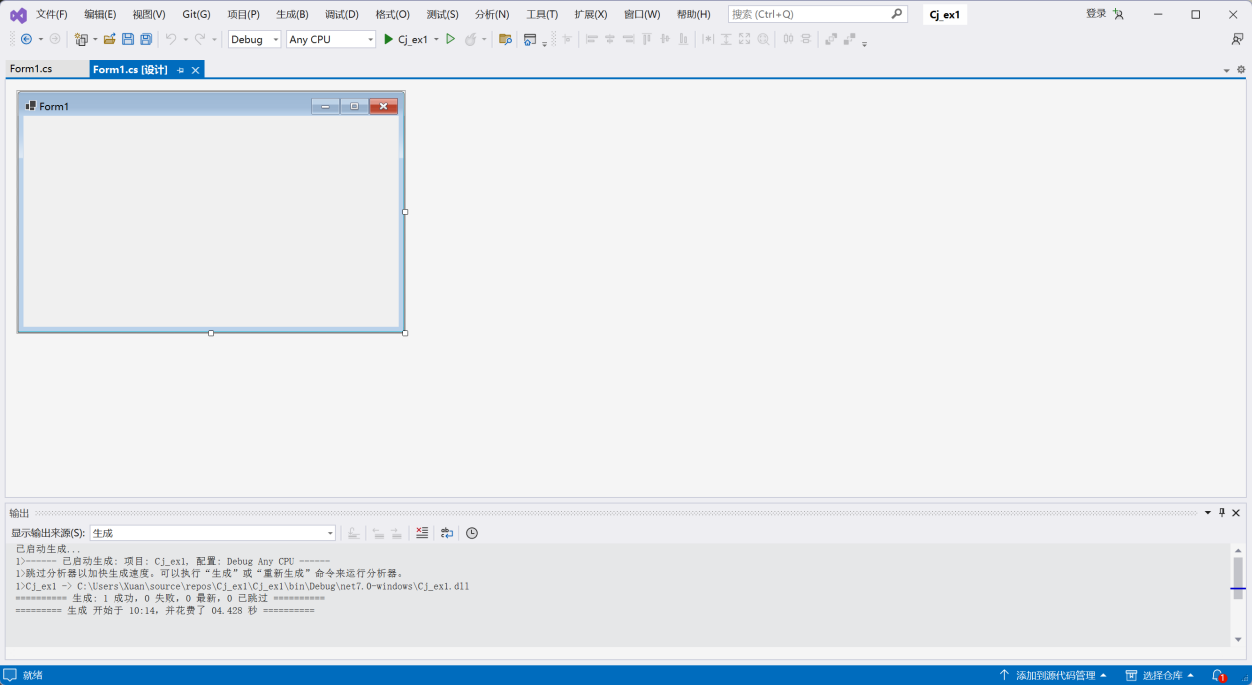
**实验报告**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **课程**  **名称** | 面向对象程序设计语言 | | | | **实验名称** | 设计计算器程序 | | |
| **系部** | 地理信息科学系 | | **班级** | 地信班 | **姓名** | 许愿 | **学号** | 109092023XXX |
| **成绩** | |  | | | | | | |

