Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Методы численного анализа

**ОТЧЁТ**

к лабораторной работе

на тему

Решение систем линейных алгебраических уравнений (СЛАУ)

методом Гаусса и с помощью его модификаций

Выполнил: студент группы 053501

Криштафович Карина Дмитриевна

Проверил: Анисимов Владимир Яковлевич

Минск 2021

**Оглавление**

[**Цели выполнения задания** 3](#_Toc65424824)

[**Краткие теоритические сведения** 4](#_Toc65424825)

[**Задание** 9](#_Toc65424826)

[**Программная реализация** 10](#_Toc65424827)

[**Полученные результаты** 17](#_Toc65424828)

[**Оценка** 18](#_Toc65424829)

[**Выводы** 19](#_Toc65424830)

Вариант 14

# **Цели выполнения задания**

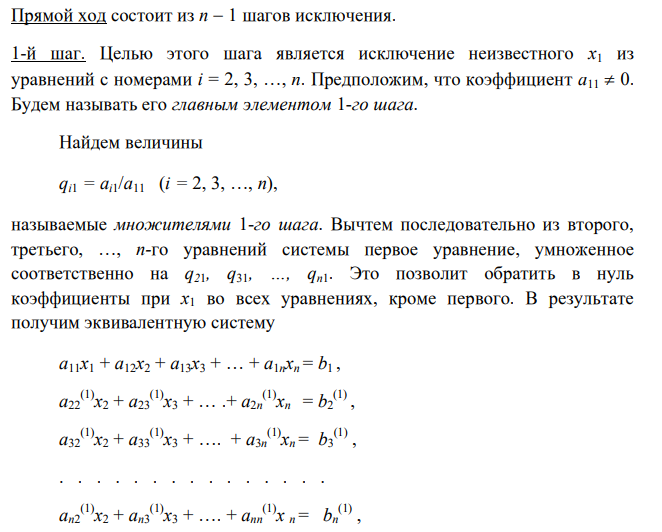
* изучить метод Гаусса и его модификации, составить алгоритм метода и программу его реализации, получить численное решение заданной СЛАУ;
* составить алгоритм решения СЛАУ указанными методами, применимый для организации вычислений на ЭВМ;
* составить программу решения СЛАУ по разработанному алгоритму;
* выполнить тестовые примеры и проверить правильность работы программы.

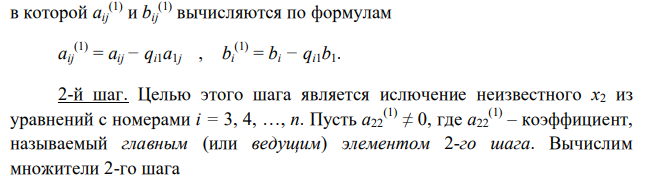
# **Краткие теоритические сведения**

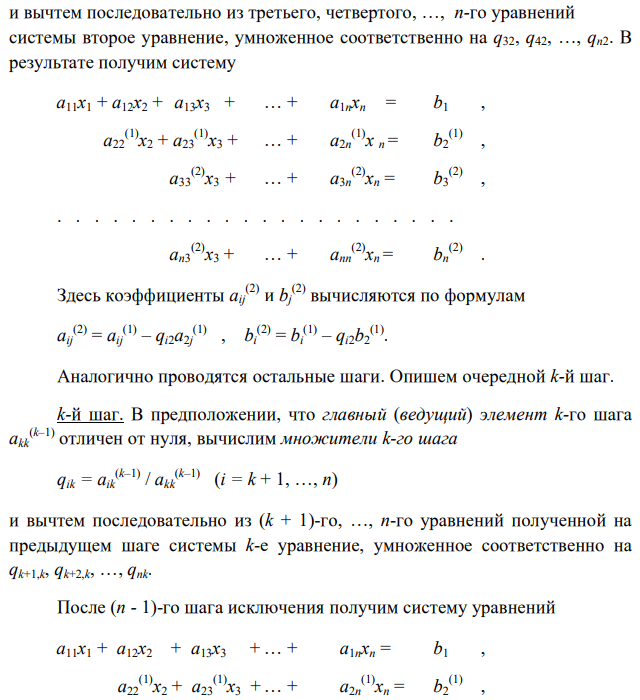
Метод Гаусса прекрасно подходит для решения систем линейных алгебраических уравнений (СЛАУ). Он обладает рядом преимуществ по сравнению с другими методами:

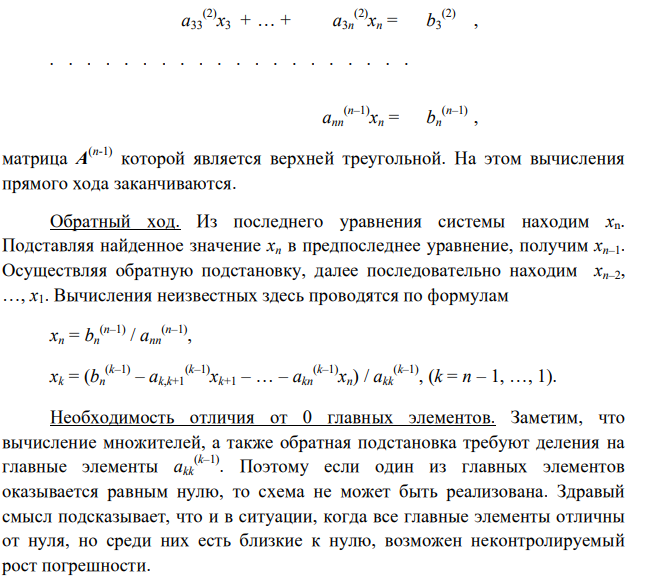
* во-первых, нет необходимости предварительно исследовать систему уравнений на совместность;
* во-вторых, методом Гаусса можно решать не только СЛАУ, в которых число уравнений совпадает с количеством неизвестных переменных и основная матрица системы невырожденная, но и системы уравнений, в которых число уравнений не совпадает с количеством неизвестных переменных или определитель основной матрицы равен нулю;
* в-третьих, метод Гаусса приводит к результату при сравнительно небольшом количестве вычислительных операций.

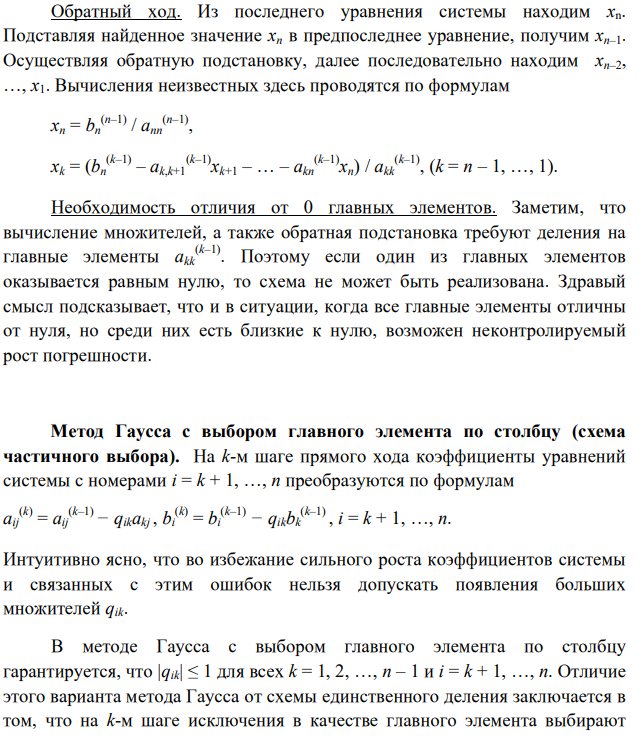
Метод Гаусса включает в себя прямой (приведение расширенной матрицы к ступенчатому виду, то есть получение нулей под главной диагональю) и обратный (получение нулей над главной диагональю расширенной матрицы) ходы. Прямой ход и называется методом Гаусса, обратный - методом Гаусса-Жордана, который отличается от первого только последовательностью исключения переменных.

