# Решение систем линейных уравнений методом Гаусса

Алгоритмизация / Решение систем линейных уравнений методом Гаусса

***Метод Гаусса*** — классический метод решения системы линейных алгебраических уравнений (СЛАУ).Рассмотрим систему линейных уравнений с действительными постоянными коэффициентами:

$$\begin{cases} a_{11}\cdot x_1 + a_{12}\cdot x_2 + \dots + a_{1n}\cdot x_n = y_1 \\ a_{21}\cdot x_1 + a_{22}\cdot x_2 + \dots + a_{2n}\cdot x_n = y_2 \\ \dots \\ a_{n1}\cdot x_1 + a_{n2}\cdot x_2 + \dots + a_{nn}\cdot x_n = y_n \end{cases}$$

или в матричной форме

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

$$\boldsymbol{A}\cdot\boldsymbol{X} = \boldsymbol{Y}$$

Метод Гаусса решения системы линейных уравнений включает в себя 2 стадии:

- последовательное (прямое) исключение;
- обратная подстановка.

## Последовательное исключение

Исключения Гаусса основаны на идее последовательного исключения переменных по одной до тех пор, пока не останется только одно уравнение с одной переменной в левой части. Затем это уравнение решается относительно единственной переменной. Таким образом, систему уравнений приводят к треугольной (ступенчатой) форме. Для этого среди элементов первого столбца матрицы выбирают ненулевой (а чаще максимальный) элемент и перемещают его на крайнее верхнее положение перестановкой строк. Затем нормируют все уравнения, разделив его на коэффициент $\mathbf{a_{i1}}$, где $\mathbf{i}$– номер столбца.

$$\begin{cases} x_1 + \dfrac{a_{12}}{a_{11}}\cdot x_2 + \dots + \dfrac{a_{1n}}{a_{11}}\cdot x_n = \dfrac{y_1}{a_{11}} \\ x_1 + \dfrac{a_{22}}{a_{21}}\cdot x_2 + \dots + \dfrac{a_{2n}}{a_{21}}\cdot x_n = \dfrac{y_2}{a_{21}} \\ \dots \\ x_1 + \dfrac{a_{n2}}{a_{n1}}\cdot x_2 + \dots + \dfrac{a_{nn}}{a_{n1}}\cdot x_n = \dfrac{y_n}{a_{n1}} \end{cases}$$

Затем вычитают получившуюся после перестановки первую строку из остальных строк:

$$\begin{cases} x_1 + \dfrac{a_{12}}{a_{11}}\cdot x_2 + \dots + \dfrac{a_{1n}}{a_{11}}\cdot x_n = \dfrac{y_1}{a_{11}} \\ 0 + \left(\dfrac{a_{22}}{a_{21}} - \dfrac{a_{12}}{a_{11}}\right)\cdot x_2 + \dots + \left(\dfrac{a_{2n}}{a_{21}} - \dfrac{a_{1n}}{a_{11}}\right)\cdot x_n = \left(\dfrac{y_2}{a_{21}} - \dfrac{y_1}{a_{11}}\right) \\ \dots \\ 0 + \left(\dfrac{a_{n2}}{a_{n1}} - \dfrac{a_{12}}{a_{11}}\right)\cdot x_2 + \dots + \left(\dfrac{a_{nn}}{a_{n1}} - \dfrac{a_{1n}}{a_{11}}\right)\cdot x_n = \left(\dfrac{y_n}{a_{n1}} - \dfrac{y_1}{a_{11}}\right) \end{cases}$$

Получают новую систему уравнений, в которой заменены соответствующие коэффициенты.

$$\begin{cases} x_1 + a'_{12} \cdot x_2 + \cdots + a'_{1n} \cdot x_n = y'_1 \\ 0 + a'_{22} \cdot x_2 + \cdots + a'_{2n} \cdot x_n = y'_2 \\ \cdots \\ 0 + a'_{n2} \cdot x_2 + \cdots + a'_{nn} \cdot x_n = y'_n \end{cases}$$

