武汉大学计算机学院

本科生课程设计报告

计算器程序的设计

专 业 名 称　：软件工程

课 程 名 称　：Windows 程序设计

学 生 学 号　：2016302580062

学 生 姓 名　：周彪

二〇一八年 12 月

郑 重 声 明

本人呈交的设计报告，是独立进行实验工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本设计报告不包含他人享有著作权的内容。对本设计报告做出贡献的其他个人和集体，均已在文中以明确的方式标明。本设计报告的知识产权归属于培养单位。

本人签名：_____　　日期：_____

1. 实验目的和意义

1.1. 实验目的

本实验采用 c#语言实现对计算器简单的实现,主要包括以下功能:

（1） 实现加、减、乘、除四种基本运算功能,还附加取模,平方,立方,根号,阶乘,倒数等运算.
（2） 上述四种运算均使用 dll 实现，其中 dll 使用 c#编程语言来创建.
（3） 上述四种运算输入、操作和运算结果均要求直接在界面中进行显示，其中界面选用 Winform 技术实现.
（4） 上述四种运算的实现均输入数据兼容数据类型 int、double，输出结果也兼容数据类型 int、double.
（5） 对于输入的非法数据或无效数据要进行相关提示.
（6） 对于输入的数据有回退删除功能.
（7） 能实现带括号的表达式运算.

1.2. 实验意义

本次实验采用 c#语言以及 dll 封装,能够更多的了解关于 dll 封装的重要性以及实用性,在以后的编程学习之中能够更多地注意到程序的模块化以及复用性,更多地规范自己的编程习惯.

2. 实验设计

2.1. 概述

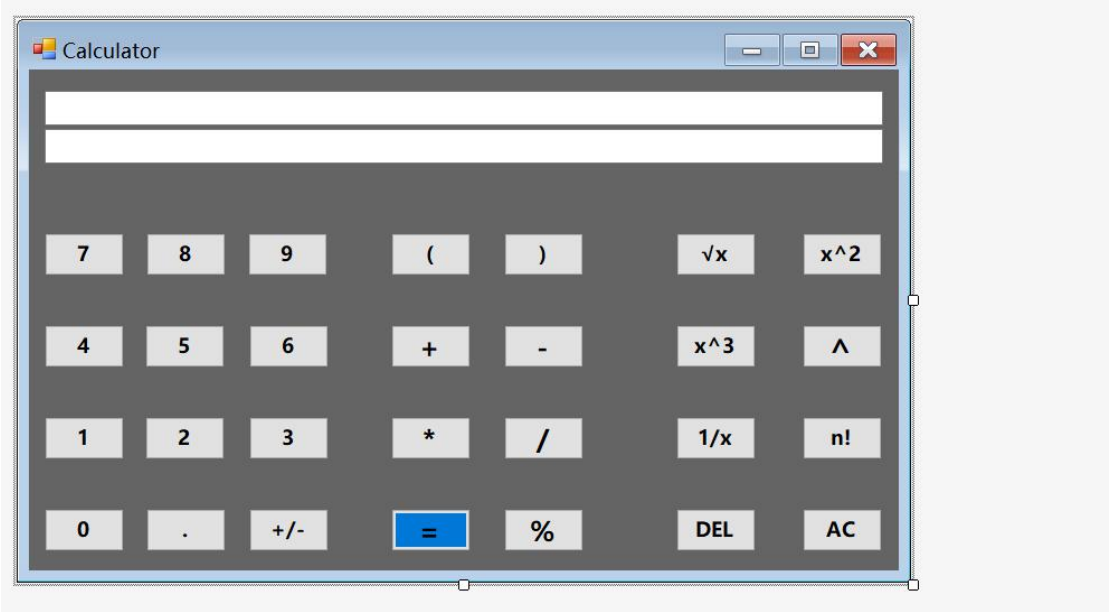
本次实验以c#语言为基础,采用winform技术设计界面,dll文件实现程序模块化,以及栈的变化来实现表达式的计算.

