实验内容	第 9 周 GUI 设计初步			成 绩	
姓 名	王秋锋	学号	2015111948	班 级	计算机 2015-03 班
专 业	计算机科学与技术			日 期	2017 年 11 月 2 日

【实验目的】--字符串处理

- 掌握 JAVA 语言 GUI 设计的概念；掌握 AWT 常用组件类；
- 掌握 JAVA 语言系统常用组件类的功能和使用；
- 掌握 JAVA GUI 委托事件处理机制。

【实验内容】

1、编辑、编译、运行下面 java 程序

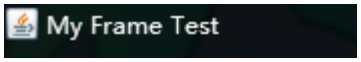
```
import java.awt.*;

public class TestFrame
 {
    public static void main( String args[])  //创建标题 My Frame Test 的窗口容器对象实例
  {
     Frame f = new Frame("My Frame Test");
      f.setSize( 170,100);  //设置 Frame 组件窗口大小（宽 、高）
      f.setBackground( Color.blue);  //设置 Frame 组件背景色为蓝色
      f.setVisible( true);  //设置 Frame 组件为可视
   }
 }
```

要求：

（1）分析该程序，写出运行结果

【实验结果与分析】

实验结果：



结果分析：

利用 awt（抽象窗口工具包）中的 Frame（窗口类）创建一个窗口容器对象，通过调用该容器的方法设置窗口的大小，背景颜色，是否可见等信息，最后的结果如上图所示。

2、编辑、编译、运行下面 java 程序

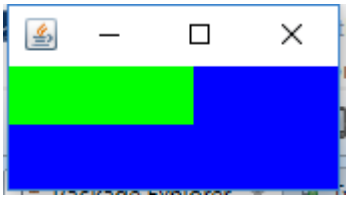
```
import java.awt.*;
public class TestFrameWithPanel
{
  public static void main(String args[])
  {

   Frame f = new Frame("My FrameWithPanel Test ");  //创建带标题的窗口容器组件对象实例
    Panel pan = new Panel();  //在窗口容器上创建面板容器对象
    f.setSize(180,100);       //设路 Frame 组件窗口大小（宽 、高）
    f.setBackground(Color.blue); //设路 Frame 组件背景色为蓝色

    f.setLayout(null);       // 取消布局管理器（可同时看到窗口和面板组件）
    pan.setSize(100,60);     //面板容器大小（宽 、高）

    pan.setBackground(Color.green);  //设路面板容器组件背景色为绿色
    f.add(pan);                 //在窗口容器组件中添加面板容器对象
    f.setVisible(true);           //设路 Frame 组件为可视
   }
}
```

