УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе №3.4

по предмету «Основы алгоритмизации и программирования»

Вариант 20

Выполнил:

Машевский Д.В

Гр. 351003

Проверил:

Данилова Г. В.

Минск 2023

**Задание:**

Построить магический квадрат нечётной степени.

**Код программы Delphi:**

Program MagicSquare;

Uses

SysUtils;

Type

TMatrix = Array Of Array Of Integer;

Procedure GetInfo();

Begin

Writeln('This program builds a magic square of odd degree ');

End;

Function ChooseOption: integer;

Var

Choice: integer;

IsCorrect: Boolean;

Begin

Choice:= 0;

Write('0 - consol; 1 - fail: ');

Repeat

IsCorrect := True;

Try

Readln(Choice);

Except

IsCorrect := False;

Write('Ошибка ввода. ');

End;

If IsCorrect and (Choice <> 1) and (Choice <> 0) then

Begin

IsCorrect := False;

Write('The number must be either 0 or 1.');

End;

Until IsCorrect;

ChooseOption := Choice;

End;

Function PTF(): String;

Var

Path: String;

IsCorrect: Boolean;

Begin

Repeat

IsCorrect := True;

Write('Path to file: ');

Readln(Path);

If Not FileExists(Path) Then

Begin

Write('No such file was found. ');

IsCorrect := False;

End;

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

Begin

IsCorrect := False;

Write('The file must have a txt extension.');

End;

Until IsCorrect;

PTF := Path;

End;

Function EnterSizeConsol: integer;

Var

Size: integer;

IsCorrect: Boolean;

Begin

Size := 0;

Writeln(' Enter square dimension');

Repeat

IsCorrect := True;

Try

Readln(Size);

Except

IsCorrect := False;

Write('Ошибка ввода. ');

End;

If IsCorrect And (Size < 2) or (Size mod 2 = 0) Then

Begin

IsCorrect := False;

Write('Re-Enter:');

End;

Until IsCorrect;

EnterSizeConsol := Size;

End;

Function ReadSizeFile(Path: String): integer;

Var

Size: integer;

IsCorrect: Boolean;

InputFile: TextFile;

Begin

Path := PTF();

AssignFile(InputFile, Path);

Reset(InputFile);

IsCorrect := True;

Try

Readln(InputFile, Size);

Except

IsCorrect := False;

Writeln('Error reading matrix order.');

End;

If IsCorrect And (Size < 2) or (Size mod 2 = 0) Then

Begin

IsCorrect := False;

Writeln('Re-Enter');

End;

Close(InputFile);

ReadSizeFile := Size

End;

Procedure СreateOddSquare(Size: integer; Var Matrix: TMatrix);

Var

I, J: integer;

P, L: integer;

DecSize: integer;

Begin

SetLength(Matrix, Size, Size);

DecSize := Size - 1;

For I := 0 To DecSize Do

For J := 0 To DecSize Do

Matrix[I, J] := 0;

J := Size Div 2;

I := 0;

P := Size \* Size;

For L := 1 To P Do

Begin

Matrix[I, J] := L;

I := (I - 1 + Size) Mod Size;

J := (J + 1) Mod Size;

If Matrix[I, J] <> 0 Then

Begin

I := (I + 1) Mod Size;

J := (J - 1 + Size) Mod Size;

J := (J - 1 + Size) Mod Size;

End;

End;

End;

Procedure ConsoleMatrix(Size: integer; Matrix: TMatrix);

Var

I, J: integer;

Begin

Dec(Size);

For I := 0 To Size Do

Begin

For J := 0 To Size Do

Write(Matrix[I, J]:4);

Writeln;

End;

End;

Procedure MatrixFile(Size: integer; Matrix: TMatrix);

Var

OutputFile: Text;

I, J: integer;

Path: String;

Begin

Path := PTF();

Assign(OutputFile, Path);

Rewrite(OutputFile);

Dec(Size);

For I := 0 To Size Do

Begin

For J := 0 To Size Do

Write(OutputFile, Matrix[I, J]:4);

Writeln(OutputFile);

End;

Close(OutputFile);

Writeln('The recording was successful');

End;

Procedure GiveResult();

Var

Size, ChoiceIn, ChoiceOut: integer;

Matrix: TMatrix;

Path: String;

Begin

GetInfo();

ChoiceIn := ChooseOption();

If ChoiceIn = 0 Then

Size := EnterSizeConsol()

Else

Size := ReadSizeFile(Path);

СreateOddSquare(Size, Matrix);

ChoiceOut := ChooseOption();

If ChoiceOut = 0 Then

ConsoleMatrix(Size, Matrix)

Else

MatrixFile(Size, Matrix);

Readln;

End;

Begin

GiveResult();

End.

