УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе №4.2

по предмету «Основы алгоритмизации и программирования»

Вариант 13

Выполнил:

Машевский Д.В

Гр. 351003

Проверил:

Данилова Г. В.

Минск 2024

**Задание:**

Вычислить определитель заданной матрицы, пользуясь формулой разложения по первой строке.

**Код программы Delphi:**

**unit Unit4Laba42;**

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.StdCtrls, Vcl.Grids,

Vcl.Imaging.jpeg, Vcl.Menus;

Const

MAX\_VALUE = 10000;

MAX\_LENGTH\_VALUE = 20;

CORRECT\_SYMBOLS: Set of Char = ['-', '0' .. '9', #8];

CORRECT\_FILE\_SYMBOLS: Set of Char = ['-', '0' .. '9', #10, #13, ' '];

COUNT\_OF\_NUMBERS = 3;

type

TArr = Array of Array of Integer;

TMatrix = Array Of Array Of Integer;

index = array [1 .. MAX\_VALUE] of boolean;

TForm4 = class(TForm)

StringGrid1: TStringGrid;

LabelInfo: TLabel;

EditYourMatrix: TEdit;

ButtonEnterValueMatrix: TButton;

MainMenu1: TMainMenu;

NFile: TMenuItem;

NInstruction: TMenuItem;

NDeveloper: TMenuItem;

NOpenFile: TMenuItem;

NSaveFile: TMenuItem;

ButtonGiveFinalResult: TButton;

LabelEnterMatrixValue: TLabel;

LabelFinalResult: TLabel;

procedure ButtonEnterValueMatrixClick(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure StringGrid1SetEditText(Sender: TObject; ACol, ARow: Integer;

const Value: string);

procedure EditYourMatrixChange(Sender: TObject);

procedure EditYourMatrixKeyPress(Sender: TObject; var Key: Char);

Procedure CheckEnter(Sender: TObject; var Key: Char);

procedure ButtonGiveFinalResultClick(Sender: TObject);

procedure NOpenFileClick(Sender: TObject);

procedure NSaveFileClick(Sender: TObject);

procedure NInstructionClick(Sender: TObject);

procedure EditYourMatrixKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure EditYourMatrixMouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure EditYourMatrixMouseUp(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure NDeveloperClick(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form4: TForm4;

implementation

{$R \*.dfm}

Function CheckZeroStr(Str: String): boolean;

Var

I, Len: Integer;

Fl: boolean;

Begin

Fl := False;

Len := Length(Str) - 1;

For I := 1 to Len do

Begin

If (Str[I] = '0') and (I <> High(Str)) then

Begin

If (Str[I + 1] <> '') then

Fl := True;

End;

End;

CheckZeroStr := Fl;

End;

Procedure DeleteColRow(Var NewMatrix: TMatrix; OldMatrix: TMatrix;

ColInd: Integer; RowInd: Integer = 0);

Var

Size, I, J, CurRow, CurCol: Integer;

Begin

CurCol := 0;

CurRow := 0;

Size := Length(OldMatrix) - 1;

SetLength(NewMatrix, Size, Size);

For I := 0 To High(OldMatrix) Do

Begin

If I <> RowInd Then

Begin

For J := 0 To High(OldMatrix) Do

If J <> ColInd Then

Begin

NewMatrix[CurRow, CurCol] := OldMatrix[I, J];

Inc(CurCol);

End;

CurCol := 0;

Inc(CurRow);

End;

End;

End;

Function CalcDet(CurMatrix: TMatrix): Integer;

Var

Det, I, Addition: Integer;

NewMatrix: TMatrix;

Begin

Det := 0;

If Length(CurMatrix) = 1 Then

Det := CurMatrix[0, 0]

Else

For I := 0 To High(CurMatrix) Do

If CurMatrix[0, I] <> 0 Then

Begin

DeleteColRow(NewMatrix, CurMatrix, I);

Addition := CurMatrix[0, I] \* CalcDet(NewMatrix);

If I Mod 2 = 1 Then

Addition := -Addition;

Inc(Det, Addition);

End;

CalcDet := Det;

End;

procedure TForm4.ButtonGiveFinalResultClick(Sender: TObject);

Var

A: TMatrix;

