Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**КУРСОВОЙ ПРОЕКТ**

**Дисциплина:** Алгоритмы и структуры данных

**Тема:** Разработка GUI приложения Пятнашки и его решателя на языке Java

Выполнил

студент гр. 3530901/80003\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Шрамков М.А.

(подпись)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Глухих М.И.

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

Санкт-Петербург   
2019

**Оглавление**

[**ТЕХНИЧЕСКОЕ ЗАДАНИЕ** 3](#_Toc26912944)

[**МЕТОД РЕШЕНИЯ** 4](#_Toc26912945)

[**ЛИСТИНГ ПРОГРАММЫ** 6](#_Toc26912946)

[**Main.java** 6](#_Toc26912947)

[**MainWindow.java** 7](#_Toc26912948)

[**mainWindow.fxml** 15](#_Toc26912949)

[**UserPositionWindow.java** 17](#_Toc26912950)

[**userPosition.fxml** 22](#_Toc26912951)

[**HelpWindow.java** 23](#_Toc26912952)

[**helpWindow.fxml** 24](#_Toc26912953)

[**Field.java** 25](#_Toc26912967)

[**Game.java** 28](#_Toc26912968)

[**Solver.java** 31](#_Toc26912969)

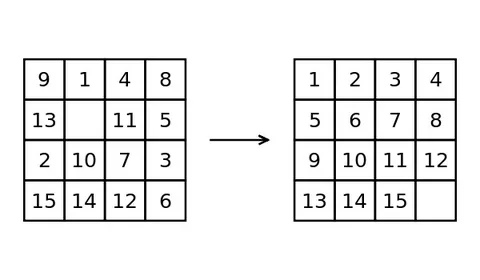
[**ТЕСТЫ** 40](#_Toc26912970)

[**СКРИНШОТЫ ПРОГРАММЫ** 46](#_Toc26912971)

# **ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

Создать игру Пятнашки, а также предоставить её решение из любой заданной позиции.

Пятнашки – головоломка, в которой игроку доступно поле размером 4x4, состоящее из 16 клеток. Все клетки кроме одной заняты костяшками с номерами от 1 до 15, которые перемешаны между собой. Цель игры - упорядочить костяшки по порядку, используя свободное поле.



GitHub репозиторий проекта: [https://github.com/k1unk/15gameAndSolver](https://github.com/killawetz/GraphSolverApp)

# **МЕТОД РЕШЕНИЯ**

В проекте использована концепция MVC (Model-View-Controller) для отделения логики от визуализации. Код разделен на два пакета классов: logic, view, а также класс Main.java

Класс Main.java – запускает главное окно нашего приложения.

Содержимое пакета view:

* MainWindow.java и mainWindow.fxml – отвечают за графическое представление главного окна приложения (меню и сама игра).
* UserPosition.java и userPosition.fxml – отвечают за графическое представление окна, которое предназначено для ввода позиции игры пользователем.
* HelpWindow.java и helpWindow.fxml – отвечают за графическое представление окна, которое предназначено для отображения правил игры.

Cодержимое пакета logic:

* Field – класс поля игры.
* Game – класс, который отвечает за сам процесс игры.
* Solver – алгоритм решателя игры.

Также имеются файл userPosition.txt, используемый классом UserPosition.java, и изображения, помещённые в пакет images.

Более подробная работа программы:

В главном меню пользователь с помощью меню может запустить игру с рандомным положением костяшек, открыть окно для ввода своего положения костяшек, запустить игру со своим положением костяшек, посмотреть правила игры, и запустить решателя игры.

Чтобы запустить игру с рандомным положением костяшек – нужно нажать на StartRandomGame в меню. При создании учитывается невозможность победы из половины стартовых позиций (метод checkWinnable() класса Field).

Для ввода своего положения костяшек – нужно ввести значения каждой из клетки в соответствующие текстовые поля. Здесь также учитывается невозможность победы из половины стартовых позиций, что не позволяет пользователю задать неправильное положение.

Можно играть самому, используя клавиши «вверх», «вниз», «вправо», «влево», или же запустить решателя.

Решатель реализован на основе алгоритма А\* (поиск в ширину с использованием функции на основе количества ходов, которые привели в данное положение и эвристической оценки положения (сумма расстояний между положением каждой из костяшек и её финального положения)).

Решателя можно запустить с интервалом 0.1, 0.2, 0.4, 1 секунды (остановить можно нажатием пробела), или можно нажатием на кнопку меню next заставить его сделать только один ход, который улучшит нашу позицию.

# **ЛИСТИНГ ПРОГРАММЫ**

## **Main.java**

package sample;

import javafx.application.Application;

import javafx.fxml.FXMLLoader;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.stage.Stage;

public class Main extends Application {

@Override

public void start(Stage primaryStage) throws Exception{

Parent root = FXMLLoader.load(getClass().getResource("view/mainWindow.fxml"));

primaryStage.setTitle("15");

primaryStage.setScene(new Scene(root, 840, 940));

primaryStage.show();

}

public static void main(String[] args) {

launch(args);

}

}

## **MainWindow.java**

package sample.view;

import javafx.animation.KeyFrame;

import javafx.animation.Timeline;

import javafx.fxml.FXML;

import javafx.fxml.FXMLLoader;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.image.Image;

import javafx.scene.image.ImageView;

import javafx.scene.text.Text;

import javafx.stage.Stage;

import javafx.util.Duration;

import sample.logic.Game;

import sample.logic.Solver;

import java.io.FileReader;

import java.io.IOException;

import java.util.Scanner;

public class MainWindow {

private Image image1 = new Image("sample/images/num1.jpg");

private Image image2 = new Image("sample/images/num2.jpg");

private Image image3 = new Image("sample/images/num3.jpg");

private Image image4 = new Image("sample/images/num4.jpg");

private Image image5 = new Image("sample/images/num5.jpg");

private Image image6 = new Image("sample/images/num6.jpg");

private Image image7 = new Image("sample/images/num7.jpg");

private Image image8 = new Image("sample/images/num8.jpg");

private Image image9 = new Image("sample/images/num9.jpg");

private Image image10 = new Image("sample/images/num10.jpg");

private Image image11 = new Image("sample/images/num11.jpg");

private Image image12 = new Image("sample/images/num12.jpg");

private Image image13 = new Image("sample/images/num13.jpg");

private Image image14 = new Image("sample/images/num14.jpg");

private Image image15 = new Image("sample/images/num15.jpg");

private Game game = new Game();

public static boolean isTextNull = false;

public static boolean gameIsUser = false;

public static boolean win = false;

static boolean userPositionWindowIsOpened = false;

private static boolean helpWindowOpened = false;

public static boolean processIntervalBot = false;

