计算器的设计与实现说明文档

完成人：邹鑫波 学号：20172131018

完成时间：2019 年 9 月 29 日

一、 软件名称

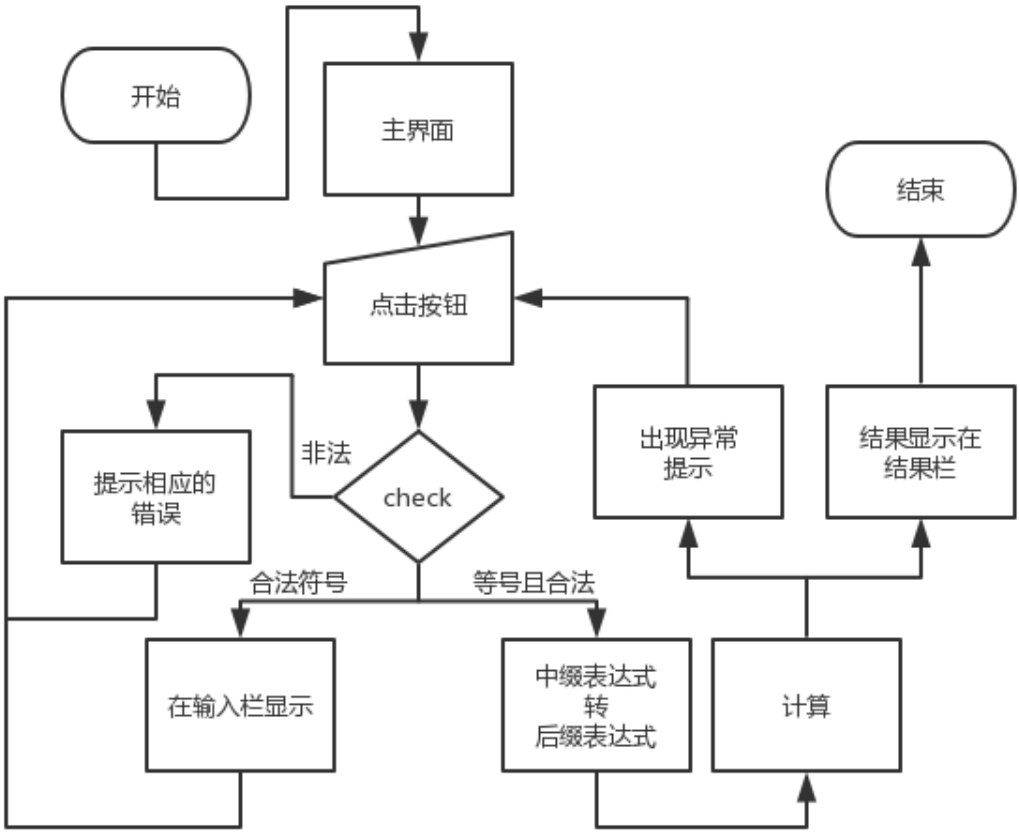
Calculator

二、 软件简介

这是一款由 Android Studio 编写的较为科学的计算器 app，它抛弃了传统手机计算器较为单调的界面，使用图片作为背景，具有较好的视觉体验，此外，它的功能也比一般手机计算器多。

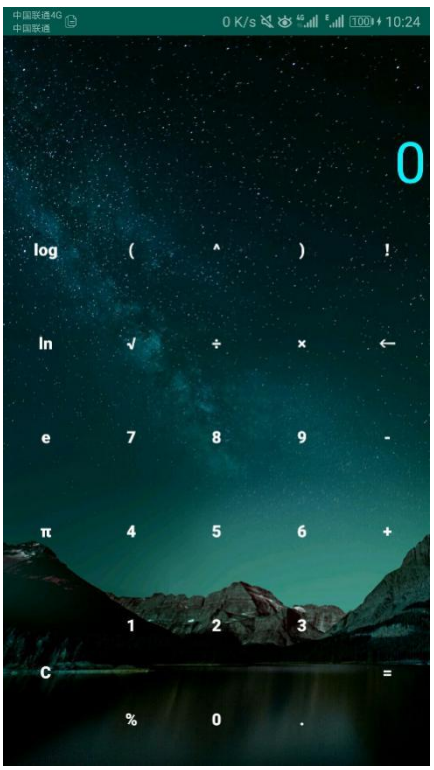
在计算器的功能实现方面，主要用到了中缀表达式转后缀表达式的算法。为了尽可能地保证输入的中缀表达式的正确性，减少转化及后续计算过程中的可能会遇到的错误，特别设计了 checkInput 类，它的作用是在按下一个 Button 时，检查已经输入的表达式，如果按下的是不符合表达式规则的 Button，则使用 Toast 提示，直至符合规则才允许输入。

