МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра технологий программирования

ОТЧЕТ по лабораторной 10

«Мобильное приложение. Java»

Щербачени Михаила Евгеньевича,
студентов 3 курса, 8 группы

Преподаватель
Терех Владимир Сергеевич

Минск 2024

1. Задача

Разработать мобильное приложение на Java, с использованием Android Studio. Приложение должно позволять вычислить размер в количестве байт и бит для введённого числа, а также вывести какой тип подойдет для введенного числа.

2. Установка среды

В рамках лабораторной работы была установлена IDE Android studio, вместе с Android SDK.

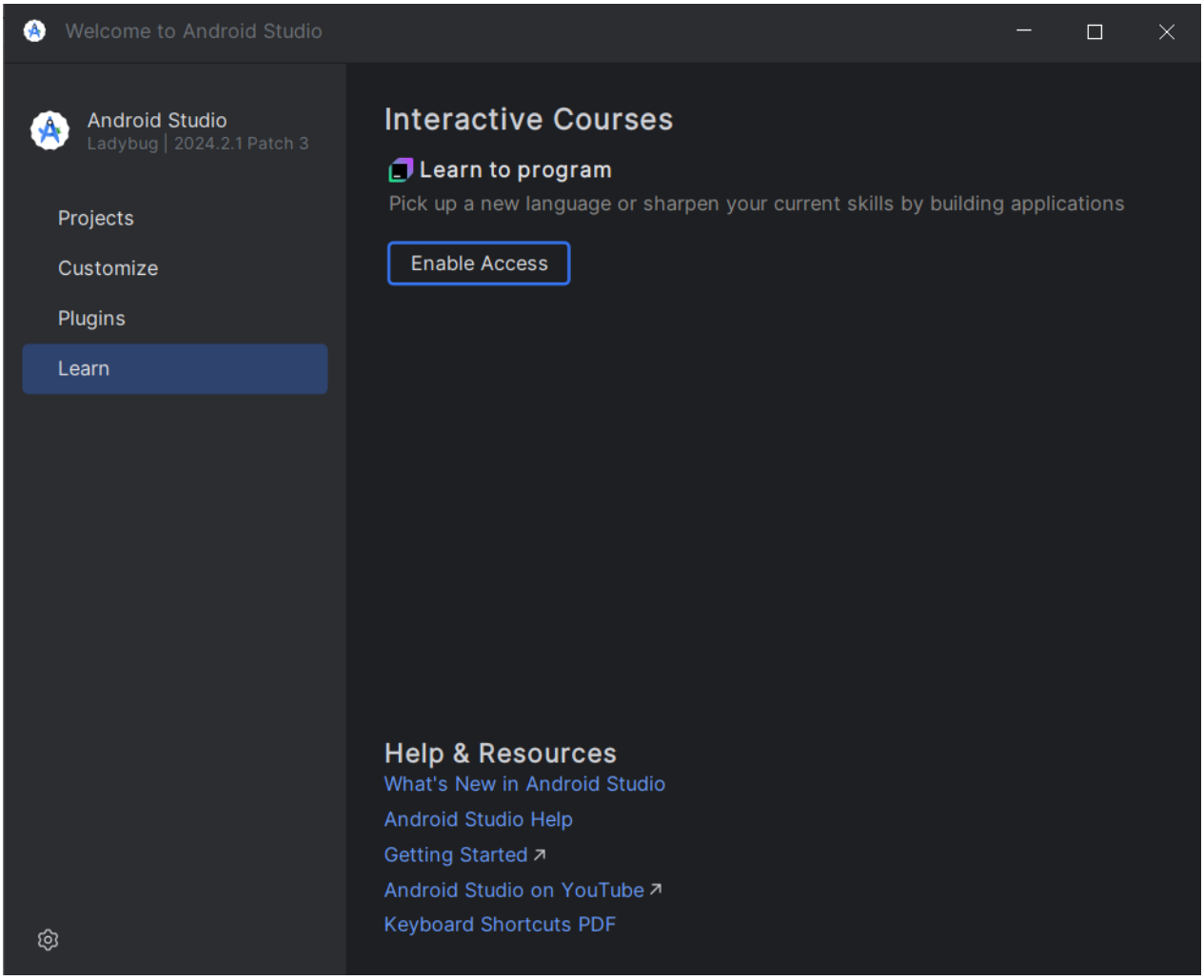

Рисунок 1. Интерфейс Android studio

3. Создание проекта

Далее был создан проект NumberSizeCalculator. В шаблоне были выбраны следующие настройки:

