НАВЧАЛЬНО-НАУКОВИЙ КОМПЛЕКС

«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»

НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

Імені ІГОРЯ СІКОРСЬКОГО

КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

**КУРСОВА РОБОТА**

**з дисципліни «Алгоритмізація та програмування»**

**на тему: «Пакет розв’язку систем лінійних рівнянь»**

Виконав:

студент І курсу, групи КА-06

Мазніченко Лев Владиславович \_\_\_\_\_\_\_\_\_\_

Керівник:

доцент, к.т.н.

**Безносик О. Ю.** \_\_\_\_\_\_\_\_\_\_

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент (-ка) \_\_\_\_\_\_\_\_\_\_\_\_\_

Київ – 2021 рік

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ННК „ІПСА” НТУУ „КПІ”** | | | | | | |
| (назва вищого закладу освіти) | | | | | | |
| Кафедра | | ***Математичних методів системного аналізу*** | | | | |
| Дисципліна | | ***Програмування*** | | | | |
| Галузь знань | | ***12 Інформаційні технології*** | | | | |
| Курс | ***перший*** | | Група | ***КА—06*** | Семестр | ***другий*** |

**ЗАВДАННЯ**

**на курсовий проект(роботу) студента**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
| (прізвище, ім’я, по батькові) | | | | | |
| 1.Тема проекту(роботи) | Інформаційно-пошукова система. Бібліотека. | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| 2. Строк здачі студентом закінченого проекту(роботи) | | | ***01.06.2021 р.*** | | |
| 3. Вихідні дані до проекту(роботи) | | Середовище програмування  CLion | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| 4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці) | | | | |  |
| ***1. Постановка задачі.*** | | | | | |
| ***2. Метод розв’язку задачі*** | | | | | |
| ***3. Загальна блок-схема алгоритму та опис алгоритму*** | | | | | |
| ***4. Опис програмного продукту.*** | | | | | |
| ***5. Результати роботи.*** | | | | | |
| ***6. Висновки.*** | | | | | |
| ***7. Список використаної літератури.*** | | | | | |
| ***Додаток*** | | | | | |
| 5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень) | | | |  | |
| ***1. Загальна блок-схема алгоритму.*** | | | | | |
| ***2. Ілюстрації роботи програми.*** | | | | | |
| 6. Дата видачі завдання | 01.02.2021 | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| №/п | Назва етапів курсового проекту (роботи) | Строк виконання  етапів роботи | Примітка |
| 1. | Вибір теми курсової роботи. Опрацювання відповідної літератури.  Оформлення листа Завдання. | 01.02.2021-10.02.2021 |  |
| 2. | Аналіз постановки задачі. | 11.02.2021-15.02.2021 |  |
| 3. | Вибір та дослідження методів, вибір відповідних структур даних, розробка алгоритму. Перше узгодження з керівником. | 16.02.2021-28.02.2021 |  |
| 4. | Проектування інтерфейсу. | 01.03.2021-16.03.2021 |  |
| 5. | Друге узгодження з керівником. | 17.03.2021-19.03.2021 |  |
| 6. | Програмна реалізація. | 20.03.2021-13.04.2021 |  |
| 7. | Демонстрація першого варіанту.  Трете узгодження з керівником. | 14.04.2021-16.04.2021 |  |
| 8. | Заключне тестування програми. | 17.04.2021-21.04.2021 |  |
| 9. | Аналіз результатів.  Оформлення звіту. | 22.04.2021-07.05.2021 |  |
| 10. | Захист та демонстрація курсової роботи. | 17.05.2021 – 19.05.2021 - «А»  20.05.2021-21.05.2021 – «В,С»  24.05.2021- 28.05.2021 – «D,E» |  |

|  |  |  |
| --- | --- | --- |
| Студент |  | Мазніченко Лев Владиславович |
|  | (підпис) |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Керівник |  |  |  |
|  | (підпис) |  | (прізвище, ім’я, по батькові) |

|  |  |
| --- | --- |
|  |  |
| (дата) |  |

**ЗМІСТ**

ВСТУП 6

РОЗДІЛ 1 ПОСТАНОВКА ЗАДАЧІ

* 1. Огляд існуючих підходів до розв’язання поставленої задачі 8
  2. Обґрунтування обраного варіанту реалізації 9
  3. Уточнена постановка задачі 11

РОЗДІЛ 2 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

* 1. Метод розв’язку задачі 13
  2. Алгоритм розв’язку задачі 15

РОЗДІЛ 3 ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ПРОДУКТУ

3.1 Опис головних структур і змінних програми 18

3.2 Опис головних функцій програми 19

3.3 Опис інтерфейсу 19

3.4 Результати роботи програмного продукту 19

ВИСНОВКИ 25

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 26

ДОДАТОК 27

**ВСТУП**

Програмне забезпечення для вирішення СЛАР(далі – системи лінійних алгебрїчних рівнянь) відіграє надзвичайно важливу роль для розвитку прикладних задач у таких сферах, як: лінійна алгебра, математичний аналіз, фізика, інженерні та економічні обчислення.

Виконання даної роботи розвиває логічне, математичне та абстрактне мислення та навички розробки програмного забезпечення.

Метою даної роботи є розробка та реалізація програмного забезпечення для виконання вирішення СЛАР, призначеного для розв’язку задач лінійної алгебри, поглиблення та закріплення набутих під час вивчення математичних дисциплін та дисципліни «Алгоритмізація та програмування» теоретичних знань та практичних навичок, в тому числі, щодо застосування математичних методів в галузі інформаційних технологій, зокрема написання програмного забезпечення для реалізації чисельних методів.

Об’єктом дослідження є програмне забезпечення для розв’язання СЛАР.

Завдання роботи: на основі аналізу літературних джерел дослідити програми-аналоги, поглибити знання з лінійної алгебри, навчитися застосовувати математичні методи у процесі автоматизації розрахункових робіт, визначити вимоги до розроблюваного програмного забезпечення, використати методи об’єктно-орієнтованого та структурного програмування, створити зручний користувацький інтерфейс використовуючи засоби С++ для роботи з консольним вікном.

Розроблюване програмне забезпечення призначене для інженерних та математичний розрахунків.

При виконанні роботи для реалізації програмного забезпечення було використане середовище розробки CLion, що функціонує на платформі MacOS BigSur, для пошуку інформації в мережі Інтернет - веб-браузер Safari, для оформлення тексту пояснювальної записки - компоненти пакету Microsoft Office, зокрема, текстовий процесор Word.

Робота складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків .

Інформаційною основою дослідження є чинні міжнародні стандарти та Державні стандарти України в галузі інформаційних технологій та наукових досліджень, праці вітчизняних та закордонних науковців та інженерів-практиків, Інтернет-джерела та інша, в тому числі, навчальна література.

# **РОЗДІЛ 1**

**ОБГРУТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ, ПОСТАНОВКА ЗАДАЧІ ПРОЕКТУВАННЯ**

* 1. **Програмне забезпечення для вирішення СЛАР**

Калькулятори для СЛАР є широко використовуваними

продуктами в усіх галузях науки і технікі, оскільки дозволяють уникнути механічних помилок, яких можна допустити під час розв’язання, витрачати менше часу та збільшити точність підрахунків.

Прикладами таких калькуляторів є Photomath і Mathcad – безкоштовні сервіси для розв’язку різного роду математичних задач, але Mathcad є набагато складнішим як в програмному плані, так і в плані інтерфейсу та своїх можливостей, саме тому Photomath є таким популярним серед учеників шкіл та студентів перших курсів спеціальностей, що не зв’язані з математикою. На відміну від звичайних калькуляторів обидва варіанти вказують шлях рішення, вказують методи за якими вони вирішили поставлену задачу. Варто сказати, що існує безліч сайтів і калькуляторів, що написані користувачами для себе, свого сайту, зроблені в якості курсових чи дипломних робіт, але ми будемо розглядати 2 найбільш популярних і значних додатки.[2]

У таблиці 1.2 подано порівняльну характеристику описаних програмних засобів для розв’язання СЛАР

|  |  |  |
| --- | --- | --- |
| Додаток | Mathcad | Photomath |
| Зручний інтерфейс | + | + |
| Наявність мобільного додатку | - | + |
| Зручність використання | + | + |
| Широкий пакет можливостей | + | + |
| Плата за продукт | + | Опціонально |
| Наявність пояснення | + | + |

Порівнявши дані додатки і їх методи вирішення СЛАР, було прийнято рішення розробити додаток, де користувач сам зможе обрати спосіб розв’язання своєї СЛАР.

* 1. **. Обґрунтування вибору середовища та інструментів розробки**

Найбільш поширеною у наш час мовою програмування, на думку багатьох дослідників, упродовж кількох останніх десятиліть, є мова С++, на основі якої з’явилося багато сучасних мов програмування і програмних середовищ. Цьому сприяли такі її властивості:

* потужність;
* гнучкість;
* можливість доступу до всіх функціональних засобів системи.
* оптимізованість;
* багатоплатформенність;
* прозорість принципу роботи;

Для того щоб вибрати C ++ як мову програмування існує чотири

причини:

* Компільований мову зі статичної типізацією.
* Поєднання високорівневих і низькорівневих засобів.
* Реалізація ООП.
* STL.

Розглянемо ці фактори більш докладно:

Компілятор. Тут у мові C ++ є багато нюансів: безліч компіляторів, консольні команди, етапи складання програми. снують опції збірки, і існує не один компілятор. Вихідний код, написаний програмістом, може бути попередньо оброблений і змінений (наприклад, препроцесором).Майбутній фахівець усвідомлює, що код сам по собі не запускається. Після роботи з С++ студент буде знати, що процес складання програми можна налаштувати і що вихідний код може бути оброблений сторонніми програмами. В майбутньому, при використанні IDE, людина буде розуміти, що це всього лише зручний комплекс програм, що виконує велику частину рутинних операцій і в разі недостатньої гнучкості від неї можна відмовитися або розширити.

Статична типізація**.** На прикладі мови зі статичної типізацією простіше зрозуміти, що таке тип даних, навіщо він потрібен і від чого залежить. Видно, що собою являє оголошення, визначення та ініціалізація. Використання мови C ++ дає це явно побачити, що сприяє подальшому розумінню того, як працюють ці механізми в інших мовах. Крім цього можна на реальних прикладах зрозуміти, чим беззнакові цілі відрізняються від цілих зі знаком, чим відрізняються числа подвійної і одинарної точності, чим відрізняється символ від рядка і т.д.

Високорівневі і низькорівневі засоби**.** Використання таких засобів, як вказівники і динамічне виділення пам'яті, дозволяє зрозуміти (або надалі сприяє розумінню), що таке стек, куча, стек викликів, розкрутка стека і т.д. Крім цього, на практиці закріплюється розуміння концепції адреси та адресної арифметики. На прикладах демонструється, що пам'ять треба виділяти, звільняти, тому що вона не нескінченна, що існують витоки пам'яті. В майбутньому, при вивченні мов з GC простіше буде зрозуміти, що ж це таке.

Окремо варто відзначити простий механізм передачі значень за посиланням, значенням, вказівником і перенесення об'єкта. Що таке змінювані і не змінюються параметри. Надалі дані концепції можуть бути використані і при вивченні інших мов. Студент буде розуміти, наприклад, що об'єкт в мові N передається по посиланню, і якщо його значення змінити в функції-члені, то воно зміниться всюди.

Реалізація ООП. Це відносно чиста реалізація ООП. Чітко розмежовані рівні доступу до членів класу, можливість множинного успадкування і динамічного поліморфізму дають можливість швидко засвоїти основні концепції ООП (абстракція, успадкування, інкапсуляція і поліморфізм). Покажчики і динамічне виділення пам'яті дозволяють наочно зрозуміти такі важливі механізми, як upcasting і downcasting. Надалі, грунтуючись на цих знаннях, легко можна зрозуміти весь синтаксис в інших мовах.

Варто відзначити такий важливий момент, як і непримусове ООП. Тобто даний підхід до програмування застосовується тоді, коли це зручно, і його можна змішувати, наприклад, з функціональним програмуванням. Це сприяє формуванню розуміння того, що методи реалізації вибираються виходячи з завдання.

STL. Сама по собі концепція шаблонів C ++, генерації коду і застосування широкого спектру алгоритмів до різних контейнерів позитивно впливає на процес навчання. Тут все на поверхні і зрозуміло, чому можна створити вектор цілих чисел і вектор об'єктів на основі одного класу-контейнера. Чому можна застосувати деяку операцію до послідовності об'єктів або як впорядкувати об'єкти, для яких не передбачена вбудована операція порівняння. Можна зрозуміти, як здійснюється доступ до елементів, і дізнатися про категорії ітераторів. Крім цього закріплюється розуміння узагальненого програмування.

