Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

по курсу: «Программирование на языке Java»

на тему: «ИП "Пять в ряд"»

Выполнил:

студент группы 21ВВП1:

Пивкин Д.А.

Принял:

Юрова О.В.

Пенза 2024

**Содержание**

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

Постановка задачи...................................................................................................4

Выбор решения........................................................................................................6

Описание программы..............................................................................................8

Описание способа организации пользовательского интерфейса........................9

Описание результатов работы программы..........................................................12

Заключение.............................................................................................................13

Список используемой литературы.......................................................................14

Приложение А. Листинг программы....................................................................15

Приложение Б. UML-диаграммы.........................................................................21

Приложение C. Результаты работы программы.................................................26

**Введение**

Компьютерные игры являются одним из самых популярных видов развлечений в современном мире. С развитием технологий, игры стали доступными на различных платформах, включая персональные компьютеры, игровые консоли и мобильные устройства. Одной из интересных и увлекательных логических игр является "Пять в ряд" (Gomoku), которая пользуется популярностью благодаря своей простоте и одновременно стратегической глубине. Цель игры заключается в том, чтобы выстроить пять своих фигур в ряд по вертикали, горизонтали или диагонали на игровом поле.

Целью данной курсовой работы является разработка компьютерной версии игры "Пять в ряд" на языке программирования Java. Выбор Java в качестве языка разработки обусловлен его широким применением в создании кроссплатформенных приложений, мощными встроенными библиотеками для создания графического интерфейса и высокой производительностью.

Java, как язык программирования, имеет долгую историю и широкое применение в различных областях разработки программного обеспечения, включая разработку компьютерных игр. Являясь объектно-ориентированным языком, Java предоставляет мощные инструменты для создания как простых, так и сложных игр с богатым графическим интерфейсом.

В условиях современного цифрового общества, где важное место занимают информационные технологии, создание качественного программного обеспечения, включая игры, является важной и актуальной задачей. Проектирование и реализация игры "Пять в ряд" способствует развитию навыков программирования, алгоритмического мышления и работы с графическим интерфейсом.

**Постановка задачи**

Целью данной курсовой работы является разработка компьютерной версии игры "Пять в ряд" на языке программирования Java. Основной задачей является создание полноценного игрового приложения, которое включает в себя реализацию всех необходимых элементов для комфортного и увлекательного игрового процесса.

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. **Анализ игры и существующих решений**:
   * Изучить правила и основные механики игры "Пять в ряд".
   * Проанализировать существующие компьютерные версии игры, выявить их сильные и слабые стороны.
2. **Проектирование и разработка игровой логики**:
   * Разработать алгоритмы для обработки ходов игроков и определения победителя.
   * Обеспечить поддержку различных уровней сложности при игре против компьютера (искусственный интеллект).
3. **Создание графического интерфейса пользователя (GUI)**:
   * Спроектировать удобный и интуитивно понятный интерфейс для игры.
   * Реализовать интерфейс с использованием библиотеки Swing, обеспечив возможность взаимодействия пользователя с игровым полем и элементами управления.
4. **Реализация игровой механики**:
   * Обеспечить корректное отображение игрового поля и игровых фигур.
   * Реализовать механику постановки фигур на игровое поле, а также проверку условий победы и ничьи.
   * Обеспечить возможность начала новой игры и перезапуска текущей игры.
5. **Тестирование и отладка приложения**:
   * Провести тестирование всех компонентов игры, выявить и исправить ошибки.
   * Обеспечить стабильную работу приложения на различных операционных системах.

Реализация всех перечисленных задач позволит создать качественное и функциональное игровое приложение "Пять в ряд", которое будет отличаться удобным интерфейсом, корректной и стабильной работой, а также интересной и увлекательной игровой механикой.

**Выбор решения**

Для разработки серверной и клиентской частей приложения была выбрана платформа Windows 10, которая является удобным, простым и быстрым решением для развертывания оперативного серверного приложения в локальной сети, не требующего серьезных знаний операционной системы и сетевых подключений. В качестве среды для разработки программного обеспечения была выбрана IntelliJ IDEA, так как она обладает широчайшим набором крайне полезных функций, проста в использовании, быстро собирает проекты, имеет удобный функционал, множество систем сборки и превосходит свои аналоги от других компаний.

IntelliJ IDEA предоставляет возможность создания точек останова, отслеживания памяти во время отладки программ, одновременного запуска нескольких экземпляров приложения, а также имеет приятный и понятный вывод сообщений от дебаггера и компоновщика. Среда разработки оснащена мощными средствами рефакторинга и подсказок кода, которые корректно отмечают все неточности в коде и грамотно их исправляют при написании.

Для разработки клиентской части приложения также была выбрана платформа Windows 10. Использование одной и той же платформы для обеих частей приложения упрощает разработку и тестирование. IntelliJ IDEA предоставляет все необходимые инструменты для разработки и отладки как серверной, так и клиентской частей приложения. Благодаря своим мощным функциям и интуитивно понятному интерфейсу, IntelliJ IDEA значительно упрощает процесс разработки, позволяя сосредоточиться на реализации функциональности приложения.

Одним из значимых преимуществ использования IntelliJ IDEA является возможность отладки приложения в реальном времени. Среда разработки поддерживает установку точек останова и отслеживание состояния памяти, что позволяет эффективно находить и устранять ошибки. Кроме того, IntelliJ IDEA предоставляет мощные средства для анализа и оптимизации кода, что позволяет повысить производительность и стабильность приложения.

Несмотря на все преимущества, единственным минусом IntelliJ IDEA можно назвать отсутствие перевода на другие языки, кроме английского. Однако, это позволяет улучшить или укрепить технический английский, что является крайне полезным навыком для программистов.

Таким образом, использование платформы Windows 10 и среды разработки IntelliJ IDEA для создания игры "Пять в ряд" обеспечивает удобство, эффективность и высокое качество разработки как серверной, так и клиентской частей приложения.