实验环境:vs2015

2.2. 实验方案

(1) 界面设计

打开VS2015,创建c#项目,创建myCalculator.Design.cs文件,进行界面设计,如图:



用到的控件:button,Textbox.

(2) Dll文件设计.

建立myexpression类,来进行表达式处理,包括加减乘除运算,取模,平方,开方,立方,倒数,阶乘等运算.利用栈来处理表达式.

主要内容:

```
namespace myExprseeion
{
    0 个引用
    public class myExpression//表达式计算,栈处理
    {
        定义栈

        判断是否为数字

        定义运算符优先级

        比较运算符的优先级

        调用getResult将最后结果显示

        getResult返回运算结果
    }
}
```

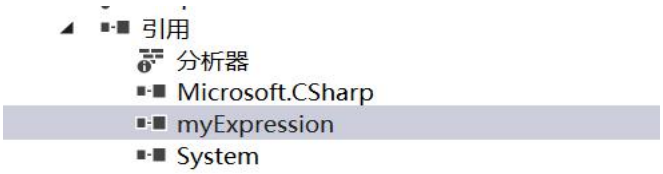
定义栈:定义符号栈和数字栈

定义运算符优先级:

```
public static int priority(String Operator)
{
    if (Operator == ("+") || Operator == ("-"))
    {
        return 1;
    }
    else if (Operator == ("*") || Operator == ("/") || Operator == ("%"))
    {
        return 2;
    }
    else if (Operator == ("^"))
    {
        return 3;
    }
    else
    {
        return 0;
    }
}
```

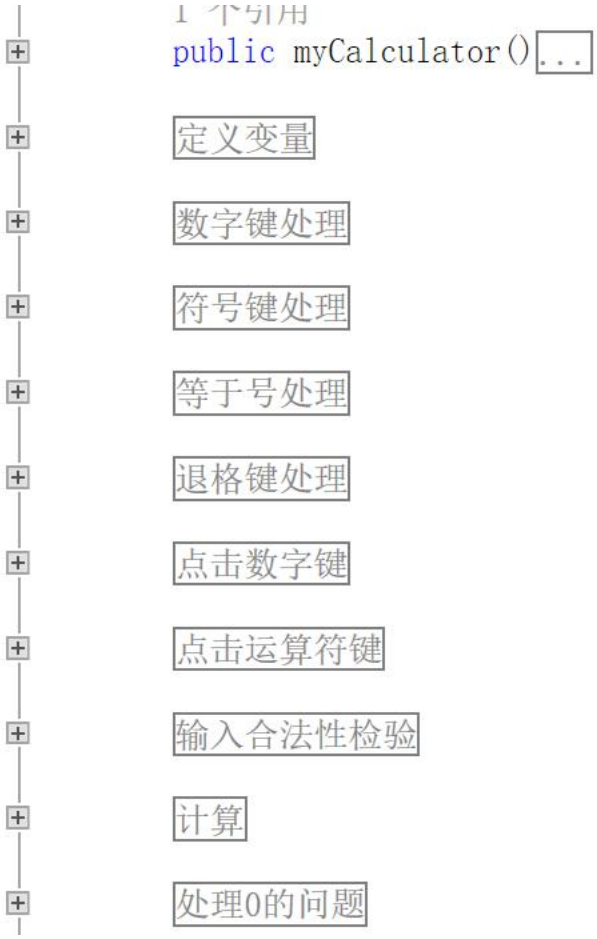
显示结果:通过栈处理(先入后出),根据优先级表示,将表达式转化成前缀表达式,实现表达式的计算,具体程序见附录.

之后转化成 dll 文件,在新项目下引用.



(3) 实现计算器交互.

通过界面上的按钮实现表达式的输入,结果的输出,完善计算器功能.



数字键处理:数字输入之前不能有右括号,数字输入之后最后一个数字不能是运算符.
符号键处理:判断符号是运算符还是小数点还是括号,分别进行处理.
等于号处理:合法性检验之后,引用 dll 文件 myExpression 类,实现运算并显示正确结果.
退格键处理:删除一个字符或者全部删除.
点击运算符键:处理不同的运算符键并显示相应的效果.
输入合法性检验:包括是否构成表达式,表达式是否逻辑错误等等.
处理 0 的问题:主要包括除数不能为 0,小数点后面的 0 处理以及出现多个 0 的情况.