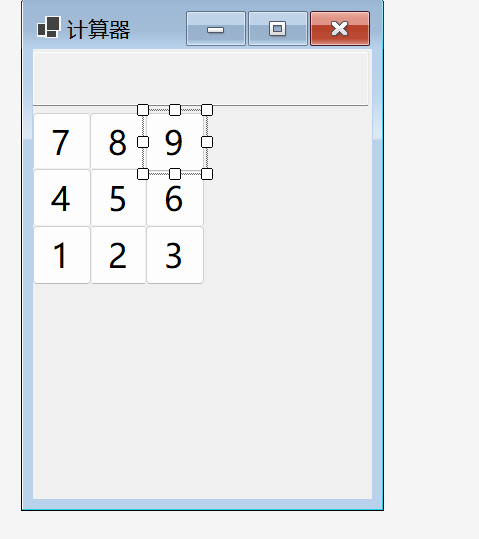
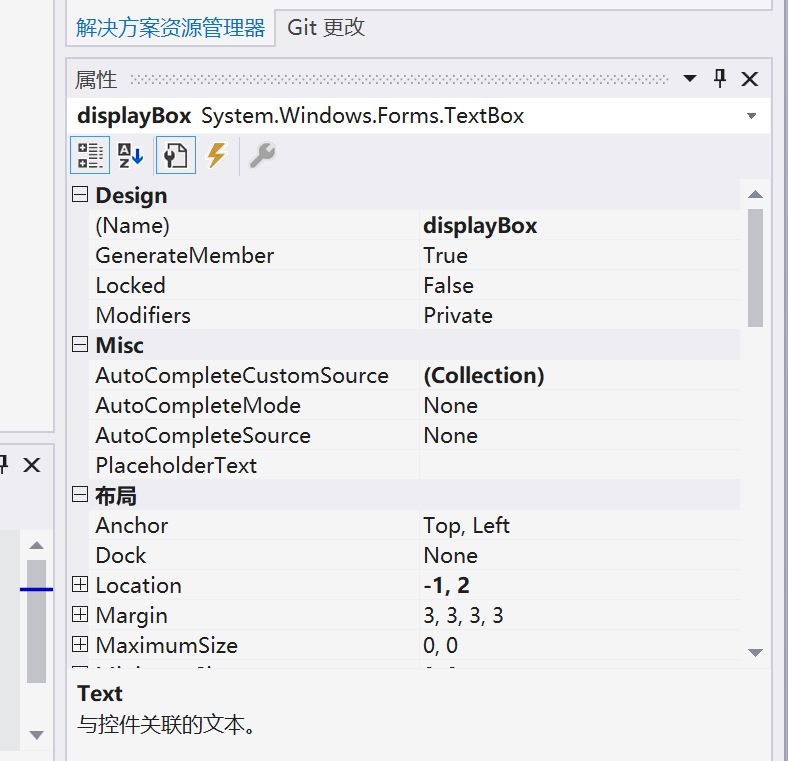
1. **实验目的**

通过设计Windows Form计算器程序，了解C#程序设计的基本过程、学习C#的基本语法、初步掌握Math类的使用方法，提升使用C#通过程序设计来解决生活中的实际问题的能力。

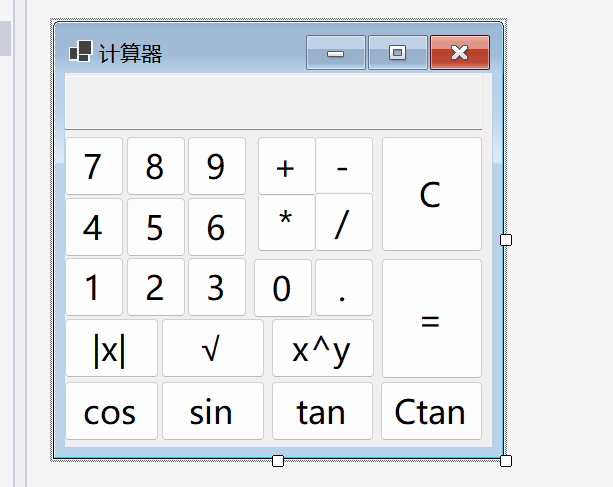
1. **实验内容**
2. 设计Windows Form计算器程序（参考系统自带计算器），实现加减乘除、求绝对值、Cos、Sin、Tan、Ctan、开方、乘方计算(参考Math类)。
3. 编写一个窗体程序，可计算并输出某一数字区间内能同时被3,5整除，且不是100的倍数的数。数字区间、每行输出个数可由用户输入指定，数与数之间间隔一个Tab宽度。
4. **实验步骤**
5. 设计Windows Form计算器程序
6. 新建项目，命名为Cj\_ex1。