@FXML

Button buttonMove;

@FXML

private ImageView gridBack;

@FXML

private ImageView im00, im10, im20, im30, im01, im11, im21, im31, im02, im12, im22, im32, im03, im13, im23, im33;

@FXML

Text text;

@FXML

void initialize() {

Image background = new Image("sample/images/num0.jpg");

gridBack.setImage(background);

start();

}

private void keyPressed(Game game) {

buttonMove.setOnKeyPressed(ke ->

{

game.keyPressed(ke, game);

repaint();

});

}

public void startRandomGame() {

start();

}

private void start() {

gameIsUser = false;

win = false;

game = new Game();

keyPressed(game);

repaint();

}

public void selectUser() throws IOException {

if (!userPositionWindowIsOpened) {

FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource("userPosition.fxml"));

Parent root = fxmlLoader.load();

Stage stage = new Stage();

stage.setTitle("");

stage.setScene(new Scene(root, 370, 400));

stage.show();

stage.setOnCloseRequest(event -> userPositionWindowIsOpened = false);

userPositionWindowIsOpened = true;

}

gameIsUser = true;

}

public void startUsersPosition() throws IOException {

start();

FileReader fileReader = new FileReader("userPosition.txt");

Scanner scanner = new Scanner(fileReader);

int value;

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

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

Game.field.field[i][j] = value;

if (value == 0) {

Game.field.zeroX = i;

Game.field.zeroY = j;

}

}

}

repaint();

}

public void help() throws IOException {

if (!helpWindowOpened) {

FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource("helpWindow.fxml"));

Parent root = fxmlLoader.load();

Stage stage = new Stage();

stage.setTitle("Help");

stage.setScene(new Scene(root, 600, 180));

stage.show();

stage.setOnCloseRequest(event -> helpWindowOpened = false);

helpWindowOpened = true;

}

}

private void botInterval(double speed) {

Solver solver = new Solver(game);

processIntervalBot = true;

text.setText("Press SPACE to stop");

solver.countBotMoves = 0;

solver.botFindPath();

solver.back();

repaint();

win = false;

final boolean[] botNotInProcessOrWin = {false};

Timeline timeline = new Timeline(new KeyFrame(Duration.seconds(speed), e -> {

if (isTextNull) text.setText("");

if (!botNotInProcessOrWin[0]) {

if (processIntervalBot) {

repaint();

solver.makeOneBotMove();

} else {

botNotInProcessOrWin[0] = true;

}

if (win) {

repaint();

text.setText("");

botNotInProcessOrWin[0] = true;

}

}

}));

timeline.setCycleCount(Timeline.INDEFINITE);

timeline.play();

}

//функции, которые запускают решателя бота с заданным интервалом (0.1, 0.2, 0.4 или 1 секунда)

public void start01() { botInterval(0.1); }

public void start02() { botInterval(0.2); }

public void start04() { botInterval(0.4); }

public void start1() { botInterval(1); }

//функция, которая запускает решателя бота, который найдет только следующий ход

public void next() {

Solver solver = new Solver(game);

solver.next();

repaint();

}

//имена ImageView складываются из "im" + номер строки + номер столбца

private void repaint() {

im00.setImage(null);

im01.setImage(null);

im02.setImage(null);

im03.setImage(null);

im10.setImage(null);

im11.setImage(null);

im12.setImage(null);

im13.setImage(null);

im20.setImage(null);

im21.setImage(null);

im22.setImage(null);

im23.setImage(null);

im30.setImage(null);

im31.setImage(null);

im32.setImage(null);

im33.setImage(null);

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

for (int k = 1; k < 16; k++) {

if (Game.field.field[i][j] == k) {

whatImageView(i, j).setImage(whatImage(k));

}

}

}

}

}

private Image whatImage(int number) {

if (number == 1) return image1;

if (number == 2) return image2;

if (number == 3) return image3;

if (number == 4) return image4;

if (number == 5) return image5;

if (number == 6) return image6;

if (number == 7) return image7;

if (number == 8) return image8;

if (number == 9) return image9;

if (number == 10) return image10;

if (number == 11) return image11;

if (number == 12) return image12;

if (number == 13) return image13;

if (number == 14) return image14;

if (number == 15) return image15;

return image1;

}

private ImageView whatImageView(int x, int y) {

if (x == 0 && y == 0) return im00;

if (x == 1 && y == 0) return im01;

if (x == 2 && y == 0) return im02;

if (x == 3 && y == 0) return im03;

if (x == 0 && y == 1) return im10;

if (x == 1 && y == 1) return im11;

if (x == 2 && y == 1) return im12;

if (x == 3 && y == 1) return im13;

if (x == 0 && y == 2) return im20;

if (x == 1 && y == 2) return im21;

if (x == 2 && y == 2) return im22;

if (x == 3 && y == 2) return im23;

if (x == 0 && y == 3) return im30;

if (x == 1 && y == 3) return im31;

if (x == 2 && y == 3) return im32;

if (x == 3 && y == 3) return im33;

return im00;

}

}

## **mainWindow.fxml**

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>

<?import javafx.scene.control.Menu?>

<?import javafx.scene.control.MenuBar?>

<?import javafx.scene.control.MenuItem?>

<?import javafx.scene.image.ImageView?>

<?import javafx.scene.layout.Pane?>

<?import javafx.scene.text.Text?>

<Pane fx:id="mainPane" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="940.0"

prefWidth="1100.0" xmlns="http://javafx.com/javafx/11.0.1" xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.view.MainWindow">

<children>

<Text fx:id="text" layoutX="360.0" layoutY="80.0" strokeType="OUTSIDE" strokeWidth="0.0"/>

<Button fx:id="buttonMove" layoutX="342.0" layoutY="145.0" mnemonicParsing="false" text="Button" />

