**程 序 综 合 实 验**

**实 验 报 告**

**班级： 软件172**

**学号： 174402**

**姓名： 吴帅龙**

**实验一 计算器设计与实现|简单文本编辑器设计与实现**

1实验采用的数据结构设计和算法设计（可采用但不限于使用流程图），程序每部分的主要功能说明等

a：采用的数据结构和算法, 主要使用了列表, 栈和字典.

列表使用了List<T> class, 栈使用了Stack<T> class, 字典使用了 Dictionary<TKey, TValue> class。

其中，列表用于后缀表达式的存放，栈结构用于中缀表达式转为后缀表达式 和 利用后缀表达式求结构 时的辅助结构，字典用于运算符优先级的表示。

b: 程序各部分的主要功能

private void On[0-9](object sender, EventArgs e)表示按钮0到9对应的函数，主要用于在显示框中追加字符，其它按钮功能类似。

private void Ondelete(object sender, EventArgs e)用于删除一个字符

private void Onequal(object sender, EventArgs e)等于号对应的按钮，主要用于计算

结果。

private void Caculate(string numbers)计算函数，在Onequal中被调用。

2、程序代码，实验实现的源程序。

using System;

using System.Windows.Forms;

using System.Text.RegularExpressions;

using System.Collections.Generic;

namespace WindowsFormsApp2

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void On1(object sender, EventArgs e)

{

this.textBox1.Text += "1";

}

private void On2(object sender, EventArgs e)

{

this.textBox1.Text += "2";

}

private void On3(object sender, EventArgs e)

{

this.textBox1.Text += "3";

}

//对中缀表达式进行求值

private void Caculate(string numbers)

{

//2.把中缀表达式转换为后缀表达式

string[] list = Regex.Split(numbers, " ");

List<string> exp = new List<string>();

Stack<string> stack = new Stack<string>();

Dictionary<string, int> priority = new Dictionary<string, int>();

priority.Add("(", 1);

priority.Add("+", 3);

priority.Add("-", 3);

priority.Add("\*", 5);

priority.Add("/", 5);

double temp1=0, temp2=0, temp3=0;

foreach(string str in list)

{

//list[i]是数字

if(str!="+" && str!="-" && str!="\*" && str!="/" && str!="(" && str!=")")

{

exp.Add(str);

}

else if(stack.Count==0 || str == "(")

{

stack.Push(str);

}

else if(str == ")")

{

while (!(stack.Count==0) && stack.Peek() != "(")

{

exp.Add(stack.Pop());

}

if(stack.Count == 0)

{

this.textBox1.Text = "括号匹配错误";

return;

}

stack.Pop();

}

else

{

while(!(stack.Count == 0) && (priority[stack.Peek()] >= priority[str]))

{

exp.Add(stack.Pop());

}

stack.Push(str);

}

}

while(!(stack.Count == 0))

{

if(stack.Peek() == "(")

{

this.textBox1.Text = "括号匹配错误";

return;

}

exp.Add(stack.Pop());

}

//后缀表达式求值

Stack<double> stack2 = new Stack<double>();

foreach (string str in exp)

{

if(str == " " ||str == "")

{

Console.WriteLine("这是空的字符");

continue;

}

if(str != "+" && str != "-" && str != "\*" && str != "/")

{

stack2.Push(Convert.ToDouble(str));

continue;

}

if(stack2.Count < 2)

{

this.textBox1.Text = "输入错误1";

return;

}

temp1 = stack2.Pop();

temp2 = stack2.Pop();

if(str == "+")

{

temp3 = temp2 + temp1;

}

else if(str == "-")

{

temp3 = temp2 - temp1;

}

else if(str == "\*")

{

temp3 = temp2 \* temp1;

}

else if(str == "/")

{

if(temp1 == Convert.ToDouble(0))

{

this.textBox1.Text = "除数不能为0";

return;

}

temp3 = temp2 / temp1;

}

stack2.Push(temp3);

}

if(stack2.Count == 1)

{

this.textBox1.Text = stack2.Pop().ToString();

}

else

{

this.textBox1.Text = "输入错误！";

}

}

//显示结果

private void show(object sender, EventArgs e)

{

}

//加法

private void Onadd(object sender, EventArgs e)

{

this.textBox1.Text += "+";

}

private void Form1\_Load(object sender, EventArgs e)

{

}

//等于

private void Onequal(object sender, EventArgs e)

{

//1、先把中缀表达式转换为列表

string numbers = string.Empty;

int i;

char s;

for (i = 0; i < this.textBox1.Text.Length; i++)

{

s = this.textBox1.Text[i];

if (s == ' ')