**Описание программы**

Серверная часть состоит из одного приложения FiveInRowServer.java, который запускается на операционной системе Windows и ожидает любые подключения на хост компьютера по порту 80. При подключении первого клиента сервер присваивает этому игроку символ Х и цвет, после чего отправляет сообщение ожидания другого клиента. При подключении второго клиента, сервер присваивает ему символ ноль и сигнализирует первому игроку о начале хода. Далее сервер обрабатывает ходы игроков, обеспечивает рассылку обоим клиентам данных и вычисляет победу игроков. В случае победы одного из игроков или ничью, рассылает соответствующее сообщение всем игрокам и завершает свою работу.

Клиентская программа состоит из одного приложения FiveInRowClient.java, который аналогично серверу запускается на операционной системе Windows. Далее эта программа взаимодействует с пользователем через графический интерфейс и отсылает игровые события на сервер, ожидая от него верификации хода, победы и иных сообщений.

**Описание способа организации пользовательского интерфейса**

Интерфейс сервера представлен консолью с текстом, так как не требует вмешательства пользователя для его работы. Для работы серверу достаточно запустить исполняемый файл и выдать ему разрешение на доступ к локальной сети, после чего можно пользоваться клиентскими приложениями, подключенными к сети сервера.

Интерфейс пользовательского интерфейса клиентского приложения реализовано оконным графическим режимом. На рисунке ниже показано клиентское приложение данного курсового проекта.

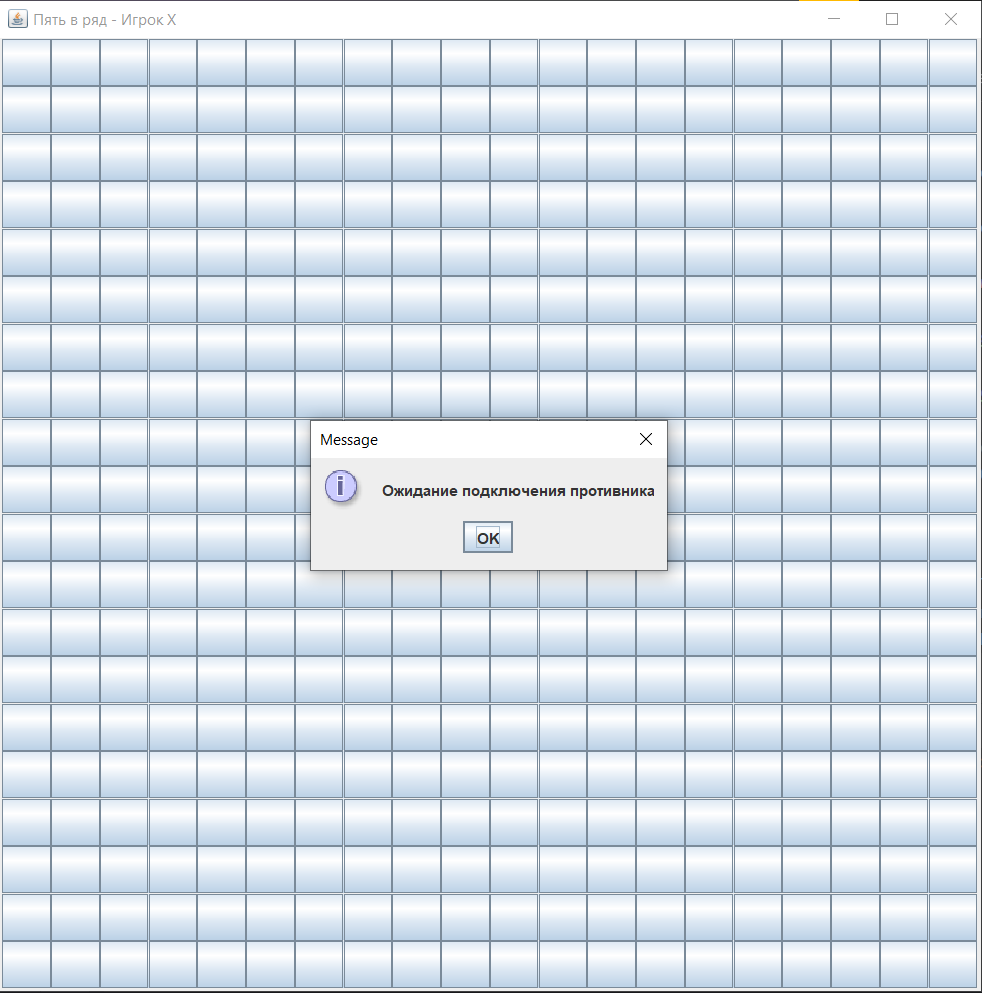
****

Рисунок 1 - Клиентское приложение

При запуске клиентского приложения программа отрисовывает окно с названием приложения – «Пять в ряд – Игрок Х», где так же можно узнать, за какого игрока (Х или О) играет пользователь. Сразу при появлении выходит окно с уведомлением о том, что необходимо дождаться подключения второго игрока, после чего можно нажать кнопочку «ОК» и начать игру.

Во время игры ходы игроков помечаются соответствующим цветом и символом (для Х – красный, для О – зеленый) на поле внутри окна, состоящего из кнопок. На рисунке ниже представлен снимок поля с зарегистрированными ходами обоих игроков.

