Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное

образовательное учреждение высшего образования

Санкт-Петербургский университет аэрокосмического приборостроения

ГУАП

КАФЕДРА № 2

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кандидат тех. Наук, доцент |  |  |  | С.Л Козенко |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №2  **“Решение СЛАУ”** |
| по дисциплине: Вычислительная математика |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 4134к |  |  |  | Костяков НА |
|  |  |  | подпись, дата |  | инициалы, фамилия |

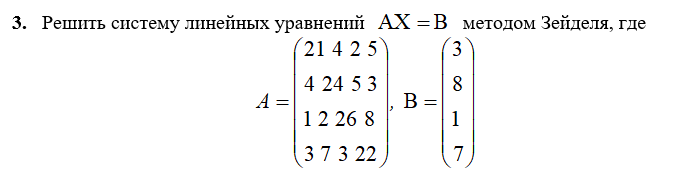
Санкт-Петербург

2023

# Цели работы

а) освоение основных методов решения систем линейных алгебраических уравнений (СЛАУ); б) совершенствование навыков по алгоритмизации и программированию вычислительных задач.

# Постановка задачи



# Математическая часть

**Описание метода решения**

Метод Зейделя относится к итерационным методам решения систем линейных уравнений, обеспечивающим хорошую сходимость итерационного процесса поиска корней системы.

Прежде всего необходимо задать значения начальных (нулевых) приближений корней С1 (0), С2 (0),..., Сm(0). Если отсутствует какая–нибудь информация об этих значениях, то их можно принять равными свободным членам системы уравнений (3.2) или даже принять равными нулю. Выбранные таким образом нулевые приближения корней подставим в первое уравнение системы (3.2) и получим первое приближение корня

С1 С1 (1) = β1 + α12 С2 (0) + α13 С3 (0) + … + α1 m Сm(0).

Используя во втором уравнении системы (3.2) найденное первое приближение корня С1 и нулевые приближения остальных корней, получим первое приближение корня

С2 С2 (1) = β2 + α21 С1 (1) + α23 С2 (0) + … + α2 m Сm(0).

Повторяя эту процедуру последовательно для всех уравнений системы (3.2), получим в итоге первое приближение корня

Сm(1) = βm + αm 1 С1 (1) + αm2 С2 (1) + … + αm m–1 Сm–1(1).

Используя первые приближение корня системы, можно аналогичным образом найти вторые приближения

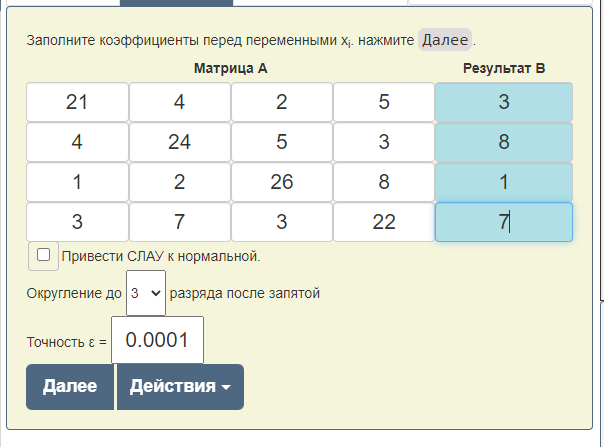
С1 (2) = β1 + α12 С2 (1) + α13 С3 (1) + … + α1 mСm(1)

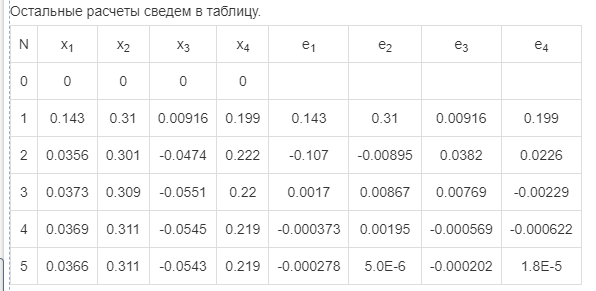
С2 (2) = β1 + α21 С1 (2) + α23 С3 (1) + … + α2 mСm(1) ..............................................................................................