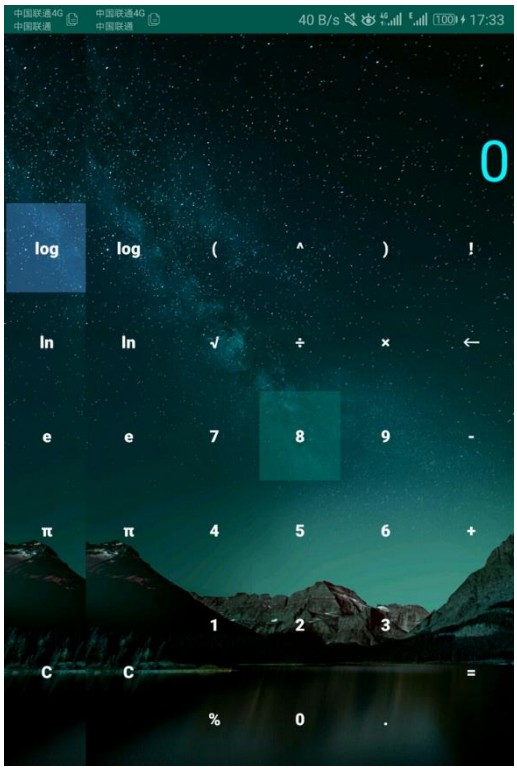
三、 软件操作流程

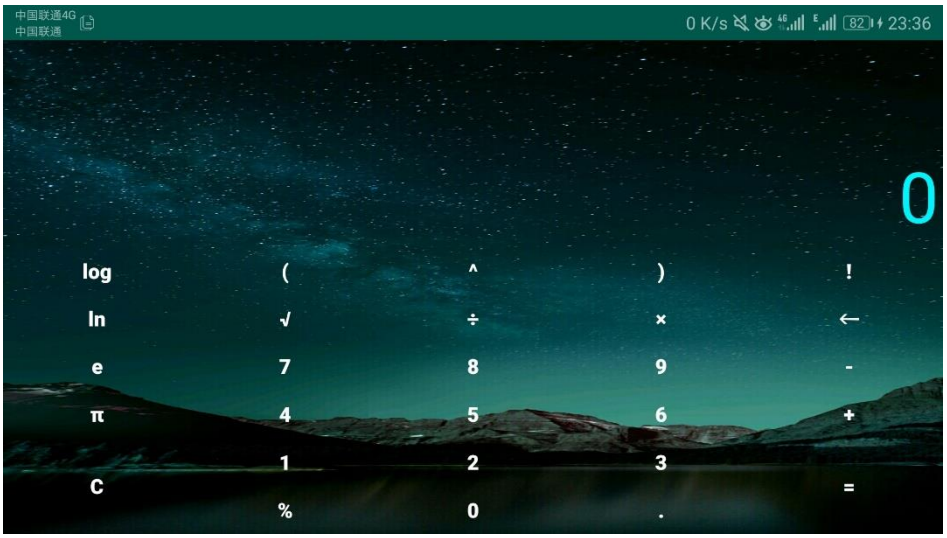

开始
主界面
结束
点击按钮
提示相应的
错误
非法
check
出现异常
提示
结果显示在
结果栏
合法符号
等号且合法
在输入栏显示
中缀表达式
转
后缀表达式
计算

四、 界面设计

（一）主界面


0
log (^) !
ln √ ÷ × ←
e 7 8 9 -
π 4 5 6 +
1 2 3
C =
% 0 .


0
log
log (^) !
ln ln √ ÷ × ←
e e 7 8 9 -
π π 4 5 6 +
1 2 3
C C =
% 0 .



（二） App 图标

五、 代码设计

（一） 界面代码

App 使用线性垂直布局，从上至下依次为：输入框(TextView)、结果框(TextView)、功能按钮(GridLayout)。背景为一张图片，按钮设为透明。由于按钮数量多，故将共同用到的属性写入 styles.xml。

具体如下：

(1)activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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"
    android:background="@drawable/back3"
    android:orientation="vertical"
    android:windowSoftInputMode="stateHidden"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textViewIn"
        style="@style/text"
        android:maxLines="3"
        android:textColor="#f1f0f3" />

    <TextView
        android:id="@+id/textViewOut"
        style="@style/text"
        android:ellipsize="end"
        android:hint="0"
```

```
        android:maxLines="2"
        android:textColor="#00F5FF"
        android:textColorHint="#00F5FF"/>

    <GridLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:rowCount="6"
        android:columnCount="5">

        <Button
            android:id="@+id/btnlog"
            style="@style/other"
            android:text="log"
            android:textAllCaps="false" />
        <Button
            android:id="@+id/btnLeft"
            style="@style/other"
            android:text="(" />
        <Button
            android:id="@+id/btnPow"
            style="@style/other"
            android:text="^" />
        <Button
            android:id="@+id/btnRight"
            style="@style/other"
            android:text=")" />
        <Button
            android:id="@+id/btnFactorial"
            style="@style/other"
            android:text="!" />

        <Button
            android:id="@+id/btnln"
            style="@style/other"
            android:text="ln"
            android:textAllCaps="false" />
        <Button
            android:id="@+id/btnSqrt"
            style="@style/other"
            android:text="√" />
        <Button
            android:id="@+id/btnDiv"
            style="@style/other"
```

```
            android:text="÷" />
        <Button
            android:id="@+id/btnMul"
            style="@style/other"
            android:text="×" />
        <Button
            android:id="@+id/btnBack"
            style="@style/other"
            android:text="←" />

        <Button
            android:id="@+id/btne"
            style="@style/other"
            android:text="e"
            android:textAllCaps="false" />
        <Button
            android:id="@+id/btn7"
            style="@style/number"
            android:text="7" />
        <Button
            android:id="@+id/btn8"
            style="@style/number"
            android:text="8" />
        <Button
            android:id="@+id/btn9"
            style="@style/number"
            android:text="9" />
        <Button
            android:id="@+id/btnSub"
            style="@style/other"
            android:text="-" />

        <Button
            android:id="@+id/btnPi"
            style="@style/other"
            android:text="π"
            android:textAllCaps="false" />
        <Button
            android:id="@+id/btn4"
            style="@style/number"
            android:text="4" />
        <Button
            android:id="@+id/btn5"
            style="@style/number"
```

```
            android:text="5" />
        <Button
            android:id="@+id/btn6"
            style="@style/number"
            android:text="6" />
        <Button
            android:id="@+id/btnAdd"
            style="@style/other"
            android:text="+" />

        <Button
            android:id="@+id/btnClear"
            style="@style/special"
            android:text="C"/>
        <Button
            android:id="@+id/btn1"
            style="@style/number"
            android:text="1" />
        <Button
            android:id="@+id/btn2"
            style="@style/number"
            android:text="2" />
        <Button
            android:id="@+id/btn3"
            style="@style/number"
            android:text="3" />
        <Button
            android:id="@+id/btnEqual"
            style="@style/special"
            android:text="=" />

        <Button
            android:id="@+id/btnPercent"
            style="@style/number"
            android:text="%" />
        <Button
            android:id="@+id/btn0"
            style="@style/number"
            android:text="0" />
        <Button
            android:id="@+id/btnPoint"
            style="@style/number"
            android:text="." />
```

```
    </GridLayout>

</LinearLayout>
```

(2)styles.xml

```
<resources xmlns:tools="http://schemas.android.com/tools">

    <!-- Base application theme. -->
    <style name="AppTheme"
parent="Theme.AppCompat.Light.DarkActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style>

    <style name="text">
        <item name="android:layout_width">match_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:layout_margin">3dp</item>
        <item name="android:background">@null</item>
        <item name="android:gravity">end</item>
        <item name="android:scrollbars">vertical</item>
        <item name="android:textSize">48sp</item>
        <item name="android:textIsSelectable">true</item>
    </style>

    <style name="special">
        <item name="android:layout_margin">2dp</item>
        <item name="android:textStyle">bold</item>
        <item name="android:textColor">@android:color/white</item>
        <item
name="android:background">@drawable/ripple_btn_special</item>
        <item name="android:layout_width">10dp</item>
        <item name="android:layout_height">10dp</item>
        <item name="android:layout_rowSpan">2</item>
        <item name="android:layout_rowWeight"
tools:targetApi="lollipop">2</item>
        <item name="android:layout_columnWeight"
tools:targetApi="lollipop">1</item>
    </style>
```

```
    <style name="number">
        <item name="android:layout_margin">2dp</item>
        <item name="android:textStyle">bold</item>
        <item name="android:textColor">@android:color/white</item>
        <item
name="android:background">@drawable/ripple_btn_number</item>
        <item name="android:layout_width">10dp</item>
        <item name="android:layout_height">10dp</item>
        <item name="android:layout_rowWeight"
tools:targetApi="lollipop">1</item>
        <item name="android:layout_columnWeight"
tools:targetApi="lollipop">1</item>
    </style>

