南京邮电大学计算机学院

2020 / 2021 学年第 2 学期

《JAVA 程序设计》课程大作业报告

课题代号（单选）☐课题 1　　☑课题 2　　☐课题 3

课题名称：　　二进制计算器程序

团队成员人数（单选）☐1 人　　☐2 人　　☑3 人　　☐4 人　☐5 人

团队成员名单(学号和姓名）：

B19030313	冯大宇
B19030317	李泽信
B19030318	杨亚鹏

课题实现采用的操作系统（可多选，如为其他请填写操作系统名称）
☑Microsoft　Windows　☐Unix　☐ Linux ☐Mac OS　☐其他

课题实现采用的软件开发工具（可多选，如为其他请填写具体工具名称）
☐NetBeans　☐Eclipse　☐JDK 命令行　☑其他:IDEA

课题实现采用的文档工具（可多选，如为其他请填写具体工具名称）
☑Microsoft Office　☑JavaDoc　☐Markdown　☐LaTex　☐其他

课题实现采用的团队协作管理工具（可多选，如为其他请填写具体工具名称）
☐人工进行　☐SubVerion　☑GIT　☐其他

课题完成时间：2021-6-21
对应提交的电子文档文件夹名称:official

==========以下部分为教师填写区，请勿填写==========

评阅编号:（　　）－（　　）

成绩评定：详见大作业对应的“《JAVA 程序设计》课程大作业评分记录表”

南京邮电大学计算机学院 2020 / 2021 学年第 2 学期

《JAVA 程序设计》课程学习反馈表

说明：本反馈表用于了解课程学习情况以便今后更好地优化课程教学，请大作业团队成员根据情况认真填写本反馈表，谢谢！（请在选项中涂黑或打勾）

一。课程学习部分

1.在选修本课程前大作业团队成员对 JAVA 技术的了解（单选）
☐一点不了解　　☐有一些了解　　☐已开始自学　　☑已完成自学

2.在完成本课程后大作业团队成员对 JAVA 技术的了解（单选）
☐一点不了解　　☐有一些了解　　☑有较多了解　　☐有很多了解

3.在完成本课程后大作业团队成员希望进一步学习的 JAVA 技术领域（多选）
☐JAVA 企业及云计算　　☑JAVA WEB 前端及手机应用
☐JAVA 嵌入式系统应用　　☑JAVA 数据分析及大数据处理

4.在完成本课程后大作业团队成员认为本课程中需要加强的教学内容（多选）
☐面向对象理论　　☑JAVA 基础语法　　☐课后作业　　☐课内实验

二．大作业部分

1.大作业团队成员认为本课程的大作业课题难度情况（单选）
☐很难　　☐比较难　　☑简单　　☐非常简单

2.在完成大作业过程中对软件开发工具的使用情况（单选）
☐不会使用工具，需要帮助　　☐基本会使用工具，但有困难
☑会使用工具，基本没有困难　　☐会使用工具，并且能对外提供帮助

3.在完成大作业过程中对软件文档工具的使用情况（单选）
☐不会使用工具，需要帮助　　☐基本会使用工具，但有困难
☑会使用工具，基本没有困难　　☐会使用工具，并且能对外提供帮助

4.在完成大作业过程中对软件协作工具的使用情况（单选）
☐不会使用工具，需要帮助　　☐基本会使用工具，但有困难
☑会使用工具，基本没有困难　　☐会使用工具，并且能对外提供帮助