要求：

（1）分析该程序，写出运行结果

【实验结果与分析】

实验结果：



结果分析：

本题在上一题的窗口的基础上加了一个面板， 设置面板组件的背景色为绿色， 同时关闭窗口 f 的布局管理器，使窗口和面板组件能够同时被看到，最后的结果如上图所示。

3、编辑并运行下面程序，理解 Date、SimpleDateFormat 类的使用

用 Data 类不带参数的构造方法创建日期，要求日期的输出格式是:星期 小时 分 秒

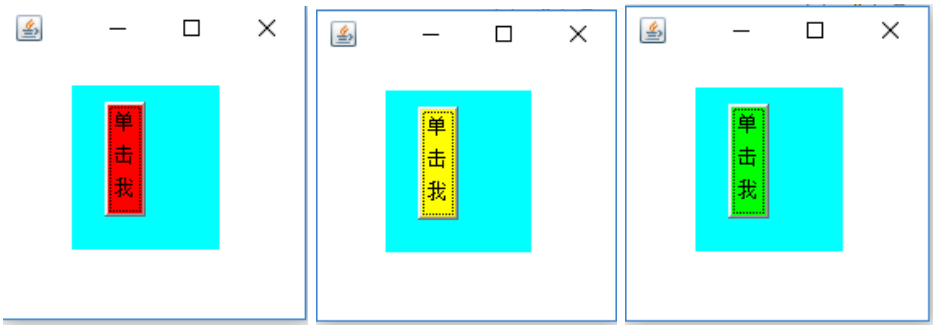
```
import java.awt.*;
import java.awt.event.*;
class WindowNull extends Frame
```

```
{  WindowNull()                    //构造函数
   {  setLayout(null);
      MyButton button=new MyButton();  //创建按钮对象
      Panel p=new Panel();   //创建面板容器
      p.setLayout(null);     //设置面板容器为 null 布局
      p.setBackground(Color.cyan);  //设置面板容器背景色
      p.add(button);              //填加按钮到面板中
      button.setBounds(20,10,25,70);  //设置按钮在面板容器中的位置
      add(p);                     //添加面板到窗口中
      p.setBounds(50,50,90,100);     //设置面板在窗口中的位置
      setBounds(120,125,200,200);    //设置窗口在屏幕的位置
      setVisible(true);              //设为可视
   }
}
class MyButton  extends Button implements ActionListener
{  int n=-1;
   MyButton()       //构造函数
   {  addActionListener(this);  //注册按钮监听器
   }
   public void paint(Graphics g)  //在子类中重写 Component  Button 类中的 paint()方法
   {  g.drawString("单",6,16);  //在坐标点处绘制单击我字符
      g.drawString("击",6,36);
      g.drawString("我",6,56);
   }
   public void actionPerformed(ActionEvent e)  //点击按钮产生事件处理方法
   {  n=(n+1)%3;
      if(n==0)
         setBackground(Color.red);      //设置按钮背景色
      else if(n==1)
         setBackground(Color.yellow);
      else if(n==2)
         setBackground(Color.green);
   }
}
public class E
{   public static void main(String args[])
    {  new WindowNull();  //创建 WindowNull 类实例并调用它的构造函数初始化
    }
}
```

要求：运行程序，给出正确的程序运行结果，理解正规式的使用。

【实验结果与分析】

实验结果：（每次单击之后的结果）



结果分析：

本程序定义了一个 Frame 的窗口派生类 WindowNull，WindowNull 采用 null 布局， 包括一个面板， 面板采用 null 布局， 面板上还有一个自定义的按钮类 MyButton 的按钮对象， 并设置各个组件间的位置。然后自定义了按钮类 MyButton， MyButton 类中重写了 Button 类中的 paint()方法， 在坐标点处绘制“单”“ 击”“我”三个字符， 并在该类的构造函数中为其注册了监听器，当监听器监听到按钮被按下这个事件之后， 系统自动调用事件处理方法， 当按下后会出现红黄绿三色循环。

4、编辑并运行下面程序，理解 Calendar 类的使用

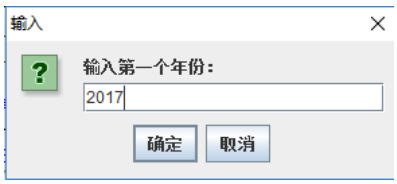
```
import java.util.*;
import javax.swing.JOptionPane;
public class TestDateDemo
{
    public static void main(String[] agrs){
    String str = JOptionPane.showInputDialog("输入第一个年份：");
    int firstYear = Integer.parseInt(str);
    str = JOptionPane.showInputDialog("输入月份：");
    int firstMonth = Integer.parseInt(str);
    str = JOptionPane.showInputDialog("日期：");
    int firstDay = Integer.parseInt(str);

    str = JOptionPane.showInputDialog("输入第二个年份：");
    int secondYear = Integer.parseInt(str);
    str = JOptionPane.showInputDialog("输入月份：");
    int secondMonth = Integer.parseInt(str);
    str = JOptionPane.showInputDialog("日期：");
    int secondDay = Integer.parseInt(str);

    Calendar calendar = Calendar.getInstance();
    calendar.set(firstYear,firstMonth,firstDay);
```

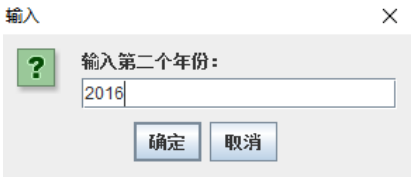
```
        long timeOne = calendar.getTimeInMillis();
        System.out.println(timeOne);
        calendar.set(secondYear,secondMonth,secondDay);
        long timeTwo = calendar.getTimeInMillis();
        Date date1=new Date(timeOne);
        Date date2=new Date(timeTwo);
        if(date2.equals(date1)){
           System.out.println("两个日期相同");
        }else if(date2.after(date1)){
           System.out.println("第二个日期大");
        }else if(date2.before(date1)){
           System.out.println("第一个日期大");
        }
        long days=(timeOne-timeTwo)/(1000*60*60*24);
        System.out.println(firstYear+"年"+firstMonth+"月"+firstDay+"日");
        System.out.println(secondYear+"年"+secondMonth+"月"+secondDay+"日");
        System.out.println("相隔天数 "+days);
    }
}
```

