нелінійних тригонометричних рівнянь.
```

```
/// </summary>
```

```
class Trig_eq: private Base_eq
```

```
{
```

```
public:
```

```
    Trig_eq(vector<vector<double>>, double, double, double, int);
```

```
    string find_x(double);
```

```
    string find_y(double);
```

```
string solution_Yacobi();  
string solution_Gaus_Zeydel();  
string it_print(double, double, int);  
string res_print(double, double);  
};
```