УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе №2.5

по предмету «Основы алгоритмизации и программирования»

Вариант 20

Выполнил:

Машевский Д.В

Гр. 351003

Проверил:

Данилова Г. В.

Минск 2023

**Задание:**

Выполнить «прямой ход» в решении системы линейных алгебраических уравнений методом Гаусса. Вводить элементы матрицы системы и вектор свободных членов.

**Код Delphi:**

Program LABA25;

Uses

SysUtils;

Type

TSet = Set Of Char;

TMatrix = Array Of Array Of Double;

TArr = Array Of Double;

Procedure DelProg();

Begin

Writeln('Данная программа выполняет "Прямой ход" в системе лин. уравн., используя

метод Гаусса');

End;

Function PTF(): String;

Var

Path: String;

IsCorrect: Boolean;

Begin

Repeat

IsCorrect := True;

Write('Укажите путь к файлу: ');

Readln(Path);

If Not FileExists(Path) Then

Begin

Write('Такой файл не найден. ');

IsCorrect := False;

End;

If IsCorrect And (ExtractFileExt(Path) <> '.txt') Then

Begin

IsCorrect := False;

Write('У файла должно быть расширение txt. ');

End;

Until IsCorrect;

PTF := Path;

End;

Function ViborRaboti(): Integer;

Var

Vib: Integer;

IsCorrect: Boolean;

Begin

Writeln('0 - Работа с консолью, 1 - работа с файлом');

Repeat

IsCorrect := True;

Write('Выбор: ');

Try

Readln(Vib);

Except

IsCorrect := False;

Write('Ошибка ввода. ');

End;

If IsCorrect And (Vib <> 0) And (Vib <> 1) Then

Begin

IsCorrect := False;

Write('Число должно быть либо 0, либо 1. ');

End;

Until IsCorrect;

ViborRaboti := Vib;

End;

Function EnterN(): Integer;

Var

N: Integer;

IsCorrect: Boolean;

Begin

N:=0;

Write('Введите кол-во уравнений в системе : ');

Repeat

IsCorrect := True;

Try

Readln(N);

Except

Writeln('Ошибка. Проверьте данные на правильность ввода.');

IsCorrect := False;

End;

If (IsCorrect And ((N < 2) Or (N > 20))) Then

Begin

Writeln('Пожалуйста, введите кол-во уравн от 2 до 20');

IsCorrect := False;

End;

Until IsCorrect;

EnterN := N;

End;

Function ReadSizeFile(Path: String): Integer;

Var

N: Integer;

IsCorrect: Boolean;

InputFile: TextFile;

Begin

N:= 0;

AssignFile(InputFile, Path);

Reset(InputFile);

IsCorrect := True;

Try

Readln(InputFile, N);

Except

IsCorrect := False;

Writeln('Ошибка при считывании порядка матрицы. ');

End;

If IsCorrect And (N < 2) Then

Begin

IsCorrect := False;

Writeln('Значение порядка матрицы в файле должно быть больше 1. ');

End;

Close(InputFile);

If IsCorrect Then

ReadSizeFile := N

Else

Write('Введите порядок матрицы с консоли. ');

ReadSizeFile := EnterN;

End;

Procedure SeeColUravnSistem(Vib, N: Integer; Path: String);

Var

OutputFile: TextFile;

IsCorrect: Boolean;

Begin

If Vib = 0 Then

Writeln('Количество уравнений в системе: ', N)

Else

Begin

Repeat

IsCorrect := True;

AssignFile(OutputFile, Path);

Try

Rewrite(OutputFile);

Except

Writeln('Ошибка.');

IsCorrect := False;

Path := PTF();

End;

Until IsCorrect;

Write(OutputFile, N);

Write(OutputFile, #13);

Close(OutputFile);

End;

End;

Function MatrixConsole(N: Integer): TMatrix;

Var

A: TMatrix;

IsCorrect: Boolean;

I, J: Integer;

Begin

SetLength(A, N, N + 1);

For I := Low(A) To High(A) Do

For J := Low(A[0]) To High(A[0]) Do

Begin

Write('A[ ', I + 1,', ', J + 1, ']: ');

Repeat

IsCorrect := True;

Try

Read(A[I, J]);

Except

Writeln('Ошибка ввода. Введите корректное значение:');

IsCorrect := False;

End;

If IsCorrect And ((A[I, J] < -2000) Or

(A[I, J] > 2000)) Then

Begin

Writeln('Ошибка. Введите значения от -2000 до 2000');