Переваги C ++ над C:

* Безпека
* Можливість писати узагальнений код за допомогою шаблонів
* Можливість використовувати об'єктно-орієнтований підхід
* Управління ресурсами за допомогою RAII
* Спрощення коду за рахунок перевантаження функцій і операторів
* Більш проста обробка помилок за рахунок винятків

[14]

* 1. **. Постановка задачі проектування**

Проаналізувавши літературні джерела, розглянувши існуючі варіанти розробки, можна уточнити задачі проектування. Створюване програмне забезпечення призначене для реалізації чисельних методів для роботи з матрицями, має забезпечувати виконання таких функцій для роботи з матрицями: знайти визначник, знайти ранг, знайти обернену матрицю , транспонувати матрицю, помножити матрицю на константу, додати дві матриці, відняти одну матрицю від іншої, перемножити матриці

Тому, задачі проектування можна сформулювати наступним чином.

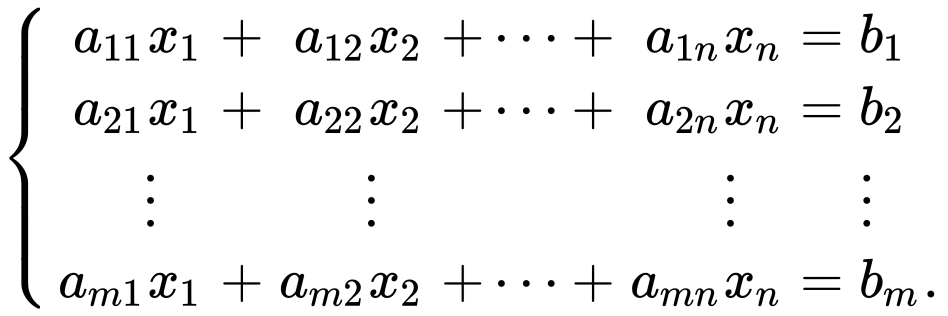
1. Дослідити предметну область, інформаційні джерела щодо способів вирішення задач лінійної алгебри.
2. Проаналізувати існуючі методи розв’язку задач з використанням СЛАР, їх особливості та специфіку.
3. Обрати методи, які будуть реалізовані в програмі.
4. Дослідити переваги та недоліки існуючих програм-аналогів.
5. Обґрунтувати вибір інструментів розробки відповідно до вимог технічного завдання, забезпечивши реалізацію всіх функцій.
6. Реалізувати інтерфейс користувача.
7. Розробити алгоритм та відповідне програмне забезпечення обраною мовою програмування (мовою С++) у середовищі CLion.
8. Дослідити якість роботи програми на контрольних прикладах.
9. Створити опис основної програми і окремих програмних модулів.
10. Розробити опис інтерфейсу та інструкцію для користувача.
11. Визначити перспективи удосконалення розробки.[10]

**РОЗДІЛ 2**

**РОЗРОБКА АЛГОРИТМУ РОЗВ’ЯЗКУ ЗАДАЧІ**

**2.1. Способи розв’язання СЛАР**

СЛАУ(система лінійних алгебраїчних рівнянь) – в лінійній алгебрі система лінійних рівнянь, що має вигляд:



В лінійній алгебрі існують 2 види методів розв’язання СЛАР: прямі та ітераційні. Серед прямих студентам першого курсу відомі метод Крамера та метод Гаусса, а серед ітераційних можна виділити метод Якобі та метод Гаусса-Зейделя.[2]

Отже для своєї роботи я обрав саме ці чотири методи. Нижче я навів опис цих методів.

Метод Гаусса:

Шляхом елементарних перетворень рядків (додавань до рядка іншого рядка, помноженого на число, і переставлянь рядків) матрицю приводять до верхньотрикутного вигляду (сходинчастого вигляду).

З цього моменту починається зворотний хід. З останнього ненульового рівняння виражають кожну з базисних змінних через небазисні й підставляють до попередніх рівнянь. Повторюючи цю процедуру для всіх базисних змінних, отримують фундаментальний розв'язок.

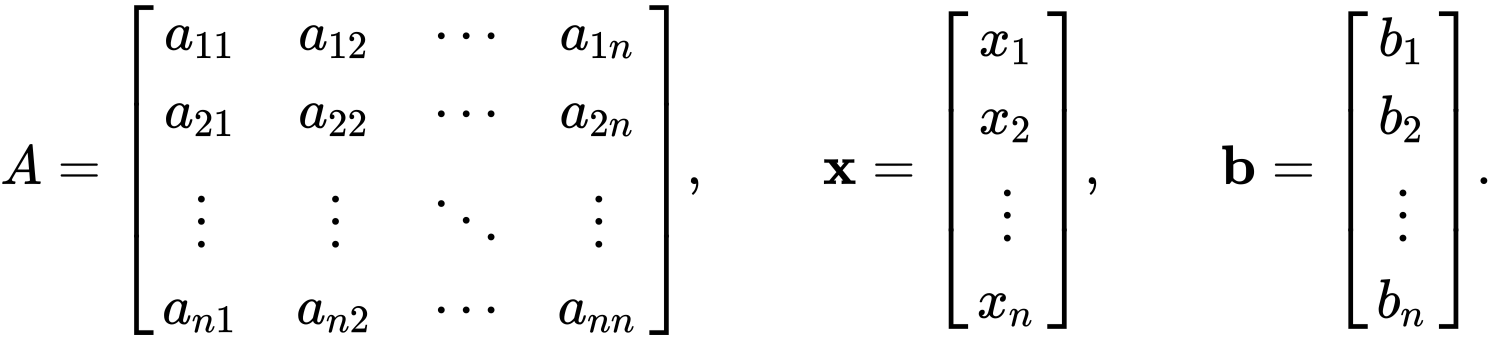
Метод Крамера:

Якщо визначник системи не дорівнює нулю, то корені системи знаходяться за формулою Крамера[3]:

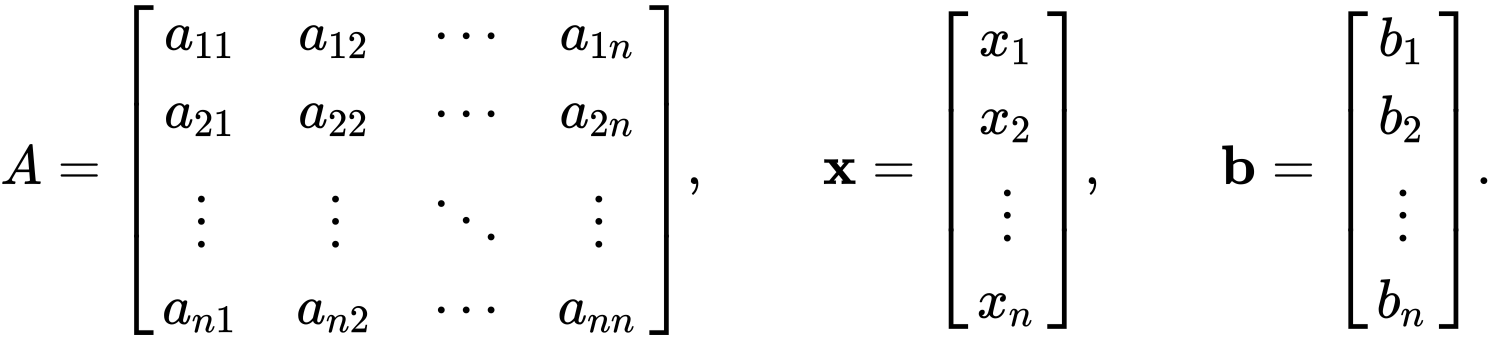
Метод Якобі:

Для СЛАР

де



Матрицю А можна розкласти на 2 доданки: діагональну матрицю D, та все інше R:



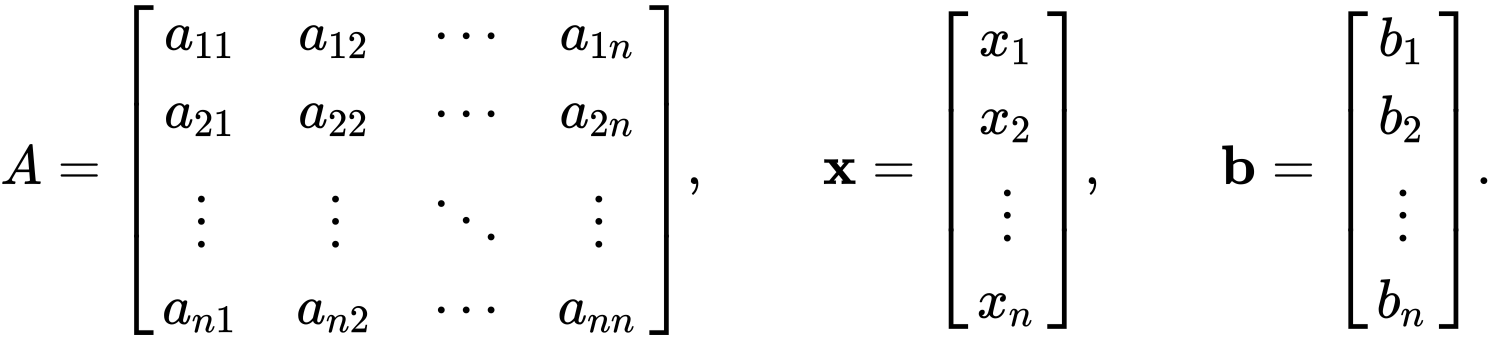
Систему лінійних рівнянь можна переписати у вигляді:

Ітераційний метод Якобі виражається формулою:

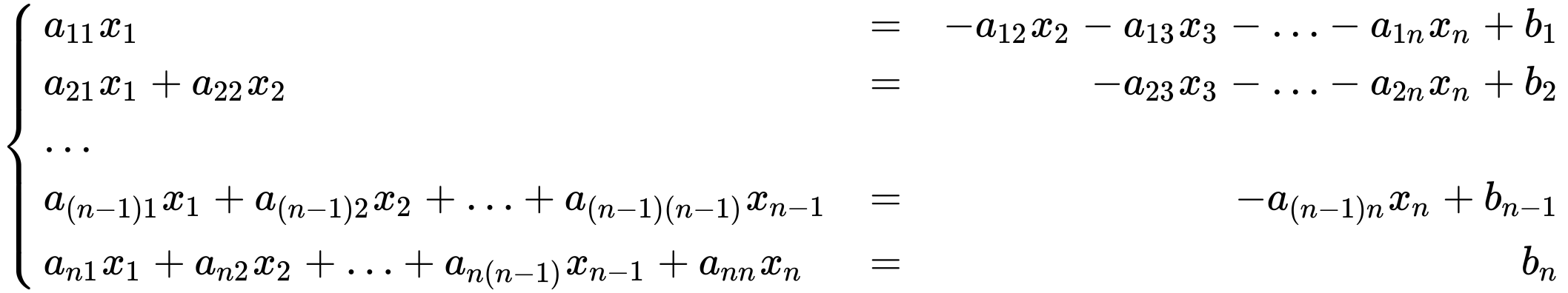
Метод Гаусса-Зейделя:

Для СЛАР

де



Перепишемо задачу у вигляді



Отримана система може бути представлена в такому вигляді:

де в прийнятих позначеннях D означає матрицю, у якої на головній діагоналі стоять відповідні елементи матриці A, а всі інші – нулі; тоді як на матриці U та L містять верхню і нижню трикутні частини A, на головній діагоналі яких нулі.[5]

Ітеративний процес будується за наведеною нижче формулою після вибору початкового наближення

Таким чином i-тий компонент (k+1)-го наближення обчислюється за формулою:

**2.2. Алгоритм розв’язку задачі**

Враховуючи специфіку задачі курсової роботи, при розробці алгоритму використано метод покрокової деталізації, згідно якої складний алгоритм розбивається на підзадачі, кожна з яких в свою чергу також розбивається на декілька простіших. Це дозволяє також розбити алгоритм на окремі модулі, у яких реалізовуватимуться окремі методи розв’язання СЛАР. Для забезпечення зручної роботи користувачу, значна увага у розроблюваному алгоритмі приділена організації каскадного меню, за допомогою якого користувач вибирає необхідні методи.[1][3][4][5]

Вхідною інформацією для функцій роботи з матрицями є задані користувачем рівняння. Результати роботи функцій роботи з матрицями – значення шуканих змінних.

Головний алгоритм програми виконується у наступні кроки:

1. Запустити програму.
2. Вибрати метод розв’язання.
3. Ввести елементи СЛАР.
4. Якщо значення було введено коректно, то функція зробить необхідні розрахунки та поверне результат операції.
5. Запропонувати користувачу розв’язати ще одну СЛАР.

