Министерство образования Калининградской области

государственное бюджетное учреждение Калининградской области

профессиональная образовательная организация

«Колледж информационных технологий и строительства»

(ГБУ КО ПОО «КИТиС»)

**Отчет по учебной практике**

УП.01 Разработка модулей программного обеспечения для компьютерных систем

по ПМ.01 Разработка модулей программного обеспечения для компьютерных систем

Специальность 09.02.07 «Информационные системы и программирование»

Сроки прохождения практики:

с «12» октября 2022 г. по «01» ноября 2022 г.

Место практики ГБУ КО ПОО «КИТиС»

|  |  |
| --- | --- |
| Выполнил: | студент 4 курса,  группы ИСп 19-2к  Майоров Алексей Андреевич  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |
| Проверила: | Большакова-Стрекалова Анна Викторовна  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |

Калининград, 2022

Содержание

Введение…………………………………………………………………………...3

Массивы……………………………………………………………………………3

1.Раздел…………………………………………………………………………….4

1.1 Задача 1………………………………………………………………………...4

1.2 Задача 2………………………………………………………………………...7

1.3 Задача 3……………………………………………………………………….10

1.4 Задача 4……………………………………………………………………….13

1.5 Задача 5……………………………………………………………………….14

1.6 Задача 6……………………………………………………………………….17

2.Раздел

Список использованных источников…………………………………………25

Приложения

Введение

Массивы

Массив в Java представляет собой класс, при этом имя объекта класса массива является объектной ссылкой на динамическую память, в которой хранятся элементы массива. Элементами массива, в свою очередь, могут быть значения базового типа или объекты. Элементы массива проиндексированы, индексирование элементов начинается с нуля. Для объявления ссылки на массив можно записать пустые квадратные скобки после имени типа, например: int a[]. Аналогичный результат получится при записи int []a. Массивы в языке Java являются динамическими. Существует два способа создания массива: с помощью оператора new или с помощью прямой инициализации присваиванием значений элементам массива в фигурных скобках. Значения элементов неинициализированного массива, для которого выделена память, устанавливаются в значения по умолчанию для массива базового типа или null для массива объектных ссылок.

1 Основной раздел

1.1

Задача 1

Условие:

Найти корни квадратного уравнения. Параметры уравнения передавать с командной строкой.

Блок-схема:

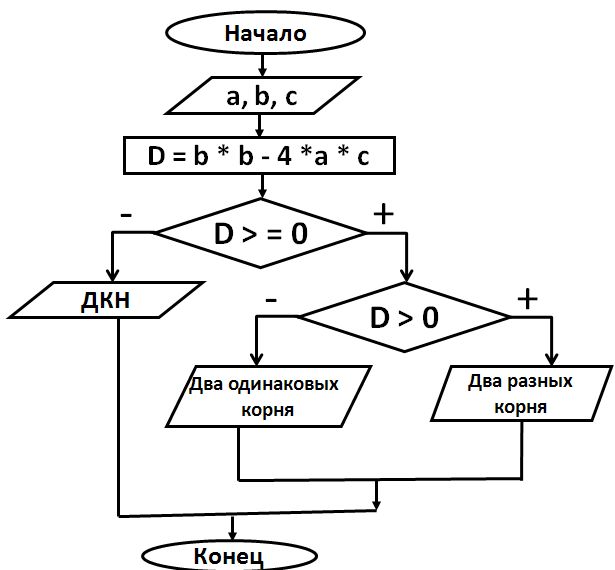


Рисунок 1.1 – Блок схема

Код программы:

package Main;

import java.util.Scanner;

public class Main {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

double a, b, c;

double D;

System.out.println("Программа решает квадратное уравнение вида:");

System.out.println("ax^2 + bx + c = 0");

System.out.println("Введите a, b и c:");

Scanner in = new Scanner(System.in);

a = in.nextDouble();

b = in.nextDouble();

c = in.nextDouble();

D = b \* b - 4 \* a \* c;

if (D > 0) {

double x1, x2;

x1 = (-b - Math.sqrt(D)) / (2 \* a);

x2 = (-b + Math.sqrt(D)) / (2 \* a);

System.out.println("Корни уравнения: x1 = " + x1 + ", x2 = " + x2);

