ЗМІСТ

[АНОТАЦІЯ 2](#_Toc419141654)

[ВСТУП 3](#_Toc419141655)

[Розділ 1 Аналіз технічного завдання 5](#_Toc419141656)

[Розділ 2 Обґрунтування алгоритму та структури програми 6](#_Toc419141657)

[Розділ 3 Розробка програми 8](#_Toc419141658)

[Розділ 4 Тестування програми і результати її виконання 10](#_Toc419141659)

[Висновок 12](#_Toc419141660)

[Перелік використаних джерел 13](#_Toc419141661)

[Додаток А 14](#_Toc419141662)

# АНОТАЦІЯ

Дано визначення СЛАР та описано методи її розв’язання. Застосовано уніфіковану мову моделювання (UML) для створення абстрактної моделі системи. Розроблено програму на мові програмування С++ для знаходження коренів системи лінійних алгебраїчних рівнянь методом Гауса.

# ВСТУП

Система рівнянь – це набір двох і більше рівнянь, заданих функціями багатьох змінних, які повинні задовольнятися одночасно. Розв’язком системи рівнянь називається упорядкований набір чисел - значень невідомих, при підстановці яких у кожному рівнянні системи виконується умова рівності.

Системою лінійних алгебраїчних рівнянь (СЛАР) називається система виду:

, (1)

Цю систему рівнянь можна записати у матричному вигляді:

 (2)

, , 

де  – шуканий вектор, при підстановці якого рівняння системи перетворюються у тотожність,  – відомий вектор розміру n (стовпець вільних членів),  – квадратна матриця з елементами ai,j.

Необхідною і достатньою умовою існування єдиного розв’язку СЛАР є умова , у такому випадку матриця A називається виродженою. При цьому рівняння має лише тривіальний розв’язок і система рівнянь (2) має єдиний розв’язок .

Чисельний розв’язок системи лінійних алгебраїчних рівнянь – одне з завдань, яке найбільш часто зустрічається у науково-технічних дослідженнях, математичній фізиці (чисельне рішення диференціальних та інтегральних рівнянь), економіці, статистиці. Дослідженню методів розв'язання систем лінійних алгебраїчних рівнянь приділяється велика увага в сучасній математиці. Серед вчених, наукові інтереси яких присвячені цій проблемі, можна назвати С. Годунова, В. Воєводіна, О. Островського, Дж. Форсайта, К. Молер та ін. Методам розв’язку СЛАР в сучасній обчислювальній математиці

приділяється велике значення.

Всі методи розв’язку СЛАР діляться на дві групи - точні (прямі) і ітераційні. Точні методи дозволяють отримати розв’язок СЛАР за скінченне число арифметичних операцій (метод Гауса, метод квадратного кореня, Крамара та ін.). Ідея методу Гауса полягає у зведенні СЛАР до трикутного вигляду і знаходженню коренів методом підстановки. Ефективність методів розв’язку СЛАР залежить від порядку системи і структури матриці.

Використання ітераційних методів дає можливість знайти наближене рішення системи з заданим ступенем точності (метод простої ітерації, метод Зейделя, метод послідовної релаксації).

# Розділ 1 Аналіз технічного завдання

Розроблено програму для знаходження коренів системи лінійних алгебраїчних рівнянь методом Гауса. У програмі реалізований ввід СЛАР, вивід трикутної матриці на екран та вивід розв’язку системи. Користувач має ввести кількість рівнянь системи, коефіцієнти, які стоять при невідомих змінних, і стовпець вільних членів.

Знаходження розв’язків СЛАР побудоване на ідеї послідовного виключення невідомих. Процес розв’язку СЛАР (2) або

, , (3)

за методом Гауса складається з двох етапів.

Перший етап (прямий хід). Система (3) приводиться до трикутного вигляду:

, (4)

де  – невідомі,  – відомий вектор,  – верхня трикутна матриця.

Другий етап (зворотній хід). Вектор розв’язку визначається починаючи з останнього рівняння системи (1) шляхом підстановки координат вектора невідомих, отриманих на попередньому кроці.

У програмі реалізовано клас gauss, який складається з п’яти методів. Ці методи відповідають за ввід СЛАР, вивід отриманої СЛАР на екран, вивід трикутної матриці, знаходження коренів системи та вивід коренів СЛАР.