I, J, Det, K, YourMatrixValue: Integer;

begin

YourMatrixValue := StrToInt(EditYourMatrix.Text);

SetLength(A, YourMatrixValue, YourMatrixValue);

for I := 1 to YourMatrixValue do

for J := 1 to YourMatrixValue do

A[I - 1, J - 1] := StrToInt(StringGrid1.Cells[J, I]);

Det := 0;

Det := CalcDet(A);

LabelFinalResult.Caption := IntToStr(Det);

MainMenu1.Items[0].Items[1].Enabled := True;

end;

Procedure TForm4.CheckEnter(Sender: TObject; var Key: Char);

var

chars: set of Char;

editText: String;

begin

editText := (Sender as TEdit).Text;

if (Length(editText) = 0) then

begin

chars := ['0' .. '9', #8, #31];

end

else

chars := ['0' .. '9', #8, #31];

if not(Key in chars) then

Key := #0;

end;

procedure TForm4.ButtonEnterValueMatrixClick(Sender: TObject);

Var

ArrLen, I, J: Integer;

begin

MainMenu1.Items[0].Items[1].Enabled := False;

ArrLen := StrToInt(EditYourMatrix.Text);

StringGrid1.RowCount := ArrLen + 1;

StringGrid1.ColCount := ArrLen + 1;

For I := 1 to ArrLen do

Begin

StringGrid1.Cells[I, 0] := IntToStr(I) + '.';

End;

For J := 1 to ArrLen do

Begin

StringGrid1.Cells[0, J] := IntToStr(J) + '.';

End;

For I := 1 to ArrLen do

Begin

For J := 1 to ArrLen do

StringGrid1.Cells[I, J] := '';

End;

StringGrid1.Options := StringGrid1.Options + [goEditing];

StringGrid1.Enabled := True;

LabelFinalResult.Caption := '';

end;

procedure TForm4.EditYourMatrixChange(Sender: TObject);

Var

I, J, Len: Integer;

begin

ButtonEnterValueMatrix.Enabled := False;

StringGrid1.Enabled := False;

MainMenu1.Items[0].Items[1].Enabled := False;

Len := StringGrid1.ColCount;

Len := StringGrid1.RowCount;

For I := 1 to Len do

Begin

For J := 1 to Len do

StringGrid1.Cells[I, J] := '';

End;

If (EditYourMatrix.Text = '0') then

Begin

EditYourMatrix.Text := '';

LabelFinalResult.Caption := '';

End;

If (EditYourMatrix.Text = '') then

Begin

ButtonEnterValueMatrix.Enabled := False;

LabelFinalResult.Caption := '';

ButtonGiveFinalResult.Enabled := False;

End;

If (Length(EditYourMatrix.Text) <> 0) then

Begin

ButtonEnterValueMatrix.Enabled := True;

End;

end;

procedure TForm4.EditYourMatrixKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

If (Key = VK\_INSERT) and (SsShift in Shift) then

Key := 0

end;

procedure TForm4.EditYourMatrixKeyPress(Sender: TObject; var Key: Char);

Var

Value: Integer;

Begin

CheckEnter(Sender, Key);

If Key = #8 then

If (CheckZeroStr(EditYourMatrix.Text)) Then

EditYourMatrix.Text := '';

If TryStrToInt(EditYourMatrix.Text + Key, Value) Then

If (Value > MAX\_LENGTH\_VALUE) Then

Key := #0;

if EditYourMatrix.Text = '0' then

Begin

EditYourMatrix.Text := '';

End;

end;

procedure TForm4.EditYourMatrixMouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

If Button = mbRight then

EditYourMatrix.ReadOnly := True;

end;

procedure TForm4.EditYourMatrixMouseUp(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

If Button = mbRight then

EditYourMatrix.ReadOnly := True;

end;

procedure FillArr(var Arr: TArr; Path: string);

var

Value: Integer;

I, J: Integer;

InputFile: TextFile;

begin

AssignFile(InputFile, Path);

Reset(InputFile);

for I := 1 to High(Arr) do

for J := I to High(Arr) do

begin

Read(InputFile, Value);

Arr[I, J] := Value;

end;

CloseFile(InputFile);

end;

Function IsInt(Value: Char): Integer;