附录.

实现表达式的运算,产生 dll 文件.

```
using System;
using System.Text.RegularExpressions;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Linq.Expressions;
using myExpression;

namespace Calculator
{
    public partial class myCalculator : Form
    {

        public myCalculator()
        {
            InitializeComponent();
        }

        #region  定义变量
        Regex RegNum = new Regex(@"^[0-9\.]*$");       //数字的正则表达式
        Regex RegBegin = new Regex(@"^[0-9\-(]");     //第一个字符的正则表达式
        Regex RegMark = new Regex(@"[-+*/^\()=.]");         //运算符的正则表达式
        Regex RegAll = new Regex(@"[0-9\-(\-+*/^\()=.SCT]");//所有运算符、数字的正则表
达式

        bool Start = true;           //判断是否为第一个字符
        bool IsMark = true;          //判断是否可以输入数字，为真，允许输入数字
        bool IsRight = false;      //判断最后一个字符是否为右括号
        bool IsDot = false;          //判断是否可以输入小数点
        int BeforeLen = 0;             //判断输入运算符之前的字符串长度
        List<bool> LeftList = new List<bool>(); //存储左括号个数
        string RealStr = string.Empty;
        string FinalResult = string.Empty;
        #endregion

        #region  数字键处理
        private void PressNumBtn(string NumTxt)
```

```
{
    if (IsRight == false)//数字键输入之前紧前不能有右括号
    {
        if (!this.InputText.Text.Contains("="))
        {
            this.InputText.Text += NumTxt;
            IsMark = false;//输入数字后，则最后一个字符不是运算符
            //IsDot = false;
        }
        else
        {
            this.InputText.Text = string.Empty;
            this.InputText.Text = this.FinalResult + NumTxt;
            IsMark = false;
        }
    }
    this.GetFocus();
}
#endregion

#region 符号键处理
private void PressMathBtn(string MarkTxt)
{
    if (MarkTxt != "." && IsMark == false)
    {
        if (FinalResult == string.Empty)
        {
            this.InputText.Text += MarkTxt;
        }
        else
        {
            if (this.InputText.Text.Contains("="))
            {
                this.InputText.Text = FinalResult;
            }
            else
            {
                this.InputText.Text += MarkTxt;
            }
        }
        IsMark = true;//输入了运算符，最后一个字符就是运算符了
        IsDot = false;//输入了运算符，最后一个字符就不是小数点了
        IsRight = false;//输入了运算符，最后一个字符就不是右括号了
        Start = false;
```

```
                this.ZeroTwoChk(this.InputText.Text.Trim(), BeforeLen);//处理 0
            }
            else
            {
                if (IsMark == false)
                {
                    if (FinalResult == string.Empty)
                    {
                        this.InputText.Text += MarkTxt;
                    }
                    else
                    {
                        if (this.InputText.Text.Contains("="))
                        {
                            this.InputText.Text = FinalResult;
                        }
                        else
                        {
                            this.InputText.Text += MarkTxt;
                        }
                    }
                    IsMark = true;//输入了运算符，最后一个字符就是运算符了
                    IsDot = false;//输入了运算符，最后一个字符就不是小数点了
                    IsRight = false;//输入了运算符，最后一个字符就不是右括号了
                    Start = false;
                }
            }
            this.GetFocus();
        }
        #endregion

        #region 等于号处理
        private void Equal()
        {
            try
            {
                if (this.InputText.Text.Contains("/0"))     //除数不能为 0
                {
                    if (this.InputText.Text.Contains("/0.")) { }         //去掉除号后为小数的
情况
                    else
                    {
                        this.OutputText.Text = "除数不能为零,请重新输入!";
                        this.InputText.Text = "";
```

```
                }
            }
            else if (this.InputText.Text.Contains("="))
            {
                this.OutputText.Text = FinalResult;
            }
            else if (IsMark == false)
            {
                RealStr = this.InputText.Text.Trim() + "=";
                this.ZeroTwoChk(RealStr, BeforeLen);
                RealStr = this.InputText.Text.Trim();//移除前面和后面空白字符