IsCorrect := False;

End;

Until IsCorrect;

End;

MatrixConsole := A;

End;

Function MatrixFile(N: Integer; Path: String): TMatrix;

Var

A: TMatrix;

IsCorrect: Boolean;

InputFile: TextFile;

I, J, E: Integer;

Symbol: AnsiChar;

Begin

AssignFile(InputFile, Path);

Reset(InputFile);

Readln(InputFile);

SetLength(A, N, N + 1);

E := 0;

While Not Eof(InputFile) Do

Begin

Read(InputFile, Symbol);

If Symbol = ' ' Then

Begin

Inc(E);

End;

End;

Close(InputFile);

If E > N \* (N + 1) Then

Begin

Writeln('Ошибка. Поработайте с консолью.');

A := MatrixConsole(N);

End

Else

Begin

AssignFile(InputFile, Path);

Reset(InputFile);

Readln(InputFile);

For I := Low(A) To High(A) Do

Begin

For J := Low(A[0]) To High(A[0]) Do

Begin

Repeat

IsCorrect := True;

Try

Read(InputFile, A[I, J]);

Except

Writeln('Ошибка. Поработайте с консолью.');

IsCorrect := False;

A := MatrixConsole(N);

End;

If (IsCorrect And ((A[I, J] < -2000) Or

(A[I, J] > 2000))) Then

Begin

Writeln('Ошибка. Поработайте с консолью.');

IsCorrect := False;

A := MatrixConsole(N);

End;

Until IsCorrect;

End;

End;

Close(InputFile);

End;

MatrixFile := A;

End;

Procedure WriteMatrix(Vib: Integer; A: TMatrix; Path: String);

Var

OutputFile: TextFile;

IsCorrect: Boolean;

I, J: Integer;

Begin

If Vib = 0 Then

Begin

For I := Low(A) To High(A) Do

Begin

Write('');

For J := Low(A) To High(A) Do

Begin

Write('', A[I, J]:5:0, '');

End;

Write('', A[I, High(A) + 1]:5:0, '');

Writeln('');

End;

End;

If Vib = 1 Then

Begin

Repeat

IsCorrect := True;

AssignFile(OutputFile, Path);

Try

Append(OutputFile);

Except

Writeln('Ошибка. Проверьте данные на правильность ввода.');

IsCorrect := False;

Path := PTF();

End;

Until IsCorrect;

For I := Low(A) To High(A) Do

Begin

Write(OutputFile, '|');

For J := Low(A) To High(A) Do

Begin

Write(OutputFile, ' ', A[I, J]:6:0, ' ');

End;

Write(OutputFile, '', A[I, High(A) + 1]:6:0, ';');

Writeln(OutputFile, '');

End;

Write(OutputFile, #13);

Close(OutputFile);

End;

End;

Function DelTreugMatr(A: TMatrix): TMatrix;

Var

I, J, K, Na, FinalPosJ: Integer;

Temp: Real;

TrMatr: TMatrix;

Begin

SetLength(TrMatr, Length(A), Length(A[0]));

For I := Low(A) To High(A) Do

Begin

For J := Low(A[0]) To High(A[0]) Do

Begin

TrMatr[I, J] := A[I, J];

End;

End;

For I := 0 To High(TrMatr) Do

Begin

Na := I + 1;

For J := Na To High(TrMatr) Do

Begin

Temp := TrMatr[J, I] / TrMatr[I, I];

For K := I To High(TrMatr) Do

Begin

TrMatr[J, K] := TrMatr[J, K] - Temp \* TrMatr[I, K];

TrMatr[J, High(TrMatr[0])] := TrMatr[J, High(TrMatr[0])] - Temp \* TrMatr[I, High(TrMatr[0])];

End;

End;

End;

DelTreugMatr := TrMatr;

End;

Procedure WriteTreugMatr(Vib: Integer; A: TMatrix; Path: String);

Var

OutputFile: TextFile;

IsCorrect: Boolean;

I, J: Integer;

Begin

If Vib = 0 Then

Begin

For I := Low(A) To High(A) Do

Begin

Write('');

For J := Low(A) To High(A) Do

Begin

Write(' ', A[I, J]:5:2, ' ');

End;

Write('', A[I, High(A) + 1]:5:2, '');

Writeln('');

End;

End

Else

Begin

Repeat

IsCorrect := True;

AssignFile(OutputFile, Path);

Try

Append(OutputFile);

Except

Writeln('Ошибка. Поработайте с консолью.');

