数字工程软件架构

——地图数据质检工具（PC 版）

学　院：遥感信息工程学院

班　级：2006 班（20F10）

姓　名：马文卓

学　号：2020302131249

目录

1. 项目要求

设计开发用于空间数据质量检查的应用软件工具，该工具运行于单 PC 机桌面环境，不需要网络连接；说明及评估所采用的软件架构风格。

1.1 项目目标

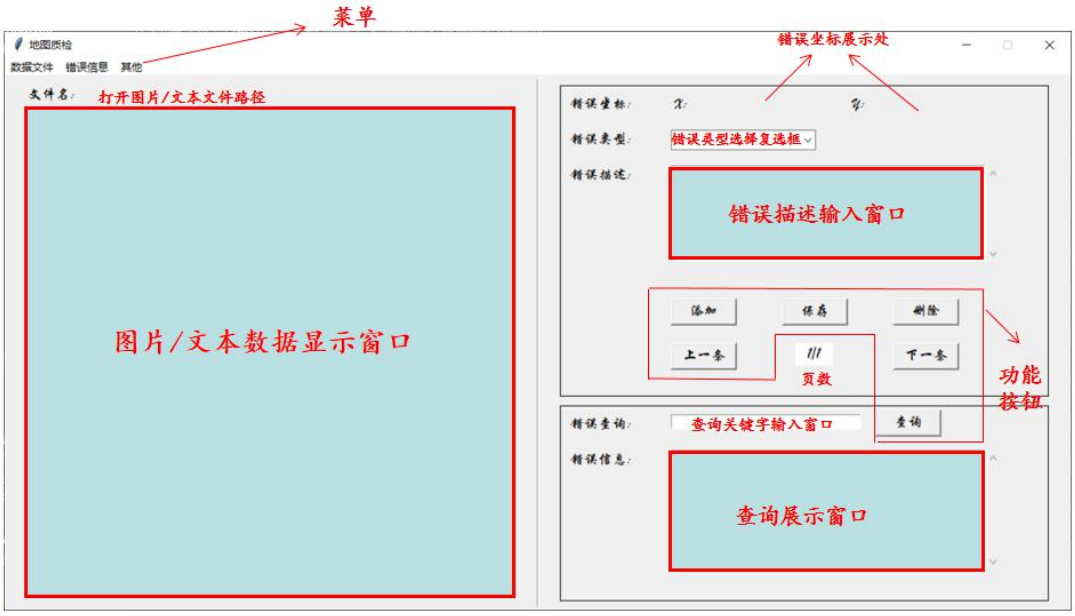
A、地图数据加载：从应用中打开指定的图形或文档类型地图数据文件，并显示在应用程序窗口中。
B、地图数据显示：针对不同类型数据采用不同方式：图形形式的可视化符号显示（主要有栅格及矢量格式类型，原型中至少实现其中一种类型的可视化进行示意）；文本形式直接显示数值内容（原型中可用普通文本文件示意）。
C、错误记录:通过在图形中或文本中指定位置，标记各项错误的位置;指出错误位置后，输入错误类型、错误描述等信息并自动给错误编号；在文件中保存所记录的错误信息。
D、错误记录查询：地图数据加载后，在质检工具中打开已保存的错误信息文件，可通过输入关键字搜索并显示出错情况。

1.2 成果要求

A、软件架构说明。说明所采用的软件架构风格及主要优缺点。
B、软件源码及可执行代码包。实现基本功能要求。提示：可使用所学编程语言平台中 FILE I/O API 及 GUI 构件。

2. 界面说明

2.1 界面 GUI 图



2.2 界面简介

本项目的界面使用的是 Python 的 tkinter 库，主要分为四大板块：分别为菜单栏、数据显示窗、错误信息操作窗、错误信息查询窗。

菜单栏（mainMenu）包括数据文件、错误信息、其他。错误文件子菜单为打开图像文件、打开文本文件；错误信息子菜单为新建错误文件、打开错误文件、删除错误文件；其他子菜单为帮助、退出。

