ВМА. Лабораторная 1. Отчет

Гамезо Валерия

Задание 1. Разработать программу численного решения СЛАУ методом Гаусса без выбора ведущего элемента. Для выполнения прямого хода воспользоваться псевдокодом (6)–(8) на странице 3; для выполнения обратного хода воспользоваться формулами (9).

Матрицу (порядка n) системы Ax=b задать с диагональным преобладанием следующим образом:

1. Недиагональные элементы $a_{i,j}$, i $\neq$ j, выбираются из чисел 0, –1, –2, –3, –4 произвольным образом;
2. $a_{i,i} = -\sum\limits_{j=1, i\neq j}^{n} a_{i,j}, 2 \leq i \leq n$;
3. $a_{1,1} = -\sum\limits_{j=2}^{n} a_{1,j} + 10^{-k}, k \geq 0$;

Правую часть b задать умножением матрицы A на вектор x=(m, m+1, ... , n+m–1): b=Ax. Для вычислений выбрать параметры:

1. m – номер в списке студенческой группы;
2. n – одно из чисел в пределах от 12 до 15 (12 для сдачи в конце семестра);
3. k – рассмотреть два случая: k=0, k=(номер студенческой группы);

Элементы $a_{i,j}$ при фиксированных i и j в обоих случаях одни и те же (матрицы отличаются только элементом $a_{1,1}$).

Программно реализовать (C или C++) вычисления для рассматриваемого примера. Для вычислений использовать тип float.

В выходных данных отчета должны быть представлены:

1. Преобразованная матрица A после первого шага алгоритма;
2. Вектор приближённого решения x^*;
3. Относительная погрешность вида $\frac{\|x-x^*\|_\infty}{\|x\|_\infty}$, где x – точное решение.

Входные данные:

$$A_{k=0} = \begin{bmatrix} 27 & -2 & 0 & -4 & 0 & 0 & -1 & -1 & -2 & 0 & -4 & -4 & -3 & -2 & -3 \\ -3 & 31 & -2 & -2 & -3 & -3 & 0 & -2 & -1 & -2 & -2 & -3 & -3 & -1 & -4 \\ -2 & -3 & 22 & -1 & 0 & -2 & -2 & 0 & -4 & 0 & -1 & -3 & -2 & -1 & -1 \\ 0 & -3 & -3 & 20 & -1 & -2 & 0 & -2 & -2 & -2 & 0 & -1 & -3 & 0 & -1 \\ -3 & -4 & -4 & -2 & 38 & -3 & -4 & -2 & 0 & -1 & -3 & -4 & -4 & 0 & -4 \\ -4 & -3 & -3 & -1 & -1 & 30 & -1 & 0 & 0 & -3 & -4 & -3 & -3 & -3 & -1 \\ 0 & 0 & -3 & -1 & -2 & -1 & 18 & -2 & 0 & -3 & -1 & -4 & 0 & -1 & 0 \\ -4 & -2 & -3 & -1 & -3 & -3 & -4 & 33 & -2 & -4 & 0 & -2 & -1 & -2 & -2 \\ -3 & -4 & -3 & -3 & -4 & -2 & -4 & 0 & 39 & -2 & -4 & -4 & -3 & 0 & -3 \\ -4 & -2 & -3 & -2 & -2 & -2 & -2 & -1 & -1 & 24 & 0 & 0 & -1 & -3 & -1 \\ -2 & -3 & -3 & -4 & -1 & -1 & -4 & -3 & -2 & -4 & 30 & 0 & 0 & -2 & -1 \\ -1 & -4 & -3 & -2 & -4 & -3 & -3 & 0 & -4 & -3 & -3 & 31 & 0 & -1 & 0 \\ -3 & 0 & -1 & 0 & -1 & -1 & -4 & -1 & -2 & -2 & -1 & -1 & 20 & -1 & -2 \\ -3 & -4 & -2 & -1 & 0 & 0 & -3 & -4 & -3 & -4 & 0 & 0 & -3 & 28 & -1 \\ 0 & -1 & 0 & 0 & 0 & -2 & -4 & -4 & -1 & -1 & -1 & -2 & 0 & -1 & 17 \end{bmatrix} \quad B_{k=0} = \begin{bmatrix} -226 \\ -204 \\ -110 \\ -72 \\ -106 \\ -61 \\ -22 \\ 29 \\ 63 \\ 92 \\ 133 \\ 174 \\ 162 \\ 197 \\ 113 \end{bmatrix}$$