Користувач має змогу взаємодіяти з комп’ютером через інтерфейс командного рядка.

# Розділ 2 Обґрунтування алгоритму та структури програми

Розглянемо алгоритм знаходження розв’язку СЛАР метод Гауса більш детально. Перший крок метода Гауса полягає у виключенні зі всіх рівнянь, крім першого, невідомого x1. Припустимо, що , поділимо перше рівняння (i=1) на і запишемо систему (3) в вигляді:

, ,

 (5)

 (6)

Помножимо рівняння (5) на , де i – будь-яке із чисел , і віднімемо отримане рівняння з i-го рівняння (6):

, .

Введемо позначення

, , , (7)

перепишемо отриману систему рівнянь у вигляді

, .

Перший стовпець матриці цієї системи складається з нулів, крім першого елемента, який рівний одиниці при .

Другий крок полягає у виключенні х2 з системи

(8)

Для цього поділимо перше рівняння на :





потім помножимо його на  і додамо до рівняння



У результаті отримаємо систему:

(9)

Для  отримаємо систему (n-2)-го порядку, аналогічну системі (8) (n-1)-го порядку для .

Продовжуючи міркування, після (n-1)-го кроку (тобто після виключення ) отримаємо

, або . (10)

У результаті отримаємо систему (4) з верхньою трикутною матрицею:

(11)

Зворотній хід метода Гауса полягає у визначенні всіх xi з системи (11) з верхньої трикутної матриці.

Підрахуємо кількість дій множення і ділення в методі Гауса. Розглянемо спочатку прямий хід. На першому кроці потрібно  дій ділення і множення, другий крок потребує  таких дій. Всього потрібно зробити n кроків прямого ходу, затративши на це