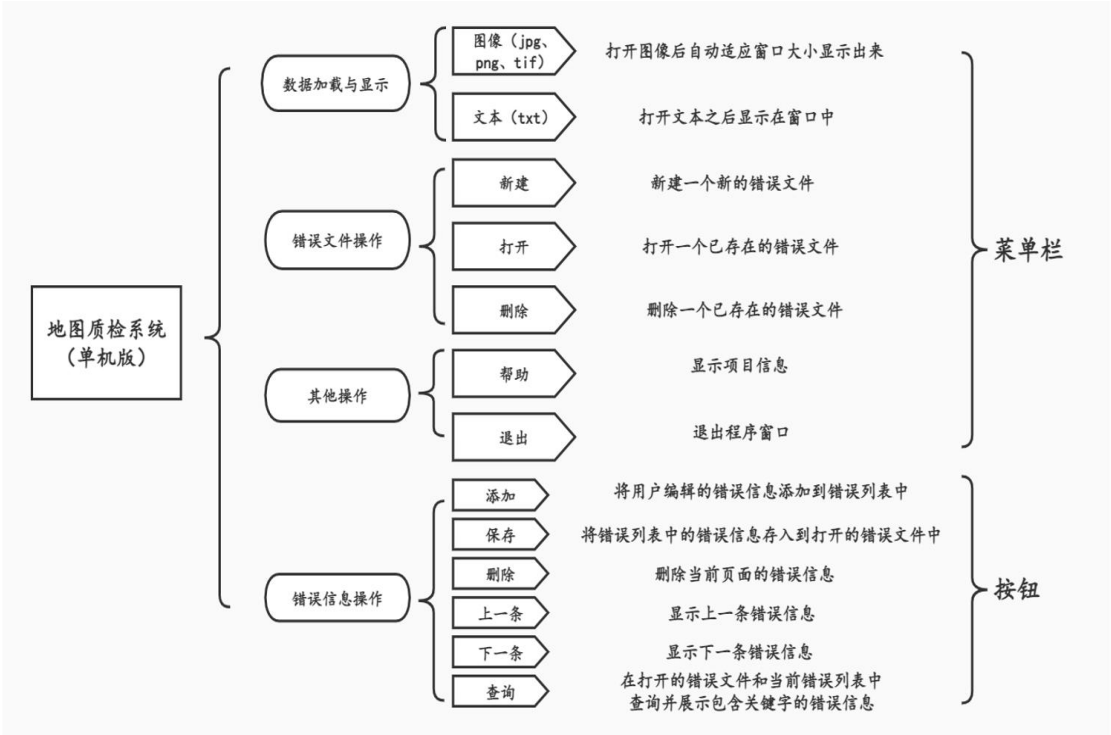
数据显示窗主要为一个用于显示图片或者文本数据的窗口（Frame）和上面显示文件路径的标签（Label）。

错误信息操作窗（Frame）包括坐标显示标签（Label）、错误类型复选框（combobox）、错误描述输入框（SroclledText）、操作（添加、保存、删除、上一条、下一条）按钮（Button）和页面显示标签（Label）

错误信息查询窗(Frame)包括查询关键字输入框(Text)和查询按钮(Button)以及查询信息展示框（SrolledText）。

3. 功能介绍

3.1 功能结构图



3.2 数据加载与显示

（1）打开并显示图像数据：点击【数据文件】下的【打开图像数据】，即可在弹出的文件对话框中选择想要打开的图像数据文件(tif、png、jpg 等格式均可)，打开图像之后会自动显示在数据显示窗并自动适应窗口大小，并且打开文件的路径和文件名会显示在窗口上方。
（2）打开并显示文本数据：点击【数据文件】下的【打开文本数据】，即可在弹出的文件对话框中选择想要打开的文本数据文件（txt 格式），打开文本之后会自动显示在数据显示窗并自动适应窗口大小，并且打开文件的路径和文件名会显示在窗口上方。