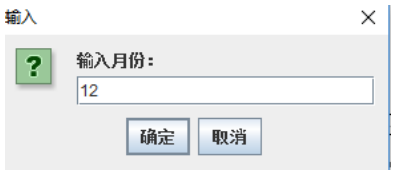
要求：运行程序，给出正确的程序运行结果，分析程序的功能。

【实验结果与分析】

实验结果:








```
1496566670503
第一个日期大
2017年5月4日
2016年12月3日
相隔天数 152
```

结果分析:

首先先介绍下 JOptionPane 类、Calendar 类、Date 类

JOptionPane 类是一个封装好的提示框类,有 4 个用于显示对话框的静态方法:

showMessageDialog://显示一条消息并等待用户 OK

showConfirmDialog://显示一条消息并等待用户确认

showOptionDialog: //显示一条消息并获得一组选项的选择
showInputDialog: //显示一条消息并获得输入的一行文本

Java 中提供了 Calendar 这个专门用于对日期进行操作的类，那么这个类有什么特殊的地方呢，首先我们来看 Calendar 的声明：
public abstract class Calendar extends Objectimplements Serializable, Cloneable, Comparable<Calendar>
该类被 abstract 所修饰，说明不能通过 new 的方式来获得实例，对此，Calendar 提供了一个类方法 getInstance，以获得此类型的一个通用的对象，getInstance 方法返回一个 Calendar 对象（该对象为 Calendar 的子类对象），其日历字段已由当前日期和时间初始化：
Calendar rightNow = Calendar.getInstance();
常用的方法如下:
Calendar calendar = Calendar.getInstance();//初始化当前时间为日历
calendar.set(2008, 8, 8,17, 35, 44);//设置日期 2008 年 8 月 8 日 17 点 35 分 44 秒
int year = calendar.get(Calendar.YEAR);// 获取年
int month = calendar.get(Calendar.MONTH) + 1; // 获取月,这里需要需要月份的范围为 0~11,因此获取
月份的时候需要+1 才是当前月份值
int day = calendar.get(Calendar.DAY_OF_MONTH);// 获取日
int hour = calendar.get(Calendar.HOUR);// 获取时 ,12 小时表示
int hour = calendar.get(Calendar.HOUR_OF_DAY); // 获取时 ,24 小时表示
int minute = calendar.get(Calendar.MINUTE);// 获取分
int second = calendar.get(Calendar.SECOND);// 获取秒
int weekday = calendar.get(Calendar.DAY_OF_WEEK);//获取星期,英语国家星期从星期日开始计算
calendar.add(Calendar.YEAR, 1);//日历变成一年后的今天
calendar.add(Calendar.DATE, -1);//日历变成今天的前一天
long time=calendar.getTimeInMillis();//距历元(即格林威治标准时间 1970 年 1 月 1 日的 00:00:00.000,格里高利历)的偏移量(毫秒)。

Date 为早期 jdk 使用的日期类,使用 UTC(国际协调时间)1970 年 1 月 1 日零时开始经过的毫秒数来保存时间。
原型如下:
public class Date extends Object implements Serializable, Cloneable, Comparable<Date>
常用方法如下:
Date d = new Date();//初始化为当前时间
Date d = new Date(long a);//初始化
Date d = new Date(2016-1900,8-1,28);//初始化为 2016.8.28 00:00:00
Date d = new Date("May 25,2004");//初始化为 2004.5.25
Date d = new Date(2004,4,25);/初始化为 2004.5.25
boolean a = d.after(Date when)//此日期是否在指定日期之后
Boolean a = d.before(Date when)//此日期是否在指定日期之前
int a = d.compareTo(Date anotherDate) //此方法比较两个日期的顺序,在当前日期前-1,和当前日期相等 0,之后 1

boolean a = d.equals(Object obj) //比较两个日期是否相等
long a = d.getTime() 此方法返回自 1970 年 1 月 1 日 00:00:00 GMT 此 Date 对象表示的毫秒数
int a = d.hashCode()//返回此对象的哈希码值
d.setTime(long time) //设置此 Date 对象 1970 年 1 月 1 日 00:00:00 GMT 以后,代表一个时间点 time 毫秒
String str = d.toString() //转换为形式的字符串
String str = (new SimpleDateFormat("yyyy-MM-dd HH:mm:ss:SSS")).format(d);//格式化输出