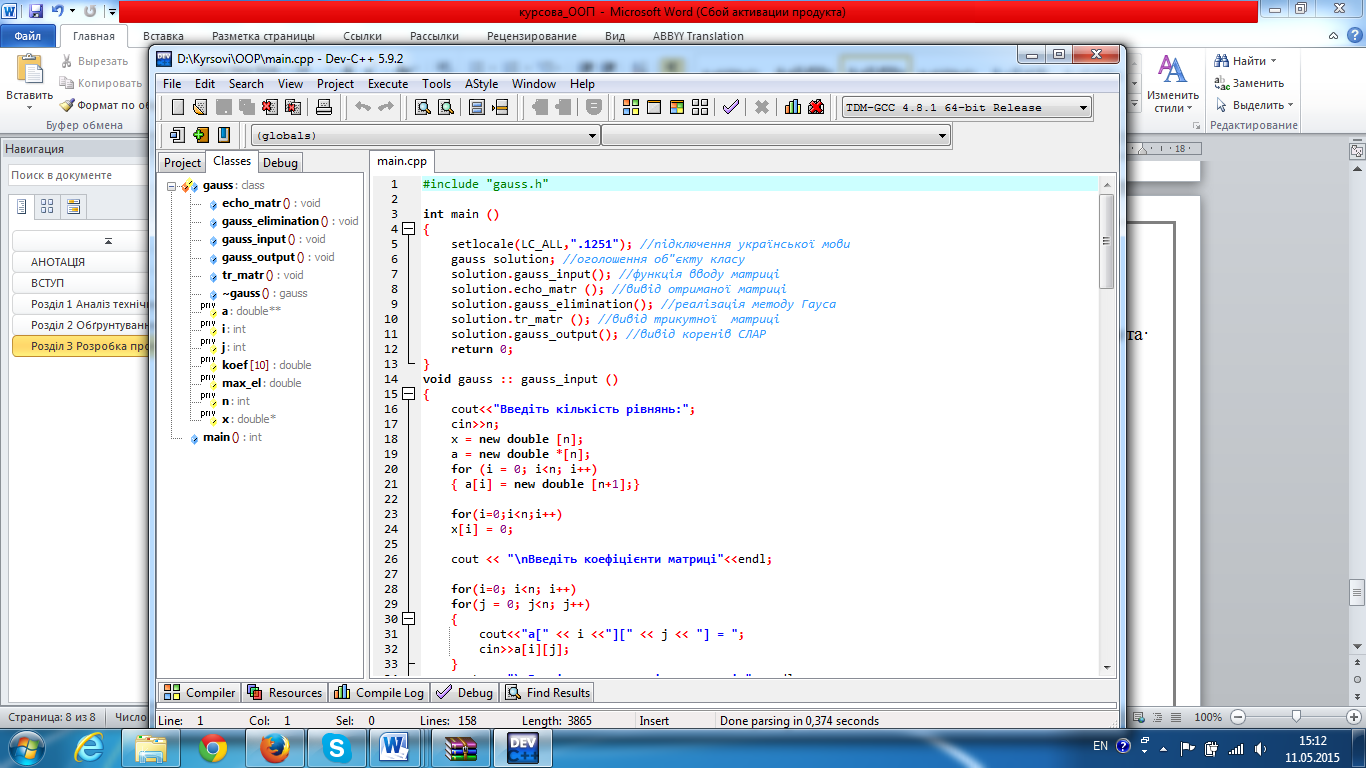


множень і ділень. Для зворотного ходу потрібно зробити n(n-1)/2 дій множення. Таким чином, для розв’язання системи рівнянь (3) потрібно  множень і ділень. Приблизно стільки ж потрібно додавань.

# Розділ 3 Розробка програми

У програмі реалізований клас Gauss, який складається з п’яти методів, семи полів та деструктора (рис.1). У класі є такі поля:

* a – двовимірний динамічний масив розміру  (розширена матриця)
* i,j використовуються у циклах
* koef[10] – масив, який використовується для збереження значень стовпця вільних членів після прямого ходу Гауса
* max\_el – максимальний елемент у стовпці
* n – кількість рівнянь
* x – динамічний масив, у якому зберігаються корені СЛАР



*Рисунок 3.1 Структура класу Gauss*

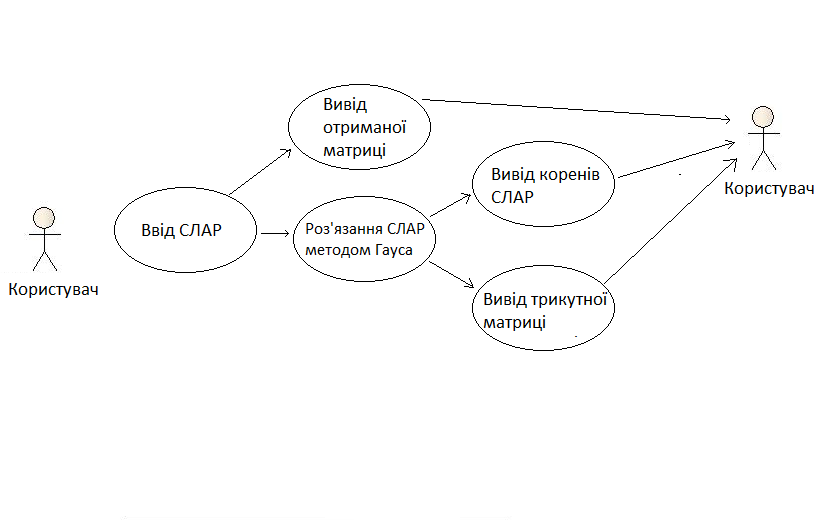
Методи класу Gauss:

* gauss\_input – ввід матриці А
* echo\_matr – вивід отриманої матриці
* gauss\_elimination – реалізація алгоритму розв’язку СЛАР методом Гауса
* gauss\_output – вивід коренів СЛАР
* tr\_matr – вивід трикутної матриці.

Також у класі реалізовано деструктор ~gauss, який використовується для звільнення пам’яті масивів a та x.

Розглянемо роботу метода gauss\_elimination. Спочатку він виконує пошук максимального елемента в стовпці та запам’ятовує рядок з цим елементом. Потім відбувається заміна рядків місцями, якщо максимальний елемент не знаходиться в поточному рядку. За допомогою ділення рядка на найбільший елемент та віднімання рядків приводимо матрицю до трикутного вигляду. Наступний кроком є знаходження коренів СЛАР зворотнім ходом. Знайдений розв’язок записуємо в масив x[].

