Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

Институт информационных технологий и анализа данных

наименование института

ОТЧЕТ  
к лабораторной работе по дисциплине

Объектно-ориентированное программирование

«Разработка приложения с оконным графическим   
пользовательским интерфейсом»

Наименование темы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы: | ИСМб-19-1 |  |  |  | ФИО |
|  | Шифр группы |  | подпись |  | Фамилия ИО |
| Проверил |  |  |  |  | Маланова Т.В. |
|  | должность |  | подпись |  | Фамилия ИО |
| Содержание отчета на: | |  | 38 | стр. | |

Иркутск 2023 г.

Содержание

[1 Введение 3](#_Toc129570022)

[2 Проектирование графического интерфейса пользователя 4](#_Toc129570023)

[3 Проектирование классов 7](#_Toc129570024)

[3.1 Словесное описание классов 7](#_Toc129570025)

[3.2 Класс MainActivity 7](#_Toc129570026)

[3.3 Класс GuessMyNumberActivity 8](#_Toc129570027)

[3.4 Класс GuessPhoneNumberActivity 9](#_Toc129570028)

[3.5 Классы StartMyNumberDialog и StartPhoneNumberDialog 9](#_Toc129570029)

[3.6 Класс GuessGame 10](#_Toc129570030)

[4 Таблица тестов 11](#_Toc129570031)

[5 Результаты тестирования 12](#_Toc129570032)

[6 Исходный код 22](#_Toc129570033)

[6.1 Класс MainActivity 22](#_Toc129570034)

[6.2 Класс GuessMyNumberActivity 23](#_Toc129570035)

[6.3 Класс GuessPhoneNumberActivity 27](#_Toc129570036)

[6.4 Класс StartMyNumberDialog 30](#_Toc129570037)

[6.5 Класс StartPhoneNumberDialog 32](#_Toc129570038)

[6.6 Класс GuessGame 35](#_Toc129570039)

[7 Заключение 37](#_Toc129570040)

[Список литературы 38](#_Toc129570041)

1 Введение

**Цель работы**

Получение общего представления о создании Android-приложения на языке программирования Java

Получение практического опыта работы с базовыми графическими библиотеками java-android.

**Варианты индивидуальных заданий**

В качестве индивидуального задания была выбрана реализация игрового приложения "Угадай число".

Игровое-приложение: игрок должен угадать загаданное число с определенного числа попыток

Дополнительное задание: дополнительный режим: игрок должен определить необходимое число вопросов для угадывания числа.

Дополнительное задание: дополнительный режим: угадывание числа, загаданного игроком

2 Проектирование графического интерфейса пользователя

Графический интерфейс приложения состоит из трех окон-активити. Стартовым окном является activity\_main. Логика данного активити выполняется в классе MainActivity. Графический интерфейс представлен двумя TextView, предназначенных для отображения статической информации (неизменяемой от пользовательского взаимодействия) и двух кнопок, соответствующих режимам игры.

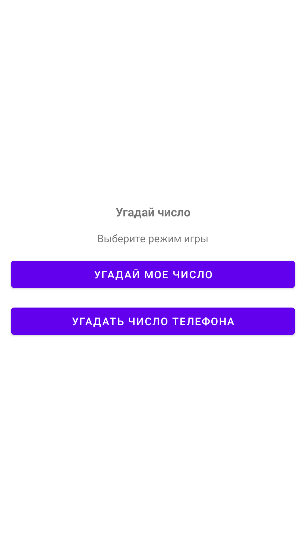


Рисунок 1 – Разметка главного окна

Взаимодействие с кнопками вызывает AlertDialog, использующий разметку из dialog\_start\_GuessGame и классы логики StartMyNumberDialog и StartPhoneNumberDialog в зависимости от выбранного режима. В данном окне пользователь задает границы значений и количество попыток для игры. Текстовые поля имеют ограничение на ввод. Максимальное значение для ввода составляет трехзначное число 999. Данное ограничение исключает сценарий выхода пользователем за диапазон значений типа Integer. Дополнительно отключена возможность ввода отрицательных значений и чисел с плавающей точкой.

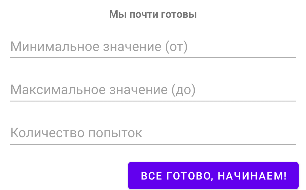


Рисунок 2 – разметка AlertDialog

Последующие окна представляют из себя набор разметки и логики для игровых режимов. Класс GuessMyNumberActivity и разметка layout activity\_my\_GuessGames представляют из себя режим игры с угадыванием телефоном загаданного пользователем числа.

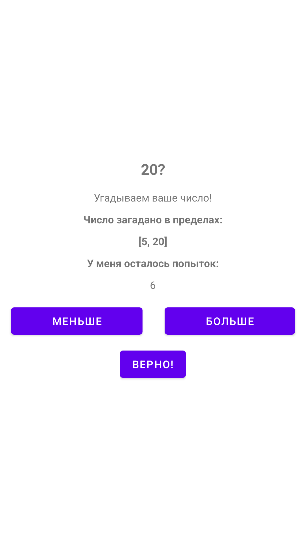


Рисунок 3 – разметка режима «Угадай мое число»

Класс GuessPhoneNumberActivityи разметка layout activity\_phone\_GuessGames представляют из себя режим игры с угадыванием пользователем загаданного телефоном числа.

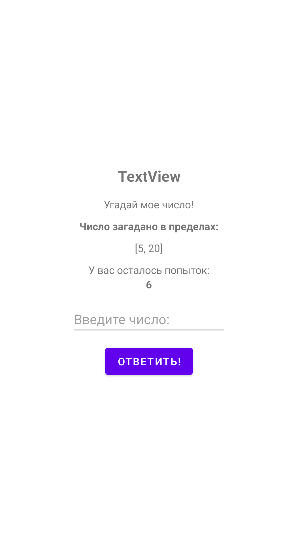


Рисунок 4 – разметка режима «Угадать число телефона»

3 Проектирование классов

3.1 Словесное описание классов

Программа состоит из 6 классов, класс MainActivity– точка входа в программу.

Описание классов:

1. MainActivity – точка входа в программу, запуск графического интерфейса, выбор игрового режима.
2. GuessGame – класс для реализации внутриигровой логики.
3. GuessMyNumberActivity – запуск графического интерфейса второго окна, первый режим игры.
4. GuessPhoneNumberActivity – запуск графического интерфейса третьего окна, второй режим игры.
5. StartMyNumberDialog – логическое представление диалогового окна для первого режима игры.
6. StartPhoneNumberDialog – логическое представление диалогового окна для второго режима игры.

3.2 Класс MainActivity

Таблица 3.1 – Таблица методов класса MainActivity

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Имя** | **Тип** | **Статический** | **Уровень доступа** | **Аргументы** | **Назначение** |
| onCreate | void | - | protected | Bundle savedInstanceState | Связывание графического интерфейса приложения и логики |
| guessMyNumBtnListener | void | - | private | - | Обработчик нажатия на кнопку guessMyNumBtn |
| guessPhoneNumBtnListener | void | - | private | - | Обработчик нажатия на кнопку guessPhoneNumBtn |
| init | void | - | private | - | Подключение элементов графического окна и логики |

3.3 Класс GuessMyNumberActivity

Таблица 3.2 – Таблица методов класса GuessMyNumberActivity

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Имя** | **Тип** | **Статический** | **Уровень доступа** | **Аргументы** | **Назначение** |
| onCreate | void | - | protected | Bundle savedInstanceState | Связывание графического интерфейса приложения и логики |
| lessBtnListener | void | - | private | - | Обработчик нажатия на кнопку lessBtn |
| bigBtnListener | void | - | private | - | Обработчик нажатия на кнопку bigBtn |
| equalsBtnListener | void | - | private | - | Обработчик нажатия на кнопку equalsBtn |
| onOptionsItemSelected | boolean | - | public | MenuItem item | Обработчик нажатий графической кнопки home для возврата в активити MainActivity |
| checkToWin | void | - | private | boolean res | Функция проверки выигрыша |
| checkAttempts | void | - | private | - | Функция уменьшения числа попыток |
| endGuessGame | void | - | private | - | Функция завершения игры и возврата в активити MainActivity |
| init | void | - | private | - | Подключение элементов графического окна и логики |

3.4 Класс GuessPhoneNumberActivity

Таблица 3.3 – Таблица методов класса GuessPhoneNumberActivity

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Имя** | **Тип** | **Статический** | **Уровень доступа** | **Аргументы** | **Назначение** |
| onCreate | void | - | protected | Bundle savedInstanceState | Связывание графического интерфейса приложения и логики |
| enterBtnListener | void | - | private | - | Обработчик нажатия на кнопку enterBtn |
| onOptionsItemSelected | boolean | - | public | MenuItem item | Обработчик нажатий графической кнопки home для возврата в активити MainActivity |
| checkEqual | void | - | private | - | Функция проверки введенного значения |
| checkAttempts | void | - | private | - | Функция уменьшения числа попыток |
| endGuessGame | void | - | private | - | Функция завершения игры и возврата в активити MainActivity |
| init | void | - | private | - | Подключение элементов графического окна и логики |

3.5 Классы StartMyNumberDialog и StartPhoneNumberDialog

Классы StartMyNumberDialog и StartPhoneNumberDialog имеют одинаковые наборы функций. Единственное различие заключается в переходах к разным режимам игры.