На рис. 2.3 зображено узагальнену блок-схему основного алгоритму

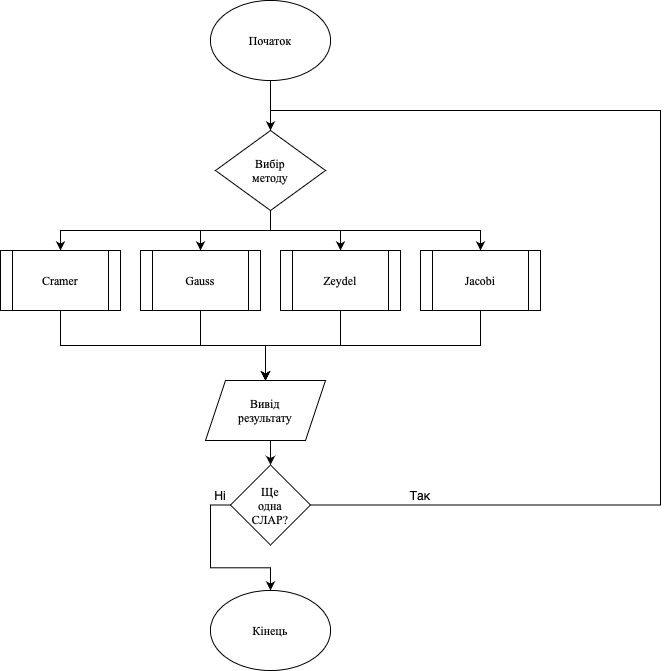


Рисунок 2.3 – Узагальнена блок-схема алгоритму

Отже, розроблений алгоритм задовольняє вимогам завдання, є гнучким, структурованим, за потреби, може бути доповнений новими модулями та функціями.[4]

**РОЗДІЛ 3**

**ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ПРОДУКТУ**

**3.1. Опис головних структур та змінних**

Розроблене програмне забезпечення має меню за допомогою якого користувач обирає потрібний йому метод розв’язання СЛАР. Оскільки усі вищеперераховані методи працюють з СЛАР яка має вигляд матриці, то, очевидно, зберігати наші СЛАР ми будемо у вигляді матриць, а саме – двовимірний масив для коефіцієнтів і одновимірний для вільних членів.

Структури у роботі не використовуються, як і глобальні змінні, усі змінні є локальними для своїх функцій або груп функцій.

У кожній функції використовується змінна a[][] яка є нашою СЛАР в матричному виді і яку можна вважати головною змінною усієї програми.

**3.2. Опис основних функцій**

У таблиці 3.1 подано опис усіх функцій, їх призначення і семантику змінних у порядку появі у коді:

|  |  |  |  |
| --- | --- | --- | --- |
| № | Прототип функції | Призначення функції | Параметри |
| 1 | void restart | Запропонувати розв’язати ще одну СЛАР | - |
| 2 | void Gauss | Метод Гаусса | - |
| 3 | void Cramer | Метод Крамера | - |
| 4 | float determinant | Знайти показник основної матриці | float matrix |
| 5 | float determinantX1 | Знайти показник матриці з заміною першого рядка | float coefMatrix, float constTermsMatrix |
| 6 | float determinantX2 | Знайти показник матриці з заміною другого рядка | float coefMatrix, float constTermsMatrix |
| 7 | float determinantX3 | Знайти показник матриці з заміною третього рядка | float coefMatrix, float constTermsMatrix |
| 8 | bool converge | Умова закінчення | double xk, double xkp, int n, double eps |
| 9 | double okr | Перевірка погрішності | double x, double eps |
| 10 | bool diagonal | Перевірка умови діагоналей | double a, int n |
| 11 | int zeydel | Метод Зейделя | - |
| 13 | void del | Видалення зайвої пам’яті | double A, double X, double Xk, double b, int n |
| 14 | int Jacobi | Метод Якобі | - |
| 15 | void menu | Виклик меню для вибору методу | - |

Табл 3.1

Отже, в процесі реалізації програми було реалізовано процедурне програмування, що пришвидшило виконання та оптимізувало код.

**3.3. Опис інтерфейсу**

У програмі було реалізовано консольний інтерфейс. Після запуску програми користувач має обрати розмірність матриці рис. 3.3:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3.3 – Вибір метода розв’язання СЛАР

На наступному етапі користувач обирає розмірність матриці і вводить свої елементи рис 3.4

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3.4 Введення порядку і коефіцієнтів

Далі користувач отримує відповідь та обирає чи хоче він розв’язати ще одну СЛАР рис 3.5

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3.5 – Корені системи та пропозиція розв’язання нової СЛАР

**3.4. Перевірка роботи програми**

Контрольний приклад 1(метод Гауса):Изображение выглядит как текст

Автоматически созданное описание

Результат виконання програми рис 3.6:



Рисунок 3.6 – Результат виконання програми методом Гауса

Результат розв’язання рівняння в додатку Photomath ріс 3.7:

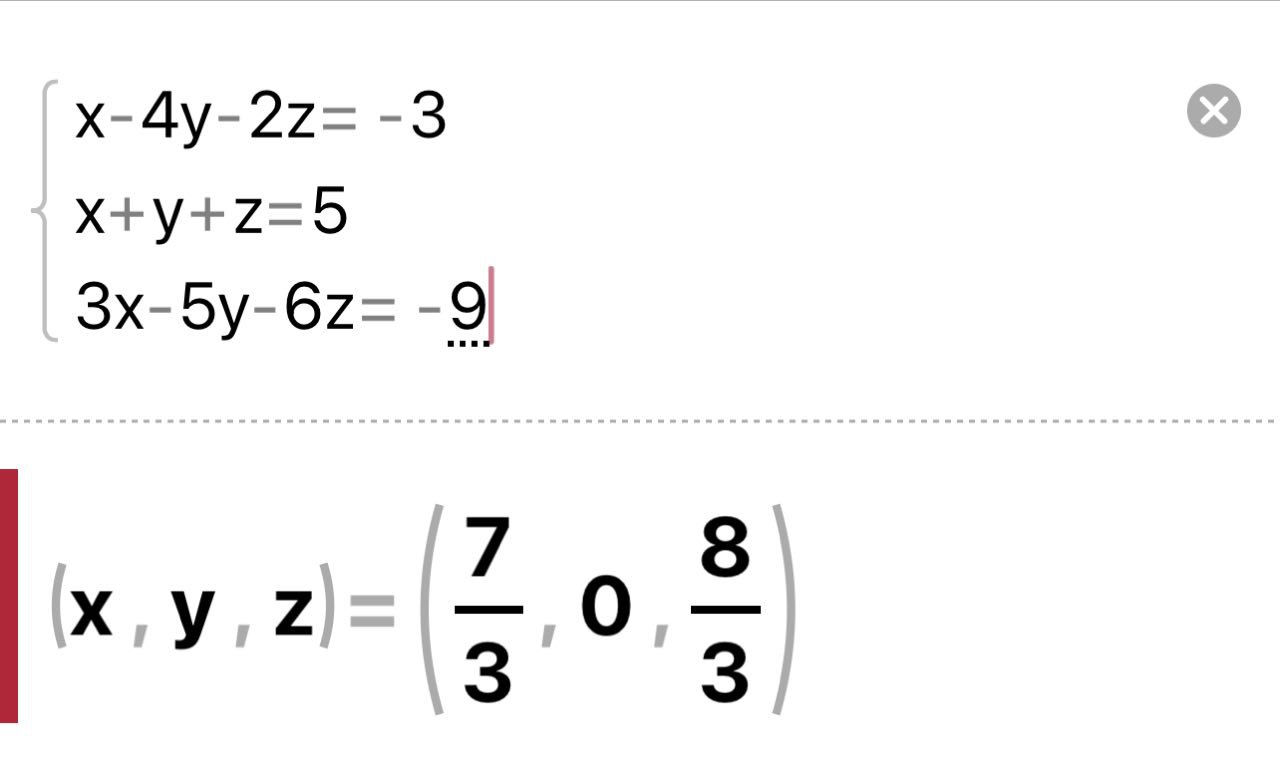
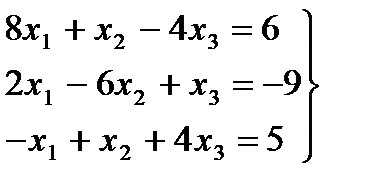


