**《Python程序设计》**

**课程设计报告**

　　　　　　　　　　（2018 —2019 学年第 一 学期）

题 目 简易计算器

学生姓名 袁 畅

专业班级 信管1601

学生学号 311609030125

教师姓名 徐 文 鹏

成 绩**：**

评 语：

教师签名：

日期：

目录

[一、 设计目的](#_Toc28401_WPSOffice_Level1) [2](#_Toc28401_WPSOffice_Level1)

1.课程设计教学目的 2

2.本课程设计目的 2

[二、 设计任务和要求](#_Toc19044_WPSOffice_Level1) [3](#_Toc19044_WPSOffice_Level1)

1.课程设计教学任务和要求 3

2.本课程设计具体任务和要求 3

[三、 总体设计 3](#_Toc10030_WPSOffice_Level1)

1.软件功能图 3

2.软件流程图 4

3.任务分配 5

[四、 设计实现](#_Toc1504_WPSOffice_Level1) [5](#_Toc1504_WPSOffice_Level1)

1.最终实现结果 5

2.实现结果评价 7

[五、 详细设计](#_Toc30472_WPSOffice_Level1) [8](#_Toc30472_WPSOffice_Level1)

1.输入输出分析 8

2.程序流程图 8

3.相关算法的详细说明 9

[六、 调试与测试](#_Toc30341_WPSOffice_Level1) [12](#_Toc30341_WPSOffice_Level1)

1.调试方法 12

2.测试结果分析和讨论 12

3.主要问题及解决措施 12

[七、 设计总结](#_Toc20259_WPSOffice_Level1) [13](#_Toc20259_WPSOffice_Level1)

[附录](#_Toc4150_WPSOffice_Level1) [14](#_Toc4150_WPSOffice_Level1)

1. **设计目的**  
    1.课程设计教学目的

本课程设计是本专业的一门重要实践性教学环节。在学习了专业基础课和《Python程序设计》课程的基础上，本课程设计旨在加深对Python程序设计的认识，对Python语言及其语言生态有一个进一步的掌握和应用，学会运用Python标准库及外接相关库来解决实际问题的基本能力，培养和提高学生分析问题、解决问题的能力，尤其是提高学生使用Python为开发语言来进行问题描述、交流与思考的能力，为毕业设计和以后的工程实践打下良好的基础。

2.本课程设计目的

Python是一种面向对象、直译式计算机程序设计语言，也是一种功能强大而完善的通用型语言，已经具有十多年的发展历史，成熟且稳定。这种语言具有非常简捷而清晰的语法特点，适合完成各种高层任务，几乎可以在所有的操作系统中运行。目前，基于这种语言的相关技术正在飞速的发展，用户数量急剧扩大，相关的资源非常多。

在本学期的学习过程中，先后学习了python的基本语句、函数、模块、类以及异常，对python语言有了一个初步的了解，并对面向对象编程有了一个整体的认识。在这里，利用课堂上所学习的python知识，编写程序客户端与服务器的运作流程。

本次课程设计简易计算器的制作是由我们小组三个人经过商议之后，综合考虑自身的能力后共同决定的，因为计算器在我们的生活中非常常见并且很有用，当代的人类生活中，我们经常会遇到数字数据比较大且数据复杂的数字运算，心算笔算起来很复杂困难，这时候就要想到用到计算器。通过本程序设计将要实现一个简单的计算器。计算生活中的数字已经成了必不可少的一段生活过程了,而随着生活水平的提高，更加繁琐的计算的已经成了必不可少的一部分了,现在生活的一部分仅仅靠自己的心算来计算已经是满足不了的。所以我们设计了这个简易计算器程序并通过python的打包功能使普通的电脑也可以使用这个软件。并且通过这个简易程序的设计使我们对python的功能及用法有更多的了解和掌握，对于我们以后的编程发展也有很大的好处。虽然我们的课程设计作品相对简单，但是同样锻炼了我们的编程能力和团队合作能力。为我们今后的学习和工作积累了经验。

1. **设计任务和要求**

1.课程设计教学任务和要求

本课程设计主要任务是以Python为开发语言完成一个100~300行左右规模的程序项目开发。

课程设计的基本要求是：在课程设计的各个阶段严格、规范地完成相关的文档，例如在初期按时完成设计目的、设计要求、总体设计，在后期完成详细设计、调试与测试、设计总结等。要求能完成所选题目的主要功能，程序运行健壮、正确，代码要有详细注释，可读性好；所写文档结构合理、内容完整、叙述清晰。更高要求是：有创意、系统界面美观。