    <style name="other">
        <item name="android:layout_margin">2dp</item>
        <item name="android:textStyle">bold</item>
        <item name="android:textColor">@android:color/white</item>
        <item
name="android:background">@drawable/ripple_btn_other</item>
        <item name="android:layout_width">10dp</item>
        <item name="android:layout_height">10dp</item>
        <item name="android:layout_rowWeight"
tools:targetApi="lollipop">1</item>
        <item name="android:layout_columnWeight"
tools:targetApi="lollipop">1</item>
    </style>

</resources>
```

(3)MainActivity.java

```
package com.calculator;

import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AppCompatActivity;

import android.os.Build;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
```

```
import java.util.Objects;

public class MainActivity extends AppCompatActivity
        implements View.OnClickListener, View.OnLongClickListener {

    private Button[] buttons = new Button[28];
    private int[] buttonIds = new int[]{ R.id.btnLeft, R.id.btnRight,
            R.id.btn0, R.id.btn1, R.id.btn2, R.id.btn3, R.id.btn4,
R.id.btn5,
            R.id.btn6, R.id.btn7, R.id.btn8, R.id.btn9, R.id.btne,
R.id.btnPi,
            R.id.btnAdd, R.id.btnSub, R.id.btnMul, R.id.btnDiv,
R.id.btnEqual,
            R.id.btnPercent, R.id.btnPoint, R.id.btnClear,
R.id.btnBack,
            R.id.btnIn, R.id.btnlog, R.id.btnPow, R.id.btnSqrt,
R.id.btnFactorial};
    private TextView input, output;
    private CheckInput checkInput;
    private String equation = "";
    private Calc calculator;

    @RequiresApi(api = Build.VERSION_CODES.KITKAT)
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Objects.requireNonNull(getSupportActionBar()).hide();
        setContentView(R.layout.activity_main);
        init();
        input = findViewById(R.id.textViewIn);
        output = findViewById(R.id.textViewOut);
        checkInput = new CheckInput();
        calculator = new Calc();
    }
    void init(){
        for(int i = 0; i < buttons.length; i++){
            buttons[i] = findViewById(buttonIds[i]);
            buttons[i].getBackground().setAlpha(96);
            buttons[i].setOnClickListener(this);
        }
        buttons[22].setOnLongClickListener(this);
    }
    public boolean onLongClick(View view){
```

```
        if(view.getId() == R.id.btnBack){
            equation = "";
            input.setText(equation);
            output.setText("");
        }
        return true;
    }
    public void onClick(View view){
        int id = view.getId();
        Button button = findViewById(id);
        String text = button.getText().toString();
        switch (id){
            case R.id.btnClear:
                equation = "";
                String res = "";
                output.setText(res);
                break;

            case R.id.btnBack:
                checkInput.setEquation(equation);
                checkInput.backSpace();
                equation = checkInput.getEquation();
                break;

            case R.id.btnEqual:
                if(!equation.isEmpty()) {
                    checkInput.setEquation(equation);
                    if(checkInput.isComplete()) {
                        checkInput.supply();
                        equation = checkInput.getEquation();
                        calculator.setIn(equation);
                        calculator.toPost();
                        calculator.doPost();
                        errorFlag();
                        res = calculator.getRes();
                        if(res.length() > 48){
                            res = res.substring(0, 49);
                            res += "...";
                        }
                        output.setText(res);
                    }
                    else {
                        Toast.makeText(this, "警告: 请输入完整的表达
式!", Toast.LENGTH_SHORT).show();
```

```
                    }
                } else {
                    Toast.makeText(this, "提示: 请输入表达式!",
Toast.LENGTH_SHORT).show();
                }
                break;

            case R.id.btn0: case R.id.btn1: case R.id.btn2: case
R.id.btn3: case R.id.btn4:
            case R.id.btn5: case R.id.btn6: case R.id.btn7: case
R.id.btn8: case R.id.btn9:
                checkInput.setEquation(equation);
                if(checkInput.checkNumber()) {
                    equation = checkInput.getEquation();
                    equation += text;
                } else {
                    Toast.makeText(this, "警告: 不可以直接接数字!",
Toast.LENGTH_SHORT).show();
                }
                break;

            case R.id.btnSub:
                checkInput.setEquation(equation);
                if(checkInput.checkSpecial()){
                    equation = checkInput.getEquation();
                    equation += text;
                } else {
                    if (!equation.isEmpty() &&
equation.charAt(equation.length() - 1) == '.')
                        Toast.makeText(this, "注意小数点",
Toast.LENGTH_SHORT).show();
                    else
                        Toast.makeText(this, "警告: 不可以是负数!",
Toast.LENGTH_SHORT).show();
                }
                break;
            case R.id.btnAdd: case R.id.btnMul: case R.id.btnDiv:
                if(text.equals("×")) text = "*";
                if(text.equals("÷")) text = "/";
                checkInput.setEquation(equation);
                if(checkInput.checkBasic()){
                    equation = checkInput.getEquation();
                    equation += text;
                } else {
```

```
                    if(!equation.isEmpty() &&
equation.charAt(equation.length() - 1) == '-')
                        Toast.makeText(this, "请退格后再输入",
Toast.LENGTH_SHORT).show();
                    else
                        Toast.makeText(this, "警告: 不合法的输入!",
Toast.LENGTH_SHORT).show();
                }
                break;

            case R.id.btnPercent:
                checkInput.setEquation(equation);
                if(checkInput.checkPercent()){
                    equation = checkInput.getEquation();
                    equation += text;
                } else{
                    Toast.makeText(this, "警告: 不合法的输入!\n 请注
意'%'的使用",
                            Toast.LENGTH_SHORT).show();
                }
                break;
            case R.id.btnPoint:
                checkInput.setEquation(equation);
                if(checkInput.checkPoint()){
                    equation = checkInput.getEquation();
                    equation += text;
                } else {
                    Toast.makeText(this, "警告: 不合法的输入!\n 请检
查小数点的使用",
                            Toast.LENGTH_SHORT).show();
                }
                break;

            case R.id.btnLeft:
                checkInput.setEquation(equation);
                if(checkInput.checkLeft()){
                    equation = checkInput.getEquation();
                    equation += text;
                } else {
                    Toast.makeText(this, "警告: 不合法的输入!\n 请注
意 '(' 的使用",
                            Toast.LENGTH_SHORT).show();
                }
                break;
```