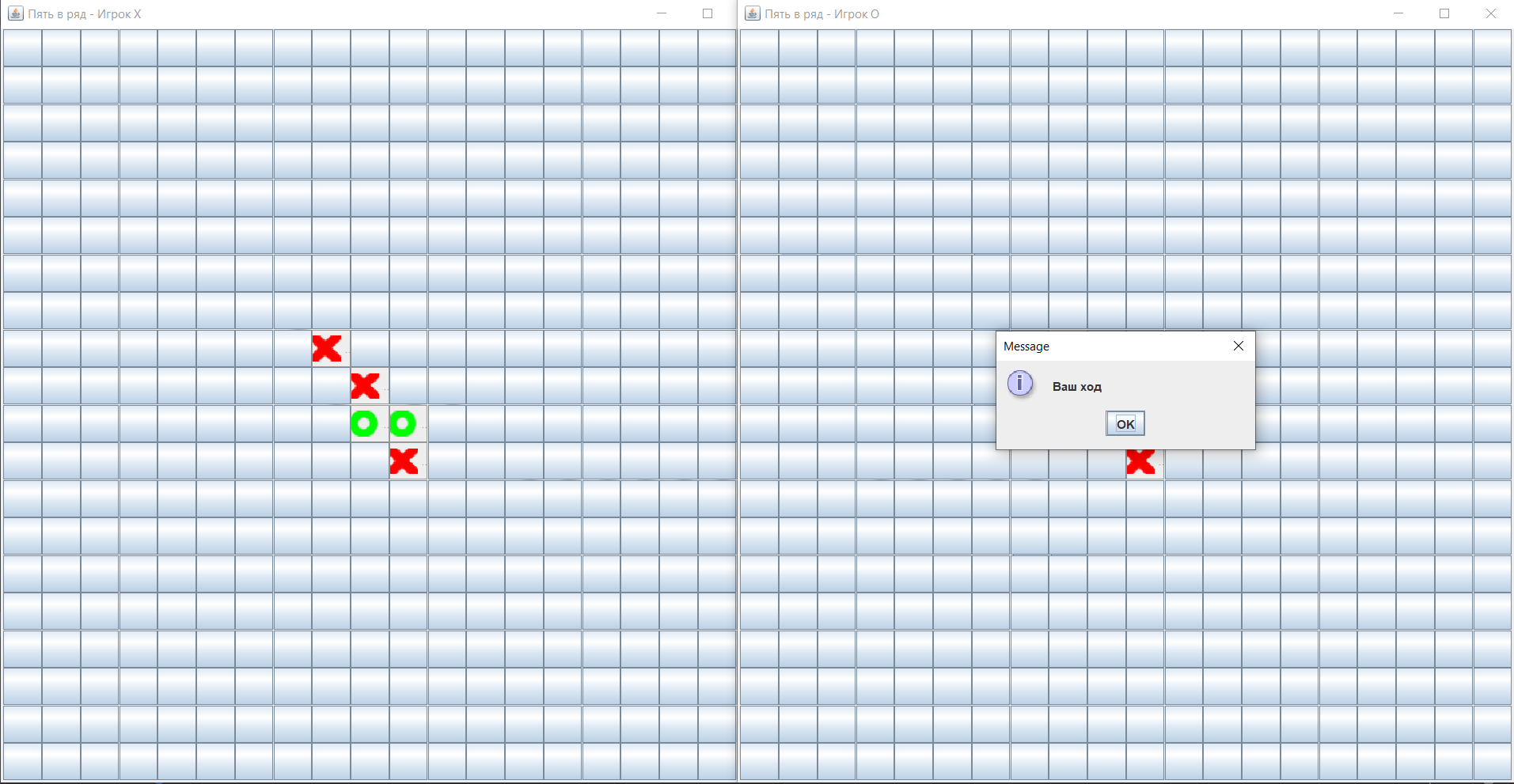


Рисунок 2 - Снимок регистрации ходов

Так же на приведенном выше снимке можно заметить окно, уведомляющее пользователя о том, что в данный такт игры его ход.

Пока другой игрок совершаем ход, первый игрок не может никак сходить. За эту игровую возможность отвечает сервер, регулируя и регламентируя действия пользователей.

На следующим рисунке, приведенном ниже, представлен снимок экранов проигрыша и выигрыша двух игроков.