Таблица 3.4 – Таблица методов классов StartMyNumberDialog и StartPhoneNumberDialog

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Имя** | **Тип** | **Статический** | **Уровень доступа** | **Аргументы** | **Назначение** |
| onCreateDialog | void | - | public | Bundle savedInstanceState | Связывание графического интерфейса приложения и логики |

Продолжение таблицы 3.4.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| startGuessGameButton.setOnClickListener | void | - | private | - | Обработчик нажатия на кнопку startGuessGameButton |
| init | void | - | private | - | Подключение элементов графического окна и логики |

3.6 Класс GuessGame

Таблица 3.5 – Таблица методов класса GuessGame

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Имя** | **Тип** | **Статический** | **Уровень доступа** | **Аргументы** | **Назначение** |
| GuessGame | - | - | public | int maxCount, int minCount | Конструктор класса |
| generateNumber | void | - | public | - | Генерация числа в заданном диапозоне |
| checkNumber | int | - | public | int userNumber | Проверка числа |
| checkNumberBigger | boolean | - | public | - | Функция поиска числа в заданном диапазоне (бинарный перебор) |
| getMidCount | int | - | public | - | Getter среднего значения |
| biggerNum | void | - | public | - | Setter-манипулятор внутренней логикой созданного объекта класса, без получения внешних данных |
| lessNum | void | - | public | - | Setter-манипулятор внутренней логикой созданного объекта класса, без получения внешних данных |

4 Таблица тестов

Таблица 4.1 – таблица тестов

|  |  |  |
| --- | --- | --- |
| **№** | **Действие** | **Результат** |
| 1 | Нажатие кнопки «Угадай мое число» | Вызов диалогового окна для получения параметров |
| 2 | Нажатие кнопки «Угадать число телефона» | Вызов диалогового окна для получения параметров |
| 3 | Нажатие кнопки «Все готово, начинаем!» без заполнения полей параметров | Вывод Toast-уведомления об ошибке |
| 4 | Нажатие кнопки «Все готово, начинаем!» с неполными параметрами | Вывод Toast-уведомления об ошибке |
| 5 | Нажатие кнопки «Все готово, начинаем!» с неверными параметрами (минимальное число превышает или равно максимальному) | Вывод Toast-уведомления об ошибке |
| 6 | Нажатие кнопки «Все готово, начинаем!» с полными параметрами | Переход в активити игрового режима |
| 7 | Нажатие на кнопки Больше / Меньше | Изменение числа предполагаемого значения |
| 8 | Нажатие на кнопку Равно | Вывод Toast-уведомления о конце игры, возвращение в активити MainActivity |
| 9 | Израсходование числа попыток | Вывод Toast-уведомления о конце игры, возвращение в активити MainActivity |
| 10 | Нажатие кнопки «Ответить!» без веденного числа | Вывод Toast-уведомления об ошибке |
| 11 | Нажатие кнопки «Ответить!» с веденным числом | Вывод сообщения-подсказки по результату |

5 Результаты тестирования

Результаты тестирования представлены на рисунках 3 – 10.