IsCorrect := False;

Path := Ptf

End;

Until IsCorrect;

For I := Low(A)To High(A)Do

Begin

Write(OutputFile, '');

For J := Low(A)To High(A) Do

Begin

Write(OutputFile, ' ', A[I, J]:5:2, ' ');

End;

Writeln(OutputFile, '', A[I, High(A) + 1] :5:2, '');

Writeln(OutputFile, '');

End;

Write(OutputFile, #13);

Close(OutputFile);

End;

End;

Procedure Arrray(Var Arr: TArr);

Var

I: Integer;

Begin

For I := 0 To High(Arr) Do

Begin

Arr[I] := 0;

End;

End;

Function SchitCorn(Const TrMatr: TMatrix): TArr;

Var

I, J, K, Pstn, FPstn: Integer;

Rez: TArr;

Temp, Dividend: Real;

Begin

SetLength(Rez, Length(TrMatr));

Arrray(Rez);

Pstn := High(TrMatr);

K := 1;

Dividend := 0;

For I := 0 To High(Rez) Do

Begin

If (TrMatr[Pstn, Pstn] <> 0) Then

Begin

Dividend := TrMatr[Pstn, Pstn + K];

FPstn := I - 1;

For J := 0 To FPstn Do

Begin

Dividend := Dividend - TrMatr[Pstn, Pstn + I - J] \* Rez[J];

End;

Temp := Dividend / TrMatr[Pstn, Pstn];

End

Else

Begin

Temp := 0;

End;

Rez[I] := Temp;

Dec(Pstn);

Inc(K);

End;

SchitCorn := Rez;

End;

Procedure WriteArray(Vib: Integer; Resh: TArr; Path: String);

Var

OutputFile: TextFile;

IsCorrect: Boolean;

I: Integer;

Begin

If Vib = 0 Then

Begin

Writeln('Корни');

For I := Low(Resh) To High(Resh) Do

Begin

Writeln('Значение ', I + 1, ' переменной ',Resh[I]:6:2, '.');

End;

End;

If Vib = 1 Then

Begin

Writeln('Запись найденных корней в файл.');

Repeat

IsCorrect := True;

AssignFile(OutputFile, Path);

Try

Append(OutputFile);

Except

Writeln('Ошибка. Проверьте вводимые значения.');

IsCorrect := False;

Path := PTF();

End;

Until IsCorrect;

For I := Low(Resh) To High(Resh) Do

Begin

Writeln(OutputFile, '', I + 1, '', Resh[I]:6:2, '');

End;

Write(OutputFile, #13);

Close(OutputFile);

End;

End;

Function VvodMatr(Var CharSet: TSet): TMatrix;

Var

InputFilePath: String;

InputChoice, N: Integer;

Mtr: TMatrix;

Begin

InputChoice := ViborRaboti();

If InputChoice = 0 Then

Begin

N := EnterN();

Mtr := MatrixConsole(N);

End;

If InputChoice = 1 Then

Begin

InputFilePath := PTF();

N := ReadSizeFile(InputFilePath);

Mtr := MatrixFile(N, InputFilePath);

End;

VvodMatr := Mtr;

End;

Procedure RabPols(Const N: Integer;Const Mtr, TrMatr: TMatrix; Const Resh: TArr);

Var

OutputFilePath: String;

OutputChoice: Integer;

Begin

OutputChoice := ViborRaboti();

If OutputChoice = 1 Then

Begin

OutputFilePath := PTF();

End;

SeeColUravnSistem(OutputChoice, N, OutputFilePath);

WriteMatrix(OutputChoice, Mtr, OutputFilePath);

WriteTreugMatr(OutputChoice, TrMatr, OutputFilePath);

WriteArray(OutputChoice, Resh, OutputFilePath);

End;

Var

N: Integer;

CharSet: TSet;

Resh: TArr;

A, TrMatr: TMatrix;

Begin

N:= 0;

DelProg();

A := VvodMatr(CharSet);

TrMatr := DelTreugMatr(A);

Resh := SchitCorn(TrMatr);

RabPols(N, A, TrMatr, Resh);

End.

