

**数据结构与算法**

**设计实践报告**

**课程编号： 2121040011**

**课程名称： 数据结构与算法设计实践**

**设计名称： 学生信息管理系统**

**院 系： 计算机与网络空间安全学院**

**专 业： 网络空间安全**

**组 长： 徐岩 2020212063014**

**组 员： 张健：2020212063016**

**邱纪霖：2020212063017**

**周靖桂：2020212063038**

**莫丹亦佳：2020212063037**

**陈雪婷：2020212063043**

**任课老师： 高玉龙**

1. **实验名称**

学生信息管理系统

（Python+数据库）

**2.实验目的**

掌握数据结构和算法设计编写过程，实现对学生成绩信息的添加、删除、修改和查询等操作，其中学生信息包括学号、姓名、成绩。

**3.实验要求**

（1） 认真阅读和掌握和本实验相关的教材内容。

（2） 认真阅读和掌握相关算法的程序。

（3） 上机运行程序。

（4） 按照对学生信息的操作需要，运行主程序，输出正确运行结果，并结合程序进行分析。

**4.设计思路**

本次项目是学生信息管理系统，是数据结构中“增删改查”的经典体现，为此，在本次实验中着重体现出对于这四个功能的合理应用以及对于数据结构相关知识的融合。基于项目具体的功能研发，我们分为了三个模块进行实现：

1. 菜单栏采用python自带的tkinter库以及Windows自带的Treeview组件进行设计的可视化界面。丰富了信息系统的感官体验。
2. 学生具体输入的数据的存储以及对于整体数据的增加、删除、修改、查找等功能的实现是基于数据库的功能实现（采用sqlite3实现底层代码框架）
3. 在实现“增删改查”基本功能的前提下进行项目具体的完善和功能的优化，我们增添了对于“学生电话”、“学生性别”的数据添加，同时，考虑到每一个数据输入框读写的数据的合法性，我们对存储进数据库的数据进行了检查，譬如对于“性别”一栏出现的输入不合法以及空白等情况下的处理。

**5.实验内容**

**一、对于菜单栏可视化的设计：**

1、引入pythhon自带的tkinter库

*import* tkinter *as* tk *# 引入python自带的用户登录界面库*

*import* tkinter.ttk

*import* tkinter.messagebox

1. 对于可视化框架布局的具体设计

使用了tkinter库中的界面布局函数以及Windows自带的Treeview组件库设置了“学号”、“姓名”、“姓名”、“成绩”等输入框以及菜单栏上方的“文件”“功能”“关于”的按钮选项。

*# 用户界面菜单栏的设置*

class StudentManageSystem037(object):

    def \_\_init\_\_(*self*):  *# 构造方法初始化变量*

*# print("初始化")*

*# 窗体初始化*

*self*.root037 = tk.Tk()

*self*.menuTabF037 = tk.Frame(*self*.root037)

*self*.studentInputF037 = tk.Frame(*self*.root037)

*self*.studentShowF037 = tk.Frame(*self*.root037)

*# 添加滚动条*

*# 设置主窗体的title*

*self*.root037.title("学生信息管理系统 V1.0")

*# 设置主窗体的大小*

*self*.root037.geometry('500x350')

*self*.root037.resizable(False, False)

*self*.nameToDelete037 = tkinter.StringVar('')

*# self.path037 = "./assets/data.db"*

*self*.path037 = "data.db"