После того, как указанные преобразования были совершены, первую строку и первый столбец мысленно вычёркивают и продолжают указанный процесс для всех последующих уравнений пока не останется уравнение с одной неизвестной:

$$\begin{cases} x_1 + a'_{12} \cdot x_2 + a'_{13} \cdot x_3 + \cdots + a'_{1n} \cdot x_n = y'_1 \\ 0 + x_2 + a''_{23} \cdot x_3 + \cdots + a''_{2n} \cdot x_n = y''_2 \\ 0 + 0 + x_3 + \cdots + a'''_{3n} \cdot x_n = y'''_3 \\ \cdots \\ 0 + 0 + 0 + \cdots + x_n = y^{n\prime}_n \end{cases}$$

## Обратная подстановка

Обратная подстановка предполагает подстановку полученного на предыдущем шаге значения переменной $\boldsymbol{x_n}$ в предыдущие уравнения:

$$x_{n-1} = y^{(n-1)'}_{n-1} - a^{(n-1)\prime}_{(n-1)n} \cdot x_n$$

$$x_{n-2} + a^{(n-2)\prime}_{(n-2)(n-1)} \cdot x_{n-1} = y^{(n-2)'}_{n-2} - a^{(n-2)\prime}_{(n-2)n} \cdot x_n$$

$$\cdots$$

$$x_2 + a''_{23} \cdot x_3 + \cdots + a''_{2(n-1)} \cdot x_{n-1} = y''_2 - a''_{2n} \cdot x_n$$

$$x_1 + a'_{12} \cdot x_2 + a'_{13} \cdot x_3 + \cdots + a'_{1(n-1)} \cdot x_{n-1} = y'_1 - a'_{1n} \cdot x_n$$

Эта процедура повторяется для всех оставшихся решений:

$$x_{n-2} = \left(y^{(n-2)'}_{n-2} - a^{(n-2)'}_{(n-2)n} \cdot x_n\right) - a^{(n-2)\prime}_{(n-2)(n-1)} \cdot x_{n-1}$$

$$\cdots$$

$$x_2 + a''_{23} \cdot x_3 + \cdots = (y''_2 - a''_{2n} \cdot x_n) - a''_{2(n-1)} \cdot x_{n-1}$$

$$x_1 + a'_{12} \cdot x_2 + a'_{13} \cdot x_3 + \cdots = (y'_1 - a'_{1n} \cdot x_n) - a'_{1(n-1)} \cdot x_{n-1}$$

## Иллюстрирующий пример

Пусть дана система уравнений

$$\begin{cases} 2 \cdot x_1 + 4 \cdot x_2 + 1 \cdot x_3 = 36 \\ 5 \cdot x_1 + 2 \cdot x_2 + 1 \cdot x_3 = 47 \\ 2 \cdot x_1 + 3 \cdot x_2 + 4 \cdot x_3 = 37 \end{cases}$$

или в матричной форме

$$\begin{bmatrix} 2 & 4 & 1 \\ 5 & 2 & 1 \\ 2 & 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 36 \\ 47 \\ 37 \end{bmatrix}$$

Выбираем строку с максимальным коэффициентом $a_{i1}$ и меняем ее с первой.

$$\begin{bmatrix} \boxed{5} & 2 & 1 \\ 2 & 4 & 1 \\ 2 & 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 47 \\ 36 \\ 37 \end{bmatrix}$$

Нормируем уравнения относительно коэффициента при $x_1$:

$$\begin{bmatrix} 1 & \frac{2}{5} & \frac{1}{5} \\ 1 & \frac{4}{2} & \frac{1}{2} \\ 1 & \frac{3}{2} & \frac{4}{2} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{47}{5} \\ \frac{36}{2} \\ \frac{37}{2} \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0.4 & 0.2 \\ 1 & 2 & 0.5 \\ 1 & 1.5 & 2 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 9.4 \\ 18 \\ 18.5 \end{bmatrix}$$

Вычитаем 1 уравнение из 2 и 3:

$$\begin{bmatrix} 1 & 0.4 & 0.2 \\ 0 & 1.6 & 0.3 \\ 0 & 1.1 & 1.8 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 9.4 \\ 8.6 \\ 9.1 \end{bmatrix}$$