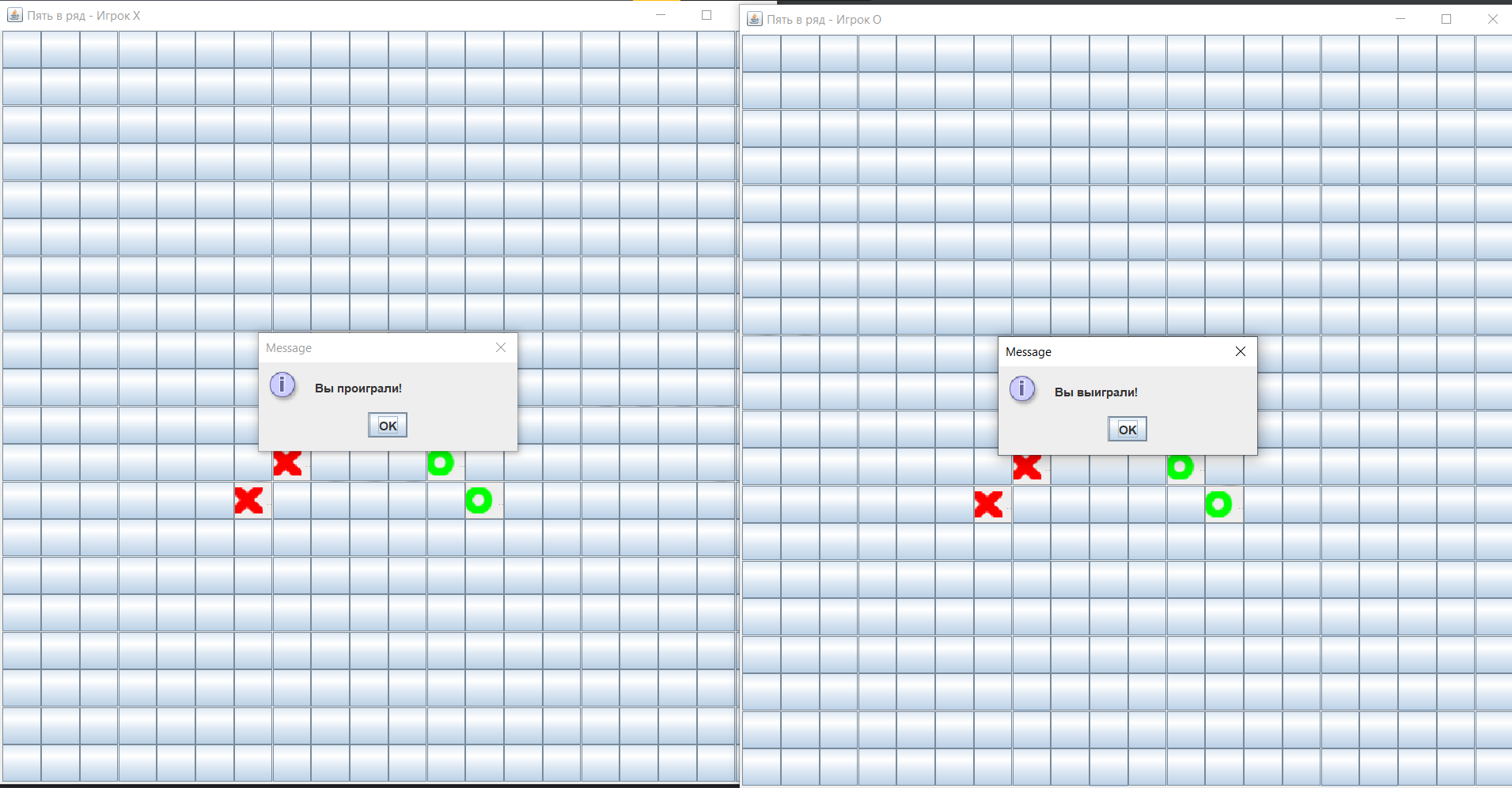


Рисунок 3 - Экраны проигрыша и выигрыша

**Описание результатов работы программы**

Программа была успешно протестирована в различных сценариях работы в локальной сети. Тестирование включало проверку подключения клиента к серверу, отправку и получение сообщений, обратку действий пользователей, вычисление побед и проигрышей, обработку ошибок соединения. Во время тестирования программа продемонстрировала следующие результаты:

1. **Подключение клиентов**: клиенты успешно подключались к серверу при запуске программы. При подключении клиент получал сообщение о том, какой символ для игры ему присвоен и чей сейчас ход.
2. **Обмен сообщениями**: сообщения, отправленные одним клиентом, корректно доставлялись серверу, который успешно обрабатывал и пересылал команду второму клиенту. Сообщения отображались на сервере и клиентах без задержек, обеспечивая непрерывный обмен информацией.
3. **Обработка выигрыша и проигрыша**: ходы успешно просчитывались на сервере и в случае победы одного из игрока успешно отображалась информация о статусе побед на обоих клиентах. Так же безошибочно работала обработка ничьей между игроками.
4. **Обработка ошибок**: в случаях некорректного ввода данных для подключения или проблем с сетью пользователю выводилось соответствующее уведомление. Программа корректно обрабатывала разрывы соединений, предотвращая зависание или некорректное поведение.

**Заключение**

В результате выполнения курсового проекта была разработана и протестирована многомодульная система игры "Пять в ряд", состоящая из серверного приложения и клиентского приложения, оба на платформе Windows с использованием среды разработки IntelliJ IDEA.

Ключевые достижения проекта включают:

* Реализация основных функций игры: подключение клиентов к серверу, обработка ходов игроков, определение победителя, сохранение истории игр.
* Применение современных технологий Java, таких как Java Collections Framework, механизм обработки исключений, Java Stream API, многопоточность и сетевое взаимодействие.
* Тестирование и отладка программы в различных условиях, что подтвердило её надежность и корректное функционирование.

Проект демонстрирует возможность создания эффективного и интуитивно понятного интерфейса для локальной многопользовательской игры, что может быть полезно для дальнейшего развития и внедрения в учебные и корпоративные сети. В будущем планируется расширение функциональности программы, включая поддержку различных режимов игры и улучшение интерфейса пользователя.

# 

# **Список литературы**

1. IntelliJ IDEA. Профессиональное программирование на Java / С. Давыдов, А. Ефимов. - М.: БХВ-Петербург, 2005. - 800 c. (Дата обращения: 15.06.24) [IntelliJ IDEA. Профессиональное программирование на Java, А. А. Ефимов – скачать pdf на ЛитРес (litres.ru)](https://www.litres.ru/book/aleksey-alekseevich/intellij-idea-professionalnoe-programmirovanie-na-jav-647725/)
2. Язык Java. Курс программирования / Уолтер Савитч. - М.: Вильямс, 2015. - 928 c. (Дата обращения: 15.06.24) [Язык Java. Курс программирования — Савитч, Уолтер | Livelib](https://www.livelib.ru/book/1000444090-yazyk-java-kurs-programmirovaniya-savitch?ysclid=lx2w3nhkfi251021343)
3. "Swing: Эффектные пользовательские интерфейсы", Иван Портянкин — Издательство «Лори», 2011 — 591 с. (Дата обращения: 15.06.24) (Дата обращения: 15.06.24) [Swing. Эффектные пользовательские интерфейсы — Иван Портянкин | Livelib](https://www.livelib.ru/book/1000812065-swing-effektnye-polzovatelskie-interfejsy-ivan-portyankin?ysclid=lx2w472k6v349211673)
4. Форум программистов и сисадминов Киберфорум. <http://www.cyberforum.ru/>.(Дата обращения: 15.06.24)
5. Stack Overflow <https://stackoverflow.com/>.(Дата обращения: 15.06.24)

**Приложение А. Листинг программы**

**FiveInRowServer.java**

import java.io.\*;

import java.net.\*;

import java.util.Arrays;

public class FiveInRowServer {

private static final int PORT = 12346;

private static final int SIZE = 20;

private static final int WINNING\_STREAK = 5;

private char[][] board = new char[SIZE][SIZE];

private PlayerHandler[] players = new PlayerHandler[2];

private int currentPlayer = 0;

public static void main(String[] args) {

new FiveInRowServer().start();

}

public FiveInRowServer() {

for (char[] row : board) {

Arrays.fill(row, ' ');

}

}

public void start() {

try (ServerSocket serverSocket = new ServerSocket(PORT)) {

System.out.println("Сервер запущен...");

while (true) {

if (players[0] == null) {

players[0] = new PlayerHandler(serverSocket.accept(), 'X');

new Thread(players[0]).start();

} else if (players[1] == null) {

players[1] = new PlayerHandler(serverSocket.accept(), 'O');

new Thread(players[1]).start();