                if (RealStr.StartsWith("-"))              //开始为负号时
                {
                    RealStr = "0" + RealStr;//前面加个 0
                }
                this.Calculate();
                IsRight = false;
            }
            this.GetFocus();
        }
        catch
        {
            this.OutputText.Text = "Error!";
        }
    }
    #endregion

    #region  退格键处理
    private void Del()
    {
        string Txt = this.InputText.Text;
        if (Txt.Length >= 1)
        {
            string L = Txt.Substring(Txt.Length - 1, 1);       //取前一个字符
            if (L == "(")
            {
                this.LeftList.Remove(true);         //左括号列表的记录删除一个
                this.InputText.Text = Txt.Remove(Txt.Length - 1, 1);
            }
            else if (L == ")")
            {
                this.LeftList.Add(true);       //左括号列表的记录增加一个
                this.InputText.Text = Txt.Remove(Txt.Length - 1, 1);
                IsRight = false;
```

```
                }
                else
                    this.InputText.Text = Txt.Remove(Txt.Length - 1, 1); //删除最后一个字
符
                IsMark = false;
            }
        }
        #endregion

        #region 点击数字键
        private void btn1_Click(object sender, EventArgs e)
        {
            PressNumBtn(btn1.Text);
            this.OutputText.Text = "";
        }

        private void btn2_Click(object sender, EventArgs e)
        {
            PressNumBtn(btn2.Text);
            this.OutputText.Text = "";
        }

        private void btn3_Click(object sender, EventArgs e)
        {
            PressNumBtn(btn3.Text);
            this.OutputText.Text = "";
        }

        private void btn4_Click(object sender, EventArgs e)
        {
            PressNumBtn(btn4.Text);
            this.OutputText.Text = "";
        }

        private void btn5_Click(object sender, EventArgs e)
        {
            PressNumBtn(btn5.Text);
            this.OutputText.Text = "";
        }

        private void btn6_Click(object sender, EventArgs e)
        {
            PressNumBtn(btn6.Text);
            this.OutputText.Text = "";
```

```
}

private void btn7_Click(object sender, EventArgs e)
{
    PressNumBtn(btn7.Text);
    this.OutputText.Text = "";
}

private void btn8_Click(object sender, EventArgs e)
{
    PressNumBtn(btn8.Text);
    this.OutputText.Text = "";
}

private void btn9_Click(object sender, EventArgs e)
{
    PressNumBtn(btn9.Text);
    this.OutputText.Text = "";
}

private void btn0_Click(object sender, EventArgs e)
{
    PressNumBtn(btn0.Text);
    this.OutputText.Text = "";
}
#endregion

#region  点击运算符键
private void btnAc_Click(object sender, EventArgs e)        //清空
{
    this.InputText.Text = string.Empty.Trim();
    this.OutputText.Text = string.Empty.Trim();
    Start = true;
    IsMark = true;
    IsRight = false;
    IsDot = false;
    BeforeLen = 0;
    LeftList.Clear();
    RealStr = string.Empty;
    FinalResult = string.Empty;

}
private void btnDot_Click(object sender, EventArgs e)           //点
{
```

```
        string Txt = this.InputText.Text;
        if (IsDot == false &&    Txt != "")
        {
            this.PressMathBtn(btnDot.Text);
            IsDot = true;
        }
        else
        {
            this.OutputText.Text = "非法输入!";
        }
        this.GetFocus();
}

private void btnDel_Click(object sender, EventArgs e)      //退格
{
        Del();
}
private void btnDs_Click(object sender, EventArgs e)      //倒数
{
        try
        {
            if (this.InputText.Text != "0")
            {
                double a = Convert.ToDouble(this.InputText.Text);
                this.InputText.Text = (1 / a).ToString();
            }
            else
            {
                this.OutputText.Text = "非法输入!";
            }
        }
        catch
        {
            this.OutputText.Text = "Error!";
        }
}
private void btnSqrt_Click(object sender, EventArgs e)          //  开方
{
        try
        {
            double a = Convert.ToDouble(this.InputText.Text);
            if (a >= 0)
                this.InputText.Text = (Math.Sqrt(a)).ToString();
