ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

***«*САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»**

Институт компьютерных наук и технологий

**Отчет о прохождении учебной практики**

Зайцева Елизавета Андреевна

*(Ф.И.О. обучающегося)*

|  |
| --- |
| 2 курс, гр.3530901/80003 |

|  |
| --- |
| 09.03.01 «Информатика и вычислительная техника» |

*(Направление подготовки (код и наименование)*

|  |  |
| --- | --- |
| **Место прохождения практики**: Высшая школа информационных систем и суперкомпьютерных технологий (ВШИСиСТ) ИКНТ ФГАОУ ВО «СПбПУ» с использованием электронного обучения и дистанционных образовательных технологий. | |
| **Сроки практики:** с 22 июня по 18 июля 2020 г. |

|  |
| --- |
| **Руководитель практики:** |

|  |
| --- |
| Жвариков В. А., к.т.н., доцент ВШИСиСТ ИКНТ |

*(Ф.И.О., уч.степень, должность)*

|  |
| --- |
| **Оценка (зачет): \_\_\_\_\_\_\_\_\_\_\_** |
|  | | |
| Руководитель практики : Жвариков В. А. | |

|  |
| --- |
| Обучающийся: Зайцева Е. А. |

|  |
| --- |
| Дата: |

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

***«*САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»**

Институт компьютерных наук и технологий

**ИНДИВИДУАЛЬНЫЙ ПЛАН (ЗАДАНИЕ И ГРАФИК)**

**ПРОВЕДЕНИЯ ПРАКТИКИ**

|  |
| --- |
| Ф.И.О. обучающегося Зайцева Елизавета Андреевна |

|  |
| --- |
| **Направление подготовки** (код/наименование): 09.03.01, «Информатика и вычислительная техника» |
| **Профиль** (код/наименование): 09.03.01, «Информатика и вычислительная техника» |
| **Вид практики:** учебная |
| **Тип практики:** ознакомительная |
| **Место прохождения практики**: Высшая школа информационных систем и суперкомпьютерных технологий (ВШИСиСТ) ИКНТ ФГАОУ ВО «СПбПУ» с использованием электронного обучения и дистанционных образовательных технологий. |

|  |
| --- |
|  |
|  |
| Руководитель практики: Жвариков В.А., к.т.н., доцент ВШИСиСТ ИКНТ |
| *(Ф.И.О., уч.степень, должность)* |

**Рабочий график проведения учебной практики**

Сроки практики: с **22.06.2020** г. по **18.07.2020** г.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Этапы (периоды) практики | Вид работ | Сроки прохождения этапа (периода) практики |
| 1 | Организационный этап | Установочная лекция (вебинар) для разъяснения целей, задач, содержания и порядка прохождения практики, выдача сопроводительных документов по практике  Основная тема: написание игры «Японские кроссворды» для Android на языке Java | 22.06.2020 г. |
| 2 | Основной  этап | Изучение среды разработки Android Studio, а также сторонних библиотек и баз данных.  Сбор информации, обработка, систематизация и анализ фактического и теоретического материала.  Содержание практики: разработка мобильного приложения согласно заданному ТЗ.  Планируемые результаты прохождения практики: получение навыков создания мобильных программ под Android и полностью готовое мобильное приложение. | 23.06.2020 - 15.07.2020 г. |
| 3 | Заключительный этап | Подготовка отчета | 16.07 - 17.07.2020 г. |
| Защита отчета по практике (сдача зачета) | 17.07 - 18.07.2020 г |

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Зайцева Е. А.

Руководитель практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Жвариков В. А.

Оглавление

[Техническое задание 4](#_Toc45570570)

[Метод решения 5](#_Toc45570571)

[Скриншоты программы 7](#_Toc45570572)

[Использованные источники 9](#_Toc45570573)

[Приложение 9](#_Toc45570574)

# Техническое задание

**Цель проекта:**

1. Разработать на языке Java игру «Японские кроссворды»
2. Освоить навык разработки под Android, ознакомиться со средой разработки Android Studio, на основе полученных навыков разработать игру на языке Java.

**Функционал приложения:**

1. Возможность проверки верности введенного рисунка, очищения поля.
2. Правила игры: поле японского кроссворда представляет собой таблицу, оно расчерчено горизонтальными и вертикальными линиями разной толщины. Более тонкими линиями поле делится на группы по 5 клеток (как по горизонтали, так и по вертикали) исключительно для удобства подсчёта. Количество черных клеток в столбце/строке указано вначале строки/столбца. Черные клетки обязательно разделяются "пустыми" (одной или более). Само изображение в японском кроссворде формируется путем закрашивания отдельных клеток в чёрный цвет. В процессе решения необходимо найти все черные клетки и восстановить картинку по имеющимся цифрам.

GitHub-репозиторий:

[https://github.com/LizaZaytseva/Nonogram](https://vk.com/away.php?to=https%3A%2F%2Fgithub.com%2FLizaZaytseva%2FNonogram&cc_key=)

# Метод решения

При включении приложения появляется меню, где представлены 3 кнопки: «Играть», «Правила игры» и «Выход». Во время игры могут быть использованы 4 кнопки: «Закрасить клетку», «Отметить пустую клетку», «Очистить поле» и «Проверить решение». Игровое поле построено на основе gridView. В ходе решения человек должен сначала нажать на кнопку с действием, а после выбрать клетку на игровом поле. Изначально выбранной кнопкой считается «Закрасить клетку».