}

else if (D == 0) {

double x;

x = -b / (2 \* a);

System.out.println("Уравнение имеет единственный корень: x = " + x);

}

else {

System.out.println("Уравнение не имеет действительных корней!");

}

}

};

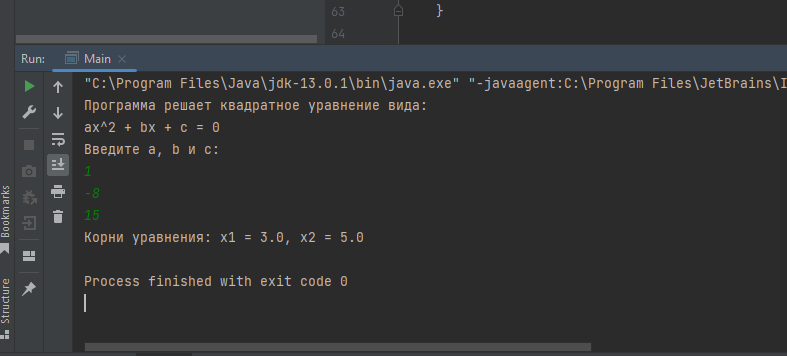


Рисунок 1 - Результат выполнения

1.2

Задача 2

Условие:

. Определить класс Булева матрица (BoolMatrix). Реализовать методы для логического сложения (дизъюнкции), умножения и инверсии матриц. Реализовать методы для подсчета числа единиц в матрице и упорядочения строк в лексикографическом порядке

Блок схема:

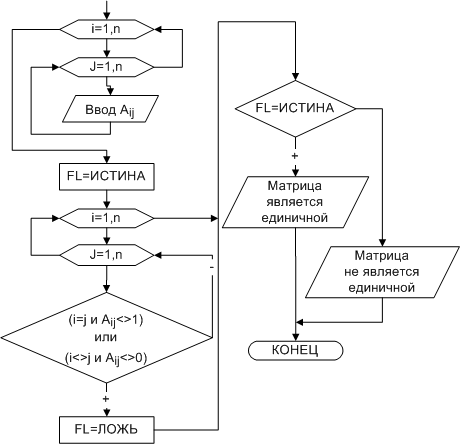


Рисунок 1.2 – Блок схема

Код программы:

package Main;

import java.util.Random;

class Matrix {

private int n,m;

private boolean[][] firstMatrix, secondMatrix;

Matrix(int nSize, int mSize) {

n = nSize;

m = mSize;

firstMatrix = new boolean[n][m];

secondMatrix = new boolean[n][m];

}

private void fillMatrixByRandom(boolean[][] matrix) { // Заполнение матрицы

Random rand = new Random();

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

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

matrix[i][j] = rand.nextBoolean();

}

}

}

private void printMatrix(boolean[][] matrix, String name) { // Печать матрицы

System.out.println(name);

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

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

System.out.print((matrix[i][j] ? 1 : 0) + "\t");

}

System.out.println();

}

}

private void inverseMatrix(boolean[][] matrix) { // Инверсия матрицы (отрицание)

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

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

matrix[i][j] = !matrix[i][j];

}

}

System.out.println();

}

private boolean[][] foldMatrix(boolean[][] matrix ,boolean[][] foldedMatrix) { // Сложение матрицы

boolean[][] foldResult = new boolean[n][m];

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

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

foldResult[i][j] = matrix[i][j] | foldedMatrix[i][j];

}

}

System.out.println();

return foldResult;

}

void run() {

fillMatrixByRandom(firstMatrix);

printMatrix(firstMatrix, "Первая матрица");

fillMatrixByRandom(secondMatrix);

printMatrix(secondMatrix, "Вторая матрица");

inverseMatrix(firstMatrix);

printMatrix(firstMatrix, "Первая матрица после инверсии");

inverseMatrix(secondMatrix);

printMatrix(secondMatrix, "Вторая матрица после инверсии");

printMatrix(foldMatrix(firstMatrix,secondMatrix), "Результат сложения двух инвертированных матриц");