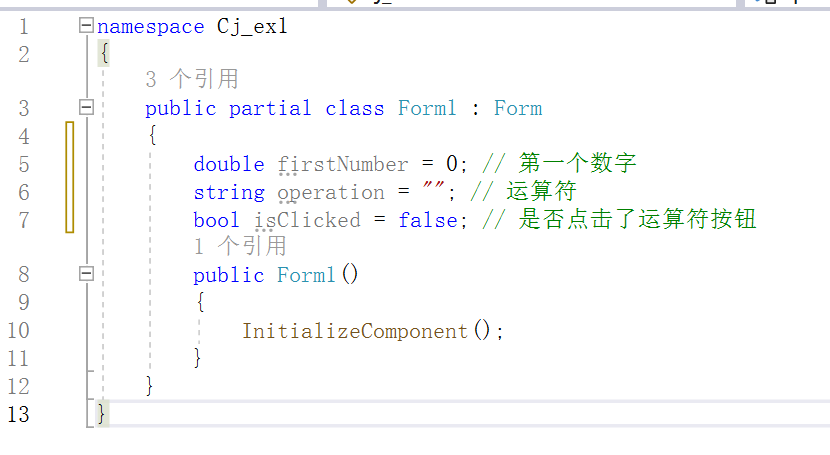
1. 设计用户界面。
2. 从工具箱中添加一个TextBox，在属性中将其更名为displayBox，设置文本向右对齐、字号增大、只读。



1. 从工具箱中添加一个Button，在属性中将其更名为btn1，设置文本为1，字号增大。复制该按钮并修改，完成各个按钮的添加。
2. 最终效果如下。



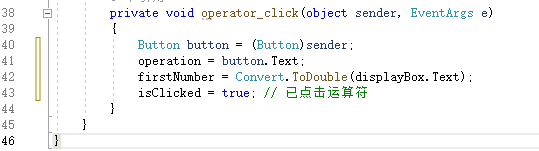
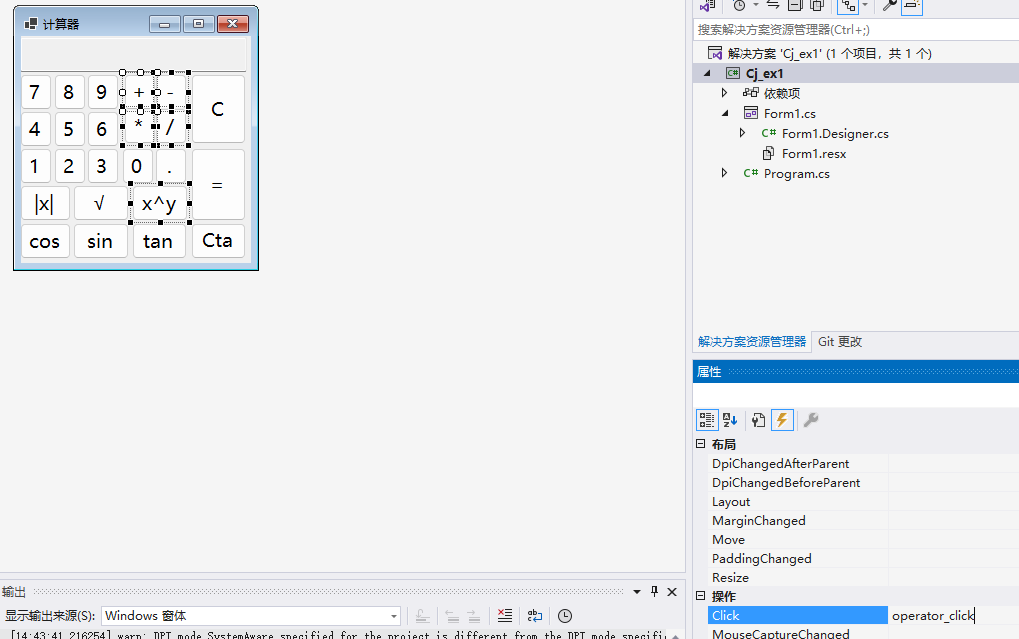
1. 定义全局变量。计算器需要一些变量来存储当前的状态。