**Код программы С++:**

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

void delProg() {

cout << "Данная программа выполняет Прямой ход в системе лин. уравн., используя

метод Гаусса \n";

}

int viborRaboti() {

int vib;

bool isIncorrect;

cout << "0 - Работа с консолью, 1 - работа с файлом ";

do {

isIncorrect = false;

cout << "Выбор: ";

cin >> vib;

if (cin.fail() || cin.peek() != '\n') {

isIncorrect = true;

cout << "\nОшибка ввода. ";

cin.clear();

while (cin.get() != '\n');

}

if (!(isIncorrect) && (vib != 1) && (vib != 0)) {

isIncorrect = true;

cout << "\nЧисло должно быть либо 0, либо 1. ";

cin.clear();

while (cin.get() != '\n');

}

} while (isIncorrect);

return vib;

}

string pTF() {

string path;

bool isIncorrect;

do {

isIncorrect = false;

cout << "Укажите путь к файлу: ";

cin >> path;

ifstream fin(path);

if (!fin.is\_open()) {

cout << "Такой файл не найден. ";

isIncorrect = true;

}

fin.close();

} while (isIncorrect);

return path;

}

int EnterN(){

const int MIN = 2;

int n;

bool isIncorrect;

do

{

cout << "Введите размер массива: ";

isIncorrect = false;

cin >> n;

if (cin.get() != '\n')

{

isIncorrect = true;

cout << "Введены некорректные данные. Повторите ввод:\n";

cin.clear();

while (cin.get() != '\n');

}

if ((!isIncorrect) && (n < MIN))

{

isIncorrect = true;

cout << "Введено недопустимое число элементов. Повторите ввод:\n";

cin.clear();

}

} while (isIncorrect);

return n;

}

int readSizeFile(string path){

int n, i = 0;

bool isIncorrect;

ifstream fin(path);

isIncorrect = false;

if (!fin.eof())

{

fin >> n;

if (fin.fail() || fin.peek() != '\n')

{

cout << "Ошибка считывания размера с файла. \n";

isIncorrect = true;

}

if (!isIncorrect && (n < 2))

{

cout << "Размер массива должен быть числом, большим 1. \n";

isIncorrect = true;

}

}

fin.close();

if (isIncorrect)

{

n = EnterN();

}

return n;

}

int readN(const int vib, const string path) {

int n;

n = 0;

if (vib == 0)

n = EnterN();

if (vib == 1)

n = readSizeFile(path);

return n;

}

void seeColUravnSistem(const int vib, string path, const int n) {

ofstream fout(path);

bool isIncorrect;

if (vib == 0)

{

cout << "Количество уравнений в системе: " << n << ". \n";

}

if (vib == 1)

{

do

{

isIncorrect = false;

ofstream fout(path);

try

{

fout << n << "\n";

}

catch (...)

{

cout << "Ошибка.\n";

isIncorrect = true;

path = pTF();

}

fout.close();

} while (isIncorrect);

}

}

float\*\* matrixConsole(const int n) {

float\*\* a = new float\* [n];

bool isIncorrect;

int p;

p = n + 1;

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

{

a[i] = new float[n + 1];

}

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

{

for (int j = 0; j < p; j++)

{

cout << "A[ " << i + 1 << "" << j + 1 << "]: ";

do

{

isIncorrect = false;

cin >> a[i][j];

if (cin.fail())

{

isIncorrect = true;

cout << "Ошибка ввода. Введите корректное значение: \n";

cin.clear();

while (cin.get() != '\n');

}

if (!isIncorrect && (a[i][j] < -2000 || a[i][j] > 2000))

{

isIncorrect = true;

cout << "Ошибка. Введите значения от -2000 до 2000: \n";

}

} while (isIncorrect);

}

}

return a;

}

float\*\* matrixFile(const int n, const string path) {

float\*\* a = new float\* [n];

int e;

e = 0;

int p;

p = n + 1;

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

a[i] = new float[n + 1];

bool isIncorrect;

string matrixInp;

ifstream fin(path);

fin >> matrixInp;

while (!fin.eof())

{

string symbol;

fin >> symbol;

e++;

}

if (e > n \* (n + 2))

{

cout << "Ошибка. Поработайте с консолью. \n";

a = matrixConsole(n);

}

else

{

fin.close();

fin.open(path);

fin >> matrixInp;

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

{

for (int j = 0; j < p; j++)

{

do

{

isIncorrect = false;

try

{

fin >> a[i][j];

}

catch (...)