3.3 错误信息操作

（1）新建错误文件：点击【错误信息】下的【新建错误文件】，即可在弹出的交互式对话框中输入想要新建的文件名（默认为 Erro.txt）。点击确定之后，程序会再此目录下新建一个错误文本文件，并且将其打开以做存储错误信息之用。注意：如果原本就已经有错误文件处于打开状态，则会被关闭；如果原来错误列表中任然有错误信息，则系统给出提示，提醒如果继续新建错误文件则错误

列表中的错误信息则会丢失。
（2）打开错误文件：点击【错误信息】下的【打开错误文件】，即可在弹出的文件对话框中选择想要打开的错误文件。选择完毕之后，程序打开选定的错误文件，可以将后面添加的错误信息追加保存到打开错误文件的末尾，也可以在查询中查询已经打开的错误文件中的错误信息。注意：如果原本就已经有错误文件处于打开状态，则会被关闭；如果原来错误列表中任然有错误信息，则系统给出提示，提醒如果继续新建错误文件则错误列表中的错误信息则会丢失。
（3）删除错误文件：点击【错误信息】下的【删除错误错误文件】，即可在打开的文件对话框中选择想要删除的错误文件。程序会给出提示，询问是否删除选择的错误文件，如果用户确定，则删除此错误文件。
（4）添加错误信息：双击【数据显示窗口】中认为出错的位置，在错误【错误坐标展示处】会显示用户所点击的位置坐标，然后用户可以再【错误类型选择复选框】中选择错误类型（名称错误、类型错误、属性错误、其他错误），用户还可以在【错误描述输入框】中输入对错误的描述信息，最后点击【添加】按钮即可把此条错误信息存入错误列表中。
（5）删除错误信息：点击【删除】可以从错误列表中删除当前错误信息展示窗里面的这条错误信息
（6）保存错误信息：点击【保存】可以将当前错误列表中的所有信息保存到已经打开的错误文件中。如果没有已经打开的错误文件，系统就会弹出一个文件对话框，可以选择已有的错误文件保存到其中。注意：保存文件之后当前作物列表就会清空。
（7）查看上一条信息：点击【上一条】可以更新错误信息，展示当前页面的上一条错误信息在错误信息操作窗中。如果是第一条错误信息，则弹出提示窗口，提示已经是第一条错误信息。
（8）查看下一条信息：点击【下一条】可以更新错误信息，展示当前页面的下一条错误信息在错误信息操作窗中。如果是最后一条错误信息，则弹出提示窗口，提示已经是最后一条错误信息。
（9）错误查询：在【查询关键字输入窗口】输入查询关键字，点击【查询】按钮即可在当前打开的错误文件（如果已经打开错误文件）和当前错误列表中查询出所有带有这个关键字的错误信息，并且展示在【查询展示窗口】。

3.4 其他功能

（1）展示项目信息：点击【其他】下面的【帮助】，程序会弹出一个窗口，展示作者信息、使用语言、版本信息等
（2）退出程序：点击【其他】下面的【退出】，如果文件还未关闭，则会先关闭文件，再退出程序。

4. 架构风格

4.1 架构风格概述

本项目主要采用了**事件驱动**风格和**面向对象**风格相结合的**异构结构**风格。事件驱动风格主要体现在构件和执行相应事件当中，面向对象风格主要体现在将数据和操作封装在一个一个抽象数据类型或对象当中。这两种风格各有优缺点，在下面将逐一分析。将它们结合到一起，有利于功能的实现和代码的简洁性、重用性、灵活性等。

4.2 风格分析

（1）事件驱动风格：构件不直接调用一个过程，而是触发或广播一个或多个事件系统中其他构件中的过程在一个或多个事件中注册，当一个事件被触发，系统自动调用这个事件中注册的所有过程，即一个事件的触发可能会导致另外一个模块中过程的调用。

优点：

①事件声明者不需要知道那些构件会影响事件，构件之间关联比较弱

②提高软件的复用能力，事件发布及处理都能复用

③系统便于升级，灵活添加事件及处理

缺点：

①构件放弃了对计算机的控制权，完全由系统来决定

②存在数据交换的问题

③数据共享能力低

④系统中各个对象的逻辑关系变得更加复杂

⑤该风格中的正确性验证存在问题

（2）面向对象风格：从现实世界中客观存在的事件出发，强调直接以问题域中的事物为中心来思考问题，根据这些事物的本质特征，将其抽象为系统中的对象，并作为系统的基本构成单位。