Рисунок 3.7 – Результат розв’язання рівняння в додатку Photomath

Контрольний приклад 2(метод Крамера):



Результат виконання програми рис 3.8:



Рисунок 3.8 – Результат виконання програми методом Крамера

Результат розв’язання рівняння в додатку Photomath рис 3.9:

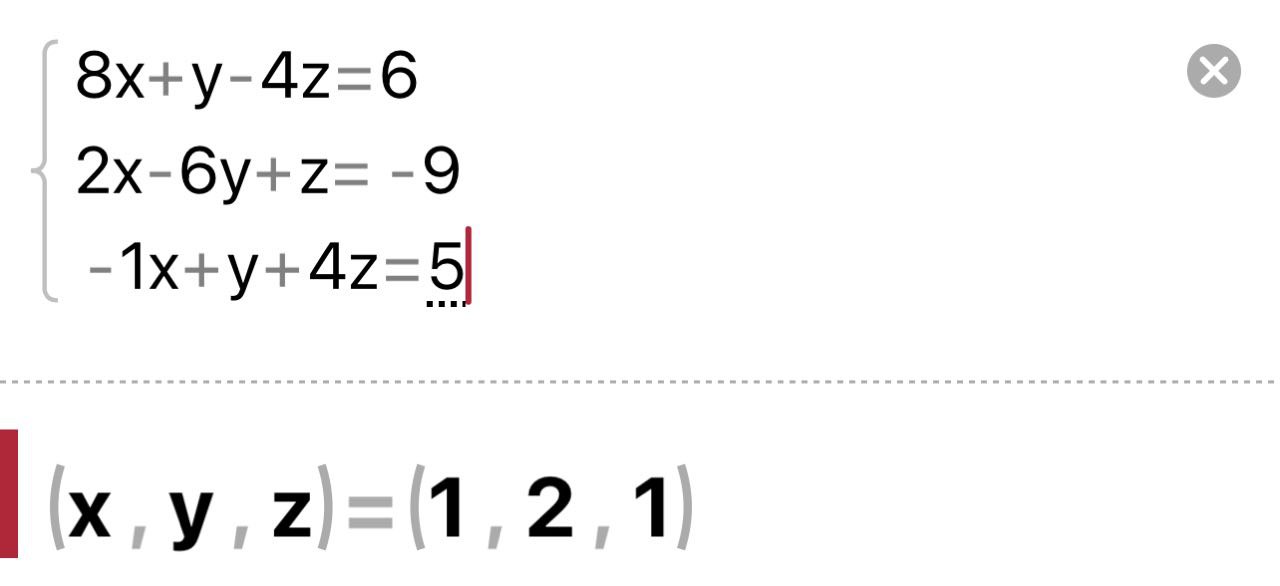
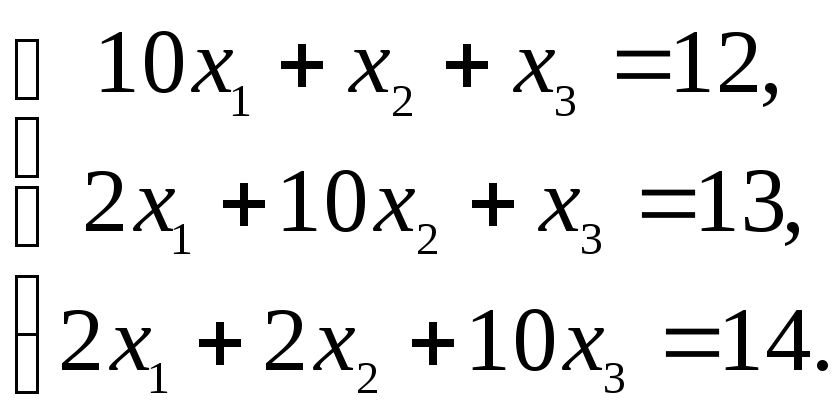


Рисунок 3.7 – Результат розв’язання рівняння в додатку Photomath

Контрольний приклад 3(метод Зейделя):



Результат виконання програми:



Рисунок 3.10 – Результат виконання програми методом Зейделя

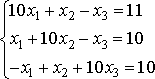
Результат розв’язання рівняння в додатку Photomath:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3.11 ­­– Результат розв’язання рівняння в додатку Photomath

Контрольний приклад 4(метод Якобі):



Результат роботи програми:



Рисунок 3.12 – Результат виконання програми методом Зейделя

Результат виконання програми в додатку Photomath:

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 3.11 ­­– Результат розв’язання рівняння в додатку Photomath

**ВИСНОВКИ**

У результаті виконання даної курсової роботи був створений додаток, у ході реалізації якої було вивчено алгоритми роботи з масивами та матрицями в С++, використано здобуті знання із курсу Алгоритмізації та програмування, Лінійної алгебри та Математичного аналізу. Також було оформлено пояснювальну записку.

У першому розділі пояснювальної записки було оглянуто вже існуючі підходи до розв’язання поставленої задачі. Було обґрунтовано обраний варіант реалізації задачі, а також детально описано постановку задачі.

У другому розділі було описано необхідний логічний апарат для вирішення поставленої задачі. Крім цього, було представлено детальний опис загального алгоритму роботи всієї програми, з наданням схеми взаємодії модулів програми.

У третьому розділі пояснювальної записки було описано головні структури та змінні програми, функції та користувацький інтерфейс. До того ж, була перевірена коректність роботи програми.

Як і передбачалося, для вирішення поставленої задачі були використані методи процедурного програмування.

У якості інтерфейсу було створене консольне меню із простими необхідними функціями та позначеннями.