За допомогою UML (англ. Unified Modeling Language — уніфікованої мови моделювання) побудуємо діаграму прецедентів. Для цього потрібно визначити акторів (діючі особи), прецедентів (опис окремого аспекту поведінки системи) і зв’язки.

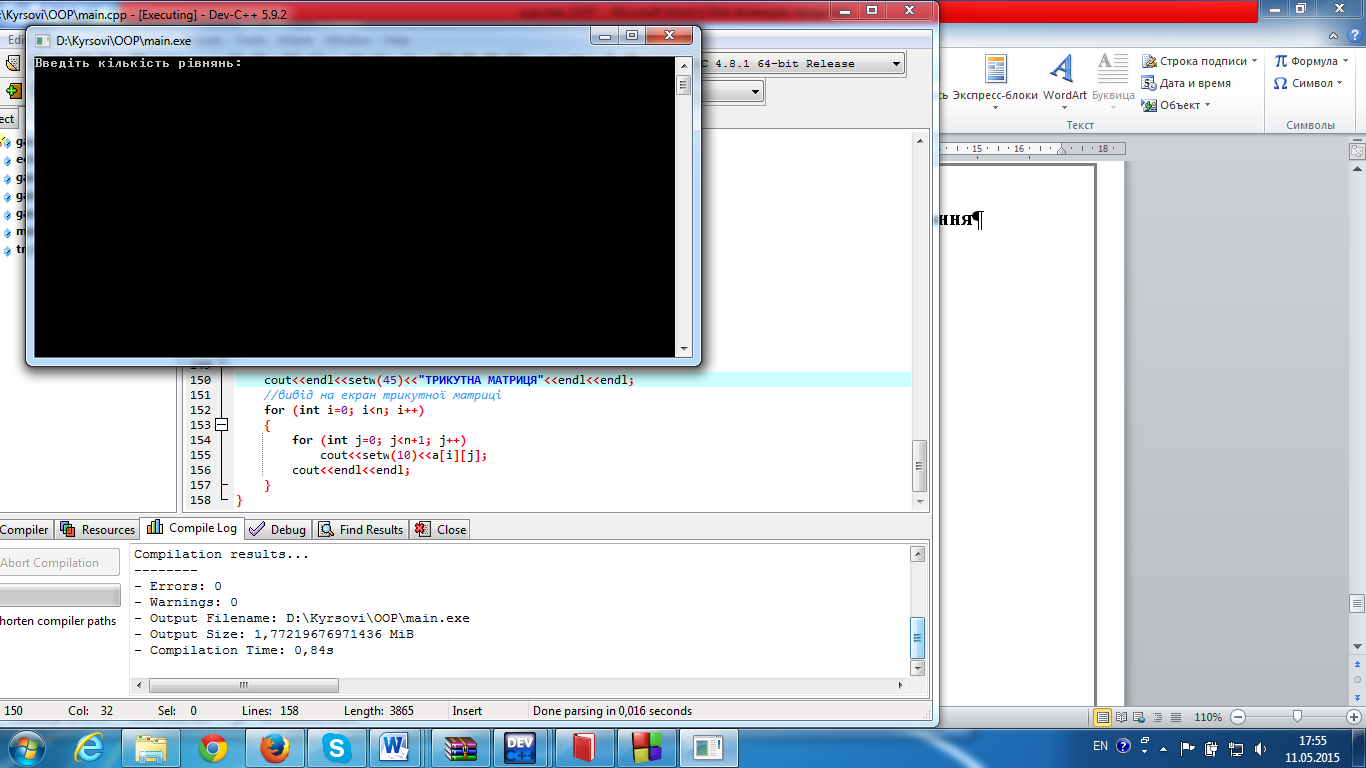


*Рисунок 3.2 Діаграма прецедентів*

У даному випадку актором буде користувач, прецедентами – ввід СЛАР, вивід отриманої матриці, розв’язання СЛАР методом Гауса, вивід коренів СЛАР і вивід трикутної матриці.

# Розділ 4 Тестування програми і результати її виконання

При запуску програми потрібно ввести кількість рівнянь системи (рис.2).



*Рисунок 4.1 Вікно запуску програми*

Далі програма запитує коефіцієнти матриці а та стовпець вільних членів.