1. 为0-9及小数点按钮定义事件，绑定number\_click函数并编写函数。



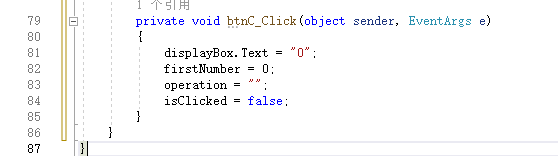
1. 为运算符按钮定义事件，绑定operator\_click函数并编写函数。



1. 为等于按钮定义事件，双击按钮绑定btndengyu\_Click函数并编写函数。



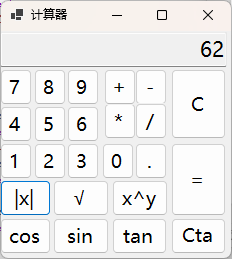
1. 为清除按钮定义事件，双击按钮绑定btnC\_Click函数并编写函数。



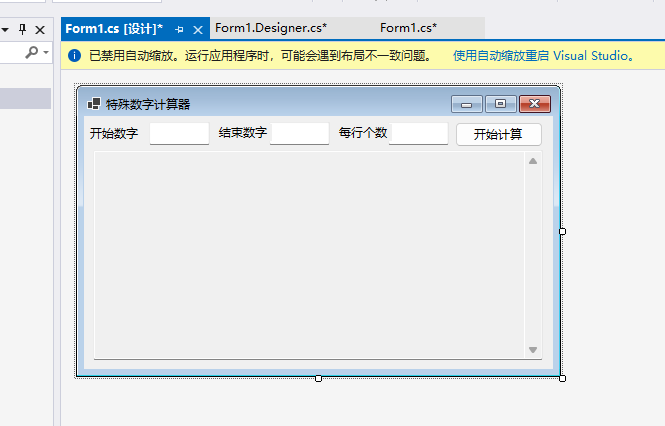
1. 为cos等特殊函数按钮定义事件，绑定spOperator\_Click函数并编写函数。



1. 运行测试。
2. 输入12 + 8，正常显示20。
3. 输入100 / 9，正常显示11.111111。
4. 输入 5 x^y 3，正常显示125。
5. 输入90 sin，正常显示1。
6. 输入- 55，输入时不显示负号，但点击=号后正常显示。点击|x|，正常变为55。



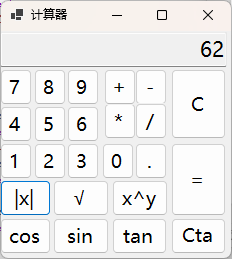
1. 编写计算并输出能同时被3,5整除且不被100整除的区间数
2. 新建项目，命名为Cj\_ex1\_2。
3. 设计用户界面。
4. 从工具箱中拖出TextBox作为输入框和结果框（作为结果框时，需要开启ReadOnly只读、ScrollBars改为Vertical以增加滚动条、开启Multiline以显示多行）。
5. 从工具箱中拖出Label作为显示输入框的提示词。
6. 从工具型中拖出TextBox作为开始计算按钮。
7. 最终效果如图所示。



1. 双击开始计算按钮进入事件函数button1\_Click，为其编写代码。对输入的数据进行处理，防止出现错误。



1. 运行测试，结果正常。
2. **程序运行结果截图**



1. **程序源代码**
2. 计算器

namespace Cj\_ex1

{

public partial class Form1 : Form

{

double firstNumber = 0; // 第一个数字

string operation = ""; // 运算符

bool isClicked = false; // 是否点击了运算符按钮

public Form1()

{

InitializeComponent();

}

private void number\_click(object sender, EventArgs e)

{

// 如果显示的是0或刚刚按下了运算符，则清空显示屏

if (displayBox.Text == "0" || isClicked){

displayBox.Clear();

}

isClicked = false;

Button button = (Button)sender;

// 如果按下的是小数点并且已经有小数点了就不再添加

if (button.Text == "."){

if (!displayBox.Text.Contains(".")){

displayBox.Text += button.Text;

}

}

else{

displayBox.Text += button.Text;

}

}

private void operator\_click(object sender, EventArgs e)

