**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение высшего образования «Российский химико-технологический университет**

**имени Д.И. Менделеева»**

Факультет цифровых технологий и химического инжиниринга

Кафедра информационных компьютерных технологий

**ОТЧЕТ**

**ПО КУРСОВОЙ РАБОТЕ**

Заведующая кафедрой ИКТ,

д.т.н., профессор **Кольцова Э.М.**

Руководитель работы

Старший преподаватель **Васецкий А.М.**

**СТУДЕНТ группы КС-28**  **Акинин Г.Д.**

**Москва**

**2022**

**Оглавление**

[Введение 2](#_30j0zll)

[Реализация 3](#_2et92p0)

[Вывод](#_kq94dxrzanq9) 8

[Литература 8](#_ofrbxjnl412x)

# **Введение**

Задание для курсовой работы было следующим:

1. Решение систем линейных уравнений методом Гаусса. Проверить точность расчётов.
2. По имеющейся матрице построить поверхность функции с форматированием легенды (наличие/отсутствие, цвет текста). Необходимые значения Х и Y, требуемые для построения поверхностной диаграммы задать самостоятельно с нецелым шагом.
3. Предусмотреть вариант считывания исходных данных из выбираемого пользователем файла последовательного доступа. Запись результатов и исходных данных в итоговый файл.

# 

# **Реализация**

Для решения поставленной задачи был выбран язык программирования Python, а само взаимодействие с графиками выполнялось с помощью библиотеки matplotlib.

В начале я импортировал библиотеку numpy для проверки результата и файл Number2.py, в котором хранится код для создания графика.

**import numpy as np**

**import Number2 as kurs**

Далее предоставляю выбор метода считывания начальных данных (1- из текстовых файлов matrixA и matrixB, 2 - заранее заданная матрицы).

**print("Выберите метод считывания данных.\n1. Из текстового файла\n2. Из программы")**

**n3 = input()**

**if n3 == "1":**

**with open('matrixA.txt') as f:**

**myA = [list(map(float, row.split())) for row in f.readlines()]**

**with open("matrixB.txt") as f:**

**for line in f:**

**myB = [float(x) for x in line.split()]**

**elif n3 == "2":**

**myA = [**

**[1.0, -2.0, 3.0, -4.0],**

**[3.0, 3.0, -5.0, -1.0],**

**[3.0, 0.0, 3.0, -10.0],**

**[-2.0, 1.0, 2.0, -3.0]]**

**myB = [**

**2.0,**

**-3.0,**

**8.0,**

**5.0]**

**else:**

**print("Введено неверное значение.")**

**quit()**

Преобразую данные в отдельные матрицы для дальнейшей проверки результата.

**myA1 = np.array(myA, int)**

**myB1 = np.array([[myB[0]], [myB[1]], [myB[2]], [myB[3]]], int)**

Функция вывода системы на экран:

**def FancyPrint(A, B, selected):**

**for row in range(len(B)):**

**print("(", end='')**

**for col in range(len(A[row])):**

**print("{1:6.2f}{0}".format(" " if (selected is None or selected != (row, col)) else "\*", A[row][col]), end='')**

**print(f") \* (X{row + 1}) = ({B[row]:6.2f})")**

Функция перемены двух строк массива местами:

**def SwapRows(A, B, row1, row2):**

**A[row1], A[row2] = A[row2], A[row1]**

**B[row1], B[row2] = B[row2], B[row1]**

Функция деления строки системы на число:

**def DivideRow(A, B, row, divider):**

**A[row] = [a / divider for a in A[row]]**

**B[row] /= divider**

Функция сложения строки системы с другой строкой умноженной на число:

**def CombineRows(A, B, row, source\_row, weight):**

**A[row] = [(a + k \* weight) for a, k in zip(A[row], A[source\_row])]**

**B[row] += B[source\_row] \* weight**

Функция проверки результата:

**def exam(A, B, X):**

**X = np.array([[X[0]], [X[1]], [X[2]], [X[3]]], int)**

**res = A.dot(X)**

**print("Проверка:")**

**for i in range(len(res)):**

**print(f"{res[i]} = {B[i]} - {(res[i] == B[i])}")**

Функция записи результата в txt файл:

**def rec(X):**

**file = open("result.txt", "w")**

**for i in range(len(X)):**

**file.write(f"{X[i]:.2}\n")**

**file.close()**

Основное тело кода выполняющее решение системы методом Гаусса (приведением к треугольному виду). А также проверяет результат и записывает его в файл.

