

1. Введение

1) Необходимо создать программу решающую системы линейных алгебраических уравнений методом Гаусса.

2) Пример кода, решающего данную задачу:

```
x = [0] * n
for i in range(n - 1, -1, -1):
    x[i] = vector[i]
    for j in range(i + 1, n):
        x[i] -= matrix[i][j] * x[j]
    x[i] /= matrix[i][i]
```

3) Скриншот программы показан на стр. 3 (Рис.1)

2. Ход работы

2.1. Код приложения

```
def gauss_elimination(matrix, vector):
    n = len(matrix)
    for i in range(n):
        max_row = i
        for j in range(i + 1, n):
            if abs(matrix[j][i]) > abs(matrix[max_row][i]):
                max_row = j
        matrix[i], matrix[max_row] = matrix[max_row], matrix[i]
        vector[i], vector[max_row] = vector[max_row], vector[i]
        for j in range(i + 1, n):
            factor = matrix[j][i] / matrix[i][i]
            vector[j] -= factor * vector[i]
            for k in range(i, n):
                matrix[j][k] -= factor * matrix[i][k]

    x = [0] * n
    for i in range(n - 1, -1, -1):
        x[i] = vector[i]
        for j in range(i + 1, n):
            x[i] -= matrix[i][j] * x[j]
        x[i] /= matrix[i][i]

    return x
```

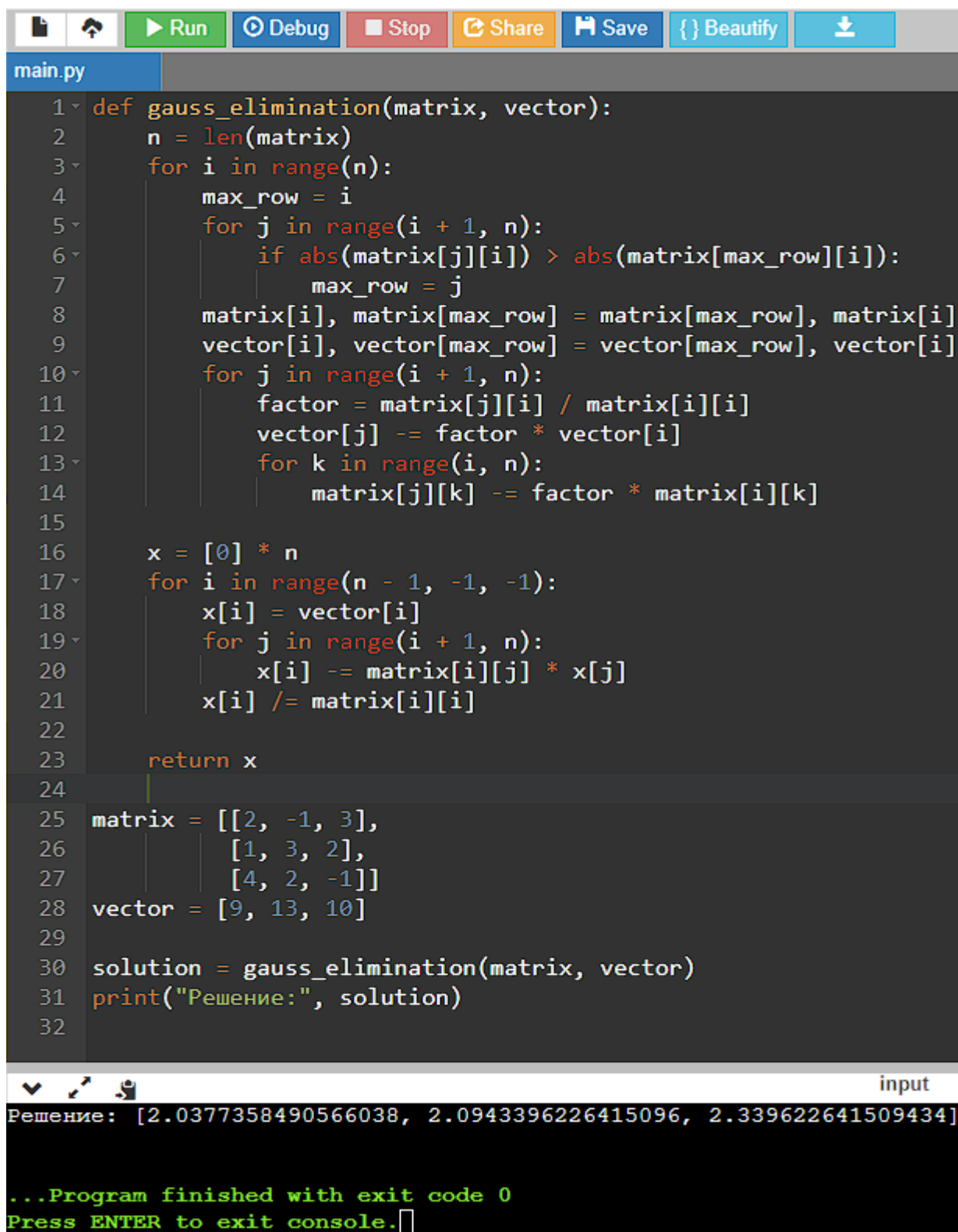
```
matrix = [[2, -1, 3],  
          [1, 3, 2],  
          [4, 2, -1]]  
vector = [9, 13, 10]  
  
solution = gauss_elimination(matrix, vector)  
print("Решение:", solution)
```