}

}

public class BooleanMatrix {

public static void main(String[] args) {

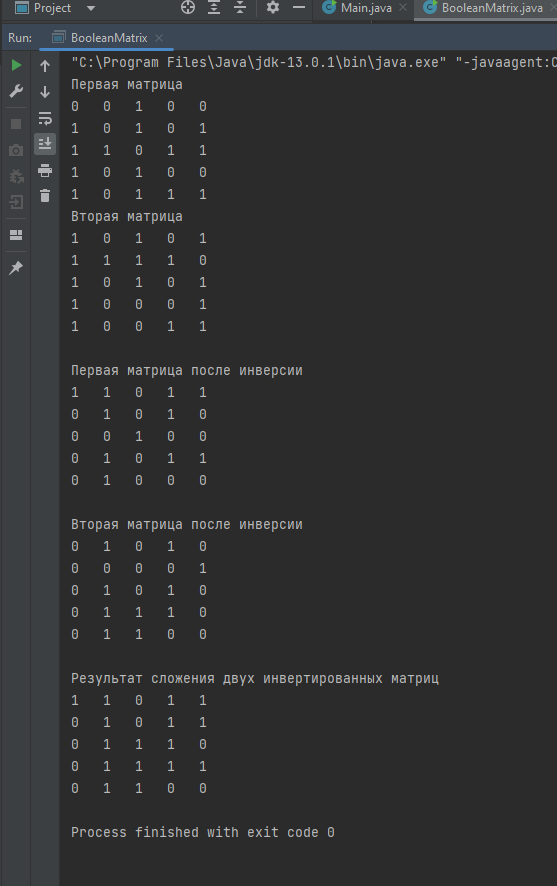
Matrix matrix = new Matrix(5,5);

matrix.run();

}

}

Рисунок 2 - Результат выполнения



1.3

Задача 3

Условие:

Создать объект класса Простая дробь, используя класс Число. Методы: вывод на экран, сложение, вычитание, умножение, деление

Блок схема:

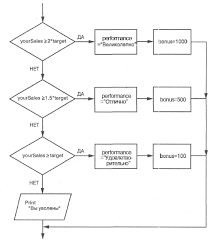


Рисунок 1.3 – Блок схема

Код программы:

**class** NumberTest{

**public** **static** **void** main(**String**[] args){

**int** i = 0;

**short** sh = 0;

**double** d = 0;

**Integer** kl = **new** **Integer**(55);

**Integer** k2 = **new** **Integer**(100);

**Double** dl = **new** **Double**(3.14);

**try**{

      i = **Integer**.parselnt(args[0]);

     sh = **Short**.parseShort(args[0]);

      d = **Double**.parseDouble(args[1]);

     dl = **new** **Double**(args[1]);

     kl = **new** **Integer**(args[0]);

    }**catch**(**Exception** e){}

**double** x = 1.0/0.0;

**System**.out.println("i = " + i) ;

**System**.outjprintln("sh - " + sh) ;

**System**.out.println("d. = " + d) ;

**System**.out.println("kl.intValue() = " + kl.intValue());

**System**.out.println("dl.intValue() '= "'+ dl.intValuei));

   System.out.println("kl > k2? " + kl.compareTo(k2));

   System.out.println ("x = " + x);

   System.out.println("x isNaN? " + Double.isNaN(x));

   System.out.println("x islnfinite? " + Double.islnfinite(x));

   System.out.println("x == Infinity? " +

          (x == Double.POSITIVE\_INFINITY) );

   System.out.println("d = " + Double.doubleToLongBits(d));

   System.out.println("i = " + Integer.toBinaryString(i));

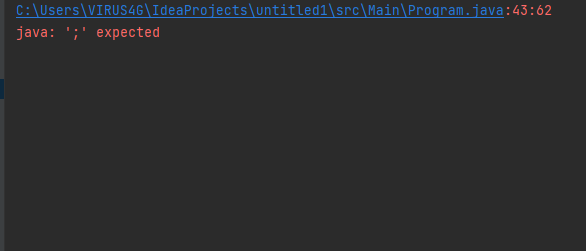
   System.out.println("i = " + Integer.toHexString(i));

   System.out.println("i = " + Integer.toOctalString(i));

 }

}

Рисунок 3 - Результат выполнения



1.4

Задача 4

Условие:

Создать класс Cправочная Cлужба Oбщественного Tранспорта с внутренним классом, с помощью объектов которого можно хранить информацию о времени, линиях маршрутов и стоимости проезда.