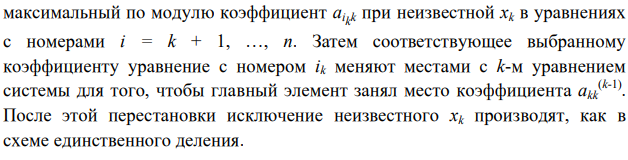
Метод Гаусса идеально подходит для решения систем содержащих больше трех линейных уравнений, для решения систем уравнений, которые не являются квадратными (чего не скажешь про метод Крамера и матричный метод). То есть метод Гаусса - наиболее универсальный метод для нахождения решения любой системы линейных уравнений, он работает в случае, когда система имеет бесконечно много решений или несовместна.

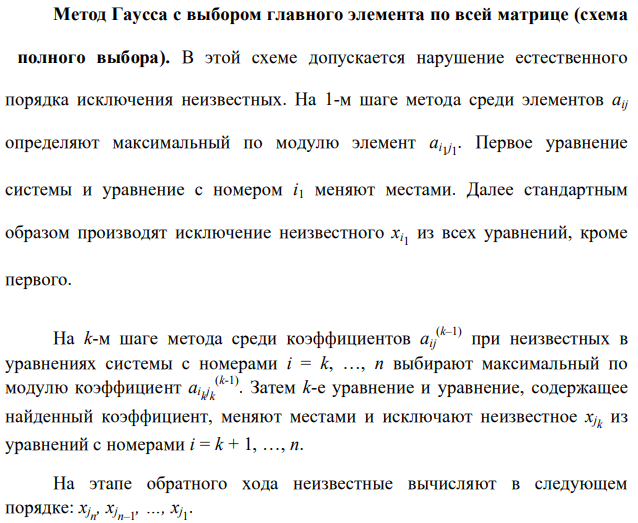




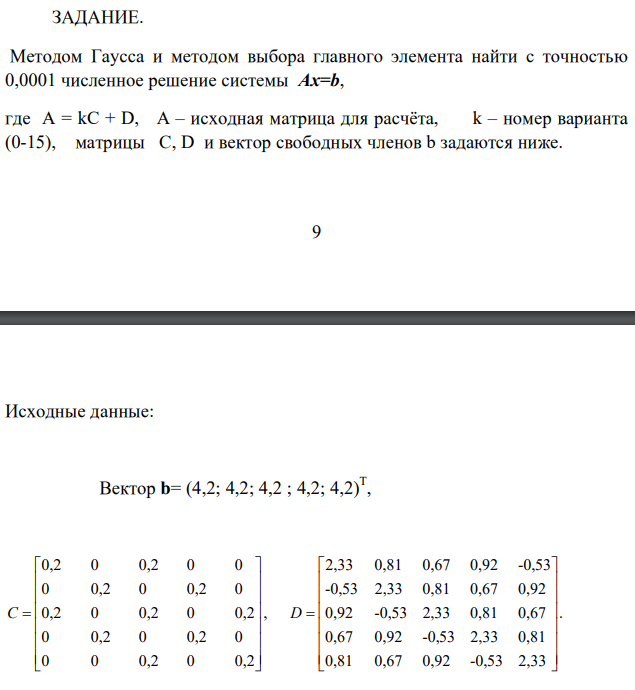




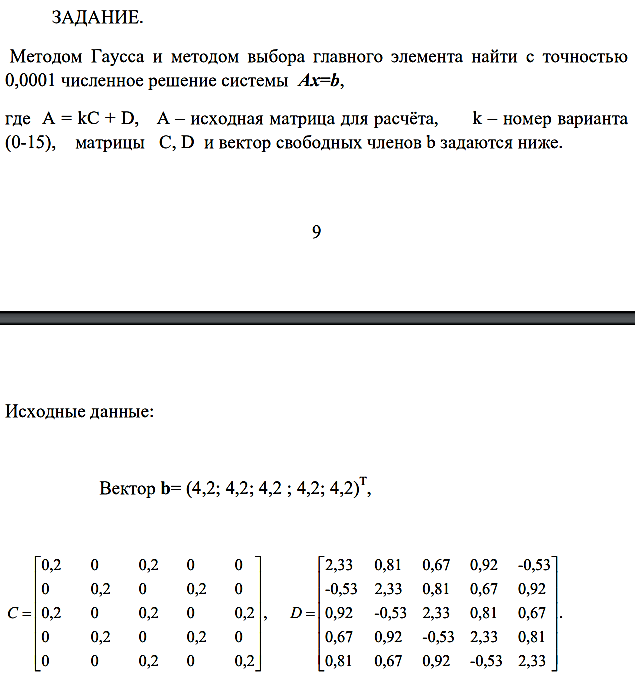




# **Задание**



Вариант 14



Полученные результаты будем сверять с решением, полученным используя подмодуль numpy linalg:

**print(numpy.linalg.solve(A, b))**

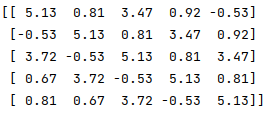
Для исходных данных получим следующий ответ:



**Программная реализация**

**1. Метод Гаусса**

Матрица А, полученная в результате вычисления A=14C+D

**Код программы:**

import numpy

*#Исходные данные*

C = numpy.array([[0.2, 0, 0.2, 0, 0],

                [0, 0.2, 0, 0.2, 0],

                [0.2, 0, 0.2, 0, 0.2],

                [0, 0.2, 0, 0.2, 0],

                [0, 0, 0.2, 0, 0.2]])

D = numpy.array([[2.33, 0.81, 0.67, 0.92, -0.53],

                [-0.53, 2.33, 0.81, 0.67, 0.92],

                [0.92, -0.53, 2.33, 0.81, 0.67],

                [0.67, 0.92, -0.53, 2.33, 0.81],

                [0.81, 0.67, 0.92, -0.53, 2.33]])

b = numpy.array([4.2, 4.2, 4.2, 4.2, 4.2])

b = b.transpose()

A = 14 \* C + D

X = numpy.zeros(len(A))

*#Прямой обход метод Гаусса*

**def** straight\_run(matrix, b):

   for nrow, row in enumerate(matrix):

