

**Национальный исследовательский университет**  
**«Высшая школа экономики»**

**Факультет компьютерных наук**

Департамент

**Программной инженерии**

**Домашняя работа по**  
**дисциплине**  
**«Архитектура вычислительных систем»**

Тема работы: Вариант 8. Используя формулы Крамера, найти решение системы линейных уравнений.
--

Выполнил: студент группы БПИ194  
Гребенщиков М. М.

тел. +7 (922) 704 5875  
e-mail адрес: mmgrebenschikov@edu.hse.ru

Преподаватель: Легалов Александр Иванович

Москва, 2020

## Структура работы

1. `Kramer.cpp` – содержит исходный код программы
2. `inputData` – тестовые наборы
3. `Note.pdf` – Отчёт

В коде программы находятся комментарии, описывавшие каждый шаг программы.

## Модель вычислений

В данной программе использовался **итеративный параллелизм** с использованием библиотеки **openMP**. Выделены 4 потока, которые рассчитывают корни методом Крамера. Выбор данной модели обусловлен тем, что при вычислении корней необходимо производить одни и те же действия в циклах. Основной детерминант рассчитывается отдельно и используется параллельными потоками для вычисления своего корня и вывода ответа на экран. Решить проблему «наложения» выходных данных в консоли позволила директива **critical**, которая образует очередь из потоков на выполнение команды вывода. Для определения номера корня в потоке использовалась функция `omp_get_thread_num()`. Библиотека оказалась очень удобной для разработки.

## Тестирование программы

В качестве входных данных программа принимает коэффициенты СЛАУ из консоли. Пользователь вводит по 1 числу. Область допустимых значений:  $[-999; 999]$ . Предусмотрена обработка некорректных данных.

22 89 93 42

58 23 874 169

444 358 232 33

22 89 91 65

$\Delta_4 = \begin{vmatrix} 46 & 29 & 93 & 42 \\ 58 & 23 & 874 & 169 \\ 444 & 358 & 232 & 33 \\ 22 & 89 & 91 & 65 \end{vmatrix} = -801800440;$

$x_1 = \Delta_1 / \Delta = \frac{-126069198}{1879959080} = \frac{-63034599}{939979540} = -0,067$

$x_2 = \Delta_2 / \Delta = \frac{894058420}{1879959080} = \frac{44702921}{93997954} = 0,5755$

$x_3 = \Delta_3 / \Delta = \frac{701550256}{1879959080} = \frac{87693782}{234994885} = 0,3731$

$x_4 = \Delta_4 / \Delta = \frac{-801800440}{1879959080} = \frac{-20045011}{46998977} = -0,4264$

```
a21 = 58
a22 = 23
a23 = 874
a24 = 385
b2 = 169

a31 = 444
a32 = 358
a33 = 232
a34 = 455
b3 = 33

a41 = 22
a42 = 89
a43 = 91
a44 = 23
b4 = 65

46*x1 + 29*x2 + 93*x3 + 8*x4 = 42
58*x1 + 23*x2 + 874*x3 + 385*x4 = 169
444*x1 + 358*x2 + 232*x3 + 455*x4 = 33
22*x1 + 89*x2 + 91*x3 + 23*x4 = 65

SLAU roots: -0.0670595 0.475573 0.373173 -0.426499
```

Test1 и проверка

Система уравнений:

$$\begin{cases} 46x_1 + 29x_2 + 93x_3 + 8x_4 = 42 \\ 58x_1 + 23x_2 + 874x_3 + 385x_4 = 169 \\ 444x_1 + 358x_2 + 232x_3 + 455x_4 = 33 \\ 22x_1 + 89x_2 + 91x_3 + 23x_4 = 65 \end{cases}$$

Консоль отладки Microsoft Visual Studio

```
a11 = 7
a12 = 7
a13 = 7
a14 = 7
b1 = 7

a21 = 1000
a21 = -1000
a21 = 999
a22 = -9999
a22 = 7
a23 = 7
a24 = 7
b2 = 7

a31 = 7
a32 = 7
a33 = 7
a34 = 7
b3 = 7

a41 = 7
a42 = 7
a43 = 7
a44 = 7
b4 = 7

7*x1    7*x2    7*x3    7*x4    = 7
999*x1   7*x2    7*x3    7*x4    = 7
7*x1     7*x2    7*x3    7*x4    = 7
7*x1     7*x2    7*x3    7*x4    = 7

Main determinant is 0. Can't use Kramer method
```

