

## Содержание

Введение.....	3
1. Анализ задачи .....	5
2. Теория.....	6
3. Выбор средств реализации .....	9
3.1 Выбор языка программирования .....	9
3.2 Выбор среды разработки приложения .....	10
4. Проектирование приложения.....	11
4.1 Алгоритм работы и схема приложения .....	11
4.2 Исходный код приложения .....	12
5. Тестирование приложения .....	13
Заключение .....	19
Литература .....	20
Приложение А .....	21

					УО «ВГТУ»					
Изм.	Лист	№ докум.	Подпись	Дата	Содержание			Лит.	Лист	Листов
Разраб.		Сташкевич. С								
Провер.		Бизюк А.Н							3	28
Реценз.								УО «ВГТУ» ИСАП гр. Ит-3		
Н. Контр.										
Утверд.										

## Введение

Оптимизация — в математике, информатике и исследовании операций задача нахождения экстремума (минимума или максимума) целевой функции в некоторой области конечномерного векторного пространства, ограниченной набором линейных и/или нелинейных равенств и/или неравенств. Теорию и методы решения задачи оптимизации изучает математическое программирование.

Математическое программирование — это область математики, разрабатывающая теорию, численные методы решения многомерных задач с ограничениями. В отличие от классической математики, математическое программирование занимается математическими методами решения задач нахождения наилучших вариантов из всех возможных.

На разработку методов прямого поиска для определения минимума функций и переменных было затрачено много усилий. Методы прямого поиска являются методами, в которых используются только значения функции. Практика показала, что этот метод эффективен и применим для широкого числа приложений. Простейшим методом поиска является метод покоординатного спуска.

Теоретически данный метод эффективен в случае единственного минимума функции. Но на практике он оказывается слишком медленным. Поэтому были разработаны более сложные методы, использующие больше информации на основании уже полученных значений функции.

					УО «ВГТУ»								
Изм.	Лист	№ докум.	Подпись	Дата									
Разраб.		Сташкевич. С			Введение				Лит.	Лист	Листов		
Провер.		Бизюк А.Н									4	28	
Реценз.													
Н. Контр.													
Утверд.													
					УО «ВГТУ» ИСАП гр. Ит-3								

## 1. Анализ задачи

Основной задачей данной курсовой работы является разработка приложения для решения задач теории игр, а именно: решение задач в чистых стратегиях (максимин, минимакс, цена игры, седловая точка), а также предусмотреть дополнительные функции:

- ввод данных из файла;
- сохранение результата в файл;

Для решения данной задачи необходимо:

- Изучить и использовать необходимые теоретические сведения.
- Выбрать язык программирования.
- Выбрать среду разработки.
- Реализовать приложение.
- Провести тестирование и отладку.

					УО «ВГТУ»		
Изм.	Лист	№ докум.	Подпись	Дата	Анализ задачи		
Разраб.		Стаскевич С					
Провер.		Бизюк А.Н					
Реценз.							
Н. Контр.							
Утверд.					УО «ВГТУ» ИСАП гр. Ит-3		
					Лит.	Лист	Листов
						5	28

## 2. Теория

### Решение матричных игр в чистых стратегиях

Целью участников любой матричной игры является выбор наиболее выгодных стратегий, доставляющих игроку А максимальный выигрыш, а игроку В минимальный проигрыш.

Определение. Чистую стратегию  $A_i$  игрока А называют оптимальной, если при ее применении выигрыш игрока А не уменьшается, какими бы своими стратегиями не пользовался игрок В. Оптимальной для игрока В называют чистую стратегию  $B_j$ , при использовании которой проигрыш игрока В не увеличивается, какие бы стратегии не применял игрок А.

При поиске оптимальных стратегий опираются на основополагающий принцип теории игр – принцип осторожности, в соответствии с которым каждый игрок, считая партнера по игре весьма разумным соперником, выбирает свои стратегии с учетом того, что противник ни в коем случае не упустит ни единой возможности использовать любую его ошибку в своих интересах. Поэтому игроки должны быть предельно внимательны при выборе каждой своей чистой стратегии.

Предположим, что игроку А надлежит сделать свой выбор. Анализируя платежную матрицу  $(a_{ij})_{i=1,m \ j=1,n}$  он для каждой чистой стратегии  $A_i$  сначала найдет минимальное значение  $\alpha_i$  ожидаемого выигрыша:

$$\alpha_i = \min_j \{a_{ij}\},$$

А затем из всех  $\alpha_i$  выделит наибольшее

$$\alpha = \max_i \{\alpha_i\}$$

УО «ВГТУ»

Изм.	Лист	№ докум.	Подпись	Дата	Теория		
Разраб.	Сташкевич С.						
Провер.	Бизюк А.Н				УО «ВГТУ» ИСАП гр. Ит-3		
Реценз.							
Н. Контр.							
Утверд.							
					Лит.	Лист	Листов
						6	28

И выберет соответствующую ему чистую стратегию  $A_i$ . Это и будет наиболее предпочтительная в данных условиях стратегия игрока А. Ее называют максиминной, поскольку она отвечает величине

$$\alpha = \max_i \min_j |a_{ij}| \quad (1)$$

Определение. Число  $\alpha$ , определяемое по этой формуле называется нижней чистой ценой игры (максимином). Оно показывает, какой минимальный выигрыш может получить игрок А, правильно применяя свои чистые стратегии при любых действиях игрока В.

Игрок В при оптимальном своём поведении должен стремиться по возможности за счёт своих стратегий максимально уменьшить выигрыш игрока А. Поэтому для игрока В отыскивается число

$$\beta_j = \max_i |a_{ij}|$$

Т. е. определяется максимальный выигрыш игрока А, при условии, что игрок В применит свою J-ю чистую стратегию, затем игрок В отыскивает такую свою стратегию, при которой игрок А получит минимальный выигрыш, т. е. находит число

$$\beta = \min_j \max_i |a_{ij}| \quad (2)$$

Определение. Число  $\beta$ , определяемое по формуле (2), называется чистой верхней ценой игры (минимаксом) и показывает, какой минимальный проигрыш за счёт своих стратегий может себе гарантировать игрок В (что соответственно определяет максимальный выигрыш игрока А при выборе правильных стратегий игроком В).

