**Міністерство освіти і науки України**

**Національний технічний університет України**

**«Київський політехнічний інститут»**

**Основи програмування та алгоритмічної мови – 2.**

**Спеціальні засоби мови програмування.**

**Курсова робота**

**«Рішення системи лінійних рівнянь методом Гауса-Жордана»**

|  |  |
| --- | --- |
| Дата : «»\_\_\_\_\_\_2015р. | Виконав студент І курсу групи ТВ |
|  | Коренівський Павло Миколайович |
| Оцінка : «\_\_\_\_\_\_\_\_\_\_» | Перевірив: Крячок Олександр Степанович |
| Дата : «»\_\_\_\_\_\_2015р. |  |

1. Зміст
2. Введення
3. Загальний розділ
   1. Методи розв'язання
   2. Технічні засоби реалізації
4. Спеціальний розділ
   1. Постановка задачі
   2. Техніко математичний опис задачі
   3. Опис алгоритму
   4. Опис реалізації програми
      1. Детальний опис класів
5. Висновок
6. Список довідкової літератури

2. Введення

Системи лінійних алгебраїчних рівнянь відіграють важливу роль у математиці, оскільки до них зводиться велика кількість задач лінійної алгебри, теорії диференціальних рівнянь, математичної фізики тощо, та областей фізики й техніки, де застосовуються ці математичні теорії.

Систему лінійних алгебраїчних рівнянь можна записати у вигляді, де кожна невідома є ваговим коефіцієнтом в лінійній комбінації вектор-стовпців.


 x_1 \begin{bmatrix}a_{11}\\a_{21}\\ \vdots \\a_{m1}\end{bmatrix} +
 x_2 \begin{bmatrix}a_{12}\\a_{22}\\ \vdots \\a_{m2}\end{bmatrix} +
 \cdots +
 x_n \begin{bmatrix}a_{1n}\\a_{2n}\\ \vdots \\a_{mn}\end{bmatrix}
 =
 \begin{bmatrix}b_1\\b_2\\ \vdots \\b_m\end{bmatrix}


Що дозволяє переформулювати задачу в термінах векторного простору: рівняння має розв'язок тоді і тільки тоді, коли лінійна комбінація (лінійна оболонка) векторів лівої частини включає вектор правої частини.

Векторна форма еквівалентна матричній формі запису

**Ax = b**

де A — матриця m×n, x — вектор з n компонент, b — вектор з m компонент.

A=
\begin{bmatrix}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{m1} & a_{m2} & \cdots & a_{mn}
\end{bmatrix},\quad
\bold{x}=
\begin{bmatrix}
x_1 \\ x_2 \\ \vdots \\ x_n
\end{bmatrix},\quad
\bold{b}=
\begin{bmatrix}
b_1 \\ b_2 \\ \vdots \\ b_m
\end{bmatrix}


Число векторів в базисі лінійної оболонки векторів є рангом матриці.

Метою цієї роботи є огляд основних методів розв’язання систем лінійних рівнянь, та програмна реалізація метода **Гауса-Жордана**. Початкові дані являюсь собою коефіцієнти при невідомих та вільні члени. Вільні члене та коефіцієнти при невідомих є головними даними, які програма зберігає у пам’яті.

Рішення систем лінійних рівнянь є однією з важливих обчислювальних завдань. Більшість задач обчислювальної практики зводяться до вирішення систем лінійних рівнянь. Це завдання з області електротехніки, радіоелектроніки, механіки, статистики. Практичні завдання часто призводять до таких систем, які містять сотні і навіть тисячі лінійних рівнянь. Без допомоги комп'ютера, ці системи вирішити дуже складно, а в системах реального часу (або наближених до таких) – це просто неможливо.