优点：

①对象隐藏了其实现细节，所以可以在不影响其他对象地情况下改变对象的实现，不仅使得对象的使用变得简单、方便，而且具有很高的安全性和可靠性。

②设计者可将一些数据存取操作问题分解成一些交互的代理程序集合

缺点：

①当一个对象和其他对象通过过程调用等方式进行交互时，必须知道其他的对象的标识。

②无论何时改变对象的标识，必须修改所有显式调用它的对象，并消除由此带来的一些副作用

（3）异构结构风格：将上述两种风格异构组合，使整个系统具有更加优越的性能、灵活的代码结构和更高的重用性。

5. 关键代码

5.1 程序概述

本次项目采取的主要是 Python 语言，所采用的可视化工具为 tkinter 库。由于所涉及的对象比较少，因此程序框架为一个类（Main），其下面的各种方法实现各种功能。类图和框架如下：

```
class Main:#主程序框架类
    def __init__(self):#构造函数…

    def layout(self):#主界面的初始布局设置…

    def openImageFile(self):#打开图像文件并显示…

    def openDataFile(self):#打开数据文件并显示…

    def newErroFile(self):#新建错误文件…

    def openErroFile(self):#打开错误文件…

    def deleteErroFile(self):#删除错误文件…

    def Help(self):#帮助…

    def Exit(self):#退出…

    def getXY(self,event):#获取点击处坐标，并显示…

    def addErro(self):#添加错误到self.erro…

    def saveErro(self):#保存错误文件…

    def deleteErro(self):#删除此条错误信息…

    def latter(self):#上一条…

    def next(self):#下一条…

    def search(self):#在已经打开的错误文件和已经添加的错误信息中查询错误信息…

    def clear_erro(self):#清空错误区…

    def set_erroinface(self,x,y,erro_type,erro_detail):#设置错误界面…
```

5.2 界面设置

主要封装在 layout 函数中，里面实现了各种构件的布局工作。部分关键代码如下：

```
def layout(self):#主界面的初始布局设置
    self.root.title('地图质检')#设置窗口标题
    self.root.geometry('1200x600')#设置窗口大小
    self.root.resizable(width=False,height=False)#大小不可变
    #菜单选项设置
    self.mainmenu=Menu(self.root)#创建主菜单
    self.menuData=Menu(self.mainmenu)#菜单分组
    self.mainmenu.add_cascade(label='数据文件',menu=self.menuData)#添加一个文件分组
    self.menuData.add_command(label='打开图像数据',command=self.openImageFile)#在下面添加选项
    self.menuData.add_command(label='打开文本数据',command=self.openDataFile)
    self.menuErro=Menu(self.mainmenu)
    self.mainmenu.add_cascade(label='错误信息',menu=self.menuErro)
    self.menuErro.add_command(label='新建错误文件',command=self.newErroFile)
    self.menuErro.add_command(label='打开错误文件',command=self.openErroFile)
    self.menuErro.add_command(label='删除错误文件',command=self.deleteErroFile)
    self.menuOther=Menu(self.mainmenu)
    self.mainmenu.add_cascade(label='其他',menu=self.menuOther)
    self.menuOther.add_command(label='帮助',command=self.Help)
    self.menuOther.add_command(label='退出',command=self.Exit)
    self.root.config(menu=self.mainmenu)#设为菜单
    #分割线设置
    self.sep=Separator(self.root,orient=VERTICAL)#竖直分割线
    self.sep.pack(padx=10,pady=3,fill='y',expand=True)
    #框架设置
    self.frame1=tk.Frame(self.root,bd=1,height=550,width=550,relief='solid')#此框架为显示地图窗口
    self.frame1.place(x=25,y=35)
    self.frame2=tk.Frame(self.root,bd=1,height=350,width=550,relief='solid')#此框架为添加错误记录窗口
    self.frame2.place(x=625,y=10)
```