Другими словами, применяя свои чистые стратегии игрок А может обеспечить себе выигрыш не меньше  $\alpha$ , а игрок В за счёт применения своих чистых стратегий может не допустить выигрыш игрока А больше, чем  $\bar{\alpha}$ .

Определение. Если в игре с матрицей  $A$   $a=b$ , то говорят, что эта игра имеет седловую точку в чистых стратегиях и чистую цену игры  $v = \alpha = \beta$

Седловая точка – определяет пару чистых стратегий  $(i_0, j_0)$  соответственно игроков  $A$  и  $B$ , при которых достигается равенство  $\alpha = \beta$ . В это понятие вложен следующий смысл: если один из игроков придерживается стратегии, соответствующей седловой точке, то другой игрок не сможет поступить лучше, чем так же придерживаться стратегии, соответствующей седловой точке.

Математически это можно записать и иначе:

$$a_{i_0 j_0} \leq a_{i_0 j} \leq a_{i_0 j_0} \quad (3)$$

Где  $i, j$  – любые чистые стратегии соответственно игроков 1 и 2;  $(i_0, j_0)$  – стратегии, образующие седловую точку.

Таким образом, исходя из (3), седловой элемент  $a_{i_0 j_0}$  является минимальным в  $i_0$ -й строке и максимальным в  $j_0$ -м столбце в матрице  $A$ . Отыскание седловой точки матрицы  $A$  происходит следующим образом: В матрице  $A$  последовательно в каждой строке находят минимальный элемент и проверяют, является ли этот элемент максимальным в своём столбце. Если да, то он и есть седловой элемент, а пара стратегий, ему соответствующая, образует седловую точку. Пара чистых стратегий  $(i_0, j_0)$  игроков 1 и 2, образующая седловую точку и седловой элемент  $a_{i_0 j_0}$ , называется решением игры. При этом  $i_0$  и  $j_0$  называются оптимальными чистыми стратегиями соответственно игроков  $A$  и  $B$ .

### 3. Выбор средств реализации

#### 3.1 Выбор языка программирования

Для реализации курсового проекта был выбран язык программирования JAVA.

У языка Java™ есть много преимуществ перед другими языками программирования, что позволяет решать с его помощью практически любые задачи.

Ниже перечислены основные преимущества Java:

- Язык Java прост для изучения.

При разработке Java было уделено большое внимание простоте языка, поэтому программы на Java, по сравнению с программами на других языках, проще писать, компилировать, отлаживать и изучать.

- Java — это объектно-ориентированный язык.

Это позволяет создавать модульные программы, исходный код которых может использоваться многократно.

- Язык Java не зависит от платформы.

Одним из основных преимуществ языка Java является возможность переноса программ из одной системы в другую. Поскольку программы на Java не зависят от платформы как на уровне исходного кода, так и на двоичном уровне, их можно запускать в различных системах, что особенно важно для программ, предназначенных для World Wide Web.

					УО «ВГТУ»		
Изм.	Лист	№ докум.	Подпись	Дата			
Разраб.		Сташкевич С.			Выбор средств реализации	Лит.	Лист
Провер.		Бизюк А.Н					9
Реценз.							28
Н. Контр.						УО «ВГТУ» ИСАП гр. Ит-3	
Утверд.							

### 3.2 Выбор среды разработки приложения

Для реализации программы решения систем линейных уравнений была выбрана свободная интегрированная среда разработки приложений (IDE) NetBeans IDE, так как она обладает следующими достоинствами:

- Сразу готова к работе
- Бесплатно и с открытым исходным кодом
- Подключенный разработчик
- Мощный GUI Builder
- Поддержка стандартов и платформ Java
- Средства профилирования и отладки
- Поддержка динамического языка
- Расширяемая платформа
- Настраиваемые проекты
- Поддержка кода, отличного от Java
- Специальная поддержка



## 4. Проектирование приложения

### 4.1 Алгоритм работы и схема приложения

Для решения платежной матрицы в чистых стратегиях должны быть реализованы следующие функции приложения:

1. Ввод платежной матрицы.
2. Нахождение максимального элемента среди столбцов платежной матрицы.
3. Нахождение верхней чистой цены игры (MinMax).
4. Нахождение минимального элемента среди строк платежной матрицы.
5. Нахождение нижней чистой цены игры (MaxMin).
6. Нахождение чистой цены игры, если возможно.
7. Нахождение седловой точки, если возможно.
8. Вывод результата на экран.
9. Сохранение результата в файл.

					УО «ВГТУ»							
Изм.	Лист	№ докум.	Подпись	Дата	Проектирование приложения			Лит.	Лист	Листов		
Разраб.	Стахкевич С.									11	28	
Провер.	Бизюк А.Н							УО «ВГТУ» ИСАП гр. Ит-3				
Реценз.												
Н. Контр.												
Утверд.												