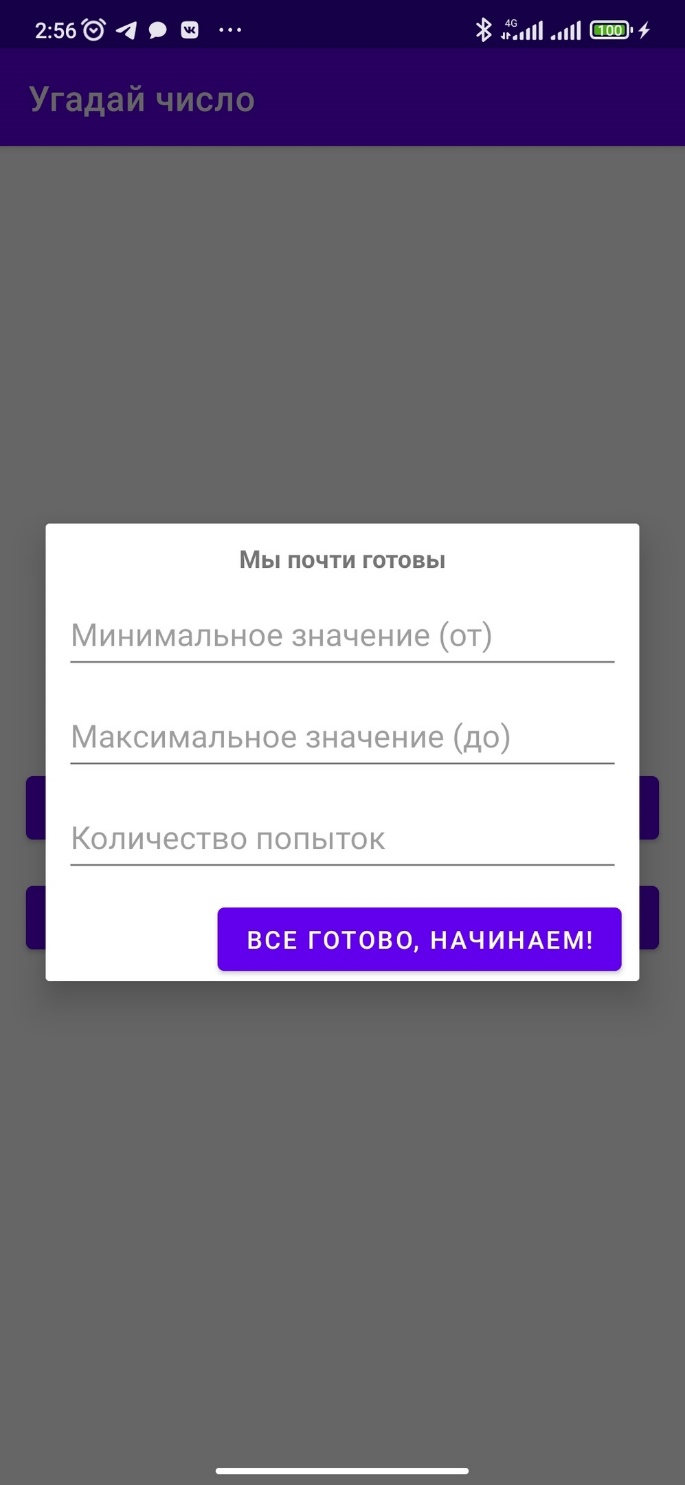


Рисунок 5 – Результат выполнения теста 1

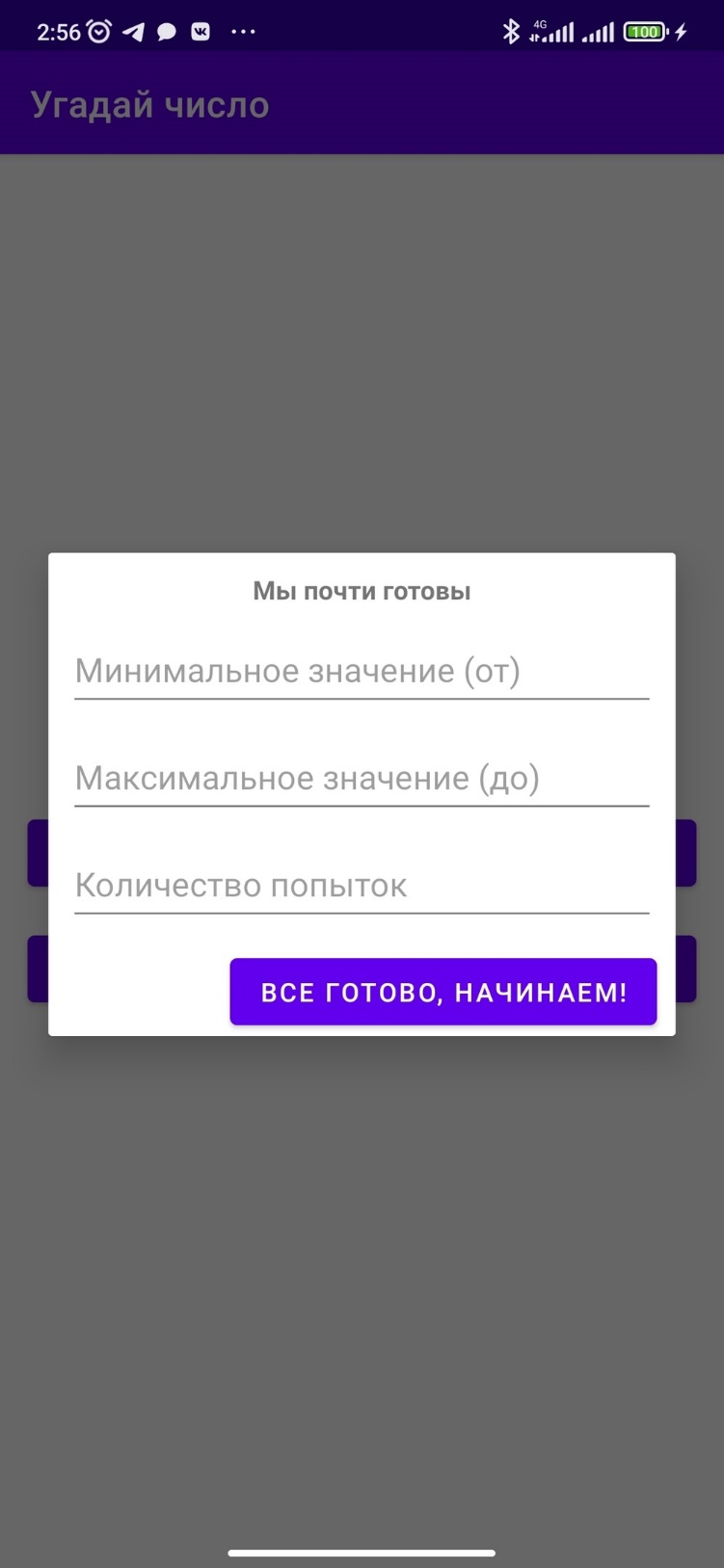


Рисунок 6 – Результат выполнения теста 2

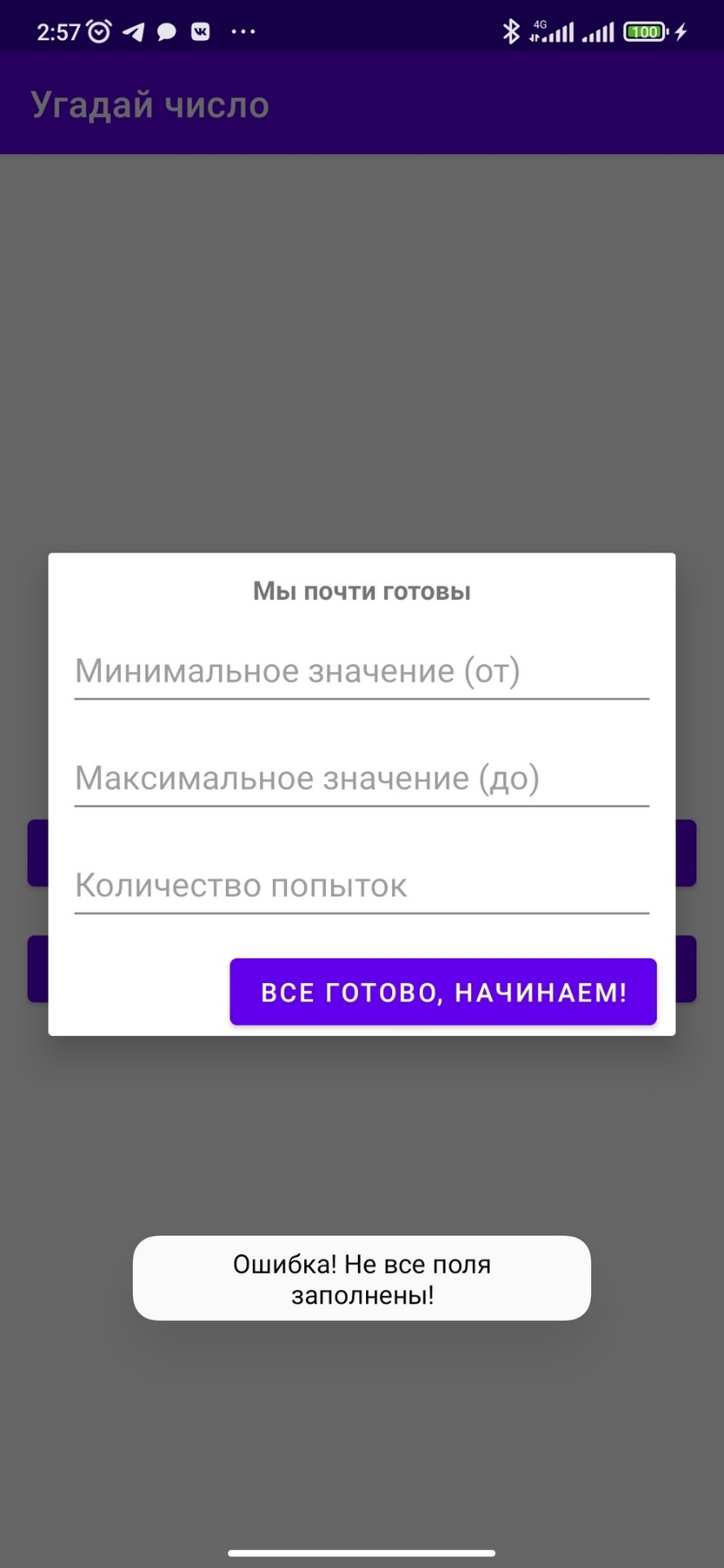


Рисунок 7 – Результат выполнения теста 3

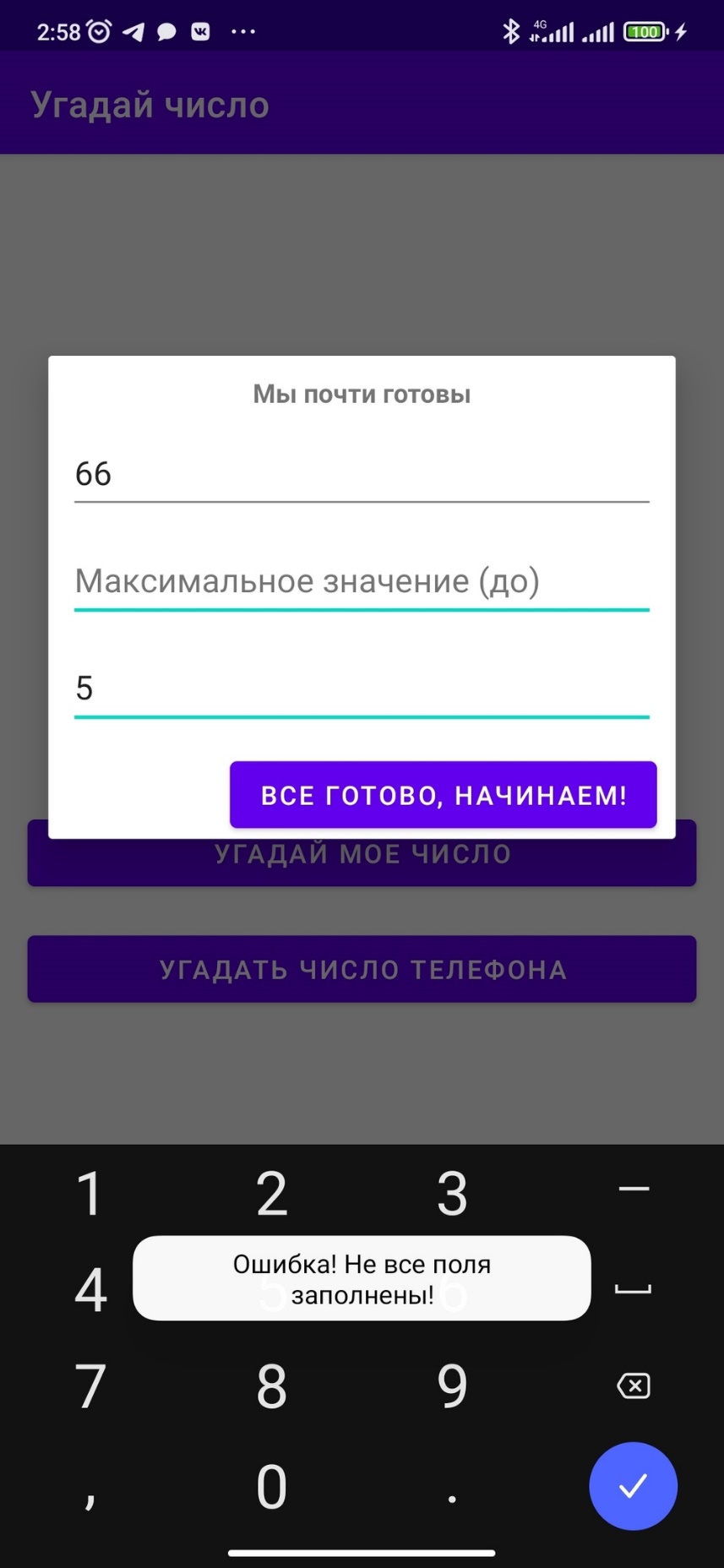


Рисунок 8 – Результат выполнения теста 4

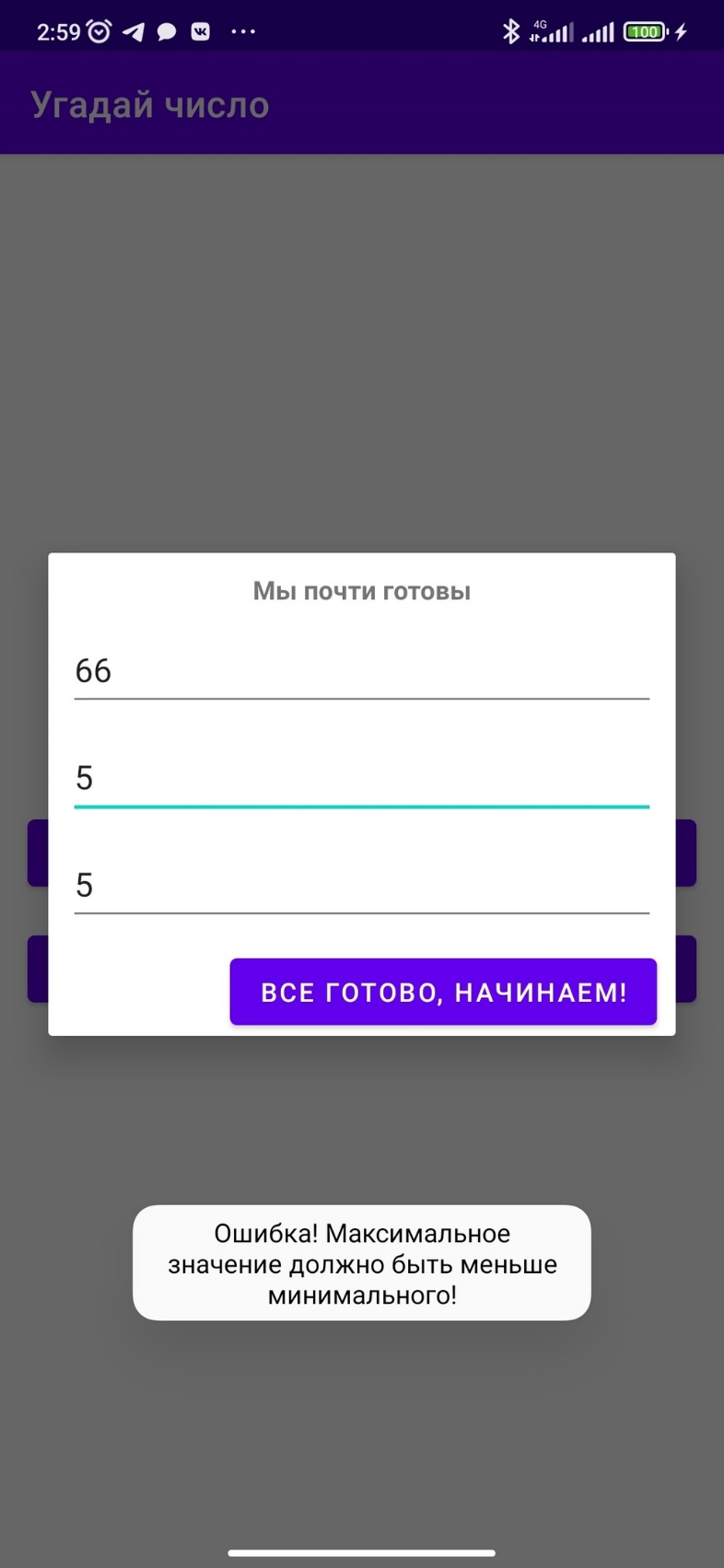


Рисунок 9 – Результат выполнения теста 5

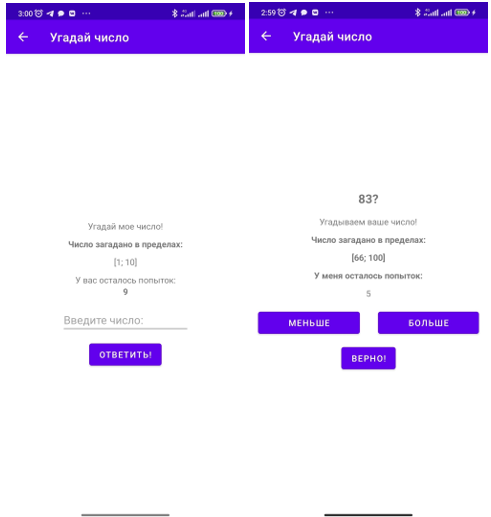


Рисунок 10 – Результат выполнения теста 6

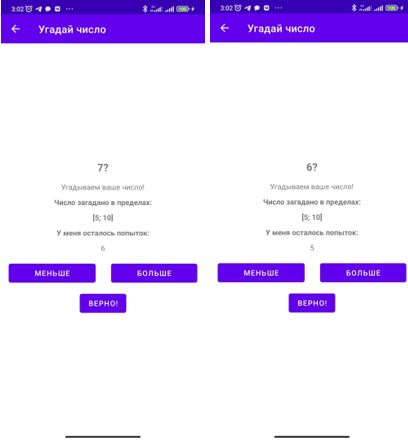


Рисунок 11 – Результат выполнения теста 7

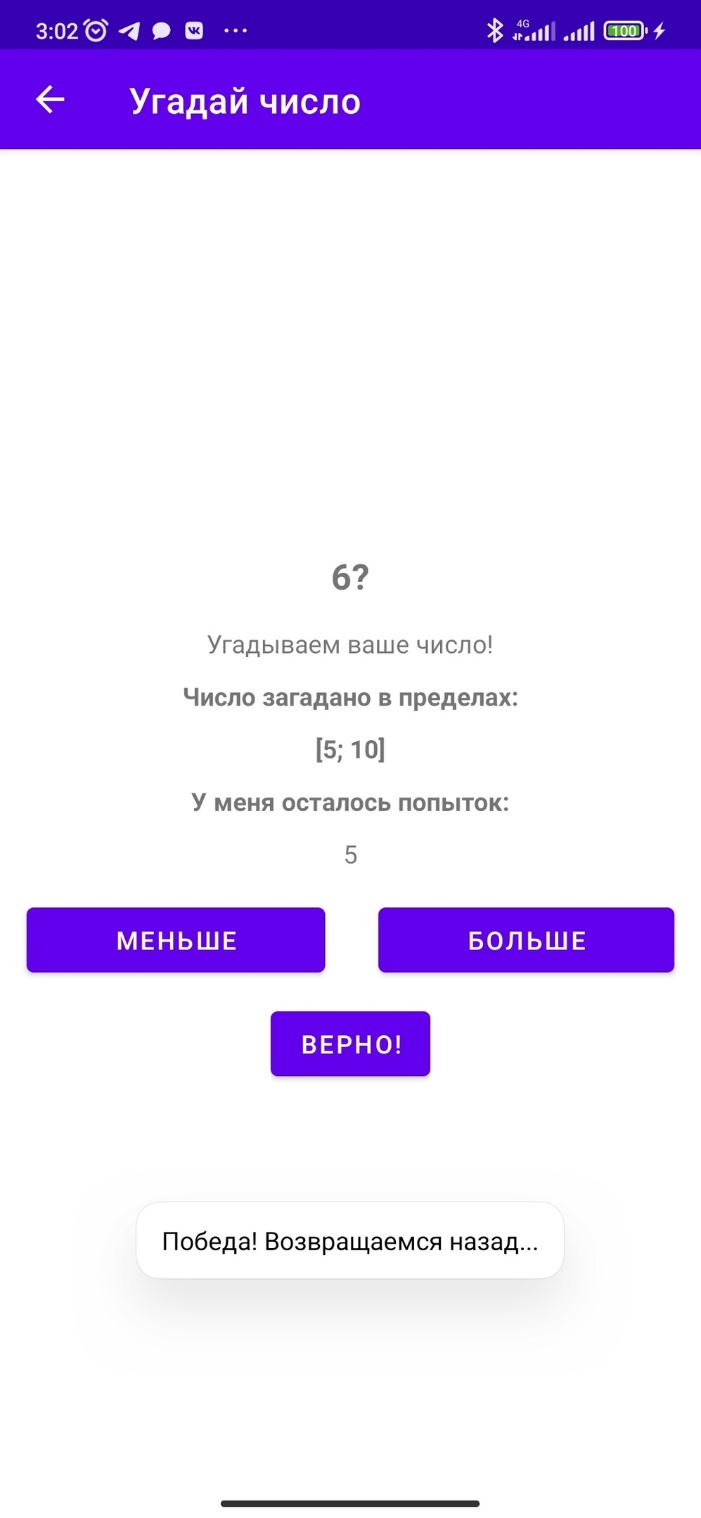


Рисунок 12 – Результат выполнения теста 8

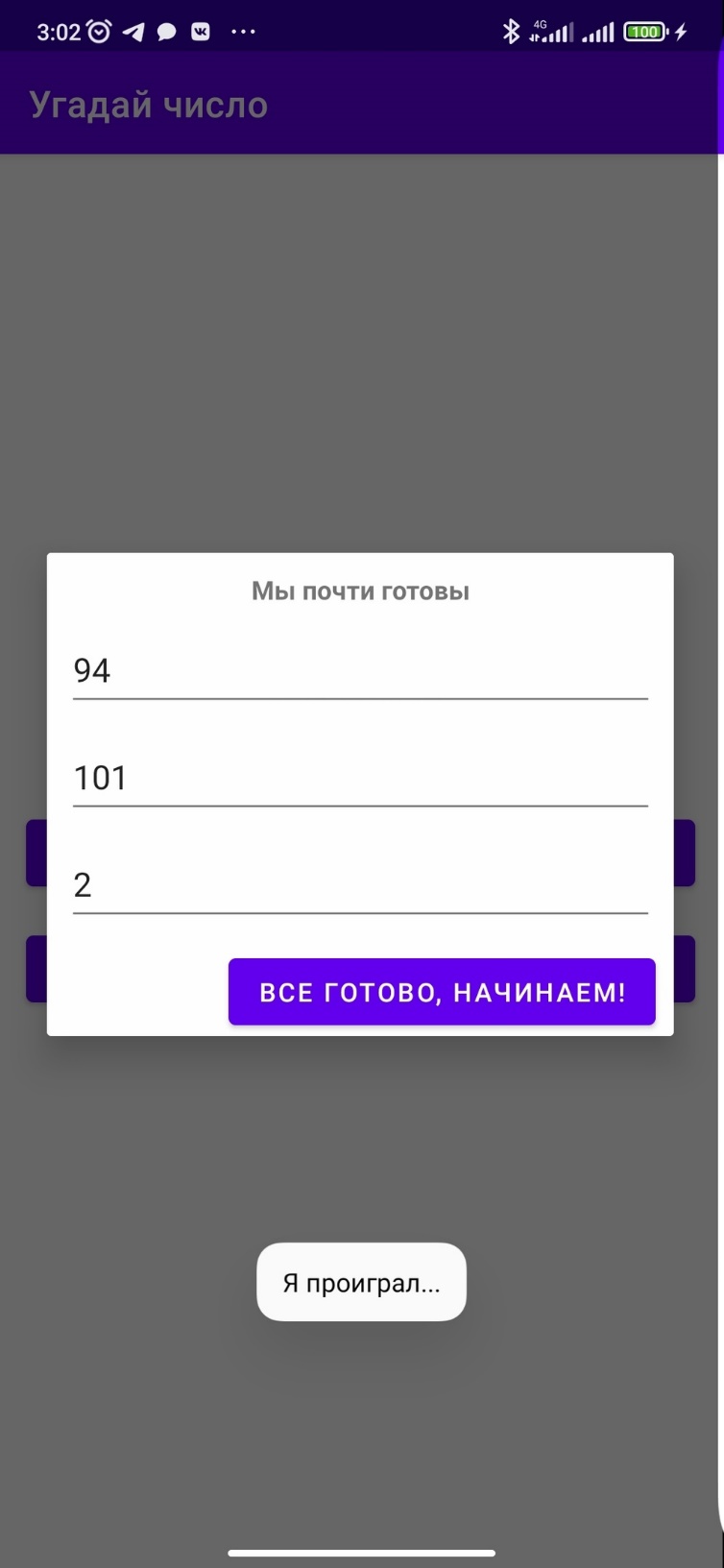


Рисунок 13 – Результат выполнения теста 9

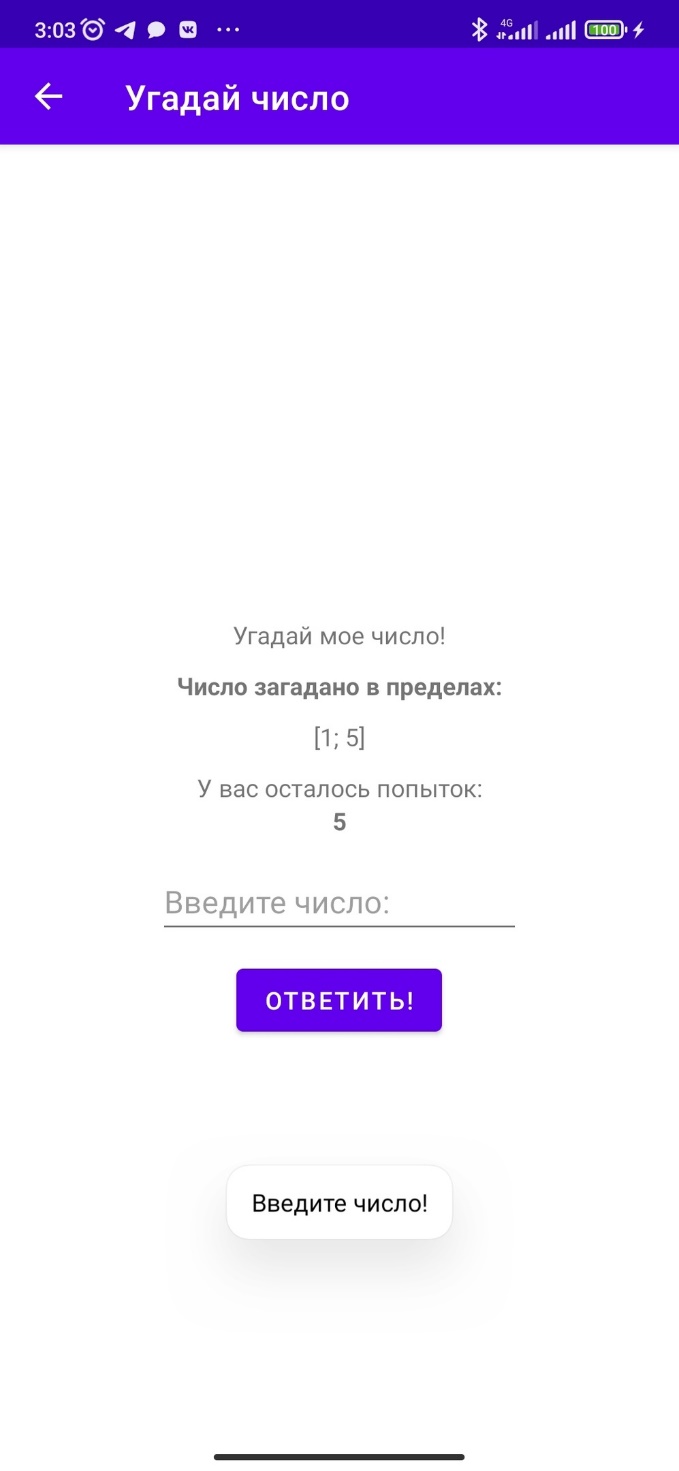


Рисунок 14 – Результат выполнения теста 10

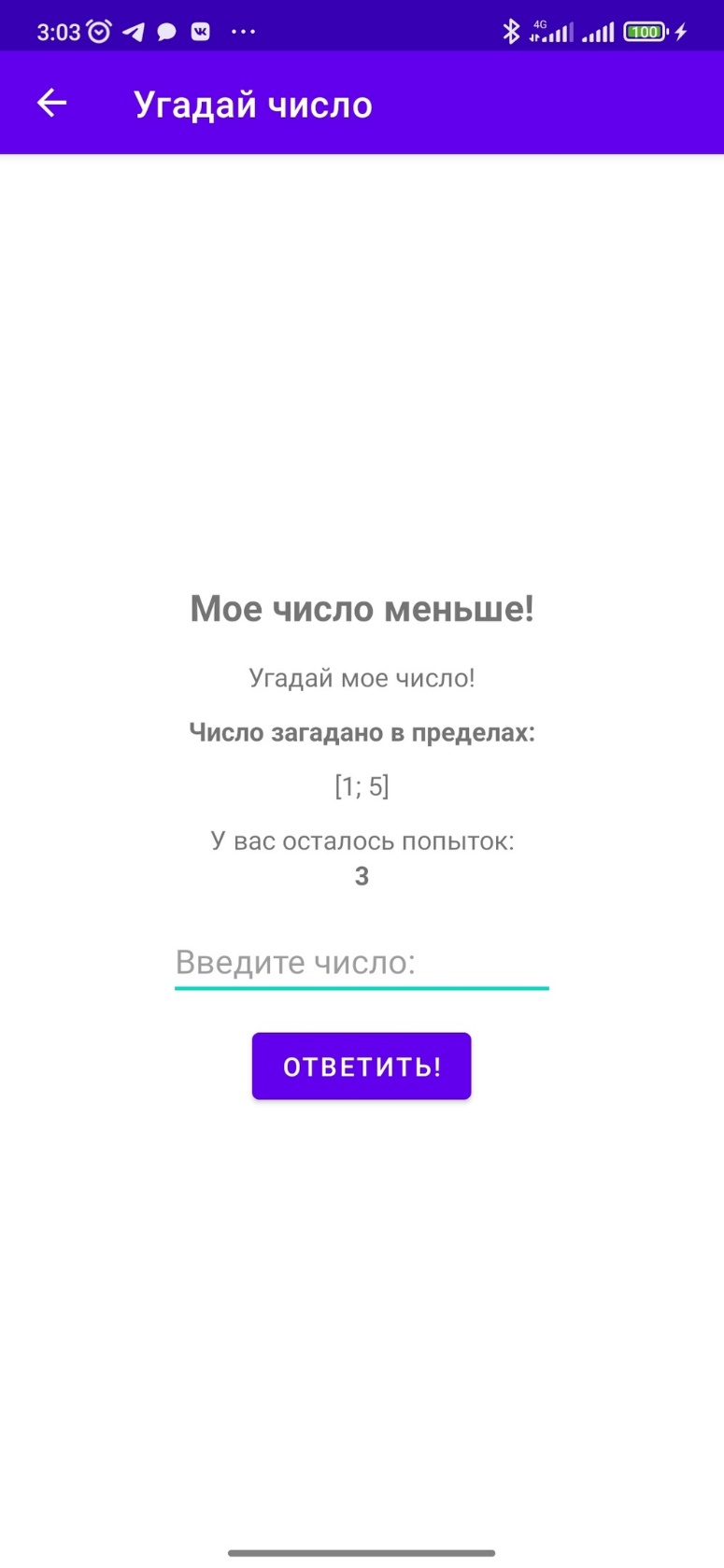


Рисунок 15 – Результат выполнения теста 11

6 Исходный код

* 1. Класс MainActivity

package com.example.guesstheumbergame.Activity;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import com.example.guesstheumbergame.Dialog.StartMyNumberDialog;

import com.example.guesstheumbergame.Dialog.StartPhoneNumberDialog;

import com.example.guesstheumbergame.R;

public class MainActivity extends AppCompatActivity {

//Иницаиализируем переменные для Активити

