МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Национальный исследовательский**

**Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

**Кафедра математического обеспечения и суперкомпьютерных технологий**

**Отчет по учебной практике**

**«Структура хранения матриц специального вида»**

**Выполнил:** студент группы 381706-1

Голованова Елена Александровна

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ

Лебедев Илья Геннадьевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись

Нижний Новгород

2018.

Оглавление

[1. Введение 3](#_Toc1954492)

[2. Постановка задачи 4](#_Toc1954493)

[3. Руководство пользователя 5](#_Toc1954494)

[4. Руководство программиста 6](#_Toc1954495)

[a. Описание структуры программы 6](#_Toc1954496)

[b. Описание структур данных 6](#_Toc1954497)

[c. Описание алгоритмов 8](#_Toc1954498)

[5. Заключение 10](#_Toc1954499)

[6. Список литературы 11](#_Toc1954500)

# Введение

**Матрица –** математический объект, записываемый в виде прямоугольной таблицы.

Они бывают разного вида. В данной лабораторной работе мы рассмотрим верхнетреугольную матрицу. Они имеют более специфичную структуру и, соответственно, для них необходим специальный способ хранения для избежания хранения в памяти неиспользующихся ячеек (к примеру, если диагональную матрицу хранить как прямоугольную).

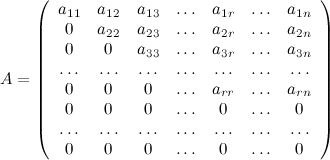


Рисунок 1. Верхнетреугольная матрица

Цель данной лабораторной работы – разработать структуру хранения верхнетреугольных матриц.

# Постановка задачи

Выполнение работы предполагает решение следующих задач:

1. Разработка и реализация вспомогательного класса TVector.
2. Разработка и реализация класса матриц TMatrix.
3. Выполнение определенных функций, операторов и конструкторов в ранее упомянутых классах.
4. Создать класс TException для избежания ошибок.
5. Выполнение примера использования программы.
6. Реализация простых тестов на базе Google Test.

# Руководство пользователя

Работа пользователя происходит так:

1. На экран выводятся автоматически созданные матрицы.
2. Далее появляются результаты арифметических операций.
3. После чего пользователю предлагается самому ввести верхнетреугольную матрицу.
4. Эта матрица выводится на экран.

# 4. Руководство программиста

## Описание структуры программы

Для реализации лабораторной работы создается несколько модулей:

* Модули Vector и Matrix.

Содержат примеры реализаций программ с использованием классов TVector и TMatrix соответственно.

* Модуль VectorLib и MatrixLib.

Содержат соответственно файлы Vector.h и Matrix.h, в котором определены интерфейс и реализация шаблонных классов TVector и TMatrix.

* Модули VectorTest и MatrixTest.

Эти модули содержат тесты, проверяющие работу каждого метода класса TMatrix и TVector.

* Модуль Exception

Этот модуль содержит класс исключений.

## Описание структур данных

Класс TVector - шаблонный.

1. В зоне protected определены два поля:

* T \* vec - указатель на вектор;
* int size - размер вектора;

2. В зоне public:

* TVector(int s = 0) – конструктор с параметром;
* TVector(const TVector &v) – конструктор копирования;
* virtual ~TVector() - деструктор;
* int GetSize() const - размер вектора;
* virtual T& operator[](int pos) – доступ;
* bool operator==(const TVector &v) const – сравнение;
* bool operator!=(const TVector &v) const – сравнение;
* virtual TVector& operator=(const TVector &v) - присваивание;
* TVector operator+(const T &val) - прибавить скаляр;
* TVector operator-(const T &val) - вычесть скаляр;
* TVector operator\*(const T &val) - умножить на скаляр;
* TVector operator+(const TVector &v) - сложение;
* TVector operator-(const TVector &v) - вычитание;
* T operator\*(const TVector &v) - скалярное произведение;
* template <class ValType1>;
* friend istream& operator>>(istream &in, TVector<ValType1>& v);
* template <class ValType1>;
* friend ostream& operator<<(ostream &out, const TVector<ValType1>& v);

Класс TMatrix – класс шаблонный. Наследуется от класса TMonom.