Var

Counter: Integer;

Begin

Counter := 0;

If not(Value in CORRECT\_FILE\_SYMBOLS) Then

Inc(Counter);

IsInt := Counter;

End;

Function CheckFileMatrixValues1(FileInputPath: String): boolean;

Var

InputFile: TextFile;

Value: Char;

IsChecked, Fl: boolean;

Counter: Integer;

Begin

IsChecked := False;

Counter := 0;

AssignFile(InputFile, FileInputPath);

Reset(InputFile);

While (not(Eof(InputFile)) and (IsChecked = False)) do

Begin

Read(InputFile, Value);

If StrToInt(Value) > MAX\_VALUE then

Begin

Inc(Counter);

End;

End;

If (Counter > 0) then

Begin

ShowMessage

('В файле некорректные данные:'#10'Было найдено значение матрицы,

превышающее максимальное.');

CheckFileMatrixValues1 := False;

End

Else

CheckFileMatrixValues1 := True;

CloseFile(InputFile);

End;

procedure TForm4.FormCreate(Sender: TObject);

begin

StringGrid1.Cells[0, 0] := 'Значения';

ButtonEnterValueMatrix.Enabled := False;

ButtonGiveFinalResult.Enabled := False;

StringGrid1.Enabled := False;

MainMenu1.Items[0].Items[1].Enabled := False;

end;

function CheckMaxValueInMatrices(const FilePath: string): boolean;

var

F: TextFile;

Value: Integer;

Line: string;

FoundMaxValue: boolean;

begin

Result := True;

FoundMaxValue := False;

AssignFile(F, FilePath);

Reset(F);

try

while not Eof(F) do

begin

ReadLn(F, Line);

if Line <> '' then

begin

var

Elements := Line.Split([' ']);

for var Element in Elements do

begin

if TryStrToInt(Element, Value) then

begin

if Value > MAX\_VALUE then

begin

Result := False;

FoundMaxValue := True;

end;

end;

end;

end;

if FoundMaxValue then

Break;

end;

finally

CloseFile(F);

end;

if not Result then

begin

ShowMessage

('В файле некорректные данные:'#10'Было найдено значение, превышающее

максимальное.');

CheckMaxValueInMatrices := False;

end

else

CheckMaxValueInMatrices := True;

end;

Function CheckFileMatrixValues(FileInputPath: String): boolean;

Var

InputFile: TextFile;

Value: Char;

IsChecked, Fl: boolean;

Counter: Integer;

IsHigher: boolean;

Begin

IsChecked := False;

IsHigher := False;

Counter := 0;

AssignFile(InputFile, FileInputPath);

Reset(InputFile);

While (not(Eof(InputFile)) and (IsChecked = False)) do

Begin

Read(InputFile, Value);

If IsInt(Value) > 0 then

Begin

IsChecked := True;

Inc(Counter);

End;

End;

If (Counter > 0) then

Begin

ShowMessage

('В файле некорректные данные:'#10'Был найден символ,не являющийся

числом');

CheckFileMatrixValues := False;

End

Else

CheckFileMatrixValues := True;

CloseFile(InputFile);

End;

Function CheckFile(Path: String): boolean;

Var

Fl: boolean;

InputFile: TextFile;

Begin

Fl := True;

Try

AssignFile(InputFile, Path);

Reset(InputFile);

Except

ShowMessage('Произошла ошибка:файл занят другим процессом');

Fl := False;

End;

CloseFile(InputFile);

CheckFile := Fl;

End;

procedure TForm4.NDeveloperClick(Sender: TObject);

Var

MessageBoxCaption: String;

MessageBoxText: String;

IsMessageShow: Boolean;

Begin

IsMessageShow := False;

if not IsMessageShow then

Begin

IsMessageShow := True;

MessageBoxCaption := 'О разработчике';

MessageBoxText :=

'Машевский Даниил Витальевич, группа 351003, Лабораторная №

4.2'#10'Задание:'#10' Вычислить определитель заданной матрицы, пользуясь

формулой разложения по первой строке.';

MessageBox(Handle, PChar(MessageBoxText),

PChar(MessageBoxCaption), MB\_OK);

End;

IsMessageShow := False;

end;

procedure TForm4.NInstructionClick(Sender: TObject);