3. Загальний розділ

3.1 Методи розв’язання

* **Метод послідовного виключення**. Найпростішим, хоча важким для практичних застосувань, методом розв'язування системи лінійних алгебраїчних рівнянь є метод послідовного виключення невідомих. Суть його в тому, що із першого рівняння змінна x1 виражається через інші змінні, й підставляється в усі інші рівняння. Це можна зробити, якщо коефіцієнт a11 відмінний від нуля. У випадку, якщо він нульовий, можна вибрати інше рівняння, оскільки перестановка рівнянь у системі дає еквівалентну систему. В результаті утворюється нова система рівнянь, в якій рівнянь на одне менше. З цією системою рівнянь можна поступити так само, отримуючи ще меншу систему рівнянь. Продовжуючи так, отримують одне лінійне рівняння, з якого можна визначити одну із змінних, а інші, виключені, виразити через неї.
* **Метод Гауса** — метод, найчастіше застосовуваний при ручному розв'язку СЛАР.
* **Метод Гауса-Жордана** - модифікація методу Гауса.
* **Метод Крамера** (за формулами Крамера) — чисто теоретичний метод, непридатний до практичного використання через обчислювальну складність і малу точність, оскільки вимагає обчислення визначників, а тільки в одному визначнику n! доданків. Метод Крамера може застосовуватися для матриць 2×2, або, щонайбільше, 3×3.

3.2 Засоби реалізації

C++ (Сі-плюс-плюс) — мова програмування високого рівня з підтримкою декількох парадигм програмування: об'єктно-орієнтованої, узагальненої та процедурної. Розроблена Б'ярном Страуструпом (англ. Bjarne Stroustrup) в AT&T Bell Laboratories (Мюррей-Хілл, Нью-Джерсі) у 1979 році та початково отримала назву «Сі з класами». Згодом Страуструп перейменував мову у C++ у 1983 р. Базується на мові С. Визначена стандартом ISO/IEC 14882:2003.[1]

У 1990-х роках С++ стала однією з найуживаніших мов програмування загального призначення. Мову використовують для системного програмування, розробки програмного забезпечення, написання драйверів, потужних серверних та клієнтських програм, а також для розробки розважальних програм таких як відеоігри. С++ суттєво вплинула на інші, популярні сьогодні, мови програмування: С# та Java.

Стандартна бібліотека Сі++ (STL) включає стандартну бібліотеку Сі з невеликими змінами, які роблять її відповідною для мови Сі++. Інша велика частина бібліотеки Сі++ заснована на Стандартній Бібліотеці Шаблонів (STL). Вона надає такі важливі інструменти, як контейнери (наприклад, вектори і списки) і ітератори (узагальнені вказівники), що надають доступ до цих контейнерів як до масивів. Крім того, STL дозволяє схожим чином працювати і з іншими типами контейнерів, наприклад, асоціативними списками, стеками, чергами. Використовуючи шаблони, можна писати узагальнені алгоритми, здатні працювати з будь-якими контейнерами або послідовностями, доступ до членів яких забезпечують ітератори.

Так само, як і в Сі, можливості бібліотек активізуються використанням директиви #include для включення стандартних файлів.

STL до включення в стандарт Сі++ була сторонньою розробкою, на початку — фірми HP, а потім SGI. Стандарт мови не називає її «STL», оскільки ця бібліотека стала невід'ємною частиною мови, проте багато людей досі використовують цю назву, щоб відрізняти її від решти частини стандартної бібліотеки (потоки введення/виведення (Iostream), підрозділ Сі тощо). Проект під назвою STLport, заснований на SGI STL, здійснює постійне оновлення STL, IOstream і рядкових класів. Деякі інші проекти також займаються розробкою приватних застосувань стандартної бібліотеки для різних конструкторських завдань. Кожен виробник компіляторів Сі++ обов'язково поставляє якусь реалізацію цієї бібліотеки, оскільки вона є дуже важливою частиною стандарту і широко використовується.

Мова Сі++ багато в чому є надмножиною Сі. Нові можливості Сі++ включають оголошення у вигляді виразів, перетворення типів у вигляді функцій, оператори new і delete, тип bool, посилання, розширене поняття константності та змінності, функції, що підставляються, аргументи за замовчанням, перевизначення, простори імен, класи (включаючи і всі пов'язані з класами можливості, такі як успадкування, функції-члени (методи), віртуальні функції, абстрактні класи і конструктори), перевизначення операторів, шаблони, оператор ::, обробку винятків, динамічну ідентифікацію і багато що інше. Сі++ є також мовою строгого типування і накладає більше вимог щодо дотримання типів, порівняно з Сі.