2.2. Алгоритм решения

Алгоритм решения СЛАУ методом Гаусса (Рис.2)

Список литературы

- [1] Рыжиков Ю.И. Вычислительные методы СПб.: БХВ-Петербург, 2007
- [2] Жидков Е.Н. Вычислительная математика - 2-е изд., перераб. - М.: Академия, 2013. - 208 с.



The image shows a screenshot of a Python IDE. At the top, there is a toolbar with icons for file operations and buttons for 'Run', 'Debug', 'Stop', 'Share', 'Save', 'Beautify', and a download icon. Below the toolbar, the file name 'main.py' is displayed. The main area contains Python code for a Gauss elimination algorithm. The code defines a function 'gauss_elimination' that takes a matrix and a vector as input. It performs row operations to solve a system of linear equations. After the function definition, the matrix and vector are initialized with specific values, and the function is called to find the solution. The output is printed in Russian. At the bottom, there is a console window showing the output of the program and a message indicating that the program finished successfully.

```
1 def gauss_elimination(matrix, vector):
2     n = len(matrix)
3     for i in range(n):
4         max_row = i
5         for j in range(i + 1, n):
6             if abs(matrix[j][i]) > abs(matrix[max_row][i]):
7                 max_row = j
8         matrix[i], matrix[max_row] = matrix[max_row], matrix[i]
9         vector[i], vector[max_row] = vector[max_row], vector[i]
10        for j in range(i + 1, n):
11            factor = matrix[j][i] / matrix[i][i]
12            vector[j] -= factor * vector[i]
13            for k in range(i, n):
14                matrix[j][k] -= factor * matrix[i][k]
15
16        x = [0] * n
17        for i in range(n - 1, -1, -1):
18            x[i] = vector[i]
19            for j in range(i + 1, n):
20                x[i] -= matrix[i][j] * x[j]
21            x[i] /= matrix[i][i]
22
23        return x
24
25 matrix = [[2, -1, 3],
26           [1, 3, 2],
27           [4, 2, -1]]
28 vector = [9, 13, 10]
29
30 solution = gauss_elimination(matrix, vector)
31 print("Решение:", solution)
32
```

input

Решение: [2.0377358490566038, 2.0943396226415096, 2.339622641509434]

...Program finished with exit code 0
Press ENTER to exit console.

Рис. 1. Скриншот программы

В простейшем случае алгоритм выглядит так:

$$\begin{cases} a_{11} \cdot x_1 + a_{12} \cdot x_2 + \dots + a_{1n} \cdot x_n = b_1 & (1) \\ a_{21} \cdot x_1 + a_{22} \cdot x_2 + \dots + a_{2n} \cdot x_n = b_2 & (2) \\ \dots & \\ a_{m1} \cdot x_1 + a_{m2} \cdot x_2 + \dots + a_{mn} \cdot x_n = b_m & (m) \end{cases}$$

Прямой ход:

$$\begin{aligned} (2) &\rightarrow (2) - (1) \cdot \left(\frac{a_{21}}{a_{11}}\right) & : & \quad a'_{22} \cdot x_2 + a'_{23} \cdot x_3 + \dots + a'_{2n} \cdot x_n = b'_2 \\ (3) &\rightarrow (3) - (1) \cdot \left(\frac{a_{31}}{a_{11}}\right) & : & \quad a'_{32} \cdot x_2 + a'_{33} \cdot x_3 + \dots + a'_{3n} \cdot x_n = b'_3 \\ &\dots & & \\ (m) &\rightarrow (m) - (1) \cdot \left(\frac{a_{m1}}{a_{11}}\right) & : & \quad a'_{m2} \cdot x_2 + a'_{m3} \cdot x_3 + \dots + a'_{mn} \cdot x_n = b'_m \\ (3) &\rightarrow (3) - (2) \cdot \left(\frac{a'_{32}}{a'_{22}}\right) & : & \quad a''_{33} \cdot x_3 + \dots + a''_{3n} \cdot x_n = b''_3 \\ &\dots & & \\ (m) &\rightarrow (m) - (m-1) \cdot \left(\frac{a'_{(m-2)n} - 1}{a'_{(m-2)n-1}}\right) & : & \quad a^{(m-1)}_{m2} \cdot x_2 + \dots + a^{(m-1)}_{mn} \cdot x_n = b^{(m-1)}_m \end{aligned}$$

Обратный ход. Из последнего ненулевого уравнения выражаем базисную переменную через небазисные и подставляем в предыдущие уравнения. Повторяя эту процедуру для всех базисных переменных, получаем фундаментальное решение.

Рис. 2. Алгоритм решения СЛАУ