**def Gauss(A, B):**

**column = 0**

**while column < len(B):**

**print("Ищем максимальный по модулю элемент в {0}-м столбце:".format(column + 1))**

**current\_row = None**

**for r in range(column, len(A)):**

**if current\_row is None or abs(A[r][column]) > abs(A[current\_row][column]):**

**current\_row = r**

**if current\_row is None:**

**print("решений нет")**

**return None**

**FancyPrint(A, B, (current\_row, column))**

**if current\_row != column:**

**print("Переставляем строку с найденным элементом повыше:")**

**SwapRows(A, B, current\_row, column)**

**FancyPrint(A, B, (column, column))**

**print("Нормализуем строку с найденным элементом:")**

**DivideRow(A, B, column, A[column][column])**

**FancyPrint(A, B, (column, column))**

**print("Обрабатываем нижележащие строки:")**

**for r in range(column + 1, len(A)):**

**CombineRows(A, B, r, column, -A[r][column])**

**FancyPrint(A, B, (column, column))**

**column += 1**

**print("Матрица приведена к треугольному виду, считаем решение")**

**X = [0 for b in B]**

**for i in range(len(B) - 1, -1, -1):**

**X[i] = B[i] - sum(x \* a for x, a in zip(X[(i + 1):], A[i][(i + 1):]))**

**print("Получили ответ:")**

**print("\n".join("X{0} = {1:.2f}".format(i + 1, x) for i, x in**

**enumerate(X)))**

**exam(myA1, myB1, X)**

**rec(X)**

**return X**

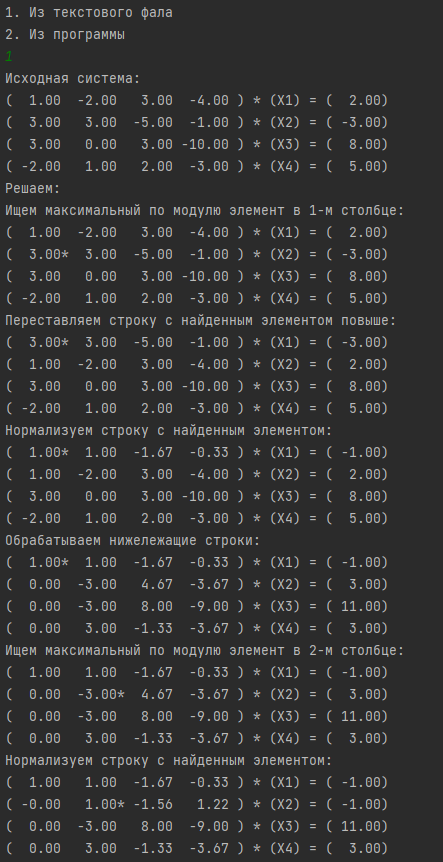
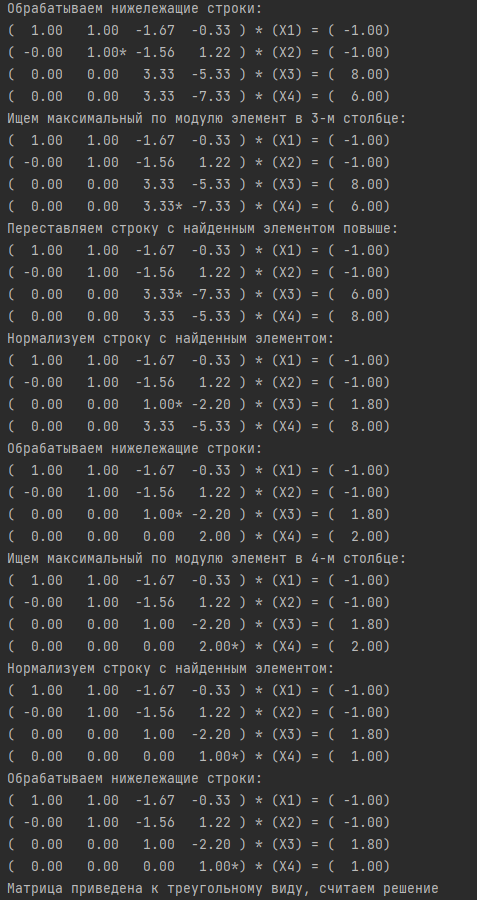
**print("Исходная система:")**