Выбираем строку с наибольшим коэффициентом при $a_{i2}$(уравнение 1 не рассматривается) и перемещаем ее на место 2.

$$\begin{bmatrix} 1 & 0.4 & 0.2 \\ 0 & \boxed{1.6} & 0.3 \\ 0 & 1.1 & 1.8 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 9.4 \\ 8.6 \\ 9.1 \end{bmatrix}$$

Нормируем 2 и 3 уравнения относительно коэффициента при $x_2$

$$\begin{bmatrix} 1 & 0.4 & 0.2 \\ 0 & 1 & 0.1875 \\ 0 & 1 & 1.636 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 9.4 \\ 5.375 \\ 8.272 \end{bmatrix}$$

Вычитаем уравнение 2 из 3

$$\begin{bmatrix} 1 & 0.4 & 0.2 \\ 0 & 1 & 0.1875 \\ 0 & 0 & 1.4489 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 9.4 \\ 5.375 \\ 2.897 \end{bmatrix}$$

Нормируем уравнение 3 относительно коэффициента при $x_3$

$$\begin{bmatrix} 1 & 0.4 & 0.2 \\ 0 & 1 & 0.166 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 9.4 \\ 5.333 \\ 2 \end{bmatrix}$$

Откуда получаем $x_3=2$. Подставляем полученное значение в уравнения 2 и 1 получаем

$$x_2 = 5.333 - 0.1666 \cdot 2 = 5.333 - 0.333 = 5$$

$$x_1 + 0.4 \cdot x2 = 9.4 - 0.2 \cdot 2 = 9.4 - 0.4 = 9$$

Подставляя полученное значение $x_2$=5 в уравнение 1, найдем

$$x_1 = 9 - 0.4 \cdot 5 = 9 - 2 = 7$$

Таким образом, решением системы уравнений будет вектор

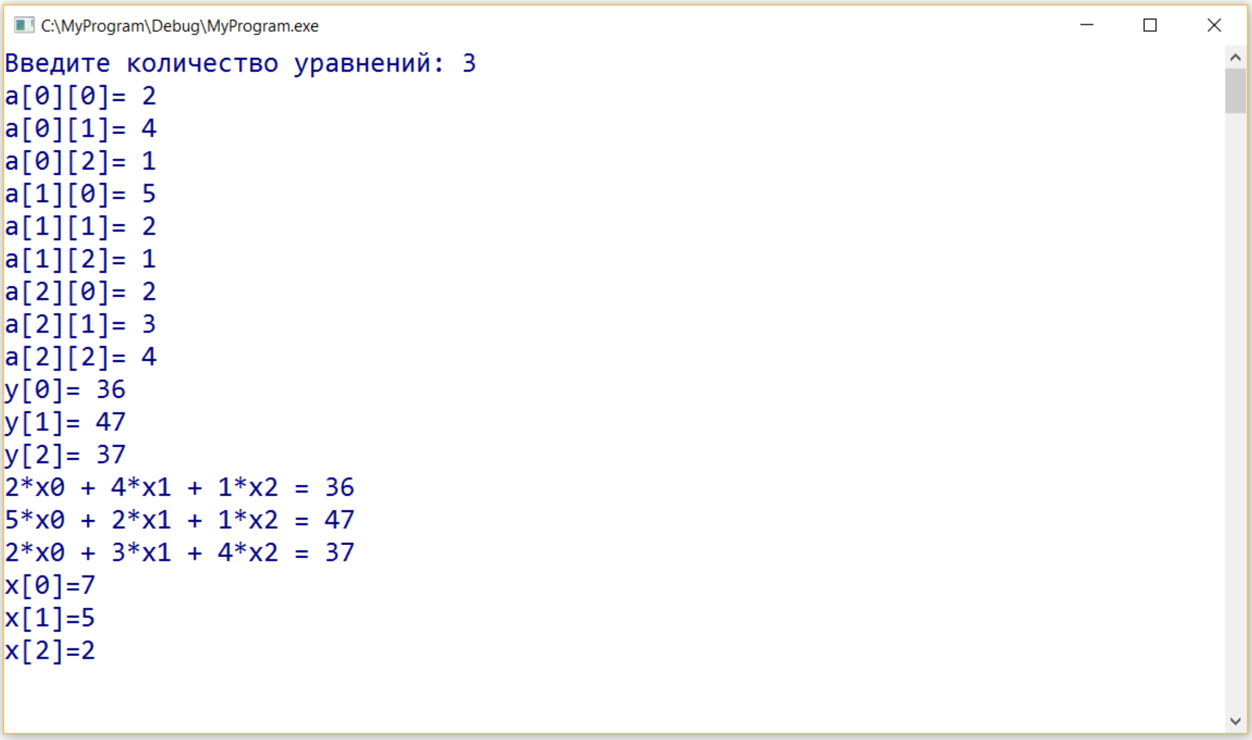
$$\boldsymbol{X} = [7 \quad 5 \quad 2]^T$$