## Блок-схема приложения

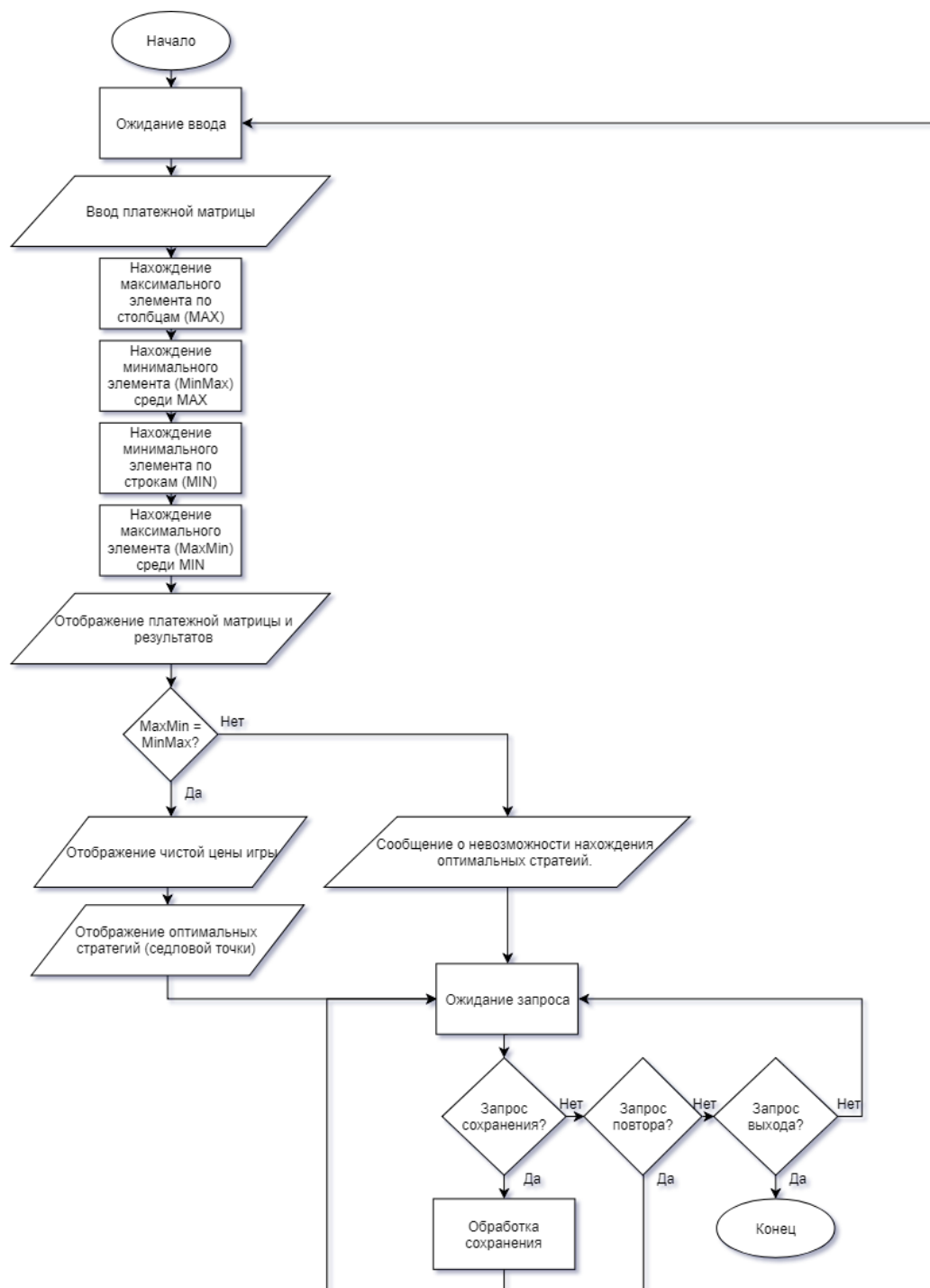


Рисунок 1. – Блок-схема приложения

### 4.2 Исходный код приложения

Исходный код программы представлен в Приложении А.

## 5. Тестирование приложения

Запуск приложения:

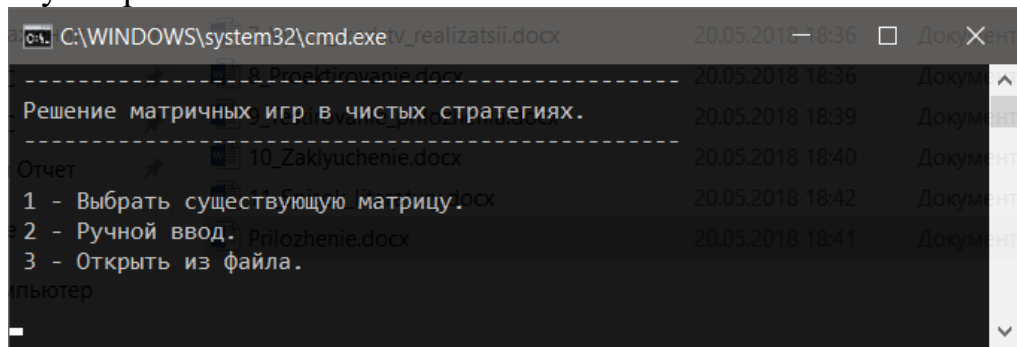


Рисунок 1 – Запуск.

Выбор из списка существующих матриц (если введенное число не равно «1-4», то выбирается матрица по умолчанию):