5.完成大作业对上述各种信息技术工具使用能力的总体提升（单选）
☐没有提升　　☐有一点提升　　☑有较多的提升　　☐有很大的提升

6.在完成大作业过程中获取的工程能力中的软件设计能力的提升（单选）
☐没有提升　　☐有一点提升　　☐有较多的提升　　☑有很大的提升

7.在完成大作业过程中获取的工程能力中的代码编写能力的提升（单选）
☐没有提升　　☐有一点提升　　☐有较多的提升　　☑有很大的提升

8.在完成大作业过程中获取的工程能力中的代码调试能力的提升（单选）
☐没有提升　　☐有一点提升　　☐有较多的提升　　☑有很大的提升

9.在完成大作业过程中获取的工程能力中的代码测试能力的提升（单选）
☐没有提升　　☐有一点提升　　☐有较多的提升　　☑有很大的提升

10.在完成大作业过程中获取的工程能力中的代码优化能力的提升（单选）
☐没有提升　　☐有一点提升　　☐有较多的提升　　☑有很大的提升

11.在完成大作业过程中获取的工程能力中的算法设计能力的提升（单选）
☐没有提升　　☐有一点提升　　☐有较多的提升　　☑有很大的提升

12.在完成大作业过程中获取的工程能力中的文档撰写能力的提升（单选）
☐没有提升　　☐有一点提升　　☐有较多的提升　　☑有很大的提升

13.在完成大作业过程中获取的工程能力中的团队协作能力的提升（单选）
☐没有提升　　☐有一点提升　　☐有较多的提升　　☑有很大的提升

14.完成大作业对上述各种实际工程能力的总体提升（单选）
☐没有提升　　☐有一点提升　　☐有较多的提升　　☑有很大的提升

三．其它反馈意见（欢迎更多的意见和建议，请填写在下面空白处，谢谢）

希望以后可以学习类似 springboot、springcloud、hadoop 等后端开发和大数据处理方面主流的技术，大作业可以布置类似“团队利用 svn 或 git 协作开发一个完整的小型 web 项目”的题目。这样既可以锻炼学生前后端分离合作的能力，也能为日后学生的实习就业打下一些基础。

目录

一. 软件系统分析和设计方案

需求分析

我们选择的题目是二进制计算器程序，要求实现二进数的基本运算，具体包括一个显示窗口，‘0’按键，‘1’按键，‘+’按键，‘-’按键，‘AND’按键，‘OR’按键，‘XOR’按键，‘NOT’按键，‘=’按键，‘CLR'按键。

功能分析

要求设计的二进制计算器可以进行二进制加法，减法，与，或，异或，取反计算。

基本要求：

- 字长为 8 位字节的运算
- ‘CLR’按键可以清除计算结果和屏幕以便下一次使用

提高要求：

- 非法输入错误提示功能
- 支持运算字长选择功能
- 支持运算标志位显示
- 增加'COMP'补码按键

除此之外，我们的二进制计算器还支持连续运算。

系统设计方案

我们采用 java 的 swing 类库实现可视化的二进制计算器，核心是将用户所有的输入分为 operand1(操作数 1)、operator(运算符)、operand2(操作数 2)三种。其中单目运算符 NOT(取反)、COMP(补码)只需要操作数 1 和运算符；双目运算符均需要操作数 1、操作数 2 及运算符。

在具体进行运算时，先利用 Long.parseLong()方法将所需的操作数从二进制转化为十进制，再根据对应的运算符对转换后的操作数进行运算，最后利用 Long.toBinaryString()方法将结果转回二进制并输出，由于该方法对于负数返回的是二进制补码，我们对其另作特殊处理。

二. 软件实现和代码编写

关键模块的实现方法

我们通过对按钮的监听得到了用户的输入，用户每点击一个按钮，输入就是按钮上的字符串。随后我们通过对用户输入的判断、分类得到了 operrand1、operand2 及 operator 三个关键的变量值，并在用户按下“=”或单目运算符“NOT”、“COMP”后将运算结果输出显示在窗口上。下面是关键代码的展示及代码的分析。