## *Реализация на C++*

```cpp
#include <iostream>
using namespace std;
// Вывод системы уравнений
void sysout(double **a, double *y, int n)
{
  for (int i = 0; i < n; i++)
  {
    for (int j = 0; j < n; j++)
    {
      cout << a[i][j] << "*x" << j;
      if (j < n - 1)
        cout << " + ";
    }
    cout << " = " << y[i] << endl;
  }
  return;
}
double * gauss(double **a, double *y, int n)
{
  double *x, max;
  int k, index;
  const double eps = 0.00001;  // точность
  x = new double[n];
  k = 0;
  while (k < n)
  {
    // Поиск строки с максимальным a[i][k]
    max = abs(a[k][k]);
    index = k;
    for (int i = k + 1; i < n; i++)
    {
      if (abs(a[i][k]) > max)
      {
        max = abs(a[i][k]);
        index = i;
      }
    }
    // Перестановка строк
    if (max < eps)
    {
      // нет ненулевых диагональных элементов
      cout << "Решение получить невозможно из-за нулевого столбца ";
      cout << index << " матрицы A" << endl;
      return 0;
    }
    for (int j = 0; j < n; j++)
    {
      double temp = a[k][j];
      a[k][j] = a[index][j];
```

```
        a[index][j] = temp;
      }
      double temp = y[k];
      y[k] = y[index];
      y[index] = temp;
      // Нормализация уравнений
      for (int i = k; i < n; i++)
      {
        double temp = a[i][k];
        if (abs(temp) < eps) continue; // для нулевого коэффициента пропустить
        for (int j = 0; j < n; j++)
          a[i][j] = a[i][j] / temp;
        y[i] = y[i] / temp;
        if (i == k)  continue; // уравнение не вычитать само из себя
        for (int j = 0; j < n; j++)
          a[i][j] = a[i][j] - a[k][j];
        y[i] = y[i] - y[k];
      }
      k++;
    }
    // обратная подстановка
    for (k = n - 1; k >= 0; k--)
    {
      x[k] = y[k];
      for (int i = 0; i < k; i++)
        y[i] = y[i] - a[i][k] * x[k];
    }
    return x;
  }
  int main()
  {
    double **a, *y, *x;
    int n;
    system("chcp 1251");
    system("cls");
    cout << "Введите количество уравнений: ";
    cin >> n;
    a = new double*[n];
    y = new double[n];
    for (int i = 0; i < n; i++)
    {
      a[i] = new double[n];
      for (int j = 0; j < n; j++)
      {
        cout << "a[" << i << "][" << j << "]= ";
        cin >> a[i][j];
      }
    }
    for (int i = 0; i < n; i++)
    {
      cout << "y[" << i << "]= ";
      cin >> y[i];
    }
    sysout(a, y, n);
    x = gauss(a, y, n);
    for (int i = 0; i < n; i++)
      cout << "x[" << i << "]=" << x[i] << endl;
    cin.get(); cin.get();
    return 0;
  }
```