**Код программы С++:**

#include <iostream>

#include <fstream>

using namespace std;

void getInfo() {

cout << "This program generates a magic square of odd order" << endl;

}

int\*\* createMatrix(int size) {

int i;

int j;

int\*\* magicSquare = new int\* [size];

for (i = 0; i < size; i++)

{

magicSquare[i] = new int[size];

for (j = 0; j < size; j++)

{

magicSquare[i][j] = 0;

}

}

return magicSquare;

}

int\*\* generateMagicSquare(int size) {

int i, j, p, l, decSize;

int\*\* magicSquare;

magicSquare = createMatrix(size);

decSize = size - 1;

for (i = 0; i <= decSize; i++)

{

for (j = 0; j <= decSize; j++)

{

magicSquare[i][j] = 0;

}

}

j = size / 2;

i = 0;

p = size \* size;

for (l = 1; l <= p; l++)

{

magicSquare[i][j] = l;

i = (i - 1 + size) % size;

j = (j + 1) % size;

if (magicSquare[i][j] != 0)

{

i = (i + 1) % size;

j = (j - 1 + size) % size;

j = (j - 1 + size) % size;

}

}

return magicSquare;

}

void consoleMatrix(int size, int\*\* magicSquare)

{

int i;

int j;

for (i = 0; i < size; i++)

{

for (j = 0; j < size; j++)

{

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

}

cout << endl;

}

}

string pathToFile() {

bool isInCorrect;

string path;

do

{

isInCorrect = false;

cout << "Path to file: ";

cin >> path;

ifstream fin(path);

if (!fin.is\_open())

{

cout << "File not found. ";

isInCorrect = true;

}

fin.close();

} while (isInCorrect);

return path;

}

void matrixFile(int size, int\*\* matrix) {

string path = pathToFile();

ofstream outputFile(path);

int i;

int j;

for (i = 0; i < size; i++)

{

for (j = 0; j < size; j++)

{

outputFile << matrix[i][j] << " ";

}

outputFile << endl;

}

outputFile.close();

cout << "The matrix was successfully written to the file";

}

int chooseOption() {

int choice;

choice = 0;

bool isInCorrect;

isInCorrect = false;

cout << "0 - console; 1 - file. ";

do

{

isInCorrect = false;

cout << "Choice: ";

cin >> choice;

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

{

isInCorrect = true;

cout << "Error. Re-Enter: ";

cin.clear();

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

}

if (!(isInCorrect) && (choice != 1) && (choice != 0))

{

isInCorrect = true;

cout << "Re-Enter. ";

cin.clear();

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

}

} while (isInCorrect);

return choice;

}

int readSizeConsole() {

int size;

bool isInCorrect;

int fsize;

fsize = 0;

size = 0;

do

{

isInCorrect = false;

cout << "Enter size of matrix: ";

cin >> size;

fsize = size % 2;

if (cin.fail() || cin.peek() != '\n' || size < 2 || fsize == 0)

{

isInCorrect = true;

cout << "Re-Enter: ";

cin.clear();

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

}

} while (isInCorrect);

return size;

}

int readSizeFile(string path)

{

int size;

int fsize;

int i;

bool isInCorrect;

ifstream fin(path);

isInCorrect = false;

i = 0;

fsize = 0;

size = 0;

fsize = size % 2;

if (!fin.eof())

{

fin >> size;

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

{

isInCorrect = true;

}

if ((isInCorrect && size < 2) || (fsize == 0))

{

isInCorrect = true;

}

}

fin.close();

if (!isInCorrect)

{

size = readSizeConsole();

}

return size;

}

void initializationMatrix(int size) {

int\*\* matrix = new int\* [size];

}

void fResuit() {

getInfo();

string path;

int size;

int choiceIn;

int choiceOut;

int\*\* matrix;

size = 0;

choiceIn = chooseOption();

initializationMatrix(size);

if (choiceIn == 0)

{

size = readSizeConsole();

}

else

{

path = pathToFile();

size = readSizeFile(path);

}

matrix = generateMagicSquare(size);

choiceOut = chooseOption();

if (choiceOut == 0)

{

consoleMatrix(size, matrix);

}

else

{

matrixFile(size, matrix);

}

}

int main() {

fResuit();

return 0;