Отже, можна стверджувати що технічне завдання було повністю виконано. Подальша розробка може бути вдосконалена шляхом додавання нових методів, можливості роботи з СЛАР, що не мають єдиного розв’язку, побудова графічного інтерфейсу.

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

**КНИГИ**

1. Амосов А. А., Дубинский Ю. А., Копченова Н. П. Вычислительные методы для инженеров. — М.: Мир, 1998.
2. Н. Ш. Кремер, 2.3. «Метод Гаусса»
3. Лінійна алгебра та аналітична геометрія: Навч. посібник /В. В. Булдигін, І. В. Алєксєєва, В. О. Гайдей, О. О. Диховичний, Н. Р. Коновалова, Л. Б. Федорова; за ред. проф. В. В. Булдигіна. — К. :ТВіМС, 2011.
4. Prinz, Peter. A complete guide to programming in C++ / Peter Prinz, Ulla Kirch-Prinz; translated by Ian Travis. ISBN: 0-7637-1817-3. C++ (Computer program language) I. Kirch-Prinz, Ulla. II. Title.
5. Мальцев И. А. Основы линейной алгебры. — Изд. 3-е, перераб., М.: «Наука», 1970

**ЕЛЕКТРОННІ РЕСУРСИ**

1. https://www.maa.org/press/periodicals/loci/joma/iterative-methods-for-solving-iaxi-ibi-gauss-seidel-method
2. <http://mathworld.wolfram.com/Gauss-SeidelMethod.html>
3. https://function-x.ru/cpp\_metod\_kramera.html
4. https://function-x.ru/cpp\_metod\_gaussa.
5. http://old.exponenta.ru/educat/class/courses/vvm/theme\_6/theme\_ex6.asp
6. https://www.cyberforum.ru/cpp-beginners/thread605734.html
7. https://pro-prof.com/forums/topic/openmp-метод-якоби-решения-слау
8. https://studfile.net/preview/2377683/page:4/
9. https://www.youtube.com/watch?v=SbZ1iluWmCU
10. https://habr.com/ru/post/139198/

**ДОДАТОК**

#include <iostream>

using namespace std;

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

void restart();

void Gauss()

{int n, i, j, k;

float d, s;

setlocale(LC\_CTYPE, "ukr");

cout << "Порядок:" << endl;

cin >> n;

float \*\*a = new float \*[n];

for (i = 0; i <= n; i++)

a [i] = new float [n];

float \*\*a1 = new float \*[n];

for (i = 0; i <= n; i++)

a1[i] = new float [n];

float \*b = new float [n];

float \*x = new float [n];

cout << "Введіть коефіціенти і вільні члени " << endl;

for (i = 1; i <= n; i++)

{

for (j = 1; j <= n; j++)

{

cout << "a[" << i << "," << j << "]= ";

cin >> a[i][j];

a1[i][j] = a[i][j];

}

cout << "b[" << i <<"]= ";

cin >> b[i];

}

for (k = 1; k <= n; k++)

{

for (j = k + 1; j <= n; j++)

{

d = a[j][k] / a[k][k];

for (i = k; i <= n; i++)

{

a[j][i] = a[j][i] - d \* a[k][i];

}

b[j] = b[j] - d \* b[k];

}

}

for (k = n; k >= 1; k--)

{

d = 0;

for (j = k + 1; j <= n; j++)

{

s = a[k][j] \* x[j];

d = d + s;

}

x[k] = (b[k] - d) / a[k][k];

}

cout << "Корені системи: " << endl;

for( i = 1; i <= n; i++)

cout << "x[" << i << "]=" << x[i] << " " << endl;

restart();

}

void Cramer()

{

float determinant(float matrix[3][3]);

float determinantX1(float coefMatrix[3][3], float constTermsMatrix[3]);

float determinantX2(float coefMatrix[3][3], float constTermsMatrix[3]);

float determinantX3(float coefMatrix[3][3], float constTermsMatrix[3]);

int i, j;

float coefficientsMatrix3x3[3][3];

float constantTermsMatrix3x1[3];

setlocale(LC\_CTYPE, "ukr");

cout << "Введіть коефіціенти і вільні члени " << endl;

for (i = 0; i < 3; i++)

{

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

{

cout << "a[" << i << "," << j << "]= ";

cin >> coefficientsMatrix3x3[i][j];

}

cout << "b[" << i << "]= ";

cin >> constantTermsMatrix3x1[i];

}

float det = determinant(coefficientsMatrix3x3);

float detX1 = determinantX1(coefficientsMatrix3x3, constantTermsMatrix3x1);

float detX2 = determinantX2(coefficientsMatrix3x3, constantTermsMatrix3x1);

float detX3 = determinantX3(coefficientsMatrix3x3, constantTermsMatrix3x1);

if (det != 0)

{

cout << "X1 = " << (float)detX1/(float)det << endl;

cout << "X2 = " << (float)detX2/(float)det << endl;

cout << "X3 = " << (float)detX3/(float)det << endl;

}

else

cout << "Система не має розв’язків " << endl << endl;

restart();

}

float determinant(float matrix[3][3])

{

float a11 = matrix[0][0];

float a12 = matrix[0][1];

float a13 = matrix[0][2];

float a21 = matrix[1][0];

float a22 = matrix[1][1];

float a23 = matrix[1][2];

float a31 = matrix[2][0];

float a32 = matrix[2][1];

float a33 = matrix[2][2];

return (a11 \* a22 \* a33) + (a12 \* a23 \* a31) + (a13 \* a21 \* a32) -

(a13 \* a22 \* a31) - (a11 \* a23 \* a32) - (a12 \* a21 \* a33);

}

float determinantX1(float coefMatrix[3][3], float constTermsMatrix[3])

{

float a12 = coefMatrix[0][1];

float a13 = coefMatrix[0][2];

float a22 = coefMatrix[1][1];

float a23 = coefMatrix[1][2];

float a32 = coefMatrix[2][1];

float a33 = coefMatrix[2][2];

float c1 = constTermsMatrix[0];

float c2 = constTermsMatrix[1];

float c3 = constTermsMatrix[2];

return (c1 \* a22 \* a33) + (a12 \* a23 \* c3) + (a13 \* c2 \* a32) -

(a13 \* a22 \* c3) - (c1 \* a23 \* a32) - (a12 \* c2 \* a33);

}

float determinantX2(float coefMatrix[3][3], float constTermsMatrix[3])

{

float a11 = coefMatrix[0][0];

float a13 = coefMatrix[0][2];

float a21 = coefMatrix[1][0];

float a23 = coefMatrix[1][2];

float a31 = coefMatrix[2][0];

float a33 = coefMatrix[2][2];

float c1 = constTermsMatrix[0];

float c2 = constTermsMatrix[1];

float c3 = constTermsMatrix[2];

return (a11 \* c2 \* a33) + (c1 \* a23 \* a31) + (a13 \* a21 \* c3) -

(a13 \* c2 \* a31) - (a11 \* a23 \* c3) - (c1 \* a21 \* a33);