}

}

} catch (IOException e) {

e.printStackTrace();

}

}

private synchronized boolean makeMove(int x, int y, char symbol) {

if (board[x][y] == ' ') {

board[x][y] = symbol;

return true;

}

return false;

}

private synchronized boolean checkWin(int x, int y, char symbol) {

return checkDirection(x, y, symbol, 1, 0) || // Горизонтально

checkDirection(x, y, symbol, 0, 1) || // Вертикально

checkDirection(x, y, symbol, 1, 1) || // Диагональ \

checkDirection(x, y, symbol, 1, -1); // Диагональ /

}

private boolean checkDirection(int x, int y, char symbol, int dx, int dy) {

int count = 1;

for (int i = 1; i < WINNING\_STREAK; i++) {

if (inBounds(x + i \* dx, y + i \* dy) && board[x + i \* dx][y + i \* dy] == symbol) count++;

else break;

}

for (int i = 1; i < WINNING\_STREAK; i++) {

if (inBounds(x - i \* dx, y - i \* dy) && board[x - i \* dx][y - i \* dy] == symbol) count++;

else break;

}

return count >= WINNING\_STREAK;

}

private boolean inBounds(int x, int y) {

return x >= 0 && x < SIZE && y >= 0 && y < SIZE;

}

private class PlayerHandler implements Runnable {

private Socket socket;

private BufferedReader input;

private PrintWriter output;

private char symbol;

public PlayerHandler(Socket socket, char symbol) {

this.socket = socket;

this.symbol = symbol;

}

@Override

public void run() {

try {

input = new BufferedReader(new InputStreamReader(socket.getInputStream()));

output = new PrintWriter(socket.getOutputStream(), true);

output.println("WELCOME " + symbol);

if (symbol == 'X') {

output.println("MESSAGE Ожидание подключения противника");

} else {

players[0].output.println("MESSAGE Ваш ход");

}

while (true) {

String command = input.readLine();

if (command == null) {

break;

}

if (command.startsWith("MOVE")) {

String[] parts = command.split(" ");

if (parts.length == 3) {

try {

int x = Integer.parseInt(parts[1].trim());

int y = Integer.parseInt(parts[2].trim());

if ((currentPlayer == 0 && symbol == 'X') || (currentPlayer == 1 && symbol == 'O')) {

if (makeMove(x, y, symbol)) {

players[currentPlayer].output.println("YOUR\_MOVE " + x + " " + y);

players[1 - currentPlayer].output.println("OPPONENT\_MOVED " + x + " " + y);

if (checkWin(x, y, symbol)) {

output.println("VICTORY");

players[1 - currentPlayer].output.println("DEFEAT");

break;

} else if (boardFull()) {

output.println("DRAW");

players[1 - currentPlayer].output.println("DRAW");

break;

}

currentPlayer = 1 - currentPlayer;

players[currentPlayer].output.println("MESSAGE Ваш ход");

} else {

output.println("MESSAGE Неверный ход");

}

}

} catch (NumberFormatException e) {

output.println("MESSAGE Неверный формат команды");

}

} else {

output.println("MESSAGE Неверный формат команды");

}

}

}

} catch (IOException e) {

e.printStackTrace();

} finally {

try {

socket.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

private boolean boardFull() {

for (int x = 0; x < SIZE; x++) {

for (int y = 0; y < SIZE; y++) {

if (board[x][y] == ' ') {

return false;

}

}

}

return true;

}

}

}

**FiveInRowClient.java**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.\*;

import java.net.\*;

public class FiveInRowClient {

private static final int SIZE = 20;

private JFrame frame = new JFrame("Пять в ряд");

private JButton[][] buttons = new JButton[SIZE][SIZE];

private PrintWriter output;

private BufferedReader input;

static Font customFont = null;

ImageIcon iconKrest;

ImageIcon iconKrug;

public static void main(String[] args) throws IOException, FontFormatException {

new FiveInRowClient().start();

}

public FiveInRowClient() {

// Загружаем иконку

URL imgURL = FiveInRowClient.class.getResource("/resources/krest.png");

if (imgURL != null) {

iconKrest = new ImageIcon(imgURL);

System.out.println("Файл открыт: resources/krestik.png");

} else {

System.err.println("Файл не найден: resources/krestik.png");

iconKrest = null; // Или установите иконку по умолчанию

}

// Загружаем иконку

imgURL = FiveInRowClient.class.getResource("/resources/krug.png");

if (imgURL != null) {

iconKrug = new ImageIcon(imgURL);

System.out.println("Файл открыт: resources/krug.png");

} else {

System.err.println("Файл не найден: resources/krug.png");

iconKrest = null; // Или установите иконку по умолчанию

}

}

public void start() {

frame.setLayout(new GridLayout(SIZE, SIZE));

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

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

final int x = i;

final int y = j;

buttons[i][j] = new JButton(" ");

if (customFont != null) {

buttons[i][j].setFont(customFont);

} else {

buttons[i][j].setFont(new Font("Arial", Font.PLAIN, 10));

}

buttons[i][j].setFocusPainted(false);

buttons[i][j].addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

output.println("MOVE " + x + " " + y);

}

});

frame.add(buttons[i][j]);

}

}

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(800, 800);

frame.setVisible(true);

try {

Socket socket = new Socket("localhost", 12346);

input = new BufferedReader(new InputStreamReader(socket.getInputStream()));

output = new PrintWriter(socket.getOutputStream(), true);

while (true) {

String response = input.readLine();

if (response == null) {

// Соединение с сервером закрыто

JOptionPane.showMessageDialog(frame, "Соединение с сервером потеряно.");

break;

}

if (response.startsWith("WELCOME")) {

char mark = response.charAt(8);

frame.setTitle("Пять в ряд - Игрок " + mark);

} else if (response.startsWith("MESSAGE")) {

JOptionPane.showMessageDialog(frame, response.substring(8));