$$A_{k=1} = \begin{bmatrix} 26.1 & -2 & 0 & -4 & 0 & 0 & -1 & -1 & -2 & 0 & -4 & -4 & -3 & -2 & -3 \\ -3 & 31 & -2 & -2 & -3 & -3 & 0 & -2 & -1 & -2 & -2 & -3 & -3 & -1 & -4 \\ -2 & -3 & 22 & -1 & 0 & -2 & -2 & 0 & -4 & 0 & -1 & -3 & -2 & -1 & -1 \\ 0 & -3 & -3 & 20 & -1 & -2 & 0 & -2 & -2 & -2 & 0 & -1 & -3 & 0 & -1 \\ -3 & -4 & -4 & -2 & 38 & -3 & -4 & -2 & 0 & -1 & -3 & -4 & -4 & 0 & -4 \\ -4 & -3 & -3 & -1 & -1 & 30 & -1 & 0 & 0 & -3 & -4 & -3 & -3 & -3 & -1 \\ 0 & 0 & -3 & -1 & -2 & -1 & 18 & -2 & 0 & -3 & -1 & -4 & 0 & -1 & 0 \\ -4 & -2 & -3 & -1 & -3 & -3 & -4 & 33 & -2 & -4 & 0 & -2 & -1 & -2 & -2 \\ -3 & -4 & -3 & -3 & -4 & -2 & -4 & 0 & 39 & -2 & -4 & -4 & -3 & 0 & -3 \\ -4 & -2 & -3 & -2 & -2 & -2 & -2 & -1 & -1 & 24 & 0 & 0 & -1 & -3 & -1 \\ -2 & -3 & -3 & -4 & -1 & -1 & -4 & -3 & -2 & -4 & 30 & 0 & 0 & -2 & -1 \\ -1 & -4 & -3 & -2 & -4 & -3 & -3 & 0 & -4 & -3 & -3 & 31 & 0 & -1 & 0 \\ -3 & 0 & -1 & 0 & -1 & -1 & -4 & -1 & -2 & -2 & -1 & -1 & 20 & -1 & -2 \\ -3 & -4 & -2 & -1 & 0 & 0 & -3 & -4 & -3 & -4 & 0 & 0 & -3 & 28 & -1 \\ 0 & -1 & 0 & 0 & 0 & -2 & -4 & -4 & -1 & -1 & -1 & -2 & 0 & -1 & 17 \end{bmatrix} \quad B_{k=1} = \begin{bmatrix} -230.5 \\ -204 \\ -110 \\ -72 \\ -106 \\ -61 \\ -22 \\ 29 \\ 63 \\ 92 \\ 133 \\ 174 \\ 162 \\ 197 \\ 113 \end{bmatrix}$$

Листинг программы:

Source.cpp

```
const int N = 15, M = 5;
int k = 1;

int main() {
	LES les(N, M, k); // LES - linear equations system; Initializing of the system of linear equations
	PrintToFile()(les); // print the system of linear equations
	les.firstStep(); // first step of calculating
	PrintToFile()(les);
	les.triangleForm(); // calculating a triangle form of LES
	PrintToFile()(les);
	les.findSolution(); // finding solution
	PrintToFile()(les);

	std::cout << std::fixed << les.relativeError(); // finding a relative error of a solution

	system("pause");

	return 0;
}
```