{

cout << "Ошибка. Поработайте с консолью. \n";

isIncorrect = true;

a = matrixConsole(n);

}

if (!isIncorrect && (a[i][j] < -2000 || a[i][j] > 2000))

{

cout << "Ошибка. Поработайте с консолью. \n";

isIncorrect = true;

a = matrixConsole(n);

}

} while (isIncorrect);

}

}

}

fin.close();

return a;

}

float\*\* fillMatrix(const int vib, const string path, const int n) {

float\*\* a = new float\* [n];

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

{

a[i] = new float[n + 1];

}

if (vib == 0)

{

a = matrixConsole(n);

}

if (vib == 1)

{

a = matrixFile(n, path);

}

return a;

}

void printMatrix(const int vib, string path, float\*\* a, const int n) {

bool isIncorrect;

ofstream fout;

if (vib == 0)

{

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

{

cout << "";

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

cout << "" << a[i][j] << " \t";

}

cout << "" << a[i][n] << "; \n";

}

cout << "\n";

}

if (vib == 1)

{

fout.open(path, ios::app);

fout << "\n";

do

{

isIncorrect = false;

try

{

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

{

fout << "";

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

fout << "" << a[i][j] << " \t";

}

fout << "" << a[i][n] << "; \n";

}

fout << "\n";

}

catch (...)

{

cout << "Ошибка. Проверьте данные на правильность ввода \n";

isIncorrect = true;

path = pTF();

}

} while (isIncorrect);

fout.close();

}

}

float\*\* delTreugMatr(const int n, float\*\* a) {

float\*\*trMatr = new float\* [n];

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

trMatr[i] = new float[n + 1];

int columns = n + 1;

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

{

for (int j = 0; j < columns; j++)

trMatr[i][j] = a[i][j];

}

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

{

int startPosJ = i + 1;

for (int j = startPosJ; j < n; j++)

{

float temp = trMatr[j][i] / trMatr[i][i];

for (int k = i; k < n; k++)

{

trMatr[j][k] = trMatr[j][k] - temp \* trMatr[i][k];

}

trMatr[j][n] = trMatr[j][n] - temp \* trMatr[i][n];

}

}

return trMatr;

}

void writeTreugMatr(const int vib, string path, float\*\* a, const int n) {

bool isIncorrect;

ofstream fout;

if (vib == 0)

{

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

{

cout << " ";

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

{

cout << " " << a[i][j] << " ";

}

cout << " " << a[i][n] << "; \n";

}

cout << "\n";

}

if (vib == 1) {

fout.open(path, ios::app);

fout << "\n";

do

{

isIncorrect = false;

try

{

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

{

fout << "";

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

{

fout << " " << a[i][j] << " ";

}

fout << " " << a[i][n] << "; \n";

}

fout << "\n";

}

catch (...)

{

cout << "Ошибка. Поработайте с консолью. \n";

isIncorrect = true;

path = pTF();

}

} while (isIncorrect);

fout.close();

}

}

float\* schitCorn(const int n, float\*\* trMatr) {

float\* rez = new float[n];

float temp;

int pstn;

pstn = n - 1;

int k;

k = 1;

float dividend;

dividend = 0;

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

{

if (trMatr[pstn][pstn] != 0)

{

dividend = trMatr[pstn][pstn + k];

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

{

dividend = dividend - trMatr[pstn][pstn + i - j] \* rez[j];

}

temp = dividend / trMatr[pstn][pstn];

}

else {

temp = 0;

}

rez[i] = temp;

k++;

pstn--;

}

return rez;

}

void writeArray(const int vib, string path, float\* rez, const int n) {

bool isIncorrect;

ofstream fout;

if (vib == 0)

{

cout << "Корни: \n";

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

cout << "Значение " << i + 1 << " переменной " << rez[i] << ". \n";

}

if (vib == 1)

{

cout << "Запись найденных корней в файл. \n";

fout.open(path, ios::app);

fout << "\n";

do

{

isIncorrect = false;

try

{

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

fout << "Значение " << i + 1 << " переменной " << rez[i] << ". \n";

}

catch (...)

{

cout << "Ошибка. Проверьте вводимые значения. \n";

isIncorrect = true;

path = pTF();

}

} while (isIncorrect);

fout.close();

}

}

void nachRab(int& vibPutVhoda, int& n, string& putVhoda) {

vibPutVhoda = viborRaboti();

if (vibPutVhoda == 1)

{

putVhoda = pTF();

}

n = readN(vibPutVhoda, putVhoda);

}

void conRab(int& vibPutVihoda, string& putVihoda, int& n, float\*\*& a, float\*\*& trMatr, float\*& rez) {

vibPutVihoda = viborRaboti();

if (vibPutVihoda == 1)

{

putVihoda = pTF();