Решение базируется на 4 классах: MainActivity.java, Rules.java, Game.java, Logic.java.

**MainActivity.java**

Данный класс описывает работу начального экрана (меню) программы. Рассмотрим описанные методы:

getItems() – соотносит определенные в классе кнопки с теми, что прописаны в файле activity\_main.xml

play() – реализует переход в окно Game при нажатии кнопки "играть"

gameRules() – реализует переход в окно Rules при нажатии кнопки "правила игры"

exit() – завершает работу приложения при нажатии кнопки "выход"

**Rules.java**

Данный класс описывает работу окна с правилами игры. Рассмотрим описанные методы:

exitR() – возвращает в меню при нажатии кнопки «X»

**Game.java**

Данный класс описывает работу игрового окна программы. Рассмотрим описанные методы:

onItemClick() – отслеживает нажатие на элементы gridView.

onClick() – обрабатывает нажатие кнопок на игровом окне

**Logic.java**

Данный класс реализует логику игры. Рассмотрим описанные методы:

clear() – заполняет рабочий массив нулями, а массив, хранящий названия изображений, строками «nothing». Итог: поле приводится в начальное состояние.

board() – добавляет названия изображений, соответствующие значениям в массиве numberArray, в массив pictures.

getView() – реализует отрисовку рабочего поля, основываясь на массиве pictures.

setNumber() – изменяет рабочий массив numberArray в зависимости от выбранной кнопки и позиции на игровом поле.

getRow() – по порядковому номеру на игровом поле определяет номер строки в рабочем массиве.

getColumn() – по порядковому номеру на игровом поле определяет номер столбца в рабочем массиве.

isWin() – определяет, выиграл ли человек после текущего хода, сравнивая рабочий массив numberArray и массив с решением solutionArray