```
            case R.id.btnRight:
                checkInput.setEquation(equation);
                if(checkInput.checkRight()){
                    equation = checkInput.getEquation();
                    equation += text;
                } else {
                    Toast.makeText(this, "警告: 不合法的输入!\n 请注
意 ‘)’ 的使用",
                            Toast.LENGTH_SHORT).show();
                }
                break;

            case R.id.btne: case R.id.btnPi:
                checkInput.setEquation(equation);
                if(checkInput.checkSpecialNum()){
                    equation = checkInput.getEquation();
                    equation += text;
                } else {
                    Toast.makeText(this, "警告: 不合法的输入!\n 请检
查表达式",
                            Toast.LENGTH_SHORT).show();
                }
                break;

            case R.id.btnPow:
                checkInput.setEquation(equation);
                if(checkInput.checkPow()){
                    equation = checkInput.getEquation();
                    text += "("; equation += text;
                } else {
                    Toast.makeText(this, "警告: 不合法的输入!\n 请注
意幂运算的使用",
                            Toast.LENGTH_SHORT).show();
                }
                break;

            case R.id.btnSqrt: case R.id.btnln: case R.id.btnlog:
                checkInput.setEquation(equation);
                if(checkInput.checkL()){
                    equation = checkInput.getEquation();
                    equation += text; equation += "(";
                } else {
                    Toast.makeText(this, "警告: 不合法的输入!",
Toast.LENGTH_SHORT).show();
```

```
                }
                break;

            case R.id.btnFactorial:
                checkInput.setEquation(equation);
                if(checkInput.checkFactorial()){
                    equation = checkInput.getEquation();
                    equation += text;
                } else {
                    Toast.makeText(this, "提示: 请确保是非负整数!",
                            Toast.LENGTH_SHORT).show();
                }
                break;
        }
        input.setText(equation);
        //input.setSelection(equation.length());
    }
    public void errorFlag(){
        if(calculator.divFlag) {
            Toast.makeText(this, "警告: 不可以除以 0!",
                    Toast.LENGTH_SHORT).show();
            calculator.divFlag = false;
        }
        if(calculator.negFlag){
            Toast.makeText(this, "警告: 不可以对 0 取负数!",
                    Toast.LENGTH_SHORT).show();
            calculator.negFlag = false;
        }
        if(calculator.sqrtFlag){
            Toast.makeText(this, "警告: 不可以对负数开根号!",
                    Toast.LENGTH_SHORT).show();
            calculator.sqrtFlag = false;
        }
        if(calculator.lnFlag){
            Toast.makeText(this, "警告: ln 的参数必须为正数!",
                    Toast.LENGTH_SHORT).show();
            calculator.lnFlag = false;
        }
        if(calculator.logFlag){
            Toast.makeText(this, "警告: log 的参数必须为正数!",
                    Toast.LENGTH_SHORT).show();
            calculator.logFlag = false;
        }
        if(calculator.facFlag){
```

```
            Toast.makeText(this, "提示: 请确保是非负整数!",
                    Toast.LENGTH_SHORT).show();
            calculator.facFlag = false;
        }
    }
}
```

（二）功能代码

1、上文提到为了提高表达式的正确性，特别设计了 CheckInput 类，它的功能包括：①检查数字、符号的位置是否合法；②判断是减号还是负号；③自动补全、删除，包括小数点前没有数字自动补 0，多次按 0 只显示一个 0，负号前自动加左括号，√、^、ln、log 后自动加左括号，按下 = 时右括号不足补全右括号等。具体如下：

(1)CheckInput.java

```
package com.calculator;

class CheckInput {
    private String equation;

    String getEquation(){ return equation; }
    void setEquation(String equation) { this.equation = equation; }

    void backSpace(){
        int len = equation.length();
        if(len == 0) return;
        equation = equation.substring(0, len - 1);
    }
    boolean isComplete(){
        char ch = equation.charAt(equation.length() - 1);
        return ch != '(' && ch != '+' && ch != '-' && ch != '*' &&
ch != '/' && ch != '.';
    }
    boolean checkNumber(){
        int len = equation.length();
        if(len == 0) return true;
        char pre = equation.charAt(len - 1);
```

```
        if(pre == ')' || pre == 'e' || pre == 'π' || pre == '%' ||
pre == '!') return false;
        if(len == 1 && pre == '0') equation = "";
        if(len > 1) {
            char prepre = equation.charAt(len - 2);
            if ((prepre == '+' || prepre == '-' || prepre == '*' ||
prepre == '/' ||
                    prepre == '(') && pre == '0')
                equation = equation.substring(0, len - 1);
        }
        return true;
    }
    boolean checkSpecial(){ // 负 or 减
        int len = equation.length();
        if(len == 0) return true; // 负
        char pre = equation.charAt(len - 1);
        if(pre == '(') {
            if(len == 1) return true;
            char prepre = equation.charAt(len - 2);
            return prepre != '√' && prepre != 'n' && prepre != 'g';
        }
        if(pre == '.') return false;
        if(pre == '+' || pre == '-' || pre == '*' || pre == '/')
            equation += "("; // 负
        return true;
    }
    boolean checkBasic(){ // + * /
        int len = equation.length();
        if(len == 0) return false;
        char pre = equation.charAt(len - 1);
        if(pre == '(' || pre == '.' || pre == '-') return false;
        if(pre == '+' || pre == '*' || pre == '/')
            equation = equation.substring(0, len - 1);
        return true;
    }
    boolean checkLeft(){
        int len = equation.length();
        if (len == 0) return true;
        char pre = equation.charAt(len - 1);
        return pre == '(' || pre == '+' || pre == '-' || pre == '*'
|| pre == '/';
        // || pre == '√'  || pre == '^' || pre == 'n' || pre == 'g';
    }
    boolean checkRight(){
```

```
        int len = equation.length();
        if(len == 0) return false;
        char pre = equation.charAt(len - 1);
        if(pre == '(') return false;
        int lNum = 0, rNum = 0;
        for(int i = len - 1; i >= 0; i--){
            char ch = equation.charAt(i);
            if(ch == '(') lNum++;
            if(ch == ')') rNum++;
            if(lNum - rNum == 1) break;
        }
        return (lNum - rNum == 1) &&
                (pre != '.' && pre != '+' && pre != '-' && pre != '*'
&& pre != '/');
    }
    boolean checkPoint(){
        int len = equation.length();
        if(len == 0) { equation += "0"; return true; }
        char pre = equation.charAt(len - 1);
        if(pre == '+' || pre == '-' || pre == '*' || pre == '/' ||
pre == '(')
            equation += "0";
        if(pre == ')' || pre == 'e' || pre == 'π' || pre == '.' ||
pre == '%' || pre == '!')
            return false;
        int pointNum = 0;
        for(int i = len - 1; i >=0; i--){
            char ch = equation.charAt(i);
            if(ch == '+' || ch == '-' || ch == '*' || ch == '/' || ch
== '(') break;
            if(ch == '.') pointNum++;
        }
        return pointNum <= 0;
    }
    boolean checkPercent(){
        int len = equation.length();
        if(len == 0) return false;
        char pre = equation.charAt(len - 1);
        if(len == 1 && pre == '0') return false;
        return pre != '(' &&  pre != '+' && pre != '-' && pre != '*'
&& pre != '/'&& pre != '.';
    }
    boolean checkSpecialNum(){ // e and π
        int len = equation.length();
```