4. Спеціальний розділ

4.1 Постановка задачі

Розробити програму для рішення системи лінійних рівнянь порядку N методом **Жордана-Гауса**.

Програму необхідно реалізувати на мові програмування С++ з використанням об’єктно орієнтованої парадигми.

В якості середовища розробки використовувати IDE Microsoft Visual Studio версії не нижче 2012.

Програма повинна функціонувати на платформі Windows. Проте, бажано якомога менше використовувати залежних від платформи розширень мови С++. В якості допоміжних бібліотек використати бібліотеку STL.

Програма повинна складатися з окремих модулів. Необхідно розділити модуль взаємодії з користувачем та модуль обчислення.

Модуль взаємодії з користувачем виконати по типу консольної програми.

Модуль обчислення реалізувати у вигляді окремих класів, які, при мінімальній модифікації, можливо буде використати у будь якій іншій програмі.

Програма повинна надати користувачу вибір у який спосіб будуть введені вхідні дані для обчислень – з клавіатури або з заздалегідь підготовленого файлу. Файл являє собою текстовий файл, в якому в кожному рядку через пробіл внесені коефіцієнти при невідомих та вільні члени.

Результат роботи програми виводиться на монітор, а також, по запиту користувача зберігаються у файл.

4.2 Техніко математичний опис задачі

Розглянемо алгоритм рішення системи лінійних рівнянь


  A=\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 
    a_{21} & a_{22} & \cdots & a_{2n} \\ 
    \vdots & \vdots & \ddots & \vdots \\ 
    a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} 
  \quad a_{ii} \ne 0 \quad 
  I=\begin{pmatrix} 1 & 0 & \cdots & 0 \\
    0 & 1 & \cdots & 0 \\
    \vdots & \vdots & \ddots & \vdots \\
    0 & 0 & \cdots & 1 \end{pmatrix}


**Прямий хід (алгоритм утворення нулів під головною діагоналлю) :**

* Поділимо перший рядок матриці А на a_{11} отримаємо: a_{1j}^1 = \frac{a_{1j} }{a_{11} }, j – стовпець матриці А.
* Повторюємо дії для матриці I , за формулою: b_{1s}^1 = \frac{b_{1s} }{a_{11} }, s – стовпець матриці I

Отримаємо :


  A=\begin{pmatrix} 1 & a_{12}^1 & \cdots & a_{1n}^1 \\ 
    a_{21} & a_{22} & \cdots & a_{2n} \\ 
    \vdots & \vdots & \ddots & \vdots \\ 
    a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} 
  \qquad I=\begin{pmatrix} b_{11}^1 & 0 & \cdots & 0 \\ 
    0 & 1 & \cdots & 0 \\ 
    \vdots & \vdots & \ddots & \vdots \\ 
    0 & 0 & \cdots & 1 \end{pmatrix}


* Будемо утворювати 0 у першому стовбці :  a_{2j}^1=a_{2j}-a_{1j}^1 a_{21} \; \dots \; a_{nj}^1=a_{nj}-a_{1j}^1 a_{n1}
* Повторюємо дії для матриці І, за формулами : b_{2s}^1=b_{2s}-b_{1s}^1 a_{21} \; \dots \; b_{ns}^1=b_{ns}-b_{1s}^1 a_{n1}

Отримаємо :


  A=\begin{pmatrix} 1 & a_{12}^1 & \cdots & a_{1n}^1 \\ 
    0 & 1 & \cdots & a_{2n}^2 \\ \vdots & \vdots & \ddots & \vdots \\ 
    0 & 0 & \cdots & 1 \end{pmatrix} 
  \qquad I=
  \begin{pmatrix} b_{11}^1 & 0 & \cdots & 0 \\ 
    b_{21}^2 & b_{22}^2 & \cdots & 0 \\ 
    \vdots & \vdots & \ddots & \vdots \\ 
    b_{n1}^n & b_{n2}^n & \cdots & b_{nn}^n 
  \end{pmatrix}