isRight() – проверят на наличие ошибок по текущему состоянии рабочего массива

# Скриншоты программы

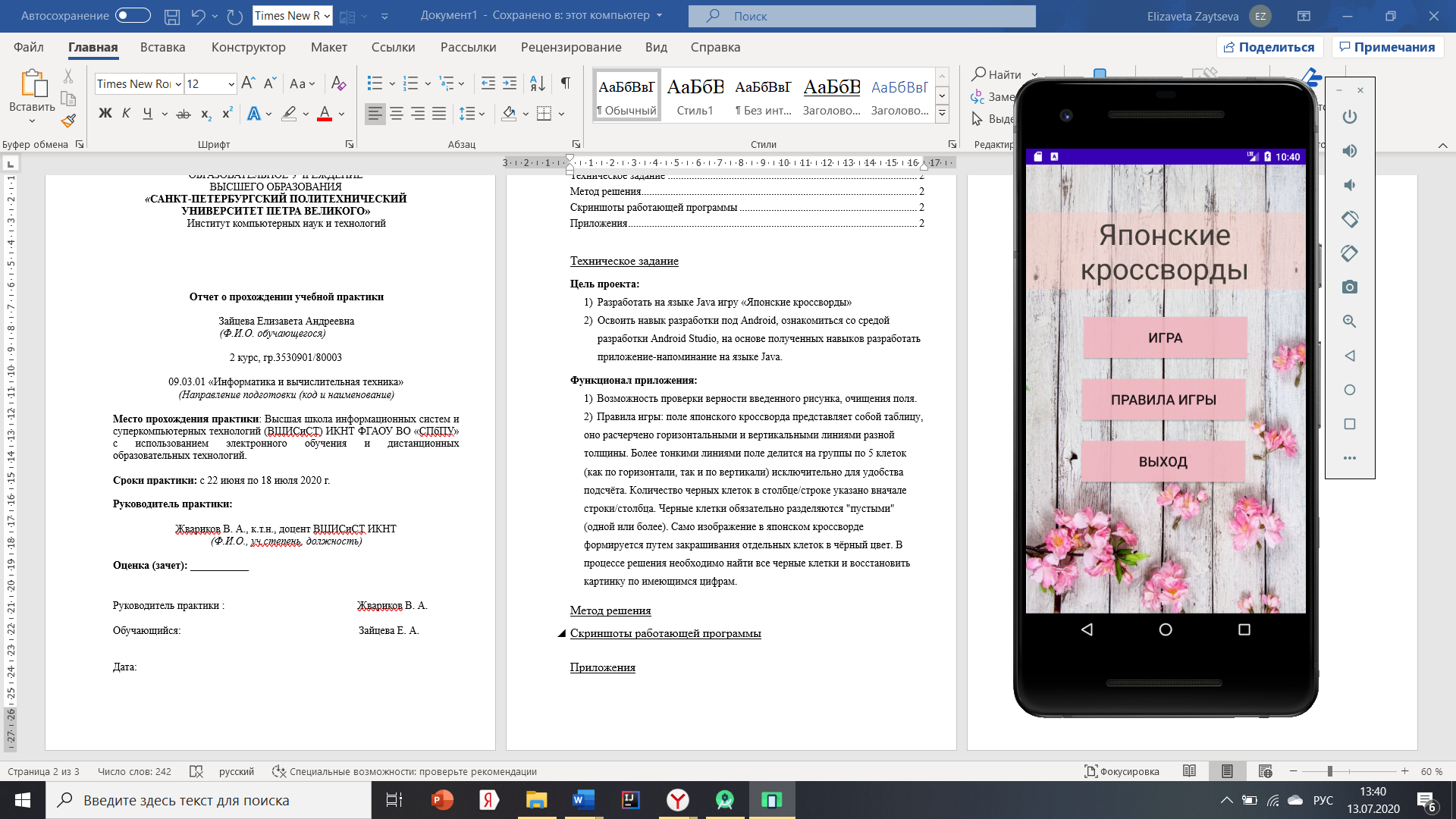
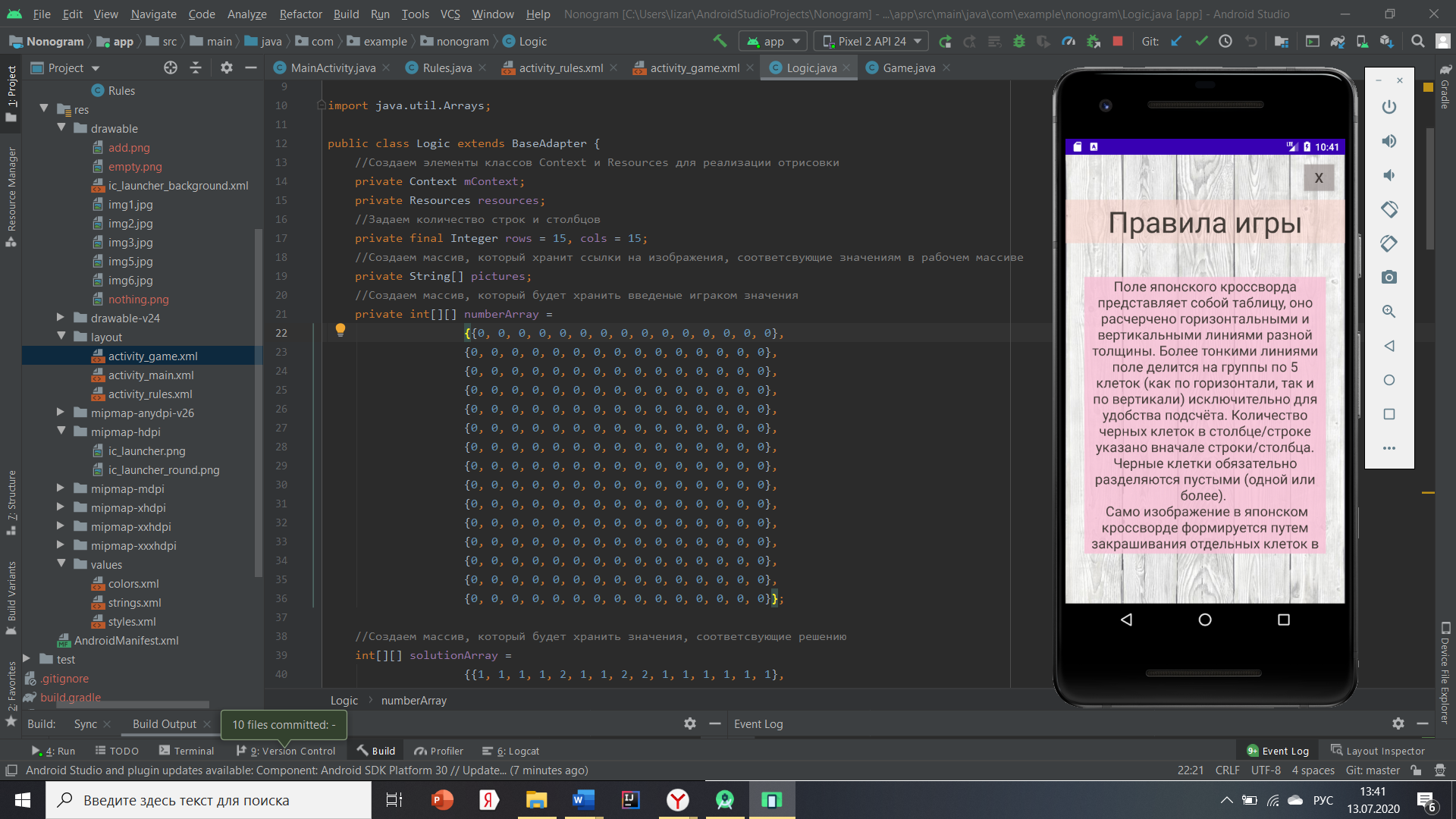
 