Сm(2) = βm + αm 1 С1 (2) + αm 2 С2 (2) + … + αm m–1 Сm–1(2).

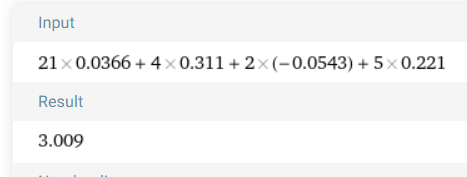
Затем, используя вторые приближения, можно вычислить третьи и т.д. Итерационный процесс решения системы линейных уравнений методом Зейделя сходится к единственному решению при любом выборе начальных приближений искомых корней, если выполняется одно из условий (3.5). Погрешность решения системы уравнений методом Зейделя принято оценивать по формулам (3.7) — (3.9).

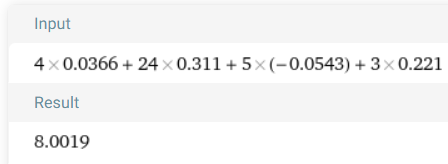
# Аналитические расчеты

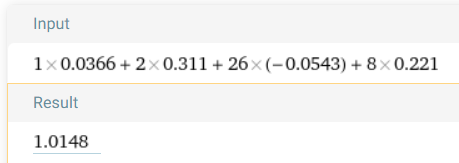


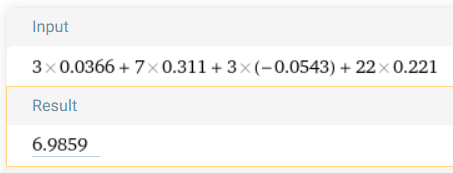


**Подставим значения**

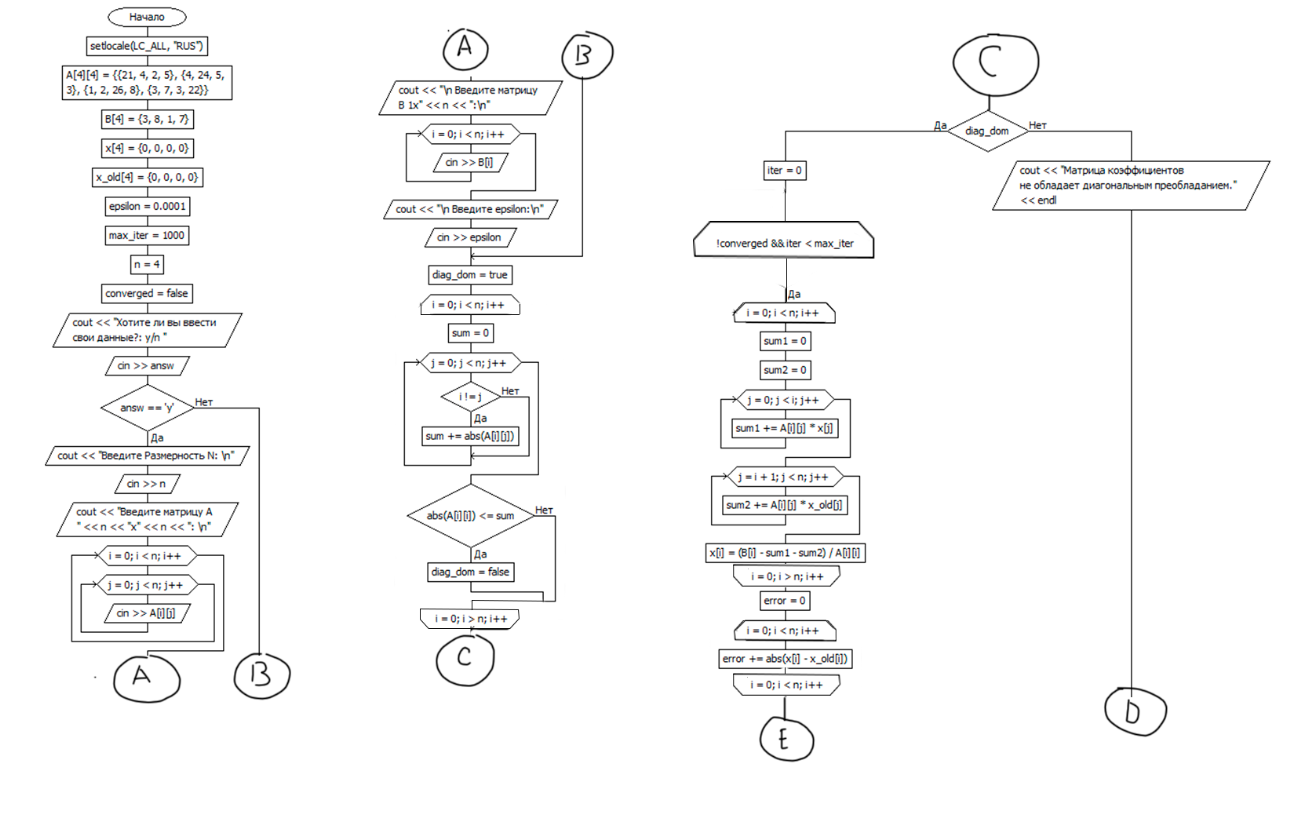
****

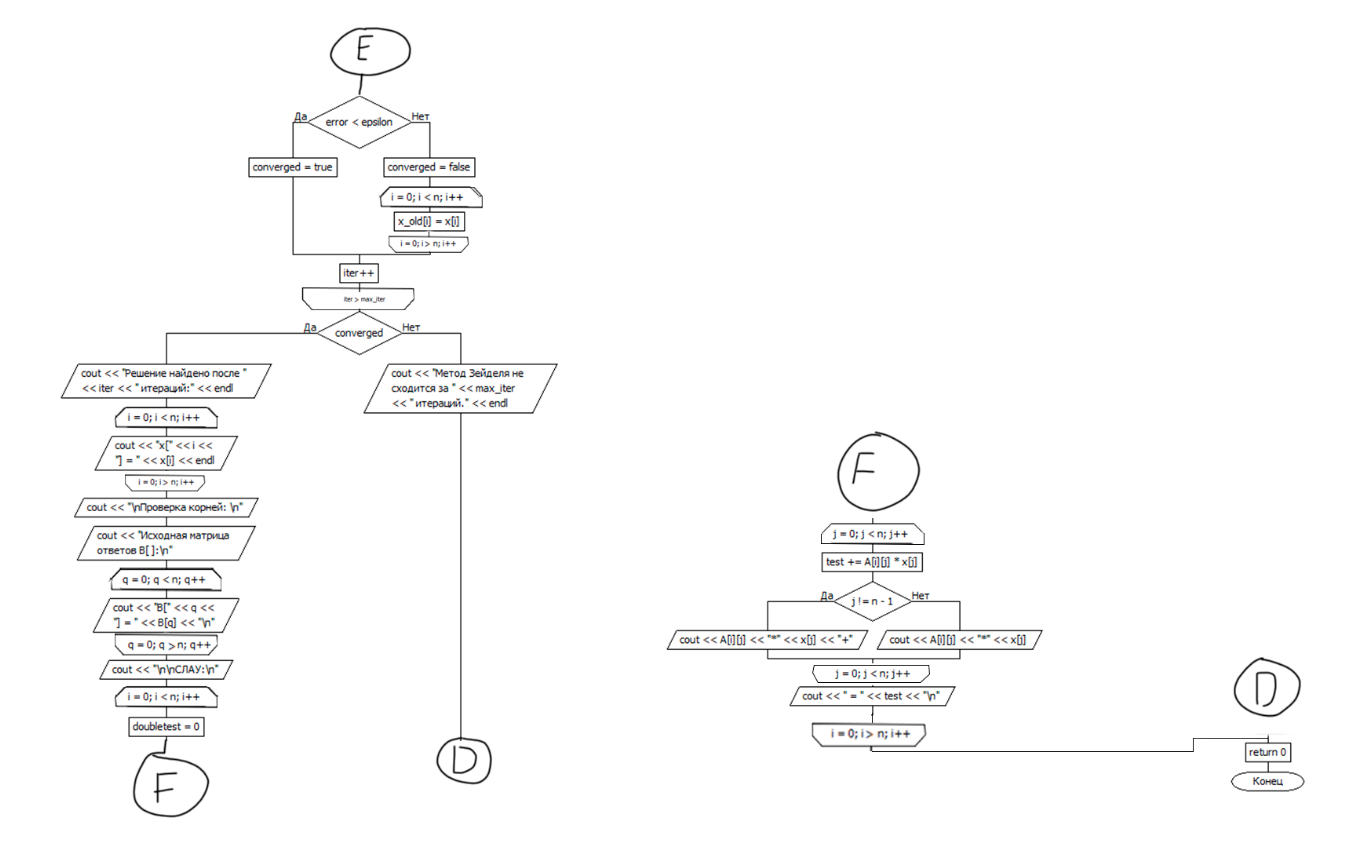
****

****

****

# Схема программы





# Листинг программы

#include <iostream>

#include <cmath>

using namespace std;

int main() {

setlocale(LC\_ALL, "RUS");

