|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  |  |
| МИНОБРНАУКИ РОССИИ | | | |  |
| |  | | --- | |  | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | | | |  |
|  | **Институт кибернетики**  *(наименование института)* | | | |
|  | **Кафедра высшей математики**  *(наименование кафедры)* | | | |

|  |  |
| --- | --- |
| **Отчет по лабораторной работе №2** | |
| **по дисциплине** | |
| **«** Численные методы **»** | |
| **Вариант 13** | |
| Студент 3-го курса  группы КМБО-2-16 | Савин В. О. |
| Преподаватель | Даева С.Г. |
| Рецензент |  |
|  |  |

|  |  |  |
| --- | --- | --- |
| Работа представлена к защите | «\_\_»\_\_\_\_\_\_\_2018 г. |  |
|  |  |  |
| «Допущен к защите» | «\_\_»\_\_\_\_\_\_\_2018 г. |  |

Москва 2018

Содержание

Задание №1 2

Теоретическая часть 2

Практическая часть 2

Задание №2 3

Теоретическая часть 3

Практическая часть 3

Задание №3 4

Практическая часть 5

Приложения 6

# Задание №1

Разработать программу для решения системы уравнений(СЛУ) методом Крамера.

## Теоретическая часть

В общем виде СЛУ из ***n***уравнений с ***n***неизвестными записывается в виде:

где B – вектор свободных членов, X – столбец неизвестных, а A – матрица коэффициентов системы:

Решением СЛУ называется такой набор значений *x1, x2, …, xn* при которых все уравнения системы обращаются в тождества. Если определитель матрицы не равен нулю, то СЛУ имеет единственное решение.

Для решения СЛУ можно применить метод Крамера, если система из ***n*** линейных уравнений имеет ***n*** неизвестных и определитель матрицы системы отличен от нуля. В этом случае СЛУ имеет единственное решение. После нахождения неизвестных переменных *x1, x2, …, xn*, матрица становится решением системы уравнений и равенство обращается в тождество. Определитель матрицы системы равен:

А решение самой системы записывается в виде:

**i**-ый столбец матрицы системы заменяется столбцом свободных членов.

**Практическая часть**

Для решения СЛУ методом Крамера(и Зейделя) была разработана программа «Matrix». Программа написана на языке С++ в операционной системе «Windows10» с использованием IDE «Visual Studio 2017» и компилятора «GNU GCC». Для отладки проводились тесты программы на примерах с заранее известным ответом.

*Результат работы программы размещен в задании №3. Код размещен в разделе «Приложения»*

# Задание №2

Разработать программу решения СЛУ методом Зейделя.

**Теоретическая часть**

Для решения системы:

где B – вектор свободных членов, X – столбец неизвестных, а A – матрица коэффициентов системы можно применить метод Зейделя.

Для этого необходимо преобразовать исходную систему. Разделим первое уравнение на и каждое уравнение на ,получим ***y*** неизвестных, где - единичные коэффициенты. Перенесем эти неизвестные в одну сторону от знака равенства и получим:

причем

В результате система получила новый вид:

Метод сходится независимо от выбора начального приближения, если есть какая-нибудь каноническая норма матрицыимеет значение меньше единицы, или если модули диагональных элементов матрицы *А* для каждой строки больше суммы остальных элементов строки. Итеративный цикл имеет вид:

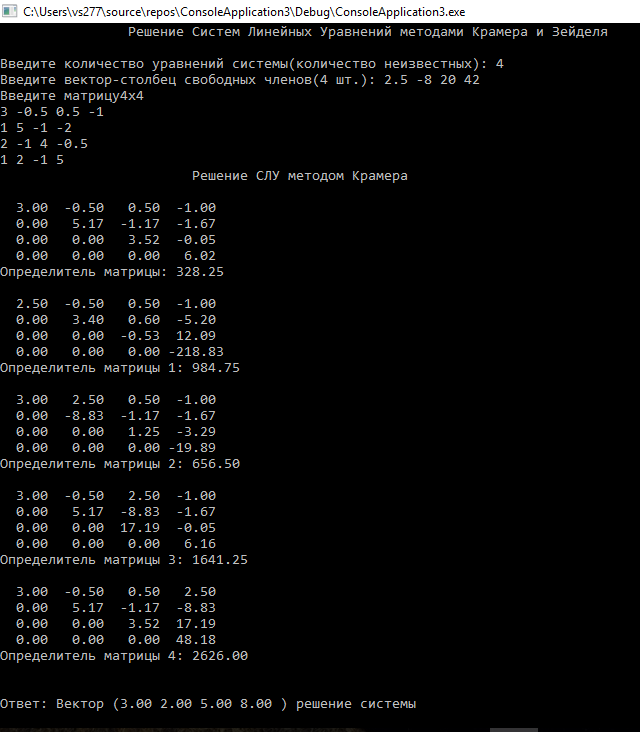
Вычисляем по следующим формулам:

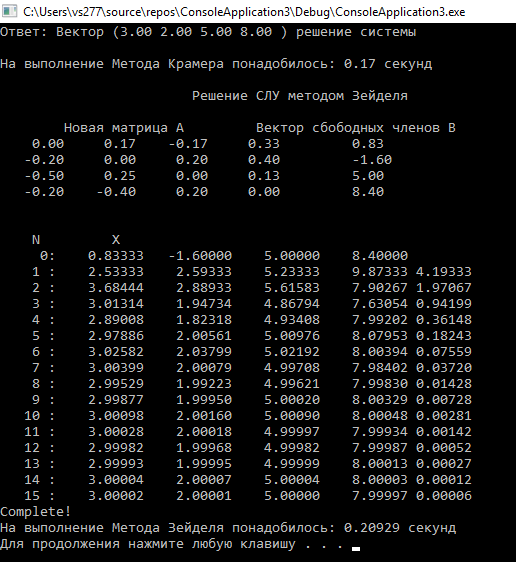
Окончания итерационного процесса в методе Зейделя произойдет при достижении точности и при условии:

**Задание №3**

Исследовать сходимость итерационных процессов для конкретной СЛУ и решить ее точным и приближенным методом, разработанным в заданиях **1** и **2**. Сопоставить время расчетов и точность результатов. Исходная СЛУ и вектор свободных членов:

# Практическая часть





Модули диагональных элементов матрицы *А* для каждой строки больше суммы остальных элементов строки, следовательно метод Зейделя будет сходиться независимо от выбора начального приближения. Как видно из вывода программы, при примерно одинаковой точности метод Зейделя в данной реализации работает медленнее метода Крамера, если начальное приближение не выбрано слишком близким к корню.

**Приложения**

// ConsoleApplication3.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

//

#include "pch.h"

#include <iostream>

#include <cmath>

#include <stdlib.h>

#include <iomanip>

#include <ctime>

#include <ratio>

#include <chrono>

using namespace std;

using namespace std::chrono;

class Matrix

{

private:

double\*\* matrix;

size\_t x\_size;

size\_t y\_size;

public:

Matrix(int x, int y) : x\_size(x), y\_size(y)

{

matrix = new double\*[x\_size];

for (int i = 0; i < x\_size; ++i)

matrix[i] = new double[y\_size];

for (int i = 0; i < x\_size; ++i)

{

for (int j = 0; j < y\_size; ++j)

{

matrix[i][j] = 0;

}

}

}

Matrix(const Matrix& m)

{

matrix = m.matrix;

x\_size = m.x\_size;

y\_size = m.y\_size;

}

~Matrix()

{

for (int j = 0; j < y\_size; ++j)

delete[]matrix[j];

delete[]matrix;

}

void fillMatrix()

{

if (x\_size == 0 && y\_size == 0)

{

cerr << "В матрице нет элементов. Создайте матрицу размера MxN и попробуйте снова.";

exit(0);