*# nrow равен номеру строки row содержит саму строку матрицы*

       divider = row[nrow]  *# диагональный элемент*

       row /= divider  *# делим на диагональный элемент.*

       b[nrow] /= divider

       bfactor = b[nrow]

*# вычитаем из всех нижележащих строчек*

       for lower\_row in range(nrow + 1, len(matrix)):

           factor = matrix[lower\_row][nrow]

*# элемент строки в колонке nrow*

           matrix[lower\_row] -= factor \* row

*# вычитаем, чтобы получить ноль в колонке nrow*

           b[lower\_row] -= factor \* bfactor

   return matrix, b

**def** reverse\_run(a, b):  *# перебор строк в обратном порядке*

   n = len(a)

   for k in reversed(range(0, n)):

       X[k] = (b[k] - sum(a[k][i] \* X[i] for i in range(k + 1, n)))/a[k][k]

*#Метод Гаусса*

**def** gauss(A, b):

   check\_zeros\_diag(A, b)

   straight\_run(A, b)

   reverse\_run(A, b)

   for i in range(len(X)):

       print("%.4f" % X[i], end=" ")

   print('\n')

Результат работы вы можете увидеть в тестовом примере 1. Заметим так же, что недостатком данного метода является необходимость отличия от 0 главных элементов. Решением данном проблемы занимается функция check\_zeros\_diag, которая проверяет каждый главный элемент и в случае, если он равен 0 заменяет колонку с этим элементом на любую другую, подходящую для решения. Результат работы вы можете увидеть в тестовом примере 2. Заметим, что если матрица окажется несовместной (все элементы строки будут равны 0, а элемент вектор-столбца b отличен от 0) решений не будет (в тестовом примере 3), а если все элементы строки, включая элемент вектор столбца равен 0, матрица имеет бесконечное количество решений. (в тестовом примере 4)