关键代码的分析

```
@Override
public void actionPerformed(ActionEvent e)
{
    String btnTag = e.getActionCommand();
    boolean error = false;

    // if the value is a number
    if (btnTag.equals("0")||btnTag.equals("1")) {
        // if operand is present then add to second no
        if (!operator.equals("")&&!operator.equals("NOT")&&!operator.equals("COMP")) {
            if(LengthLimit(operand2)) {operand2 = operand2 + btnTag;}
        }
        else {
            if(LengthLimit(operand1)) {operand1 = operand1 + btnTag;}
        }
        // set the value of text
```

```
            jTextField.setText(operand1 + " " + operator + " " + operand2);
        }
        else if (btnTag.equals("CLR")) {
            // clear the one letter
            showSignal("#");
            operand1 = operator = operand2 = result = "";
            // set the value of text
            jTextField.setText("");
            lb_result.setText("");
        }
        else if (btnTag.charAt(0) == '=') {
            if((!operand1.equals("") && !operator.equals("") && !operand2.equals("")) ||
(!operand1.equals("") && operator.equals("") && operand2.equals(""))){
                // only @operand1 is obtained
                if(operand2.equals("")) {
                    result = Long.toBinaryString(Long.parseLong(operand1, 2));
                }
                // all three parameters are obtained
                if (operator.equals("+")) {
                    Long tempLong = Long.parseLong(operand1,2) + Long.parseLong(operand2,2);
                    if(tempLong < 0) {
                        result = "-" + Long.toBinaryString(Math.abs(tempLong));
                    } else {
                        result = Long.toBinaryString(tempLong);
                    }
                }
                else if (operator.equals("-")){
                    Long tempLong = Long.parseLong(operand1,2) - Long.parseLong(operand2,2);
                    if(tempLong < 0) {
                        result = "-" + Long.toBinaryString(Math.abs(tempLong));
                    } else {
                        result = Long.toBinaryString(tempLong);
                    }
                }else if (operator.equals("AND")) {
                    if(operand1.charAt(0) == '-') {
                        error = ErrorWarning("非法操作。");
                    }else{
                        result = Long.toBinaryString(Long.parseLong(operand1, 2) &
Long.parseLong(operand2, 2));
                    }
                }else if (operator.equals("OR")) {
                    if(operand1.charAt(0) == '-') {
                        error = ErrorWarning("非法操作。");
                    }else{
                        result = Long.toBinaryString(Long.parseLong(operand1, 2) |
Long.parseLong(operand2, 2));
                    }
                }else if (operator.equals("XOR")) {
                    if(operand1.charAt(0) == '-') {
                        error = ErrorWarning("非法操作。");
                    }else {
                        result = Long.toBinaryString(Long.parseLong(operand1, 2) ^
Long.parseLong(operand2, 2));
                    }
                }
                if(!error){
                    // set the value of text
                    if(operand2 == "") {
                        operand2="0";
                        jTextField.setText(ZeroSupplement(operand1,binaryBits) + " =");
                    }else{
                        jTextField.setText(ZeroSupplement(operand1,binaryBits) + " " +
operator + " " + ZeroSupplement(operand2,binaryBits) + " =");
                    }
                    String tempStr = ZeroSupplement(result,binaryBits);
                    lb_result.setText(tempStr.substring(tempStr.length()-binaryBits));
                    showSignal(result);
                    operand1 = tempStr.substring(tempStr.length()-binaryBits);
```

```
                    operator = operand2 = result = "";
                }
            }else{
                error = ErrorWarning("非法输入");
            }
        }
        else if(btnTag.equals("NOT")){
            lb_result.setText("");
            if(!operand1.equals("")){
                if(operand1.charAt(0) == '-') {
                    error = ErrorWarning("操作数不符合要求，请输入源码形式。");
                }else{
                    if(operand1.length()<= binaryBits){
                        operand1 = ZeroSupplement(operand1, binaryBits);
                        StringBuilder strBuilder = new StringBuilder(operand1);
                        for(int i = 0; i< binaryBits; i++){
                            if(operand1.charAt(i)=='0') {
                                strBuilder.setCharAt(i,'1');
                            }else{
                                strBuilder.setCharAt(i,'0');
                            }
                        }
                        result=strBuilder.toString();
                    }
                    jTextField.setText("NOT " + operand1 + " =");
                    lb_result.setText(ZeroSupplement(result, binaryBits));
                    showSignal(result);
                    operand1 =result;
                    operator = operand2 = result = "";
                }
            }else{
                error = ErrorWarning("请先输入源码形式的操作数。");
            }
        }