```
        if(len == 0) return true;
        char pre = equation.charAt(len - 1);
        return pre == '(' || pre == '+' || pre == '-' || pre == '*'
|| pre == '/';
    }
    boolean checkPow(){
        int len = equation.length();
        if(len == 0) return false;
        char pre = equation.charAt(len - 1);
        if(pre == '+' || pre == '-' || pre == '*' || pre == '/' ||
                pre == '(' || pre == '.') return false;
        int sum = 0;
        for(int i = len - 1; i >= 0; i--){
            char ch = equation.charAt(i);
            if(ch == '^') sum++;
            if(ch == '+' || ch == '-' || ch == '*' || ch == '/')
break;
        }
        return sum <= 0;
    }
    boolean checkL(){
        int len = equation.length();
        if(len == 0) return true;
        char pre = equation.charAt(len - 1);
        return pre == '(' || pre == '+' || pre == '-' || pre == '*'
|| pre == '/';
    }
    boolean checkFactorial(){
        int len = equation.length();
        if(len == 0) return false;
        char pre = equation.charAt(len - 1);
        if(pre == '(' || pre == '.' || pre == '%' || pre == 'e' ||
pre == 'π' ||
                pre == '+' || pre == '-' || pre == '*' || pre == '/')
            return false;
        for(int i = len - 2; i >= 0; i--){
            char ch = equation.charAt(i);
            if(ch == '.') return false;
            if(ch == '+' || ch == '-' || ch == '*' || ch == '/')
break;
        }
        return true;
    }
    void supply(){
```

```
        int len = equation.length();
        if(len < 2) return;
        int lnum = 0, rnum = 0;
        for(int i = 0; i < len; i++){
            if(equation.charAt(i) == '(') lnum++;
            if(equation.charAt(i) == ')') rnum++;
        }
        if(lnum > rnum){
            for(int i = 0; i < lnum - rnum; i++){
                equation += ")";
            }
        }
    }
}
```

2、由于中缀表达式的存储结构是字符串，在处理中缀表达式时需提取出数字，为了减少数字与字符串之间的转换，方便后缀表达式的计算，特别设计了 Post 类,它包含 BigDecimal 类型的数字与 char 类型的运算符，若运算符为空格字符，则表示是数字，否则表示是运算符。具体如下：

(2)Post.java

```
package com.calculator;

import java.math.BigDecimal;
public class Post
{
    private BigDecimal value;
    private char operator;

    Post(BigDecimal n, char op) { value = n; operator = op; }

    BigDecimal getValue() { return value; }
    char getOperator() { return operator; }
    public void setValue(BigDecimal n) { value = n; }
    public void setOperator(char s) { operator = s; }
}
```

3、中缀转后缀使用 List<Post> post 存储后缀表达式，使用 Boolean 类型的 point 记录是否遇到小数点，使用整数栈

(Stack<Integer> left)和小数栈(Stack<Integer> right)暂存扫描到的数字，使用栈暂存符号。算法从头至尾扫描中缀表达式，当扫描到符号时，先从整数栈和小数栈中提取完整的操作数，然后插入到post，再根据符号的优先级，决定符号是入符号栈还是插入到 post。具体如下：

```
void toPost() {
    post = new ArrayList<>();
    left = new Stack<>(); right = new Stack<>();
    Stack<Character> ops = new Stack<>(); //符号栈
    for (int i = 0; i < in.length(); i++) {
        if (in.charAt(i) == '.') point = true;
        else if (isNumber(in.charAt(i))) {
            if(in.charAt(i) == 'e') { post.add(new Post(e, ' '));
continue; }
            if(in.charAt(i) == 'π') { post.add(new Post(π, ' '));
continue; }
            if (!point) left.push(in.charAt(i) - 48);
            else right.push(in.charAt(i) - 48);
        }
        else {
            if(!left.empty() || !right.empty()) {
                post.add(new Post(getSum(), ' '));
            }
            switch (in.charAt(i)){
                case '(': ops.push('('); break;
                case ')':
                    while (ops.peek() != '('){
                        post.add(new Post(valueOf(0), ops.peek()));
ops.pop();
                    }
                    ops.pop(); break;
                case '^': case '%': case '!':
                case '√': case 'n': case 'g': // n: ln, g: log
                    if(ops.empty()) ops.push(in.charAt(i));
                    else{
                        while(!ops.empty()){
                            char ch = ops.peek();
                            if(ch == '(' || ch == '+' || ch == '-' ||
ch == '*' || ch == '/')
```

```
                            { ops.push(in.charAt(i)); break; }
                            post.add(new Post(valueOf(0), ch));
ops.pop();
                        }
                        if(ops.empty()) ops.push(in.charAt(i));
                    }
                    break;
                case '*': case '/':
                    if(ops.empty()) ops.push(in.charAt(i));
                    else{
                        while(!ops.empty()){
                            char ch = ops.peek();
                            if(ch == '(' || ch == '+' || ch == '-')
                            { ops.push(in.charAt(i)); break; }
                            post.add(new Post(valueOf(0), ch));
ops.pop();
                        }
                        if(ops.empty()) ops.push(in.charAt(i));
                    }
                    break;
                case '+': case '-':
                    if(i == 0 && in.charAt(i) == '-')
{ ops.push('~'); break; }
                    else {
                        if(in.charAt(i) == '-' && in.charAt(i - 1) ==
'('){
                            if(ops.empty()) { ops.push('~'); break; }
                            if(ops.peek() == '(') { ops.push('~');
break; }
                        }
                        while (!ops.empty()) {
                            if (ops.peek() == '(')
{ ops.push(in.charAt(i)); break; }
                            post.add(new Post(valueOf(0),
ops.peek())); ops.pop();
                        }
                        if (ops.empty()) ops.push(in.charAt(i));
                    }
                    break;
            }
        }
    }
    if(!left.empty() || !right.empty()){
        post.add(new Post(getSum(), ' '));
```

```
    }
    while(!ops.empty()){
        post.add(new Post(valueOf(0), ops.peek())); ops.pop();
    }
}
```

4、计算后缀表达式时用 Boolean 类型的 divFlag(除以 0), negFlag(0 取负数), facFlag(阶乘的参数不是非负整数), sqrtFlag(负数开根号), lnFlag、logFlag(参数为负数)记录相应的错误,根据运算符结合的操作数个数是 1 个还是 2 个设计了两个 calc 函数。具体如下：