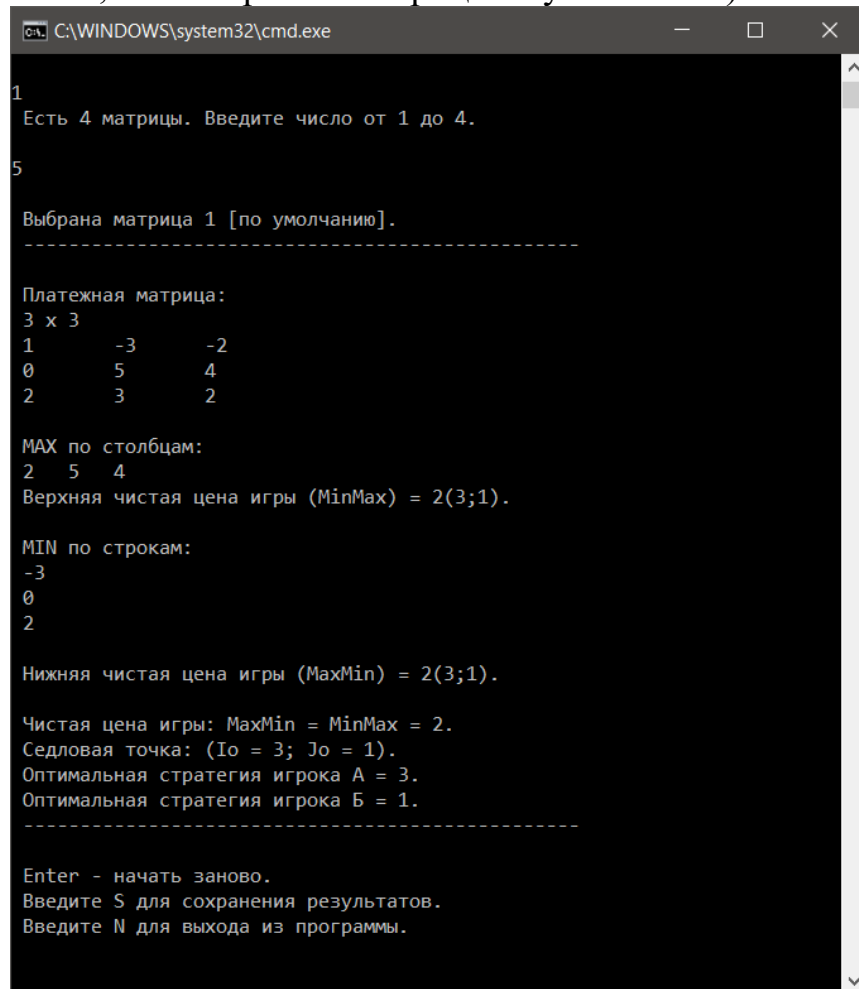
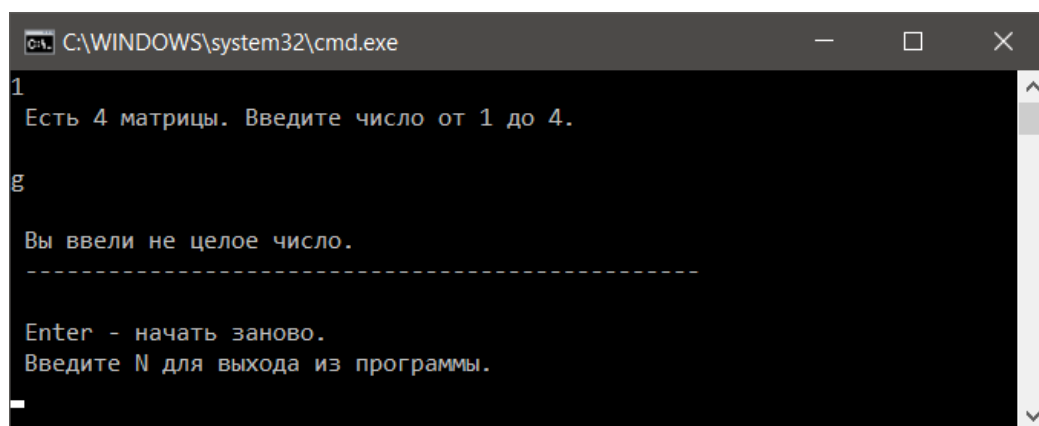


Рисунок 2 – Выбор существующей матрицы.

					УО «ВГТУ»		
Изм.	Лист	№ докум.	Подпись	Дата	Тестирование приложения		
Разраб.	Стажкович С						
Провер.	Бизюк А.Н						
Реценз.							
Н. Контр.							
Утверд.					УО «ВГТУ» ИСАП гр. Им-3		
					Лит.	Лист	Листов
						13	28

Если при выборе существующей матрицы ввести не число:



```
C:\WINDOWS\system32\cmd.exe
1
Есть 4 матрицы. Введите число от 1 до 4.

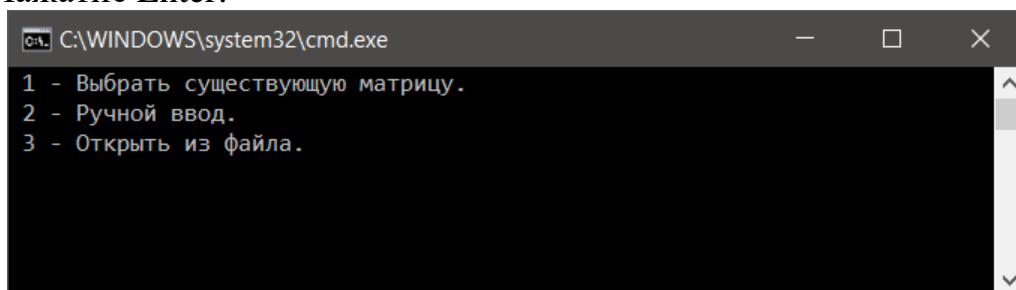
g

Вы ввели не целое число.
-----

Enter - начать заново.
Введите N для выхода из программы.
```

Рисунок 3 – Реакция программы на ввод нечислового значения.

Нажатие Enter:



```
C:\WINDOWS\system32\cmd.exe
1 - Выбрать существующую матрицу.
2 - Ручной ввод.
3 - Открыть из файла.
```

Рисунок 4 – Нажатие Enter.

## Ручной ввод:

```

C:\WINDOWS\system32\cmd.exe

2
Введите количество строк матрицы (минимум 2):
2
Введите количество столбцов матрицы (минимум 2):
2
Введите элемент матрицы [1][1]:
1
Введите элемент матрицы [1][2]:
2
Введите элемент матрицы [2][1]:
3
Введите элемент матрицы [2][2]:
4

-----

Платежная матрица:
2 x 2
1      2
3      4

MAX по столбцам:
3      4
Верхняя чистая цена игры (MinMax) = 3(2;1).

MIN по строкам:
1
3
Нижняя чистая цена игры (MaxMin) = 3(2;1).

Чистая цена игры: MaxMin = MinMax = 3.
Седловая точка: (Iо = 2; Jо = 1).
Оптимальная стратегия игрока А = 2.
Оптимальная стратегия игрока Б = 1.

-----

Enter - начать заново.
Введите S для сохранения результатов.
Введите N для выхода из программы.
  