*#Изменение колонок*

**def** swap\_columns(a, i, j):

   for k in range(len(a)):

       a[k][i], a[k][j] = a[k][j], a[k][i]

*#Проверка на 0 на главной диагонали*

**def** check\_zeros\_diag(matrix, b):

   nstr = len(matrix)

   for i in range(0, nstr):

       if matrix[i][i] == 0:

           check = True

           for j in range(0, nstr):

               if matrix[i][j] != 0 and matrix[j][i] != 0:

                   swap\_columns(matrix, i, j)

                   check = False

                   break

           if check:

               if b[i] == 0:

                   print('The system has infinitive amount of solutions')

               else:

                   print('The system has no solutions')

               exit()

Так же программа была протестирована для СЛАУ с достаточно маленькими коэффициентами, см. в тестовом примере 5. Программа была протестирована для СЛАУ с достаточно большими коэффициентами, см. в тестовом примере 6.

**2. Метод Гаусса с выбором главного элемента по столбцу**

*#Максимальный элемент по столбцу*

**def** straight\_run\_column(matrix, b):

   for nrow in range(len(matrix)): *# nrow равен номеру строки*

*# argmax возвращает номер строки с максимальным элементом*

*в уменьшенной матрице, которая начинается со строки*

*# nrow. Поэтому нужно прибавить nrow к результату*

       pivot = nrow + numpy.argmax(abs(matrix[nrow:, nrow]))

       if pivot != nrow:

           matrix[[nrow, pivot]] = matrix[[pivot, nrow]] # swap

           change = b[pivot]

           b[pivot] = b[nrow]

           b[nrow] = change

       row = matrix[nrow]

       divider = row[nrow]  *# диагональный элемент*

       if abs(divider) < 1e-10:

*# почти нуль на диагонали.*

*#Продолжать не имеет смысла, результат счёта неустойчив*

           print(f"Matrix is incompatible. Max element in column {nrow}: {divider:.3g}")

           exit()

       row /= divider # *делим на диагональный элемент.*

       b[nrow] /= divider

       bfactor = b[nrow]

*# теперь надо вычесть приведённую строку из всех нижележащих строчек*

       for lower\_row in range(nrow + 1, len(matrix)):

           factor = matrix[lower\_row][nrow]

*# элемент строки в колонке nrow*

           matrix[lower\_row] -= factor \* row

*# вычитаем, чтобы получить ноль в колонке nrow*

           b[lower\_row] -= factor \* bfactor

*#Метод Гаусса поиск максимального по столбцу*

**def** gauss\_maxcolumn (A, b):

   check\_zeros\_diag(A, b)

   straight\_run\_column(A,b)

   reverse\_run(A,b)

   for i in range(len(X)):

       print("%.4f" % X[i], end=" ")

   print('\n')

**3. Метод Гаусса с выбором главного элемента по всей матрице**

*#Поиск макс элемента по всей матрице*

**def** straight\_run\_max(matrix, b):

   for nrow in range(len(matrix)): *# nrow равен номеру строки*

       mrow, mcol = max\_element(matrix, nrow)

       if mrow != nrow:

           matrix[[nrow, mrow]] = matrix[[mrow, nrow]] *# swap*

           change = b[mrow]

           b[mrow] = b[nrow]

           b[nrow] = change

       swap\_columns(matrix, nrow, mcol)

       row = matrix[nrow]

       divider = row[nrow]  *# диагональный элемент*

       if abs(divider) < 1e-10:

*# почти нуль на диагонали.*

*Продолжать не имеет смысла, результат счёта неустойчив*

           print(f"Matrix is incompatible. Max element in matrix: {divider:.3g}")

           exit()

       row /= divider *# делим на диагональный элемент.*

       b[nrow] /= divider

       bfactor = b[nrow]

