**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ**

Лабораторная №1

**Метод Гаусса**

**Выполнил:**

Кендысь Алексей Максимович

студент 2 курса, 9 группы,

специальность

“прикладная математика”

**Преподавательница:**

Ассистентка кафедры вычислительной

математики ФПМИ,

Ю.Н. Горбачёва

Минск, 2021

**Содержание:**

Постановка задачи ------------------------------------------------------------------ 2

Краткие теоретические сведения ------------------------------------------------ 2

Листинг программы ---------------------------------------------------------------- 3-7

Результаты --------------------------------------------------------------------------- 8-9

Выводы ------------------------------------------------------------------------------- 9

1

**Постановка задачи**

Написать и отладить программу, реализующую метод Гаусса с выбором главного элемента по столбцу для численного решения систем линейных алгебраических уравнений Ax = f, вычисления обратной матрицы A-1. Предусмотреть сообщения, предупреждающие о невозможности решения указанных задач с заданной матрицей A порядка n.

**Краткие теоретические сведения**

Метод Гаусса (без выбора главного элемента по столбцу):

Прямой ход:

= , k, j = 1, n

= , k = 1, n

k = 1, n-1, j = k+1, n

=

= , i = k+1,n

=

= , i = k+1,n

=

Обратный ход:

=

= – , k = n-1, … ,1

2

**Листинг программы**

Файл Matrix.java:

import java.util.Random;  
import java.util.\*;  
public class Matrix {  
 private Float[][] a;  
 private final Float[] x;  
 private Float[] f;  
 private Float[] x0;  
 private Float[][] a1;  
 Matrix (int n) {  
 //Заполнение матрицы A  
 Random r = new Random();  
 a = new Float[n][n];  
 for (int i = 0; i < n; i++) {  
 for (int j = 0; j < n; j++) {  
 a[i][j] = r.nextFloat() \* 200 – 100;  
 }  
 }  
  
 //Заполнение векторов f и x, выделение памяти для x0  
 x = new Float[n];  
 for (int i = 0; i < n; i++) {  
 x[i] = (float) i + 1;  
 }  
 f = new Float[n];  
 f = multiply(a, x);  
 x0 = new Float[n];  
 }  
 public void outA() {  
 //Вывод матрицы A, outMatr – метод для вывода матриц  
 outMatr(a);  
 }  
 public void outX() {  
 //Вывод вектора x  
 Formatter fmt = new Formatter();  
 for (int i = 0; i < x.length; i++) {  
 fmt.format(“%.1f\n”, x[i]);  
 }  
 System.*out*.println(fmt);  
 fmt.close();  
 }  
 public void outF() {  
 //Вывод вектора f  
 Formatter fmt = new Formatter();  
 for (int i = 0; i < f.length; i++) {  
 fmt.format(“%f\n”, f[i]);  
 }  
 System.*out*.println(fmt);  
 fmt.close();  
 }  
 public void outX0() {  
 //Вывод вектора x0 (приближённого решения)  
 if (x0 == null) return;  
 Formatter fmt = new Formatter();  
 for (int i = 0; i < x0.length; i++) {  
 fmt.format(“%f\n”, x0[i]);  
 }

3

System.*out*.println(fmt);  
 fmt.close();  
 }  
 public void outError() {  
 //Вывод погрешности  
 if (x0 == null) return;  
 Formatter fmt = new Formatter();  
 fmt.format(“%.16f\n”, mesError(x, x0));  
 System.*out*.println(fmt);  
 fmt.close();  
 }  
 public void outA1() {  
 //Вывод матрицы A1  
 outMatr(a1);  
 }  
 public void outE() {  
 //Вывод матрицы A1\*A  
 outMatr(multiply(a1, a));  
 }  
 public void outTable() {  
 //Вывод таблицы точности решения СЛАУ разных порядков  
 Formatter fmt = new Formatter();  
 fmt.format("| Порядок матрицы | Относительная погрешность |\n");  
 for (int i = 5; i <= 105; i += 10) {  
 Matrix myMatrix = new Matrix(i);  
 fmt.format("|%10d |", i);  
 myMatrix.gauss(false);  
 fmt.format(" % 22.16f |\n", mesError(myMatrix.x, myMatrix.x0));  
 }  
 System.*out*.println(fmt);  
 fmt.close();  
 }

public void gauss(boolean doA1) {  
 //Метод Гаусса  
 //Значение doA1 – находить ли матрицу, обратную данной  
  