Блок схема:

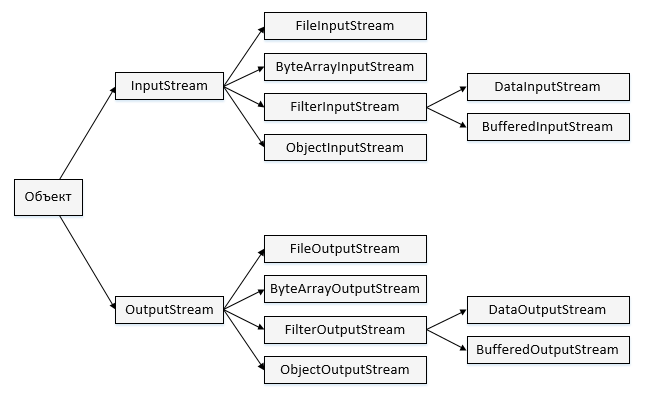


Рисунок 1.4 – Блок схема

Код программы:

Рисунок 5 – Результат выполнения

??????

1.5

Задача 5

Условие:

В тексте найти и напечатать все слова максимальной и все слова минимальной длины.

Блок схема:

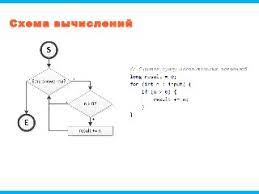


Рисунок 1.5 – Блок схема

**import** java.util.\*;

**import** java.lang.\*;

**import** java.io.\*;

*/\* Name of the class has to be "Main" only if the class is public. \*/*

**class** Ideone

{

**private** **static** **final** **String**[] WORDS\_ARRAY = **new** **String**[] {

        "APL", "PASCAL", "PHP", "APL", "ALGOL", "ADA", "SNOBOL", "PASCAL", "APL", "PHP"

    };

**private** **static** **int** mResultArraySize;

**private** **static** **String**[] mResultArray;

**public** **static** **void** main (**String**[] args) **throws** java.lang.**Exception**

    {

**int** minimalLength = WORDS\_ARRAY[0].length();

**int** maximalLength = minimalLength;

**for** (**int** i = 0; i < WORDS\_ARRAY.length; i++) {

**int** currentWordLength = WORDS\_ARRAY[i].length();

            minimalLength = currentWordLength < minimalLength ? currentWordLength : minimalLength;

            maximalLength = currentWordLength > maximalLength ? currentWordLength : maximalLength;

        }

**for** (**int** i = 0; i < WORDS\_ARRAY.length; i++) {

**int** currentWordLength = WORDS\_ARRAY[i].length();

**if** ( (currentWordLength == minimalLength) || (currentWordLength == maximalLength) ) {

                mResultArraySize++;

            }

        }

        mResultArray = **new** **String** [mResultArraySize];

        mResultArraySize = 0;

**for** (**int** i = 0; i < WORDS\_ARRAY.length; i++) {

**int** currentWordLength = WORDS\_ARRAY[i].length();

**if** ( (currentWordLength == minimalLength) || (currentWordLength == maximalLength) ) {

**boolean** isContains = **false**;

**for** (**int** j = 0; j < mResultArraySize; j++) {

**if** (WORDS\_ARRAY[i].equals(mResultArray[j])) {

                        isContains = **true**;