}

float determinantX3(float coefMatrix[3][3], float constTermsMatrix[3])

{

float a11 = coefMatrix[0][0];

float a12 = coefMatrix[0][1];

float a21 = coefMatrix[1][0];

float a22 = coefMatrix[1][1];

float a31 = coefMatrix[2][0];

float a32 = coefMatrix[2][1];

float c1 = constTermsMatrix[0];

float c2 = constTermsMatrix[1];

float c3 = constTermsMatrix[2];

return (a11 \* a22 \* c3) + (a12 \* c2 \* a31) + (c1 \* a21 \* a32) -

(c1 \* a22 \* a31) - (a11 \* c2 \* a32) - (a12 \* a21 \* c3);

}

bool converge(double xk[10], double xkp[10], int n, double eps)

{

double norm = 0;

for (int i = 0; i < n; i++)

norm += (xk[i] - xkp[i]) \* (xk[i] - xkp[i]);

return (sqrt(norm) < eps);

}

double okr(double x, double eps)

{

int i = 0;

double neweps = eps;

while (neweps < 1)

{

i++;

neweps \*= 10;

}

int okr = pow(double(10), i);

x = int(x \* okr + 0.5) / double(okr);

return x;

}

bool diagonal(double a[10][10], int n)

{

int i, j, k = 1;

double sum;

for (i = 0; i < n; i++) {

sum = 0;

for (j = 0; j < n; j++) sum += abs(a[i][j]);

sum -= abs(a[i][i]);

if (sum > a[i][i])

{

k = 0;

cout << a[i][i] << " < " << sum << endl;

}

else

{

cout << a[i][i] << " > " << sum << endl;

}

}

return (k == 1);

}

void zeydel()

{

setlocale(LC\_ALL, "");

double eps, a[10][10], b[10], x[10], p[10];

int n, i, j, m = 0;

int method;

cout << "Введіть порядок: ";

cin >> n;

eps = 7;

cout << "Заповніть матрицю А: " << endl << endl;

for (i = 0; i < n; i++)

for (j = 0; j < n; j++)

{

cout << "A[" << i << "][" << j << "] = ";

cin >> a[i][j];

}

cout << endl << endl;

cout << "Ваша матриця А: " << endl << endl;

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

cout << a[i][j] << " ";

cout << endl;

}

cout << endl;

cout << "Введіть вільні члени: " << endl << endl;

for (i = 0; i < n; i++)

{

cout << "В[" << i + 1 << "] = ";

cin >> b[i];

}

cout << endl << endl;

for (int i = 0; i < n; i++)

x[i] = 1;

if (diagonal(a, n)) {

do

{

for (int i = 0; i < n; i++)

p[i] = x[i];

for (int i = 0; i < n; i++)

{

double var = 0;

for (int j = 0; j < n; j++)

if(j!=i) var += (a[i][j] \* x[j]);

x[i] = (b[i] - var) / a[i][i];

}

m++;

}

while (!converge(x, p, n, eps));

cout << "Корені системи:" << endl << endl;

for (i = 0; i < n; i++) cout << "x" << i << " = " << okr(x[i], eps) << "" << endl;

cout << "Ітерацій: " << m << endl;

}

else {

cout << "Не выполняется умова діагоналей" << endl;

}

restart();

}

void del(double \*\*A, double \*X, double \*Xk, double \*b, int n);

void jacobi()

{

int n;

int i,j;

double \*\*A;

double \*X;

double \*Xk;

double \*b;

double sumd = 0;

double eps;

printf("Введіть порядок: \n");

cin>>n;

A = new double\* [n];

X = new double [n];

Xk = new double [n];

b = new double [n];

for(i = 0; i < n; i++)

A[i] = new double [n];

printf("Введіть елементи системи для системи: \n");

for(i = 0; i < n; i++)

for(j = 0; j < n; j++)

scanf("%lf",&A[i][j]);

printf("Введіть вільні члени: \n");

for(i = 0; i < n; i++)

scanf("%lf",&b[i]);

printf("Задайте точність: \n");

scanf(" %lf",&eps);

for(i = 0; i < n; i++)

if (A[i][i] == 0) {del( A, X, Xk, b, n); printf("Головна діагональ обнулена \n"); restart();}

for(i = 0; i < n; i++)

{

for(j = 0; j < n; j++)

if (i != j) sumd += fabs(A[i][j]);

if (fabs(A[i][i]) < sumd) {del( A, X, Xk, b, n); printf("Даний метод не підходить для цієї системи \n"); restart();}

sumd = 0;

}

for(i = 0; i < n; i++)

X[i]=0.0;

int count = 0, flag = 1;

double x, zh = 0;

for(i = 0; i < n; i++)

Xk[i] = X[i];

do

{

count++;

for(i = 0; i < n; i++)

{

x=0;

for(j = 0; j < n; j++)

{

if (i != j) x += Xk[j] \* A[i][j];

if (i == j) zh = A[i][j];

}

x = (b[i] - x) / zh;

X[i] = x;

if ((fabs(X[i]-Xk[i]))<=eps) flag=0;

}

for(i = 0; i < n; i++)

Xk[i] = X[i];

}while(flag);

printf("Корені системи\n");

for(i = 0; i < n; i++)

printf("x[%d]=%0.5lf\n",i+1,X[i]);

printf("iteration=%d\n",count);

del( A, X, Xk, b, n);

restart();

}

void del(double \*\*A, double \*X, double \*Xk, double \*b, int n)

{

for(int i = 0; i < n; i++)

delete[]A[i];

delete[]A;

delete[]X;

delete[]Xk;

delete[]b;

}

void menu()

{

int r;

cout<<"Виберіть потрібний вам метод" <<endl<<"1 - Метод Гаусса"<<endl<<"2 - Метод Крамера"<<endl<<"3 - Метод Зейделя"<<endl<<"4 - Метод Якобі"<<endl;

cin>>r;

if (r==1)

Gauss();

else if(r==2)

Cramer();

else if(r==3)

zeydel();

else if(r==4)

jacobi();

else

cout<<"Неприпустимий варіант"<<endl;

}

void restart()

{

int restart;

cout<<"Вам потрібно розв’язати ще одну СЛАР?"<<endl<<"1 - так"<<endl<<"0 - ні"<<endl;

cin>>restart;

if(restart)

menu();

else

exit(0);

}

int main()

{

menu();

return 0;

}