1) Без активности
2) Язык Java
3) SDK API 24

4) Язык сборки Groovy DSL

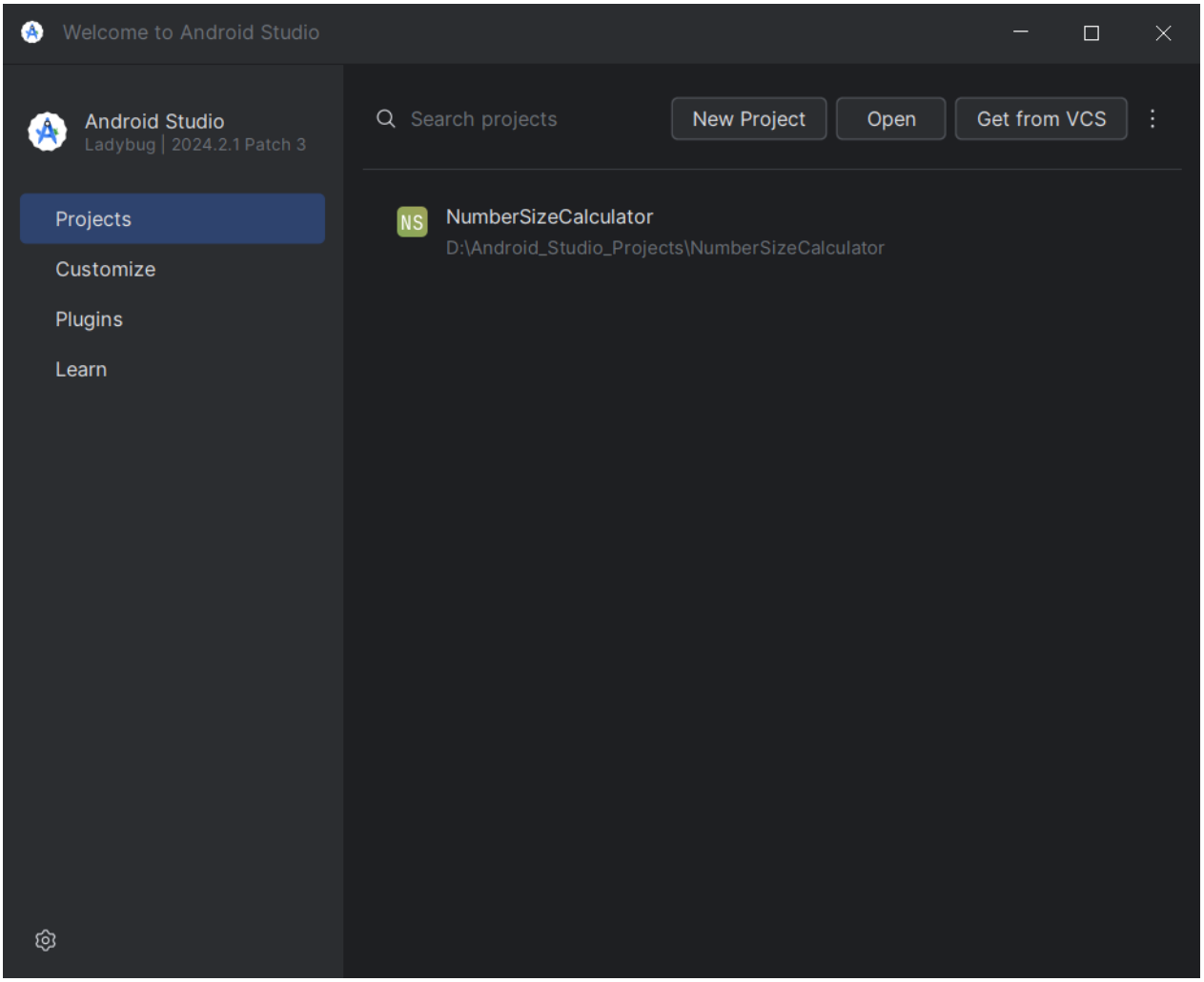

Рисунок 2. Созданный проект

4. Реализация

Сначала создаём основной класс активностей MainActivity, наследуясь от AppCompatActivity:

```
public class MainActivity extends AppCompatActivity {
    private EditText numberInput;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        numberInput = findViewById(R.id.InputNumber);
        numberInput.addTextChangedListener(new NumberWatcher(
                findViewById(R.id.BitSizeOut),
                findViewById(R.id.ByteSizeOut),
                findViewById(R.id.TypeOut)
        ));
    }
}
```

Класс основной активности

Затем необходимо прописать его как главную активность в файле манифеста:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.numbersizecalculator">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.NumberSizeCalculator"
        tools:targetApi="31">
        <activity android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Файл манифеста

Теперь необходимо создать GUI, для этого, как было видно из класса MainActivity, в ресурсах был создан layout файл с под названием activity_main.xml. В нём, с помощью средств визуальной вёрстки был создан следующий интерфейс:

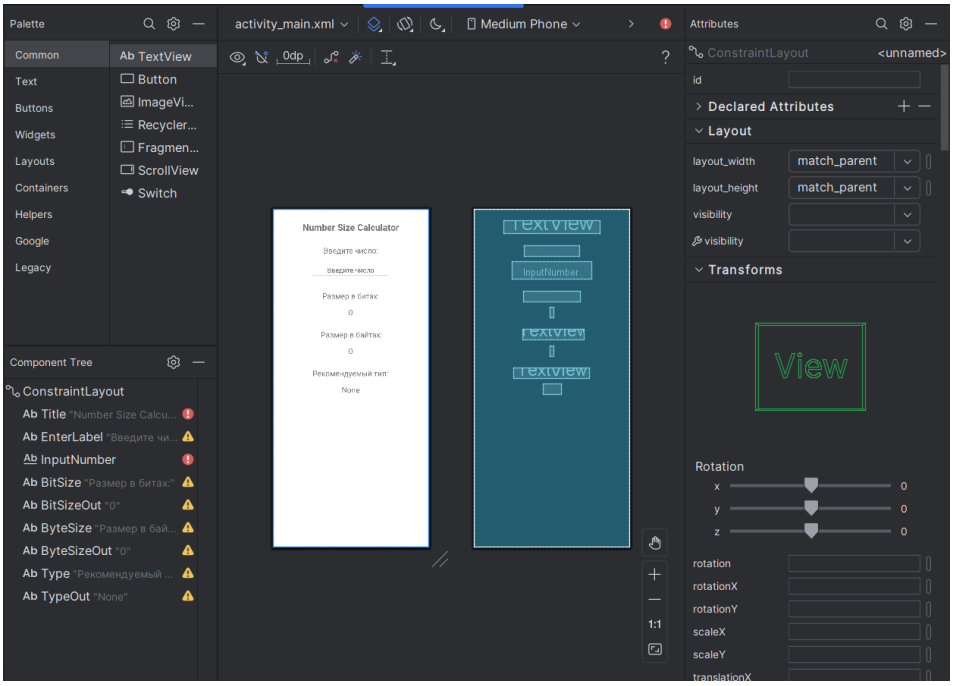

Рисунок 3. Графический интерфейс приложения

В текстовом виде это выглядит:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/Title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center_horizontal"
        android:text="Number Size Calculator"
        android:textAlignment="center"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        tools:layout_editor_absoluteY="30dp" />

    <TextView
        android:id="@+id/EnterLabel"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:text="Введите число:"
        android:textAlignment="center"
        android:textSize="20sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/Title" />

    <EditText
        android:id="@+id/InputNumber"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:ems="10"
        android:hint="Введите число"
        android:inputType="number|numberDecimal|numberSigned"
        android:textAlignment="center"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/EnterLabel" />

    <TextView
        android:id="@+id/BitSize"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:text="Размер в битах:"
        android:textAlignment="center"
        android:textSize="20sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/InputNumber" />

    <TextView
```

```
        android:id="@+id/BitSizeOut"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:text="0"
        android:textAlignment="center"
        android:textSize="20sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/BitSize" />