由于课程设计项目具有一定的综合性，鼓励具有不同特长和不同能力的学生互相组队。项目小组自己推荐一名组长，实行“组长负责制”。组长组织组员进行项目选题、任务分配、方案确定、方案设计、系统调试测试，组员分工协作。小组成员开展项目讨论，互相支持，形成协作意识。

2.本课程设计具体任务和要求

经过小组成员的讨论，为满足Python课程教学任务与要求，锻炼小组成员利用Python编写程序与自主学习的能力，我们小组各成员决定选择利用Python语言实现一个简易的计算器。

所要实现的简易计算器的功能如下：

（1）程序能实现简单的加减乘除取余等计算功能；

（2）程序由鼠标来输入，具有良好的用户界面；

（3）利用pyinstaller工具将程序打包成exe文件。

1. **总体设计**

1.软件功能图

简单计算器

加减乘除取余运算

退格

清零

2.软件流程图

开始

按下数字按钮

显示数字在文本框内

加减乘除运算

在文本框中显示结果

结束

3.任务分配

设计内容：能实现简单的加减乘除取模等简单计算，可以使用鼠标来输入，用户界面美观简洁。能够使用pyinstaller工具将程序打包为exe文件。

任务分配情况：

余自立负责实现程序基本运算功能；

袁畅负责实现鼠标点击输入功能和程序打包；

杨中义负责用户图形界面设计和代码修改等工作。

1. **设计实现**

1.最终实现结果

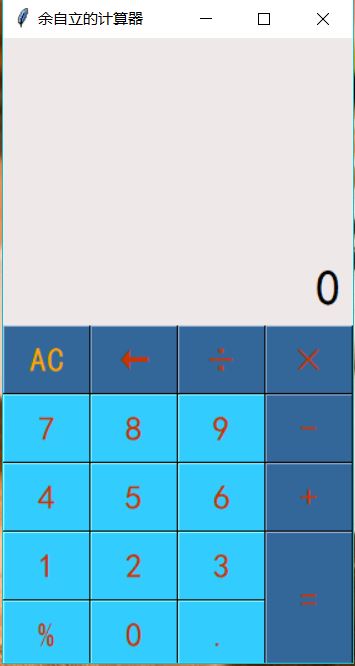


图1程序运行初始界面 图2计算器执行加操作



图3 计算器执行减操作 图4 计算器执行乘操作

图5 计算器执行除操作 图6计算器执行综合运算

2.实现结果评价

首先，在拿到了老师给的课程设计指导书后，我们小组仔细、认真的研读，综合考虑自身的编程能力，实事求是选定课程设计备选题目中的第一题就开始着手准备。然而，题目虽然简单但是想要做好课程设计也不是那么简单，我们使用IDLE编写Python代码。在百度、google和查阅相关书籍一番后，并对简单的程序编写和调试，其次，在课程设计中，有很多的、复杂的数据计算，这就必须要求我心细谨慎。更重要的是，在对软件编写程序的时候，每一个字母，每一个标点符号都要求做到百分之百。因此，我也由一开始的热情高涨变得浮躁，开始抓狂。在同学们的鼓励支持下，我又重获自信，细化目标，一步一步的沿着小小的目标进发，逐步地检查计算结果和编写，修改，调试程序。最终，完善课程设计中的任务。

再次，我们在课程设计中，锻炼了我们的编程能力和团队合作能力。我认为此次课程设计不仅磨练了我们的耐心，还增加了我们对自动控制学习的兴趣，虽然有汗水，有心酸，但我知道这一切都值得的，而且我还会继续坚持的走下去。同时也要感谢老师对我们的帮助和同学们对我们的鼓励。