}

cout << "Введите матрицу" << x\_size << "x" << y\_size << endl;

for (int i = 0; i < x\_size; ++i)

{

for (int j = 0; j < y\_size; ++j)

{

cin >> matrix[i][j];

}

}

}

void printMatrix()const

{

cout << "Матрица " << x\_size << "x" << y\_size << endl;

cout.setf(ios::fixed);

cout.precision(3);

for (int i = 0; i < x\_size; i++)

{

for (int j = 0; j < y\_size; j++)

{

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

}

cout << endl;

}

}

void kramer(double\* vec)

{

double det = 1;

size\_t col = -1;

bool calcDet = false;

double\* v = nullptr;

v = new double[x\_size];

for (size\_t i = 0; i < x\_size; i++)

{

v[i] = vec[i];

}

double\* deter = nullptr;

deter = new double[x\_size];

for (size\_t i = 0; i < x\_size; i++)

{

deter[i] = 1.0;

}

double\*\* mat = nullptr;

for (size\_t c = 0; c <= x\_size; c++)

{

mat = new double\*[x\_size];

for (size\_t j = 0; j < x\_size; j++)

{

mat[j] = new double[x\_size];

}

for (size\_t i = 0; i < x\_size; i++)

{

for (size\_t j = 0; j < x\_size; j++)

{

mat[i][j] = matrix[i][j];

}

}

if (calcDet == true)

{

for (int i = 0; i < x\_size; i++)

{

mat[i][col] = vec[i];

}

}

for (size\_t i = 0; i < x\_size; i++)

{

if (!mat[i][i])

{

for (size\_t j = i + 1; j < x\_size; j++)

{

if (mat[j][i])

{

for (size\_t k = 0; k < x\_size; k++)

{

mat[i][k] += mat[j][k];

}

break;

}

}

}

if (mat[i][i])

{

for (size\_t p = i + 1; p < x\_size; p++)

{

double s = mat[p][i] / mat[i][i];

for (size\_t q = 0; q < x\_size; q++)

{

mat[p][q] -= mat[i][q] \* s;

}

}

}

}

for (size\_t i = 0; i < x\_size; i++)

{

cout.setf(ios::fixed);

cout.precision(2);

for (size\_t j = 0; j < x\_size; j++)

{

cout << setw(6) << mat[i][j] << " ";

}

cout << endl;

}

for (size\_t i = 0; i < x\_size; ++i)

{

if (calcDet == false) det \*= mat[i][i];

else deter[col] \*= mat[i][i];

}

if (calcDet == false) cout << "Определитель матрицы: " << setw(5) << det << endl << endl;

else cout << "Определитель матрицы " << col + 1 << ": " << setw(5) << deter[col] << endl << endl;

calcDet = true;

col++;

}

double\* ansVector = nullptr;

ansVector = new double[x\_size];

for (size\_t x = 0; x < x\_size; ++x)

{

ansVector[x] = deter[x] / det;

}

cout << endl << "Ответ: Вектор (";

for (size\_t x = 0; x < x\_size; ++x)

{

cout << ansVector[x] << " ";

}

cout << ") решение системы" << endl << endl;

for (size\_t i = 0; i < x\_size; ++i)

{

delete[]mat[i];

}

delete[]mat;

mat = nullptr;

delete[]v;

delete[]deter;

delete[]ansVector;

}

void zeidel(double\* vec)

{

const size\_t MAX = 100;

double\* iterVector = nullptr;

iterVector = new double[x\_size];

for (size\_t i = 0; i < x\_size; i++)

{

iterVector[i] = vec[i];

}

double\*\* vector = nullptr;

vector = new double\*[x\_size];

for (size\_t i = 0; i < x\_size; i++)

{

vector[i] = new double[x\_size];

}

for (size\_t i = 0; i < x\_size; i++)

for (size\_t j = 0; j < x\_size; j++)

{

vector[i][j] = 0;