LES.cpp

```
LES::LES(int size, int offset, int option)
: coefs(size, size), constTerms(1, size), ApproximateSol(1, size), option_(option) {
	generateCoefs(option);
	generateCTerms(offset);

	state = "LinearEquationsSystem"; // state of LES
}

void LES::firstStep() {
	int k = 0;
	for (int i = k + 1; i < coefs.height_; ++i) {
		float l = coefs[i][k] / coefs[k][k];
		coefs[i][k] = 0;
		constTerms[i][0] -= l * constTerms[k][0];
		for (int j = k + 1; j < coefs.height_; ++j) {
			coefs[i][j] -= l * coefs[k][j];
		}
	}

	state = "FirstStep";
}

void LES::triangleForm() {
	if (state != "FirstStep") {
		firstStep();
	}
	for (int k = 1; k < coefs.height_ - 1; ++k) {
		for (int i = k + 1; i < coefs.height_; ++i) {
			float l = coefs[i][k] / coefs[k][k];
			coefs[i][k] = 0;
			constTerms[i][0] -= l * constTerms[k][0];
			for (int j = k + 1; j < coefs.height_; ++j) {
				coefs[i][j] -= l * coefs[k][j];
			}
		}
	}

	state = "TriangleForm";
```

```
}

void LES::findSolution() {
        for (int i = coefs.height_ - 1; i >= 0; --i) {
                ApproximateSol[i][0] = constTerms[i][0];
                for (int j = i + 1; j < coefs.height_; ++j) {
                        ApproximateSol[i][0] -= coefs[i][j] * ApproximateSol[j][0];
                }
                ApproximateSol[i][0] /= coefs[i][i];
        }

        state = "ApproximateSolution";
}

void LES::generateCoefs(int option) {
        for (int i = 0; i < coefs.height_; ++i) {
                int sum = 0;
                for (int j = 0; j < coefs.height_; ++j) {
                        coefs[i][j] = std::rand() % 5 - 4;
                        sum += coefs[i][j];
                }
                sum -= coefs[i][i];
                coefs[i][i] = -1 * sum;
        }

        (option == 0) ? coefs[0][0] += 1 : coefs[0][0] += 0.1; // depends on k
}

void LES::generateCTerms(int offset) { // setting coefficients for a given exact solution
                                       // X = (m, m + 1, ..., m + n - 1)
        for (int i = 0; i < constTerms.height_; ++i) {
                constTerms[i][0] = 0;
                for (int j = 0; j < constTerms.height_; ++j) {
                        constTerms[i][0] += (j + offset) * coefs[i][j];
                }
        }
}

float LES::relativeError() { // calculating a relative error of a solution
        int maxI = 0;
        for (int i = 1; i < coefs.height_; ++i) {
                if (abs(approximateSol[i][0] - (i + offset_)) > abs(approximateSol[maxI][0] - (maxI + offset_))) {
                        maxI = i;
                }
        }

        return abs(approximateSol[maxI][0] - (maxI + offset_)) / (coefs.height_ + offset_ - 1) * 100;
}
```

Выходные данные:
Преобразованная матрица A после первого шага алгоритма:

$$A_{k=0} = \begin{bmatrix}
27.00 & -2.00 & 0.00 & -4.00 & 0.00 & 0.00 & -1.00 & -1.00 & -2.00 & 0.00 & -4.00 & -4.00 & -3.00 & -2.00 & -3.00 \\
0.00 & 30.78 & -2.00 & -2.44 & -3.00 & -3.00 & -0.11 & -2.11 & -1.22 & -2.00 & -2.44 & -3.44 & -3.33 & -1.22 & -4.33 \\
0.00 & -3.15 & 22.00 & -1.30 & 0.00 & -2.00 & -2.07 & -0.07 & -4.15 & 0.00 & -1.30 & -3.30 & -2.22 & -1.15 & -1.22 \\
0.00 & -3.00 & -3.00 & 20.00 & -1.00 & -2.00 & 0.00 & -2.00 & -2.00 & -2.00 & 0.00 & -1.00 & -3.00 & 0.00 & -1.00 \\
0.00 & -4.22 & -4.00 & -2.44 & 38.00 & -3.00 & -4.11 & -2.11 & -0.22 & -1.00 & -3.44 & -4.44 & -4.33 & -0.22 & -4.33 \\
0.00 & -3.30 & -3.00 & -1.59 & -1.00 & 30.00 & -1.15 & -0.15 & -0.30 & -3.00 & -4.59 & -3.59 & -3.44 & -3.30 & -1.44 \\
0.00 & 0.00 & -3.00 & -1.00 & -2.00 & -1.00 & 18.00 & -2.00 & 0.00 & -3.00 & -1.00 & -4.00 & 0.00 & -1.00 & 0.00 \\
0.00 & -2.30 & -3.00 & -1.59 & -3.00 & -3.00 & -4.15 & 32.85 & -2.30 & -4.00 & -0.59 & -2.59 & -1.44 & -2.30 & -2.44 \\
0.00 & -4.22 & -3.00 & -3.44 & -4.00 & -2.00 & -4.11 & -0.11 & 38.78 & -2.00 & -4.44 & -4.44 & -3.33 & -0.22 & -3.33 \\
0.00 & -2.30 & -3.00 & -2.59 & -2.00 & -2.00 & -2.15 & -1.15 & -1.30 & 24.00 & -0.59 & -0.59 & -1.44 & -3.30 & -1.44 \\
0.00 & -3.15 & -3.00 & -4.30 & -1.00 & -1.00 & -4.07 & -3.07 & -2.15 & -4.00 & 29.70 & -0.30 & -0.22 & -2.15 & -1.22 \\
0.00 & -4.07 & -3.00 & -2.15 & -4.00 & -3.00 & -3.04 & -0.04 & -4.07 & -3.00 & -3.15 & 30.85 & -0.11 & -1.07 & -0.11 \\
0.00 & -0.22 & -1.00 & -0.44 & -1.00 & -1.00 & -4.11 & -1.11 & -2.22 & -2.00 & -1.44 & -1.44 & 19.67 & -1.22 & -2.33 \\
0.00 & -4.22 & -2.00 & -1.44 & 0.00 & 0.00 & -3.11 & -4.11 & -3.22 & -4.00 & -0.44 & -0.44 & -3.33 & 27.78 & -1.33 \\
0.00 & -1.00 & 0.00 & 0.00 & 0.00 & -2.00 & -4.00 & -4.00 & -1.00 & -1.00 & -1.00 & -2.00 & 0.00 & -1.00 & 17.00
\end{bmatrix}$$