*# теперь надо вычесть приведённую строку из всех нижележащих строчек*

       for lower\_row in range(nrow + 1, len(matrix)):

           factor = matrix[lower\_row][nrow]

*# элемент строки в колонке nrow*

           matrix[lower\_row] -= factor \* row

*# вычитаем, чтобы получить ноль в колонке nrow*

           b[lower\_row] -= factor \* bfactor

**def** max\_element(A, k):

   maximum = A[k-1][k-1]

   max\_index = [k-1, k-1]

   for i in range(k-1, len(A)):

       for j in range(k-1, len(A)):

           if maximum < A[i][j]:

               maximum = A[i][j]

               max\_index = [i, j]

   return max\_index

**def** gauss\_max(A, b):

   check\_zeros\_diag(A, b)

   straight\_run\_max(A, b)

   reverse\_run(A, b)

   for i in range(len(X)):

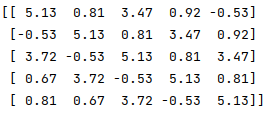
       print("%.4f" % X[i], end=" ")

   print('\n')

**Полученные результаты**

**Тестовый пример 1.**

Матрица А, полученная в результате вычисления A=14C+D

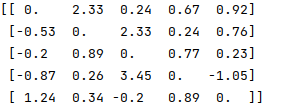
****

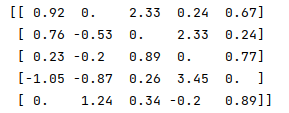
|  |  |  |  |
| --- | --- | --- | --- |
| Метод Гаусса | Метод Гаусса с выбором главного элемента по столбцу | Метод Гаусса с выбором главного элемента по всей матрице | Вычисление при помощи встроенной функции |
| 0.9395  0.8818  -0.2508  -0.0844  0.7284 | 0.9395  0.8818  -0.2508  -0.0844  0.7284 | 0.9395  0.8818  -0.2508  -0.0844  0.7284 | 0.9395  0.8818  -0.2508  -0.0844  0.7284 |
| 0.939526617559100830 0.881828624471086120  -0.250834802692053049  -0.084368248177271399 0.728372038179723957 | 0.939526617559100719 0.881828624471086120  -0.250834802692052827 -0.084368248177271399 0.728372038179723846 | 0.939526617559100719  0.881828624471086009 -0.250834802692052883  -0.084368248177271177  0.728372038179723846 | 0.939526617559100385 0.881828624471085787  -0.250834802692052494  -0.084368248177271024 0.728372038179723735 |

**Тестовый пример 2.**

Проверка наличия 0 на главной диагонали.

Исходная матрица:

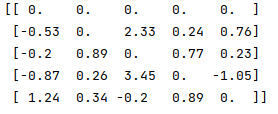
 Матрица после функции check\_zeros\_diag:

Как видно из примера, нулей на главной диагонали нет и получен ответ:

|  |  |
| --- | --- |
| Метод Гаусса | Вычисление при помощи встроенной функции |
| -0.1984  0.1424  1.1877  5.2080  0.1021 | -0.1984  0.1424  1.1877  5.2080  0.1021 |

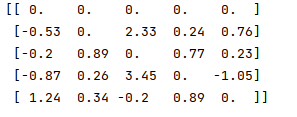
**Тестовый пример 3.**

Исходная матрица и вектор столбец:

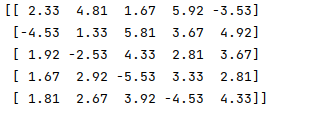
Получен ответ: The system has no solutions

**Тестовый пример 4.**

Исходная матрица и вектор столбец:

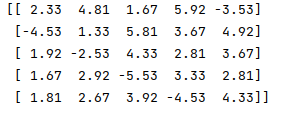
Получен ответ: The system has infinitive amount of solutions