<ImageView fx:id="gridBack" fitHeight="800.0" fitWidth="800.0" layoutX="20.0" layoutY="120.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im00" fitHeight="200.0" fitWidth="200.0" layoutX="20.0" layoutY="120.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im01" fitHeight="200.0" fitWidth="200.0" layoutX="220.0" layoutY="120.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im02" fitHeight="200.0" fitWidth="200.0" layoutX="420.0" layoutY="120.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im03" fitHeight="200.0" fitWidth="200.0" layoutX="620.0" layoutY="120.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im10" fitHeight="200.0" fitWidth="200.0" layoutX="20.0" layoutY="320.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im11" fitHeight="200.0" fitWidth="200.0" layoutX="220.0" layoutY="320.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im12" fitHeight="200.0" fitWidth="200.0" layoutX="420.0" layoutY="320.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im13" fitHeight="200.0" fitWidth="200.0" layoutX="620.0" layoutY="320.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im20" fitHeight="200.0" fitWidth="200.0" layoutX="20.0" layoutY="520.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im21" fitHeight="200.0" fitWidth="200.0" layoutX="220.0" layoutY="520.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im22" fitHeight="200.0" fitWidth="200.0" layoutX="420.0" layoutY="520.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im23" fitHeight="200.0" fitWidth="200.0" layoutX="620.0" layoutY="520.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im30" fitHeight="200.0" fitWidth="200.0" layoutX="20.0" layoutY="720.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im31" fitHeight="200.0" fitWidth="200.0" layoutX="220.0" layoutY="720.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im32" fitHeight="200.0" fitWidth="200.0" layoutX="420.0" layoutY="720.0" pickOnBounds="true" preserveRatio="true" />

<ImageView fx:id="im33" fitHeight="200.0" fitWidth="200.0" layoutX="620.0" layoutY="720.0" pickOnBounds="true" preserveRatio="true" />

<MenuBar layoutY="-1.0" prefHeight="26.0" prefWidth="1100.0">

<Menu mnemonicParsing="false" text="Start game">

<MenuItem mnemonicParsing="false" onAction="#startRandomGame" text="Random"/>

<MenuItem mnemonicParsing="false" onAction="#selectUser" text="Input user's values"/>

<MenuItem mnemonicParsing="false" onAction="#startUsersPosition" text="Start user's position"/>

<MenuItem mnemonicParsing="false" onAction="#help" text="Help"/>

</Menu>

<Menu mnemonicParsing="false" text="Bot (auto)">

<MenuItem mnemonicParsing="false" onAction="#start01" text="0.1 sec"/>

<MenuItem mnemonicParsing="false" onAction="#start02" text="0.2 sec"/>

<MenuItem mnemonicParsing="false" onAction="#start04" text="0.4 sec"/>

<MenuItem mnemonicParsing="false" onAction="#start1" text="1 sec"/>

<MenuItem mnemonicParsing="false" onAction="#next" text="Next (press Q)"/>

</Menu>

</MenuBar>

</children>

</Pane>

## **UserPositionWindow.java**

package sample.view;

import javafx.fxml.FXML;

import javafx.scene.control.Button;

import javafx.scene.control.TextField;

import javafx.scene.text.Text;

import javafx.stage.Stage;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.Arrays;

import java.util.HashSet;

import java.util.Scanner;

import java.util.Set;

import static sample.logic.Field.f1;

import static sample.logic.Field.f2;

import static sample.logic.Field.f3;

import static sample.logic.Field.f4;

import static sample.logic.Field.f5;

import static sample.logic.Field.f6;

import static sample.logic.Field.f7;

import static sample.logic.Field.f8;

import static sample.logic.Field.f9;

import static sample.logic.Field.f10;

import static sample.logic.Field.f11;

import static sample.logic.Field.f12;

import static sample.logic.Field.f13;

import static sample.logic.Field.f14;

import static sample.logic.Field.f15;

import static sample.logic.Field.f16;

public class UserPositionWindow {

private static boolean checkingSymbolsOK;

public boolean checking\_is\_ok;

@FXML

private Text text;

@FXML

private Button okButton;

@FXML

public TextField t1, t2, t3, t4, t5, t6, t7, t8, t9, t10, t11, t12, t13, t14, t15, t16;

@FXML

void initialize() throws IOException {

FileReader fileReader = new FileReader("userPosition.txt");

Scanner scanner = new Scanner(fileReader);

t1.setText(scanner.nextLine());

t2.setText(scanner.nextLine());

t3.setText(scanner.nextLine());

t4.setText(scanner.nextLine());

t5.setText(scanner.nextLine());

t6.setText(scanner.nextLine());

t7.setText(scanner.nextLine());

t8.setText(scanner.nextLine());

t9.setText(scanner.nextLine());

t10.setText(scanner.nextLine());

t11.setText(scanner.nextLine());

t12.setText(scanner.nextLine());

t13.setText(scanner.nextLine());

t14.setText(scanner.nextLine());

t15.setText(scanner.nextLine());

t16.setText(scanner.nextLine());

}

@FXML

private void back() throws IOException {

try {

f1 = Integer.parseInt(t1.getText());

f2 = Integer.parseInt(t2.getText());

f3 = Integer.parseInt(t3.getText());

f4 = Integer.parseInt(t4.getText());

f5 = Integer.parseInt(t5.getText());

f6 = Integer.parseInt(t6.getText());

f7 = Integer.parseInt(t7.getText());

f8 = Integer.parseInt(t8.getText());

f9 = Integer.parseInt(t9.getText());

f10 = Integer.parseInt(t10.getText());

f11 = Integer.parseInt(t11.getText());

f12 = Integer.parseInt(t12.getText());

f13 = Integer.parseInt(t13.getText());

f14 = Integer.parseInt(t14.getText());

f15 = Integer.parseInt(t15.getText());

f16 = Integer.parseInt(t16.getText());

check();

} catch (NumberFormatException ignored) {

}

if (checkingSymbolsOK && checking\_is\_ok) {

Stage window = (Stage) okButton.getScene().getWindow();

window.close();

MainWindow.userPositionWindowIsOpened = false;

rewriteFile();

} else {

if (!checkingSymbolsOK) text.setText("Введите правильные значения");

else text.setText("Поле неправильно перемешано");

}

}

private void rewriteFile() throws IOException {

FileWriter writer = new FileWriter("userPosition.txt", false);

writer.write(f1 + "\n" + f2 + "\n" + f3 + "\n" + f4 + "\n" + f5 + "\n" + f6 + "\n" + f7 + "\n" + f8 + "\n" +

f9 + "\n" + f10 + "\n" + f11 + "\n" + f12 + "\n" + f13 + "\n" + f14 + "\n" + f15 + "\n" + f16);

writer.flush();

}

//проверка правильности введённого пользователем поля по формуле, описанной на http://pyatnashki.wmsite.ru/kombinacyi

public void check() {

Set<Integer> numbers = new HashSet<>(Arrays.asList(f1, f2, f3, f4, f5, f6, f7, f8,

f9, f10, f11, f12, f13, f14, f15, f16));