1. В зоне public определены:

* TMatrix(int s = 10) – конструктор;
* TMatrix(const TMatrix &mt) – конструктор копирования;
* TMatrix(const TVector<TVector<T> > &mt);
* virtual ~TMatrix() - деструктор;
* bool operator==(const TMatrix &mt) const – сравнение;
* bool operator!=(const TMatrix &mt) const – сравнение;
* TMatrix& operator=(const TMatrix &mt) - присваивание;
* TMatrix operator+(const TMatrix &mt) - сложение;
* TMatrix operator-(const TMatrix &mt) - вычитание;
* TMatrix<T> operator\*(const TMatrix<T> &MT) - умножение;
* TMatrix<T> operator/(const TMatrix<T> &MT) - деление;
* template <class ValType2>;
* friend istream& operator>>(istream &in, TMatrix<ValType2> &mt);
* template <class ValType2>;
* friend ostream& operator<<(ostream &out, const TMatrix<ValType2> &mt);

## Описание алгоритмов

* Умножение верхнетреугольных матриц:

Произведением матриц А и B является такая матрица C = AB, у которой элемент , стоящий в *i*-ой строке и *j*-ом столбце, равен сумме произведений элементов *i*-ой строки матрицы А на соответствующие элементы *j*-го столбца матрицы B.

Реализация перегрузки оператора умножения представлена, с помощью трех вложенных циклов: по строкам первой матрицы, по столбцам второй матрицы и по элементам текущего столбца второй матрицы.

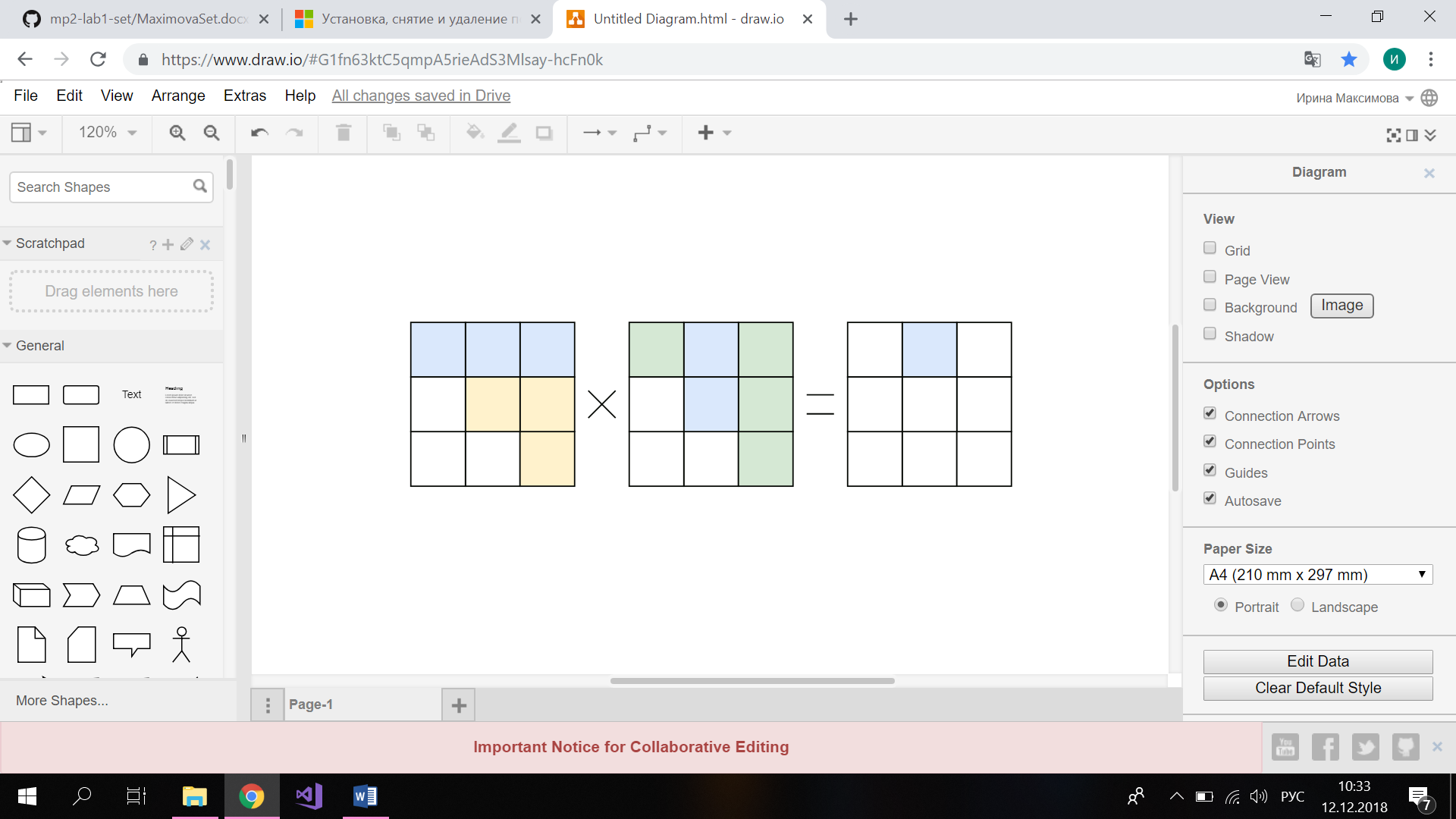


Рисунок 2. Умножение верхнетреугольных матриц

Деление верхнетреугольных матриц:

Делением матриц D и A является такая матрица C = D\*, где - обратная матрица к матрице A. Обратная матрица может быть найдена методом Гаусса: Записываем расширенную матрицу :

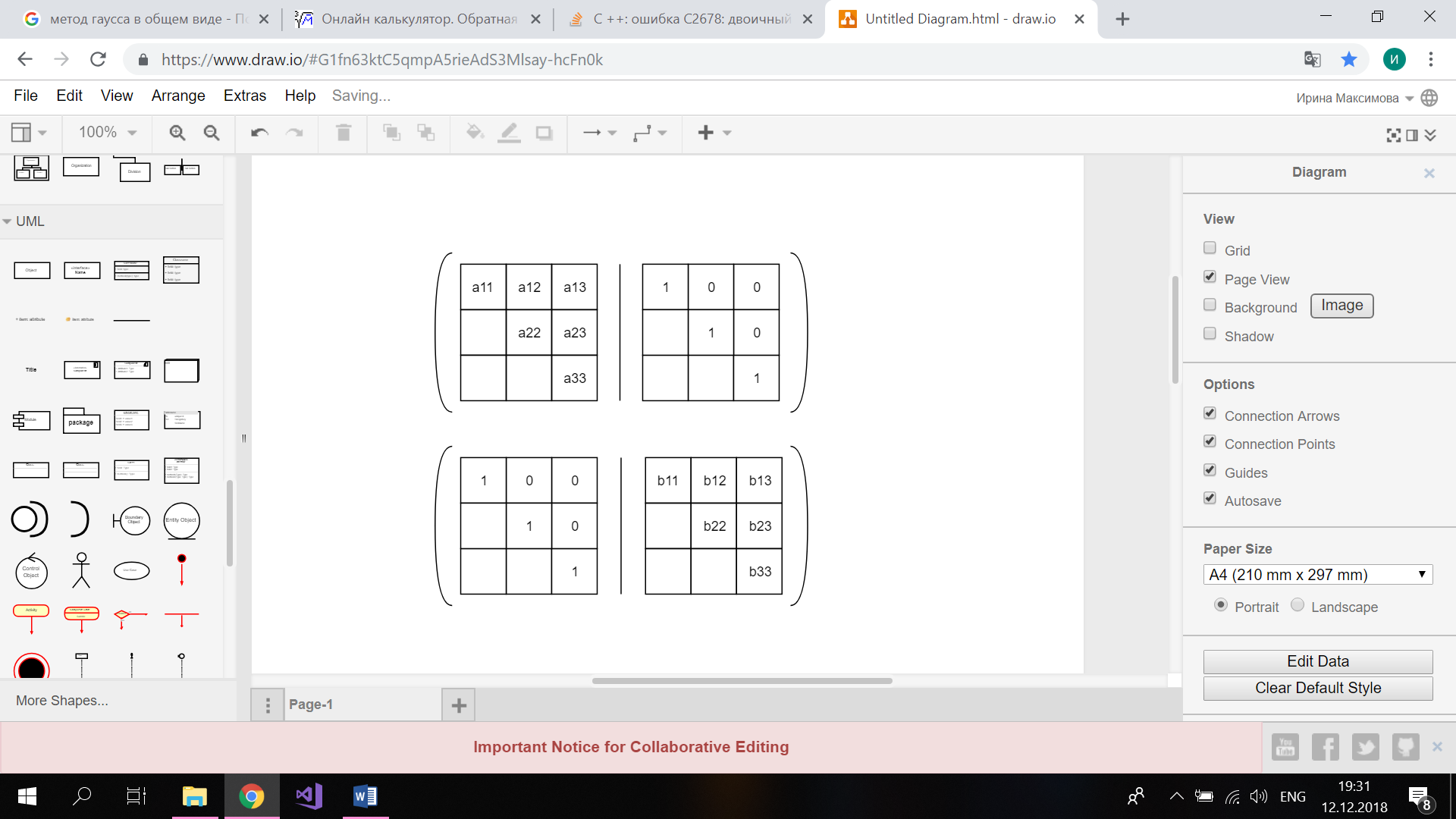


Рисунок 3. Расширенная матрица

Далее с помощью элементарных преобразований строк матрицы справа от черты получаем единичную матрицу, а то что получается справа и есть искомая обратная матрица :