{

continue;

}

//s是数字

if (s >= '0' && s <= '9')

{

numbers += s.ToString();

}

//s是运算符

else

{

numbers += " " + s + " ";

}

}

//去除字符串首位的空格

numbers = numbers.Trim();

//返回运算结果到前端界面

Caculate(numbers);

//this.textBox1.Text = result.ToString();

}

//减法

private void Onminus(object sender, EventArgs e)

{

this.textBox1.Text += "-";

}

private void button7\_Click(object sender, EventArgs e)

{

}

private void On5(object sender, EventArgs e)

{

this.textBox1.Text += "5";

}

private void On4(object sender, EventArgs e)

{

this.textBox1.Text += "4";

}

private void On6(object sender, EventArgs e)

{

this.textBox1.Text += "6";

}

//乘法

private void Onmultiplicative(object sender, EventArgs e)

{

this.textBox1.Text += "\*";

}

//除法

private void Ondevides(object sender, EventArgs e)

{

this.textBox1.Text += "/";

}

private void On7(object sender, EventArgs e)

{

this.textBox1.Text += "7";

}

private void button16\_Click(object sender, EventArgs e)

{

}

private void On8(object sender, EventArgs e)

{

this.textBox1.Text += "8";

}

private void On9(object sender, EventArgs e)

{

this.textBox1.Text += "9";

}

private void On0(object sender, EventArgs e)

{

this.textBox1.Text += "0";

}

private void Clear(object sender, EventArgs e)

{

this.textBox1.Text = "";

}

private void Ondot(object sender, EventArgs e)

{

this.textBox1.Text += ".";

}

//左括号

private void Onleft(object sender, EventArgs e)

{

this.textBox1.Text += "(";

}

//右括号

private void Onright(object sender, EventArgs e)

{

this.textBox1.Text += ")";

}

//

private void Ondelete(object sender, EventArgs e)

{

string str = "";

int i = 0;

for(i=0; i<this.textBox1.Text.Length-1; i++)

{

str = str + this.textBox1.Text[i];

}

this.textBox1.Text = str;

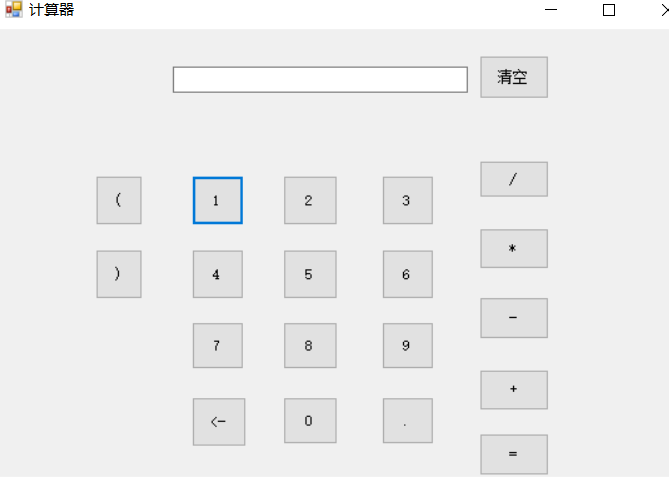
}

}

}

3、实验结果，程序运行结果截图及各模块功能说明。

界面：



除0报错

混合运算的优先级

删除键

4、实验总结，总结实验过程及取得实验成果等收获，分析不足之处

实验的过程和成果

1. 在完成实验过程中，我想用c++中的STL。我找到它的API并导入命名空间后，发现导入错误。原因是要在右上角的引用添加上你要using的namespace。

2在使用List, Stack, Dictionary class 时，定义变量后必须为它初始化。但不要把他们初始化为null。如果这样的话后面调用该对象的方法会报错。正确做法是为他们new一块空间，如: Stack<double> stack = new Stack<double>();

3在c和c++中 “”和‘’ 是完全不一样的，前一个是字符串，后一个是单个字符。但是在有些语言中(如Python)是一样的。

4使用foreach()时要注意循环对象，如果对栈结构进行迭代则从栈顶开始返回元素。

5在实验的最后，利用后缀表达式进行求值时，把每个数字字符串转为double类型，即

Convert.ToDouble(str)时报了 [FormatException](https://docs.microsoft.com/zh-cn/dotnet/api/system.formatexception)的错误，调试后发现str可能是空字符，用if对空字符进行过滤，完美解决。

实验不足：

1. 查找函数API的时间过长，有时不如自己实现要用的结构。
2. 计算器只实现了混合运算的优先级运算，还不能输入小数
3. 计算器还不支持负号及其他复杂运算。