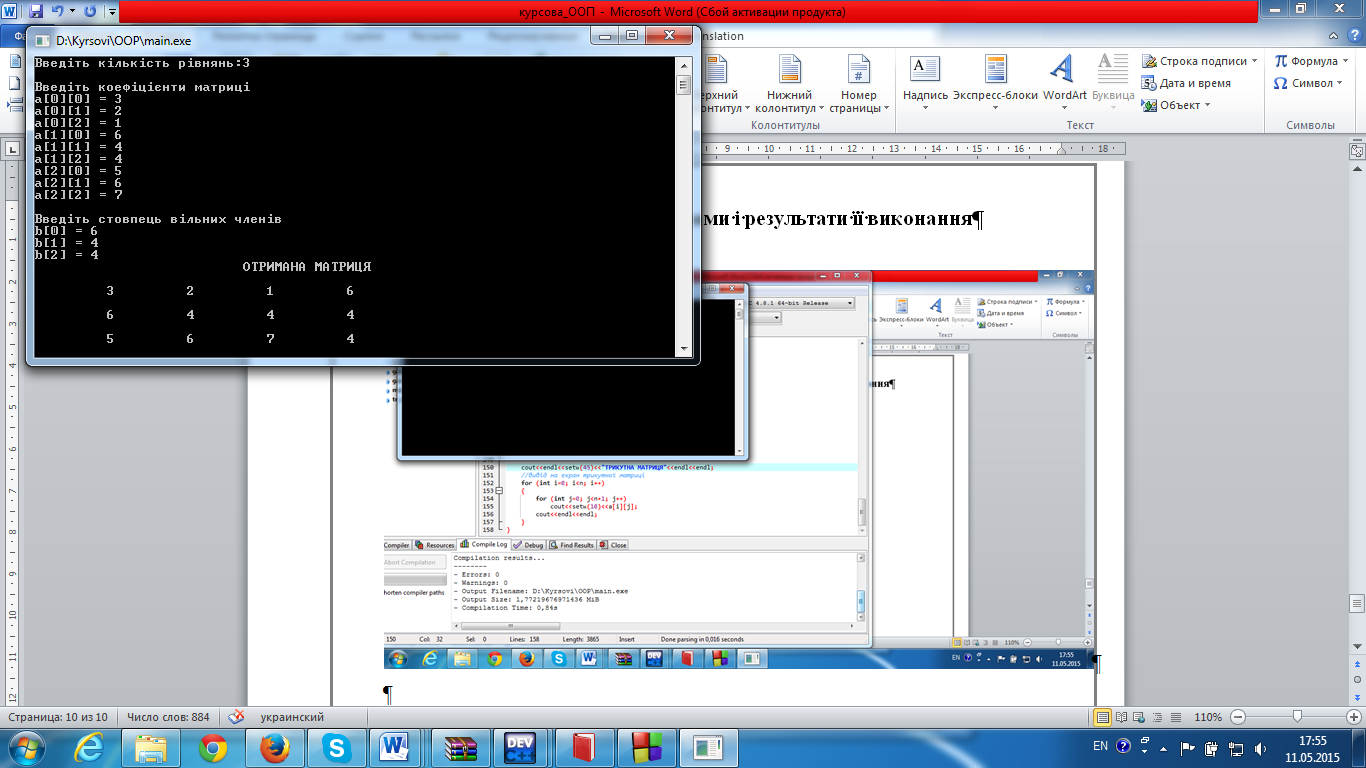
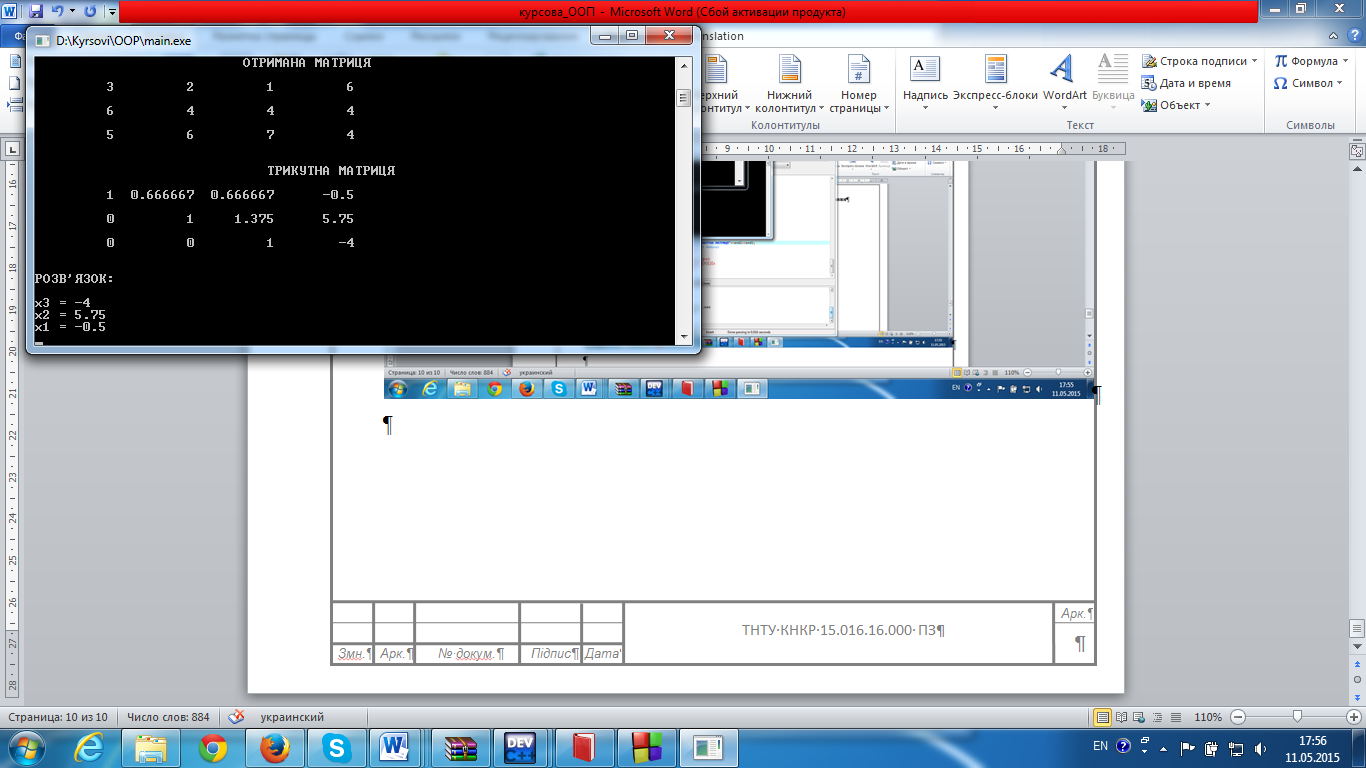


