

```

/**
 * Класс для файла
 */
public class FileManager {
    /**
     * Конструктор класса
     * @param args Путь к файлу
     */
    public FileManager(String[] args){
        // Инициализировать путь к файлу
    }

    /**
     * Чтение матрицы из файла.
     * @return Матрица
     */
    public Float[][] readFile(){
        //Читать числа из файла, и сохранить в массив, возвращать тот массив.
    }

    /**
     * Запись матрицы в файл
     * @param matrix, который нужно сохранить в файл.
     */
    public void writeFile(Float[][] matrix){
        // Записывать результат в файл.
    }
}

```

```

/**
 * Класс матрицы
 */
public class Matrix {
    /**
     * Хранилище матрицы
     */
    private final Float[][] matrix;
    /**
     * Размер матрицы
     */
    private final int size;
    /**
     * Конструктор класса
     * @param matrix, которые получить из файла
     */
    public Matrix(Float [][] matrix){
        // Инициализировать матрицу и ее размер
    }
    /**
     * Хранение результат
     */
    private final Float [] x = new Float[3];

    /**
     * Вычислить результат
     */
    public void solve(){

```

```

        // Нужно использовать вложенный цикл. Цикл продолжается от окончания
матрицы до начала.
        // Найти последний элемент, т.е X_n. А потом его умножить на
соответствующее число верной строки и сохранить.
        // Вычитать последний элемент строки от предыдущего полученного
результата и делить на текущий результат
        // Найти результат и сохранить его в отдельном массиве.
    }
    /**
     * Приведение матрицы к треугольному виду
     */
    public void transform(){
        // Нужно использовать вложенный цикл. Цикл продолжается до приведения
матрицы к треугольному виду
        // Вычислить коэффициент с помощью числа, которые расположены на одной
столбце.

        // Сложить две строки.
        // Распечатать результат после каждого шага.
    }
    /**
     * Сложение две строки.
     * @param row_1 Первое слагаемое
     * @param row_2 Второе слагаемое
     */
    private void addEquation(int row_1, int row_2){
        // Сложить две строки и сохранить в месте 2 строки.
    }
    /**
     * Умножить строку на коэффициент
     * @param coefficient коэффициент
     * @param row Строка
     */
    private void multiple(Float coefficient, int row){
        // Умножить все число в строке на коэффициент и сохранить в своем месте.
    }
    /**
     * Вычисление коэффициент
     * @param a Число, на которое должен умножить коэффициент.
     * @param b Число, который должен превратиться в нуль.
     * @return коэффициент, на которой нужно умножить строку.
     */
    private Float findCoefficient(Float a, Float b){
        // Если число a равно 0, то возвращает 0. возвращать значение -b/2.
    }
    /**
     * Метод, который проверяет размер матрицу
     * @return Если размер матрицы не удовлетворил требование, возвращает false.
     */
    private boolean checkSystem(){
        // Если размер меньше 2 или разность между размерами строки и столбца
больше чем 1 возвращает false
    }
    /**
     * Вычисление размер матрицы
     * @return Размер матрицы
     */
    private int size(){
    }

```

```
/**
 * Распечатать матрицу
 */
public void printSystem(){
}
/**
 * Распечатать результат
 */
public void printResult(){
}
}
```

```
/**
 * Main class
 */
public class App {
    /**
     * main метод
     * @param args путь к файлу.
     */
    public static void main(String[] args) {
        FileManager fileManager = new FileManager(args);
        Matrix matrix = new Matrix(fileManager.readFile());
        matrix.printSystem();
        matrix.transform();
        matrix.solve();
        matrix.printResult();
    }
}
```