        else if(btnTag.equals("COMP")){
            lb_result.setText("");
            if(!operand1.equals("")){
                if(operand1.charAt(0) == '-') {
                    error = ErrorWarning("操作数不符合要求，请输入原码形式。");
                }else{
                    if(operand1.length()<= binaryBits){
                        operand1 = ZeroSupplement(operand1, binaryBits);
                        if(operand1.charAt(0)=='0'){
                            result= operand1;
                        }else{
                            StringBuilder strBuilder = new StringBuilder(operand1);
                            for(int i = 1; i< binaryBits; i++){
                                if(operand1.charAt(i)=='0') {
                                    strBuilder.setCharAt(i,'1');
                                }else{
                                    strBuilder.setCharAt(i,'0');
                                }
                            }

result=Long.toBinaryString(Long.parseLong(strBuilder.toString(),2)+1);
                        }
                    }
                    jTextField.setText("COMP " + operand1 + " =");
                    String tempStr = ZeroSupplement(result,binaryBits);
                    lb_result.setText(tempStr.substring(tempStr.length()-binaryBits));
                    showSignal(result);
                    operand1 =tempStr.substring(tempStr.length()-binaryBits);
                    operator = operand2 = result = "";
                }
            }else{
                error = ErrorWarning("请先输入原码形式的操作数。");
            }
        }
```

```
        else if(btnTag.equals("8") || btnTag.equals("16") || btnTag.equals("32")){
            if(btnTag.equals("8")) {
                binaryBits =8;
                jTextField.setFont(new Font("Times New Roman", Font.ITALIC, 35));
                lb_result.setFont(new Font("Consolas", Font.BOLD, 70));
            }
            else if(btnTag.equals("16")) {
                binaryBits =16;
                jTextField.setFont(new Font("Times New Roman", Font.ITALIC, 25));
                lb_result.setFont(new Font("Consolas", Font.BOLD, 50));
            }
            else if(btnTag.equals("32")) {
                binaryBits =32;
                jTextField.setFont(new Font("Times New Roman", Font.ITALIC, 15));
                lb_result.setFont(new Font("Consolas", Font.BOLD, 28));
            }
        }
        else{
            // Judge the situation of "+" and "-"
            lb_result.setText("");
            if(operand1.equals("")){
                // when operand1 is forgot
                JOptionPane.showMessageDialog(new JFrame(), "非法输入！", "警告
",JOptionPane.WARNING_MESSAGE);
                showSignal("#");
                jTextField.setText("");
                lb_result.setText("");
                operand1 = operator = operand2 = result = "";
            }else if (operator.equals("") || operand2.equals("")) {
                // if we change the operator before operand2 input
                operator = btnTag;
                showSignal("#");
            }else {
                // In continuous operation, let the result be operand1
                lb_result.setText("");
                if (operator.equals("+")) {
                    Long tempLong = Long.parseLong(operand1,2) + Long.parseLong(operand2,2);
                    if(tempLong < 0) {
                        result = "-" + Long.toBinaryString(Math.abs(tempLong));
                    } else {
                        result = Long.toBinaryString(tempLong);
                    }
                }
                else if (operator.equals("-")){
                    Long tempLong = Long.parseLong(operand1,2) - Long.parseLong(operand2,2);
                    if(tempLong < 0) {
                        result = "-" + Long.toBinaryString(Math.abs(tempLong));
                    } else {
                        result = Long.toBinaryString(tempLong);
                    }
                }else if (operator.equals("AND")) {
                    if(operand1.charAt(0) == '-') {
                        error = ErrorWarning("非法操作。");
                    }else{
                        result = Long.toBinaryString(Long.parseLong(operand1, 2) &
Long.parseLong(operand2, 2));
                    }
                }else if (operator.equals("OR")) {
                    if(operand1.charAt(0) == '-') {
                        error = ErrorWarning("非法操作。");
                    }else{
                        result = Long.toBinaryString(Long.parseLong(operand1, 2) |
Long.parseLong(operand2, 2));
                    }
                }else if (operator.equals("XOR")) {
                    if(operand1.charAt(0) == '-') {
                        error = ErrorWarning("非法操作。");
                    }else {
```

```
                    result = Long.toBinaryString(Long.parseLong(operand1, 2) ^
Long.parseLong(operand2, 2));
                }
            }
            // convert it to string
            showSignal(result);
            operand1 = result;
            // place the operator
            operator = btnTag;
            // make the operand blank
            operand2 = result = "";
        }

        // reset the value of text
        jTextField.setText(operand1 + " " + operator + " " + operand2);
    }
}
```