} else if (response.startsWith("OPPONENT\_MOVED")) {

String[] parts = response.split(" ");

if (parts.length == 3) {

int x = Integer.parseInt(parts[1].trim());

int y = Integer.parseInt(parts[2].trim());

buttons[x][y].setIcon(currentMark() == 'O' ? iconKrest : iconKrug);

buttons[x][y].setDisabledIcon(currentMark() == 'O' ? iconKrest : iconKrug);

buttons[x][y].setEnabled(false);

}

} else if (response.startsWith("YOUR\_MOVE")) {

String[] parts = response.split(" ");

if (parts.length == 3) {

int x = Integer.parseInt(parts[1].trim());

int y = Integer.parseInt(parts[2].trim());

buttons[x][y].setIcon(currentMark() == 'X' ? iconKrest : iconKrug);

buttons[x][y].setDisabledIcon(currentMark() == 'X' ? iconKrest : iconKrug);

buttons[x][y].setEnabled(false);

}

} else if (response.startsWith("VICTORY")) {

JOptionPane.showMessageDialog(frame, "Вы выиграли!");

break;

} else if (response.startsWith("DEFEAT")) {

JOptionPane.showMessageDialog(frame, "Вы проиграли!");

break;

} else if (response.startsWith("DRAW")) {

JOptionPane.showMessageDialog(frame, "Ничья!");

break;

}

}

} catch (IOException e) {

e.printStackTrace();

} finally {

frame.setVisible(false);

frame.dispose();

}

}

private char currentMark() {

return frame.getTitle().contains("X") ? 'X' : 'O';

}

}

**Приложение Б. UML-диаграммы**

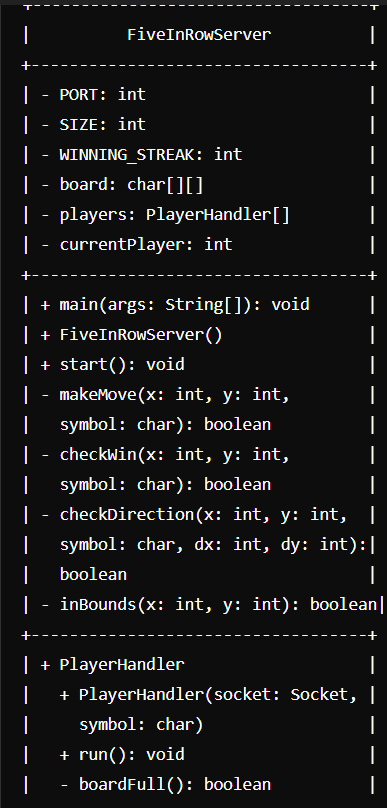
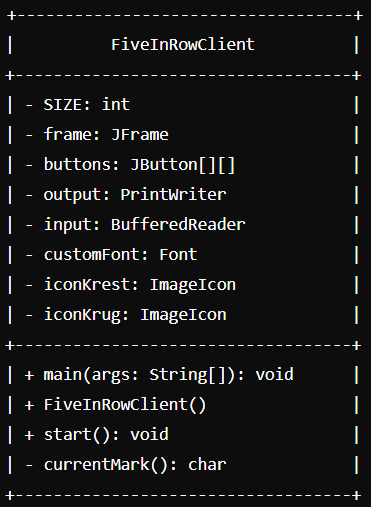