所以程序执行后,首先弹出输入对话框要求我们分别输入两组年份、月份、日期,然后以当前日期初始化日历类,用刚得到的整型常量调用 set 方法设置日历对象,然后将日历距离 1970 年 1 月 1 日零时开始经过的毫秒数返回存储在 timeOne 中并且输出。最后用 timeOne 和 timeTwo 分别初始化两个 data 对象,调用 data 对象中的比较方法判断两个日期的大小。最后将两个毫秒时间差求出相隔天数,分别输出两组时间和相隔天数。

5、编辑并运行下面程序，理解掌握 TextField 的使用

```
import java.applet.*;
import java.awt.*;
public class TestTextArea extends Applet{
TextArea text1,text2,text3,text4,text5;
public void init() {
  //创建多行文本框对象，初始字符串为请输入用户名，2 行 20 列
 text1=new TextArea ("请输入用户名：",2,20);
 text2=new TextArea (3, 30);//创建文本区对象，文本框行是 3 ，列数是 30
 text3=new TextArea ("请输入个人信息：",3, 20);
   text4=new TextArea (3, 30);
   text5=new TextArea ("登录",3, 20);
   add(text1);add(text2);add(text3); add(text4);add(text5);//添加 5 个多行文本框对象到 Applet 容器中
   text1.append("(英文)");  //在文本区 1 的请输入用户名后附加文本"(英文)"
   text5.insert("欢迎你" , 1);  //在文本区 5 指定位置 1 处查入"欢迎你"字符串
 }
}
```

要求：（1）运行程序，给出正确的程序运行结果，分析程序的功能。

【实验结果与分析】

实验结果:

结果分析:

首先介绍 Applet 容器

在 Java Applet 中，可以实现图形绘制，字体和颜色控制，动画和声音的插入，人机交互及网络交流等功能。 Applet 还提供了名为抽象窗口工具箱（Abstract Window Toolkit， AWT）的窗口环境开发工具。 AWT 利用用户计算机的 GUI 元素，可以建立标准的图形用户界面，如窗口、按钮、滚动条等等。

Applet 有 5 个 public 的 void 方法(Applet 的生命周期):

1. init()启动 Applet,浏览器总是调用 Applet 类的默认构造器生成对象,然后调用 init()方法经行初始化.一般在这个方法中生成 Applet 运行所需的对象并初始化 Applet 类的所有数据成员.
2. start()由浏览器调用.启动或者重新启动 Applet.当 Applet 第一此启动时,start 方法将紧跟 init()方法后被浏览器调用.如果用户离开当前的 HTML 页面后,在重新返回到当前 HTML 页面时,start()方法也会调用.start()方法一般用来启动 applet 需要的人和附加线程
3. paint(Graphics g) 在 init()方法执行结束,start()方法启动之后,就调用此方法画图.另外,每次需要重新绘制 Applet 时,也将调用此方法.本方法的典型应用,包括是使用 Applet 容器传递给 paint()方法 Graphics 对象 g 画图.
4. stop()当用户离开包含该 Applet 的 HTML 页面时,浏览器调用此方法.stop 方法被调用后,将立即停止所有在 start()方法中启动的操作.
5. destory() 在终止 Applet 运行时,调用 destory()方法,以便释放 Applet 占用的,由本地操作系统管理的任何系统资源.此方法执行之前,总是先调用 stop()方法.

然后介绍下 TextArea 的常用构造方法:

TextField text1 = TextArea() //初始化

TextField text1 = TextArea(String text) //初始化指定内容

TextField text1 = TextArea(int rows, int columns)//初始化指定行列数

TextField text1 = TextArea(String text, int rows, int columns)//初始化指定行列数和内容

所以本题我们定义了一个 Applet 的派生类,由浏览器调用默认构造方法生成一个 TextArea 对象,我们在 init 方法中对其进行初始化,创建五个文本区,初始化各自的大小,然后添加 5 个多行文本框对象到 Applet 容器中，调用 TextArea 中的方法,在文本区输出信息中添加或插入信息。

