**《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 余 自 立

专业班级 信管1601

学生学号 311609030122

教师姓名 徐 文 鹏

成 绩**：**

评 语：

教师签名：

日期：

目录

**[1、设计目的 1](#_Toc25471_WPSOffice_Level1)**

[1.1课程设计教学目的 1](#_Toc5284_WPSOffice_Level2)

[1.2本课程设计目的 1](#_Toc28404_WPSOffice_Level2)

**[2、设计任务和要求 2](#_Toc5284_WPSOffice_Level1)**

[2.1课程设计教学任务和要求 2](#_Toc3881_WPSOffice_Level2)

[2.2本课程设计具体任务和要求 2](#_Toc31951_WPSOffice_Level2)

**[3、总体设计 2](#_Toc28404_WPSOffice_Level1)**

[3.1总体设计思路 2](#_Toc30202_WPSOffice_Level2)

[3.2软件功能图 3](#_Toc32084_WPSOffice_Level2)

[3.3软件流程图 4](#_Toc8999_WPSOffice_Level2)

3.4任务分配 4

**[4.设计实现 4](#_Toc3881_WPSOffice_Level1)**

[4.1最终实现结果 5](#_Toc186_WPSOffice_Level2)

[4.2实现结果评价 6](#_Toc13002_WPSOffice_Level2)

**[5.详细设计 7](#_Toc31951_WPSOffice_Level1)**

[5.1 内容描述 7](#_Toc21616_WPSOffice_Level2)

[5.2 数字输入函数 7](#_Toc16233_WPSOffice_Level2)

[5.3 运算符号输入函数 8](#_Toc26733_WPSOffice_Level2)

[5.4 获取运算结果函数 9](#_Toc26733_WPSOffice_Level2)

**[6.调试与测试 1](#_Toc30202_WPSOffice_Level1)0**

[1.数字输入函数 1](#_Toc25218_WPSOffice_Level2)1

[2.运算符号输入函数 1](#_Toc15380_WPSOffice_Level2)1

[3.获取运算结果函数 1](#_Toc8798_WPSOffice_Level2)1

**[7.设计总结 1](#_Toc32084_WPSOffice_Level1)2**

[7.1课程设计缺点 1](#_Toc30510_WPSOffice_Level2)2

[7.2课程设计总结 1](#_Toc32674_WPSOffice_Level2)2

**[附录 1](#_Toc8999_WPSOffice_Level1)3**

**1、设计目的**

1.1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。

1.2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。

**2、设计任务和要求**

2.1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。

2.2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。

**3、总体设计**

3.1总体设计思路

课程设计主要使用Python语言中tkinter模块,Tkinter 是 Python 的标准 GUI 库。使用 Tkinter 可以快速的创建 GUI 应用程序,图形化用户界面。我们先在稿纸设计了简易计算器的基本界面，包括数字按钮，运算符号以及退格和归零按钮等。参照设计图，我们使用tkinter中的label组件和button组件来快速构建用户图形化界面。主界面绘制完成后在主显示版上添加两个子显示屏一个用来显示计算过程另一个用来显示计算结果。使用result = tkinter.StringVar()语句与label组件中的textvariable语句实现数据的绑定。每个按钮添加一个回调函数，用来输入运算表达式，最后使用eval（）语句来计算最终运算结果。