```
private BigDecimal calc1(BigDecimal n, char op){
    switch (op){
        case '~':
            if(n.equals(valueOf(0))) { negFlag = true; break; }
            return n.negate();
        case '%':
            return n.divide(valueOf(100), 16, ROUND_HALF_UP);
        case '√':
            if(n.equals(valueOf(0))) break;
            if(n.compareTo(valueOf(0)) < 0) { sqrtFlag = true;
break; }
            return sqrt(n);
        case 'n':
            if(n.compareTo(valueOf(0)) < 1) { lnFlag = true; break; }
            return valueOf(Math.log(n.doubleValue()));
        case 'g':
            if(n.compareTo(valueOf(0)) < 1) { logFlag = true;
break; }
            return valueOf(Math.log10(n.doubleValue()));
        case '!':
            if(n.equals(valueOf(0))) { return
BigDecimal.valueOf(1); }
            if(n.compareTo(valueOf(0)) < 0) { facFlag = true;
break; }
            if(n.toString().contains(".")) { facFlag = true; break; }
            return factorial(n);
    }
    return BigDecimal.valueOf(0);
}
```

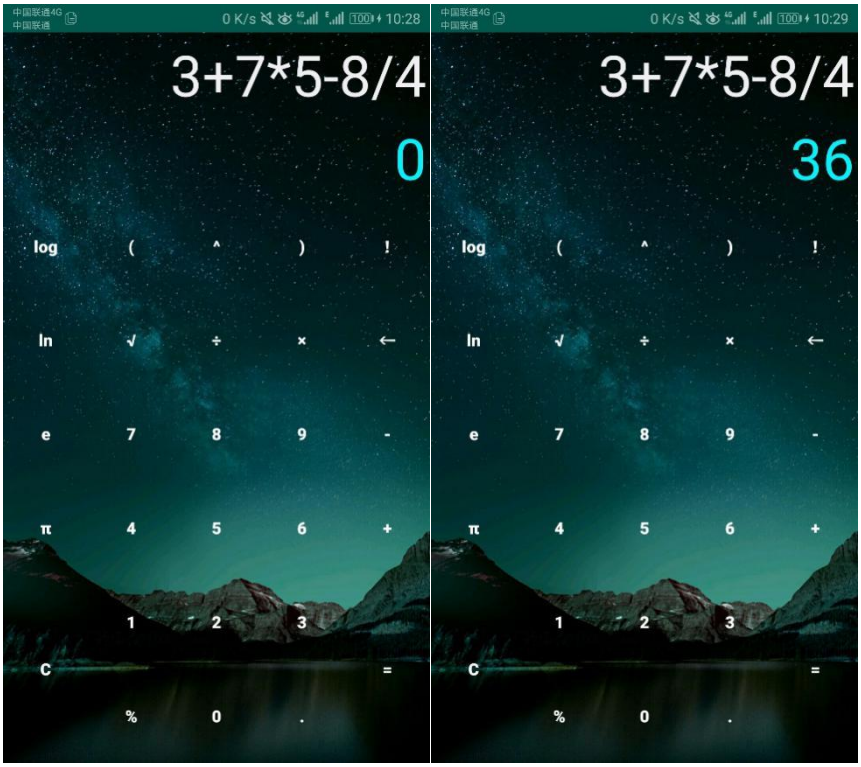
```
private BigDecimal calc2(BigDecimal l, BigDecimal r, char op){
    switch (op){
        case '+': return l.add(r);
        case '-': return l.subtract(r);
        case '*': return l.multiply(r);
        case '/':
            if(r.equals(valueOf(0))) { divFlag = true; break; }
            return l.divide(r, 16, ROUND_HALF_UP);
        case '^':
            if(r.equals(valueOf(0))) { return
BigDecimal.valueOf(1); }
            return valueOf(Math.pow(l.doubleValue(),
r.doubleValue()));
    }
    return BigDecimal.valueOf(0);
}