```

Рисунок 5 – Ручной ввод.

Ввод буквы «S» для сохранения результата в файл:

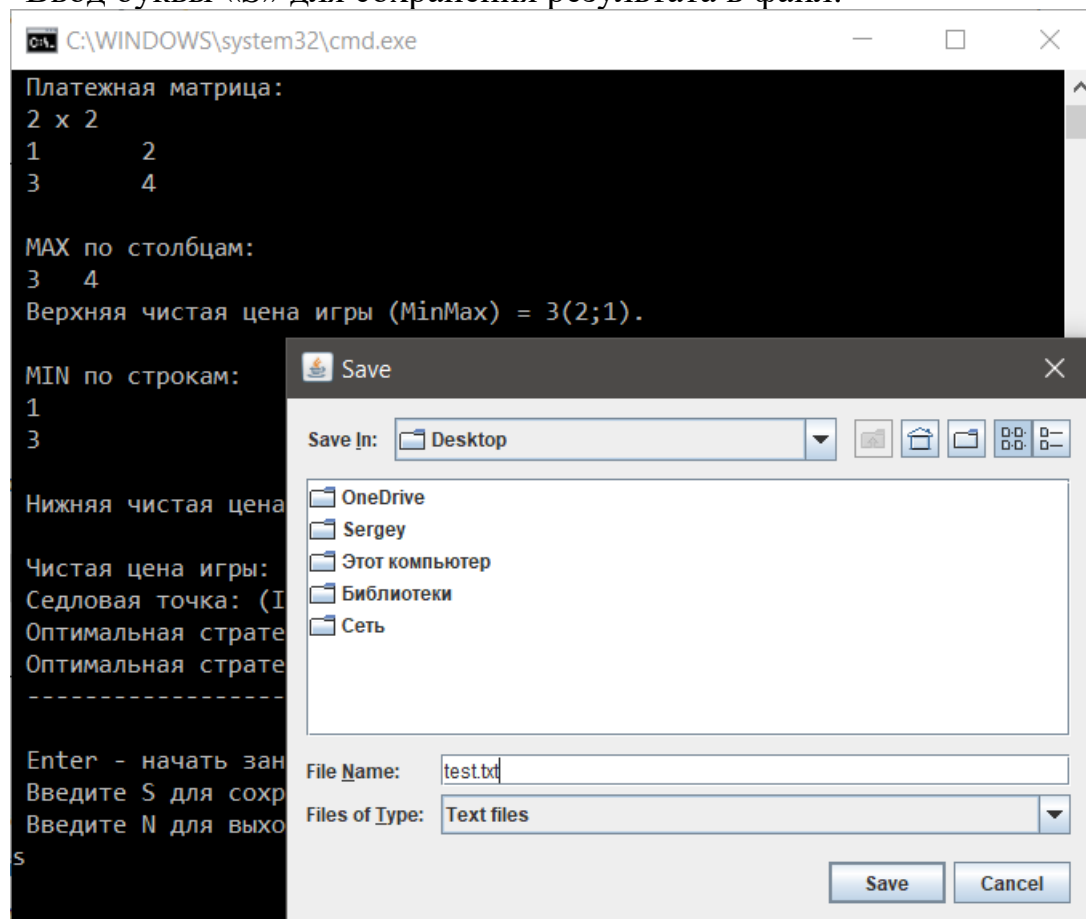


Рисунок 6 – Сохранение в файл.

Сообщение об успешном сохранении:

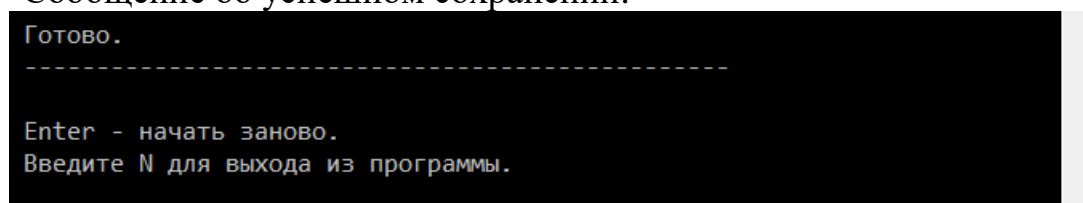


Рисунок 7 – Успешное сохранение.

### Содержимое сохраненного файла:

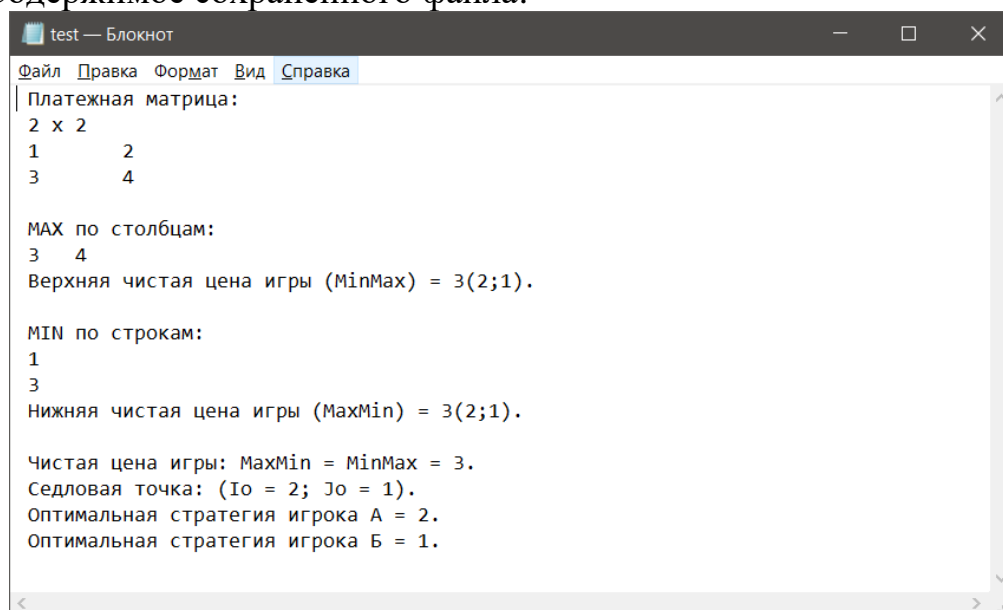


Рисунок 4 – Содержимое файла.