* Продовжуємо виконувати аналогічні операції використовуючи формули : a_{ij}^k=\frac{a_{ij}^k}{a_{ii} } \qquad a_{ij}^k=a_{ij}^{k-1}-a_{kj}^k a_{ik}^{k-1}

при умові, що k = 1 \to n,i = k + 1 \to n,j = 1 \to n

* Повторюємо дії для матриці І, за формулами : b_{ik}^k=\frac{b_{ik}^k}{a_{ii} } \qquad b_{is}^k=b_{is}^{k-1}-b_{ks}^k a_{ik}^{k-1}

при умові, що k=1 \to n,\; i=k+1 \to n,\; s=1 \to n

Отримаємо :


  A=\begin{pmatrix} 1 & a_{12}^1 & \cdots & a_{1n}^1 \\ 
    0 & 1 & \cdots & a_{2n}^2 \\ \vdots & \vdots & \ddots & \vdots \\ 
    0 & 0 & \cdots & 1 \end{pmatrix} 
  \qquad I=
  \begin{pmatrix} b_{11}^1 & 0 & \cdots & 0 \\ 
    b_{21}^2 & b_{22}^2 & \cdots & 0 \\ 
    \vdots & \vdots & \ddots & \vdots \\ 
    b_{n1}^n & b_{n2}^n & \cdots & b_{nn}^n 
  \end{pmatrix}


**Зворотній хід (алгоритм утворення нулів над головною діагоналлю) :**

* Використаємо формулу: a_{ij}^{k-1}=a_{ij}^{k-1}-a_{ij}^k a_{ik}^i, при умові, що k=n \to 1,\; i=1 \to k-1,\; j=1 \to n
* Повторюємо дії для матриці І, за формулою b_{is}^{k-1}=b_{is}^{k-1}-b_{is}^k a_{ik}^i: , при умові, що k=n \to 1,\; i=1 \to k-1,\; s=1 \to n

Остаточно отримуємо :


  A=\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 
    0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 
    0 & 0 & \cdots & 1 \end{pmatrix} 
  \qquad I=A^{-1}


4.3 Опис алгоритму

1. Обирається перша зліва колонка, що містить хоч одне ненульове значення.
2. Якщо верхнє число у цій колонці - нуль, то обмінюється увесь перший рядок матриці з іншим рядком матриці, де у цій колонці нема нуля.
3. Усі елементи першого рядка діляться на верхній елемент обраної колонки.
4. Від рядків, що залишились, віднімається перший рядок, помножений на перший елемент відповідного рядка, з метою отримання у якості першого елемента кожного рядка (крім першого) нуля.
5. Далі, повторюємо ці операції із матрицею, отриманою з початкової матриці після викреслювання першого рядка та першого стовпчика.
6. Після повторення операцій n-1 разів отримаємо верхню трикутну матрицю.
7. Віднімаємо від передостаннього рядка останній рядок, помножений на відповідний коефіцієнт, щоб у передостанньому рядку залишилась лише 1 на головній діагоналі.
8. Повторюємо попередній крок для наступних рядків. У результаті отримуємо одиничну матрицю і рішення на місці вільного вектора (над ним необхідно виконувати ті самі перетворення).

4.4 Опис реалізації програми

Для забезпечення роботи програми реалізуємо наступні класи

1. CBaseGauss - базовий абстрактний клас для забезпечення введення початкових даних системи лінійних рівнянь, реалізацію обчислень методом Жордана-Гауса та збереження результату роботи програми.   
   В якості сховища даних буде використано контейнери std::vector.

В якості збереження результатів обчислень буде використано контейнер std::vector.

1. CInputGauss - похідний від CBaseGauss. Цей клас дозволяє обчислити дані введені безпосередньо з клавіатури.
2. CFileGauss - похідний від CBaseGauss. Цей клас дозволяє обчислити дані зчитані з файлу

Детальний опис реалізованих класів наданий у наступному розділі.