}

SeeColUravnSistem(vibPutVihoda, putVihoda, n);

printMatrix(vibPutVihoda, putVihoda, a, n);

writeTreugMatr(vibPutVihoda, putVihoda, trMatr, n);

writeArray(vibPutVihoda, putVihoda, rez, n);

}

int main() {

setlocale(LC\_ALL, "Rus");

int vibPutVhoda, vibPutVihoda, n;

string putVhoda, putVihoda;

delProg();

nachRab(vibPutVhoda, n, putVhoda);

float\*\* a = new float\* [n];

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

{

a[i] = new float[n];

}

a = fillMatrix(vibPutVhoda, putVhoda, n);

float\*\* trMatr = new float\* [n];

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

{

trMatr[i] = new float[n];

}

trMatr = delTreugMatr(n, a);

float\* rez = new float[n];

rez = schitCorn(n, trMatr);

conRab(vibPutVhoda, putVihoda, n, a, trMatr, rez);

return 0;

}

**Код программы Java:**

import java.util.Scanner;

import java.io.\*;

public class LABA25 {

private static final Scanner scan = new Scanner(System.in);

public static void delProg() {

System.out.println("Данная программа выполняет Прямой ход в системе лин.

уравн., используя метод Гаусса" + "\n");

}

public static int viborRaboti() {

int vib;

vib = 0;

boolean isIncorrect;

do {

isIncorrect = false;

try {

vib = Integer.parseInt(scan.nextLine());

} catch (Exception e) {

System.out.println("Ошибка ввода.");

isIncorrect = true;

}

if (!isIncorrect && (vib != 0 && vib != 1)) {

System.out.println("Число должно быть либо 0, либо 1. ");

isIncorrect = true;

}

} while (isIncorrect);

return vib;

}

public static String pTF() {

String path;

boolean isIncorrect;

do {

isIncorrect = false;

System.out.println("Укажите путь к файлу: ");

path = scan.nextLine();

File file = new File(path);

if (!file.exists()) {

isIncorrect = true;

System.out.println("Такой файл не найден. ");

}

if (!path.endsWith(".txt")) {

isIncorrect = true;

System.out.println("У файла должно быть расширение txt. ");

}

} while (isIncorrect);

return path;

}

public static int enterN() {

int n;

boolean isIncorrect;

n = 0;

do {

isIncorrect = false;

System.out.print("Введите размер массива: ");

try {

n = Integer.parseInt(scan.nextLine());

} catch (Exception e) {

isIncorrect = true;

System.out.println("Ошибка ввода. ");

}

if (!isIncorrect && (n < 2)) {

isIncorrect = true;

System.out.println("Введены некорректные данные. Повторите ввод:");

}

} while (isIncorrect);

return n;

}

public static int readSizeFile(String path) {

int n;

boolean isIncorrect;

n = 0;

isIncorrect = false;

try {

Scanner scannerFile = new Scanner(new File(path));

n = scannerFile.nextInt();

scannerFile.close();

} catch (Exception e) {

isIncorrect = true;

System.out.println("Ошибка считывания размера с файла. ");

}

if (!isIncorrect && n < 2) {

isIncorrect = true;

System.out.println("Размер массива должен быть числом, большим 1. ");

}

if (isIncorrect) {

System.out.println("Введите pазмер массива с консоли. ");

n = enterN();

}

return n;

}

public static void seeColUravnSistem(final int vib, int n, String path) {

boolean isIncorrect;

if (vib == 0)

System.out.println("Количество уравнений в системе: " + n + ".");

if (vib == 1) {

do {

isIncorrect = false;

try {

FileWriter writer = new FileWriter(path);

writer.write(n + "\n");

writer.close();

} catch (IOException e) {

isIncorrect = true;

System.out.println("Ошибка.");

path = pTF();

}

} while (isIncorrect);

}

}

public static int[][] matrixConsole(final int n) {

int[][] a = new int[n][n + 1];

boolean isIncorrect;

int p = n + 1;

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

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

System.out.println("A[" + (i + 1) + ", " + (j + 1) + "]: ");

do {

isIncorrect = false;

try {

a[i][j] = Integer.parseInt(scan.nextLine());

} catch (Exception e) {

System.out.println("Ошибка ввода. Введите корректное

значение: ");

isIncorrect = true;

}

if (!isIncorrect && (a[i][j] < -2000 || a[i][j] > 2000)) {

isIncorrect = true;

System.out.println("Ошибка. Введите значения от -2000 до

2000: ");