int A[4][4] = { {21, 4, 2, 5}, {4, 24, 5, 3},{1, 2, 26, 8}, {3,7 ,3, 22} };

int B[4] = { 3, 8, 1, 7 };

double x[4] = { 0, 0, 0, 0 };

double x\_old[4] = { 0, 0, 0, 0 };

double epsilon = 0.0001;

int max\_iter = 1000;

int n = 4;

bool converged = false;

cout << "Хотите ли вы ввести свои данные?: y/n ";

char answ;

cin >> answ;

if (answ == 'y')

{

cout << "Введите Размерность N: \n";

cin >> n;

cout << "Введите матрицу A "<<n<<"x"<<n<<": \n";

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

cin >> A[i][j];

}

}

cout << "\n Введите матрицу B 1x"<<n<<":\n";

for (int i = 0; i < n; i++) {

cin >> B[i];

}

cout << "\n Введите epsilon:\n";

cin >> epsilon;

}

// Проверка диагонального преобладания

bool diag\_dom = true;

for (int i = 0; i < n; i++) {

int sum = 0;

for (int j = 0; j < n; j++) {

if (i != j) {

sum += abs(A[i][j]);

}

}

if (abs(A[i][i]) <= sum) {

diag\_dom = false;

break;

}

}

// Итерационный процесс метода Зейделя

if (diag\_dom) {

int iter = 0;

while (!converged && iter < max\_iter) {

for (int i = 0; i < n; i++) {

double sum1 = 0;

double sum2 = 0;

for (int j = 0; j < i; j++) {

sum1 += A[i][j] \* x[j];

}

for (int j = i + 1; j < n; j++) {

sum2 += A[i][j] \* x\_old[j];

}

x[i] = (B[i] - sum1 - sum2) / A[i][i];

}

double error = 0;

for (int i = 0; i < n; i++) {

error += abs(x[i] - x\_old[i]);

}

if (error < epsilon) {

converged = true;

}

else {

converged = false;

for (int i = 0; i < n; i++) {

x\_old[i] = x[i];

}

}

iter++;

}

if (converged) {

cout << "Решение найдено после " << iter << " итераций:" << endl;

for (int i = 0; i < n; i++) {

cout << "x[" << i << "] = " << x[i] << endl;

}

cout << "\nПроверка корней: \n";

cout << "Исходная матрица ответов B[]:\n";

for (int q = 0; q < n; q++) {

cout << "B["<<q<<"] = " << B[q] << "\n";