Результат выполнения



```
C:\MyProgram\Debug\MyProgram.exe
Введите количество уравнений: 3
a[0][0]= 2
a[0][1]= 4
a[0][2]= 1
a[1][0]= 5
a[1][1]= 2
a[1][2]= 1
a[2][0]= 2
a[2][1]= 3
a[2][2]= 4
y[0]= 36
y[1]= 47
y[2]= 37
2*x0 + 4*x1 + 1*x2 = 36
5*x0 + 2*x1 + 1*x2 = 47
2*x0 + 3*x1 + 4*x2 = 37
x[0]=7
x[1]=5
x[2]=2
```

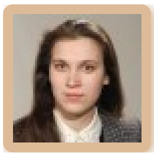
Назад: *Алгоритмизация*

## Комментариев к записи: 46

**Валерий** 21.05.2019 в 12:56

В программе ошибка. В строке 77 должно быть return y; Не будьте пожирателями времени - исправьте!

Ответить ↓



**Елена Вставская** 24.05.2019 в 10:07

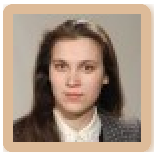
Пока не увидела проблемы. Мы определяем вектор x, который и возвращает функция.

Ответить ↓

**Матвей** 14.05.2019 в 22:31

А как проверить систему на совместность и на случай бесконечного множества решений? Помогите, пожалуйста

Ответить ↓



**Елена Вставская** 24.05.2019 в 10:09

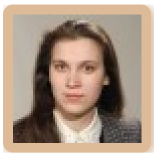
Ну, например, посчитать определитель.
Ответить ↓

---

**Дмитрий** 09.03.2019 в 14:30

Можете подсказать, в цикле while значит переменная k? Как я понял, это номер столбца, это верно?)
Ответить ↓

---



**Елена Вставская** 09.03.2019 в 14:41

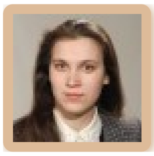
Это номер рассматриваемого уравнения
Ответить ↓

---

**Кирилл** 21.12.2018 в 22:22

А вы в курсе, что вы использовали метод Жордана-Гаусса? И что это две разные вещи?
Ответить ↓

---



**Елена Вставская** 22.12.2018 в 07:22

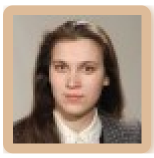
Теперь в курсе :)
Ответить ↓

---

**Satoru** 27.06.2018 в 19:41

Здраствуйте не подскажите как решить уравнение с вырожденной матрицей (нету решения, либо множество) или какой алгоритм позволяет найти решения.
Ответить ↓

---



**Елена Вставская** 28.06.2018 в 05:30

С вырожденный матрицей действительно будет множество решений
Ответить ↓

---

**Олег** 16.05.2018 в 18:08

Все сделано замечательно, только небольшое замечание Елена. Ваша программа не будет выполнятся корректно, если система линейных уравнений, имеет бесконечное множество решений - тут Вам нужно подумать что возвращать при if (max < eps). С уважением, Олег

Ответить ↓

---

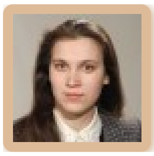
**Денис** 20.02.2017 в 16:16

```
for (int i = 0; i < n; i++)
delete[] a[i];
delete[] a;
delete[] x;
delete[] y;
```

Дабы не было утечек памяти.

Ответить ↓

---



**Елена Вставская** 20.02.2017 в 16:21

Согласна с замечанием

Ответить ↓

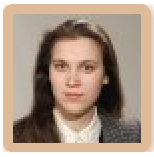
---

**Алексей** 10.11.2016 в 21:18

Программа вылетает при несовместной системе

Ответить ↓

---



**Елена Вставская** 11.11.2016 в 19:34

Да, есть такая недоработка. Проверка совместности системы в примере отсутствует.

Ответить ↓

---

**Эдуард** 12.10.2016 в 08:48

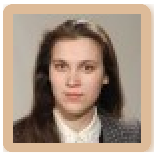
Вместо

```
const double eps = 0.00001;  // точность
```

можно прописать

```
const double eps = DBL_EPSILON;  // точность
// DBL_EPSILON определено во float.h
```

Ответить ↓



**Елена Вставская** 12.10.2016 в 09:28

Да, но для одной константы придется вставлять в проект целую библиотеку `float.h`.