HashSet<Integer> numbers2 = new HashSet<>(Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 0));

checkingSymbolsOK = numbers.containsAll(numbers2);

int[][] winfield = {{f1, f5, f9, f13}, {f2, f6, f10, f14}, {f3, f7, f11, f15}, {f4, f8, f12, f16}};

int counter = 0;

int noY = 0;

for (int i = 0; i < 16; i++) {

for (int j = i + 1; j < 16; j++) {

if (winfield[j % 4][j / 4] != 0) {

if (winfield[i % 4][i / 4] > winfield[j % 4][j / 4]) {

counter++;

}

}

}

}

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

if (winfield[i][j] == 0) noY = j;

}

}

checking\_is\_ok = (counter + noY) % 2 != 0;

}

}

## **userPosition.fxml**

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>

<?import javafx.scene.control.TextField?>

<?import javafx.scene.layout.Pane?>

<?import javafx.scene.text.Text?>

<Pane fx:id="mainPane" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity"

prefHeight="400.0"

prefWidth="370.0" xmlns="http://javafx.com/javafx/11.0.1" xmlns:fx="http://javafx.com/fxml/1"

fx:controller="sample.view.UserPositionWindow">

<Button fx:id="okButton" layoutX="144.0" layoutY="351.0" mnemonicParsing="false" onAction="#back" prefHeight="25.0"

prefWidth="73.0" text="ok"/>

<TextField fx:id="t16" layoutX="260.0" layoutY="280.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t14" layoutX="120.0" layoutY="280.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t13" layoutX="50.0" layoutY="280.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t12" layoutX="260.0" layoutY="210.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t10" layoutX="120.0" layoutY="210.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t9" layoutX="50.0" layoutY="210.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t8" layoutX="260.0" layoutY="140.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t15" layoutX="190.0" layoutY="280.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t11" layoutX="190.0" layoutY="210.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t4" layoutX="260.0" layoutY="70.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t1" layoutX="50.0" layoutY="70.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t7" layoutX="190.0" layoutY="140.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t5" layoutX="50.0" layoutY="140.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t6" layoutX="120.0" layoutY="140.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t2" layoutX="120.0" layoutY="70.0" prefHeight="60.0" prefWidth="60.0"/>

<TextField fx:id="t3" layoutX="190.0" layoutY="70.0" prefHeight="60.0" prefWidth="60.0"/>

<Text fx:id="text" layoutX="78.0" layoutY="44.0" strokeType="OUTSIDE" strokeWidth="0.0"/>

</Pane>

## **HelpWindow.java**

package sample.view;

public class HelpWindow {

}

## **helpWindow.fxml**

## <?xml version="1.0" encoding="UTF-8"?>

## <?import javafx.scene.layout.Pane?>

## <?import javafx.scene.text.Text?>

## <Pane fx:id="mainPane" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0"

## prefWidth="370.0" xmlns="http://javafx.com/javafx/11.0.1" xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.view.HelpWindow">

## <children>

## <Text fx:id="text1" layoutX="48.0" layoutY="44.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Игра Пятнашки."/>

## <Text fx:id="text2" layoutX="48.0" layoutY="64.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Цель игры: собрать поле таким образом, чтобы плитки расположились"/>

## <Text fx:id="text3" layoutX="48.0" layoutY="84.0" strokeType="OUTSIDE" strokeWidth="0.0" text="по порядку от 1 до 15."/>

## <Text fx:id="text4" layoutX="48.0" layoutY="104.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Играть можно самому, используя клавиши ВВЕРХ, ВНИЗ, ВЛЕВО, ВПРАВО"/>

## <Text fx:id="text5" layoutX="48.0" layoutY="124.0" strokeType="OUTSIDE" strokeWidth="0.0" text="или же получить автоматическое решение."/>

## </children>

## </Pane>

## **Field.java**

package sample.logic;

import sample.view.MainWindow;

import java.util.HashSet;

import java.util.Random;

import java.util.Set;

public class Field {

public static int f1, f2, f3, f4, f5, f6, f7, f8, f9, f10, f11, f12, f13, f14, f15, f16;

public int[][] field;

public int zeroX;

public int zeroY;

private Random random = new Random();

private Set<Integer> numbers = new HashSet<>();

public boolean checking\_is\_ok = true;

public Field() {

if (MainWindow.gameIsUser) {

field = new int[][]{{f1, f5, f9, f13}, {f2, f6, f10, f14}, {f3, f7, f11, f15}, {f4, f8, f12, f16}};

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

if (field[i][j]==0) {

zeroY = j;

zeroX = i;

}

}

}

}

else {

field = new int[4][4];

fill();

}

MainWindow.gameIsUser=false;

}

public void fill() {

numbers.clear();

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

boolean numbersContainsC = false;

int value = 0;

while (!numbersContainsC) {

value = random.nextInt(16);

if (!numbers.contains(value)) {

numbers.add(value);

numbersContainsC = true;

}

}

if (value == 0) {

zeroX = i;

zeroY = j;

}

field[i][j] = value;

}

}

checkWinnable();

if (!checking\_is\_ok) fill();

}

//проверка решаемости - по формуле, описанной на http://pyatnashki.wmsite.ru/kombinacyi

public void checkWinnable() {

checking\_is\_ok = true;

int counter = 0;

for (int i = 0; i < 16; i++) {

for (int j = i + 1; j < 16; j++) {

if (field[j % 4][j / 4] != 0)

if (field[i % 4][i / 4] > field[j % 4][j / 4]) {

counter++;

}

}

}

if ((counter + zeroY ) % 2 == 0) checking\_is\_ok = false;

}

/\*private void tostring() {

StringBuilder s = new StringBuilder();

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

s.append(field[i][j]);

s.append(", ");

}

s.append("\n");

}

System.out.println(s);

}\*/

}

## **Game.java**

package sample.logic;

import javafx.scene.input.KeyCode;

import javafx.scene.input.KeyEvent;

import sample.view.MainWindow;

public class Game {

public static Field field;

public Game() {

field = new Field();

}

public void keyPressed(KeyEvent keyEvent, Game game) {

if (keyEvent.getCode().getName().equals("D") || keyEvent.getCode().equals(KeyCode.RIGHT)) {

if (field.zeroX != 0) {

moveRight();

checkWin();

}

}

if (keyEvent.getCode().getName().equals("S") || keyEvent.getCode().equals(KeyCode.DOWN)) {

if (field.zeroY != 0) {

moveDown();

checkWin();

}

}

if (keyEvent.getCode().getName().equals("A") || keyEvent.getCode().equals(KeyCode.LEFT)) {

if (field.zeroX != 3) {

moveLeft();

checkWin();