6、编译运行下面程序，理解 Java 委托事件处理机制和鼠标事件的处理

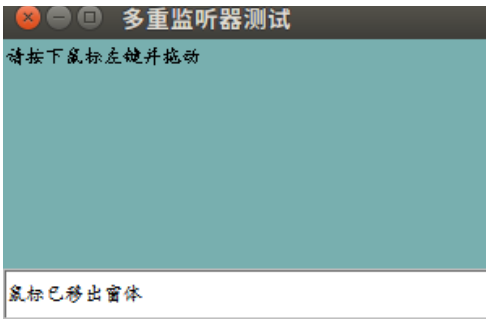
```
import java.awt.*;
import java.awt.event.*;
public class TestMultiListener implements MouseMotionListener, MouseListener {
```

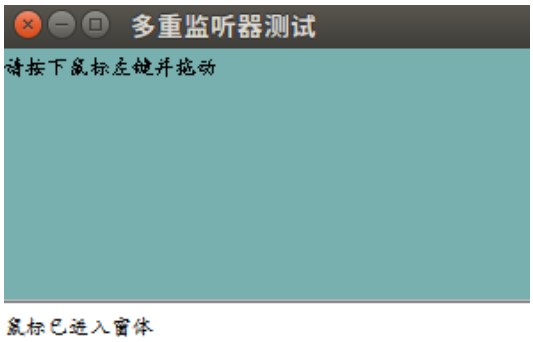
```
        Frame f = new Frame("多重监听器测试");  //创建窗口框架容器组件对象
        TextField tf = new TextField(30);//创建文本框对象，长度为 30 个字符
        public TestMultiListener(){//定义多重监听构造方法
                f.add(new Label("请按下鼠标左键并拖动"), "North");  //在窗口容器中添加标签
                f.add(tf, "South");//在窗口容器中添加文本框（底部）
                f.setBackground(new Color(120,175,175));//设置窗口背景色
        //在窗口容器中注册(添加)鼠标移动监听器（this 代表窗口实例），实现多重监听
                f.addMouseMotionListener(this);
        //在窗口容器中注册(添加)鼠标监听器（this 代表窗口），实现多重监听
                f.addMouseListener(this);
                f.setSize(300, 200);      //设置窗口容器大小
        f.setVisible(true);//设为可视
        }
        public static void main(String args[]) {
                TestMultiListener t = new TestMultiListener();//创建 TestMultiListener 类的对象实例
        }
public void mouseDragged(MouseEvent e) {//定义鼠标拖动方法
                String s = "鼠标拖动到位置（" + e.getX() + "," + e.getY() + ")"; //取得鼠标坐标
                tf.setText(s); //设置文本框显示鼠标拖动坐标
        }
        public void mouseEntered(MouseEvent e) {//定义鼠标进入窗体方法
                String s = "鼠标已进入窗体";
                tf.setText(s);//设置文本框显示鼠标已进入窗体字符串
        }
        public void mouseExited(MouseEvent e) { //定义鼠标退出窗体方法
                String s = "鼠标已移出窗体";
                tf.setText(s);
        }
        public void mouseMoved(MouseEvent e) { }//鼠标移动方法
        public void mousePressed(MouseEvent e) { } //鼠标按下方法
        public void mouseClicked(MouseEvent e) { }//鼠标点击方法
        public void mouseReleased(MouseEvent e) { }//鼠标释放组件方法
}
```

要求：编辑上面的程序，给出运行结果，并对程序进行分析，理解委托事件处理模型，理解掌握鼠标事件的处理。

【实验结果与分析】

实验结果:





结果分析：
首先我们定义了一个类 TestMultiListener,它同时继承了两个接口,为复合监听类。它的成员变量为一个窗口和一块文本框,在该类的构造函数中为窗口添加标签,文本框等设置,最重要的是它为这个窗口注册了两个监听器,可以监听鼠标产生的多组信息。接着我们定义了监听器的响应函数,去处理鼠标事件,并将处理结果在文本框中设置输出。

7、完成图 1 所示学籍管理主界面的设计与制作。 基本要求：如果输入用户名：admin，密码：123456，则弹出如图 2 所示窗口，否则弹出如图 3 所示窗口。