Ответить ↓

**Эдуард** 12.10.2016 в 10:59

Это не библиотека, нет идущих вместе никаких исполняемых модулей, это просто определения констант, точнее макросов препроцессора, т.е. вместо вашего 0.00001 просто подставится число, в итоге даже размер исполняемого файла не изменится.

Ответить ↓

**Эдуард** 12.10.2016 в 08:40

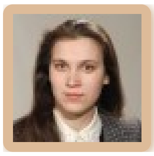
Забыли про `delete`, освобождайте память выделенную `new`.

Ответить ↓

**Алексей** 14.05.2016 в 00:07

пишет [Error] 'abs' was not declared in this scope, объявляю и появляется куча ошибок, можете сказать как поправить?

Ответить ↓



**Елена Вставская** 14.05.2016 в 06:44

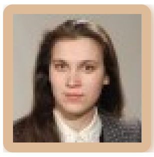
Попробуйте подключить <cmath> или воспользоваться функцией `fabs()`.

Ответить ↓

**Дмитрий** 07.11.2015 в 03:36

А есть ли какой-то способ решить систему из 2х уровнений, но с 3мя неизвестными ? К примеру, если нам дана сумма их a+b+c=const(s) и const(1)*a+const(2)*b+const(3)*c = const(s)*const(4) ? Ну или, что-то подобное, Спасибо.

Ответить ↓



**admin** 07.11.2015 в 07:09

Дмитрий, для того чтобы система имела единственное решение необходимым условием является одинаковое количество уравнений и неизвестных. В случае если количество неизвестных больше, чем количество уравнений, система будет иметь бесконечное множество решений.

Ответить ↓

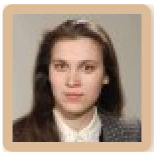
---

**Игорь** 25.11.2015 в 12:59

Добрый день, а можно ли перепрограммировать текущий код, так чтобы программа могла также решать системы с неодинаковым количеством переменных и количеством строк, как например сделали на этом сайте: http://math.semestr.ru/gauss/gauss.php ? С Уважением, Игорь.

Ответить ↓

---



**Елена Вставская** 25.11.2015 в 19:38

Если количество переменных больше количества уравнений, то часть переменных, превышающих количество уравнений, задается в качестве свободных. Если количество уравнений превышает количество переменных, то такая система вообще может не иметь решения в случае если она не является вырожденной. Пример вырожденной системы уравнений: x1 + 2*x2 = 3 3*x1 + 6*x2 = 9

Ответить ↓

---

**Олег** 24.10.2015 в 10:30

Здравствуйте! Спасибо за код. У меня вылазят ошибки sysout: идентификатор не найден и gauss: идентификатор не найден хотя все сделал, как у вас. В чем может быть хитрость?

Ответить ↓

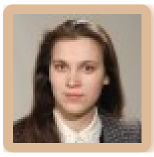
---

**Олег** 24.10.2015 в 11:11

sysout я разобрался, а вот ошибка с gauss осталась

Ответить ↓

---



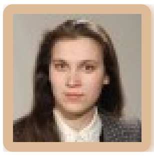
**admin** 24.10.2015 в 15:01

Ответить без Вашего кода довольно сложно. У меня приведенный пример компилируется и работает.

Ответить ↓

**Анастасия** 23.10.2015 в 21:04

Можно еще вопрос? в моей системе программа не выдает 0. т.е. у меня система:
10x1-7x2=7;
-3x1+2x2+6x3=4;
5x1-x2+5x3=6.
ответ, посчитанный вручную :
x1=0;
x2=-1;
x3=1.
программа выдает:
x1=1.11022e-016;
x2=-1;
x3=1.
почему так может быть?
Ответить ↓



**admin** 23.10.2015 в 21:10

Это связано с представлением вещественных чисел в разрядной сетке вычислительной машины. Посмотрите эту статью.
Ответить ↓