private Button guessMyNumBtn;

private Button guessPhoneNumBtn;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

init(); //Функиця инициализации всех компонентов

//Подвязка слушателей нажатий к кнопкам

guessMyNumBtn.setOnClickListener(guessMyNumBtnListener);

guessPhoneNumBtn.setOnClickListener(guessPhoneNumBtnListener);

}

View.OnClickListener guessMyNumBtnListener = new View.OnClickListener(){ //Слушатель кнопки запуска игры Угадай мое число

@Override

public void onClick(View v) {

//Вызов диалога

StartMyNumberDialog startGameDialog = new StartMyNumberDialog();

startGameDialog.show(getSupportFragmentManager(), "my");

}

};

View.OnClickListener guessPhoneNumBtnListener = new View.OnClickListener(){ //Слушатель кнопки запуска игры Угадай число телефона

@Override

public void onClick(View v) {

//Вызов диалога

StartPhoneNumberDialog startPhoneNumberDialog = new StartPhoneNumberDialog();

startPhoneNumberDialog.show(getSupportFragmentManager(), "phone");

}

};

private void init(){ //Функция привязки полей к объектам для связки графики и кода

guessMyNumBtn = findViewById(R.id.guessMyNumBtn);

guessPhoneNumBtn = findViewById(R.id.guessPhoneNumBtn);

}

}

6.2 Класс GuessMyNumberActivity

package com.example.guesstheumbergame.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.os.Handler;

import android.view.MenuItem;

import android.view.View;