Рис. 1 Меню приложения Рис. 2 Окно с правилами

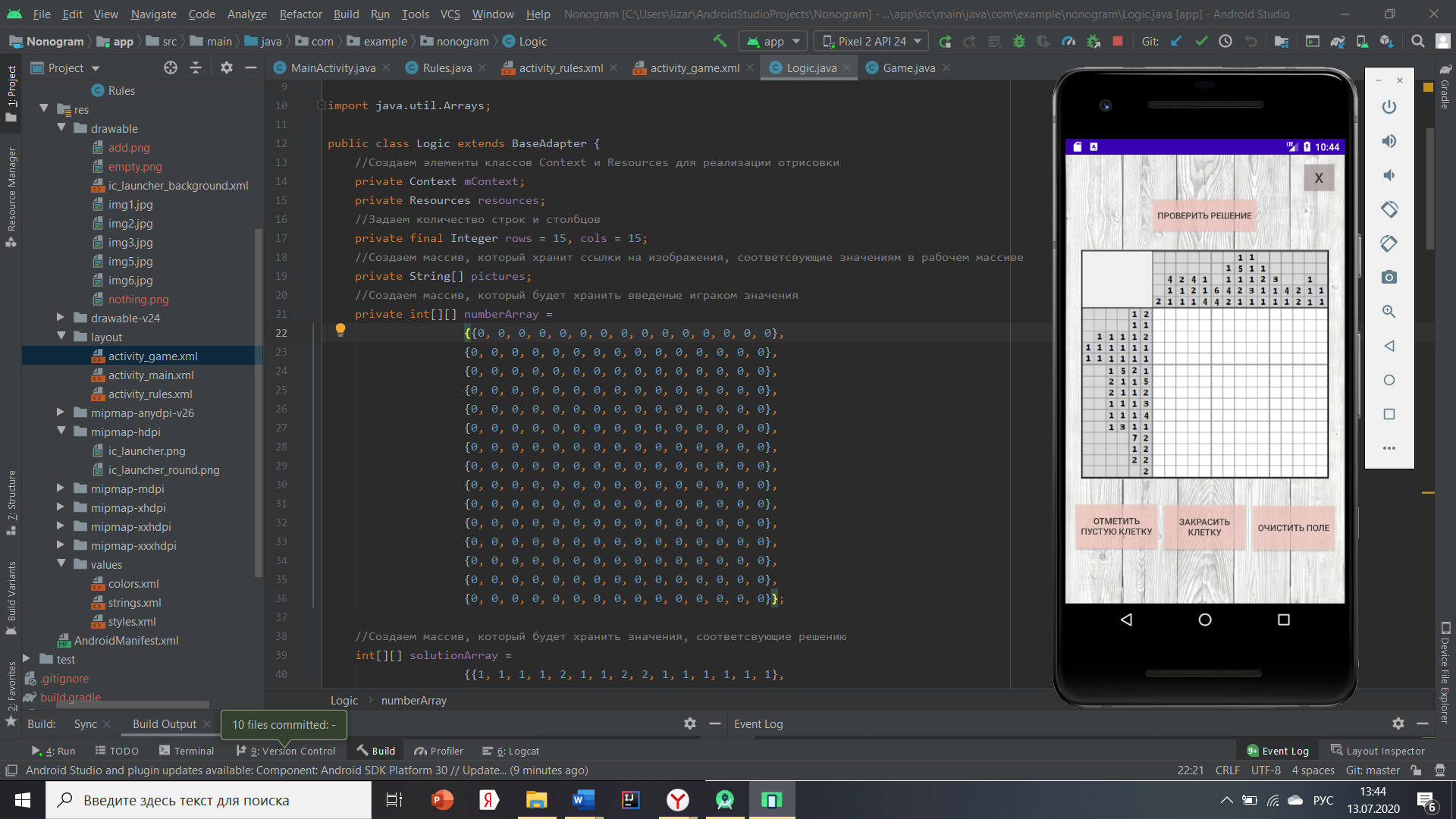
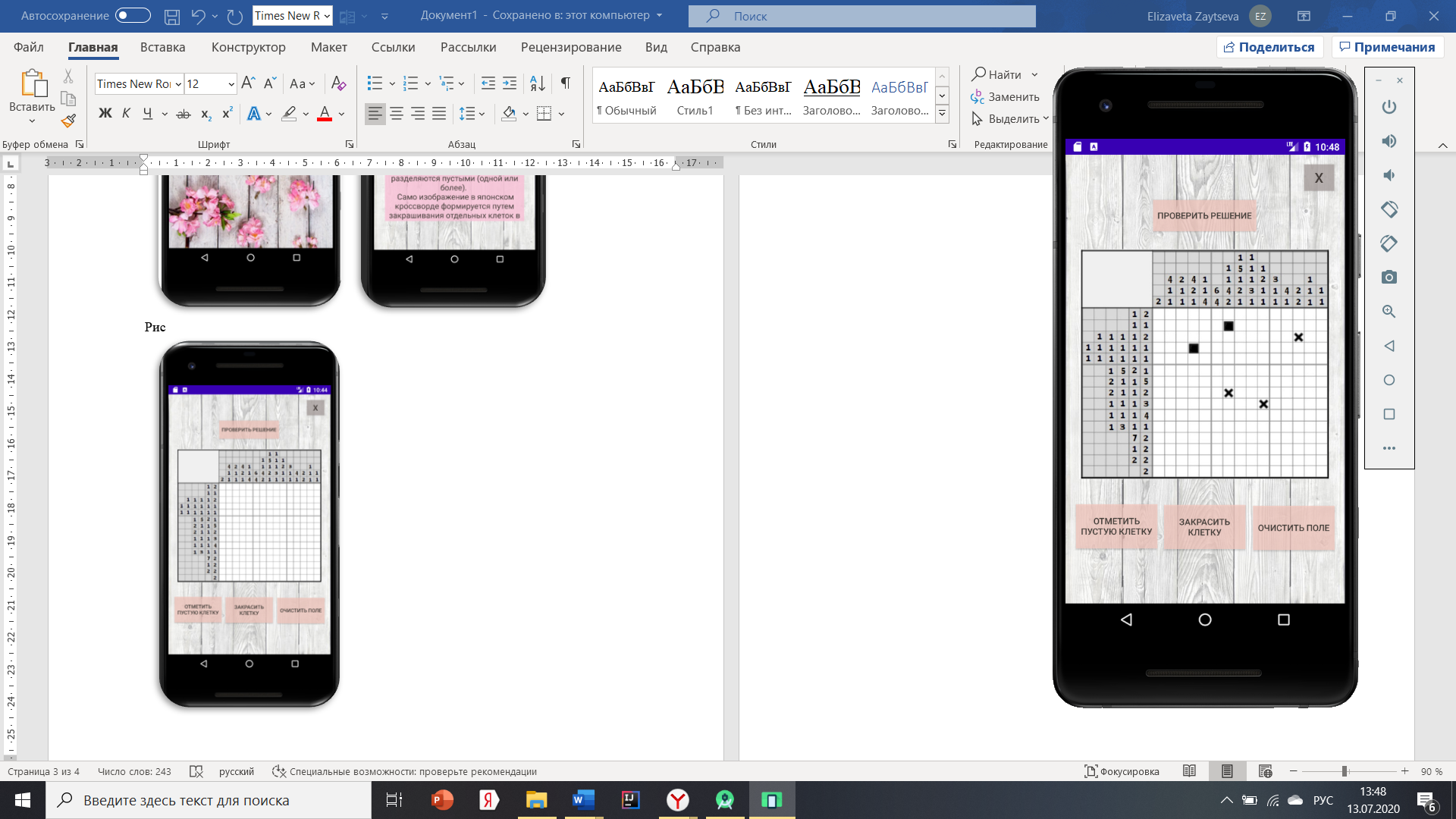
 