{

Button button = (Button)sender;

operation = button.Text;

firstNumber = Convert.ToDouble(displayBox.Text);

isClicked = true; // 已点击运算符

}

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

{

double secondNumber = Convert.ToDouble(displayBox.Text);

double result = 0;

switch (operation){ // 通过switch函数判断不同的运算符

case "+":

result = firstNumber + secondNumber;

break;

case "-":

result = firstNumber - secondNumber;

break;

case "\*":

result = firstNumber \* secondNumber;

break;

case "/":

if (secondNumber == 0){ // 对除以0的情况做出特殊判断

MessageBox.Show("不能除0");

return;

}

result = firstNumber / secondNumber;

break;

case "x^y":

result = Math.Pow(firstNumber, secondNumber);

break;

}

displayBox.Text = result.ToString();

operation = "";

}

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

{

displayBox.Text = "0";

firstNumber = 0;

operation = "";

isClicked = false;

}

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

{

Button button = (Button)sender;

double number = Convert.ToDouble(displayBox.Text); // 转换为双精度

double result = 0;

double ang = number \* Math.PI / 180.0; // 转换为弧度

switch (button.Text){

case "cos":

result = Math.Cos(ang);

break;

case "sin":

result = Math.Sin(ang);

break;

case "tan":

result = Math.Tan(ang);

break;

case "Ctan":

if (Math.Tan(ang) == 0){

MessageBox.Show("错误：输入值无效！");

return;

}

result = 1 / Math.Tan(ang);

break;

case "√":

result = Math.Sqrt(number);

break;

case "|x|":

result = Math.Abs(number);

break;

}

displayBox.Text = result.ToString();

isClicked = true;

}

}

}

1. 获取区间指定数

namespace Cj\_ex1\_2

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

int start, end, Geshu;

string result = "";

if (!int.TryParse(textBoxStart.Text, out start)){

MessageBox.Show("起始数字必须是一个有效的整数！");

return; // 中断执行

}

if (!int.TryParse(textBoxEnd.Text, out end)){

MessageBox.Show("结束数字必须是一个有效的整数！");

return;

}

if (!int.TryParse(textBoxGeshu.Text, out Geshu)){

MessageBox.Show("每行个数必须是一个有效的整数！");

return;

}

if (start > end){

MessageBox.Show("起始数字不能大于结束数字！");

return;

}

if (Geshu <= 0){

MessageBox.Show("每行个数必须大于0！");

return;

}

textBoxResult.Clear();

// 用于计算当前行已经输出了多少个数

int countOnCurrentLine = 0;

// 使用 for 循环遍历从起始到结束的每一个数字

for (int i = start; i <= end; i++)

{

if ((i % 3 == 0) && (i % 5 == 0) && (i % 100 != 0)){

result += i;

result += "\t";

countOnCurrentLine++;

if (countOnCurrentLine >= Geshu){ // 达到换行条件

result += "\r\n";

countOnCurrentLine = 0;

}

}

}

textBoxResult.Text = result;

// 如果一个数都没找到，就提示

if (string.IsNullOrWhiteSpace(textBoxResult.Text)){

textBoxResult.Text = "没有符合条件的数字";

}

}

}

}

1. **收获，体会及问题**

通过完成计算器与数字查找计算器这两个C#窗体项目，我初步掌握了Visual Studio的可视化界面设计、控件使用及属性配置，尝试使用了循环、条件判断等逻辑来解决生活中可能会遇到的实际问题。

实践中，我也发现了我编写的程序依然存在不足，可能还需要进一步的修改和优化，故这些问题成为了我深入学习的动力。例如，计算器在计算如sin0.5的数字时会生成0.49999999994，此为double类型会产生的浮点数精度误差；计算器无法处理三个及以上数据连续的计算；数字查找器在处理大数据（比如起始3结束99999999）时则会阻塞卡死一段时间。

这次经历让我认识到，要开发出高质量的应用，除了实现基本功能，还必须深入理解和选择功能模块，不断优化程序，采用更好的办法解决问题。这为我后续的学习指明了清晰的方向。