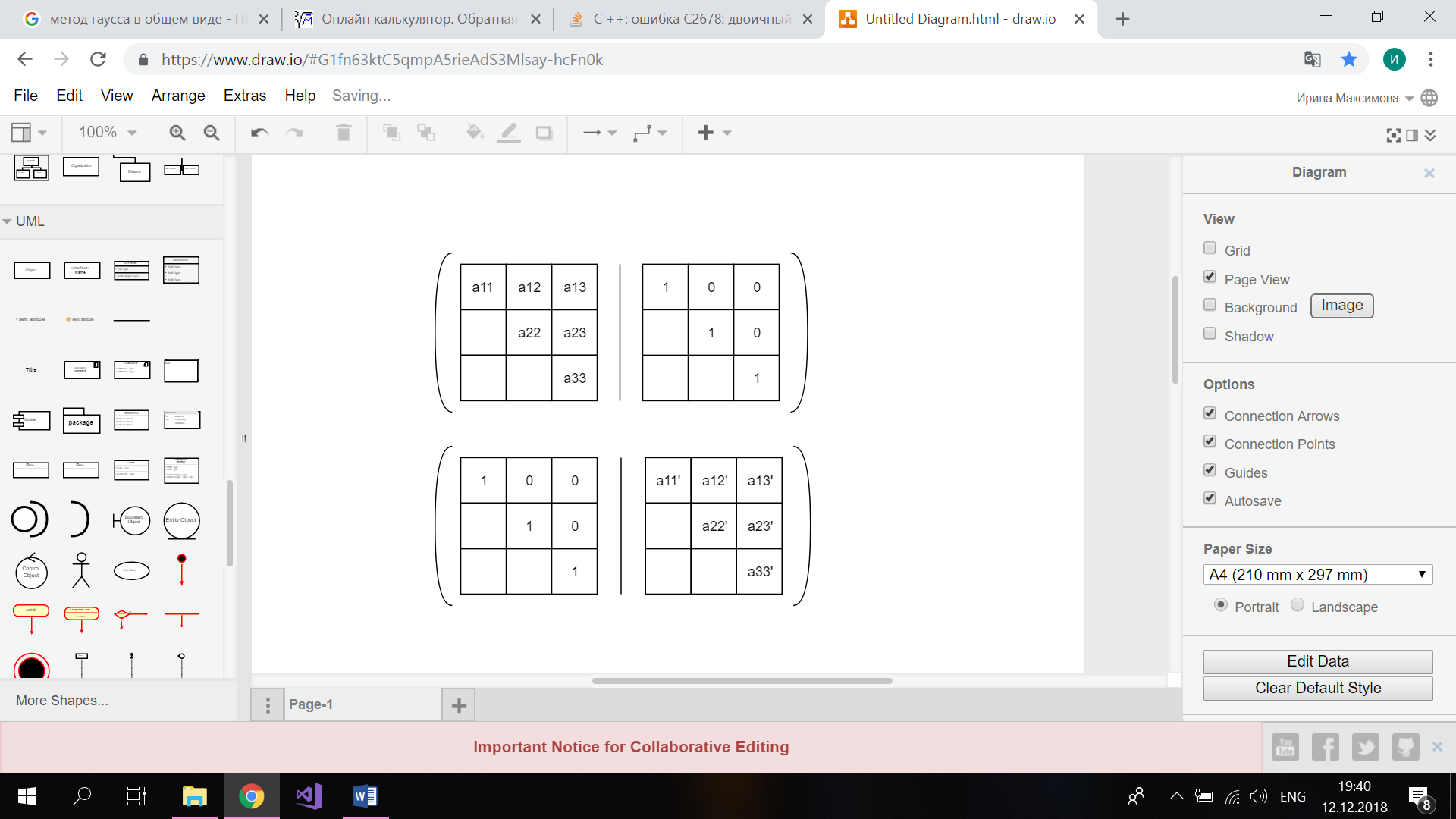


Рисунок 4. Конечная таблица метода Гаусса ))

# 5. Заключение

В ходе выполнения лабораторной я смогла создать такую структуру как верхнетреугольную матрицу и реализовать множество операций, связанных с ней. Она реализована с помощью вспомогательного класса TVector.

Получены новые знания и навыки.

Были реализованы тесты для проверки работоспособности вышеперечисленных классов на базе GoogleTest.

# 6. Список литературы

1. Гергель В.П. Методические материалы по курсу «Методы программирования 2»:

[<http://www.itmm.unn.ru/files/2018/10/Primer-1.2.-Struktury-hraneniya-matrits-spetsialnogo-vida.pdf>], 2015.

1. Википедия: свободная электронная энциклопедия: на русском языке [https://ru.wikipedia.org/wiki/Матрица\_(математика)]

[https://ru.wikipedia.org/wiki/Треугольная\_матрица]