Рис. 3 Начало игры Рис. 4 В процессе игры

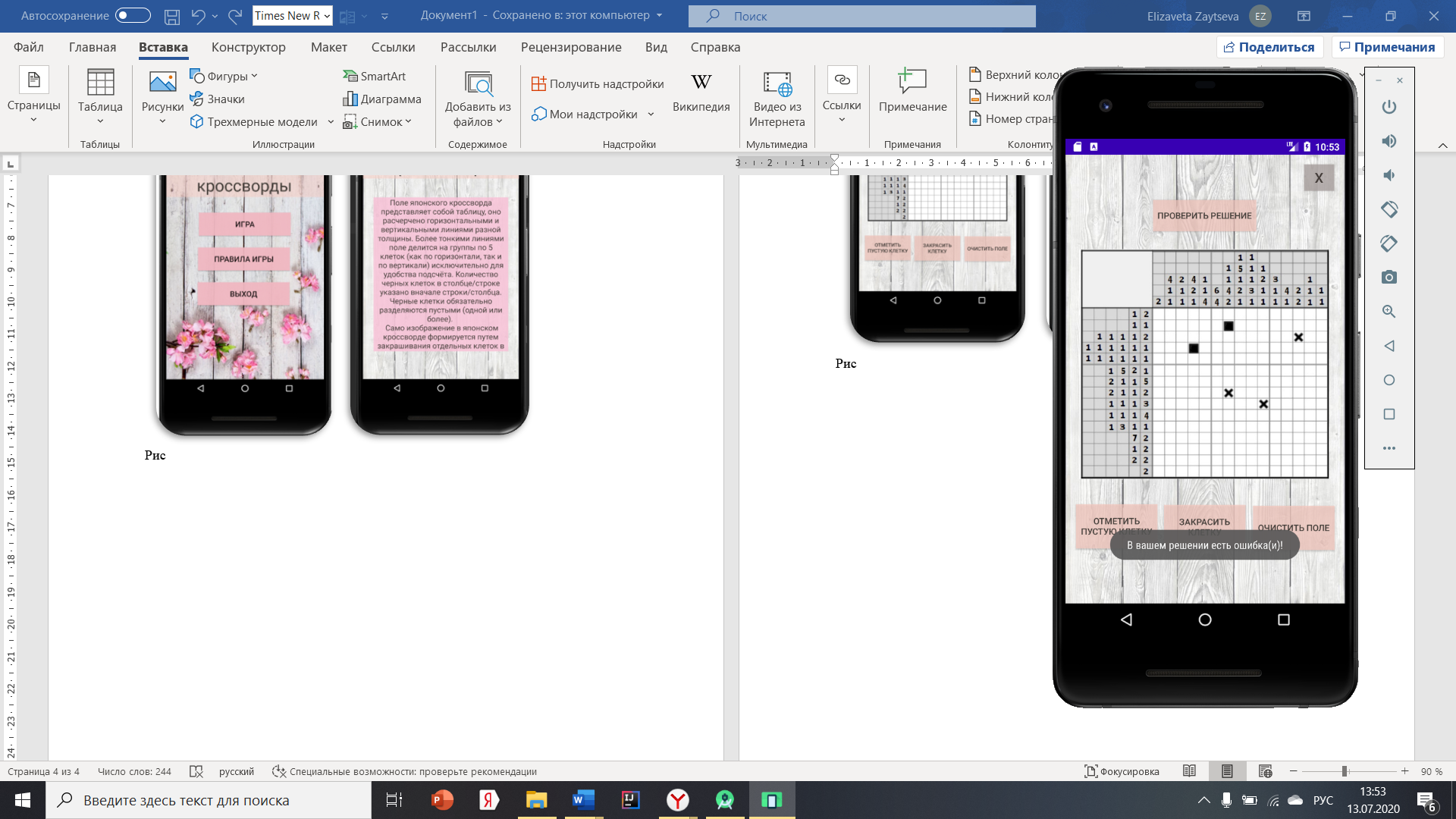
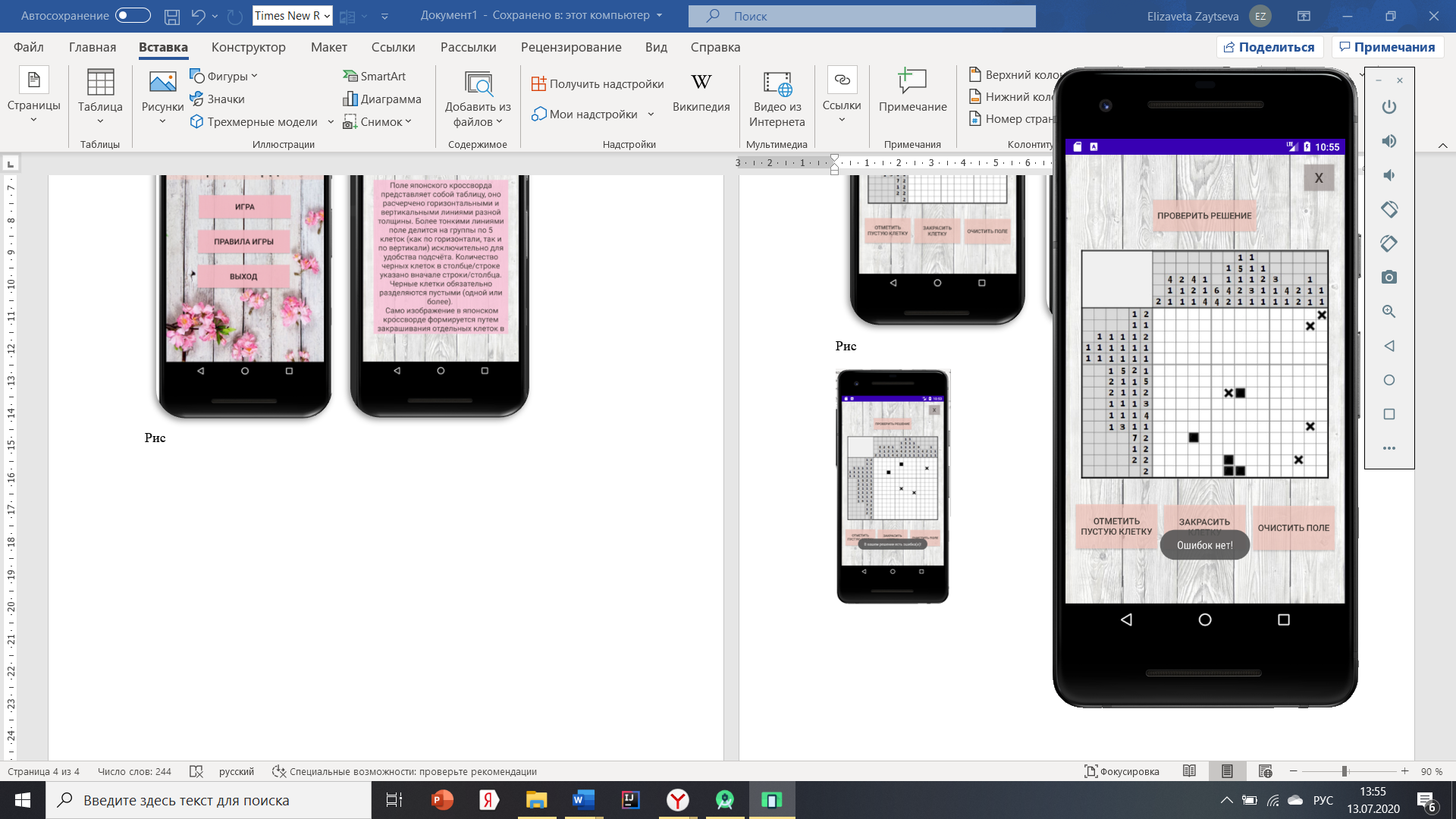
 