要求：按照要求编写程序，给出程序运行结果。

图 1 登录界面

图 2 登录界面

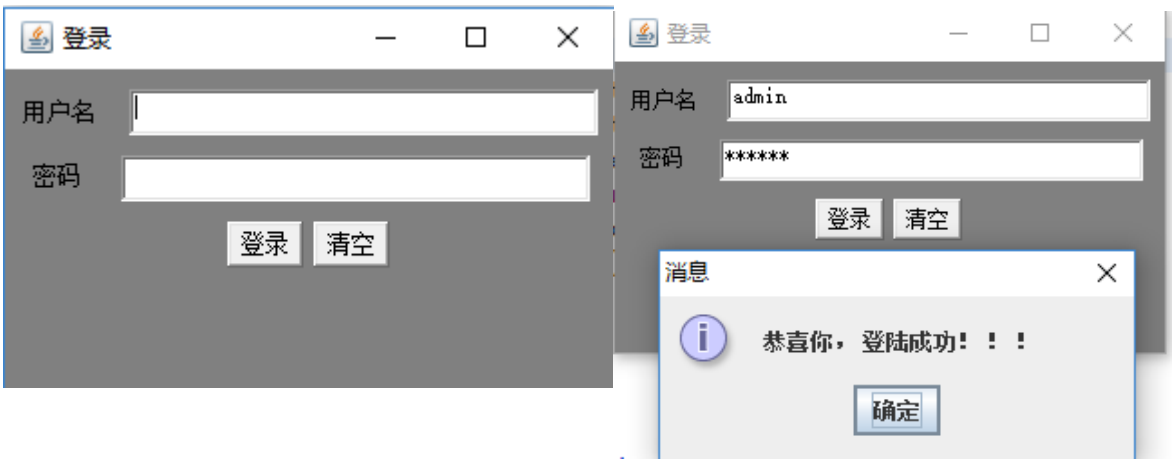
图 **3**

【实验结果与分析】

```
package b;

import java.awt.*;
import java.awt.event.*;
import javax.swing.JOptionPane;
public class Demo implements ActionListener{
	Frame fra = new Frame("登录");
	TextField id = new TextField(30);
	TextField pw = new TextField(30);
	Button b1=new Button("登录");
	Button b2=new Button("清空");
	public void display() {
			fra.setSize(320,200);	fra.setLocation(500, 500);
			fra.setBackground(Color.gray);
			pw.setEchoChar('*');
			fra.setLayout(new FlowLayout(FlowLayout.CENTER,5,10));
			fra.add(new Label("用户名"));	fra.add(id);
			fra.add(new Label("密码"));	fra.add(pw);
			fra.add(b1);	fra.add(b2);
			b1.addActionListener(this);
			b2.addActionListener(this);
			fra.addWindowListener(new WinClass());
			fra.setVisible(true);
	}
	public void actionPerformed(ActionEvent e) {
		if(e.getSource()==b2){
			id.setText("");
			pw.setText("");
		}
		else{
			String str1 = id.getText();
			String str2 = pw.getText();
			if(str1.equals("admin")&&str2.equals("123456") )
				JOptionPane.showMessageDialog(null, "恭喜你,登陆成功!!!");
			else
				JOptionPane.showMessageDialog(null, "登陆失败!!!");
		}
```

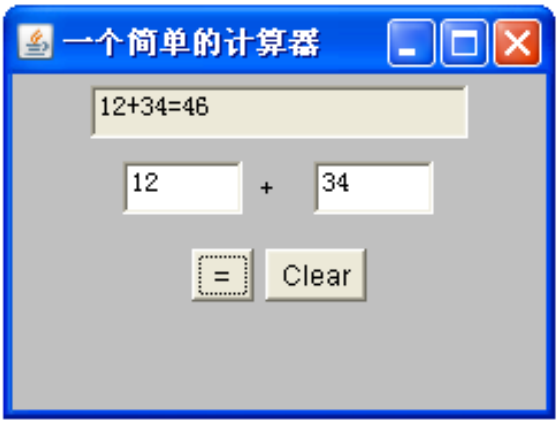
```
	}
	class WinClass extends WindowAdapter{
		public void windowClosing(WindowEvent e) {
			System.exit(0);
		}
	}
	public static void main(String[] args) {
		Demo a = new Demo();
		a.display();
	}
}
```

实验结果:



8、设计一个简单的加法器，在所讲解的例子基础上，添加一个“=”按钮，单击它可以计算文本行的加法算式，如果文本行不是一个合法的加法算式，要抛出一个异常！！如下图：

要求：按照要求编写程序，给出程序运行结果。



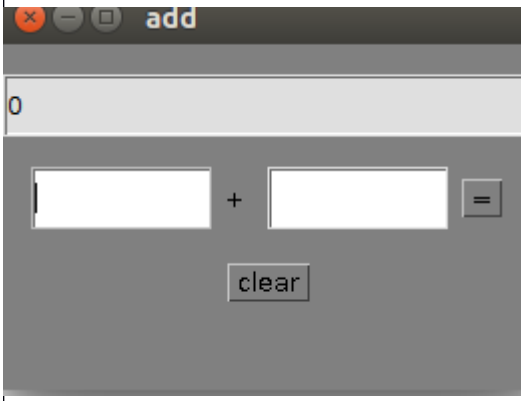
【实验结果与分析】

```
import java.awt.Button;
import java.awt.*;
import java.awt.event.*;
import javax.swing.JOptionPane;
public class Test implements ActionListener{
	Frame fra = new Frame("add");
	TextField result = new TextField(32);
	TextField num1 = new TextField(10);
	TextField num2 = new TextField(10);
	Button b1=new Button("=");
	Button b2=new Button("clear");
	public void display() {
		fra.setSize(270,200);  fra.setLocation(500, 500);
		fra.setBackground(Color.gray);
		result.setEditable(false);
		result.setText("0");
		fra.setLayout(new FlowLayout(FlowLayout.CENTER,5,15));
		fra.add(result);fra.add(num1);
		fra.add(new Label("+"));fra.add(num2);
		fra.add(b1);fra.add(b2);
		b1.addActionListener(this);b2.addActionListener(this);
```

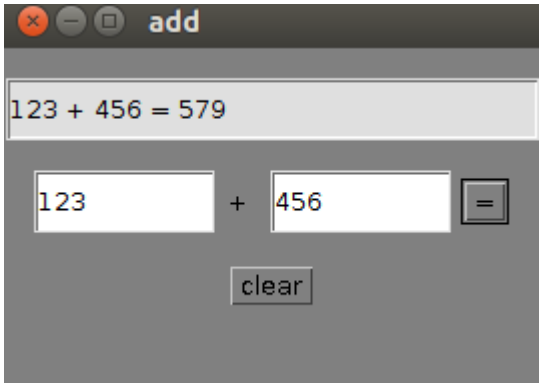
```
            fra.addWindowListener(new WinClass());fra.setVisible(true);
      }
      public void actionPerformed(ActionEvent e) {
            if(e.getSource()==b2){
                  num1.setText("");
                  num2.setText("");
                  result.setText("0");
            }
            else{
                  try{
                        String str1 = num1.getText();
                        String str2 = num2.getText();
                        int op1 = Integer.parseInt(str1);
                        int op2 = Integer.parseInt(str2);
                        long sum = op1 + op2;
                        result.setText(str1+" + "+str2+" = "+String.valueOf(sum));
                  }catch(NumberFormatException ex) {
                        JOptionPane.showMessageDialog(null, "请重新输入!", "格式非法",
JOptionPane.ERROR_MESSAGE);
                  }
            }
      }
      class WinClass extends WindowAdapter{
            public void windowClosing(WindowEvent e) {
                  System.exit(0);
            }
      }
      public static void main(String[] args) {
            Test tt = new Test();
            tt.display();
      }
}
```

实验结果：

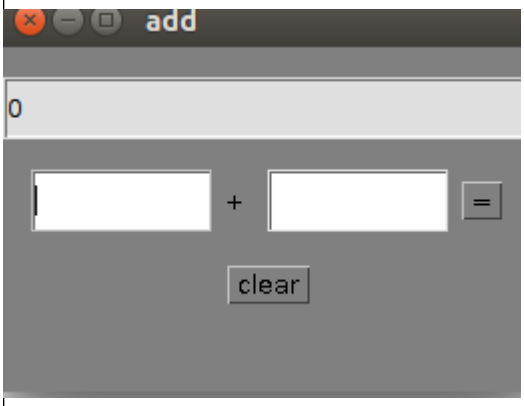
初始的时候：



计算后：


add
123 + 456 = 579
123
+
456
=
clear

清空：


add
0
+
=
clear

异常处理：


add
0
76765757657
+
12
=
clear
格式非法
请重新输入！
OK