}

}

if (keyEvent.getCode().getName().equals("W") || keyEvent.getCode().equals(KeyCode.UP)) {

if (field.zeroY != 3) {

moveUp();

checkWin();

}

}

if (keyEvent.getCode().equals(KeyCode.SPACE)) {

if (MainWindow.processIntervalBot) {

MainWindow.processIntervalBot = false;

MainWindow.isTextNull = true;

}

}

if (keyEvent.getCode().getName().equals("Q")) {

Solver solver = new Solver(game);

solver.next();

}

}

void moveRight() {

field.field[field.zeroX][field.zeroY] = field.field[field.zeroX - 1][field.zeroY];

field.field[field.zeroX - 1][field.zeroY] = 0;

field.zeroX--;

}

void moveDown() {

field.field[field.zeroX][field.zeroY] = field.field[field.zeroX][field.zeroY - 1];

field.field[field.zeroX][field.zeroY - 1] = 0;

field.zeroY--;

}

void moveLeft() {

field.field[field.zeroX][field.zeroY] = field.field[field.zeroX + 1][field.zeroY];

field.field[field.zeroX + 1][field.zeroY] = 0;

field.zeroX++;

}

void moveUp() {

field.field[field.zeroX][field.zeroY] = field.field[field.zeroX][field.zeroY + 1];

field.field[field.zeroX][field.zeroY + 1] = 0;

field.zeroY++;

}

private static int[][] winfield = {{1, 5, 9, 13}, {2, 6, 10, 14}, {3, 7, 11, 15}, {4, 8, 12, 0}};

void checkWin() {

int counter = 0;

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

if (winfield[i][j] == field.field[i][j]) counter++;

}

}

MainWindow.win = counter == 16;

}

}

## **Solver.java**

package sample.logic;

import sample.view.MainWindow;

import java.util.ArrayList;

import java.util.List;

import static java.lang.Math.abs;

public class Solver {

private Game game;

public Solver(Game game) {

this.game = game;

MainWindow.isTextNull = false;

}

private static int[][] winfield = {{1, 5, 9, 13}, {2, 6, 10, 14}, {3, 7, 11, 15}, {4, 8, 12, 0}};

// эвристическая функция, которая находит наилучшее положение для следующего хода

// fx = hx \* kf + gx,

// где fx - самое значение функции,

// gx - количество ходов, которые привели к данной позиции,

// hx - сумма расстояний между расположением каждой из плиток и их финишной позиции,

// kf - коэффициент, показывающий, насколько hx в данном алгоритме важнее gx

// (чем меньше kf - тем дольше, но эффективнее происходит поиск пути)

private double fx;

private int hx;

private int gx;

private final double kf = 30;

private static Field fieldBeforeBot = new Field();

private List<int[][]> path;

private static List<int[][]> pathReversed;

private List<int[][]> opened;

private List<Double> heuristicOfOpened;

private List<int[][]> closed;

private List<Integer> intArrayOpenedSize;

private List<Integer> intArrayClosedSize;

private int countWhiles;

private int openedSize;

private int closedSize;

public int countBotMoves = 0;

public void botFindPath() {

opened = new ArrayList<>();

heuristicOfOpened = new ArrayList<>();

closed = new ArrayList<>();

path = new ArrayList<>();

pathReversed = new ArrayList<>();

intArrayOpenedSize = new ArrayList<>();

intArrayClosedSize = new ArrayList<>();

countWhiles = 0;

openedSize = 0;

closedSize = 0;

opened.add(Game.field.field);

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

fieldBeforeBot.field[i][j] = Game.field.field[i][j];

}

}

while (!opened.isEmpty()) {

if (countWhiles % 1000 == 1) System.out.println("countWhiles = " + countWhiles);

//check win

int counter = 0;

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

if (winfield[i][j] == Game.field.field[i][j]) counter++;

}

}

if (counter == 16) {

boolean isSizeNotZero = true;

int size = intArrayClosedSize.size();

while (isSizeNotZero) {

if (closedSize != 0) path.add(closed.get(size));

if (size == 0) isSizeNotZero = false;

else size = intArrayClosedSize.get(size - 1);

}

System.out.println("PathSize = " + path.size());

MainWindow.win = true;

for (int j = 1; j < path.size(); j++) {

pathReversed.add(path.get(path.size() - j - 1));

}

return;

}

//find min in heuristic

double min = Integer.MAX\_VALUE;

int minInArray = 0;

if (openedSize != 0) {

for (int j = 0; j < openedSize - 1; j++) {

if (heuristicOfOpened.get(j) < min) {

min = heuristicOfOpened.get(j);

minInArray = j;

}

}

}

//field=best

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

Game.field.field[i][j] = opened.get(minInArray)[i][j];

if (Game.field.field[i][j] == 0) {

Game.field.zeroY = j;

Game.field.zeroX = i;

}

}

}

//clear

opened.remove(minInArray);

if (openedSize != 0) {

openedSize--;

heuristicOfOpened.remove(minInArray);

}

//add closed

int[][] fieldPosition = new int[4][4];

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

fieldPosition[i][j] = Game.field.field[i][j];

}

}

closed.add(fieldPosition);

closedSize++;

if (openedSize != 0) {

intArrayClosedSize.add(intArrayOpenedSize.get(minInArray));

intArrayOpenedSize.remove(minInArray);

}

//neighbors

if (Game.field.zeroY != 3) {

game.moveUp();

checkPosition();

game.moveDown();

}

if (Game.field.zeroY != 0) {

game.moveDown();

checkPosition();

game.moveUp();

}

if (Game.field.zeroX != 3) {

game.moveLeft();

checkPosition();

game.moveRight();

}

if (Game.field.zeroX != 0) {

game.moveRight();

checkPosition();

game.moveLeft();

}

countWhiles++;

}

}

public void next() {

countBotMoves = 0;

botFindPath();

back();

makeOneBotMove();

}

private void checkPosition() {

//find gx

gx=0;

boolean isSizeNotZero = true;

int size = intArrayClosedSize.size();

while (isSizeNotZero) {

if (closedSize != 0) gx++;

if (size == 0) isSizeNotZero = false;

else size = intArrayClosedSize.get(size - 1);

}

//check is it in closed

boolean inClosed = false;

for (int c = 0; c < closedSize; c++) {

int counter = 0;

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

if (Game.field.field[i][j] == closed.get(c)[i][j]) {

counter++;

}

}

}

if (counter == 16) {

inClosed = true;