Var

MessageBoxCaption: String;

MessageBoxText: String;

IsMessageShow: Boolean;

Begin

IsMessageShow := False;

if not IsMessageShow then

Begin

IsMessageShow := True;

MessageBoxCaption := 'Инструкция';

MessageBoxText :=

'1)Размер матрицы не должен превышать 20'#10'2)Размер значений в массиве не

должен превышать 9999'#10'3)Файл должен быть формата .txt'#10'4)Пример

вида матрицы в файле: '#10'3'#10'2 3 4'#10'3 2 3'#10'2 2 1'#10'5) В

файле должны наход. квадр. матрица и первым элем. файла - размерность.';

MessageBox(Handle, PChar(MessageBoxText),

PChar(MessageBoxCaption), MB\_OK);

End;

IsMessageShow := False;

end;

procedure TForm4.NOpenFileClick(Sender: TObject);

var

OpenDialog: TOpenDialog;

InputFile: TextFile;

NumArr1, NumArr2: TArr;

Values: string;

Len, I, J: Integer;

begin

OpenDialog := TOpenDialog.Create(nil);

OpenDialog.Title := 'Выберите файл';

OpenDialog.Filter := 'Текстовые файлы (\*.txt)|\*.txt';

if OpenDialog.Execute then

begin

if CheckFile(OpenDialog.FileName) then

begin

if CheckFileMatrixValues(OpenDialog.FileName) then

Begin

if CheckMaxValueInMatrices(OpenDialog.FileName) = True then

Begin

AssignFile(InputFile, OpenDialog.FileName);

Reset(InputFile);

ReadLn(InputFile, Len);

if Len > MAX\_LENGTH\_VALUE then

ShowMessage

('Произошла ошибка: значение массива в файле превышает

максимальное')

Else

Begin

EditYourMatrix.Text := IntToStr(Len);

StringGrid1.RowCount := Len + 1;

StringGrid1.ColCount := Len + 1;

SetLength(NumArr1, Len, Len);

For I := 1 to Len do

Begin

StringGrid1.Cells[I, 0] := IntToStr(I) + '.';

End;

For J := 1 to Len do

Begin

StringGrid1.Cells[0, J] := IntToStr(J) + '.';

End;

for I := 0 to Len - 1 do

begin

for J := 0 to Len - 1 do

begin

Read(InputFile, NumArr1[I][J]);

end;

end;

for I := 0 to Len - 1 do

begin

for J := 0 to Len - 1 do

begin

StringGrid1.Cells[J + 1, I + 1] :=

IntToStr(NumArr1[I][J]);

end;

end;

CloseFile(InputFile);

ButtonGiveFinalResult.Enabled := True;

End;

End;

end;

end;

end;

end;

Procedure WriteStrIntoFile(Path: String; S1: String);

Var

F: TextFile;

I, J: Integer;

Mes: String;

Begin

Mes := 'Результат вычислений:'#13#10;

AssignFile(F, Path);

Rewrite(F);

Write(F, Mes, S1);

CloseFile(F);

End;

procedure TForm4.NSaveFileClick(Sender: TObject);

var

SaveDialog: TSaveDialog;

MText, S1: String;

FileName, Path: String;

MessageBoxCaption: String;

MessageBoxText: String;

IsMessageShow: boolean;

InputFile: TextFile;

Massiv: TArr;

I, J, ArrLen: Integer;

begin

SaveDialog := TSaveDialog.Create(nil);

SaveDialog.Title := 'Сохранить файл';

SaveDialog.Filter := 'Текстовые файлы (\*.txt)|\*.txt';

SaveDialog.DefaultExt := 'txt';

S1 := LabelFinalResult.Caption;

if SaveDialog.Execute then

begin

FileName := SaveDialog.FileName;

if not FileName.EndsWith('.txt') then

FileName := FileName + '.txt';

Path := FileName;

WriteStrIntoFile(Path, S1);

SaveDialog.Free;

IsMessageShow := False;

if not IsMessageShow then

Begin

IsMessageShow := True;

MessageBoxCaption := 'Работа с файлом:';

MessageBoxText := 'Запись в файл прошла успешно!';