}

cout << "\n\nСЛАУ:\n";

for (int i = 0; i < n; i++) {

double test=0;

for (int j = 0; j < n; j++) {

test += A[i][j] \* x[j];

if (j != n - 1) {

cout << A[i][j] << "\*" << x[j] << "+";

}

else

{

cout << A[i][j] << "\*" << x[j];

}

}

cout << " = " << test <<"\n";

}

}

else {

cout << "Метод Зейделя не сходится за " << max\_iter << " итераций." << endl;

}

}

else {

cout << "Матрица коэффициентов не обладает диагональным преобладанием." << endl;

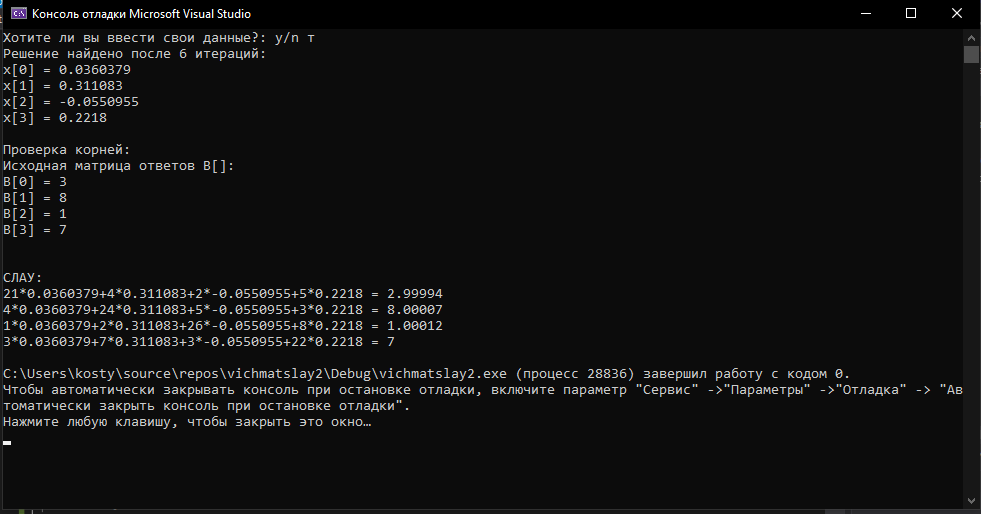
}

return 0;

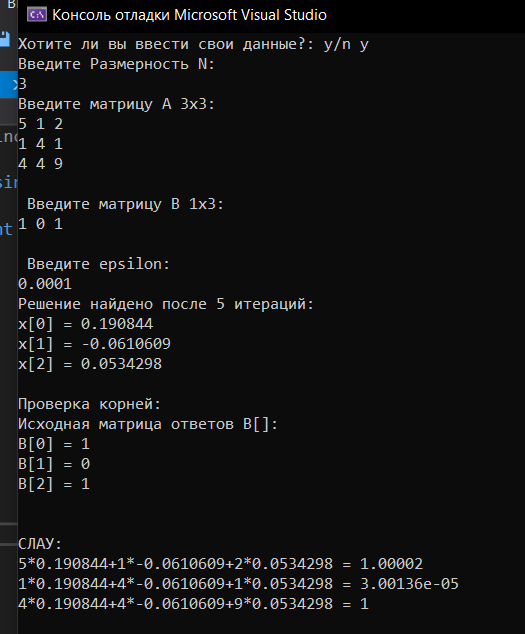
}

# Результат работы

**Для параметров по варианту**

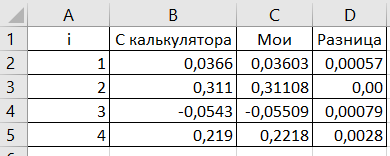


**Для параметров с клавиатуры**



# Сравнение результатов

Результаты вычислений программы и моего кода c учетом погрешности совпадают. Можно сделать вывод, что результаты достоверны.



# Вывод

Я освоил один из основных методов решения СЛАУ и совершенствовал навыки проектировки программ на языке C++