    <TextView
        android:id="@+id/ByteSize"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:text="Размер в байтах:"
        android:textAlignment="center"
        android:textSize="20sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.509"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/BitSizeOut" />

    <TextView
        android:id="@+id/ByteSizeOut"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:text="0"
        android:textAlignment="center"
        android:textSize="20sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ByteSize" />

    <TextView
        android:id="@+id/Type"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:text="Рекомендуемый тип:"
        android:textAlignment="center"
        android:textSize="20sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ByteSizeOut" />

    <TextView
        android:id="@+id/TypeOut"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:text="None"
        android:textAlignment="center"
        android:textSize="20sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/Type" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Статический layout приложения

Теперь создадим наблюдателя, следящего за изменениями input-формы интерфейса NumberWatcher, который будет реализовывать TextWatcher.

Определим в нём 3 абстрактных метода:

```
public class NumberWatcher implements TextWatcher {

    private TextView bitSize;
    private TextView byteSize;
    private TextView type;

    public NumberWatcher(TextView bitSize, TextView byteSize, TextView type)
{
        this.bitSize = bitSize;
        this.byteSize = byteSize;
        this.type = type;
    }

    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int
after) {
    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int
count) {
        String format = FormatChecker.checkNumberType(s.toString());
        type.setText(format);
        int bitCount;
        int byteCount;
        if (
                format.equals("Byte") ||
                format.equals("Short") ||
                format.equals("Int") ||
                format.equals("Long") ||
                format.equals("BigInteger")
        ) {
            bitCount = new BigInteger(s.toString()).bitLength();
        } else if (
                        format.equals("Float") ||
                        format.equals("Double") ||
                        format.equals("BigDecimal")
        ) {
            bitCount = new
BigDecimal(s.toString()).unscaledValue().bitLength();
        } else {
            bitCount = 0;
        }
        if (bitCount != 0) {
            byteCount = (bitCount - 1) / 8 + 1;
        } else {
            byteCount = 0;
        }
        bitSize.setText(Integer.valueOf(bitCount).toString());
        byteSize.setText(Integer.valueOf(byteCount).toString());
    }

    @Override
    public void afterTextChanged(Editable s) {
    }
}
```

Наблюдатель поля ввода числа

И напишем класс-инструмент, для нахождения типа введённого числа:

```
public class FormatChecker {

    private static boolean isByte(String input) {
        try {
            Byte.parseByte(input);
            return true;
        } catch (NumberFormatException e) {
            return false;
        }
    }

    private static boolean isShort(String input) {
        try {
            Short.parseShort(input);
            return true;
        } catch (NumberFormatException e) {
            return false;
        }
    }

    private static boolean isInt(String input) {
        try {
            Integer.parseInt(input);
            return true;
        } catch (NumberFormatException e) {
            return false;
        }
    }

    private static boolean isLong(String input) {
        try {
            Long.parseLong(input);
            return true;
        } catch (NumberFormatException e) {
            return false;
        }
    }

    private static boolean isBigInteger(String input) {
        try {
            new BigInteger(input);
            return true;
        } catch (NumberFormatException e) {
            return false;
        }
    }

    private static boolean isFloat(String input) {
        try {
            if (!Float.isInfinite(Float.parseFloat(input))) {
                return true;
            } else {
                throw new NumberFormatException("Too big value");
            }
        } catch (NumberFormatException e) {
            return false;
        }
    }

    private static boolean isDouble(String input) {
```

```
        try {
            if (!Double.isInfinite(Double.parseDouble(input))) {
                return true;
            } else {
                throw new NumberFormatException("Too big value");
            }
        } catch (NumberFormatException e) {
            return false;
        }
    }

    private static boolean isBigDecimal(String input) {
        try {
            new BigDecimal(input);
            return true;
        } catch (NumberFormatException e) {
            return false;
        }
    }

    public static String checkNumberType(String input) {
        if (isByte(input)) {
            return "Byte";
        } else if (isShort(input)) {
            return "Short";
        } else if (isInt(input)) {
            return "Int";
        } else if (isLong(input)) {
            return "Long";
        } else if (isBigInteger(input)) {
            return "BigInteger";
        } else if (isFloat(input)) {
            return "Float";
        } else if (isDouble(input)) {
            return "Double";
        } else if (isBigDecimal(input)) {
            return "BigDecimal";
        } else {
            return "None";
        }
    }
}
```