import android.widget.Button;

import android.widget.TextView;

import android.widget.Toast;

import androidx.annotation.NonNull;

import androidx.appcompat.app.ActionBar;

import androidx.appcompat.app.AppCompatActivity;

import com.example.guesstheumbergame.Game.GuessGame;

import com.example.guesstheumbergame.R;

public class GuessMyNumberActivity extends AppCompatActivity {

//Иницаиализируем переменные для Активити

private Button lessBtn;

private Button bigBtn;

private Button equalsBtn;

private TextView phoneAttemptsText;

private TextView limitPhoneText;

private TextView numText;

private GuessGame guessGame; //Инициализируем класс игры

private int attempts; //счетчик

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_my\_game);

ActionBar actionBar = getSupportActionBar(); //Инициализируем Action Bar (фиолетовая полоска сверху)

actionBar.setDisplayHomeAsUpEnabled(true); //Отображаем на ней стрелку, которая будет вести на главную Активити

init(); //Функиця инициализации всех компонентов

Bundle arguments = getIntent().getExtras(); //Получаем из главной Активити переданные данные

//Распихиваем данные по переменным

String min = arguments.get("minCount").toString();

String max = arguments.get("maxCount").toString();

attempts = Integer.parseInt(arguments.get("attempts").toString());

phoneAttemptsText.setText("[" + min + "; " + max + "]"); //выводим диапозон значений чисел

limitPhoneText.setText("" + attempts); //Выводим число попыток

//Подвязка слушателей нажатий к кнопкам

lessBtn.setOnClickListener(lessBtnListener);

bigBtn.setOnClickListener(bigBtnListener);

equalsBtn.setOnClickListener(equalsBtnListener);

guessGame = new GuessGame(Integer.parseInt(max), Integer.parseInt(min)); //Создаем объект игры и пихаем в него минимальное и максимальное значение чисел

guessGame.checkNumberBigger(); //Начинаем перебор значений

numText.setText(guessGame.getMidCount() + "?"); //Получаем из класса-игры результат выбранного числа и выводим в текстовое поле

}

View.OnClickListener lessBtnListener = new View.OnClickListener(){ //Слушатель кнопки Меньше

@Override

public void onClick(View v) { //Переопределенная функция клика

guessGame.lessNum(); //Дергаем в игре функцию Меньше

boolean res = guessGame.checkNumberBigger(); //Получаем новое число

checkToWin(res); //Вызываем функцию проверки на выигрыш

}

};

View.OnClickListener bigBtnListener = new View.OnClickListener(){ //Слушатель кнопки Больше

@Override

public void onClick(View v) { //Переопределенная функиця клика

guessGame.biggerNum(); //Дергаем в игре функцию Больше

boolean res = guessGame.checkNumberBigger(); //Получаем новое число

checkToWin(res); //Вызываем функцию проверки на выигрыш

}

};