### Открытие матрицы из файла:

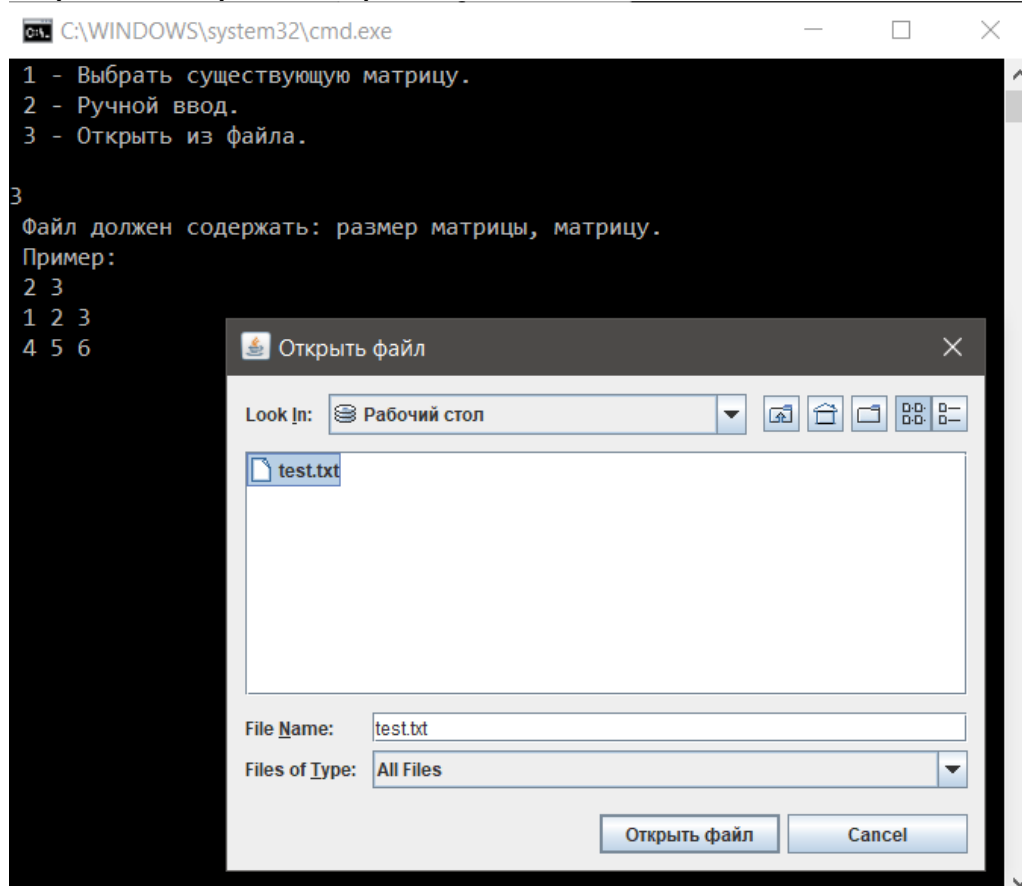


Рисунок 8 – Открытие матрицы из файла.

Результат открытия матрицы из файла:

```

C:\WINDOWS\system32\cmd.exe
Файл должен содержать: размер матрицы, матрицу.
Пример:
2 3
1 2 3
4 5 6
-----

Платежная матрица:
2 x 2
1      2
3      4

MAX по столбцам:
3      4
Верхняя чистая цена игры (MinMax) = 3(2;1).

MIN по строкам:
1
3

Нижняя чистая цена игры (MaxMin) = 3(2;1).

Чистая цена игры: MaxMin = MinMax = 3.
Седловая точка: (I0 = 2; J0 = 1).
Оптимальная стратегия игрока А = 2.
Оптимальная стратегия игрока Б = 1.
-----

Enter - начать заново.
Введите S для сохранения результатов.
Введите N для выхода из программы.

```

Рисунок 9 – Результат открытия матрицы из файла.

Ввод буквы «N» осуществляет закрытие программы.



## Заключение

В результате работы над данным курсовым проектом было разработано приложение для решения матричных игр в чистых стратегиях.

Выполнены все основные и дополнительные задачи.

При тестировании программы все ошибки были устранены.

					УО «ВГТУ»						
Изм.	Лист	№ докум.	Подпись	Дата							
Разраб.		Сташкевич С.			Заключение			Лит.	Лист	Листов	
Провер.		Бизюк А.Н								19	28
Реценз.								УО «ВГТУ» ИСАП гр. Им-3			
Н. Контр.											
Утверд.											

## Литература

1. Шилдт Г. Java 8. Руководство для начинающих.: Вильямс, 2015. – 720 с.
2. <http://matica.org.ua/metodichki-i-knigi-po-matematike/teoriia-igr-issledovanie-operacii/1-3-1-reshenie-matrichnykh-igr-v-chistykh-strategiiakh>
3. <https://math.semestr.ru/games/index.php>
4. <https://www.youtube.com/watch?v=4SeUROXfMfY>

					УО «ВГТУ»					
Изм.	Лист	№ докум.	Подпись	Дата	Литература			Лит.	Лист	Листов
Разраб.		Стажкович С								
Провер.		Бизюк А.Н							20	28
Реценз.								УО «ВГТУ» ИСАП зр. Им-3		
Н. Контр.										
Утверд.										