*# self.window = 0*

    def menuMain037(*self*):

        print("menu")

        menubar037 = tk.Menu(*self*.root037)  *# 创建菜单栏*

        file\_menu037 = tk.Menu(menubar037, *tearoff*=False)  *# 创建空菜单*

        file\_menu037.add\_command(*label*="打开")  *# 向file\_menu菜单中添加label*

        file\_menu037.add\_command(*label*="保存")

        file\_menu037.add\_command(*label*="退出", *command*=*self*.root037.quit)

        menubar037.add\_cascade(*label*="文件", *menu*=file\_menu037)  *# 将file\_menu菜单添加到菜单栏*

        do\_menu037 = tk.Menu(menubar037, *tearoff*=False)  *# 创建空菜单*

        do\_menu037.add\_command(*label*="添加学生", *command*=*self*.insertStudentInfo037)

        do2\_menu037 = tk.Menu(do\_menu037)  *# 二级菜单*

        do2\_menu037.add\_command(*label*="修改一个学生", *command*=*self*.updateStudentInfo037)

        do2\_menu037.add\_command(*label*="批量修改学生信息", *command*=*self*.updateStudentInfo037)

        do\_menu037.add\_cascade(*label*='修改学生', *menu*=do2\_menu037)

        do\_menu037.add\_command(*label*="删除学生", *command*=*self*.delStudentInfo037)

        menubar037.add\_cascade(*label*="功能", *menu*=do\_menu037)  *# 将file\_menu菜单添加到菜单栏*

        myself\_menu037 = tk.Menu(menubar037, *tearoff*=False)  *# 创建空菜单*

        myself\_menu037.add\_command(*label*="系统信息", *command*=*self*.showSysInfo037)

        menubar037.add\_cascade(*label*="关于", *menu*=myself\_menu037)  *# 将file\_menu菜单添加到菜单栏*

*self*.root037.config(*menu*=menubar037)  *# display the menu*

    def table037(*self*):

*# 在窗口上放置用来显示通信录信息的表格，使用Treeview组件实现*

        frame037 = tk.Frame(*self*.root037)

        frame037.place(*x*=0, *y*=130, *width*=490, *height*=200)

*# Treeview组件（参考Microsoft关于Treeview的官方文档）*

*# 使用 Windows 窗体 TreeView 控件，可以为用户显示节点层次结构，就像在 Windows 操作系统的 Windows 资源管理器功能的左窗格中显示文件和文件夹一样。*

*# 树视图中的每个节点可能包含其他节点，称为 子节点。*

*# 以按展开或折叠的方式显示父节点或包含子节点的节点。*

*# 还可以通过将树视图的 CheckBoxes 属性设置为 true 来显示在节点旁边带有复选框的树视图。*

*# 然后，通过将节点的 Checked 属性设置为 true 或 false，可以采用编程方式来选中或清除节点。*

*self*.tree037 = tk.ttk.Treeview(frame037, *columns*=(

            'c1', 'c2', 'c3', 'c4', 'c5'), *show*="headings")

*# 滚动条*

*# 使用Scrollbar库*

        scrollBar037 = Scrollbar(frame037, *orient*=VERTICAL)

        scrollBar037.pack(*side*=tkinter.RIGHT, *fill*=Y)

*self*.tree037.config(*yscrollcommand*=scrollBar037.set)

*# Treeview组件与垂直滚动条结合*

        scrollBar037.config(*command*=*self*.tree037.yview)

*# self.tree037.heading('c0', text='id')*

*self*.tree037.heading('c1', *text*='学号')

*self*.tree037.heading('c2', *text*='姓名')

*self*.tree037.heading('c3', *text*='性别')

*self*.tree037.heading('c4', *text*='电话')

*self*.tree037.heading('c5', *text*='成绩')

*# self.tree037.heading('c6', text='QQ')*

*# self.tree037.column('c0', width=70, anchor='center')*

*self*.tree037.column('c1', *width*=90, *anchor*='center')

*self*.tree037.column('c2', *width*=90, *anchor*='center')

*self*.tree037.column('c3', *width*=90, *anchor*='center')

*self*.tree037.column('c4', *width*=90, *anchor*='center')

*self*.tree037.column('c5', *width*=110, *anchor*='center')

*# self.tree037.column('c6', width=70, anchor='center')*

*self*.tree037.pack(*side*=tkinter.LEFT)