**Тестовый пример 5.**

****

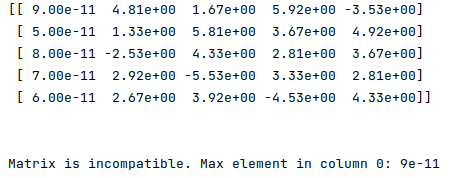
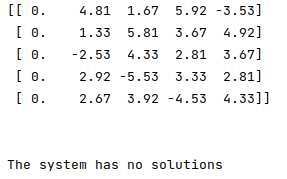
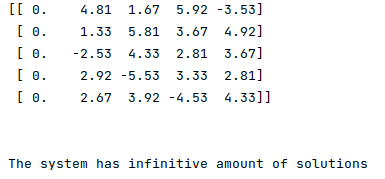
|  |  |
| --- | --- |
| Метод Гаусса | Вычисление при помощи встроенной функции |
| 64.2458  61.2394  28.0797  47.7166  76.3256 | 64.2458  61.2394  28.0797  47.7166  76.3256 |
| 64.24576835257857965189 61.23936912266844956321 28.07968852883874433246 47.71661005510677000530 76.32561391244648518750 | 64.24576835257855123018 61.23936912266852772291 28.07968852883877275417 47.71661005510672026730 76.32561391244649939836 |

**Тестовый пример 6.**

****

|  |  |
| --- | --- |
| Метод Гаусса | Вычисление при помощи встроенной функции |
| 0.0006  0.0006  0.0003  0.0005  0.0008 | 0.0006  0.0006  0.0003  0.0005  0.0008 |
| 0.00064245768352578581 0.00061239369122668454 0.00028079688528838767 0.00047716610055106748 0.00076325613912446458 | 0.00064245768352578570 0.00061239369122668497 0.00028079688528838783 0.00047716610055106726 0.00076325613912446501 |

**Тестовый пример 7.**

**Тестовый пример 8.**

**Тестовый пример 9.**

Малые изменения входных данных:



Ответ: 5.000000000010362378 2.000000000004440892



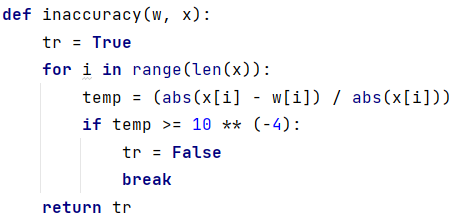
Ответ: 0.33333333333333331483 0.00000000000000000000



Ответ: Singular matrix

# **Оценка**

Следующая функция производит проверку на допустимость погрешности, однако для полной уверенности посчитаем относительную погрешность вручную.



0,2029-0,2029398289=0,0000398289 0,0000024091

# **Выводы**

Таким образом, в ходе выполнения лабораторной работы был применён метод Гаусса, метод Гаусса с выбором главного элемента по столбцу (схема частичного выбора) и метод Гаусса с выбором главного элемента по всей матрице (схема полного выбора) для решения системы линейных уравнений, рассмотрены решения СЛАУ методом Гаусса на конкретном примере,составлены алгоритмы и созданы реализации соответствующих программ на языке Python для решения поставленной задачи, также проведена оценка и проверена правильность работы программы.

Итак, метод Гаусса применим к любой системе линейных уравнений,

он идеально подходит для решения систем, содержащих больше трех

линейных уравнений. Метод Гаусса решения СЛАУ с числовыми

коэффициентами в силу простоты и однотипности выполняемых операций

пригоден для счета на электронно-вычислительных машинах.

Достоинства метода:

1. менее трудоёмкий по сравнению с другими методами;

2. позволяет однозначно установить, совместна система или нет, и если

совместна, найти её решение;

3. позволяет найти максимальное число линейно независимых

уравнений – ранг матрицы системы.

Существенным недостатком этого метода является невозможность

сформулировать условия совместности и определенности системы в

зависимости от значений коэффициентов и свободных членов. С другой

стороны, даже в случае определенной системы этот метод не позволяет найти общие формулы, выражающие решение системы через ее коэффициенты и свободные члены, которые необходимо иметь при теоретических исследованиях.