**计算器**——设计开发

项目需求分析文档

2019.3.25

目录

1．引言

1．1 编写目的

1．2 文档范围

1. 项目概述

2．1 项目背景

2．2 建设目标

2．3 用户人群

2．4 运行环境

2．5 条件与限制

1. 功能需求

3．1 产品结构图

1. 引言

1．1 编写目的

•方便使用简单计算的人群

1．2 文档范围

•向阅读者分析说明该程序的需求，所用到的软件，开放该程序的目的以及成品效果图。

1. 项目概述

2．1 项目背景

•名称：简单计算器

•代码：

import tkinter  
import math  
  
class jisuan:  
    def \_\_init\_\_(self):  
        #初始化主界面  
        self.root = IMG_256tkinter.Tk()  
        self.root.title('计算器 1.0')  
        self.root['bg'] = '#cadddb'  
        self.root.minsize(215, 300)  
        self.root.resizable(False, False)  
  
        # 设置显示区域的值  
        self.value1 = tkinter.StringVar()  
        self.value1.set('0')  
  
        # 定义一个变量，确定是否按下运算符号  
        self.ispresign = False  
        # 定义一个变量，储存输入的数字  
        self.numlist = []  
        # 定义一个记录是否按了等于号的变量  
        self.isequalsign = 0  
        # 定义一个变量记录是否按下了特殊符号的变量  
        self.specialsign = 0  
  
        #调用界面布局的方法  
        self.show()  
  
        self.root.mainloop()  
    #界面的布局方法  
    def show(self):  
  
        # 显示区域  
        show = tkinter.Label(textvariable=self.value1, anchor='e', bg='white', font=('宋体', 15), bd=10)  
        show.place(x=10, y=10, width=200, height=60)  
  
        # 数字按钮  
        btn0 = tkinter.Button(text='0', command=lambda: self.pressnum('0'))  
        btn0.place(x=10, y=250, width=95, height=40)  
  
        btndian = tkinter.Button(text='.', command=lambda: self.pressnum('.'))  
        btndian.place(x=110, y=250, width=45, height=40)  
  
        btn1 = tkinter.Button(text='1', command=lambda: self.pressnum('1'))  
        btn1.place(x=10, y=205, width=45, height=40)  
  
        btn2 = tkinter.Button(text='2', command=lambda: self.pressnum('2'))  
        btn2.place(x=60, y=205, width=45, height=40)  
  
        btn3 = tkinter.Button(text='3', command=lambda: self.pressnum('3'))  
        btn3.place(x=110, y=205, width=45, height=40)  
  
        btn4 = tkinter.Button(text='4', command=lambda: self.pressnum('4'))  
        btn4.place(x=10, y=160, width=45, height=40)  
  
        btn5 = tkinter.Button(text='5', command=lambda: self.pressnum('5'))  
        btn5.place(x=60, y=160, width=45, height=40)  
  
        btn6 = tkinter.Button(text='6', command=lambda: self.pressnum('6'))  
        btn6.place(x=110, y=160, width=45, height=40)  
  
        btn7 = tkinter.Button(text='7', command=lambda: self.pressnum('7'))  
        btn7.place(x=10, y=115, width=45, height=40)  
  
        btn8 = tkinter.Button(text='8', command=lambda: self.pressnum('8'))  
        btn8.place(x=60, y=115, width=45, height=40)  
  
        btn9 = tkinter.Button(text='9', command=lambda: self.pressnum('9'))  
        btn9.place(x=110, y=115, width=45, height=40)

# 四则运算符号按钮  
        btnjia = tkinter.Button(text='+', command=lambda: self.presign('+'))  
        btnjia.place(x=160, y=160, width=45, height=40)  
  
        btnjian = tkinter.Button(text='-', command=lambda: self.presign('-'))  
        btnjian.place(x=160, y=115, width=45, height=40)  
  
        btncheng = tkinter.Button(text='X', command=lambda: self.presign('\*'))  
        btncheng.place(x=110, y=75, width=45, height=35)  
  
        btnchu = tkinter.Button(text='÷', command=lambda: self.presign('/'))  
        btnchu.place(x=60, y=75, width=45, height=35)  
        # 特殊按钮  
        btndel = tkinter.Button(text='←', command=lambda: self.special('←'))  
        btndel.place(x=160, y=75, width=45, height=35)  
        #  
        btnclear = tkinter.Button(text='CE', command=lambda: self.special('CE'))  
        btnclear.place(x=10, y=75, width=45, height=35)  
        #  
        btnden = tkinter.Button(text='=', command=lambda: self.presseq('='))  
        btnden.place(x=160, y=205, width=45, height=85)  
  