 //Создание копии матрицы A  
 Float[][] a2 = new Float[a.length][];  
 for(int i = 0; i < a.length; i++) {  
 a2[i] = new Float[a[i].length];  
 a2[i] = Arrays.*copyOf*(a[i], a[i].length);  
 }  
  
 //Заполнение матрицы A1  
 if (doA1) {  
 a1 = new Float[a.length][a.length];  
 for (int i = 0; i < a1.length; i++) {  
 for (int j = 0; j < a1.length; j++) {  
 if (i == j)  
 a1[i][j] = (float) 1;  
 else  
 a1[i][j] = (float) 0;  
 }  
 }  
 }  
  
  
 float max;  
 int kMax;  
 Float[] temp;  
 float temp2;

4

//Прямой ход  
 for (int i = 0; i < a.length; i++) {  
 //Поиск максимального по модулю элемента (выбор главного элемента по столбцу)  
 kMax = i;  
 max = 0;  
 for (int k = i; k < a.length; k++) {  
 if (Math.*abs*(a[k][i]) > Math.*abs*(max)) {  
 max = a[k][i];  
 kMax = k;  
 }  
 }  
 //Проверка на невозможность применения  
 if (max == 0.000001) {  
 x0 = null;  
 System.*out*.println(“Ошибка. Деление на ноль.”);  
 return;  
 }  
 //Перестановка строк  
 if (max != a[i][i]) {  
 temp = a[i];  
 a[i] = a[kMax];  
 a[kMax] = temp;

temp2 = f[i];  
 f[i] = f[kMax];  
 f[kMax] = temp2;  
 if (doA1) {  
 temp = a1[i];  
 a1[i] = a1[kMax];  
 a1[kMax] = temp;  
 }  
 }  
  
 //Деление на главный элемент  
 for (int j = i + 1; j < a.length; j++) {  
 a[i][j] /= a[i][i];  
 }  
 f[i] /= a[i][i];  
 if (doA1) {  
 for (int j = 0; j < a1.length; j++) {  
 a1[i][j] /= a[i][i];  
 }  
 }  
  
 //Обнуление столбца  
 for (int j = i + 1; j < a.length; j++) {  
 for (int k = i + 1; k < a.length; k++) {  
 a[j][k] -= a[j][i] \* a[i][k];  
 }  
 f[j] -= a[j][i] \* f[i];  
 if (doA1) {  
 for (int k = 0; k < a1.length; k++) {  
 a1[j][k] -= a[j][i] \* a1[i][k];  
 }  
 }  
 }  
 }  
  
 //Обратный ход  
 x0[a.length-1] = f[a.length-1];

5

for (int i = a.length – 2; i >= 0; i--) {

x0[i] = f[i];  
 for (int j = a.length – 1; j > i; j--) {  
 x0[i] -= a[i][j] \* x0[j];  
 if (doA1) {  
 for (int k = 0; k < a1.length; k++) {  
 a1[i][k] -= a[i][j] \* a1[j][k];  
 }  
 }  
 }  
 }  
  
 //Сохранение изначальной матрицы A (для операции A1\*A)  
 a = a2;  
 }  
  
 private void outMatr(Float[][] x1) {  
 //Метод вывода матриц  
 Formatter fmt = new Formatter();  
 for (int i = 0; i < x1.length; i++) {  
 for (int j = 0; j < x1[i].length; j++) {  
 fmt.format(“% -13.8f”, x1[i][j]);  
 }

fmt.format(“\n”);  
 }  
 System.*out*.println(fmt);  
 fmt.close();  
 }  
  
 private Float[][] multiply(Float[][] a, Float[][] b) {  
 //Метод умножения матриц  
 Float[][] res = new Float [a.length][b[0].length];  
 float c = 0;  
 for (int i = 0; i < a.length; i++) {  
 for (int j = 0; j < b[i].length; j++) {  
 for (int k = 0; k < a[i].length; k++) {  
 c += a[i][k] \* b[k][j];  
 }  
 res[i][j] = c;  
 c = 0;  
 }  
 }  
 return res;  
 }  
 private Float[] multiply(Float[][] x1, Float[] x2) {  
 //Метод умножения матрицы на столбец  
 Float[] res = new Float[x1.length];  
 float c = 0;  
 for (int i = 0; i < x1.length; i++) {  
 for (int j = 0; j < x2.length; j++) {  
 c += x1[i][j] \* x2[j];  
 }  
 res[i] = c;  
 c = 0;  
 }  
 return res;  
 }  
 private Float mesError(Float[] x1, Float[] x2) {  
 //Подсчет относительной погрешности  
 float max1 = 0;  
 float max2 = 0;