5.3 数据文件操作

（1）打开图像数据：在打开前做了数据清理工作。使用 resize 函数将使图片标签适应窗口大小。最后和获取坐标的函数 getXY 建立联系。

```
def openImageFile(self):#打开图像文件并显示
    for data in self.frame1.winfo_children():
        data.destroy()  # 清理已打开的数据
    filename=tkf.askopenfilename()#打开文件
    self.label2.config(text=filename)#在上方显示文件路径
    if filename!='':
        im=Image.open(filename)#打开图片
        im=im.resize((545,545),Image.ANTIALIAS)#调整大小
        self.image=ImageTk.PhotoImage(im)#将图像存入全局变量中
        label_image=tk.Label(self.frame1,image=self.image)#将图像借助标签显示
        label_image.pack()
        label_image.bind("<Double-Button-1>",self.getXY)#双击左键获取坐标
```

（2）打开文本数据：基本与打开图像数据类似。

```
def openDataFile(self):#打开数据文件并显示
    for data in self.frame1.winfo_children():
        data.destroy()  # 清理已打开的数据
    filename=tkf.askopenfilename()#打开文件
    self.label2.config(text=filename)#在上方显示文件路径
    with open(filename, encoding='utf-8', errors='ignore') as f:
        content = f.readlines()  # 按行读取
    self.txt=content#将文本数据存到全局变量中
    # 将读取到的内容放入Listbox
    lb = tk.Listbox(self.frame1,height=30,width=80)
    lb.pack()
    for line in content:
        lb.insert('end', line)
    lb.bind("<Double-Button-1>",self.getXY)#双击左键获取坐标
```

5.4 错误文件操作

（1）新建错误文件：新建前先做数据清理工作，然后获取用户想要新建的文件名，以追加的方式新建错误文件。

```
def newErroFile(self):#新建错误文件
    if self.erro!=[]:
        if messagebox.askokcancel('提示', '确定打开新的错误文件吗？\n如果错误信息未保存\n信息会丢失')==False:
            return None
    self.erro=[]#错误列表清空
    self.index=1#错误索引归零
    self.label12.config(text=str('1/1'))#重置页面
    self.clear_erro()
    if self.file!=None:#若果已打开文件则关闭
        self.file.close()
    filename = askstring('新建错误文件', prompt='输入您创建的文件名', initialvalue='Erro')#打开对话框让用户输入文件名
    self.file=open(filename+'.txt','a+',encoding='utf-8')#新建文件
    self.file.write('12')
    if self.file!=None:#新建成功，给出提示
        messagebox.askokcancel('提示', '成功新建文件')
    else:#新建失败，给出提示
        messagebox.askokcancel('提示', '新建文件失败')
```

（2）打开错误文件：打开文件前先做清理工作，清理完毕之后使用 askopenfilename 来获取用户想要打开的错误文件。以追加的方式打开该错误文件，然后给出提示。

```
def openErroFile(self):#打开错误文件
    if self.erro!=[]:
        if messagebox.askokcancel('提示', '确定打开新的错误文件吗？\n如果错误信息未保存\n信息会丢失')==False:
            return None
    self.erro=[]#错误列表清空
    self.index=1#错误索引归零
    self.label12.config(text=str('1/1'))#重置页面
    self.clear_erro()
    if self.file!=None:#如果已经有文件打开就将其关闭
        self.file.close()
    filename=tkf.askopenfilename()#打开文件对话框，储存文件名
    self.file=open(filename,'a+',encoding='utf-8')#以追加的方式打开文件
    if self.file!=None:#打开成功，给出提示
       messagebox.askokcancel('提示', '成功打开文件')
    else:#打开失败，给出提示
        messagebox.askokcancel('提示', '打开文件失败')
```