对 actionPerformed()函数的分析:

首先判断如果输入的是“0”或“1”，代表用户当前在进行操作数的输入。如果当前运算符不为空且不为单目运算符“NOT”、“COMP”，则认为用户输入的是操作数 2；否则认为用户输入的是操作数 1。通过字符串的拼接，我们获得了完整的操作数。

如果输入的是“CLR”，即清零按钮，我们将两个操作数、运算符及显示窗口均置为空，以此达到清零的效果。

如果输入的是“=”，我们认为只有当前的操作数 1、操作数 2、运算符均不为空，这次的三目运算才是合法的，允许根据运算符进行运算后输出；否则显示非法输入警告，并将操作数 1、2 及运算符清空。具体运算时，我们先将二进制字符串转换为对应的十进制整数，运算后将结果转回二进制数输出。

如果输入的是单目运算符“NOT”或“COMP”，我们认为当前的操作数 1 不为空时，这次的单目运算就是合法的，允许根据运算符进行运算后输出。在进行“NOT”运算时，我们先将二进制数按照选择的位数(8 位，16 位，32 位)在前面补足“0”，随后通过对每个字符的操作进行按位取反得到结果。在进行“COMP”操作时，我们先按照上面“NOT”运算的步骤得到输入二进制数的反码，再将二进制数反码转换为十进制数后在末尾加一得到输入二进制数补码的十进制形式，再转为二进制后得到最终结果。由于补码的前几位可能为“0”，默认输出时会被舍弃，所以我们在输出前先进行了补“0”操作，保证输出的位数始终和用户选择的位数相同。此外本次运算结束后，本次运算的结果会赋值给操作数 1，操作数 2 和操作符置为空，以此实现了用户的连续运算。

如果输入的是“8”、“16”及“32”，即用户在进行位数的选择，计算器会根据用户输入进行位数的变更。

如果输入的是剩余的双目运算符，如果操作数 1 为空，弹出“非法输入”的提示；如果操作数 1 不为空，但操作数 2 为空或者操作符为空，则让输入的运算符覆盖之前的运算符。如果操作数 1、2 及操作符都不为空，对应的就是用户想进行连续运算的情况，此时将本次运算的结果作为操作数 1，与输入的操作符一起显示在窗口上。同单目运算符一样，本次运算结束后，本次运算的结果会赋值给操作数 1，操作数 2 和操作符置为空，以此实现了用户的连续运算。