View.OnClickListener equalsBtnListener = new View.OnClickListener(){ //Слушатель кнопки Равно

@Override

public void onClick(View v) {

endGame();

} //Переопределяем функцию клика и вызываем метод конца игры

};

@Override

public boolean onOptionsItemSelected(@NonNull MenuItem item) { //Слушатель кнопок на верхней панели Action Bar

switch (item.getItemId()) { //Перебераем свичем нажатые кнопки по id

case android.R.id.home: //Если id == home (что есть одна наша кнопка со стрелкой, то)

this.finish(); //Останавливаем эту Активити

return true;

}

return super.onOptionsItemSelected(item);

}

private void checkToWin(boolean res){ //Функция проверки на выигрыш (данный метод нужен для того,

// чтобы если пользователь намеренно не нажимает кнопку Равно, то выдать ему два результата, которые могут быть и закончить игру намеренно )

//Собираем из ресурсов res -> string строку для тоста

String partOne = getString(R.string.end\_guess1);

String partTwo = getString(R.string.end\_guess2);

if (res){ //Проверка результата на True

checkAttempts(); //Вызываем функицю уменьшения количества попыток

int temp = guessGame.getMidCount() + 1; //Вычисляем второе число

numText.setText(partOne + "\n" + guessGame.getMidCount() + "\n" + partTwo + "\n" + temp); //Выводим два числа на Активити

endGame(); //Вызываем функицю конца игры

}

else{

checkAttempts(); //Уменьшаем число попыток

numText.setText(guessGame.getMidCount() + "?"); //Обновляем на Активити число попыток

}

}

private void checkAttempts(){ //Функция уменьшения числа попыток

attempts = attempts - 1; //Уменьшаем попытки на 1

limitPhoneText.setText("" + attempts); //Выводим новое число попыток

if(attempts == 0){ //Если попытки кончались, то

Toast.makeText(this, "Я проиграл...", Toast.LENGTH\_SHORT).show(); //Выводим тост с сообщением о конце игры

this.finish(); //Закрываем эту Активити и возвращаемся на Главную

}

}

private void endGame(){ //Функция конца игры

Toast.makeText(this, getText(R.string.end\_guess\_game).toString(), Toast.LENGTH\_SHORT).show(); //Выводим тост с поздравлением

Handler handler = new Handler(); //Создаем объект класса для задержки

Runnable r = new Runnable() { //Создаем объект Рунейбл, чтобы вывести что-то после конца задержки

public void run() { //Переходим на Главную Активити после конца задержки

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

startActivity(intent);

}

};

handler.postDelayed(r, 1000); //Ждем секунду и выполняем код из Рунейбл

}

private void init(){ //Функция привязки полей к объектам для связки графики и кода

lessBtn = findViewById(R.id.lessBtn);

bigBtn = findViewById(R.id.bigBtn);

equalsBtn = findViewById(R.id.equalsBtn);

phoneAttemptsText = findViewById(R.id.phoneAttemptsText);

limitPhoneText = findViewById(R.id.limitPhoneText);

numText = findViewById(R.id.gameText);

}

}

6.3 Класс GuessPhoneNumberActivity

package com.example.guesstheumbergame.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.os.Handler;

import android.view.MenuItem;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

import androidx.annotation.NonNull;

import androidx.appcompat.app.ActionBar;

import androidx.appcompat.app.AppCompatActivity;

import com.example.guesstheumbergame.Game.GuessGame;

import com.example.guesstheumbergame.R;

public class GuessPhoneNumberActivity extends AppCompatActivity {

//Иницаиализируем переменные для Активити

private TextView myAttemptsText;

private TextView limitText;

private EditText numEditText;

private Button enterBtn;

private TextView resText;

private GuessGame guessTheGame; //Инициализируем класс игры

private int attempts; //счетчик

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_phone\_game);

ActionBar actionBar = getSupportActionBar(); //Инициализируем Action Bar (фиолетовая полоска сверху)

actionBar.setDisplayHomeAsUpEnabled(true); //Отображаем на ней стрелку, которая будет вести на главную Активити

init(); //Функиця инициализации всех компонентов

Bundle arguments = getIntent().getExtras(); //Получаем из главной Активити переданные данные

//Распихиваем данные по переменным

String min = arguments.get("minCount").toString();

String max = arguments.get("maxCount").toString();

attempts = Integer.parseInt(arguments.get("attempts").toString());

myAttemptsText.setText("[" + min + "; " + max + "]"); //выводим диапозон значений чисел

limitText.setText("" + attempts); //Выводим число попыток

guessTheGame = new GuessGame(Integer.parseInt(max), Integer.parseInt(min)); //Создаем объект игры и пихаем в него минимальное и максимальное значение чисел

guessTheGame.generateNumber(); //Начинаем перебор значений

enterBtn.setOnClickListener(enterBtnListener);//Подвязка слушателя нажатий к кнопке

}

private void checkAttempts(){ //Функция уменьшения числа попыток

attempts = attempts - 1; //Уменьшаем попытки на 1

limitText.setText("" + attempts); //Выводим новое число попыток

if(attempts == 0){ //Если попытки кончались, то

Toast.makeText(this, "Вы проиграли...", Toast.LENGTH\_SHORT).show(); //Выводим тост с сообщением о конце игры

this.finish(); //Закрываем эту Активити и возвращаемся на Главную

}

}

View.OnClickListener enterBtnListener = new View.OnClickListener(){ //Слушатель кнопки Ввода

@Override

public void onClick(View v) { //Переопределенная функиця клика

if(numEditText.getText().toString().isEmpty()){ //Проверка строки на пустоту

Toast toast = Toast.makeText(getApplicationContext(), getString(R.string.num\_error), Toast.LENGTH\_SHORT); //Вывод тоста с сообщением о пустоте

toast.show();

return;

}

else{

checkEqual(); //Вызов функции проверки введенного значения

numEditText .setText(""); //Очистка поля ввода

}

}

};

private void checkEqual() { //Функция проверки введенного значения

int res = guessTheGame.checkNumber(Integer.parseInt(numEditText.getText().toString())); //Вызов из класса-игры функции проверки введенного числа

switch (res){ //Проверка результата

case 1 : //Если res = 1

resText.setText(R.string.num\_is\_big); //Установка текста

checkAttempts(); //Вызов функции

break;

case 2 : //Если res = 2

resText.setText(R.string.num\_is\_small); //Установка текста

checkAttempts(); //Вызов функции

break;

case 0 : //Если res = 0

resText.setText(R.string.num\_is\_true); //Установка текста

endGame(); //Функция завершения игры

break;

}

}

private void endGame(){ //Функция конца игры

Toast toast = Toast.makeText(getApplicationContext(), getText(R.string.end\_guess\_game).toString(), Toast.LENGTH\_SHORT); //Выводим тост с поздравлением

toast.show();

Handler handler = new Handler(); //Создаем объект класса для задержки

Runnable r = new Runnable() { //Создаем объект Рунейбл, чтобы вывести что-то после конца задержки

public void run() { //Переходим на Главную Активити после конца задержки

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

startActivity(intent);

}

};

handler.postDelayed(r, 1000); //Ждем секунду и выполняем код из Рунейбл

}

@Override

public boolean onOptionsItemSelected(@NonNull MenuItem item) { //Слушатель кнопок на верхней панели Action Bar

switch (item.getItemId()) { //Перебераем свичем нажатые кнопки по id

case android.R.id.home: //Если id == home (что есть одна наша кнопка со стрелкой, то)

this.finish(); //Останавливаем эту Активити

return true;

}

return super.onOptionsItemSelected(item);

}

private void init(){ //Функция привязки полей к объектам для связки графики и кода

myAttemptsText = findViewById(R.id.myAttemptsText);

limitText = findViewById(R.id.limitText);

numEditText = findViewById(R.id.numEditText);

enterBtn = findViewById(R.id.enterBtn);

resText = findViewById(R.id.gamePhoneText);

}

}

6.4 Класс StartMyNumberDialog

package com.example.guesstheumbergame.Dialog;

import android.app.AlertDialog;

import android.app.Dialog;

import android.content.Intent;

import android.os.Bundle;

import android.view.LayoutInflater;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import androidx.annotation.NonNull;

import androidx.fragment.app.DialogFragment;

import com.example.guesstheumbergame.Activity.GuessMyNumberActivity;

import com.example.guesstheumbergame.R;

public class StartMyNumberDialog extends DialogFragment{

//Иницаиализируем переменные для Активити

private EditText minCountText;

private EditText maxCountText;

private EditText attemptsText;

private Button startGameButton;