$$A_{k=1} = \begin{bmatrix} 26.10 & -2.00 & 0.00 & -4.00 & 0.00 & 0.00 & -1.00 & -1.00 & -2.00 & 0.00 & -4.00 & -4.00 & -3.00 & -2.00 & -3.00 \\ 0.00 & 30.77 & -2.00 & -2.46 & -3.00 & -3.00 & -0.11 & -2.11 & -1.23 & -2.00 & -2.46 & -3.46 & -3.34 & -1.23 & -4.34 \\ 0.00 & -3.15 & 22.00 & -1.31 & 0.00 & -2.00 & -2.08 & -0.08 & -4.15 & 0.00 & -1.31 & -3.31 & -2.23 & -1.15 & -1.23 \\ 0.00 & -3.00 & -3.00 & 20.00 & -1.00 & -2.00 & 0.00 & -2.00 & -2.00 & -2.00 & 0.00 & -1.00 & -3.00 & 0.00 & -1.00 \\ 0.00 & -4.23 & -4.00 & -2.46 & 38.00 & -3.00 & -4.11 & -2.11 & -0.23 & -1.00 & -3.46 & -4.46 & -4.34 & -0.23 & -4.34 \\ 0.00 & -3.31 & -3.00 & -1.61 & -1.00 & 30.00 & -1.15 & -0.15 & -0.31 & -3.00 & -4.61 & -3.61 & -3.46 & -3.31 & -1.46 \\ 0.00 & 0.00 & -3.00 & -1.00 & -2.00 & -1.00 & 18.00 & -2.00 & 0.00 & -3.00 & -1.00 & -4.00 & 0.00 & -1.00 & 0.00 \\ 0.00 & -2.31 & -3.00 & -1.61 & -3.00 & -3.00 & -4.15 & 32.85 & -2.31 & -4.00 & -0.61 & -2.61 & -1.46 & -2.31 & -2.46 \\ 0.00 & -4.23 & -3.00 & -3.46 & -4.00 & -2.00 & -4.11 & -0.11 & 38.77 & -2.00 & -4.46 & -4.46 & -3.34 & -0.23 & -3.34 \\ 0.00 & -2.31 & -3.00 & -2.61 & -2.00 & -2.00 & -2.15 & -1.15 & -1.31 & 24.00 & -0.61 & -0.61 & -1.46 & -3.31 & -1.46 \\ 0.00 & -3.15 & -3.00 & -4.31 & -1.00 & -1.00 & -4.08 & -3.08 & -2.15 & -4.00 & 29.69 & -0.31 & -0.23 & -2.15 & -1.23 \\ 0.00 & -4.08 & -3.00 & -2.15 & -4.00 & -3.00 & -3.04 & -0.04 & -4.08 & -3.00 & -3.15 & 30.85 & -0.11 & -1.08 & -0.11 \\ 0.00 & -0.23 & -1.00 & -0.46 & -1.00 & -1.00 & -4.11 & -1.11 & -2.23 & -2.00 & -1.46 & -1.46 & 19.66 & -1.23 & -2.34 \\ 0.00 & -4.23 & -2.00 & -1.46 & 0.00 & 0.00 & -3.11 & -4.11 & -3.23 & -4.00 & -0.46 & -0.46 & -3.34 & 27.77 & -1.34 \\ 0.00 & -1.00 & 0.00 & 0.00 & 0.00 & -2.00 & -4.00 & -4.00 & -1.00 & -1.00 & -1.00 & -2.00 & 0.00 & -1.00 & 17.00 \end{bmatrix}$$

$$B_{k=0} = \begin{bmatrix} -226.00 \\ -229.11 \\ -126.74 \\ -72.00 \\ -131.11 \\ -94.48 \\ -22.00 \\ -4.48 \\ 37.89 \\ 58.52 \\ 116.26 \\ 165.63 \\ 76.89 \\ 171.89 \\ 113.00 \end{bmatrix} \quad B_{k=1} = \begin{bmatrix} -230.50 \\ -230.49 \\ -127.66 \\ -72.00 \\ -132.49 \\ -96.33 \\ -22.00 \\ -6.33 \\ 36.51 \\ 56.67 \\ 115.34 \\ 165.17 \\ 75.51 \\ 170.51 \\ 113.00 \end{bmatrix}$$