需注意,为满足连减功能仍然正确,我们实现了负数的计算,但由于题目要求中并未给出“+/-”按钮，用户可以正确的到负数结果，但无法输入负数，只能对负数的结果进行进一步操作。此外，负数

```
public void showSignal(String result){

       if (result != "#"){
           String supplementaryOperand1 = ZeroSupplement(operand1,binaryBits);
           String supplementaryOperand2 = ZeroSupplement(operand2,binaryBits);
           String supplementaryResult = ZeroSupplement(result,binaryBits);

           int A=0;
           int C=0;
           int O=0;
           int P=0;
           int Z=1;
           int S=0;
```

```
            // Determine the status symbol @S
            if(supplementaryResult.charAt(0)=='-') {
                S=1;
            }else{
                // @A and @C are only considered when is not negative
                // Determine the status symbol @A
                if(Long.parseLong(supplementaryOperand1.substring(binaryBits/2),2) +
Long.parseLong(supplementaryOperand2.substring(binaryBits/2),2) >
Math.pow(2,binaryBits/2)-1) { A=1; }

                // In order to determine the status symbol @C, create params to store the highest
bits of the Operand and Result respectively
                int highestOperandBit = binaryBits, highestResultBit = binaryBits;
                for(int i=0, count=0; i<binaryBits; i++){
                    // Record the highest bit of operand 1 and operand 2
                    if(supplementaryOperand1.charAt(i) == '1') {
                        highestOperandBit = Math.min(i,highestOperandBit);
                    }else if(supplementaryOperand2.charAt(i) == '1') {
                        highestOperandBit = Math.min(i,highestOperandBit);
                    }
                    // Determine whether the highest bit is carried
                    if(supplementaryResult.charAt(i) == '1') {
                        highestResultBit = Math.min(i, highestResultBit);
                    }
                }
                if(highestResultBit < highestOperandBit) { C = 1; }
            }

            // Determine the status symbol @O

if(Long.parseLong(supplementaryOperand1,2)+Long.parseLong(supplementaryOperand2,2) >
Math.pow(2,binaryBits)-1
                ||
Long.parseLong(supplementaryOperand1,2)+Long.parseLong(supplementaryOperand2,2) <
-(Math.pow(2,binaryBits)-1)) { O=1; }

            // Determine the status symbol @Z and @P
            for(int i=0, count=0;i<result.length();i++){
                if(result.charAt(i)=='1') {
                    count++;
                    Z=0;
                }
                // Execute when the loop is about to end
                if(result.length() == (i+1)) {
                    if (count % 2 == 0) {
                        P = 1;
                    }
                }
            }

            //set status
            if(A == 1){
                labelA.setFont(new Font("Concolas",Font.BOLD,20));
                labelA.setForeground(Color.red);
            }
            if(C == 1){
                labelC.setFont(new Font("Concolas",Font.BOLD,20));
                labelC.setForeground(Color.red);
            }
            if(O == 1){
                labelO.setFont(new Font("Concolas",Font.BOLD,20));
                labelO.setForeground(Color.red);
            }
            if(P == 1){
                labelP.setFont(new Font("Concolas",Font.BOLD,20));
                labelP.setForeground(Color.red);
            }
            if(Z == 1) {
                labelZ.setFont(new Font("Concolas", Font.BOLD, 20));
```

```
                labelZ.setForeground(Color.red);
            }
            if(S == 1) {
                labelS.setFont(new Font("Concolas",Font.BOLD,20));
                labelS.setForeground(Color.red);
            }

        }else{
            //if str="#", reset
            labelA.setFont(new Font("Concolas",Font.PLAIN,20));
            labelA.setForeground(Color.gray);
            labelC.setFont(new Font("Concolas",Font.PLAIN,20));
            labelC.setForeground(Color.gray);
            labelO.setFont(new Font("Concolas",Font.PLAIN,20));
            labelO.setForeground(Color.gray);
            labelP.setFont(new Font("Concolas",Font.PLAIN,20));
            labelP.setForeground(Color.gray);
            labelZ.setFont(new Font("Concolas", Font.PLAIN, 20));
            labelZ.setForeground(Color.gray);
            labelS.setFont(new Font("Concolas",Font.PLAIN,20));
            labelS.setForeground(Color.gray);
        }
    }
```

显示运算后 *A,C,O,P,S,Z* 六个状态标志，其中：

- *A* 等于 1 时表示发生辅助进位(不考虑减法借位)
- *C* 等于 1 时表示发生最高位产生进位
- *O* 等于 1 时表示溢出
- *P* 等于 1 时表示最低 8 位 1 的个数为偶数
- *S* 等于 1 时表示运算结果为负性（原码为最高位的值，但由于本计算机设计逻辑上以真值形式进行计算，需另行设计）
- *Z* 等于 1 时表示运算结果为 0

以上六个状态标志值为 1 时在界面上显示为红色加粗。

三. 算法分析

将每一次运算都拆解成操作数 1、操作数 2 及操作符三个部分，并用 operand1、operand2 及 operator 三个变量分别保存，简化了运算流程，并且对异常输入输出更容易检查和维护。