@NonNull

@Override

public Dialog onCreateDialog(Bundle savedInstanceState) {

AlertDialog.Builder builder = new AlertDialog.Builder(getActivity()); //Создаем объект конструктора диалогов

LayoutInflater inflater = requireActivity().getLayoutInflater(); //Создаем инфлейтор для отображения диалога

View view = inflater.inflate(R.layout.dialog\_start\_game, null); //Привязываем верстку к диалогу

init(view); //Инициализируем элементы

startGameButton.setOnClickListener(new View.OnClickListener() { //Слушатель кнопки начала игры

public void onClick(View v) { //Переопределенный Клик

//Получение данных с элементов

String minCount = minCountText.getText().toString();

String maxCount = maxCountText.getText().toString();

String attempts = attemptsText.getText().toString();

if(minCount.isEmpty() || maxCount.isEmpty() || attempts.isEmpty()){ //Проверка на пустоту заполняемых элементов

Toast toast = Toast.makeText(getContext(), getString(R.string.error\_text),Toast.LENGTH\_LONG); //Вывод сообщения о пустоте

toast.show();

}

else {

if(Integer.parseInt(minCount) < Integer.parseInt(maxCount)){ //Если минимальное значение меньше максимального

Intent intent = new Intent(getContext(), GuessMyNumberActivity.class); //Готовим переход на игровую Активити

// указываем первым параметром ключ, а второе значение

// по ключу мы будем получать значение с Intent

intent.putExtra("minCount", minCount);

intent.putExtra("maxCount", maxCount);

intent.putExtra("attempts", attempts);

// показываем новое Activity

startActivity(intent);

}

else{ //Иначе выводим тост с предупреждением

Toast toast = Toast.makeText(getContext(), getString(R.string.num\_error\_text),Toast.LENGTH\_LONG);

toast.show();

}

}

}

});

builder.setView(view);

return builder.create();

}

private void init(View view){ //Функция привязки полей к объектам для связки графики и кода

minCountText = view.findViewById(R.id.minCountText);

maxCountText = view.findViewById(R.id.maxCountText);

attemptsText = view.findViewById(R.id.attemptsText);

startGameButton = view.findViewById(R.id.startGameButton);

}

}

6.5 Класс StartPhoneNumberDialog

package com.example.guesstheumbergame.Dialog;

import android.app.AlertDialog;

import android.app.Dialog;

import android.content.Intent;

import android.os.Bundle;

import android.view.LayoutInflater;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import androidx.annotation.NonNull;

import androidx.fragment.app.DialogFragment;

import com.example.guesstheumbergame.Activity.GuessMyNumberActivity;

import com.example.guesstheumbergame.Activity.GuessPhoneNumberActivity;

import com.example.guesstheumbergame.Game.GuessGame;

import com.example.guesstheumbergame.R;

public class StartPhoneNumberDialog extends DialogFragment {

//Иницаиализируем переменные для Активити

private EditText minCountText;

private EditText maxCountText;

private EditText attemptsText;

private Button startGameButton;

@NonNull

@Override

public Dialog onCreateDialog(Bundle savedInstanceState) { //Функиця связывания логики и разметки

AlertDialog.Builder builder = new AlertDialog.Builder(getActivity()); //Создаем объект конструктора диалогов

LayoutInflater inflater = requireActivity().getLayoutInflater(); //Создаем инфлейтор для отображения диалога

View view = inflater.inflate(R.layout.dialog\_start\_game, null); //Привязываем верстку к диалогу

init(view);

startGameButton.setOnClickListener(new View.OnClickListener() { //Слушатель кнопки начала игры

public void onClick(View v) { //Переопределенный Клик

//Получение данных с элементов

String minCount = minCountText.getText().toString();

String maxCount = maxCountText.getText().toString();

String attempts = attemptsText.getText().toString();

if(minCount.isEmpty() || maxCount.isEmpty() || attempts.isEmpty()){ //Проверка на пустоту заполняемых элементов

Toast toast = Toast.makeText(getContext(), getString(R.string.error\_text),Toast.LENGTH\_LONG); //Вывод сообщения о пустоте

toast.show();

}

else {

if(Integer.parseInt(minCount) < Integer.parseInt(maxCount)){ //Если минимальное значение меньше максимального

Intent intent = new Intent(getContext(), GuessPhoneNumberActivity.class); //Готовим переход на игровую Активити

// указываем первым параметром ключ, а второе значение

// по ключу мы будем получать значение с Intent

intent.putExtra("minCount", minCount);

intent.putExtra("maxCount", maxCount);

intent.putExtra("attempts", attempts);

// показываем новое Activity

startActivity(intent);

}

else{ //Иначе выводим тост с предупреждением

Toast toast = Toast.makeText(getContext(), getString(R.string.num\_error\_text),Toast.LENGTH\_LONG);

toast.show();

}

}

}

});

builder.setView(view);

return builder.create();

}

private void init(View view){ //Функция привязки полей к объектам для связки графики и кода

minCountText = view.findViewById(R.id.minCountText);

maxCountText = view.findViewById(R.id.maxCountText);

attemptsText = view.findViewById(R.id.attemptsText);

startGameButton = view.findViewById(R.id.startGameButton);

}

}

6.6 Класс GuessGame

package com.example.guesstheumbergame.Game;

import android.util.Log;

import com.example.guesstheumbergame.R;

import java.util.Random;

public class GuessGame {

//Набор переменных для класса

private int maxCount;

private int minCount;

private int midCount;

private int number;

/\*

Идея режима Угадай число телефона:

Мы задаем рамки и количество попыток

Рандом генерирует число в заданных рамках

Мы пытаемся угадать, в ответ получаем интовое значение, которое преобразуется в Активити в "Больше", "Меньше" или "Верно"

Счетчик попыток отслеживается на Активити, классе Игры данного функционала нет

\*/

public GuessGame(int maxCount, int minCount) { //Конструктор класса

this.maxCount = maxCount;

this.minCount = minCount;

}

public void generateNumber(){ //Генерация числа в заданном диапозоне

if(maxCount == minCount +1){

number = minCount;

}

else {

Random random = new Random();

number = random.nextInt(maxCount - minCount) + minCount;

System.out.println("Результат генератора: " + number);

}

}

public int checkNumber(int userNumber){ //Проверка числа

if (number > userNumber) //Если введенное пользователем число меньше заданного

return 1;

else if(number < userNumber)//Если введенное пользователем число больше заданного

return 2;

else //Если введенное пользователем число равно заданному

return 0;

}

public boolean checkNumberBigger(){ //Функция поиска числа в заданном диапазоне (бинарный перебор)

midCount = (maxCount + minCount) / 2; //Вычисляем среднее значение

String res = "";

if (midCount == maxCount){ //Если среднее равно максимальному, то

midCount = midCount - 1;

return true;

}

else if (midCount == minCount){ //Если среднее равно минимальному, то

midCount = midCount - 1;

return true;

}

else {

return false;

}

}

public int getMidCount(){

return midCount;

} //Геттер среднего значения

//Необычные сеттеры, так как они манипулируют внутренней логикой созданного объекта класса, не получая внешние данные

public void biggerNum(){

minCount = midCount;

} //Сеттер присвоения минимальному значению среднего

public void lessNum(){

maxCount = midCount;

} //Сеттер приспоения максимальному значению среднего

}

7 Заключение

В ходе выполнения лабораторной работы были получены и применены навыки объектного ориентированного программирования на языке Java для Android-приложений. Были изучены механизмы создания графического интерфейса приложения и дальнейшего подключения его к управляющим классам. Получен практический опыт работы с базовыми графическими библиотеками java-android.

Список литературы

1. Java. Экспресс-курс [электронный ресурс] // Сайт Александра Климова [сайт], URL: http://developer.alexanderklimov.ru/android/java/java.php

2. API Specification for the Java 7 SE. [официальный сайт] URL: http://docs.oracle.com/javase/7/docs/api/

3. The Java Tutorials. SE [электронный ресурс], URL: http://docs.oracle.com/javase/tutorial/index.html

4. Гради Буч, Роберт А. Максимчук, Майкл У. Энгл, Бобби Дж. Янг, Джим Коналлен, Келли А. Хьюстон. Объектно-ориентированны анализ и проек-тирование с примерами приложений. Третье издание. М.: "Вильямс", 2010.

5. Хабибуллин И.Ш. Java 7: для программистов / И. Ш. Хабибуллин. – Санкт-Петербург : БХВ–Петербург, 2014.