（3）删除错误文件：使用 askokcancel 询问用户是否删除文件，得到确认之后使用 os.path.exists() 方法删除错误文件。

```
def deleteErroFile(self):#删除错误文件
    filename =tkf.askopenfilename()  # 文件路径
    if messagebox.askokcancel('提示', '是否删除'+filename)==True:
        os.remove(filename)#删除文件
        if os.path.exists(filename):  # 如果文件存在
            messagebox.askokcancel('提示', '删除文件失败')
        else:
            messagebox.askokcancel('提示', '删除文件成功')
```

5.5 错误信息操作

（1）获取错误坐标：使用 event 的 x 和 y 属性直接获取点击处的坐标，并且通过标签的 config 方法显示在相应区域。

```
def getXY(self,event):#获取点击处坐标，并显示
    self.x=event.x#存储坐标
    self.y=event.y
    self.label10.config(text=str(self.x))#改变标签中的数值
    self.label11.config(text=str(self.y))
```

（2）添加错误信息：添加前先判断坐标值是否合理，合理的话才添加，并且改变页面标签信息。

```
def addErro(self):#添加错误到self.erro
    if self.x>=0 and self.y>=0:#坐标合理时
        erro_type=self.combobox.get()#获取错误类型
        erro_detail=self.text1.get(1.0,END)#获取错误描述
        self.erro+=[[self.x,self.y,erro_type,erro_detail]]#加入一条错误
        self.clear_erro()#清空
        self.index=len(self.erro)+1
        self.label12.config(text=str(str(self.index)+'/'+str(len(self.erro)+1)))#更新下面的页面信息
    else:
        messagebox.askokcancel('提示', '请双击图像以获得一个合理的坐标')
```

（3）删除错误信息：现判断是否还有错误信息在错误列表中，如果没有给出提示，如果有则使用 pop 函数删除，删除的同时改变页面所展示的信息。

```
def deleteErro(self):#删除此条错误信息
    if len(self.erro)==0:#如果没有错误信息
        messagebox.askokcancel('提示', '无错误信息')
    elif len(self.erro)==self.index:#如果是最后一项错误信息，则特殊处理（清空页面）
        self.erro.pop(self.index-1)
        self.clear_erro()#清空界面
        self.label12.config(text=str(self.index)+'/'+str(len(self.erro)+1))#更新下面的页面信息
    else:
        if self.index>=len(self.erro)+1:
            messagebox.askokcancel('提示', '此条信息未编辑')
        else:
            print(self.index)
            self.erro.pop(self.index-1)
            self.label12.config(text=str(self.index)+'/'+str(len(self.erro)+1))#更新下面的页面信息
            self.set_erroinface(self.erro[self.index-1][0],self.erro[self.index-1][1],self.erro[self.index-1][2],self.erro[self.index-1][3])
```

（4）保存错误信息：如果没有错误文件打开，则使用 asksaveasfilename 获取用户想要存储的文件名，并且以追加的方式打开文件。使用 write 韩式将错误列表中的错误信息按照一定的格式写入打开错误文件的末尾。

```
def saveErro(self):#保存错误文件
    if self.file==None:#没有新建或者打开文件
        filename=tkf.asksaveasfilename()#弹出保存文件对话框
        if filename=='':#如果没有选择文件
            return None
        self.file=open(filename,'a+',encoding='utf-8')
    for i in range(len(self.erro)):
        self.file.write('('+str(self.erro[i][0])+','+str(self.erro[i][1])+')    '+self.erro[i][2]+'    '+self.erro[i][3])#写入信息
    messagebox.askokcancel('提示', '文件保存成功')
    self.clear_erro()
    self.erro=[]
    self.file.close()
    self.file=None
    self.index=1
    self.label12.config(text=str('1/1'))#重置页面
```

（5）查看上一条错误信息：先判断是否为第一条信息，如果是就给出提示；如果不是就把上一条信息展示出来。

```
def latter(self):#上一条
    if self.index==1:
        messagebox.askokcancel('提示', '已经是第一条')
    else:
        self.index-=1
        self.label12.config(text=str(self.index)+'/'+str(len(self.erro)+1))#更新下面的页面信息
        self.set_erroinface(self.erro[self.index-1][0],self.erro[self.index-1][1],self.erro[self.index-1][2],self.erro[self.index-1][3])#
```