设计完成结果小组自我评价：良好

1. **详细设计**

在本小组的本次简易计算器python课程设计中，本人负责的模块是实现计算器的鼠标键盘的输入和在程序完成之后对程序的打包操作，进而实现写出的程序能够在没有安装python环境的计算机上能够运行。在实现对计算器的鼠标键盘的输入的时候本人主要借助了python第三方库的tkinter库的button控件，Button小部件是一个标准的Tkinter的部件，用于实现各种按钮。按钮可以包含文本或图像，可以调用Python函数或方法用于每个按钮。Tkinter的按钮被按下时，会自动调用该函数或方法。在button控件中我通过text 函数来设置按钮的内容，通command函数来定义响应按钮被点击时被调用的函数。并通过设置fg,bg,font,bd的值来设置按钮的前景色、背景色、字体和边框。在设置按钮时还用到了place函数来设置按钮的位置和宽高。最后，在程序经过测试调试之后，我还负责通过python的第三方库pyinstaller来将程序打包成exe文件，这里通过命令pip install pyinstaller来完成对Python第三方扩展库pyinstaller的安装，之后通过命令pyinstaller -F 文件绝对路径+文件名 来实现对计算器的打包。

1.输入输出分析

我负责的模块中，在实现计算器的键盘输入功能时输入便是使用鼠标点击计算器的数字或者运算符号按钮，而输出便是在计算器显示页面显示相应点击按钮的内容。该部分通过tkinter库的button控件内的command函数调用响应函数，在点击数字按键时调用了我和另一个小组成员共同编写的pressNum()函数，在点击运算符号时调用的响应函数是pressCompute()函数，在点击等号得出结果时，响应函数是pressEqual()函数，而之后的两个函数都由我另一个小组成员完成。

2.程序流程图

开始

是否为数字

是

调用pressNum()函数

显示数字

否

是否为等号

调用pressCompute()函数

显示运算符

否

是

调用pressEqual()函数

显示运算结果

3.相关算法的详细说明

在实现计算器鼠标键盘输入部分的代码时，我主要利用了tkinter库的button控件。在button控件中我通过text 函数来设置按钮的内容，通command函数来定义响应按钮被点击时被调用的函数。并通过设置fg,bg,font,bd的值来设置按钮的前景色、背景色、字体和边框。在设置按钮时还用到了place函数来设置按钮的位置和宽高。在利用command函数调用触发事件的回调函数时，当按下数字按钮时调用了pressNum()函数并将按下的数字显示在计算器文本框内，当按下运算符号按钮时调用了pressCompute()函数并将按下的运算符号显示在文本框内，当按下等号时调用了pressEqual()函数来实现运算工作并将运算结果显示在计算器的界面上。

实现的相关代码如下：

#button的用法：Button(根对象, [属性列表])；

#root为根对象；

#text为button组件内的函数来设置按钮显示的内容，这里text = '0'表示按钮内显示数字0；

#font为设置字体的属性，这里font = ('黑体',20)表示设置为20号黑体；

#fg为设置前景色的属性，这里fg = ('#cc3300')表示设置按钮前景色为#cc3300；

#bg为设置背景色的属性，这里bg = ('#33ccff')表示设置按钮背景色为#33ccff；

#bd为设置边框的属性，这里bd = 0.5表示设置边框的大小为0.5个单位；

#command为响应按钮点击事件的回调函数，command = lambda : pressNum('0')表示点击数字0时，调用pressNum()函数来响应事件；command = lambda:pressCompute('/')表示点击运算符‘/’时，调用pressCompute()函数来响应按钮点击事件；command = lambda :pressEqual()表示点击‘=’时，调用pressEqual()函数来响应事件进行计算。

#place(x,y,width,height)来设置按钮的位置与宽高。

#数字键按钮

btn0 = tkinter.Button(root,text = '0',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('0'))

btn0.place(x = 70,y = 450,width = 70,height = 55)

btn1 = tkinter.Button(root,text = '1',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('1'))

btn1.place(x = 0,y = 395,width = 70,height = 55)

btn2 = tkinter.Button(root,text = '2',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('2'))

btn2.place(x = 70,y = 395,width = 70,height = 55)

btn3 = tkinter.Button(root,text = '3',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('3'))

btn3.place(x = 140,y = 395,width = 70,height = 55)

btn4 = tkinter.Button(root,text = '4',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('4'))

btn4.place(x = 0,y = 340,width = 70,height = 55)

btn5 = tkinter.Button(root,text = '5',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('5'))

btn5.place(x = 70,y = 340,width = 70,height = 55)

btn6 = tkinter.Button(root,text = '6',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('6'))

btn6.place(x = 140,y = 340,width = 70,height = 55)

btn7 = tkinter.Button(root,text = '7',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('7'))

btn7.place(x = 0,y = 285,width = 70,height = 55)

btn8 = tkinter.Button(root,text = '8',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('8'))

btn8.place(x = 70,y = 285,width = 70,height = 55)

btn9 = tkinter.Button(root,text = '9',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('9'))

btn9.place(x = 140,y = 285,width = 70,height = 55)