**Игорь** 06.10.2015 в 14:32

Добрый день, после x = gauss(a, y, n); в главной функции main, нужно добавить проверку исключения, а именно:

```
sysout(a, y, n);
x = gauss(a, y, n);
if (x != 0)
{
for (int i = 0; i < n; i++) {
cout << "x[" << i << "]=" << x[i] << endl;
}
```

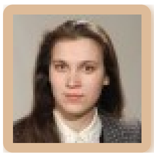
еще меня вот что смутило, например есть такая система уравнений: (в скобочках я имею в виду нижний регистр)
x(2)+3x(3)-x(4)=2
2*x(1)-3*x(2) +2*x(3)=-2
2*x(1)-4*x(2)+x(4)=3
-2*x(1)+5*x(2)-3*x(3)+3*x(4)=1
результат вычисления на бумаге у меня получился такой:
x1=23,3
x2=-10,2
x3=7
x4=8.8
в программе такой:
x[0]=-2.25833
x[1]=-1.25
x[2]=1.38333

`x[3]=2.8`

где может быть ошибка, давайте вместе разберем?

Ответить ↓

---



**admin** 06.10.2015 в 22:20

Спасибо за замечание, Игорь! Ошибку исправила, сейчас Ваша система решается верно.

Ответить ↓

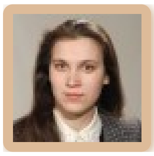
---

**Игорь** 05.10.2015 в 18:09

Добрый день, а если бы, например, было 4 уравнения с 5-ю неизвестными (x1,x2,x3,x4,x5), то это можно запрограммировать? (так как вроде по формулировке - бесконечное множество решений) С Уважением, Игорь.

Ответить ↓

---



**admin** 05.10.2015 в 18:18

Да, так и есть. Количество уравнений должно соответствовать количеству неизвестных, причем система должна быть невырожденной, чтобы она имела единственное решение.

Ответить ↓

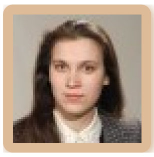
---

**Игорь** 28.09.2015 в 19:08

Добрый день, почему после нормирования 2 и 3 уравнения относительно коэффициента при x2 в первой строке вместо [1 0,4 0,2] * [x1] = [9,4] стало [1 0,5 0,25]*[x1] = [10]? С Уважением, Игорь.

Ответить ↓

---



**admin** 28.09.2015 в 19:33

Игорь, спасибо! Исправила.

Ответить ↓

---

**Тимур** 10.06.2015 в 20:28

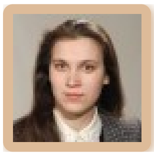
только 0 и 1

Ответить ↓

---

**Тимур** 09.06.2015 в 23:16

здравствуйте, а над полем GF(2) как сделать?
Ответить ↓



**admin** 10.06.2015 в 17:47

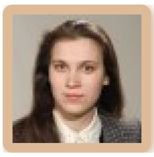
не поняла вопроса
Ответить ↓

**Тимур** 10.06.2015 в 20:28

ну по mod 2( чтоб были 0 или 1)
Ответить ↓

**Тимур** 10.06.2015 в 20:29

а это только для квадратных матриц или любых?
Ответить ↓

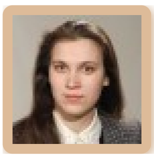


**admin** 10.06.2015 в 20:53

Для однозначного решения количество уравнений должно соответствовать количеству неизвестных. Поэтому для прямоугольных матриц однозначного решения не существует.
Ответить ↓

**Тимур** 10.06.2015 в 20:56

спасибо, а для 0 и 1 ?

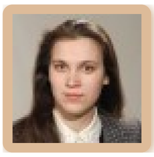


**admin** 11.06.2015 в 01:11

Все равно не поняла вопроса. Какую задачу требуется решить?

**Тимур** 11.06.2015 в 02:35

*Z2* Решение СЛАУ по модулю, по модулю 2 *Z2 Z2*



**admin** 11.06.2015 в 18:37

Тут метод Гаусса вряд ли поможет. Это - тема для отдельной статьи.