Рисунок 4.2 Ввід коефіцієнтів СЛАР

Програма виводить дві матриці: отриману розширену і трикутну.



*Рисунок 4.3 Розв’язок СЛАР*

Також на екран виведено розв’язок СЛАР.

# Висновок

Описано важливість використання СЛАР у сучасній обчислювальній математиці. Наведено методи розв’язання СЛАР. Проаналізовано та за допомогою мови програмування C++ реалізовано алгоритм знаходження коренів СЛАР методом Гауса. Побудовано діаграму прецедентів засобами UML. Представлено результати виконання програми знаходження коренів СЛАР методом Гауса.

# Перелік використаних джерел

1. Самарский А.А. Введение в численные методы: Учебн. пособие для вузов. – 2-е изд., перераб. и доп. – М.: Наука. Гл. ред. физ.-мат. лит., 1987. -288 с.
2. Дейтел Х.М. Как программировать на С++: 5-е издание. /  
    Х.М. Дейтел, П. Дж. Дейтел : Пер. с англ. – М.: ООО «Бином-Пресс», 2008. – 1456 с. ISBN 978-5-9518-0224-8.
3. Прата С. Язык программирования С++. Лекции и упражнения,  
   6-е изд. / Стивен Прата : Пер. с англ. – М.: ООО “И.Д. Вильямс”, 2012. – 1248 с. ISBN 978-5-8459-1778-2.
4. Страуструп Б. Язык программирования С++: Специальное издание. / Бьерн Страуструп. Пер. с англ. – М.: Издательство Бином, 2011. – 1136 с. ISBN 978-5-7989-0425-9.