Рисунок 4 - UML-диаграмма классов

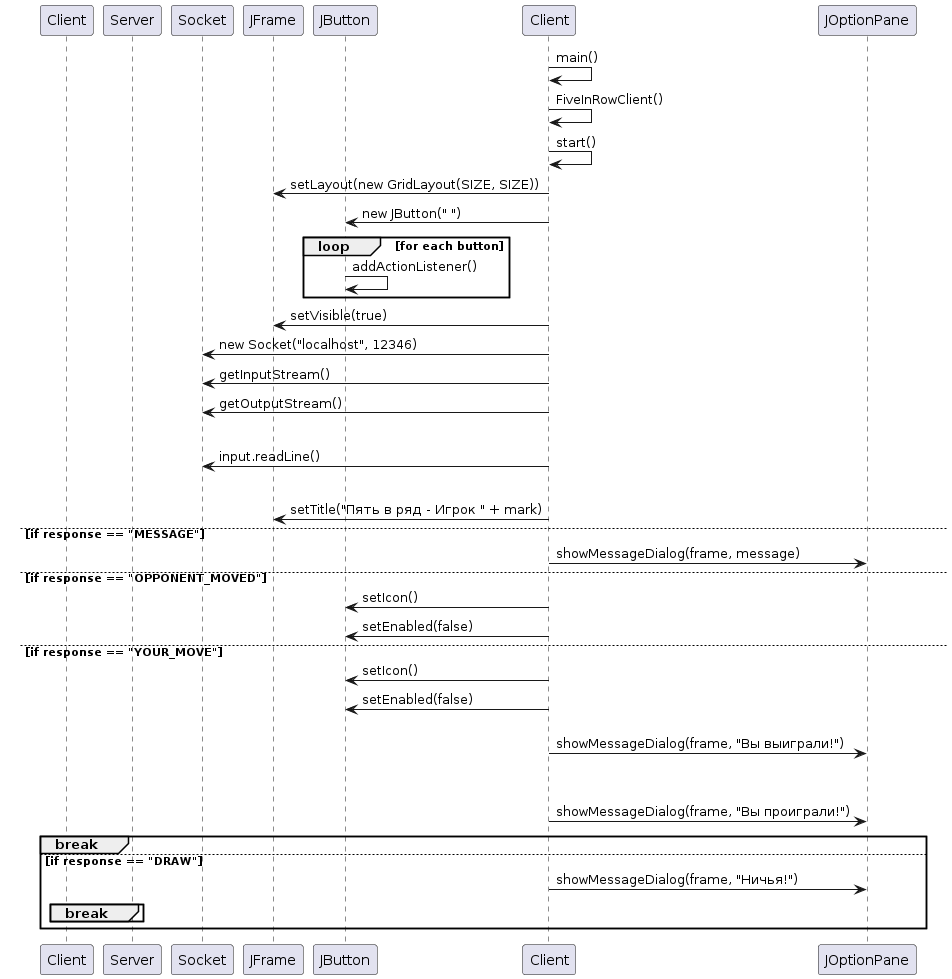


Рисунок 5 - UML-диаграмма последовательности клиента

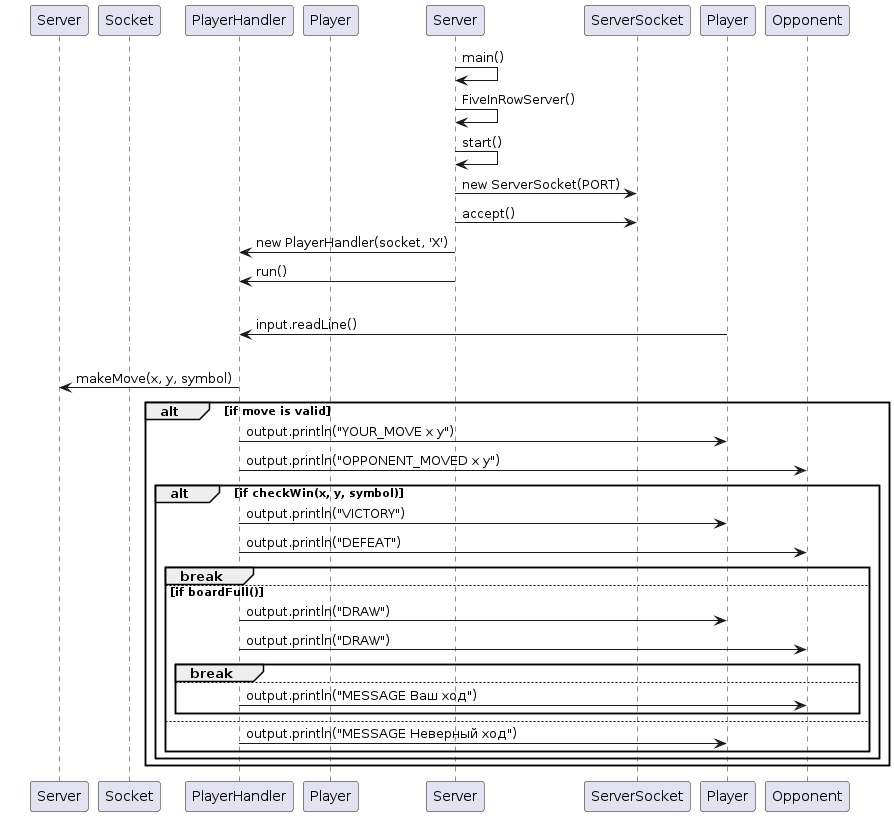


Рисунок 6 - UML-диаграмма последовательности сервера

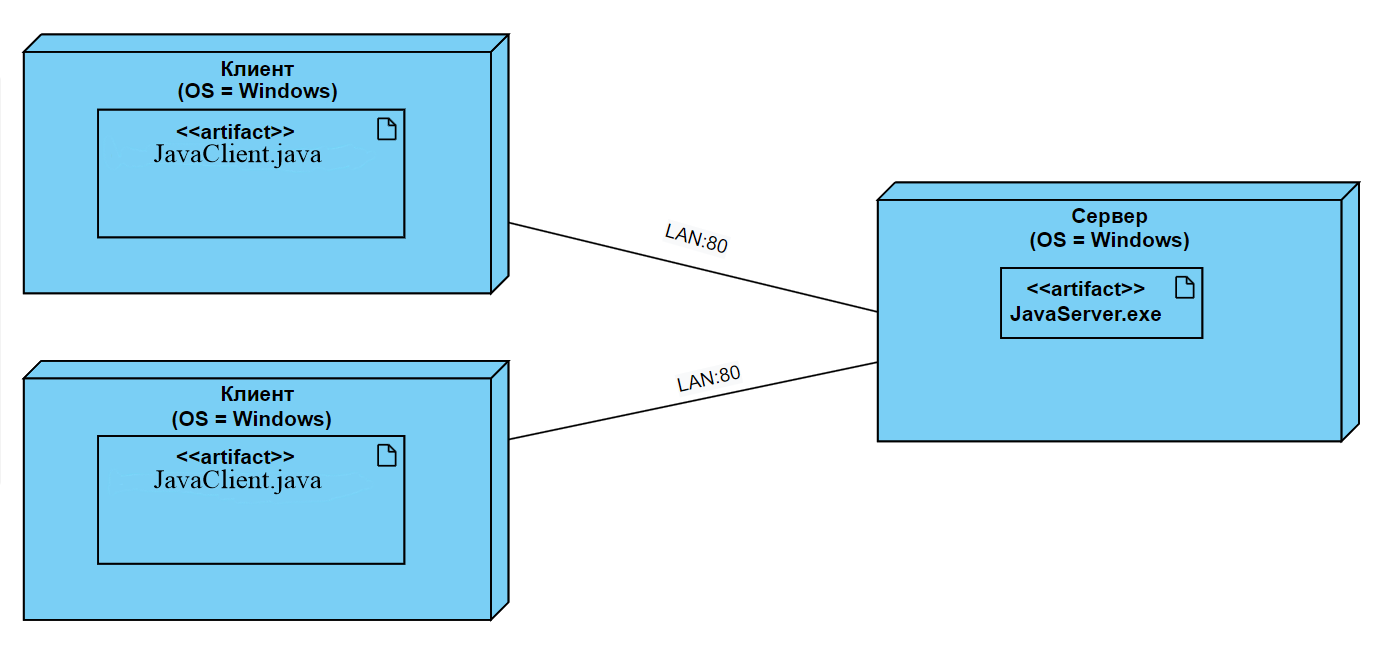


Рисунок 7 - UML-диаграмма развертывания