**break**;

                    }

                }

**if** (!isContains) mResultArray[mResultArraySize++] = WORDS\_ARRAY[i];

            }

        }

**System**.out.print("Result: " );

**for** (**int** i = 0; i < mResultArraySize; i++) {

**System**.out.print(mResultArray[i] + " ");

        }

    }

}

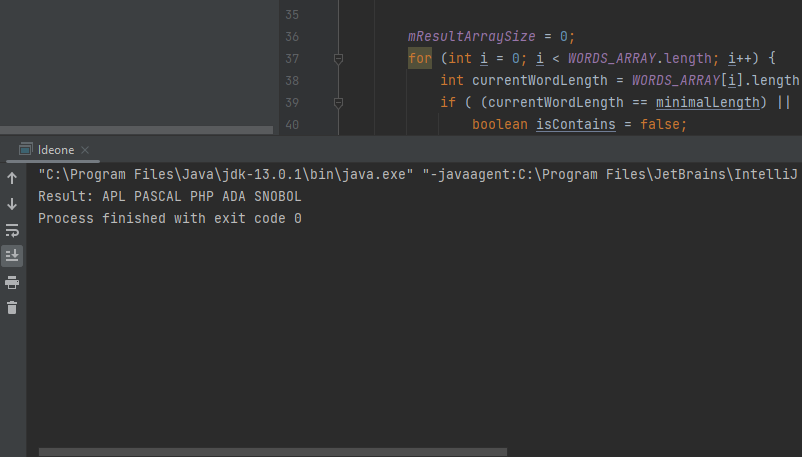


Рисунок 5 – Результат выполнения

1.6

Задача 6

Условие:

Вариант A

Выполнить задания на основе варианта А задачи 3, контролируя состояние потоков ввода/вывода. При возникновении ошибок, связанных с корректностью выполнения математических операций, генерировать и обрабатывать исключительные ситуации. Предусмотреть обработку исключений, возникающих при нехватке памяти, отсутствии требуемой записи (объекта) в файле, недопустимом значении поля и т. д.

Вариант B

Выполнить задания из варианта В задачи 3, реализуя собственные обработчики исключений и исключения ввода/вывода.

Блок схема:

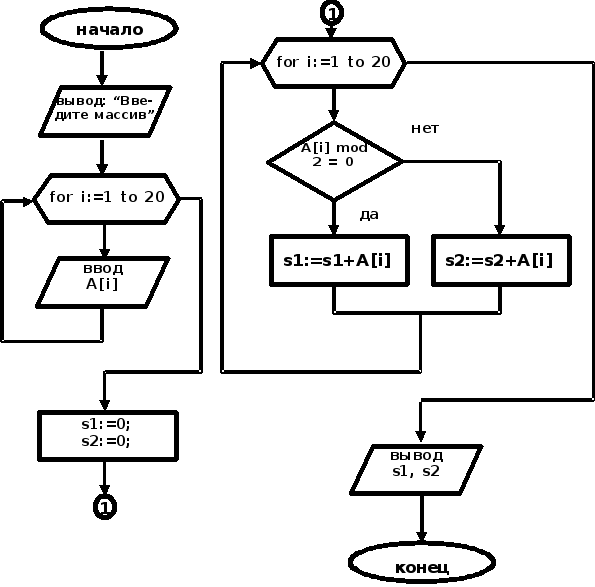


Рисунок 1.6 – Блок схема

Код программы:

package by.practical5.run;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

public class ExceptionExample {

public static String fileName = "exampleFile.txt";

public static String filePath =

"/Users/olga/Documents/workspace/ADP/src/by/practical5/run/";

public static void main(String[] args) {

// для получения ошибки раскомментируйте 6 строку

// triggerStackOverflowError();

int[] intArray = { 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000

};

/\*

\* RuntimeException – это непроверяемых(unchecked) исключения. Они

возникают во

\* время выполнения приложения. К таким исключениям относится,

например,

\* NullPointerException. Они не требуют обязательного заключения в

блок

\* try-catch. Когда RuntimeException возникает, это свидетельствует

о

\* ошибке, допущенной программистом (неинициализированный объект,

выход

\* за пределы массива и т.д.). Поэтому данное исключение не нужно

\* обрабатывать, а нужно исправлять ошибку в коде, чтобы исключение

\* вновь не возникало.

\*/

// Примеры непроверяемых исключений

// Деление на ноль приведет к ArithmeticException

// раскомментируйте строки 35 - 36 для того, чтобы получить

// ArithmeticException

// так как непроверяемых исключения не обязательно заключать

// в try-catch

// блок,

// то при компиляции компилятор не выдаст ошибку

// double res1 = intArray[8] / 0;

// System.out.println("res1 = " + res1);

try {

double catchedRes1 = intArray[8] / 0;

System.out.println("catchedRes1 = " + catchedRes1);

} catch (ArithmeticException e) {

System.out.println("Catching exception: " + e);

}

// внутри одного блока можно обрабатывать более одного исключения

try {

Double d = null;

System.out.println(" d = " + d.toString()); // сгенерируется

// NullPointerException,

// т.к. d не

// проинициализирован

double res2 = intArray[10] / 9; // сгенерируется

// ArrayIndexOutOfBoundsException,

// т.к. вышли за пределы массива

System.out.println("res2 = " + res2);