}

} while (isIncorrect);

}

}

return a;

}

public static int[][] matrixFile(final int n, String path) {

int[][] a = new int[n][ + 1];

int finalPosI = n + 1;

int i;

int count = 1;

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

try (BufferedReader fReader = new BufferedReader(new FileReader(path))){

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

fReader.readLine();

}

count++;

String[] integerInString = fReader.readLine().split(" ");

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

a[i][j] = Integer.parseInt(integerInString[j]);

}

} catch (Exception e) {

System.out.println("Ошибка. Поработайте с консолью. ");

a = matrixConsole(n);

}

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

if (a[i][j] < -2000 || a[i][j] > 2000) {

System.out.println("Ошибка. Поработайте с консолью. ");

a = matrixConsole(n);

}

}

}

return a;

}

public static void fillMatrix(final int vib, String path, final int[][] a, final int n) {

boolean isIncorrect;

if (vib == 0) {

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

{

System.out.print(" ");

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

System.out.print(" " + a[i][j] + " ");

}

System.out.print(" " + a[i][n] + "; \n");

}

System.out.print("\n");

}

if (vib == 1) {

do {

isIncorrect = false;

try {

FileWriter writer = new FileWriter(path, true);

BufferedWriter bufferWriter = new BufferedWriter(writer);

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

{

bufferWriter.write("|");

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

bufferWriter.write(a[i][j] + " ");

bufferWriter.write("| " + a[i][n] + "; \n");

}

bufferWriter.write("\n");

bufferWriter.close();

writer.close();

} catch (IOException e) {

isIncorrect = true;

System.out.println("Ошибка. Проверьте данные на правильность

ввода ");

path = pTF();

}

} while (isIncorrect);

}

}

public static float[][] delTreugMatr(final int[][] a, final int n) {

float[][] trMatr = new float[n][n + 1];

int columns = n + 1;

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

for (int j = 0; j < columns; j++)

trMatr [i][j] = a[i][j];

}

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

{

int startPosJ = i + 1;

for (int j = startPosJ; j < n; j++)

{

float temp = trMatr [j][i] / trMatr [i][i];

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

trMatr [j][k] = trMatr [j][k] - temp \* trMatr [i][k];

}

trMatr [j][n] = trMatr [j][n] - temp \* trMatr [i][n]; }

}

return trMatr ;

}

public static void writeTreugMatr(final int vib, String path, final float[][] a, final int n) {

boolean isIncorrect;

if (vib == 0) {

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

{

System.out.print(" ");

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

System.out.print(" " + a[i][j] + " ");

System.out.print(" " + a[i][n] + "; \n");

}

System.out.print("\n");

}

if (vib == 1) {

do {

isIncorrect = false;

try {

FileWriter writer = new FileWriter(path, true);

BufferedWriter bufferWriter = new BufferedWriter(writer);

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

{

bufferWriter.write("");

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

bufferWriter.write(a[i][j] + " ");

bufferWriter.write(" " + a[i][n] + "; \n");

}

bufferWriter.write("\n");

bufferWriter.close();

writer.close();

} catch (IOException e) {

isIncorrect = true;

System.out.println("Ошибка. Поработайте с файлом. ");

path = pTF();

}

} while (isIncorrect);

}

}

public static float[] schitCorn(float[][] trMatr) {

float[] rez = new float[trMatr.length];

float temp;

int pstn = trMatr.length - 1;

int k = 1;

float dividend;

for (int i = 0; i < trMatr.length; i++)

{

if (trMatr[pstn][pstn] != 0)

{

dividend = trMatr[pstn][pstn + k];

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

dividend = dividend - trMatr[pstn][pstn + i - j] \* rez[j];

}

temp = dividend / trMatr[pstn][pstn];

}

else {

temp = 0;

}

rez[i] = temp;

k++;

pstn--;

}

return rez;

}

public static void writeArray(final int vib, String path, final float[] rez, final int n ) {

boolean isIncorrect;

if (vib == 0) {

System.out.println("Корни: ");

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

{

System.out.println("Значение " + (i + 1) + " переменной: " + rez[i] +

".");

}

}

if (vib == 1) {

do {

isIncorrect = false;

try {

FileWriter writer = new FileWriter(path, true);

BufferedWriter bufferWriter = new BufferedWriter(writer);

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

{

bufferWriter.write("Значение " + (i + 1) + " переменной: " +

rez[i] + ".");

bufferWriter.write("\n");