3.2软件功能图

简单计算器

加减乘除取余运算

退格

清零

3.3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

3.4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，在任何电脑上都可以运行，提高了代码的可移植性。

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。

**4.设计实现**

4.1最终实现结果

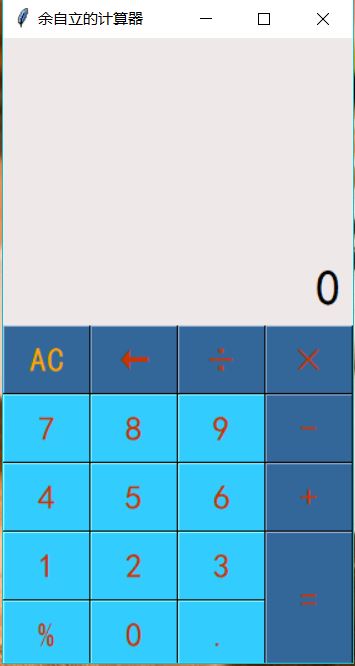
 图1程序运行初始界面 图2计算器执行加操作

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计算器执行综合操作

4.2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。

作品设计完成后能够实现基本的加减乘除取余等运算，用户图形化界面良好因此我对我们小组设计完成结果评价：良好！

**5.详细设计**

5.1 内容描述

我负责实现程序基本运算功能，功能的实现主要靠三个函数分别是数字输入函数、运算符号输入函数、获取运算结果函数。首先定义一个空列表用来保存运算数字和符号；然后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，令isPressSign = False和isPressNum = False。数字输入函数用来获取数字按钮输入的数字，并将输入的数字保存在全局列表list中，为下一步的计算做准备。运算符号输入函数用来获取符号按钮输入的运算符号，并将输入的运算符号保存在全局列表list中，为下一步的计算或输入做准备。获取运算结果函数主要功能是当用户输入一行正确的运算表达式后，按下等号按钮可以正确的计算出最终结果。

5.2 数字输入函数

输入输出分析：

输入一行数字能将数字转化为正确的数学值，比如输入000001，可以转化为1，并将数值保存在列表list中。

算法详细说明：

通过Python保留字def定义一个数字输入函数pressNum(num)，全局化lists和按钮状态isPressSign，以便其他两个函数可以修改或调用其中的数据。

第一步、重新将运算符号状态设置为否isPressSign == False。利用result.set(0) 语句设置输入栏初始显示为0；

第二步、判断界面的数字是否为0，如果界面上数字为0 ，则获取按下的数字，如果界面上的而数字不是0，则链接上新按下的数字，即newnum = oldnum + num。

第三步、将按下的数字写到面板中。

开始

判断是否按下数字按钮

是

输入一串数字

将数字转化为正确的数学值输出在面板上

结束

5.3 运算符号输入函数

输入输出分析：

输入一个运算符，将输入的运算符保存到列表list中，在面板上输出所写入的运算符。

算法详细说明：

通过Python保留字def定义一个运算符号输入函数 pressCompute(sign)，全局化lists和按钮状态isPressSign，以便其他两个函数可以修改或调用其中的数据。

第一步、利用num = result.get()语句获取界面数字

第二步、保存界面获取的数字到列表中lists.append(num)，将按下的运算符号保存到列表中lists.append(sign)，并将输入的运算符写在面板中

第三部、进行判断，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；如果按下的是退格‘←’，则选取当前数字第一位到倒数第二位。

开始

判断是否按下运算符按钮

是

输入一个运算符

输出运算符或执行功能

结束

5.4 获取运算结果函数

输入输出分析：

按下等号运算符，正确计算出表达式结果，并在主面板中输出运算结果

算法详细说明：

通过Python保留字def定义一个获取运算结果函数 pressEqual()，全局化lists和按钮状态isPressSign，以便其他两个函数可以修改或调用其中的数据。

第一步、利用curnum = result.get()语句设置当前数字变量，并获取添加到列表

第二步、将列表内容用join命令将字符串链接起来 computrStr = ''.join(lists) ，用eval命令运算字符串中的内容 endNum = eval(computrStr。

第三步、 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

开始

判断是否按下 =

是

输出计算结果

结束

6.调试与测试

工具：IDLE

主要任务：检验实现运算功能函数能否正确执行，并加以优化。

1.数字输入函数

问题：当输入0001时，程序会直接将0001打印在面板上。

解决方案：

在pressNum(num)函数中加入以下代码

oldnum = result.get() #先获取面板中的数字

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

2.运算符号输入函数

问题：面板中不显示运算符，AC键和退格键功能不全

解决方案：

在 pressCompute(sign)函数中加入以下代码

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)