四. 软件调试和测试

软件调试中遇到的问题及解决过程

1. 当操作数、运算符不全时直接按等号或单目运算符“NOT”、“COMP”，idea 报错。原因是没有考虑到用户的非法输入，没有对进入运算前的条件进行限制，导致运算中的操作数或运算符为空。增添判断条件及非法输入提示后解决。
2. 当进行减法运算运算结果为负数时，默认得到的结果是补码形式，显示结果不够直观且跟位数有关。我们增添了符号“-”作为输出结果的负号，如果运算结果为负，会自动在最前面加上“-”。
3. 字体大小设置不合理，导致在进行 16 位及 32 位运算时，结果显示不全。修改字体后成功解决。
4. 考虑到默认的 int 类型在内存中占 4 个字节，也就是用 32 位二进制表示。为了预防溢出的问题，我们将 int 类型改为了 long 类型，使运算器支持更高位的运算。

软件的测试方案和实施过程

1. 首先测试 8 位、16 位、32 位下各个运算符的运算：
   a) 启动软件，修改位数，任意取操作数和运算符进行运算。
   b) 清零。在输入操作数和运算符的过程中修改位数。

c) 在已有输出的前提下，修改运算符，重新取操作数及运算符进行运算。

2. 测试输入位数超过最大位数限制的非法输入：

连续输入一串数字，观察是否出现警告。

3. 测试操作数、运算符输入不全的非法输入：

当输入的操作数 1、操作数 2、运算符三者缺少 1 个或两个，一共 6 种情况下计算器是否能正确识别。

4. 测试连续运算：

当输入操作数 1、操作数 2、运算符后，不按等号，继续输入操作数，观察计算器显示结果。

5. 测试运算状态标志：

a) 设置触发 1 个状态标志的运算

b) 设置触发 2 个或多个状态标志的运算

c) 测试连续运算时状态标志是否正常显示

6. 测试运算结果为负数的运算：

a) 设置两数相减结果为负的运算，观察结果

b) 在连续加减的情况下，观察结果为负的情况

c) 当出现负值时，连续加减，观察结果是否正确

五. 课题项目工程实践总结

软件开发工具的选择和使用方案

开发工具包括 idea 集成开发环境，用 git 进行版本控制，idea 中集成了 git 工具。

文档工具的使用和使用方案

项目需求文档使用石墨文档在线协作，课程大作业报告使用 Microsoft Word 进行编写。该文档工具能提供较为详尽的各类文本编辑工具，大大减少了编辑文本的工作量。

项目团队协作的管理方法

团队使用 QQ 群进行联系，三人各自分工，分别负责项目需求文档中的某一部分进行编程，并通过使用 git 上传远程仓库再拉取最新版本到本地的方式保证本地版本的一致。

整个项目遇到的问题以及解决方法和体会等

在本次大作业完成过程中，我们遇到了很多困难，其中包括 git 因为网络问题 push 失败，jdk 版本不兼容，需求文档多次修改完善等等。这些问题我们都通过交流讨论成功解决，并获得了一定的经验，这更让我们体会到团队协作的重要性。

在编程过程中，加强了我们对所学课程的理解，提升了我们的编程水平。经过这次实践操作，使我们在课堂上学到的理论知识技能融会贯通。在进行代码的编写的时候，我们对一些概念有了更加深刻的理解。例如：对于 JFrame 的运用，对于线程的使用，还有 git 版本控制中的流程，swing 编程等等。

此外，还要养成良好的编程习惯，代码的编写进行到一定阶段后就要做一下调试，在平时细小方面随时注重细节，这些工作在我们在编写程序的时候有很大的帮助。另外，每次编写程序，代码出现错误是难免的，所以也要锻炼自己的心理承受能力。最初进行测试的时候，可能会发现自己的程序有很多的漏洞。这时就需要克服自己的焦躁心态，不要想办法去掩盖程序中的错误，而是尽可能的多发现程序中存在的问题，并通过解决这些问题逐步提高自己的编程素养、对于课程的理解和程序开发能力。在学习过程中，一定要多动手多进行试验，不要纸上谈兵流于表面光在那里进行思考。

六. 参考文献和材料

- Java Swing 图形界面开发
- BeautyEye 快速入门介绍

七. 团队成员姓名和联系方法

学号	姓名	邮箱	联系电话
B19030313	冯大宇	2218785142@qq.com	18913210942
B19030317	李泽信	toulzx@foxmail.com	17715290317
B19030318	杨亚鹏	2087369435@qq.com	15850530959

八. 附录

README.txt