MessageBox(Handle, PChar(MessageBoxText),

PChar(MessageBoxCaption), MB\_OK);

End;

IsMessageShow := False;

end;

MainMenu1.Items[0].Items[1].Enabled := True;

end;

procedure TForm4.StringGrid1SetEditText(Sender: TObject; ACol, ARow: Integer;

const Value: string);

Var

I, ArrLen, Counter1, J, CellValue: Integer;

Begin

If not TryStrToInt(Value, CellValue) then

Begin

StringGrid1.Cells[ACol, ARow] := '';

End;

ArrLen := StrToInt(EditYourMatrix.Text);

Counter1 := 0;

For I := 1 to ArrLen do

for J := 1 to ArrLen do

If (StringGrid1.Cells[I, J] = '') Then

Inc(Counter1);

if Counter1 = 0 then

Begin

ButtonGiveFinalResult.Enabled := True;

End

Else

Begin

ButtonGiveFinalResult.Enabled := False;

End;

end;

end.

**Код программы Java:**

import java.util.Scanner;

import java.io.\*;

public class Main {

static Scanner scanner = new Scanner(System.in);

public static int inputChoice() {

int vib;

boolean isIncorrect;

vib = 0;

System.out.println("Введите 0, если хотите сделать ввод с консоли; 1, если

хотите ввод из файла. ");

do {

isIncorrect = false;

System.out.print("Выбор: ");

try {

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

} catch (Exception e) {

isIncorrect = true;

System.out.println("Ошибка ввода. ");

}

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

isIncorrect = true;

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

}

} while (isIncorrect);

return vib;

}

public static int enterN() {

int n;

boolean isIncorrect;

n = 0;

do {

isIncorrect = false;

System.out.print("Введите порядок матрицы: ");

try {

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

} catch (Exception e) {

isIncorrect = true;

System.out.println("Ошибка ввода. ");

}

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

isIncorrect = true;

System.out.println("Порядок матрицы должен быть числом, большим 1.

");

}

} while (isIncorrect);

return n;

}

public static int[][] createTriangularMatrix(int n) {

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

boolean isIncorrect;

int i;

int j;

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

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

do {

isIncorrect = false;

System.out.println("Enter a[" + (i + 1) + "][" + (j + 1) + "]:

");

try {

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

} catch (Exception e) {

isIncorrect = true;

System.out.println("Input error");

}

} while (isIncorrect);

}

}

return a;

}

public static String pTF() {

String path;

boolean isIncorrect;

do {

isIncorrect = false;

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

path = scanner.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 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 при считывании порядка матрицы. ");

}

if (!isIncorrect && n < 2) {

isIncorrect = true;

System.out.println("Значение порядка матрицы в файле должно быть больше

1. ");

}

if (isIncorrect) {

System.out.println("Введите порядок матрицы с консоли. ");

n = enterN();

}

return n;

}

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

int a[][];

a = new int[n][n];

int i;

int j;

try {

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

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

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

a[i][j] = scannerFile.nextInt();

}

}

scannerFile.close();

} catch (Exception e) {

System.out.println("Ошибка при считывании файла. ");

}

return a;

}

public static int[][] matrixRead(int vib) {

int[][] a;

int n;

String path;

if (vib == 0) {

n = enterN();

a = createTriangularMatrix(n);

} else {

path = pTF();

n = readSizeFile(path);

a = matrixFile(path, n);

}

return a;

}

public static int outputChoice() {

int vib;

boolean isIncorrect;

vib = 0;

System.out.println("Введите 0, если хотите сделать вывод в консоль; 1, если

хотите вывод в файл. ");

do {

isIncorrect = false;

System.out.print("Выбор: ");

try {

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

} catch (Exception e) {

isIncorrect = true;

System.out.println("\nОшибка ввода. ");

}

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

isIncorrect = true;

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

}

} while (isIncorrect);

return vib;

}

static int[][] deleteColRow(int[][] oldMatrix, int colInd, int rowInd) {

int size = oldMatrix.length-1;

int[][] newMatrix = new int[size][size];

int curCol = 0;

int curRow = 0;

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

if (i != rowInd) {

for (int j = 0; j < oldMatrix.length; j++) {

if (j != colInd) {

newMatrix[curRow][curCol] = oldMatrix[i][j];

curCol++;