（6）查看下一条错误信息：先判断是否为最后一条信息，如果是则给出提示；否则，跳转到下一条信息，并且显示出来。

```
def next(self):#下一条
    if self.index==len(self.erro)+1:
            messagebox.askokcancel('提示', '已经是最后一条')
    else:
        if self.index==len(self.erro):#如果是倒数第二条，则清空界面（最后一条还未编辑）
            self.index+=1
            self.label12.config(text=str(self.index)+'/'+str(len(self.erro)+1))#更新下面的页面信息
            self.clear_erro()
        else:
            self.index+=1
            self.label12.config(text=str(self.index)+'/'+str(len(self.erro)+1))#更新下面的页面信息
            self.set_erroinface(self.erro[self.index-1][0],self.erro[self.index-1][1],self.erro[self.index-1][2],self.erro[self.index-1][3])
```

（7）查询错误信息：这里获取输入框里面的关键字之后要注意获取的关键字是带有换行符”\n”的，所以先去掉之后再使用 find 函数在错误文件和错误列表中查询。

```
def search(self):#在已经打开的错误文件和已经添加的错误信息中查询错误信息
    self.text3.delete(1.0,END)#清空展示框
    search_content=self.text2.get(1.0,END)#获取输入的关键词
    search_content=search_content[0:len(search_content)-1]#上面读取的字符串最后带有\n一定要去除
    search_result=[]#储存查询结果
    if self.file!=None:#如果有错误文件打开，则先在错误文件中查询
        self.file.seek(0,0)#将指针调到文件开始
        erro_content=self.file.readlines()#按行读取
        for i in range(len(erro_content)):
            if erro_content[i].find(search_content)>=0:#如果关键字被包含在这条错误信息中
                search_result+=[erro_content[i]]
    for i in range(len(self.erro)):#遍历所有错误信息
        #将每条错误信息串联起来便于查询，中间用空格隔开避免数字连接造成查询出现偏差
        erro_content=str(self.erro[i][0])+'  '+str(self.erro[i][1])+'  '+self.erro[i][2]+'  '+self.erro[i][3]
        if erro_content.find(search_content)>=0:#如果关键字被包含在这条错误信息中
            search_result+=['('+str(self.erro[i][0])+','+str(self.erro[i][1])+')    '+self.erro[i][2]+'    '+self.erro[i][3]]
    for i in range(len(search_result)):#展示查询结果
        self.text3.insert(END,search_result[i])
```

5.6 其他操作

（1）帮助：使用提示窗口的形式给出信息

```
def Help(self):#帮助
    messagebox.showinfo('帮助','作者：马文卓\n版本:1.0\n语言:Python\n库:tkinter')
```

（2）退出：退出前先询问是否退出

```
def Exit(self):#退出
    if messagebox.askokcancel('提示', '您确定退出程序吗？')==True:#弹出提示框等待选择
        if self.file!=None:#如果文件还未关闭，则关闭文件
            self.file.close()
        self.root.withdraw()#如果确认则退出窗口
        sys.exit() #退出程序
```

6. 结果展示

代码运行结果见附件《运行视频》，详细代码见附件《main.py》，运行文件为《main.exe》，项目相关图示均见附录。

7. 总结评价

本项目主要实现了PC单击版的地图质检系统，包括数据文件载入、错误文件操作、错误信息操作这三个大块。完成度方面，实现了所有功能要求，基本达到了项目最初的预期。与此同时，还添加了一些优化功能，如帮助、退出等，以及一系列的边界限制操作，大大的增加了软件的鲁棒性和可用性，赋予了其高度的稳定性。

当然由于本人水平的局限性，项目程序仍然存在很多不足。在数据文件加载方面，由于时间原因只实现了各种图片格式(jpg、png、tif 等)和文本格式(txt)，后续还可以实现 doc 等文件格式的打开方式。在错误文件操作中，对于打开错误文件就会清空错误列表的问题，后续会继续研究。

本项目还有较强的扩展性，后续还可以像多用户数据库连接方向扩展。