Util-класс

Получаем итоговое приложение:

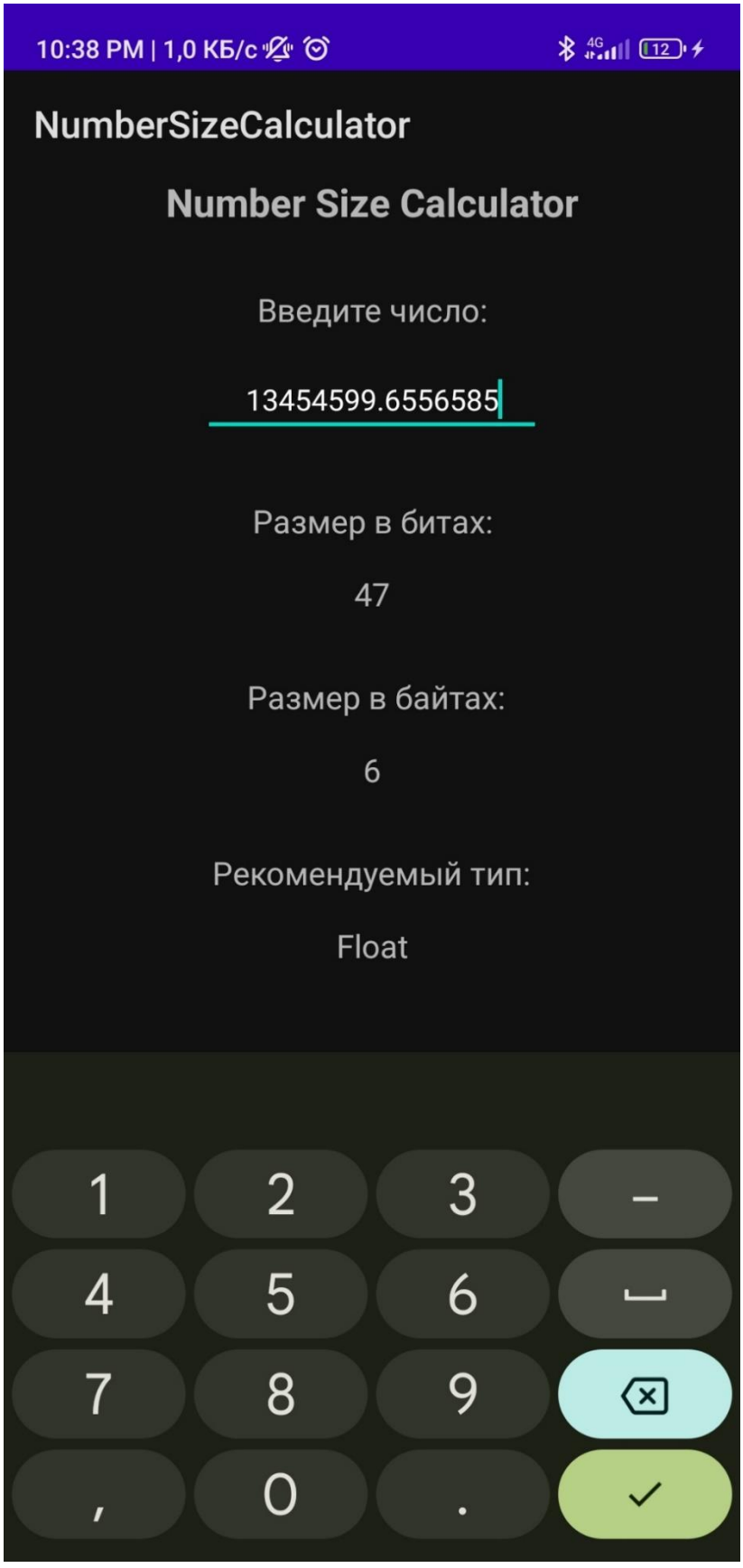

Рисунок 4. Графический интерфейс приложения