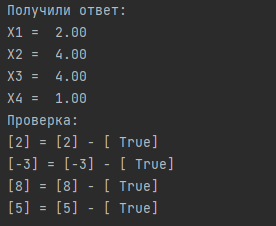
**FancyPrint(myA, myB, None)**

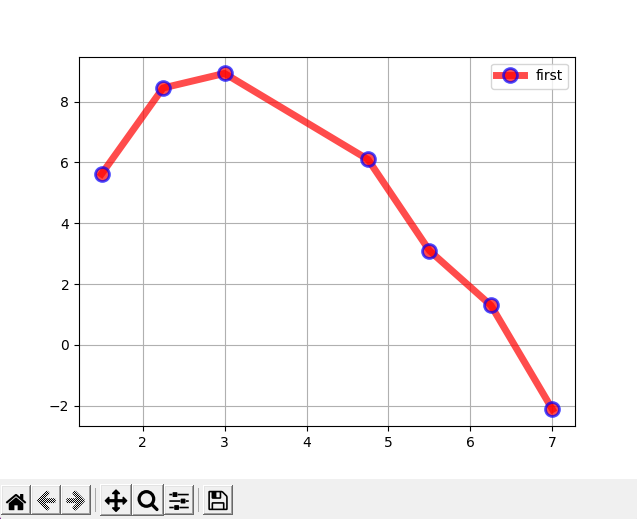
**print("Решаем:")**

**Gauss(myA, myB)**

**kurs.graf() # Построение графика(номер 2)**

Вывод программы:



Сам интерфейс выглядит следующим образом: на вкладке мы видим сам график, а также инструменты для изменения масштаба и сохранения графика в файл.

# **Вывод**

В данной работе я работал с библиотекой matplotlib и я научился работе с графиками в Python. Эта работа позволила мне как вводить и форматировать данные в графике, так и сохранять и удалять их.

# **Литература**

1. А. В. Пантелеев, Т. А. Летова Методы оптимизации в примерах и задачах. Прикладная математика для ВТУЗов, Высшая школа, 2008
2. В. М. Вержбицкий, Основы численных методов, Высшая школа, 2005 г.
3. А.А. Амосов, Ю.А. Дубинский, Н.В. Копченова, Вычислительные методы для инженеров, Высшая школа, 1994
4. Справочник Python [Электронный ресурс]. - Режим доступа: https://www.python.org/ (дата обращения 20.05.2022)
5. П.Н. Коробов. Математическое программирование и моделирование экономических процессов. – М.: ДНК, 2006. – 376 с.
6. И.Г. Семакин, А.П. Шестаков. Основы алгоритмизации и программирования. – М.: Академия, 2012. – 400 с.
7. А.И. Заковряшин. Алгоритмизация и программирование вычислительных задач. – М.: Science Press, 2002. – 80 с.
8. П.Б. Хорев. Объектно-ориентированное программирование. – М.: Академия, 2012. – 448 с.
9. И.Г. Семакин, А.П. Шестаков. Основы алгоритмизации и программирования. – М.: Академия, 2013. – 144 с.
10. С.А. Канцедал. Алгоритмизация и программирование. – М.: Форум, Инфра-М, 2008. – 352 с.
11. О.Л. Голицына, И.И. Попов. Программирование на языках высокого уровня. – М.: Форум, 2010. – 496 с.
12. Ф.П. Васильев, А.Ю. Иваницкий. Линейное программирование. – М.: Факториал Пресс, 2008. – 352 с.