}

**Код программы Java:**

import java.util.Scanner;

import java.io.File;

import java.io.FileWriter;

import java.io.IOException;

public class MagicSquare {

public static void main(String[] args) {

fResult();

}

public static void fResult(){

getInfo();

int choiceIn;

int choiceOut;

int size;

int[][] matrix;

choiceIn = chooseOption();

if (choiceIn == 0) {

size = enterSizeConsole();

} else {

String path = getPathToFile();

size = readSizeFile(path);

}

matrix = createOddSquare(size);

choiceOut = chooseOption();

if (choiceOut == 0) {

consoleMatrix(size, matrix);

} else {

String path = getPathToFile();

matrixFile(size, matrix, path);

}

}

public static void getInfo() {

System.out.println("This program builds a magic square of odd degree");

}

public static int chooseOption() {

int choice;

choice = 0;

boolean isInCorrect;

isInCorrect = false;

Scanner scanner = new Scanner(System.in);

System.out.print("0 - console; 1 - file: ");

while (!isInCorrect) {

try {

choice = scanner.nextInt();

isInCorrect = true;

} catch (Exception e) {

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

scanner.nextLine();

}

if (isInCorrect && (choice != 1) && (choice != 0)) {

isInCorrect = false;

System.out.print("The number must be either 0 or 1. ");

}

}

return choice;

}

public static String getPathToFile() {

String path = "";

boolean isInCorrect;

isInCorrect = false;

Scanner scanner = new Scanner(System.in);

while (!isInCorrect) {

System.out.print("Path to file: ");

path = scanner.nextLine();

File file = new File(path);

if (!file.exists()) {

System.out.print("No such file was found. ");

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

System.out.print("The file must have a txt extension. ");

} else {

isInCorrect = true;

}

}

return path;

}

public static int enterSizeConsole() {

int size;

size = 0;

boolean isInCorrect;

isInCorrect = false;

Scanner scanner = new Scanner(System.in);

System.out.println("Enter square dimension");

while (!isInCorrect) {

try {

size = scanner.nextInt();

isInCorrect = true;

} catch (Exception e) {

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

scanner.nextLine();

}

if (isInCorrect && (size < 2) || (size % 2 == 0)) {

isInCorrect = false;

System.out.print("Re-Enter: ");

}

}

return size;

}

public static int readSizeFile(String path) {

int size;

size = 0;

boolean isInCorrect;

isInCorrect = false;

try {

File file = new File(path);

Scanner scanner = new Scanner(file);

size = scanner.nextInt();

isInCorrect = true;

scanner.close();

} catch (IOException e) {

System.out.println("Error reading matrix order.");

}

if (isInCorrect && (size < 2) || (size % 2 == 0)) {

System.out.println("Re-Enter");

}

return size;

}

public static int[][] createOddSquare(int size) {

int[][] matrix;

int decSize;

int i, p, j, l;

matrix = new int[size][size];

decSize = size - 1;

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

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

matrix[i][j] = 0;

}

}

j = size / 2;

i = 0;

p = size \* size;

for (l = 1; l <= p; l++) {

matrix[i][j] = l;

i = (i - 1 + size) % size;

j = (j + 1) % size;

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

i = (i + 1) % size;

j = (j - 1 + size) % size;

j = (j - 1 + size) % size;

}

}

return matrix;

}

public static void consoleMatrix(int size, int[][] matrix) {

int decSize;

int i, j;

decSize = size - 1;

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

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

System.out.printf("%4d", matrix[i][j]);

}

System.out.println();

}

}

public static void matrixFile(int size, int[][] matrix, String path) {

int i, j;

try {

FileWriter writer = new FileWriter(path);

int decSize = size - 1;

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

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

writer.write(String.format("%4d", matrix[i][j]));

}

writer.write("\n");

}

writer.close();

System.out.println("The recording was successful");