*self*.tree037.bind('<ButtonRelease>', *self*.midify\_item037)

1. **处理用户输入的数据：**
2. 设置数据输入函数：

def studentInput037(*self*):

        tk.Label(*self*.studentInputF037, *text*="学号：").grid(*row*=0, *column*=0)

*self*.stuNum037 = tk.StringVar(*self*.studentInputF037)  *# 学号*

*self*.numberInput037 = tk.Entry(

*self*.studentInputF037, *width*=15, *textvariable*=*self*.stuNum037)

*self*.numberInput037.grid(*row*=0, *column*=1)

        tk.Label(*self*.studentInputF037, *text*="姓名：").grid(*row*=0, *column*=2)

*self*.stuName037 = tk.StringVar(*self*.studentInputF037)  *# 姓名*

        tk.Entry(*self*.studentInputF037, *width*=15, *textvariable*=*self*.stuName037).grid(

*row*=0, *column*=3, *sticky*='w')

        tk.Label(*self*.studentInputF037, *text*="性别：").grid(*row*=0, *column*=4)

        stuSex037 = tk.StringVar(*self*.studentInputF037)  *# 性别*

*self*.stuSex037 = tk.ttk.Combobox(*self*.studentInputF037, *width*=10,

*values*=('男', '女'))

*self*.stuSex037.grid(*row*=0, *column*=5)

*# tk.Entry(self.studentInputF037, width=10, textvariable=self.stuSex037).grid(row=0, column=5)*

        tk.Label(*self*.studentInputF037, *text*="电话：").grid(*row*=1, *column*=0)

*self*.stuPhone037 = tk.StringVar(*self*.studentInputF037)  *# 电话*

        tk.Entry(*self*.studentInputF037, *width*=15,

*textvariable*=*self*.stuPhone037).grid(*row*=1, *column*=1)

        tk.Label(*self*.studentInputF037, *text*="成绩：").grid(*row*=1, *column*=2)

*self*.stuAdd037 = tk.StringVar(*self*.studentInputF037)  *# 成绩*

        tk.Entry(*self*.studentInputF037, *width*=15,

*textvariable*=*self*.stuAdd037).grid(*row*=1, *column*=3)

*self*.studentInputF037.pack(*pady*=10)

1. 设置数据清除函数

*# 数据清除函数*

    def clearText037(*self*, *target*):

        '''

        清除输入框文本

        '''

*target*.delete(0, len(*target*.get()))

*# 数据清除函数*

    def studentInputSetNull037(*self*):

        '''

        清空输入

        :return:

        '''

*self*.stuNum037.set("")

*self*.stuName037.set("")

*# self.stuSex037.set("")*

*self*.clearText037(*self*.stuSex037)

*self*.stuPhone037.set("")

*self*.stuAdd037.set("")

*self*.results037 = ''

*self*.numberInput037['state'] = 'normal'

    def menuTab037(*self*):

        tk.Button(*self*.menuTabF037, *text*="添加", *command*=*self*.insertStudentInfo037).grid(

*row*=0, *column*=0, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="删除", *command*=*self*.delStudentInfo037).grid(

*row*=0, *column*=1, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="修改", *command*=*self*.updateStudentInfo037).grid(

*row*=0, *column*=2, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="查找", *command*=*self*.getStudentInfo037).grid(

*row*=0, *column*=3, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="清空", *command*=*self*.studentInputSetNull037).grid(

*row*=0, *column*=4, *ipadx*=10, *padx*=15)

*self*.menuTabF037.pack(*pady*=5)

    def studentShow037(*self*):

*self*.getStudentInfo037()

*pass*

    def getStudentInfo037(*self*):

*# self.connectDb037("search")*

*# 删除表格中原来的所有行*

*for* row037 *in* *self*.tree037.get\_children():

*self*.tree037.delete(row037)