Решения СЛАУ:

$$X = \begin{bmatrix} 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \\ 16 \\ 17 \\ 18 \\ 19 \end{bmatrix} \quad X^*_{k=0} = \begin{bmatrix} 5.000002 \\ 6.000002 \\ 7.000003 \\ 8.000002 \\ 9.000004 \\ 10.000003 \\ 11.000002 \\ 12.000004 \\ 13.000003 \\ 14.000004 \\ 15.000006 \\ 16.000004 \\ 17.000000 \\ 18.000004 \\ 19.000002 \end{bmatrix} \quad X^*_{k=1} = \begin{bmatrix} 5.000257 \\ 6.000259 \\ 7.000259 \\ 8.000258 \\ 9.000258 \\ 10.000259 \\ 11.000260 \\ 12.000258 \\ 13.000259 \\ 14.000257 \\ 15.000257 \\ 16.000259 \\ 17.000257 \\ 18.000259 \\ 19.000257 \end{bmatrix}$$

Относительная погрешность:

$$\frac{\|x - x^*_{k=0}\|_\infty}{\|x\|_\infty} \approx 0,00003\%$$

$$\frac{\|x - x^*_{k=1}\|_\infty}{\|x\|_\infty} \approx 0,00137\%$$

Вывод:

Решение матрицы со строгим диагональным преобладанием имеет малую погрешность при решении методом Гаусса без выбора главного элемента. Чем больше преобладание диагональных элементов над остальными элементами таблицы, тем меньше погрешность.

Задание 2. Разработать программу численного решения СЛАУ методом Гаусса с выбором ведущего элемента по столбцу. Для выполнения прямого хода воспользоваться псевдокодом на странице 13.

Для заполнения матрицы A использовать случайные числа из диапазона -100 до 100. Правую часть b задать умножением матрицы A на вектор x=(m, m+1, ... , n+m−1): b=Ax.

Для вычислений выбрать параметры:

1. m – номер в списке студенческой группы;

2. n – одно из чисел в пределах от 15 до 20 (12 для сдачи в конце семестра).

Программно реализовать вычисления для рассматриваемого примера методом Гаусса с выбором ведущего элемента и методом Гаусса без выбора ведущего элемента (система уравнений в обоих случаях одна и та же). Для вычислений использовать тип float.

Для обоих случаев в выходных данных отчета должны быть представлены:

1. Преобразованная матрица A (и номер ведущего элемента в столбце в случае выбора ведущего элемента) после первого шага прямого хода метода Гаусса;

2. Вектор приближённого решения x^*;

3. Относительная погрешность вида $\frac{\|x-x^*\|_\infty}{\|x\|_\infty}$, где x – точное решение.

Входные данные:

$$A = \begin{bmatrix} -59 & 76 & 3 & 69 & -26 & -54 & -79 & -88 & -72 & 43 & -23 & -95 & 66 & 44 & 12 \\ -11 & 81 & -17 & -97 & -91 & -70 & 32 & -17 & 53 & -9 & 21 & 35 & -77 & -80 & 97 \\ -80 & -82 & -2 & -19 & -40 & -87 & 40 & 69 & 51 & -55 & 61 & -7 & 67 & 83 & 86 \\ -59 & -34 & -27 & 13 & -66 & -80 & 7 & 0 & 95 & 38 & 96 & -20 & -7 & 93 & 75 \\ -45 & -80 & -20 & -67 & -13 & 57 & 96 & -2 & 69 & 36 & 10 & -13 & -89 & -4 & -51 \\ -26 & 32 & 47 & -28 & 31 & 32 & 61 & 65 & -45 & -37 & 82 & 42 & 40 & -38 & 78 \\ -40 & -22 & 4 & -33 & 10 & -69 & -62 & -28 & -85 & -41 & -91 & 61 & -84 & 11 & 92 \\ -19 & 8 & -5 & 16 & -25 & 97 & -98 & 91 & 78 & -61 & -100 & -56 & -4 & -28 & -70 \\ -33 & -76 & 16 & 44 & -56 & -56 & 38 & -3 & -89 & 93 & -86 & 45 & -43 & -84 & -3 \\ -87 & 53 & -59 & 18 & -19 & 81 & -74 & -85 & 32 & -29 & -35 & -51 & 10 & 1 & -4 \\ 91 & -78 & 70 & 66 & 32 & -77 & -4 & -71 & -31 & -53 & 24 & 28 & -13 & -65 & -59 \\ -49 & -42 & -79 & 85 & -71 & -60 & -17 & 28 & 66 & 74 & 2 & -88 & -16 & 71 & 63 \\ -60 & 11 & -47 & 90 & -13 & 100 & -34 & 70 & -63 & -35 & 10 & -81 & 26 & 72 & 19 \\ -91 & -61 & 85 & 0 & -33 & -62 & 79 & -59 & 65 & -2 & -77 & -63 & 100 & -15 & 53 \\ 94 & 54 & -85 & -53 & 15 & 40 & 80 & 84 & -22 & -28 & 72 & 80 & 69 & -44 & -89 \end{bmatrix} \quad B = \begin{bmatrix} -2022 \\ -1303 \\ 3943 \\ 3966 \\ -1107 \\ 4494 \\ -3736 \\ -3333 \\ -3564 \\ -2800 \\ -3222 \\ 1259 \\ 1150 \\ 109 \\ 2570 \end{bmatrix}$$