6

for (int i = 0; i < x1.length; i++) {  
 if (Math.*abs*(x1[i] – x2[i]) > max1) {  
 max1 = Math.*abs*(x1[i] – x2[i]);  
 }  
 if(Math.*abs*(x1[i]) > max2) {  
 max2 = Math.*abs*(x1[i]);  
 }  
 }  
 return (max1/max2);  
 }  
}

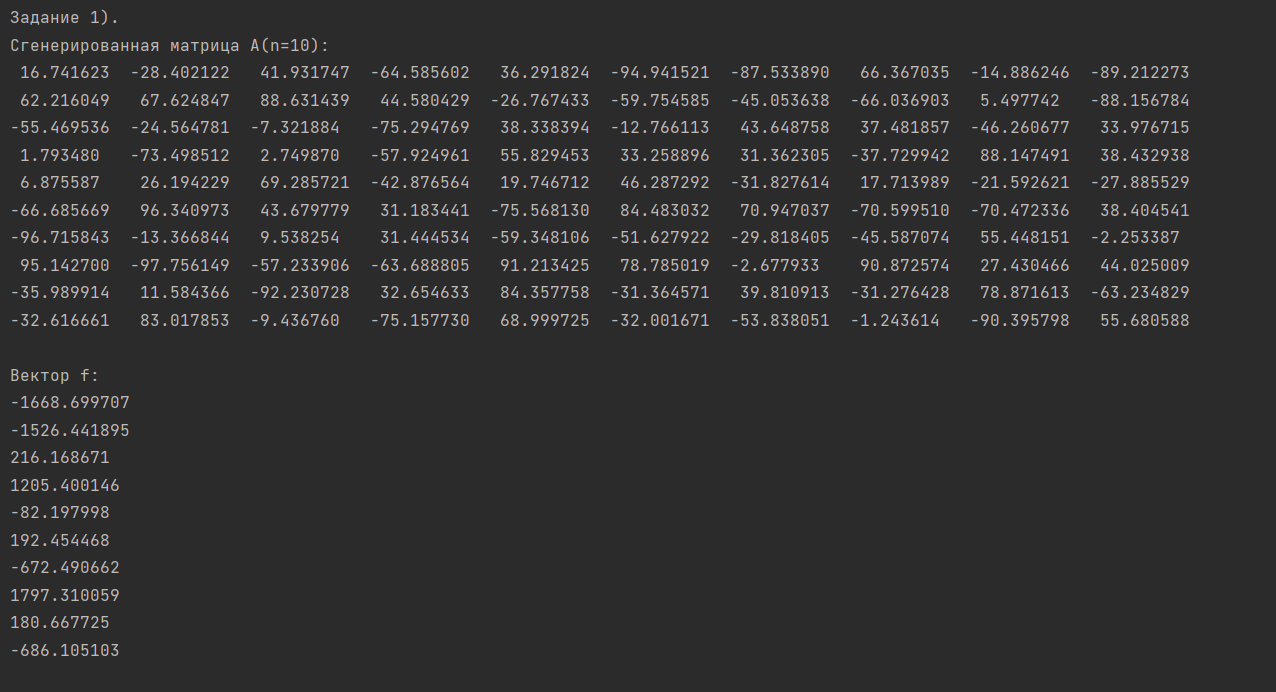
Файл Main.java:

import java.io.\*;  
public class Main {  
  
 public static void main(String[] args) throws IOException {  
 PrintWriter out = new PrintWriter(System.*out*, true);  
 Matrix myMatrix = new Matrix(10);  
 System.*out*.println(“Задание 1).”);  
 System.*out*.println(“Сгенерированная матрица A(n=10):”);  
 myMatrix.outA();  
 System.*out*.println(“Вектор f:”);

myMatrix.outF();  
 System.*out*.println(“Точное решение x:”);  
 myMatrix.outX();  
 myMatrix.gauss(true);  
 System.*out*.println(“Полученное приближённое решение x0:”);  
 myMatrix.outX0();  
 System.*out*.print(“Относительная погрешность = “);  
 myMatrix.outError();  
 System.*out*.println(“Задание 2).”);  
 System.*out*.println(“Обратная матрица A1:”);  
 myMatrix.outA1();  
 System.*out*.println(“Результат умножения A1\*A:”);  
 myMatrix.outE();  
 System.*out*.println(“Задание 3).”);  
 myMatrix.outTable();  
 }  
}