}

bufferWriter.close();

writer.close();

} catch (IOException e) {

isIncorrect = true;

System.out.println("Ошибка. Поработайте с файлом.");

path = pTF();

}

} while (isIncorrect);

}

}

public static int[][] nachRab() {

int n;

int vibPutVhoda;

int[][] a = new int[0][];

String pathToIn;

System.out.println("0 - работа с консолью, 1 - работа с файлами.");

vibPutVhoda = viborRaboti();

if (vibPutVhoda == 0) {

n = enterN();

a = matrixConsole(n);

}

if (vibPutVhoda == 1) {

pathToIn = pTF();

n = readSizeFile(pathToIn);

a = matrixFile(n, pathToIn);

}

return a;

}

public static void conRab(int n, int[][] a, float[][] trMatr, float[] rez) {

int vibPutVihoda ;

String pathToOut = "";

System.out.println("0 - работа с консолью, 1 - работа с файлами.");

vibPutVihoda = viborRaboti();

if (vibPutVihoda == 1) {

pathToOut = pTF();

}

seeColUravnSistem(vibPutVihoda , n, pathToOut);

fillMatrix(vibPutVihoda , pathToOut, a, n);

writeTreugMatr(vibPutVihoda , pathToOut, trMatr, n);

writeArray(vibPutVihoda , pathToOut, rez, n);

}

public static void main (String[] rez) {

delProg();

int[][] a = nachRab();

float[][] trMatr = delTreugMatr(a, a.length);

float[] ansArr = schitCorn(trMatr);

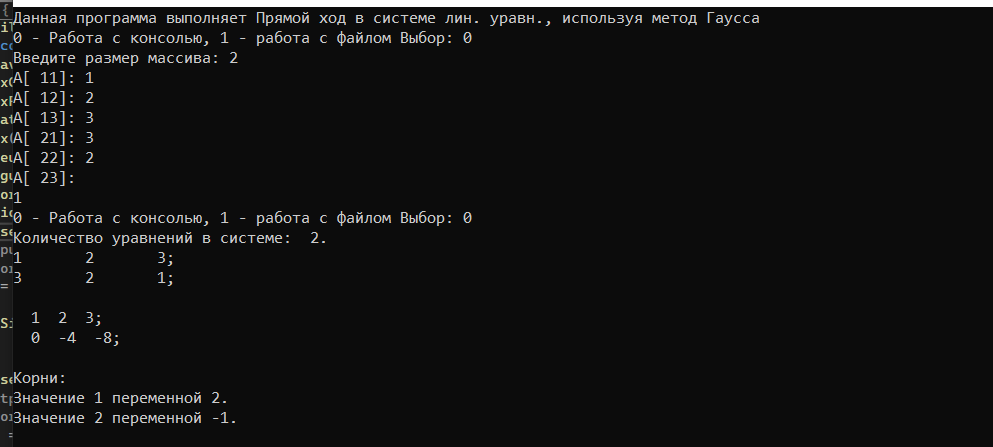
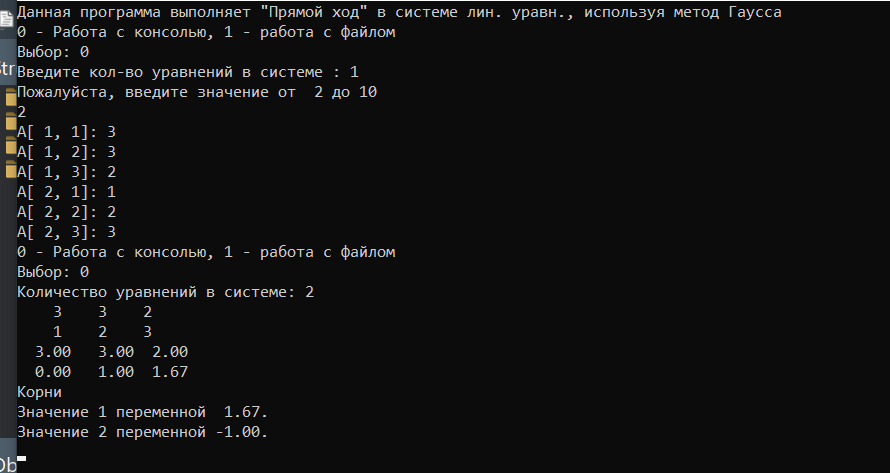
conRab(a.length, a, trMatr, ansArr);

scan.close();

}

}

**Скриншоты:**



**Блок-схема:**