1. 读取存入数据库的数据，将数据存入列表并进行储存

*# 读取数据库中的所有数据*

*with* sqlite3.connect(*self*.path037) *as* conn037:

            cur037 = conn037.cursor()

*try*:

                cur037.execute('SELECT \* FROM studentInfo ORDER BY id ASC')

*except* OperationalError *as* error:

                print(error)

*if* str(error) == "no such table: studentInfo":

                    cur037.execute(

                        'CREATE TABLE "studentInfo" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,"number" varchar,"name" varchar,"sex" varchar,"telephone" integer,"address" varchar)')

                    cur037.execute('SELECT \* FROM studentInfo ORDER BY id ASC')

            temp037 = cur037.fetchall()

*# 把数据插入列表*

*for* i, item *in* enumerate(temp037):

            print(i, item)

*self*.tree037.insert(*parent*='', *index*=i, *iid*=i,

*text*='', *values*=(item[1:]))

1. 对于输入框中输入的具体口令进行检查，防止出现数据不合法、数据空白等极端情况。

*# 在窗口上放置用于添加通信录的按钮，并设置按钮单击事件函数*

*# 对于用户输入的姓名输入框进行一个判断*

    def buttonAddClick037(*self*):

*# 检查姓名*

        name037 = *self*.stuName037.get().strip()

*if* name037 == '':

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='必须输入姓名')

*return*

*# if btn\_text.get() == '添加':*

*# 获取选择的性别*

        sex037 = *self*.stuSex037.get()

*if* sex037 not in ('男', '女'):

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='性别不合法')

*return*

*# 检查学号*

        number037 = *self*.stuNum037.get().strip()

*# 学号不能重复*

*with* sqlite3.connect(*self*.path037) *as* conn:

            cur037 = conn.cursor()

            cur037.execute(

                'SELECT COUNT(id) from studentInfo where number="' + number037 + '"')

            c = cur037.fetchone()[0]

*if* c != 0:

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='学号不能重复')

*return*

*# 检查成绩*

        address037 = *self*.stuAdd037.get().strip()

*if* address037 == '':

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='必须输入成绩')

*return*

*# 检查电话号码*

        telephone037 = *self*.stuPhone037.get().strip()

*if* telephone037 == '' or (not telephone037.isdigit()):

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='电话号码必须是数字')

*return*

1. 所有数据通过检查，插入数据库，并对要存入数据库的数据进行匹配检查。

*# 所有输入都通过检查，插入数据库*

*# 对要存入的数据进行匹配检查*

        sql037 = 'INSERT INTO studentInfo(name,sex,number,address,telephone) VALUES("' \

                 + name037 + '","' + sex037 + '","' + number037 + '","' + address037 + '","' \

                 + telephone037 + '")'

*self*.connectDb037(sql037)

*self*.studentInputSetNull037()

*self*.bindData037()

        tkinter.messagebox.showinfo(*title*='成功', *message*='添加成功')

*pass*

    def bindData037(*self*):

*self*.getStudentInfo037()

*pass*

    def get\_select037(*self*):

        selected = *self*.tree037.focus()

*if* selected:

*self*.nameToDelete037.set(*self*.tree037.item(selected, 'values')[0])

*return* *self*.tree037.item(selected, 'values')

    def midify\_item037(*self*, *event*):

*# print(event)*

*# print(event.widget)*

        item = *self*.get\_select037()

*if* item is None:

            item = *self*.get\_select037()

*self*.stuNum037.set(item[0])

*self*.stuName037.set(item[1])

*self*.clearText037(*self*.stuSex037)

*self*.stuSex037.insert(0, item[2])

*self*.stuPhone037.set(item[3])

*self*.stuAdd037.set(item[4])

*self*.numberInput037['state'] = 'readonly'

*# entryName.insert(0, item[1])*

*# comboSex.insert(0, item[2])*

*# entryAge.insert(0, item[3])*

*# entryDepartment.insert(0, item[4])*

*# entryTelephone.insert(0, item[5])*

*# entryQQ.insert(0, item[6])*

        print("修改")

    def insertStudentInfo037(*self*):