#运算符号按钮

btnac = tkinter.Button(root,text = 'AC',bd = 0.5,font = ('黑体',20),fg = 'orange',bg = ('#336699'),command = lambda :pressCompute('AC'))

btnac.place(x = 0,y = 230,width = 70,height = 55)

btnback = tkinter.Button(root,text = '←',font = ('黑体',20),fg = '#cc3300',bg = ('#336699'),bd = 0.5,command = lambda:pressCompute('b'))

btnback.place(x = 70,y = 230,width = 70,height = 55)

btndivi = tkinter.Button(root,text = '÷',font = ('黑体',20),fg = '#cc3300',bg = ('#336699'),bd = 0.5,command = lambda:pressCompute('/'))

btndivi.place(x = 140,y = 230,width = 70,height = 55)

btnmul = tkinter.Button(root,text ='×',font = ('黑体',20),fg = "#cc3300",bd = 0.5,bg = ('#336699'),command = lambda:pressCompute('\*'))

btnmul.place(x = 210,y = 230,width = 70,height = 55)

btnsub = tkinter.Button(root,text = '-',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#336699'),command = lambda:pressCompute('-'))

btnsub.place(x = 210,y = 285,width = 70,height = 55)

btnadd = tkinter.Button(root,text = '+',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#336699'),command = lambda:pressCompute('+'))

btnadd.place(x = 210,y = 340,width = 70,height = 55)

btnequ = tkinter.Button(root,text = '=',bg = ('#336699'),font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,command = lambda :pressEqual())

btnequ.place(x = 210,y = 395,width = 70,height = 110)

btnper = tkinter.Button(root,text = '%',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda:pressCompute('%'))

btnper.place(x = 0,y = 450,width = 70,height = 55)

btnpoint = tkinter.Button(root,text = '.',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda:pressCompute('.'))

btnpoint.place(x = 140,y = 450,width = 70,height = 55)

1. **调试与测试**
2. 调试方法

在实现计算器键盘鼠标的输入时，本人应用的调试方法便是在IDLE中一次次运行，通过翻阅资料文献在网上查资料，不断修改代码来进行不断地调试，进而最后得到了需要得到的结果。

1. 测试结果分析和讨论

在实现键盘鼠标的输入而进行不断测试的时候，各个按钮的布局首先要设置好，通过place()函数来实现，然而在编写过程中按钮常常出现在自己意想不到的位置，这里就通过查阅资料了解place()函数的内部参数组成再继续更改参数来继续设置，而在实现按钮功能时通过调用函数来实现，这里在开始时按下按钮并没有数字或符号显示在界面上，这就要考虑是button控件内部函数设置错误还是调用我另一个小组成员函数时出现了错误，最后通过不断试错完成了功能。

1. 主要问题及解决措施

在调试过程遇到的主要问题就是按下按钮，屏幕上没有显示该显示的数字或运算符，解决措施就是先检查是自身在调用command函数的应用时出现错误，还是我另一个小组成员编写的功能算法有误，然后通过我们配合分工确定出错位置，再修改相应代码，最后得到想要的结果。

1. **设计总结**

学完python语言程序设计这门课，又在近几天利用之前在网上和课堂上所学的python知识和我的小组成员共同完成实现一个简易计算器，我能够把所学的python语言知识初步运用到实践中，而我也对python语言也有了更深层次的理解。这两天系统设计过程所遇到的难题和遗漏的知识内容，让我更加清晰的认识到自己平时学习生活中存在的不少漏洞，而我也对自己本来有些生疏的知识内容有了更加深刻的印象。这两天遇到了很多难题，通过在网上查阅资料不断尝试和向同学请教，最后这些难题都被完美的解决了，这些给了我很大的满足感。这次python的课程设计还给我带来了前所未有的充实感觉，因为我们进行了团队合作，并且明确分工做出了东西，完成了任务，感觉这门课收获很大。而且，这次的课程设计对我不仅在学习上，在生活中也都让我学了很多，给我产生了很深的影响。