## Приложение А

### (Исходный код)

```

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.io.PrintStream;
import java.util.Scanner;
import javax.swing.JFileChooser;
import javax.swing.filechooser.FileNameExtensionFilter;
import org.apache.commons.io.output.TeeOutputStream;
/**
 *
 * @author Sergey
 */
public class PureStrategyConsole {
    int x1, y1, x2, y2;
    int maxmin, minmax;
    int[][] arrayS;
    boolean save = false;
    boolean bNaN = false;
    ByteArrayOutputStream buffer;
    OutputStream teeStream;

    PureStrategyConsole(){
        x1 = 0; y1 = 0;
        x2 = 0; y2 = 0;
        maxmin = Integer.MIN_VALUE;
        minmax = Integer.MAX_VALUE;
        arrayS = new int [0][0];
    }

    protected void saveToFileBeg() throws IOException{
        buffer = new ByteArrayOutputStream();
        teeStream = new TeeOutputStream(System.out, buffer);
        System.setOut(new PrintStream(teeStream));
        //System.setOut(new PrintStream(teeStream, true, "windows-1251"));
    }

```

					УО «ВГТУ» КР.019 1-40 05 01-01 ПЗ							
Изм.	Лист	№ докум.	Подпись	Дата								
Разраб.		Сташкевич С			Приложение				Лит.	Лист	Листов	
Провер.		Бизюк А.Н									21	28
Реценз.									УО «ВГТУ» ИСАП зр. Ит-3			
Н. Контр.												
Утверд.												

```

protected void saveToFileEnd() throws IOException{
    try(OutputStream fileStream = new FileOutputStream("temp.txt")) {
        buffer.writeTo(fileStream);
    }
}

void startMove() throws IOException, InterruptedException {
    int a = 0; int n = 0;
    Scanner sc = new Scanner(System.in);
    while (a < 1 || a > 3) {
        //System.out.println(" -----\n");
        System.out.println(" 1 - Выбрать существующую матрицу.");
        System.out.println(" 2 - Ручной ввод. ");
        System.out.println(" 3 - Открыть из файла.\n ");

        if(sc.hasNextInt()) {
            a = sc.nextInt();
        }
        else {
            System.out.println("\n Вы ввели не целое число.");
            bNaN = true;
            endMove();
        }
    }
    if (a == 1) {
        Scanner sc1 = new Scanner(System.in);
        System.out.println(" Есть 4 матрицы. Введите число от 1 до 4.\n ");
        if(sc1.hasNextInt()) {
            n = sc1.nextInt();
        }
        else {
            System.out.println("\n Вы ввели не целое число.");
            bNaN = true;
            endMove();
        }
        arrayS = createMatrix(n);
    }

    if (a == 2) {
        arrayS = createMatrix();
    }

    if (a == 3) {
        try {
            arrayS = openFile();
        } catch (IOException ex) {
            System.out.println(" Неверный файл.");
        }
    }
}

```

```

void endMove() throws IOException, InterruptedException {
    String s = "";
    System.out.println(" -----\n");
    System.out.println(" Enter - начать заново.");
    if (save && bNaN != true){
        System.out.println(" Введите S для сохранения результатов.");
    }
    System.out.println(" Введите N для выхода из программы.");
    Scanner scEnd = new Scanner(System.in);
    s = scEnd.nextLine();
    if (s.matches("n")) {
        scEnd.close();
        System.exit(0);
    }
    if (s.matches("s")) {
        try {
            saveToFile();
        } catch (FileNotFoundException ex) {
            System.out.println("\n Ошибка сохранения.");
        } catch (IOException ex) {
            System.out.println("\n Ошибка сохранения.");
        }
    }
    endMove();
}
else
    new ProcessBuilder("cmd", "/c", "cls").inheritIO().start().waitFor();
bNaN = false;
launchProject();
}

private static int currentIndex = -1;
private static Integer next(String numbers[]) {
    ++currentIndex;
    while (currentIndex < numbers.length && numbers[currentIndex].equals(""))
        ++currentIndex;
    return currentIndex < numbers.length ? Integer.parseInt(numbers[currentIndex]) : null;
}

int[][] openFile() throws FileNotFoundException, IOException, InterruptedException{
    System.out.println(" Файл должен содержать: размер матрицы, матрицу.");
    System.out.println(" Пример:");
    System.out.println(" 2 3");
    System.out.println(" 1 2 3");
    System.out.println(" 4 5 6");

    int[][] array = new int [0][0];
    JFileChooser fileopen = new JFileChooser();
    int ret = fileopen.showDialog(null, "Открыть файл");
    if (ret == JFileChooser.APPROVE_OPTION) {
        FileInputStream inFile = new FileInputStream(fileopen.getSelectedFile());
        byte[] str = new byte[inFile.available()];
        inFile.read(str);
    }
}

```

```

String text = new String(str);

try{
String[] numbers = text.split("\\D");
int i, j;
int n = next(numbers), m = next(numbers);
int matr[][] = new int[n][m];

for (i = 0; i < n; ++i)
    for (j = 0; j < m; ++j){
        matr[i][j] = next(numbers);
        array = matr;
    }
}
catch(Throwable th){
    System.out.println(" Неверное содержимое файла.");
    endMove();
}
inFile.close();
}

currentIndex = -1;
return array;
}

private static void copyFileUsingStream(File dest) throws IOException {
InputStream is = null;
OutputStream os = null;
try {
    is = new FileInputStream("temp.txt");
    os = new FileOutputStream(dest);
    byte[] buffer = new byte[1024];
    int length;
    while ((length = is.read(buffer)) > 0) {
        os.write(buffer, 0, length);
    }
} finally {
    is.close();
    os.close();
}
}

void saveToFile() throws FileNotFoundException, IOException{
FileNameExtensionFilter filter = new FileNameExtensionFilter("Text files", "txt");
JFileChooser fc = new JFileChooser();
fc.setFileFilter(filter);
if ( fc.showSaveDialog(null) == JFileChooser.APPROVE_OPTION ) {
    File file = fc.getSelectedFile(); {
        copyFileUsingStream(file);
    }
}
System.out.println("\n Готово.");
save = false;
}