# Додаток А

Лістинг «Gauss.h»

//заголовочний файл класу gauss.h

//інтерфейс класу

#include <iostream>

#include <math.h>

#include <process.h>

#include <conio.h>

#include <iomanip>

using namespace std;

//оголошення класу

class gauss //ім'я класу

{

private: //специфікатор доступу private

int i,j,n;

double \*x, \*\*a, max\_el, koef[10];

public://специфікатор доступу public

void gauss\_input(); //функція вводу матриці

void echo\_matr (); //вивід отриманої матриці

void gauss\_elimination(); //реалізація методу Гауса

void tr\_matr (); //вивід трикутної матриці

void gauss\_output(); //вивід коренів СЛАР

~gauss () //деструктор

{

delete [] x;

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

delete [] a[i];

}

delete [] a;

}

};//кінець оголошення класу

Лістинг «Gauss.cpp»

using namespace std;

//підключення інтерфейсу класу до його реалізації

#include "gauss.h"

void gauss :: gauss\_input ()

{

cout<<"Введiть кiлькiсть рiвнянь:";

cin>>n;

x = new double [n];

a = new double \*[n];

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

{ a[i] = new double [n+1];}

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

x[i] = 0;

cout << "\nВведiть коефiцiєнти матрицi"<<endl;

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

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

{

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

cin>>a[i][j];

}

cout << "\nВведiть стовпець вiльних членiв"<<endl;

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

{

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

cin >> a[i][n];

}

}

void gauss :: gauss\_output ()

{

//вивід коренів СЛАУ на екран

cout<<endl<<setw(5)<<"РОЗВ'ЯЗОК:"<<endl<<endl;

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

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

getch();

}

void gauss :: echo\_matr ()

{

cout<<setw(42)<<"ОТРИМАНА МАТРИЦЯ"<<endl<<endl;

//вивід на екран отриманої матриці

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

{

for (int j=0; j<n+1; j++)

cout<<setw(10)<<a[i][j];

cout<<endl<<endl;

}

}

void gauss :: gauss\_elimination ()

{

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

{

int str = i-1;

max\_el = a[i-1][i-1];

//пошук максимального елемента в стовпці

for (int ch=i-1; ch<n; ch++)

{

double buf = a[ch][i-1];

if (max\_el<0) max\_el\*=(-1);

if (buf<0) buf\*=(-1);

if (max\_el<buf)

{

max\_el = buf;

//запам"ятати строку з максимальним елементом

str = ch;

}

}

//якщо максимальний елемент нульовий - вихід

if (max\_el == 0)

{

cout<<"РОЗВ'ЯЗКУ НЕМАЄ";

getch();

}

//заміна рядків місцями

//якщо максимальний елемент не знаходиться в поточному рядку

if (str != (i-1))

{

for (int ch=0; ch<n+1; ch++)

{

double t = a[i-1][ch];

a[i-1][ch] = a[str][ch];

a[str][ch] = t;

}

}

//ділення рядка на найбільший елемент

for (int ch = 0; ch<n+1; ch++)

a[i-1][ch]/=max\_el;

//усунення нулів з головної діагоналі

if (a[i-1][i-1]<0)

for (int ch=0; ch<n+1; ch++)

{

if (a[i-1][ch]!=0)

a[i-1][ch]\*=(-1);

}

//знаходження коефіцієнтів

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

{

koef[z] = a[i+z][i-1]/a[i-1][i-1];

}

//віднімання рядків помножених на коефіцієнти

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

for (int j=i-1; j<n+1; j++)

a[k][j] -= koef[k-i]\*a[i-1][j];

}

//знаходження коренів СЛАУ зворотнім ходом

for (int i=n-1; i>=0; i--)

{

if (n-i-1 == 0 )

x[0] = (a[n-1][n])/a[n-1][n-1];

else

{

for (int j=n; j>i; j-- )

a[i][n] = a[i][n] - (a[i][j-1]\*x[n-j]);

x[n-i-1] = a[i][n]/a[i][i];