} catch (IOException e) {

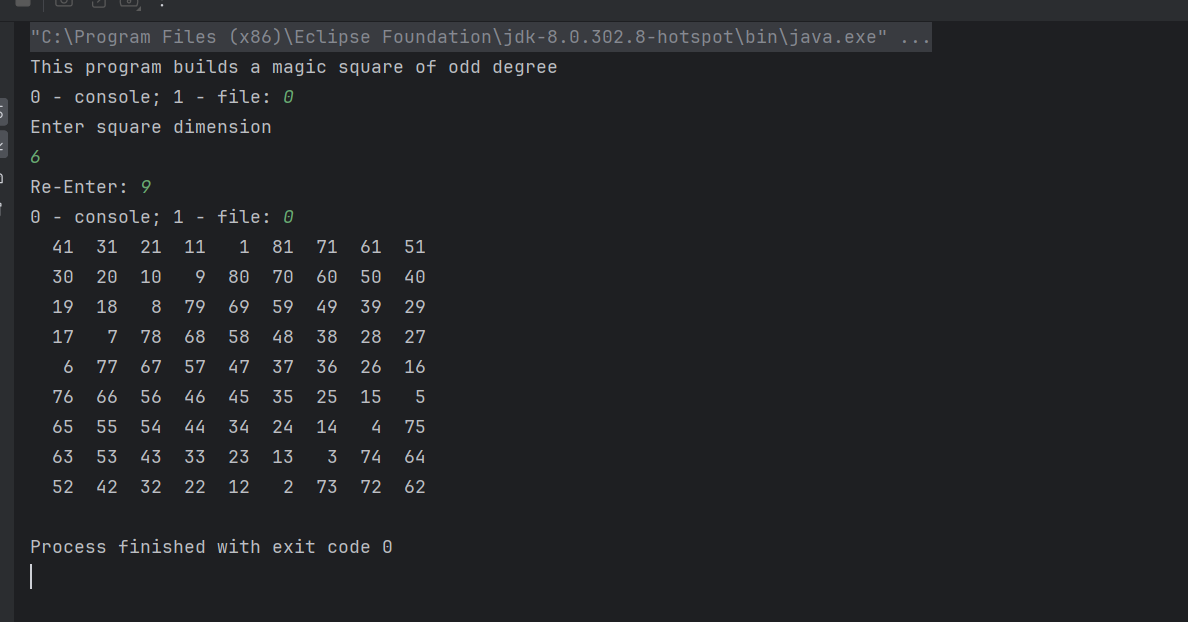
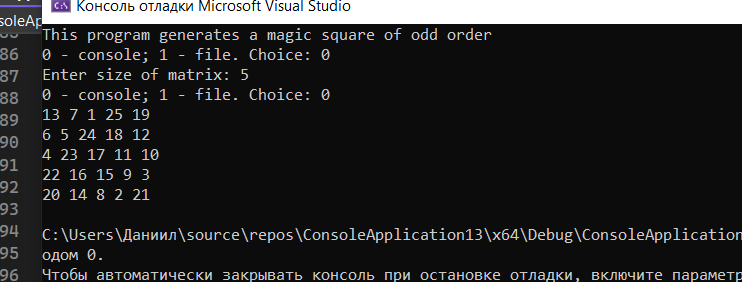
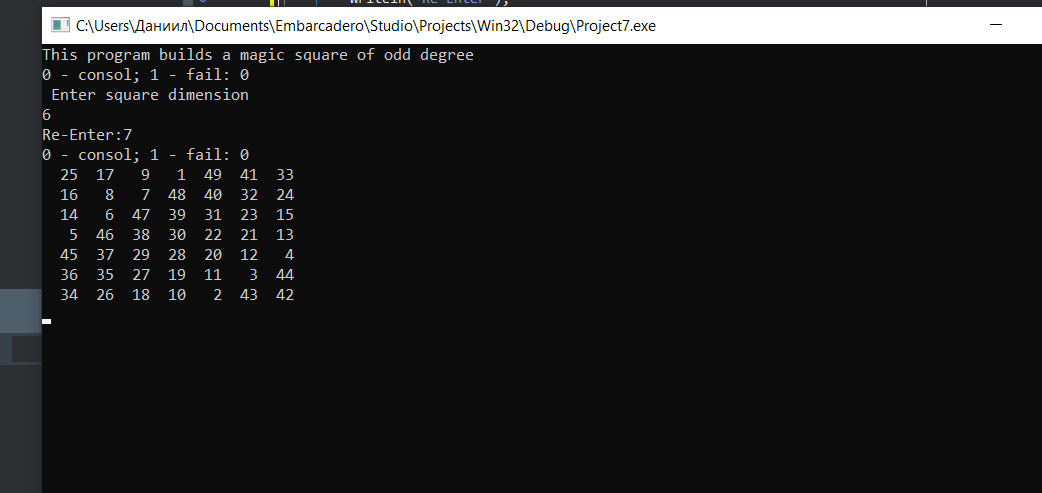
System.out.println("Error writing to file.");

}

}

}

**Скриншоты:**



**Блок-схема:**