}

}

if (!inClosed) {

//add in opened and heuristic

int[][] arr = new int[4][4];

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

arr[i][j] = Game.field.field[i][j];

}

}

opened.add(arr);

findH();

fx = hx \* kf + gx;

heuristicOfOpened.add(fx);

openedSize++;

intArrayOpenedSize.add(closedSize - 1);

}

}

public void makeOneBotMove() {

if (pathReversed.size() > countBotMoves) Game.field.field = pathReversed.get(countBotMoves);

countBotMoves++;

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

if (Game.field.field[i][j] == 0) {

Game.field.zeroX = i;

Game.field.zeroY = j;

}

}

}

game.checkWin();

if (MainWindow.win) {

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

Game.field.field[i][j] = winfield[i][j];

}

}

Game.field.zeroX = 3;

Game.field.zeroY = 3;

}

}

public void back() {

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

Game.field.field[i][j] = fieldBeforeBot.field[i][j];

if (Game.field.field[i][j] == 0) {

Game.field.zeroX = i;

Game.field.zeroY = j;

}

}

}

}

private void findH() {

hx = 0;

for (int j = 0; j < 4; j++) {

for (int i = 0; i < 4; i++) {

for (int k = 1; k < 16; k++) {

findHContinue(i, j, k);

}

}

}

}

private void findHContinue(int i, int j, int value) {

if (value == Game.field.field[i][j]) {

int row = value % 4;

if (value % 4 == 0) {

row = 4;

}

int column = 0;

if (value % 4 == 0) {

column = 1;

}

hx += (abs(i - row + 1) + abs(j - value / 4 + column));

}

}

}

# **ТЕСТЫ**

Были созданы тесты в классе Test.java, проверяющие работу программы.

package test;

import sample.logic.Field;

import sample.view.UserPositionWindow;

import static org.junit.jupiter.api.Assertions.\*;

import static sample.logic.Field.\*;

import static sample.logic.Field.f16;

class Test {

//проверка на то, является ли стартовое поле решаемым

private Field field = new Field();

@org.junit.jupiter.api.Test

void startRandomPositionTest() {

field.fill();

field.checkWinnable();

assertTrue(field.checking\_is\_ok);

}

//Показывает, что поле, заданное пользователем, является решаемым

private UserPositionWindow userPositionWindow = new UserPositionWindow();

@org.junit.jupiter.api.Test

void startUsersPositionTestTrue() {

String t1 = "1";

String t2 = "2";

String t3 = "3";

String t4 = "4";

String t5 = "5";

String t6 = "6";

String t7 = "8";

String t8 = "7";

String t9 = "9";

String t10 = "11";

String t11 = "10";

String t12 = "12";

String t13 = "13";

String t14 = "14";

String t15 = "15";

String t16 = "0";

f1 = Integer.parseInt(t1);

f2 = Integer.parseInt(t2);

f3 = Integer.parseInt(t3);

f4 = Integer.parseInt(t4);

f5 = Integer.parseInt(t5);

f6 = Integer.parseInt(t6);

f7 = Integer.parseInt(t7);

f8 = Integer.parseInt(t8);

f9 = Integer.parseInt(t9);

f10 = Integer.parseInt(t10);

f11 = Integer.parseInt(t11);

f12 = Integer.parseInt(t12);

f13 = Integer.parseInt(t13);

f14 = Integer.parseInt(t14);

f15 = Integer.parseInt(t15);

f16 = Integer.parseInt(t16);

userPositionWindow.check();

assertTrue(userPositionWindow.checking\_is\_ok);

}

//Показывает, что поле, заданное пользователем, не является решаемым

@org.junit.jupiter.api.Test

void startUsersPositionTestFalse() {

String t1 = "2";

String t2 = "1";

String t3 = "3";

String t4 = "4";

String t5 = "5";

String t6 = "6";

String t7 = "8";

String t8 = "7";

String t9 = "9";

String t10 = "11";

String t11 = "10";

String t12 = "12";

String t13 = "13";

String t14 = "14";

String t15 = "15";

String t16 = "0";

f1 = Integer.parseInt(t1);

f2 = Integer.parseInt(t2);

f3 = Integer.parseInt(t3);

f4 = Integer.parseInt(t4);

f5 = Integer.parseInt(t5);

f6 = Integer.parseInt(t6);

f7 = Integer.parseInt(t7);

f8 = Integer.parseInt(t8);

f9 = Integer.parseInt(t9);

f10 = Integer.parseInt(t10);

f11 = Integer.parseInt(t11);

f12 = Integer.parseInt(t12);

f13 = Integer.parseInt(t13);

f14 = Integer.parseInt(t14);

f15 = Integer.parseInt(t15);

f16 = Integer.parseInt(t16);

userPositionWindow.check();

assertFalse(userPositionWindow.checking\_is\_ok);

}

//Показывает, что поле, заданное пользователем имеет неверные входные значения

@org.junit.jupiter.api.Test

void wrongValues() {

String t1 = "2qqqqqq";

String t2 = "1";

String t3 = "3";

String t4 = "4";

String t5 = "5";

String t6 = "6";

String t7 = "8";

String t8 = "7";

String t9 = "9";

String t10 = "11";

String t11 = "10";

String t12 = "12";

String t13 = "13";

String t14 = "14";

String t15 = "15";

String t16 = "0";

try {

f1 = Integer.parseInt(t1);

f2 = Integer.parseInt(t2);

f3 = Integer.parseInt(t3);

f4 = Integer.parseInt(t4);

f5 = Integer.parseInt(t5);

f6 = Integer.parseInt(t6);

f7 = Integer.parseInt(t7);

f8 = Integer.parseInt(t8);

f9 = Integer.parseInt(t9);

f10 = Integer.parseInt(t10);

f11 = Integer.parseInt(t11);

f12 = Integer.parseInt(t12);

f13 = Integer.parseInt(t13);

f14 = Integer.parseInt(t14);

f15 = Integer.parseInt(t15);

f16 = Integer.parseInt(t16);

userPositionWindow.check();

assertFalse(userPositionWindow.checking\_is\_ok);

} catch (Exception e) {

System.out.println("Значения введены неверно");