Test2

Консоль отладки Microsoft Visual Studio

```
a11 = 23
a12 = 45
a13 = 213
a14 = 4
b1 = 34

a21 = 22
a22 = 46
a23 = 23
a24 = 11
b2 = 542

a31 = 32
a32 = 54
a33 = 34
a34 = 654
b3 = 23

a41 = 53
a42 = 34
a43 = 12
a44 = 78
b4 = 34

23*x1    45*x2    213*x3    4*x4    = 34
22*x1    46*x2    23*x3    11*x4    = 542
32*x1    54*x2    34*x3    654*x4   = 23
53*x1    34*x2    12*x3    78*x4    = 34

SLAU roots: -8.72189 -2.56711 17.4395 -0.84457
```

Test3

Test3

## Алгоритм работы программы

1. Считывание коэффициентов СЛАУ
2. Вычисление основного определителя
3. Проверка определителя на равенство 0
4. Запуск 4 поток вычисления корней
5. Каждый поток по одному выводит подсчитанный  $x_i$  на экра

## Список литературы

1. [https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4\\_%D0%9A%D1%80%D0%B0%D0%BC%D0%B5%D1%80%D0%B0](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%9A%D1%80%D0%B0%D0%BC%D0%B5%D1%80%D0%B0)
2. <http://www.williamspublishing.com/PDF/5-8459-0388-2/part.pdf>

## Приложение. Код программы

```
#include <iostream>
#include <fstream>
#include <thread>
#include <stdlib.h>
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <string>
#include <omp.h>

//Метод вывода СЛАУ на экран
void printSLAU(int arc[4][5]) {
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 5; j++) {
            if (j == 4)
                std::cout << "=" << arc[i][j] << std::endl;
            else
                std::cout << arc[i][j] << "x" << j + 1 << "\t";
        }
    }
}

//Вычисление определителя 3 на 3
long long calcDetThreeXThree(int arc[3][3]) {
    long long det = arc[0][0] * arc[1][1] * arc[2][2] + arc[0][1] * arc[1][2] * arc[2][0] + arc[1][0] *
arc[2][1] * arc[0][2];
    det -= (arc[0][2] * arc[1][1] * arc[2][0] + arc[1][0] * arc[0][1] * arc[2][2] + arc[0][0] * arc[2][1] *
arc[1][2]);
    return det;
}

//Вычисление определителя 4 на 4
long long calcDetFourXFour(int arc[4][5]) {
    long long det = 0;
    int minor[3][3];

    //Разложение матрицы по первой строке
    for (int i = 0; i < 4; i++) {
        for (int j = 1; j < 4; j++)
        {
            int ind = 0;
            //Составление минора
            for (int k = 0; k < 4; k++)
            {
                if (k == i)
                    continue;

                minor[j - 1][ind] = arc[j][k];
                ind++;
            }
        }
    }
}
```

```

    }
    //Вычисление определителя минора
    long long minorDet = calcDetThreeXThree(minor);
    if (i % 2 == 1)
        minorDet *= -1;
    det += (long long)arc[0][i] * minorDet;
}
return det;
}

int main()
{
    int arc[4][5];

    //Считывание входных данных из консоли
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            while (true)
            {
                try
                {
                    std::string s;
                    if (j != 4)
                        std::cout << "a" << i + 1 << j + 1 << " = ";
                    else
                        std::cout << "b" << i + 1 << " = ";

                    std::cin >> s;
                    arc[i][j] = std::stoi(s);
                    //Обработка некорректных входных данных
                    if (arc[i][j] >= 1000 || arc[i][j] <= -1000)
                        continue;
                    break;
                }
                catch (std::invalid_argument) {
                    continue;
                }
            }
        }
        std::cout << std::endl;
    }

    //Вывод СЛАУ на экран
    printSLAU(arc);

    //Вычисление детерминанта коэффициентов
    long long mainDet = calcDetFourXFour(arc);

    //Если детерминант 0 - Крамера использовать нельзя
    if (mainDet == 0)
    {
        std::cout << std::endl << "Main determinant is 0. Can't use Kramer method";
        return 0;
    }

    std::cout << std::endl << "SLAU roots: ";

    omp_set_num_threads(4);

    #pragma omp parallel shared(arc, mainDet)

```

```

{
    //Замена столбца коэффициентов свободными членами
    int arrc[4][5];
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++) {
            if (j == omp_get_thread_num())
                arrc[i][j] = arc[i][4];
            else
                arrc[i][j] = arc[i][j];
        }
    }

    //Вычисление определителя
    long long det = calcDetFourXFour(arrc);

    //Вывод и вычисление корня, тут образуется очередь на вход
    #pragma omp critical
    {
        std::cout << (double)det / (double)mainDet << " ";
    }
}

```