3.获取运算结果函数

问题：当按下“=”号运算符时程序无响应，不能正确计算表达式的结果。

解决方案：

在 pressEqual()函数中加入以下代码

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

在修改完代码之后，我们发现计算器只能进行一次运算，反复查看代码始终没有找出问题所在。上网查询相关资料后，在代码的最后加上root.mainloop()语句，它的作用就是进入消息循环，使我们这个小程序可以一直运行

**7.设计总结**

7.1课程设计缺点

计算器功能过于简单，类比科学计算器，还可以添加许多其他的计算功能，例如求三角函数、开n次方、求阶乘等。除此之外，用户界面相对简单。在课程设计展示过程中，看到其他组做的计算器，感觉我们的用户界面过于原始。总体浏览一遍代码之后，发现代码有许多重复的部分。例如再设置按钮样式的时候可以编写一个函数来实现，这样就不用一个一个设置按钮了，可以统一设置。另外一点是运算部分代码过于冗余，鉴于个人水平有限，以后学习的道路还很长。

7.2课程设计总结

首先很庆幸自己能够学到这门个性化课的课程，可能是我个人比较认为这门课程所用的语言很特别很奇妙，老师也很有趣，能让我们更好的了解Python这门课程真正的用途。Python是一种计算机程序设计语言。是一种动态的、面向对象的脚本语言，最初被设计用于编写自动化脚本(shell)，随着版本的不断更新和语言新功能的添加，越来越多被用于独立的、大型项目的开发。这是百度百科对Python的描述。在学习Python这门课程的这段时间以来，并且自己也能认识并且学习到很多知识，以下就是对这门课程所做的总结：Python是一种高级动态，完全面向对象的语言，运行程序时，方便快捷。打个比方，计算圆周率类的高级算术题 只要import math math.sqrt()就可以轻松的计算出圆周率，Python就类似于一个超级计算机。本次课程设计不仅锻炼了我们的编程能力，也锻炼了我们的团队合作能力。

通过此次Python课程设计，使我更加扎实的掌握了有关方面的知识，在设计过程中虽然遇到了一些问题，但经过一次又一次的思考，一遍又一遍的检查终于找出了原因所在，也暴露出了前期我在这方面的知识欠缺和经验不足。实践出真知，通过亲自动手制作，使我们掌握的知识不再是纸上谈兵。简易计算器本来就是一个很简单的题目，但是在实现起来依旧出现很多问题。所以以后一定要更加努力的学习专业知识。以后的工作只会比这困难，所以以后要更加努力。

我认为，在这学期的实验中，不仅培养了独立思考、动手操作的能力，在各种其它能力上也都有了提高。更重要的是，在实验课上，我们学会了很多学习的方法。而这是日后最实用的，真的是受益匪浅。要面对社会的挑战，只有不断的学习、实践，再学习、再实践。这对于我们的将来也有很大的帮助。以后，不管有多苦，我想我们都能变苦为乐，找寻有趣的事情，发现其中珍贵的事情。就像中国提倡的艰苦奋斗一样，我们都可以在实验结束之后变的更加成熟，会面对需要面对的事情。

我觉得不管做什么，我们都要相信自己，不能畏惧，不能怕遇到困难，什么都需要去尝试，有些你开始认为很难的事在你尝试之后你可能会发现原来它并没有你以前觉得的那样，自己也是可以的。如果没有自信，没有目标，没有信心就不可能把事情做好，当其他人都在迷茫的时候，自己一定要坚信目标，大学毕业出去即面临找工作，从学习这个专业，到以后从事这方面的工作都需要不断地去学习去实践，这次实践可以给我们敲一个警钟，我们面临毕业，面临择业，需要这些实践经验，在困难面前要勇于尝试，这是这次课程设计给我的最大感想!

以上基本是这次课程设计的体会了， 此次设计也让我明白了思路即出路，也明白了团队合作的重要性，有什么不懂不明白的地方要及时请教或上网查询，只要认真钻研，动脑思考，动手实践，就没有弄不懂的知识，我感觉收获颇丰。同时也非常感谢老师的耐心指导！

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）