Похідні тексти класів наведені в кінці документу.

4.4.1 Детальний опис класів

**Клас CBaseGauss.**

Базовий *абстрактний* клас для забезпечення введення початкових даних системи лінійних рівнянь, реалізацію обчислень методом Жордана-Гауса та збереження результату роботи програми.

Має стандартний конструктор, та віртуальний деструктор.

**Відкриті методи класу :**

* ***void ShowInData() const****;*

За допомогою цього методу виконується друк матриці коефіцієнтів при невідомих та вільних членів на екран монітора.

* ***void ShowResult() const;***

За допомогою цього метода виконується друк результатів обчислень на екран монітора.

* ***void SaveResult() const;***

За допомогою цього метода виконується збереження результатів обчислень у файл.

* ***const bool Calc();***

За допомогою цього метода виконуються всі необхідні обчислення. Цей метод містить у собі лише виклик захищеного віртуального методу ***calc()***.

**Абстрактні методи класу :**

* ***virtual const bool InputData() abstract;***

За допомогою цього метода виконується ввід початкових даних у програму. Цей метод не має реалізації в класі CBaseGaus, його

необхідно реалізувати в похідних від CBaseGaus класах.

**Закриті методи класу :**

* ***virtual const bool validate() const;***

За допомогою цього метода виконується перевірка вхідних даних. Метод позначений як віртуальний. За необхідності можна змінити алгоритм перевірки у похідних класах.

* ***virtual const bool calc();***

За допомогою цього класу виконується обчислення вхідних даних і отримання результату. Метод позначений як віртуальний. За необхідності можна змінити логіку обчислень у похідних класах.

**Захищені методи :**

* ***const size\_t findNonZCol(size\_t n) const;***

Це допоміжний метод, за допомогою якого знаходиться перша зліва колонка, що містить хоч одне ненульове значення.

**Клас CFileGauss**

Похідний від **CBaseGauss**. Цей клас дозволяє обчислити дані зчитані з файлу.

Має стандартний конструктор, та віртуальний деструктор.

Клас позначений як ***final*** і не може бути використаним у якості базового класу.

**Відкриті методи :**

* ***virtual const bool InputData() override final;***

Цей метод являє собою реалізацію абстрактного метода базового класу. Реалізує зчитування вхідних даних з файлу.

Метод позначений як **final** і не може бути перекритим в класах нащадках

**Клас CInputGauss**

Похідний від CBaseGauss. Цей клас дозволяє обчислити дані введені безпосередньо з клавіатури.

Має стандартний конструктор, та віртуальний деструктор.

Клас позначений як ***final*** і не може бути використаним у якості базового класу.

**Відкриті методи :**

* ***virtual const bool InputData() override final;***

Цей метод являє собою реалізацію абстрактного метода базового класу. Реалізує зчитування вхідних даних безпосередньо з клавіатури.

Метод позначений як **final** і не може бути перекритим в класах нащадках

6 Висновок

В результаті виконання цієї роботи була розроблена програма обчислення системи лінійних рівнянь методом Гауса-Жордана. Програма може працювати як самостійна система. Класи, роздроблені для програми, можуть бути використані в інших програмних модулях. Для цих класів неважко буде розробити графічний інтерфейс користувача. Також за допомогою мови C++.net можливо розробити класи-врапери для подальшого їх використання у програмах розроблених на популярних сьогодні мовах програмування для платформи Microsoft .Net.

5 Список довідкової літератури

1. Бьерн Страуструп - Язык программирования C++. Специальное издание
2. <https://uk.wikipedia.org/wiki/Метод_Гауса_—_Жордана>
3. <https://uk.wikipedia.org/wiki/Система_лінійних_алгебраїчних_рівнянь>
4. <https://msdn.microsoft.com/en-us/library/vstudio/1fe2x6kt(v=vs.120).aspx>
5. <https://msdn.microsoft.com/en-us/library/3bstk3k5(v=vs.120).aspx>
6. <https://msdn.microsoft.com/en-us/library/cscc687y(v=vs.120).aspx>