```

```

int[][] createMatrix() throws IOException, InterruptedException{
    int a = 0, b = 0;
    Scanner in = new Scanner(System.in);
    while (a < 2) {
        System.out.println(" Введите количество строк матрицы (минимум 2): ");
        if(in.hasNextInt()) {
            a = in.nextInt();
        }
        else {
            System.out.println("\n Вы ввели не целое число.");
            bNaN = true;
            endMove();
        }
    }
    while (b < 2) {
        System.out.println(" Введите количество столбцов матрицы (минимум 2): ");
        if(in.hasNextInt()) {
            b = in.nextInt();
        }
        else {
            System.out.println("\n Вы ввели не целое число.");
            bNaN = true;
            endMove();
        }
    }
    PureStrategyConsole newArray = new PureStrategyConsole();
    int[][] array = newArray.createMatrix(a,b);

    for (int i = 0; i < array.length; i++) {
        for (int j = 0; j < array[i].length; j++) {
            System.out.println(" Введите элемент матрицы [" + (i+1) + "][" + (j+1) + "]:");
            if(in.hasNextInt()) {
                array[i][j] = in.nextInt();
            }
            else {
                System.out.println("\n Вы ввели не целое число.");
                bNaN = true;
                endMove();
            }
        }
    }
    System.out.println();
    return array;
}

int[][] createMatrix(int n) throws IOException {
    int[][] array1 = {{1,-3,-2},
        {0,5,4},
        {2,3,2}};
}

```

```

switch (n) {
    case 0:
        int[][] array0 = {{3,9,2},
                        {7,8,5}};
        return array0;
    case 1:
        return array1;
    case 2:
        int[][] array2 = {{3,9,2,1},
                        {7,8,5,6},
                        {4,7,3,5},
                        {5,6,1,7}};
        return array2;
    case 3:
        int[][] array3 = {{3,1,2,5},
                        {2,0,0,3},
                        {-3,-5,-5,-2},
                        {0,-2,-2,1}};
        return array3;
    case 4:
        int[][] array4 = {{3,1},
                        {1,3}};
        return array4;
    default:
        System.out.println("\n Выбрана матрица 1 [по умолчанию].");
        return array1;
}
}

int[][] createMatrix(int width, int height){
    int[][] array = new int [width][height];

    return array;
}

void displayMatrix(int[][] target) throws IOException{
    int[][] array = target;
    System.out.println(" -----");
    System.out.println("\n Платежная матрица: ");
    System.out.println(" " + array.length + " x " + array[0].length);
    for (int i = 0; i < array.length; i++) {
        for (int j = 0; j < array[i].length; j++) {
            System.out.print(" "+array[i][j] + "\t");
        }
        System.out.println();
    }
    System.out.println();
}

int culumnOps(int[][] target) throws IOException{
    int iMaxMin = -1;
    int jMaxMin = -1;
    int[][] array = target;

```



```

System.out.println(" MAX по столбцам: ");
for(int i = 0; i < array[0].length; i++){
    int max = array[0][i];
    int jMin = 0;
    for(int j = 0; j < array.length; j++)
    {
        if(array[j][i] > max)
        {
            max = array[j][i];
            jMin = j;
        }
    }
    if (i == 0 || max < minmax){
        minmax = max;
        iMaxMin = i+1;
        jMaxMin = jMin+1;
    }
    System.out.print(" " + max + " ");

}
x1 = jMaxMin;
y1 = iMaxMin;
System.out.println();
System.out.println(" Верхняя чистая цена игры (MinMax) = " + minmax + "("+x1+";" +y1+").");
System.out.println();
return minmax;
}

int rowOps(int[][] target) throws IOException{
    int iMaxMin = -1;
    int jMaxMin = -1;
    int[][] array = target;

    System.out.println(" MIN по строкам: ");
    for(int i = 0; i < array.length; i++){
        int min = array[i][0];
        int jMin = 0;
        for(int j = 0; j < array[0].length; j++)
        {
            if(array[i][j] < min)
            {
                min=array[i][j];
                jMin = j;
            }
        }
        if (i == 0 || min > maxmin)
        {
            maxmin = min;
            iMaxMin = i+1;
            jMaxMin = jMin+1;
        }
    }
}

```

```

        System.out.println(" " + min + " ");
    }
    x2 = iMaxMin;
    y2 = jMaxMin;
    System.out.println("\n Нижняя чистая цена игры (MaxMin) = " + maxmin + "("+x2 +" ; "+y2+").");
    System.out.println();
    return maxmin;
}

void clearPrice() throws IOException {
    save = true;
    if (maxmin == minmax)
        System.out.println(" Чистая цена игры: MaxMin = MinMax = " + maxmin + ".");
    else
        System.out.println(" Эта задача решения в чистых стратегиях не имеет, так как MaxMin не
равно MinMax.");
}

void saddlePoint() throws IOException {
    if (maxmin == minmax) {
        if (x1 == x2 & y1 == y2){
            System.out.println(" Седловая точка: (Iо = " + x1 + " ; Jo = " + y2 + ").");
            System.out.println(" Оптимальная стратегия игрока А = " + x1 + ".");
            System.out.println(" Оптимальная стратегия игрока Б = " + y2 + ".");
        }
        else
            System.out.println(" Эта задача решения в чистых стратегиях не имеет.");
    }
}

void launchProject() throws IOException, InterruptedException{
    startMove();
    saveToFileBeg();
    displayMatrix(arrayS);
    culumnOps(arrayS);
    rowOps(arrayS);
    clearPrice();
    saddlePoint();
    saveToFileEnd();
    endMove();
}

public static void main(String[] args) throws IOException, InterruptedException {
    //System.setProperty("console.encoding","Cp866");
    PureStrategyConsole newArray = new PureStrategyConsole();
    System.out.println("\n -----");
    System.out.println(" Решение матричных игр в чистых стратегиях. ");
    System.out.println(" ----- \n");
    newArray.launchProject();
}
}

```