}

}

curCol = 0;

curRow++;

}

}

return newMatrix;

}

static int calculateDet(int[][] matrix) {

int det = 0;

if (matrix.length == 1)

det = matrix[0][0];

else {

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

if (matrix[0][i] != 0) {

int[][] newMatrix = deleteColRow(matrix, i, 0);

int addition = matrix[0][i] \* calculateDet(newMatrix);

addition = i % 2 == 0 ? addition : -addition;

det += addition;

}

}

}

return det;

}

public static void printResultConsole(int determinanta, int[][] matrix) {

int n = matrix.length;

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

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

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

}

System.out.println();

}

System.out.println("Определитель: " + determinanta);

}

public static void printResultFile(int determinanta, int[][] matrix) {

int i;

int j;

String path;

path = pTF();

try (PrintWriter zapis = new PrintWriter(new FileWriter(path))) {

for (i = 0; i < matrix.length; i++) {

for (j = 0; j < matrix[i].length; j++) {

zapis.write(matrix[i][j] + " ");

}

zapis.write(System.lineSeparator());

}

zapis.write("Определитель: " + determinanta);

System.out.println("Запись в файл прошла успешно!");

} catch (IOException e) {

System.out.println("Не удалось записать данные в файл.");

}

}

public static void vivodSum(int vib, int determinanta, int [][] matrix) {

if (vib == 0) {

printResultConsole(determinanta, matrix);

} else {

printResultFile(determinanta, matrix);

}

}

public static void main(String[] args) {

int vib1;

int vib2;

int determinanta;

int[][] matrix;

vib1 = inputChoice();

matrix = matrixRead(vib1);

determinanta = calculateDet(matrix);

vib2 = outputChoice();

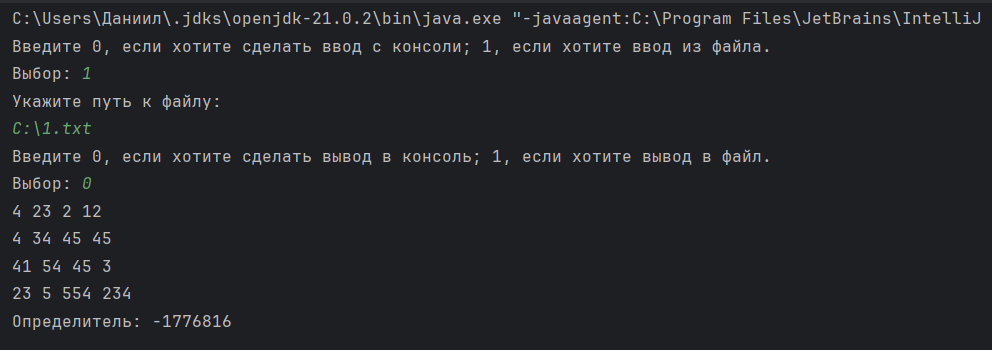
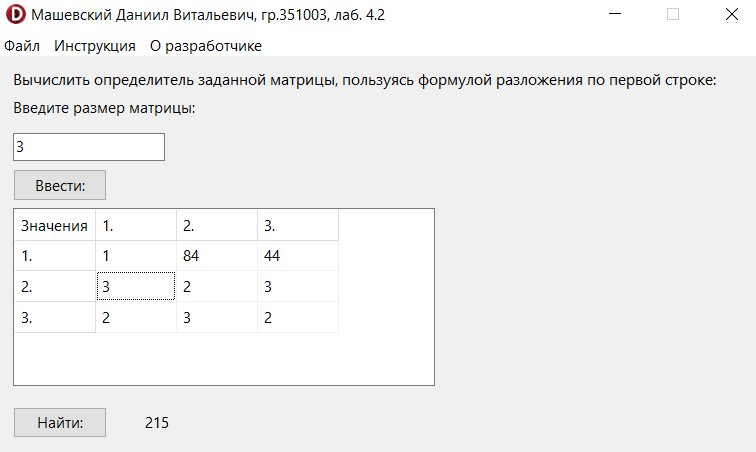
vivodSum(vib2, determinanta, matrix);

scanner.close();

}

}

**Скриншоты:**



**Блок-схема:**