Рис. 5 Проверка решения Рис. 6 Проверка решения

при наличии ошибок при отсутствии ошибок

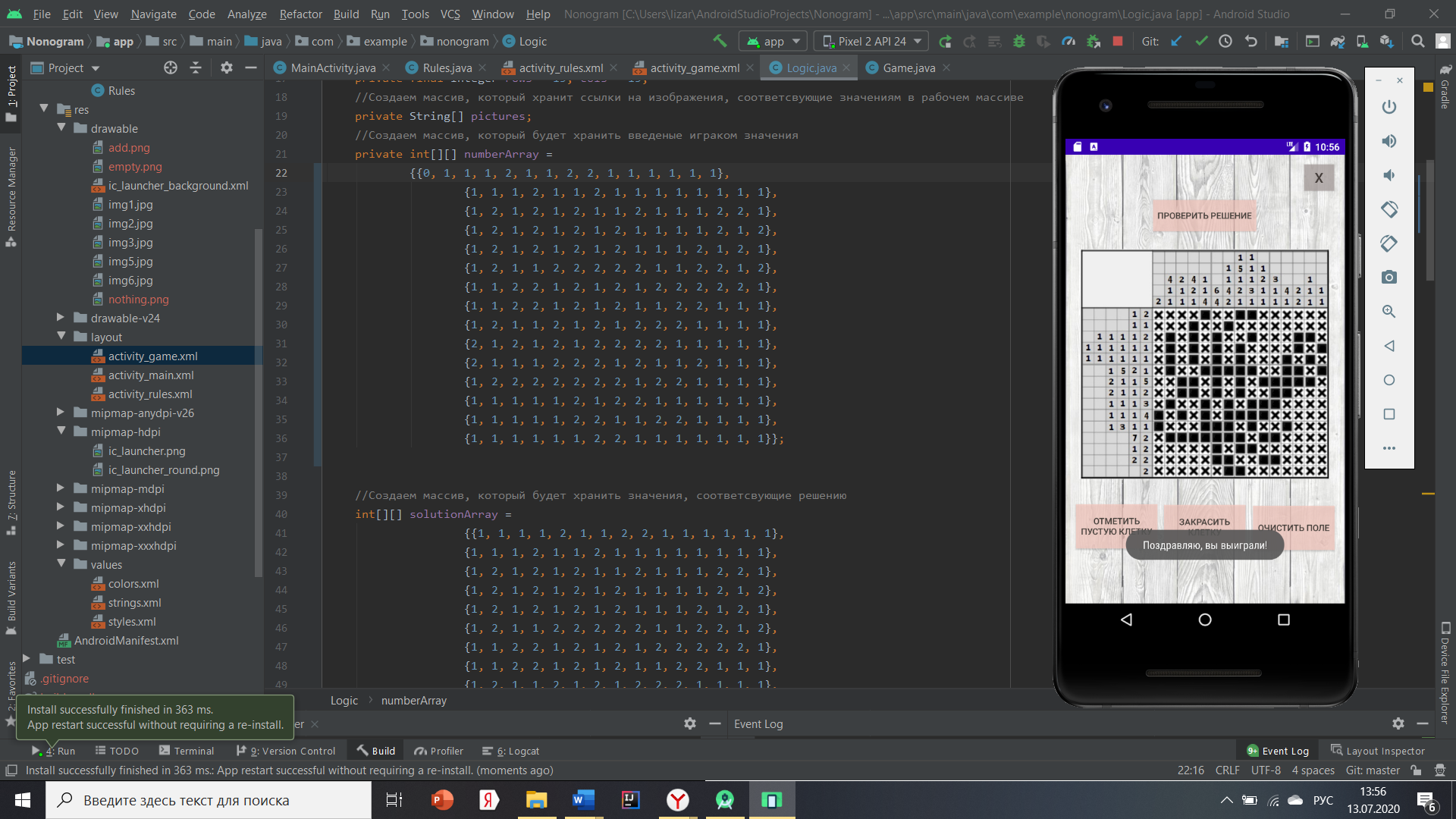


Рис. 7 Пройденная игра

# Использованные источники

stackoverflow.com

developer.alexanderklimov.ru

# Приложение

**MainActivity.java**

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.TextView;

public class MainActivity extends Activity {

//Определяем кнопки и заголовок на экране

Button rulesB;

Button playB;

Button exitB;

TextView mainTitle;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

getItems();

}

//Соотносим определенные кнопки с теми, что прописаны в файле xml

public void getItems(){

rulesB = (Button) findViewById(R.id.rules);

playB = (Button) findViewById(R.id.play);

exitB = (Button) findViewById(R.id.exit);

mainTitle = (TextView) findViewById(R.id.mainTitle);

}

//Реализуем переход в окно Rules при нажатии кнопки "правила игры"

public void gameRules(View view) {

Intent intent = new Intent(this, Rules.class);

startActivity(intent);

}

//Реализуем переход в окно Game при нажатии кнопки "играть"

public void play(View view) {

Intent intent = new Intent(this, Game.class);

startActivity(intent);

}

//Реализуем выход из приложения при нажатии кнопки "выход"

public void exit(View view) {

finishAffinity();

}

}

**Rules.java**

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

public class Rules extends Activity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_rules);

}

//Метод реализует возвращение к начальному окну

public void exitR(View view) {

Intent intent = new Intent(this, MainActivity.class);

startActivity(intent);

}

}

**Game.java**

private String selectedB = "add";

private Button addB, emptyB, clearB, solutionB;

private Button exitGameB;

private GridView gameBoard;

private Logic logic;

//Изначально выбранной кнопкой считается “add”

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_game);

//Создаем объект класса Logic

logic = new Logic(this);

//Соотносим GridView с тем, что описано в файле xml

gameBoard = (GridView) findViewById(R.id.gridView);

//Задаем кол-во столбцов в GridView

gameBoard.setNumColumns(15);

gameBoard.setHorizontalSpacing(1);

gameBoard.setVerticalSpacing(1);

//Определяем, что Grid View незаблокированный элемент

gameBoard.setEnabled(true);

//Подключаем адаптер к классу Logic

gameBoard.setAdapter(logic);

//Подключаем метод, который будет отслеживать нажатие на элементы GridView

gameBoard.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {

//Используем метод, изменяющий массив по выбранной кнопке и позиции в GridView

logic.setNumber(i, selectedB);

//Проверяем, выиграл ли игрок на данном шаге

if (logic.isWin()) {

Toast.makeText(getApplicationContext(), "Поздравляю, вы выиграли!", Toast.LENGTH\_LONG).show();

}

}

});

//Соотносим определенные кнопки с теми, что прописаны в файле xml

addB = (Button) findViewById(R.id.add);

exitGameB = (Button) findViewById(R.id.exitGame);

emptyB = (Button) findViewById(R.id.empty);

clearB = (Button) findViewById(R.id.clear);

solutionB = (Button) findViewById(R.id.solution);

//Подключаем метод, который будет отслеживать нажатие на кнопки

addB.setOnClickListener(this);

exitGameB.setOnClickListener(this);

emptyB.setOnClickListener(this);

clearB.setOnClickListener(this);

solutionB.setOnClickListener(this);