    # 按下数字的函数  
    def pressnum(self,num):  
        # 判断是否按下了运算符号  
        if self.ispresign == True:  
            # 如果按下了符号，将面板数字重置为 0  
            self.value1.set('0')  
            # 将按下运算符号的标志重置  
            self.ispresign = False  
        # 判断是否按下了等于号  
        if self.isequalsign == 1:  
            self.value1.set('0')  
            self.isequalsign = 0  
        if self.specialsign == 1:  
            self.value1.set('0')  
            self.specialsign = 0  
        # 获得面板上的数字  
        oldnum = self.value1.get()  
        # 判断面板上的是否为指定的数据  
        if oldnum == '除数不能为0':  
            return  
        else:  
            # 判断按下的  . 点)是否在已有数据中  
            if num == '.' and num in oldnum:  
                res = oldnum  
            # 判断按下点的时候原有数据是否为0  
            elif num == '.' and oldnum == '0':  
                res = oldnum + '.'  
            # 如果面板上的数字是0，则把第一个数字储存起来  
            elif oldnum == '0':  
                res = num  
            # 如果不是0，则和之前的数字链接起来，给变量self.value1  
            else:  
                res = oldnum + num  
            self.value1.set(res)  
  
    # 按下运算符号的函数()  
    def presign(self,sign):  
        # 判断之前是否已经按过运算符号  
        if self.ispresign == True and self.numlist != []:  # True  表示上一次按过运算符号  
            self.numlist[-1] = sign  # 把第二次按的运算符号替换成上次的运算符号  
        else:  
            # 获得面板上的数字  
            oldnum = self.value1.get()  
            if oldnum == '除数不能为0':  
                self.value1.set('除数不能为0')  
            else:  
                # 先将面板上的数字储存到列表中  
                self.numlist.append(oldnum)  
                # 再将按下的符号储存到列表  
                self.numlist.append(sign)  
        # 按下符号，会将按下符号标志的状态记录下来  
        self.ispresign = True  
 # 按下特殊符号的按钮（C清空所有，CE清空当前，消除最后一位，）  
    def special(self,sign):  
        # 获得当前面板上的内容  
        strs = self.value1.get()  
        if sign == 'C':  
            # 直接清空列表  
            self.numlist.clear()  
            # 将面板上的置为0  
            res = 0  
        elif sign == 'CE':  
            # 将当前面板数字置为0 ，保留列表上次储存的数字  
            res = 0  
        elif sign == '←':  
            if strs != '除数不能为0':  
                # 判断面板上数字是否为0  
                if strs != '0':  
                    # 判断面板上数字是否就一位数  
                    if len(strs) != 1:  
                        # 进行切片操作  
                        res = strs[0:-1]  
                    else:  
                        # 如果就一位数字直接变为0  
                        res = '0'  
                    if self.isequalsign == 1:  
                        res = strs  
                else:  
                    # 若等于0 则不进行操作  
                    res = '0'  
            else:  
                res = strs  
  
        # 将是否按下特殊符号的标志置为  已按（用1 表示）  
        self.specialsign = 1  
        # 将上面处理的数字放到面板中  
        self.value1.set(res)  
  
    # 计算结果  
    def presseq(self,signeq):  
        # 获得面板上数字  
        oldnum = self.value1.get()  
        if self.isequalsign == 1:  
            self.value1.set(oldnum)  
            self.isequalsign = 0  
        elif self.specialsign == 1:  
            self.value1.set(oldnum)  
            self.specialsign = 0  
        elif self.numlist == []:  
            self.value1.set(oldnum)  
        else:  
            if oldnum == '除数不能为0':  
                self.value1.set(oldnum)  
            else:  
                # 判断进行除法操作的时候，除数是否为0  
                if self.numlist[-1] == '/' and eval(oldnum) == 0:  
                    self.value1.set('除数不能为0')  
                    self.numlist.clear()  
                else:  
                    # 将获得的数字添加到列表中  
                    self.numlist.append(oldnum)  
                    # 将列表中的字符串连接成字符串  
                    result = ''.join(self.numlist)  
                    # 将连接的字符串进行运算  
                    self.value1.set(eval(result))  
                    # 清空列表  
                    self.numlist.clear()  
            # 将是否按下等号的标志置为 已按 （用1 表示）  
        self.isequalsign = 1  
  
js = jisuan()

•项目提出者：李敬辉老师

小组人员： 唐可名； 尤挺； 王艺纯； 杨雅淇

小组负责人：尤挺

2．2 建设目标

•预期效果：支持加减乘除、乘方、小数点，运算符优先级为括号>乘方>乘除>加减，同级别运算按照从左向右的顺序计算

2．3 用户人群

•有简单计算需求的人

2．4 运行环境

• Python3

功能需求

3．1 项目结构图：