在这个互联网所统治的时代下，网络给我们生活学习和工作都提供了很大的便利，不仅我们在生活和娱乐上越来越离不开互联网，我们的工作也越来越深入的扎根在了互联网中，而越来越多的依赖于互联网，网络中有游戏软件供我们娱乐，有学习软件供我们学习，有购物软件供我们网上购物，也有很多有关工作的软件管理着我们员工工作，互联网和软件已经渐渐深入我们的生活当中，我们已经越来越离不开它们。而通过python语言的初步学习我们已经能过实现一些简单功能的计算器软件和设计出一些简单的小游戏，而近段时间在各种编程语言中python已经排到了第三位，这些都足以说明Python语言的强大和受欢迎，我们学习python对以后的学习和工作都有非常好的影响。

在实现计算器程序的过程中，我们也遇到了一些问题。例如，因为要使计算器图形化显示出来我们之前也没有学过图形化相关的知识，设计之初我们都是很迷茫，但这些不足以难倒我们，我们通过在网上搜索和查阅资料最终确定使用Python第三方库tkinter库来实现计算器的图形化界面，然后之后的问题就变得简单了，我先自学了tkinter库包含的控件和相关函数的使用，然后便将学习的内容付诸到实践中去，我不断的尝试编写代码，调试，修改代码，再调试...最后通过不断学习和试验，我们小组完成了基本的简易计算器的功能。但平心而论，我感觉我们的计算器有些平庸，只是进行了加、减、乘、除、取余等运算，对于复杂的开平方，三角函数，对数等运算并没有实现，这算是本次课程设计的一大遗憾之处，在以后的日子里，我会不断练习使用python语言，锻炼自己的编程能力，希望自己的python学习之路越走越远。

总的来说，通过本次实现简易计算器的python课程设计，本人在各方面都深入了解了python语言的知识理论，并将其付诸到实践中去，对python语言的学习得到了一次全面的巩固。这次课程设计给了我以后熟练使用python创建了坚实的基础。

**附录**

程序代码：

import tkinter #导入tkinter模块,Tkinter 是 Python 的标准 GUI 库。Python 使用 Tkinter 可以快速的创建 GUI 应用程序,图形化用户界面。

root=tkinter.Tk('red')#生成root主窗口

root.minsize(280,500)#设置窗口大小

root.title('余自立的计算器')#计算机的名字

#1.界面布局

#显示面板

result = tkinter.StringVar()#StringVar是Tk库内部定义的字符串变量类型，在这里用于管理部件上面的字符；不过一般用在按钮button上。改变StringVar，按钮上的文字也随之改变。

result.set(0) #显示面板显示结果1，用于显示默认数字0

result2 = tkinter.StringVar() #显示面板显示结果2，用于显示计算过程

result2.set('')

#显示板

label = tkinter.Label(root,font = ('黑体',20),bg = '#EEE9E9',bd ='9',fg = '#828282',anchor = 'se',textvariable = result2)# Anchor 标签中文本的位置 顶对齐'n',底对齐's',左'w',右'e'；background(bg)背景色； foreground(fg)前景色；borderwidth(bd)边框宽度；textvariable 可变文本，与StringVar等配合着用

label.place(width = 280,height = 170) #width: 组件的宽度；

label2 = tkinter.Label(root,font = ('黑体',30),bg = '#EEE9E9',bd ='9',fg = 'black',anchor = 'se',textvariable = result)

label2.place(y = 170,width = 280,height = 60)#y: 组件右上角的y坐标；

#数字键按钮

btn0 = tkinter.Button(root,text = '0',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('0')) #command指定按钮消息的回调函数；

btn0.place(x = 70,y = 450,width = 70,height = 55)

btn1 = tkinter.Button(root,text = '1',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('1'))

btn1.place(x = 0,y = 395,width = 70,height = 55)

btn2 = tkinter.Button(root,text = '2',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('2'))

btn2.place(x = 70,y = 395,width = 70,height = 55)

btn3 = tkinter.Button(root,text = '3',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('3'))

btn3.place(x = 140,y = 395,width = 70,height = 55)

btn4 = tkinter.Button(root,text = '4',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('4'))

btn4.place(x = 0,y = 340,width = 70,height = 55)

btn5 = tkinter.Button(root,text = '5',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('5'))

btn5.place(x = 70,y = 340,width = 70,height = 55)

btn6 = tkinter.Button(root,text = '6',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('6'))

btn6.place(x = 140,y = 340,width = 70,height = 55)

btn7 = tkinter.Button(root,text = '7',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('7'))

btn7.place(x = 0,y = 285,width = 70,height = 55)

btn8 = tkinter.Button(root,text = '8',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('8'))

btn8.place(x = 70,y = 285,width = 70,height = 55)

btn9 = tkinter.Button(root,text = '9',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda : pressNum('9'))

btn9.place(x = 140,y = 285,width = 70,height = 55)