}

//Метод, обрабатывающий нажатие кнопок

@Override

public void onClick(View view) {

switch (view.getId()){

case R.id.add:

selectedB = "add";

break;

case R.id.empty:

selectedB = "empty";

break;

case R.id.exitGame:

Intent intentE = new Intent(this, MainActivity.class);

startActivity(intentE);

break;

case R.id.clear:

selectedB = "nothing";

break;

case R.id.solution:

if (logic.isRight()) {

Toast.makeText(getApplicationContext(), "Ошибок нет!", Toast.LENGTH\_LONG).show();

} else {

Toast.makeText(getApplicationContext(), "В вашем решении есть ошибка(и)!", Toast.LENGTH\_LONG).show();

}

break;

}

}

}

**Logic.java**

import android.content.Context;

import android.content.res.Resources;

import android.view.View;

import android.view.ViewGroup;

import android.widget.BaseAdapter;

import android.widget.ImageView;

import java.util.Arrays;

public class Logic extends BaseAdapter {

//Создаем элементы классов Context и Resources для реализации отрисовки

private Context mContext;

private Resources resources;

//Задаем количество строк и столбцов

private final Integer rows = 15, cols = 15;

//Создаем массив, который хранит ссылки на изображения, соответсвующие значениям в рабочем массиве

private String[] pictures;

//Создаем массив, который будет хранить введеные играком значения

private int[][] numberArray =

{{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}};

//Создаем массив, который будет хранить значения, соответствующие решению

int[][] solutionArray =

{{1, 1, 1, 1, 2, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1},

{1, 1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1},

{1, 2, 1, 2, 1, 2, 1, 1, 2, 1, 1, 1, 2, 2, 1},

{1, 2, 1, 2, 1, 2, 1, 2, 1, 1, 1, 1, 2, 1, 2},

{1, 2, 1, 2, 1, 2, 1, 2, 1, 1, 1, 2, 1, 2, 1},

{1, 2, 1, 1, 2, 2, 2, 2, 2, 1, 1, 2, 2, 1, 2},

{1, 1, 2, 2, 1, 2, 1, 2, 1, 2, 2, 2, 2, 2, 1},

{1, 1, 2, 2, 1, 2, 1, 2, 1, 1, 2, 2, 1, 1, 1},

{1, 2, 1, 1, 2, 1, 2, 1, 2, 2, 2, 1, 1, 1, 1},

{2, 1, 2, 1, 2, 1, 2, 2, 2, 2, 1, 1, 1, 1, 1},

{2, 1, 1, 1, 2, 2, 2, 1, 2, 1, 1, 2, 1, 1, 1},

{1, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 1, 1, 1, 1},

{1, 1, 1, 1, 1, 2, 1, 2, 2, 1, 1, 1, 1, 1, 1},

{1, 1, 1, 1, 1, 2, 2, 1, 1, 2, 2, 1, 1, 1, 1},

{1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1}};

//Создаем конструктор класса Logic

public Logic(Context context){

mContext = context;

pictures = new String[cols\*rows];

resources = mContext.getResources();

board();

}

//Приводим массив в начальное состояние (заполняем нулями)

public void clear(){

for (int i = 0; i < rows\*cols; i++){

numberArray[getRow(i)][getColumn(i)] = 0;

pictures[i] = "nothing";

}

}

//Создаем метод, который будет добавлять названия изображений, соответствующие значениям в массиве numberArray

private void board() {

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

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

if (numberArray[i][j] == 0) pictures[i\*15 + j] = "nothing";

if (numberArray[i][j] == 1) pictures[i\*15 + j] ="empty";

if (numberArray[i][j] == 2) pictures[i\*15 + j] = "add";

}

}

}

@Override

public int getCount() { return cols \* rows; }

@Override

public Object getItem(int i) {

return null;

}

@Override

public long getItemId(int i) {

return 0;

}

//Метод, реализующий отрисовку рабочего поля

@Override

public View getView(int i, View view, ViewGroup viewGroup) {

//создаем объект типа ImageView, нужен для для отображения изображений

ImageView imView;

//присваиваем iView значение

if(view == null) imView = new ImageView(mContext);

else imView = (ImageView)view;

//определяем индификатор по имени файла

Integer picId = resources.getIdentifier(pictures[i], "drawable", mContext.getPackageName());

//добавляем во iView то, но нашлось по полученному идентификатору

imView.setImageResource(picId);

return imView;

}

//Изменяем массив, в зависимости от выбранной кнопки

public void setNumber(int i, String selectedButton) {

pictures[i] = selectedButton;

if (selectedButton.contains("empty")) numberArray[getRow(i)][getColumn(i)] = 1;

if (selectedButton.contains("add")) numberArray[getRow(i)][getColumn(i)] = 2;

if (selectedButton.contains("nothing")) clear();

notifyDataSetChanged();

}

//По порядковому номеру определяем номер строки в рабочем массиве

public int getRow(int i){

return i/rows;

}

//По порядковому номеру определяем номер строки в рабочем массиве

public int getColumn(int i){

return i%cols;

}

//Проверка на победу, если игрок верно собрал кроссворд, то массивы должны совпасть

public boolean isWin(){

return (Arrays.deepEquals(numberArray, solutionArray));

}

//Проверка на наличие ошибок

public boolean isRight() {

int k = 0;

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

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

if ((numberArray[i][j] != 0) && (numberArray[i][j] != solutionArray[i][j])) k++;

}

}

return (k == 0);

}