}

int n = 0;

for (size\_t i = 0; i < x\_size; i++)

{

for (size\_t j = 0; j < x\_size; j++)

{

if (i == j)

{

vector[i][j] = 0;

}

else vector[i][j] = -matrix[i][j] / matrix[i][n];

}

iterVector[i] = vec[i] / matrix[i][n];

n++;

}

cout << setw(15) << "\tНовая матрица A" << setw(15) << "\t\tВектор сбободных членов В" << endl;

for (size\_t i = 0; i < x\_size; i++) {

for (size\_t j = 0; j < x\_size; j++)

{

cout << setw(8) << vector[i][j] << " ";

}

cout << setw(8) << " " << iterVector[i];

cout << endl;

}

cout << endl << endl;

double\*\* mat = nullptr;

mat = new double\*[MAX];

for (size\_t j = 0; j < MAX; j++)

{

mat[j] = new double[x\_size];

}

for (size\_t i = 0; i < MAX; i++)

{

for (size\_t j = 0; j < x\_size; j++)

mat[i][j] = 0;

}

cout.setf(ios::fixed);

cout.precision(5);

double max = 0;

for (size\_t i = 0; i < x\_size; ++i)

{

mat[0][i] = iterVector[i];

}

cout << setw(5) << "N" << setw(10) << "X" << endl;

cout << " 0: ";

for (size\_t i = 0; i < x\_size; ++i)

cout << setw(10) << mat[0][i] << " ";

cout << endl;

for (size\_t i = 1; ; ++i)

{

cout << setw(5) << i << " : ";

for (size\_t j = 0; j < x\_size; ++j)

{

mat[i][j] += iterVector[j];

for (size\_t k = 0; k < x\_size; ++k)

{

mat[i][j] += vector[j][k] \* mat[i - 1][k];

}

double k = fabs(mat[i][j] - mat[i - 1][j]);

if (k > max) max = k;

cout << setw(10) << mat[i][j] << " ";

}

cout << setw(6) << max;

if (max < 0.0001)

{

cout << endl << "Complete!" << endl;

break;

}

else max = 0;

cout << endl;

}

for (size\_t i = 0; i < MAX; ++i)

{

delete[]mat[i];

}

delete[]mat;

for (size\_t i = 0; i < x\_size; ++i)

{

delete[]vector[i];

}

delete[]vector;

delete[]iterVector;

}

};

int main()

{

setlocale(LC\_CTYPE, "");

size\_t M;

cout << "\t\tРешение Систем Линейных Уравнений методами Крамера и Зейделя" << endl << endl;

cout << "Введите количество уравнений системы(количество неизвестных): "; cin >> M;

cout << "Введите вектор-столбец свободных членов(" << M << " шт.): ";

double\* vec;

vec = new double[M];

for (size\_t i = 0; i < M; ++i) cin >> vec[i];

Matrix matrix(M, M);

matrix.fillMatrix();

cout << "\t\t\tРешение СЛУ методом Крамера" << endl << endl;

steady\_clock::time\_point kramer\_time1 = steady\_clock::now();

matrix.kramer(vec);

steady\_clock::time\_point kramer\_time2 = steady\_clock::now();

duration<double> kramer\_span = duration\_cast<duration<double>>(kramer\_time2 - kramer\_time1);

cout << "На выполнение Метода Крамера понадобилось: " << kramer\_span.count() << " секунд" << endl << endl;

cout << "\t\t\t Решение СЛУ методом Зейделя" << endl << endl;

steady\_clock::time\_point zeidel\_time1 = steady\_clock::now();

matrix.zeidel(vec);

steady\_clock::time\_point zeidel\_time2 = steady\_clock::now();

duration<double> zeidel\_span = duration\_cast<duration<double>>(zeidel\_time2 - zeidel\_time1);

cout << "На выполнение Метода Зейделя понадобилось: " << zeidel\_span.count() << " секунд" << endl;

system("Pause");

return 0;

}