            else
```

```
                this.OutputText.Text = "输入的数字应为非负数!";
        }
        catch
        {
            this.OutputText.Text = "非法输入!";
        }

}
private void btnJc_Click(object sender, EventArgs e)        //     阶乘
{
        try
        {
            if (this.InputText.Text.Contains("."))
            {
                this.OutputText.Text = "格式错误!";
            }
            else if (this.InputText.Text.Contains("-"))
            {
                this.OutputText.Text = "格式错误!";
            }
            else
            {
                int a = int.Parse(this.InputText.Text);
                long result = 1;
                for (int i = 1; i <= a; i++)
                {
                    result = result * i;
                }
                this.InputText.Text = result.ToString();
            }
        }
        catch
        {
            this.OutputText.Text = "Error!";
        }

}
private void btnPow2_Click(object sender, EventArgs e)//平方
{
        PressMathBtn("^");
        PressNumBtn("2");

}
private void btnPow3_Click(object sender, EventArgs e)//立方
```

```
        {
            PressMathBtn("^");
            PressNumBtn("3");
        }
        private void btnPow_Click(object sender, EventArgs e)//     乘方
        {
            PressMathBtn(btnPow.Text);
        }
        private void btnLeft_Click(object sender, EventArgs e)          //左括号
        {
            string Txt = this.InputText.Text;
            if (Txt.Trim() == string.Empty || (!RegNum.IsMatch(Txt.Substring(Txt.Length - 1,
1)) && Txt.Substring(Txt.Length - 1, 1) != ")" && IsDot == false) )
            {
                this.InputText.Text = this.InputText.Text + "(";
                IsMark = true;//输入了左括号，最后一个字符就是运算符了
                this.LeftList.Add(true);
            }
            this.GetFocus();
        }
        private void btnRight_Click(object sender, EventArgs e)         //右括号
        {
            try
            {
                string Txt = this.InputText.Text;
                if (LeftList.Contains(true) && (RegNum.IsMatch(Txt.Substring(Txt.Length -
1, 1)) || Txt.Substring(Txt.Length - 1, 1) == ")"))
                {
                    this.PressMathBtn(")");
                    this.LeftList.Remove(true);//配对去掉一个左括号
                    IsMark = false;//输入了右括号，最后一个字符不是运算符了
                    IsRight = true;//输入了右括号，最后一个字符是右括号
                }
                this.GetFocus();
            }
            catch
            {
                this.OutputText.Text = "Error!";
            }
        }
        private void btnEqual_Click(object sender, EventArgs e)//    等于号
        {
            this.Equal();
        }
```

```
private void btnDiv_Click(object sender, EventArgs e)//除
{
    PressMathBtn(btnDiv.Text);
}
private void btnMul_Click(object sender, EventArgs e)//乘
{
    PressMathBtn(btnMul.Text);
}
private void btnSub_Click(object sender, EventArgs e)//减
{
    PressMathBtn(btnSub.Text);
}
private void btnPlus_Click(object sender, EventArgs e)//加
{
    PressMathBtn(btnPlus.Text);
}
private void btnFu_Click(object sender, EventArgs e)//负数
{
    this.InputText.Text += "-";
}
private void btnMod_Click(object sender, EventArgs e)//取模
{
    PressMathBtn(btnMod.Text);
}
#endregion

#region 输入合法性检验
private string CheckLegal(string StrInput, bool IsStart, List<bool> List)
{
    if (RegAll.IsMatch(StrInput))
    {
        if (IsStart == true)
        {
            if (RegBegin.IsMatch(StrInput))
            {
                Start = false;
                return "GoOn";          //输入的第一个字符合格，继续
            }
            else
            {
                return "RollBack";          //输入的第一个字符不合格则，输入无
效

            }
```

```
                }
                else
                {
                    if (StrInput.Contains("="))
                    {
                        if (List.Contains(true))
                        {
                            return "LeftProblem";    //括号没有完全，则返回“左括号问题”，应用于等号按键的按下
                        }
                        else
                        {
                            return "Over";          //算式合符规范，返回计算结束