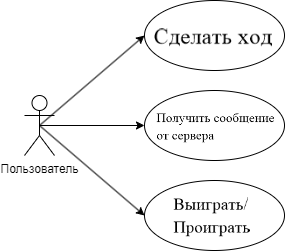


Рисунок 8 - UML-диаграмма поведения

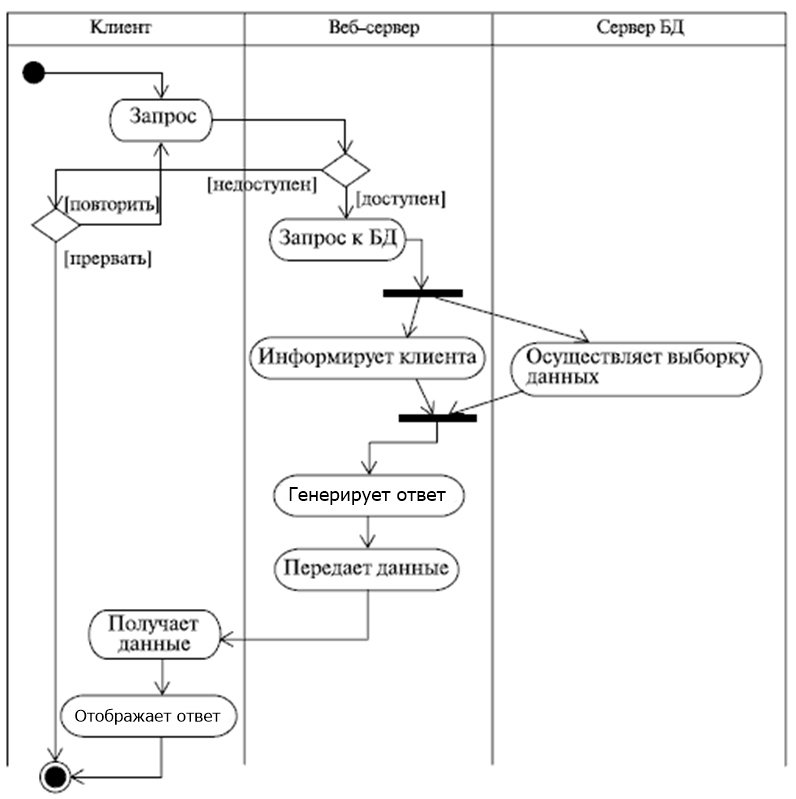


Рисунок 9 - UML-диаграмма деятельности

**Приложение С. Результаты работы программы**

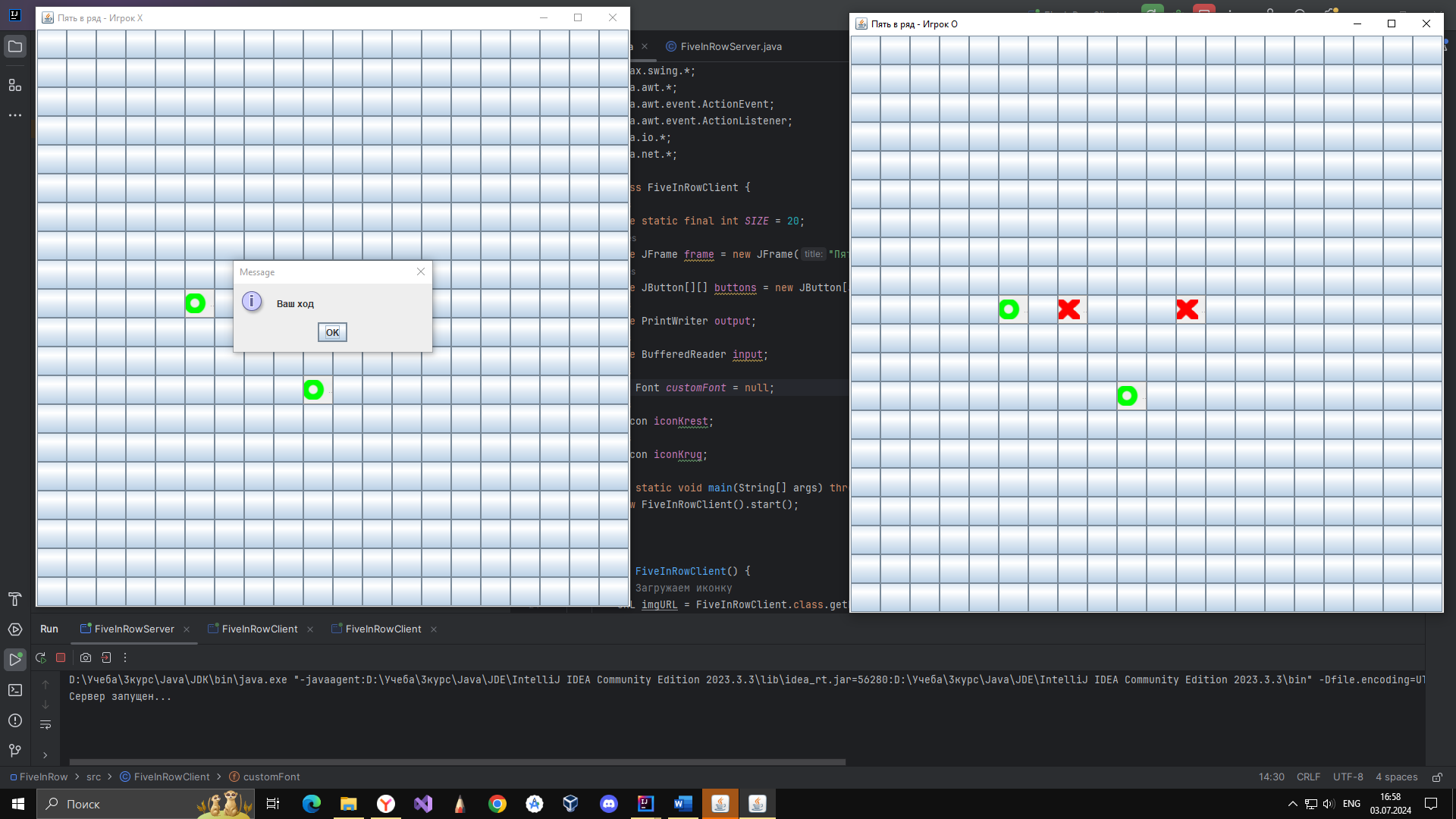


Рисунок 10 - Результат работы программы - Сервер