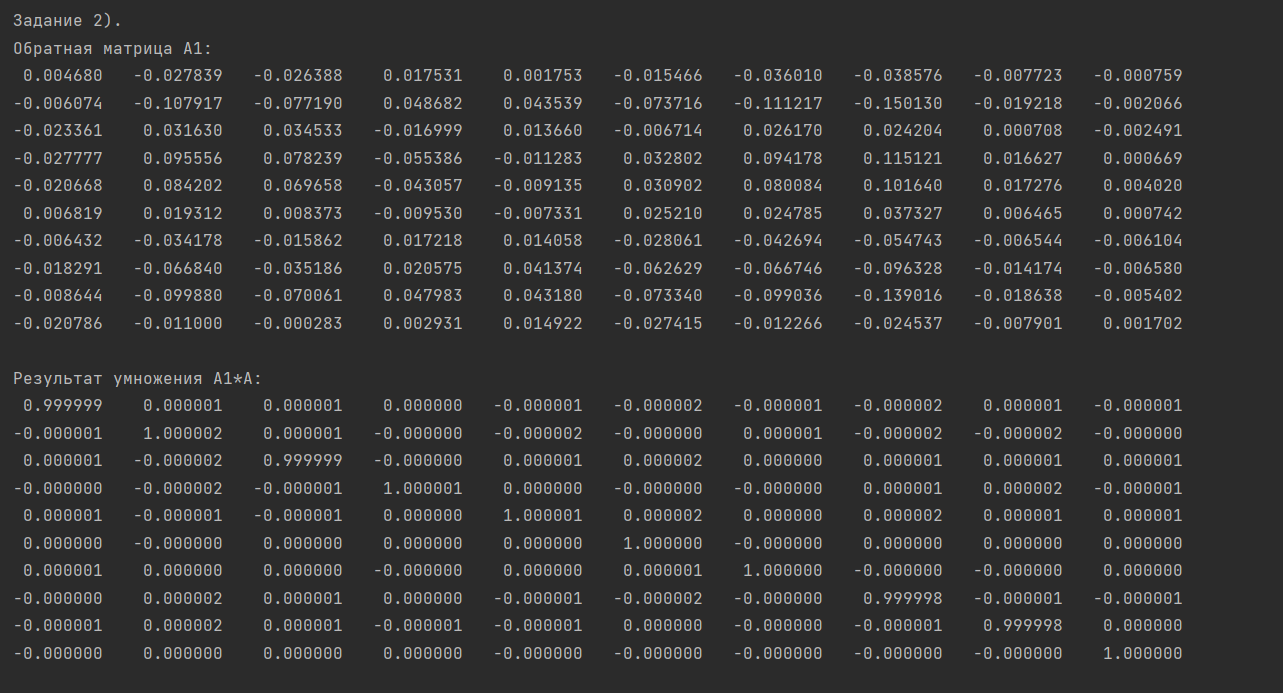
7

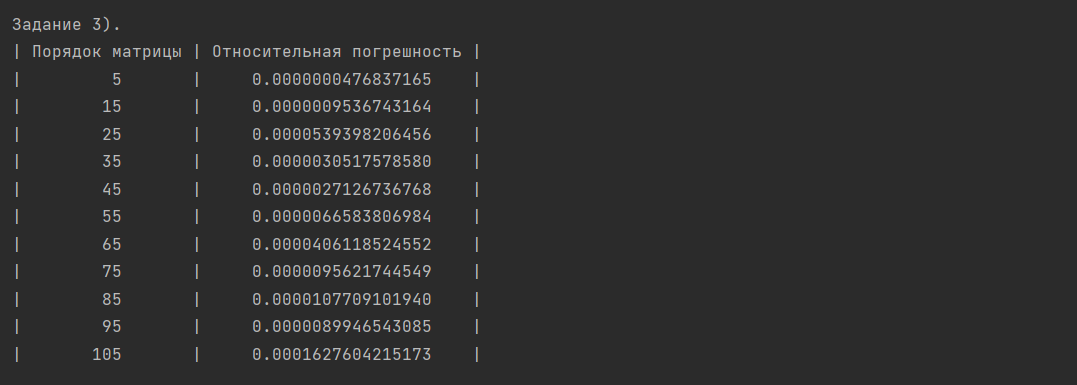
**Результаты**





8





**Выводы**

1) Из таблицы, полученной в задании 3, можно заметить, что в общем случае при возрастании порядка матрицы растёт относительная погрешность её решения.

2) По методу Гаусса требуется произвести операций умножения и деления.

3) Также стоит отметить, что с помощью метода Гаусса можно вычислить определитель матрицы:

detA = , где k – кол-во перестановок

9