                        }
                    }
                    else
                    {
                        return "GoOn";          //不是等号，则继续
                    }
                }
            }
            else
            {
                return "RollBack";          //其他情况，输入无效
            }
        }
        #endregion

        #region 计算
        private void Calculate()
        {
            if (this.CheckLegal(RealStr, Start, LeftList) == "Over")
            {
                myExp Exp = new myExp();
                FinalResult = Exp.GetExpression(RealStr.Replace("=", ""));//获得最终结果
                if (FinalResult == "Error")
                {
                    FinalResult = "Error";
                }
                else
                {
                    this.InputText.Text = FinalResult;                  //显示最后结果
                    if (FinalResult.Contains("."))
                    {
```

```
                    IsDot = true;
                }
                if (RealStr.StartsWith("0-"))
                {
                    this.InputText.Text = this.InputText.Text.TrimStart('0');
                }
                this.InputText.Refresh();
            }
        }
        else if (this.CheckLegal(RealStr, Start, LeftList) == "RollBack")
        {
            FinalResult = "Error";
        }
        else if (this.CheckLegal(RealStr, Start, LeftList) == "LeftProblem")
        {
            FinalResult = "Error";
        }
    }
    #endregion

    #region 处理 0 的问题
    private void ZeroTwoChk(string inPutTxt, int beforeLen)      //处理 0 的问题，如两个
0 和 6.30 等
    {
        try
        {
            string lastMark = inPutTxt.Trim().Substring(inPutTxt.Trim().Length - 1, 1);
            if (beforeLen != 0)
            {
                string beforeStr = inPutTxt.Trim().Substring(0, beforeLen);
                string nowStr = inPutTxt.Trim().Substring(beforeLen, inPutTxt.Length -
beforeLen - 1);
                if (nowStr.Contains("."))
                {
                    nowStr = nowStr.Trim('0');
                    if (nowStr.StartsWith(".") && !nowStr.EndsWith("."))
                    {
                        nowStr = "0" + nowStr;
                    }
                    if (nowStr == ".")
                    {
                        nowStr = "0";
                    }
                    if (nowStr.EndsWith("."))
```

```
            {
                nowStr = nowStr.TrimEnd('.');
            }
        }
        else
        {
            if (nowStr != string.Empty)
            {
                nowStr = nowStr.TrimStart('0');
                if (nowStr == string.Empty)
                {
                    nowStr = "0";
                }
            }
        }
        this.InputText.Text = beforeStr + nowStr + lastMark;
    }
    else
    {
        inPutTxt = inPutTxt.Trim().Substring(0, inPutTxt.Length - 1);
        if (inPutTxt.Contains("."))
        {
            inPutTxt = inPutTxt.Trim('0');
            if (inPutTxt.StartsWith(".") && !inPutTxt.EndsWith("."))
            {
                inPutTxt = "0" + inPutTxt;
            }
            if (inPutTxt == ".")
            {
                inPutTxt = "0";
            }
            if (inPutTxt.EndsWith("."))
            {
                inPutTxt = inPutTxt.TrimEnd('.');
            }
        }
        else
        {
            inPutTxt = inPutTxt.TrimStart('0');
            if (inPutTxt == string.Empty)
            {
                inPutTxt = "0";
            }
        }
```

```
                    this.InputText.Text = inPutTxt + lastMark;
                }
            }
            catch { }
            BeforeLen = this.InputText.Text.Trim().Length;
        }
        #endregion

        #region 将焦点定位在最后一个字符之后
        private void GetFocus()
        {
            if (this.InputText.ContainsFocus == false)
            {
                this.InputText.Focus();
                this.InputText.ScrollToCaret();
                this.InputText.SelectionStart = this.InputText.TextLength;
            }
        }
        #endregion


    }
}
```