} catch (ArrayIndexOutOfBoundsException oobe) {

System.out.println("Getting " + oobe.getMessage());

} catch (NullPointerException npe) {

npe.printStackTrace();

System.out.println("Getting NullPointerException");

}

// также несколько типов исключения можно обработать в одном catch блоке

int intFirstArgument = 0;

try {

String firstArgument = args[1];

System.out.println("First argument = " + firstArgument);

intFirstArgument = Integer.parseInt(firstArgument);

System.out.println("You've entered valid integer value " +

intFirstArgument);

} catch (ArrayIndexOutOfBoundsException | NumberFormatException e) {

System.out.println("Handling exception:" + e);

}

System.out.println("after cathing exception");

double square = getSquare(2, 7);

System.out.println(String.format("square = %f", square));

System.out.println("after printing square");

// пример работы с потоками ввода/вывода

// запись строки в файл

String textToFile = "Hello from file";

boolean isFileUpdated = writeStringToFile(textToFile, fileName,

filePath);

System.out.println(String.format("file updated [%b] with text [%s]",

isFileUpdated, textToFile));

// чтение из файла в строку

System.out.println("file " + filePath + fileName + " content:" +

readFromFile(fileName, filePath));

}

static void triggerStackOverflowError() {

triggerStackOverflowError();

}

public static double getSquare(double height, double width) throws

IllegalArgumentException {

if (height < 0) {

throw new IllegalArgumentException("height<0");

} else if (width < 0) {

throw new IllegalArgumentException("width<0");

}

return height \* width;

}

public static boolean writeStringToFile(String textToFile, String

pFileName, String pFilePath) {

boolean fileUpdated = true;

FileOutputStream fos = null;

File file;

try {

// необходимо задать полный путь к файлу

file = new File(pFilePath + pFileName);

fos = new FileOutputStream(file);

/\*

\* Необходимо проверить, существует ли файл. Если

\* файла нет -создаем новый

\*/

if (!file.exists()) {

file.createNewFile();

}

/\*

\* Объект String не может быть напрямую записан в

\* файл. Необходимо

\* преобразовать объект String в массив bytes

\*/

byte[] bytesArray = textToFile.getBytes();

fos.write(bytesArray);

fos.flush();

System.out.println("File Written Successfully");

} catch (IOException ioe) { // обработка проверяемых

// (checked) исключений

fileUpdated = false;

ioe.printStackTrace();

} finally {

try {

if (fos != null) {

fos.close();

}

} catch (IOException ioe) {

System.out.println("Error in closing the Stream");

}

}

return fileUpdated;

}

public static String readFromFile(String pFileName, String pFilePath) {

File file = new File(pFilePath + pFileName);

FileInputStream fis = null;

StringBuilder builder = new StringBuilder();

try {

fis = new FileInputStream(file);

System.out.println("Размер файла (в байтах) : "

+ fis.available());

int iCh;

while ((iCh = fis.read()) != -1) {

char ch = (char) iCh;

builder.append(ch);

}

} catch (IOException e) {

e.printStackTrace();

} finally {

try {

if (fis != null)

fis.close();

} catch (IOException ex) {

ex.printStackTrace();

}

}

return builder.toString();

}

}

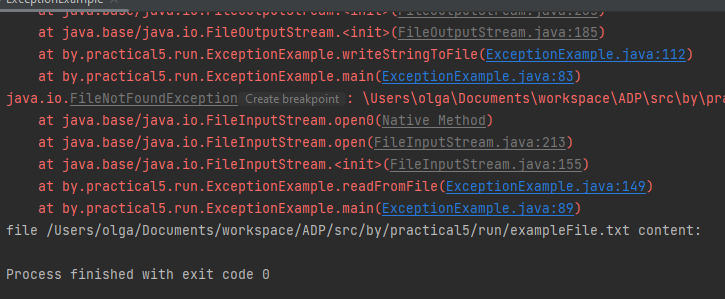


Рисунок 6 – Результат выполнения

Список использованных источников

1.Java Массивы (Java Array)

<https://javarush.ru/groups/posts/massivy-java>

[alexeyGGG/practica6 (github.com)](https://github.com/alexeyGGG/practica6)