#运算符号按钮

btnac = tkinter.Button(root,text = 'AC',bd = 0.5,font = ('黑体',20),fg = 'orange',bg = ('#336699'),command = lambda :pressCompute('AC'))

btnac.place(x = 0,y = 230,width = 70,height = 55)

btnback = tkinter.Button(root,text = '←',font = ('黑体',20),fg = '#cc3300',bg = ('#336699'),bd = 0.5,command = lambda:pressCompute('b'))

btnback.place(x = 70,y = 230,width = 70,height = 55)

btndivi = tkinter.Button(root,text = '÷',font = ('黑体',20),fg = '#cc3300',bg = ('#336699'),bd = 0.5,command = lambda:pressCompute('/'))

btndivi.place(x = 140,y = 230,width = 70,height = 55)

btnmul = tkinter.Button(root,text ='×',font = ('黑体',20),fg = "#cc3300",bd = 0.5,bg = ('#336699'),command = lambda:pressCompute('\*'))

btnmul.place(x = 210,y = 230,width = 70,height = 55)

btnsub = tkinter.Button(root,text = '-',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#336699'),command = lambda:pressCompute('-'))

btnsub.place(x = 210,y = 285,width = 70,height = 55)

btnadd = tkinter.Button(root,text = '+',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#336699'),command = lambda:pressCompute('+'))

btnadd.place(x = 210,y = 340,width = 70,height = 55)

btnequ = tkinter.Button(root,text = '=',bg = ('#336699'),font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,command = lambda :pressEqual())

btnequ.place(x = 210,y = 395,width = 70,height = 110)

btnper = tkinter.Button(root,text = '%',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda:pressCompute('%'))

btnper.place(x = 0,y = 450,width = 70,height = 55)

btnpoint = tkinter.Button(root,text = '.',font = ('黑体',20),fg = ('#cc3300'),bd = 0.5,bg = ('#33ccff'),command = lambda:pressCompute('.'))

btnpoint.place(x = 140,y = 450,width = 70,height = 55)

#操作函数

lists = [] #设置一个变量 保存运算数字和符号的列表

isPressSign = False #添加一个判断是否按下运算符号的标志,假设默认没有按下按钮

isPressNum = False

#数字函数

def pressNum(num): #设置一个数字函数 判断是否按下数字 并获取数字将数字写在显示版上

global lists #全局化lists和按钮状态isPressSign

global isPressSign

if isPressSign == False:

pass

else: #重新将运算符号状态设置为否

result.set(0) #输入栏初始显示为0

isPressSign = False

#判断界面的数字是否为0

oldnum = result.get() #第一步

if oldnum =='0': #如果界面上数字为0 则获取按下的数字

result.set(num)

else: #如果界面上的而数字不是0 则链接上新按下的数字

newnum = oldnum + num

result.set(newnum) #将按下的数字写到面板中

#运算函数

def pressCompute(sign):

global lists

global isPressSign

num = result.get() #获取界面数字

lists.append(num) #保存界面获取的数字到列表中

result.set(sign) #将输入的运算符写在面板中

lists.append(sign) #将按下的运算符号保存到列表中

isPressSign = True

if sign =='AC':#如果按下的是'AC'按键，则清空列表内容，将屏幕上的数字键设置为默认数字0

lists.clear()

result.set(0)

if sign =='b': #如果按下的是退格‘’，则选取当前数字第一位到倒数第二位

a = num[0:-1]

lists.clear()

result.set(a)

#获取运算结果函数

def pressEqual():

global lists

global isPressSign

curnum = result.get() #设置当前数字变量，并获取添加到列表

lists.append(curnum)

computrStr = ''.join(lists) #讲列表内容用join命令将字符串链接起来

endNum = eval(computrStr) #用eval命令运算字符串中的内容

result.set(endNum) #讲运算结果显示到屏幕1

result2.set(computrStr) #将运算过程显示到屏幕2

lists.clear() #清空列表内容

root.mainloop()#进入消息循环（必需组件）