}

}

}

# **СКРИНШОТЫ ПРОГРАММЫ**

# 

Рис.1 Внешний вид приложения

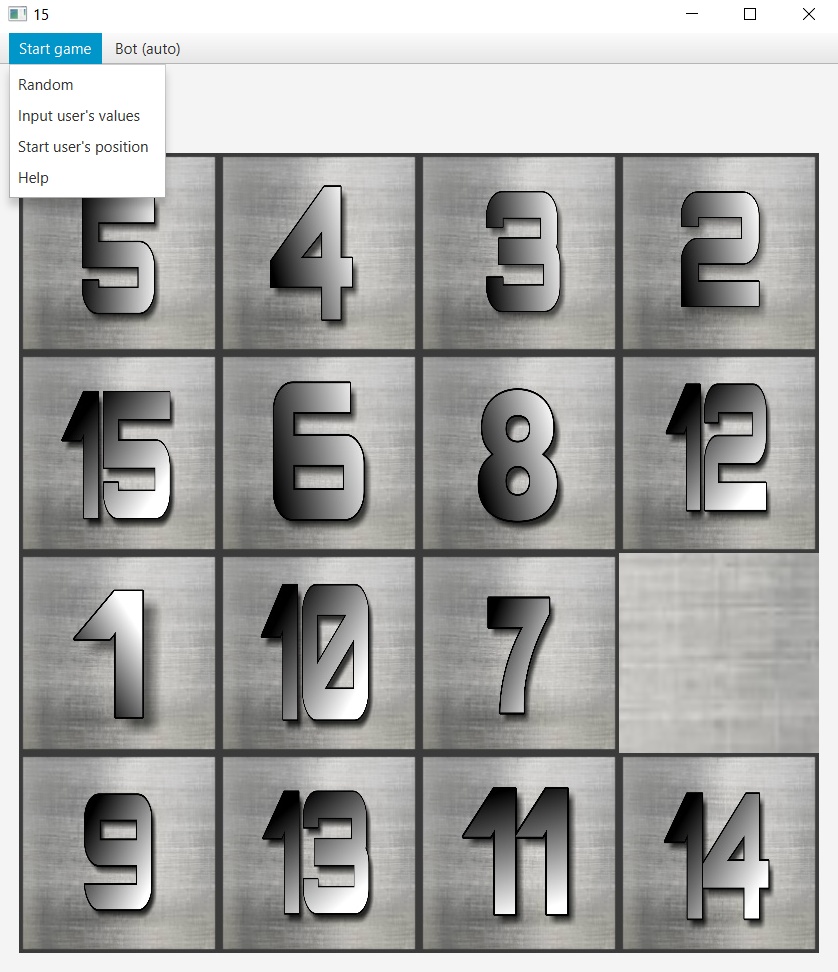


Рис. 2.1 Меню игры

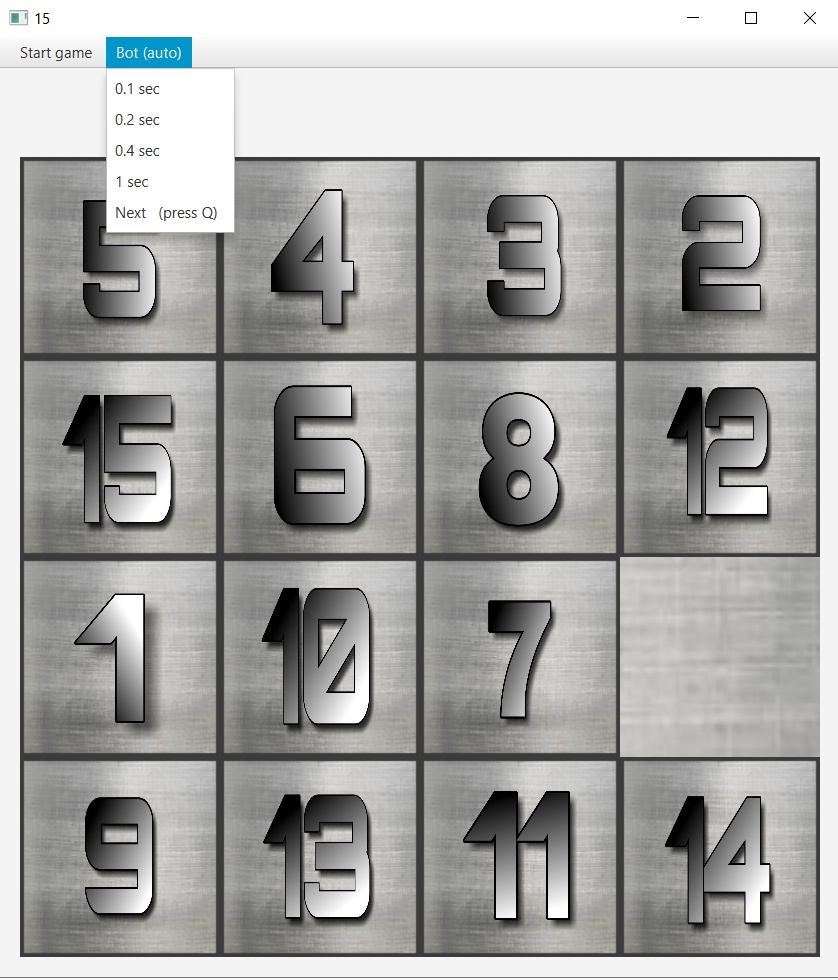


Рис. 2.2 Меню запуска решателя

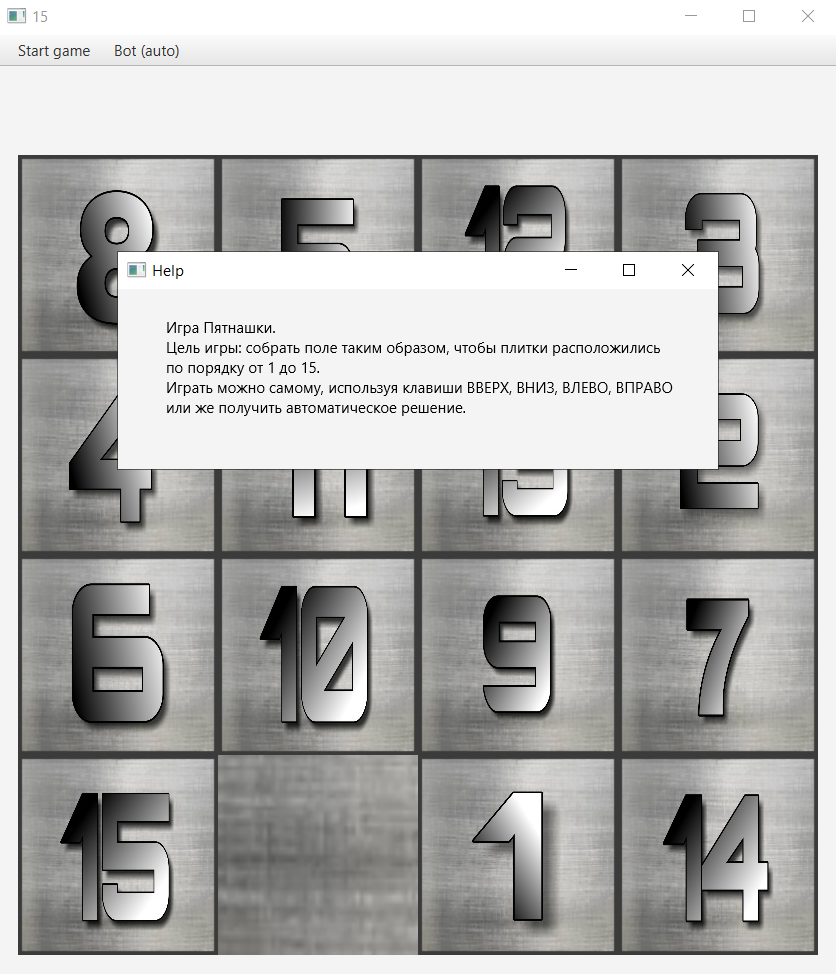