void doPost() {
    res = valueOf(0);
    Stack<BigDecimal> num = new Stack<>();
    BigDecimal l, r;
    for(int i = 0; i < post.size(); i++){
        if(post.get(i).getOperator() == ' ')
num.push(post.get(i).getValue());
        else{
            char ch = post.get(i).getOperator();
            switch (ch){
                case '+': case '-': case '*' : case '/': case '^':
                    if(num.isEmpty()) return;
                    r = num.peek(); num.pop();
                    if(num.isEmpty()) return;
                    l = num.peek(); num.pop();
                    res = calc2(l, r, ch);
                    if(divFlag) return;
                    break;
                case '~': case '%': case '√': case 'n': case 'g':
case '!':
                    if(!num.empty())
                    { res = calc1(num.peek(), ch); num.pop(); }
                    else return;
                    if(negFlag || sqrtFlag || lnFlag || logFlag ||
facFlag) return;
                    break;
```

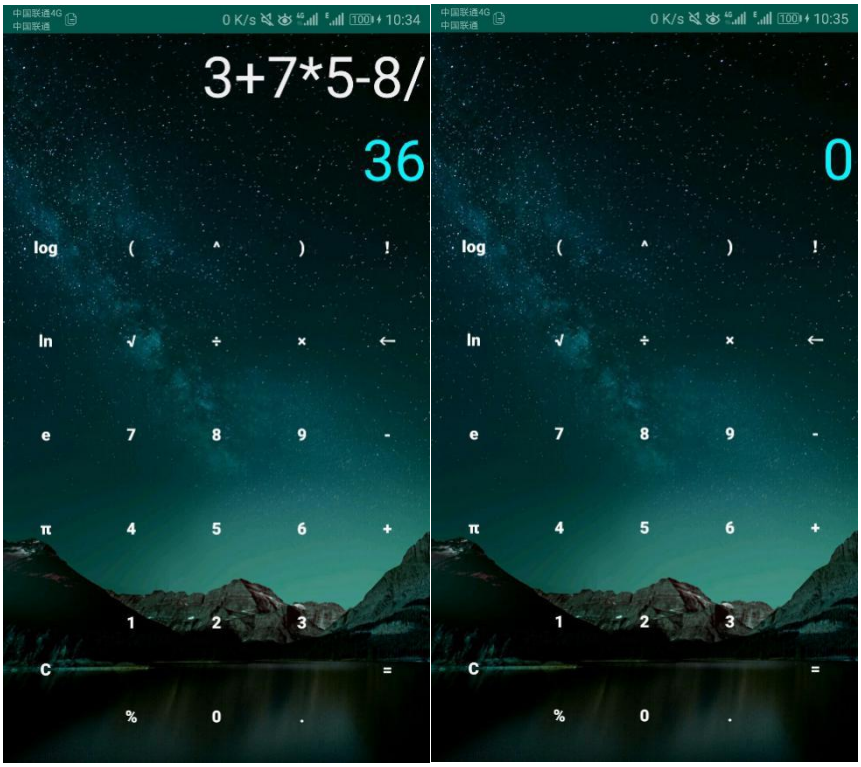
```
                }
                num.push(res);
            }
        }
        if(!post.isEmpty()) res = num.peek();
        post.clear();
    }
```

六、 运行测试

（一）基本计算

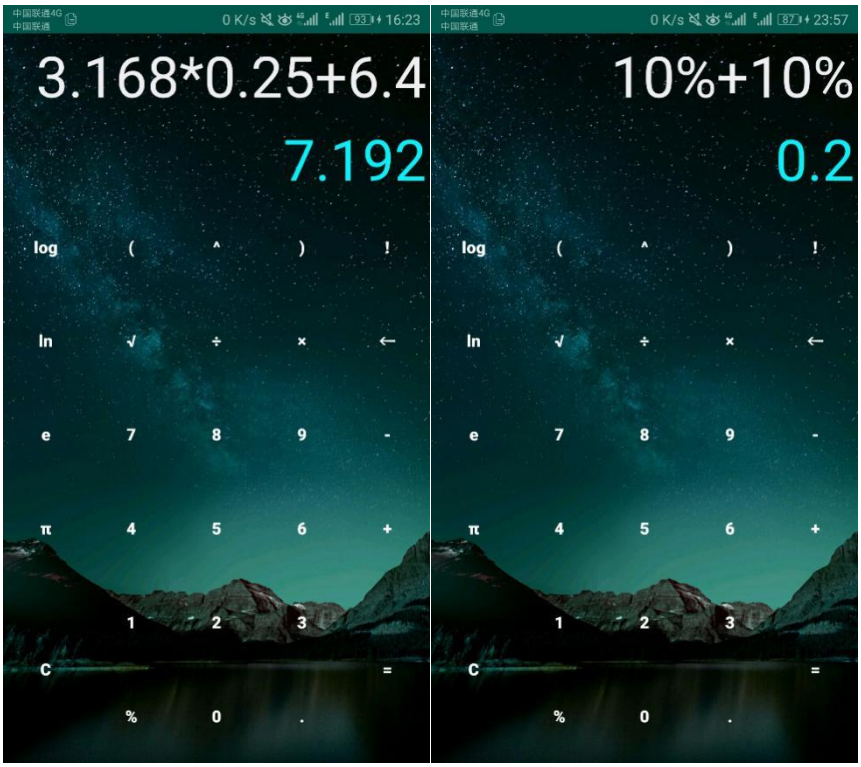


（二）退格/删除按钮

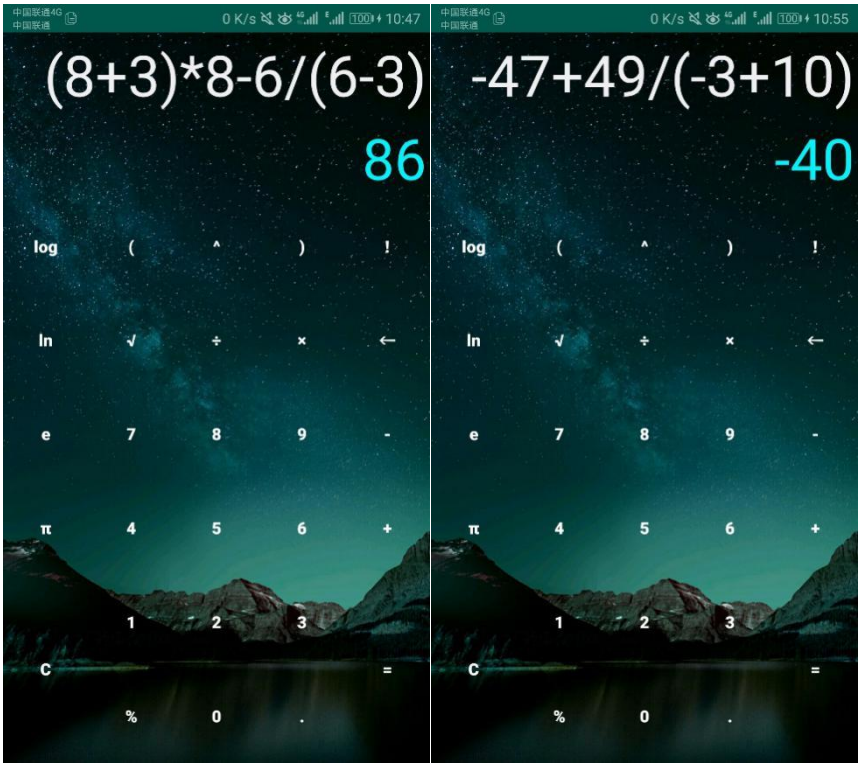