}

}

}

void gauss :: tr\_matr()

{

cout<<endl<<setw(45)<<"ТРИКУТНА МАТРИЦЯ"<<endl<<endl;

//вивід на екран трикутної матриці

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

{

for (int j=0; j<n+1; j++)

cout<<setw(10)<<a[i][j];

cout<<endl<<endl;

}

}

Лістинг «Main.cpp»

#include "gauss.h"

int main ()

{

setlocale(LC\_ALL,".1251"); //підключення української мови

gauss solution; //оголошення об"єкту класу

solution.gauss\_input(); //функція вводу матриці

solution.echo\_matr (); //вивід отриманої матриці

solution.gauss\_elimination(); //реалізація методу Гауса

solution.tr\_matr (); //вивід трикутної матриці

solution.gauss\_output(); //вивід коренів СЛАР

return 0;

}

void gauss :: gauss\_input ()

{

cout<<"Введiть кiлькiсть рiвнянь:";

cin>>n;

x = new double [n];

a = new double \*[n];

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

{ a[i] = new double [n+1];}

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

x[i] = 0;

cout << "\nВведiть коефiцiєнти матрицi"<<endl;

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

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

{

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

cin>>a[i][j];

}

cout << "\nВведiть стовпець вiльних членiв"<<endl;

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

{

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

cin >> a[i][n];

}

}

void gauss :: gauss\_output ()

{

//вивід коренів СЛАУ на екран

cout<<endl<<setw(5)<<"РОЗВ'ЯЗОК:"<<endl<<endl;

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

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

getch();

}

void gauss :: echo\_matr ()

{

cout<<setw(42)<<"ОТРИМАНА МАТРИЦЯ"<<endl<<endl;

//вивід на екран отриманої матриці

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

{

for (int j=0; j<n+1; j++)

cout<<setw(10)<<a[i][j];

cout<<endl<<endl;

}

}

void gauss :: gauss\_elimination ()

{

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

{

int str = i-1;

max\_el = a[i-1][i-1];

//пошук максимального елемента в стовпці

for (int ch=i-1; ch<n; ch++)

{

double buf = a[ch][i-1];

if (max\_el<0) max\_el\*=(-1);

if (buf<0) buf\*=(-1);

if (max\_el<buf)

{

max\_el = buf;

//запам"ятати строку з максимальним елементом

str = ch;

}

}

//якщо максимальний елемент нульовий - вихід

if (max\_el == 0)

{

cout<<"РОЗВ'ЯЗКУ НЕМАЄ";

getch();

}

//заміна рядків місцями

//якщо максимальний елемент не знаходиться в поточному рядку

if (str != (i-1))

{

for (int ch=0; ch<n+1; ch++)

{

double t = a[i-1][ch];

a[i-1][ch] = a[str][ch];

a[str][ch] = t;

}

}

//ділення рядка на найбільший елемент

for (int ch = 0; ch<n+1; ch++)

a[i-1][ch]/=max\_el;

//усунення нулів з головної діагоналі

if (a[i-1][i-1]<0)

for (int ch=0; ch<n+1; ch++)

{

if (a[i-1][ch]!=0)

a[i-1][ch]\*=(-1);

}

//знаходження коефіцієнтів

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

{

koef[z] = a[i+z][i-1]/a[i-1][i-1];

}

//віднімання рядків помножених на коефіцієнти

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

for (int j=i-1; j<n+1; j++)

a[k][j] -= koef[k-i]\*a[i-1][j];

}

//знаходження коренів СЛАР зворотнім ходом

for (int i=n-1; i>=0; i--)

{

if (n-i-1 == 0 )

x[0] = (a[n-1][n])/a[n-1][n-1];

else

{

for (int j=n; j>i; j-- )

a[i][n] = a[i][n] - (a[i][j-1]\*x[n-j]);

x[n-i-1] = a[i][n]/a[i][i];

}

}

}

void gauss :: tr\_matr()

{

cout<<endl<<setw(45)<<"ТРИКУТНА МАТРИЦЯ"<<endl<<endl;

//вивід на екран трикутної матриці

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

{

for (int j=0; j<n+1; j++)

cout<<setw(10)<<a[i][j];

cout<<endl<<endl;

}

}