Рис. 3.1 Информация об игре

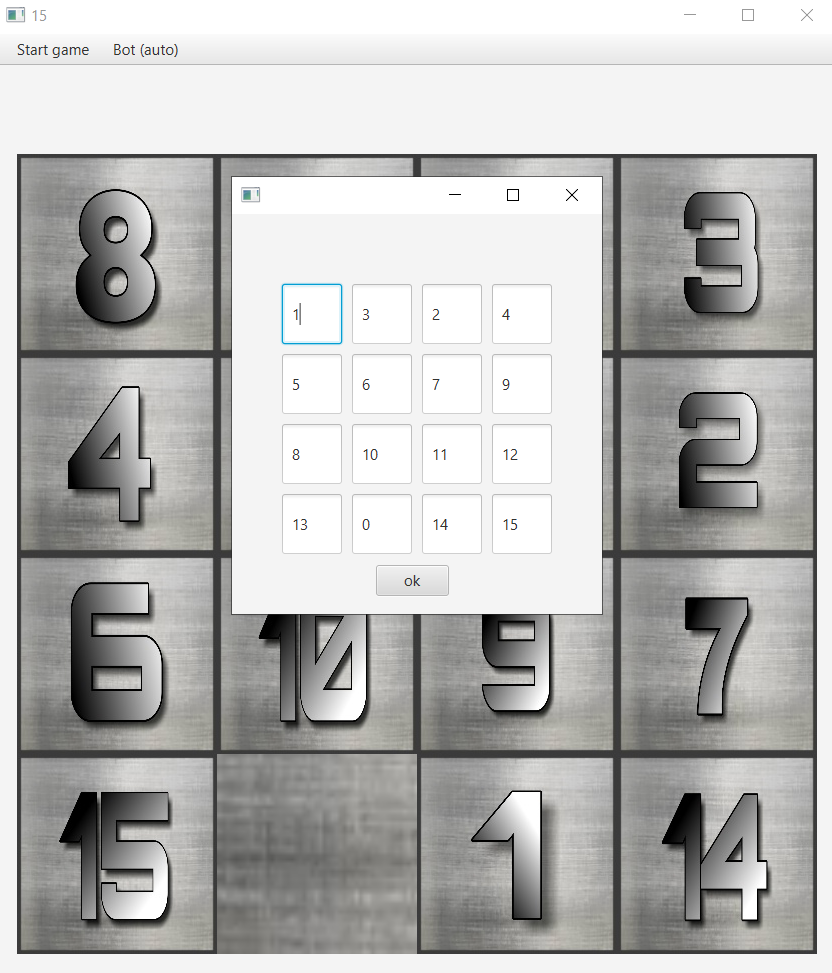


Рис. 4.1 Окно ввода данных пользователем

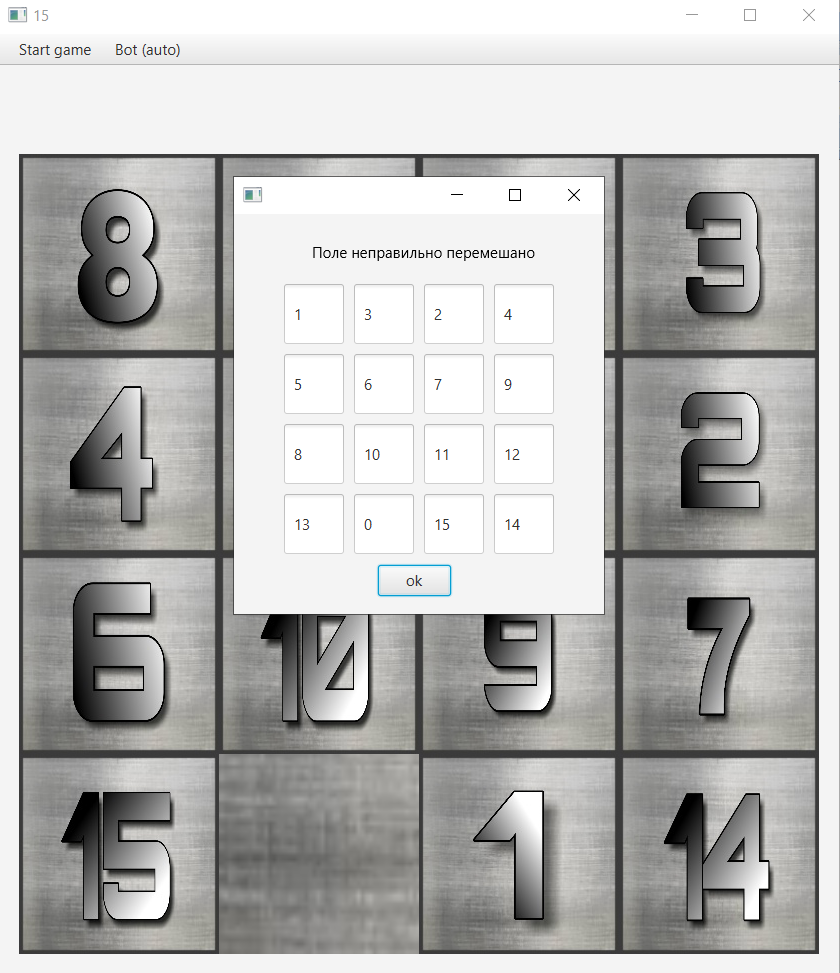


Рис. 4.2 Ошибка при неверном вводе данных пользователем