（注：①退格时不清除结果；②删除按钮(C)可以清空输入框和结果；③长按退格按钮，效果与删除按钮一致）

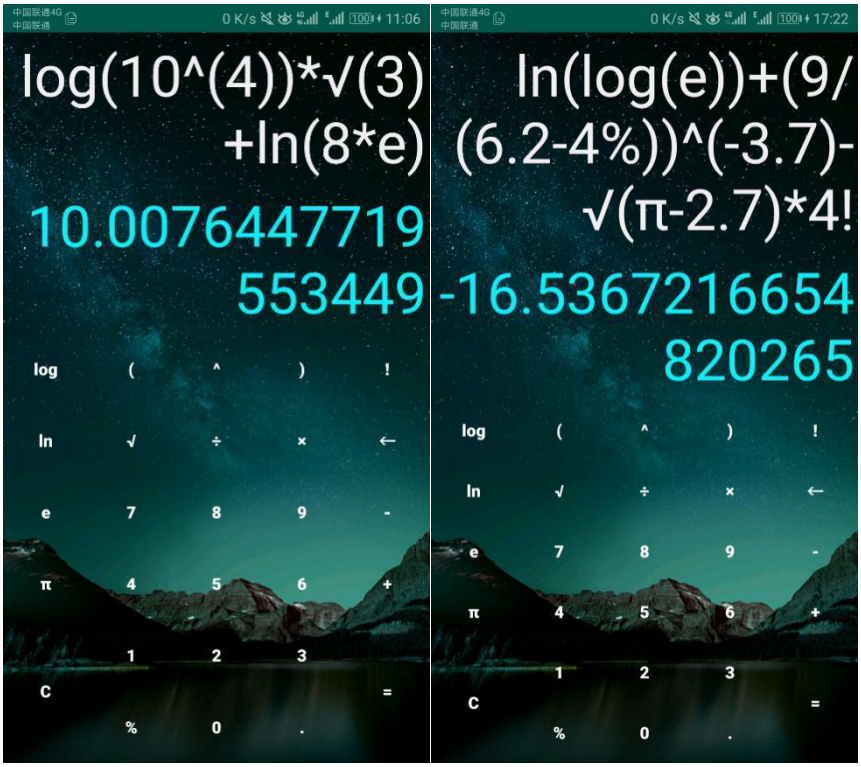
（三）小数、百分号



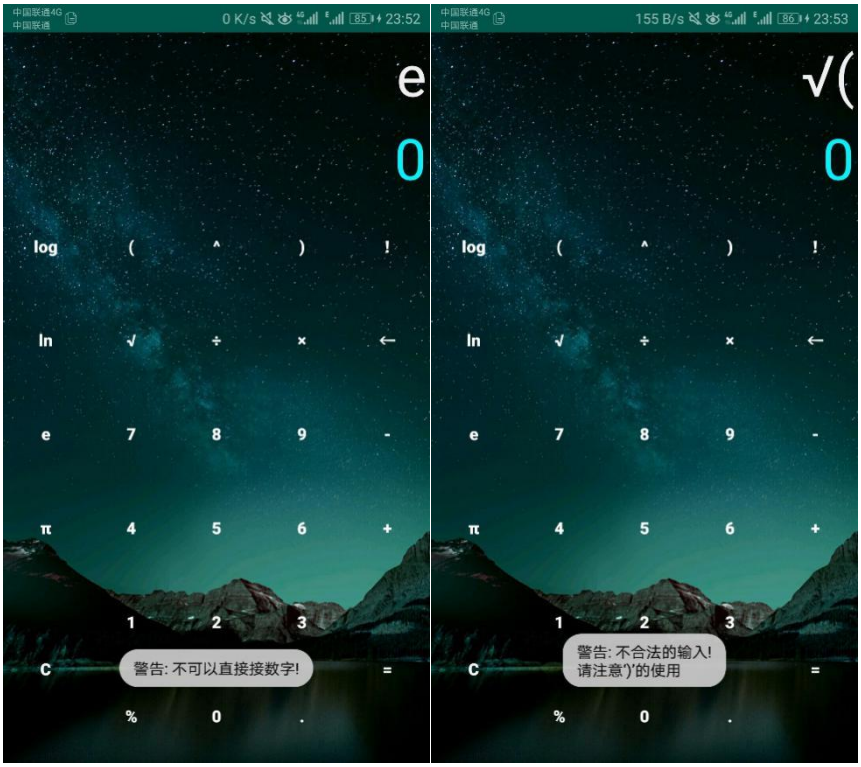
（四）括号、负数



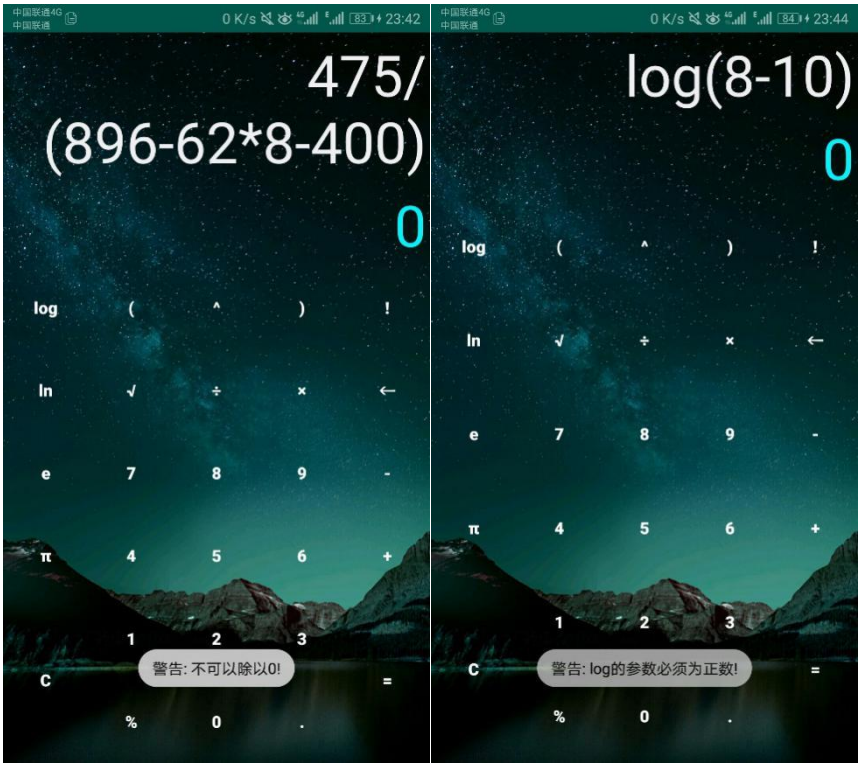
（五）对数、指数、根号、阶乘



（六）输入提示



（七）错误提示



七、 遇到的问题与解决方案

（一）使用校园网 build 时下载相关文件很慢甚至出错。

解决方案：尝试使用手机流量下载。

(二) 整体粘贴 activity_main.xml 时界面不是预期效果。

解决方案：尝试逐项粘贴或检查 width 和 height 是否和 weight 冲突。

(三) TextView 底部有一条横线影响观感。

解决方案：android:background="@null"

(四) Button 的 text 即使设置为小写也会被自动转换为大写显示。

解决方案：android:textAllCaps="false"

(五) 布局文件很多都是相同的属性，导致代码冗长，且不方便修改。

解决方案：在 styles.xml 里自定义 style。

(六) 表达式输入栏一开始设置为 EditText，然而解决了光标保持在最后的问题之后，又发现点击输入栏时，会调用虚拟键盘，网上找了很多方法都不见效，最后不得已将 EditText 改为了 TextView。

(七) 测试指数运算时发现当指数不是整数时结果不对，查了才发现 BigDecimal 的 pow 函数的参数是 int 类型的，只好又改用 Math.pow 函数。

(八) 遇到玄学错误，如：Button 的 text 是居中显示的，但偏偏有一个 Button 显示在左边。

解决方案：新建项目，复制粘贴原项目的代码。

八、 不足之处

(一) 输入框不能自由编辑，只能在最后添加和删除，当检查到已输

入的表达式某处有错误时，只能退格回去修改后重新输入；

（二）当运算结果不是实数时，app 会异常退出；

（三）当运算结果数值很大时没有自动转为科学计数法；

（四）Button 与 Button 之间看不出间隙，TextView 多行显示时字符有时不填充满整行，且滑动效果不佳；

（五）没有实现记录计算历史；

（六）功能相对偏少，离真正的科学计算器还有很大差距。

九、 今后的设想

这是自己设计编写的第一个有点用处的 Android app。在完成这个 app 的过程中，我发现一款优秀的软件在呈现给用户之前是需要花费大量的时间与精力去打磨界面与功能。此外，在调试 app 时，自己想到什么就测什么(估计现在还有很多 bug 没有发现)，不系统全面。所以，今后的开发将要更加注重界面设计的美观，细节的优化,操作与功能的人性化，也要学习科学完备的测试方法。