Листинг программы:

Source.cpp

```
const int N = 15, M = 5;

int main() {
	LES lesPartialPivoting(N, M); // LES will be solved using
			//the Gaussian elimination with partial pivoting
	LES lesWithoutPivoting(lesPartialPivoting); // LES will be solved using
			//the Gaussian elimination without pivoting
	PrintToFile()(lesPartialPivoting);


	lesWithoutPivoting.firstStepWithoutPivoting();
	PrintToFile()(lesWithoutPivoting);
	lesWithoutPivoting.triangleFormWithoutPivoting();
	PrintToFile()(lesWithoutPivoting);
	lesWithoutPivoting.findSolution();
	PrintToFile()(lesWithoutPivoting);


	lesPartialPivoting.firstStepPartialPivoting();
	PrintToFile()(lesPartialPivoting);
	lesPartialPivoting.triangleFormPartialPivoting();
	PrintToFile()(lesPartialPivoting);
	lesPartialPivoting.findSolution();
	PrintToFile()(lesPartialPivoting);

	std::cout << std::fixed << lesWithoutPivoting.relativeError() << std::endl;
	std::cout << std::fixed << lesPartialPivoting.relativeError() << std::endl;

	system("pause");

	return 0;
}
```

LES.cpp

```
void LES::firstStepPartialPivoting() {
        int k = 0;
        int colNum = k;

        for (int i = k + 1; i < coefs.height_; ++i) {
                if (abs(coefs[i][k]) > abs(coefs[colNum][k])) {
                        colNum = i;             // choosing index of max absolute element in a column
                }
        }

        for (int j = 0; j < coefs.height_; ++j) {
                std::swap(coefs[colNum][j], coefs[k][j]); // swap rows a[i]
        }
        std::swap(constTerms[colNum][0], constTerms[k][0]); // swap constant terms b[i]

        for (int i = k + 1; i < coefs.height_; ++i) {

                float l = coefs[i][k] / coefs[k][k];
                coefs[i][k] = 0;
                constTerms[i][0] -= l * constTerms[k][0];
                for (int j = k + 1; j < coefs.height_; ++j) {
                        coefs[i][j] -= l * coefs[k][j];
                }
        }

        state = "firstStepPartialPivoting";
}

void LES::triangleFormPartialPivoting() {
        if (state != "firstStepPartialPivoting") {
                firstStepPartialPivoting();
        }
        for (int k = 1; k < coefs.height_ - 1; ++k) {
                int colNum = k;

                for (int i = k + 1; i < coefs.height_; ++i) {
                        if (abs(coefs[i][k]) > abs(coefs[colNum][k])) {
                                colNum = i;
                        }
                }

                for (int j = 0; j < coefs.height_; ++j) {
                        std::swap(coefs[colNum][j], coefs[k][j]);
                }
                std::swap(constTerms[colNum][0], constTerms[k][0]);

                for (int i = k + 1; i < coefs.height_; ++i) {

                        float l = coefs[i][k] / coefs[k][k];
                        coefs[i][k] = 0;
                        constTerms[i][0] -= l * constTerms[k][0];
                        for (int j = k + 1; j < coefs.height_; ++j) {
                                coefs[i][j] -= l * coefs[k][j];
                        }
                }
        }

        state = "triangleFormPartialPivoting";
}
```

Выходные данные:

Преобразованная матрица A после первого шага алгоритма:
Без выбора главного элемента:

$$A = \begin{bmatrix}
-59.00 & 76.00 & 3.00 & 69.00 & -26.00 & -54.00 & -79.00 & -88.00 & -72.00 & 43.00 & -23.00 & -95.00 & 66.00 & 44.00 & 12.00 \\
0.00 & 66.83 & -17.56 & -109.86 & -86.15 & -59.93 & 46.73 & -0.59 & 66.42 & -17.02 & 25.29 & 52.71 & -89.31 & -88.20 & 94.76 \\
0.00 & -185.05 & -6.07 & -112.56 & -4.75 & -13.78 & 147.12 & 188.32 & 148.63 & -113.31 & 92.19 & 121.81 & -22.49 & 23.34 & 69.73 \\
0.00 & -110.00 & -30.00 & -56.00 & -40.00 & -26.00 & 86.00 & 88.00 & 167.00 & -5.00 & 119.00 & 75.00 & -73.00 & 49.00 & 63.00 \\
0.00 & -137.97 & -22.29 & -119.63 & 6.83 & 98.19 & 156.25 & 65.12 & 123.92 & 3.20 & 27.54 & 59.46 & -139.34 & -37.56 & -60.15 \\
0.00 & -1.49 & 45.68 & -58.41 & 42.46 & 55.80 & 95.81 & 103.78 & -13.27 & -55.95 & 92.14 & 83.86 & 10.92 & -57.39 & 72.71 \\
0.00 & -73.53 & 1.97 & -79.78 & 27.63 & -32.39 & -8.44 & 31.66 & -36.19 & -70.15 & -75.41 & 125.41 & -128.75 & -18.83 & 83.86 \\
0.00 & -16.47 & -5.97 & -6.22 & -16.63 & 114.39 & -72.56 & 119.34 & 101.19 & -74.85 & -92.59 & -25.41 & -25.25 & -42.17 & -73.86 \\
0.00 & -118.51 & 14.32 & 5.41 & -41.46 & -25.80 & 82.19 & 46.22 & -48.73 & 68.95 & -73.14 & 98.14 & -79.92 & -108.61 & -9.71 \\
0.00 & -59.07 & -63.42 & -83.75 & 19.34 & 160.63 & 42.49 & 44.76 & 138.17 & -92.41 & -1.08 & 89.08 & -87.32 & -63.88 & -21.69 \\
0.00 & 39.22 & 74.63 & 172.42 & -8.10 & -160.29 & -125.85 & -206.73 & -142.05 & 13.32 & -11.47 & -118.53 & 88.80 & 2.86 & -40.49 \\
0.00 & -105.12 & -81.49 & 27.69 & -49.41 & -15.15 & 48.61 & 101.08 & 125.80 & 38.29 & 21.10 & -9.10 & -70.81 & 34.46 & 53.03 \\
0.00 & -66.29 & -50.05 & 19.83 & 13.44 & 154.92 & 46.34 & 159.49 & 10.22 & -78.73 & 33.39 & 15.61 & -41.12 & 27.25 & 6.80 \\
0.00 & -178.22 & 80.37 & -106.42 & 7.10 & 21.29 & 200.85 & 76.73 & 176.05 & -68.32 & -41.53 & 83.53 & -1.80 & -82.86 & 34.49 \\
0.00 & 175.08 & -80.22 & 56.93 & -26.42 & -46.03 & -45.86 & -56.20 & -136.71 & 40.51 & 35.36 & -71.36 & 174.15 & 26.10 & -69.88
\end{bmatrix}$$

С выбором главного элемента:

$$A = \begin{bmatrix} 94.00 & 54.00 & -85.00 & -53.00 & 15.00 & 40.00 & 80.00 & 84.00 & -22.00 & -28.00 & 72.00 & 80.00 & 69.00 & -44.00 & -89.00 \\ 0.00 & 87.32 & -26.95 & -103.20 & -89.24 & -65.32 & 41.36 & -7.17 & 50.43 & -12.28 & 29.43 & 44.36 & -68.93 & -85.15 & 86.59 \\ 0.00 & -36.04 & -74.34 & -64.11 & -27.23 & -52.96 & 108.09 & 140.49 & 32.28 & -78.83 & 122.28 & 61.09 & 125.72 & 45.55 & 10.26 \\ 0.00 & -0.11 & -80.35 & -20.27 & -56.59 & -54.89 & 57.21 & 52.72 & 81.19 & 20.43 & 141.19 & 30.21 & 36.31 & 65.38 & 19.14 \\ 0.00 & -54.15 & -60.69 & -92.37 & -5.82 & 76.15 & 134.30 & 38.21 & 58.47 & 22.60 & 44.47 & 25.30 & -55.97 & -25.06 & -93.61 \\ 0.00 & 46.94 & 23.49 & -42.66 & 35.15 & 43.06 & 83.13 & 88.23 & -51.09 & -44.74 & 101.91 & 64.13 & 59.09 & -50.17 & 53.38 \\ 0.00 & 0.98 & -32.17 & -55.55 & 16.38 & -51.98 & -27.96 & 7.74 & -94.36 & -52.91 & -60.36 & 95.04 & -54.64 & -7.72 & 54.13 \\ 0.00 & 18.91 & -22.18 & 5.29 & -21.97 & 105.09 & -81.83 & 107.98 & 73.55 & -66.66 & -85.45 & -39.83 & 9.95 & -36.89 & -87.99 \\ 0.00 & -57.04 & -13.84 & 25.39 & -50.73 & -41.96 & 66.09 & 26.49 & -96.72 & 83.17 & -60.72 & 73.09 & -18.78 & -99.45 & -34.24 \\ 0.00 & 102.98 & -137.67 & -31.05 & -5.12 & 118.02 & 0.04 & -7.26 & 11.64 & -54.91 & 31.64 & 23.04 & 73.86 & -39.72 & -86.37 \\ 0.00 & -130.28 & 152.29 & 117.31 & 17.48 & -115.72 & -81.45 & -152.32 & -9.70 & -25.89 & -45.70 & -49.45 & -79.80 & -22.40 & 27.16 \\ 0.00 & -13.85 & -123.31 & 57.37 & -63.18 & -39.15 & 24.70 & 71.79 & 54.53 & 59.40 & 39.53 & -46.30 & 19.97 & 48.06 & 16.61 \\ 0.00 & 45.47 & -101.26 & 56.17 & -3.43 & 125.53 & 17.06 & 123.62 & -77.04 & -52.87 & 55.96 & -29.94 & 70.04 & 43.91 & -37.81 \\ 0.00 & -8.72 & 2.71 & -51.31 & -18.48 & -23.28 & 156.45 & 22.32 & 43.70 & -29.11 & -7.30 & 14.45 & 166.80 & -57.60 & -33.16 \\ 0.00 & 109.89 & -50.35 & 35.73 & -16.59 & -28.89 & -28.79 & -35.28 & -85.81 & 25.43 & 22.19 & -44.79 & 109.31 & 16.38 & -43.86 \end{bmatrix}$$

$$B_{without\ pivoting} = \begin{bmatrix} -2022.00 \\ -926.02 \\ 6684.70 \\ 5988.00 \\ 435.20 \\ 5385.05 \\ -2365.15 \\ -2681.85 \\ -2433.05 \\ 181.59 \\ -6340.68 \\ 2938.29 \\ 3206.27 \\ 3227.68 \\ -651.49 \end{bmatrix} \quad B_{with\ pivoting} = \begin{bmatrix} 2570.00 \\ -1002.26 \\ 6130.23 \\ 5579.08 \\ 123.32 \\ 5204.85 \\ -2642.38 \\ -2813.53 \\ -2661.77 \\ -421.38 \\ -5709.98 \\ 2598.68 \\ 2790.43 \\ 2596.98 \\ -408.91 \end{bmatrix}$$

Решения СЛАУ:

$$X = \begin{bmatrix} 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \\ 16 \\ 17 \\ 18 \\ 19 \end{bmatrix} \quad X^*_{without\ pivoting} = \begin{bmatrix} 5.000788 \\ 5.998564 \\ 6.999993 \\ 7.998513 \\ 8.999789 \\ 10.000587 \\ 10.998563 \\ 11.999774 \\ 12.999682 \\ 14.000873 \\ 15.000555 \\ 15.999166 \\ 17.000727 \\ 17.999231 \\ 19.000841 \end{bmatrix} \quad X^*_{with\ pivoting} = \begin{bmatrix} 5.000005 \\ 5.999987 \\ 6.999998 \\ 8.000000 \\ 8.999991 \\ 10.000018 \\ 10.999995 \\ 11.999993 \\ 12.999993 \\ 14.000005 \\ 15.000006 \\ 15.999997 \\ 17.000008 \\ 17.999990 \\ 19.000011 \end{bmatrix}$$

Относительная погрешность:

$$\frac{\|x - x^*_{without\ pivoting}\|_\infty}{\|x\|_\infty} \approx 0.007825\%$$

$$\frac{\|x - x^*_{with\ pivoting}\|_\infty}{\|x\|_\infty} \approx 0.000095\%$$

Вывод:

В общем случае решение методом Гаусса с выбором главного элемента позволяет получить меньшую погрешность.