        '''

        添加学生

        :return:

        '''

*self*.buttonAddClick037()

*pass*

    def delStudentInfo037(*self*):

        '''

        删除学生

        :return:

        '''

        number = *self*.stuNum037.get()

*if* number == '':

            tk.messagebox.showinfo('提示', "请先输入信息再进行对应功能操作")

*return*

*#     # 如果已经选择了一条通信录，执行SQL语句将其删除*

        sql = 'DELETE FROM studentInfo WHERE number="' + number + '"'

*self*.connectDb037(sql)

*self*.studentInputSetNull037()

*self*.getStudentInfo037()

        tkinter.messagebox.showinfo('恭喜', '删除成功')

    def buttonModifyClick037(*self*):

        '''

        修改

        :return:

        '''

*# 从数据库读取数据*

*# 检查姓名*

        name037 = *self*.stuName037.get().strip()

*if* name037 == '':

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='必须输入姓名')

*return*

*# if btn\_text.get() == '添加':*

*# 获取选择的性别*

        sex037 = *self*.stuSex037.get()

*if* sex037 not in ('男', '女'):

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='性别不合法')

*return*

*# 检查学号*

        number037 = *self*.stuNum037.get().strip()

*# 检查成绩*

        address037 = *self*.stuAdd037.get().strip()

*if* address037 == '':

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='必须输入成绩')

*return*

*# 检查电话号码*

        telephone037 = *self*.stuPhone037.get().strip()

*if* telephone037 == '' or (not telephone037.isdigit()):

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='电话号码必须是数字')

*return*

*# 所有输入都通过检查，插入数据库*

        sql037 = 'UPDATE studentInfo SET name = "' + name037 + '",sex = "' + sex037 + '",number = "' + number037 + \

                 '",address = "' + address037 + '",telephone = "' + \

                 telephone037 + '" WHERE number=' + number037

*self*.connectDb037(sql037)

*self*.studentInputSetNull037()

*self*.numberInput037['state'] = 'normal'

        tkinter.messagebox.showinfo(*title*='成功', *message*='修改成功')

*self*.bindData037()

*pass*

    def updateStudentInfo037(*self*):

        '''

        修改学生信息

        :return:

        '''

*self*.buttonModifyClick037()

    def connectDb037(*self*, *do*):

        '''用来执行SQL语句，尤其是INSERT和DELETE语句'''

*with* sqlite3.connect(*self*.path037) *as* conn:

            cur037 = conn.cursor()

            cur037.execute(*do*)

            conn.commit()

*pass*

    def showSysInfo037(*self*):

        tk.messagebox.showinfo(

            '提示信息', "学生信息管理系统 V1.0\n单击表格项目可以修改，修改后即可添加，和删除。")

    def gui037(*self*):

*# print("ui 渲染")*

*self*.menuMain037()

*self*.studentInput037()

*self*.menuTab037()

*self*.table037()

*self*.studentShow037()

*# self.window = 1*

1. **对于功能的优化** 增加了清空功能：

*# 数据清除函数*

    def clearText037(*self*, *target*):

        '''

        清除输入框文本

        '''

*target*.delete(0, len(*target*.get()))

*# 数据清除函数*

    def studentInputSetNull037(*self*):

        '''

        清空输入

        :return:

        '''

*self*.stuNum037.set("")

*self*.stuName037.set("")

*# self.stuSex037.set("")*

*self*.clearText037(*self*.stuSex037)

*self*.stuPhone037.set("")

*self*.stuAdd037.set("")

*self*.results037 = ''

*self*.numberInput037['state'] = 'normal'

    def menuTab037(*self*):

        tk.Button(*self*.menuTabF037, *text*="添加", *command*=*self*.insertStudentInfo037).grid(

*row*=0, *column*=0, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="删除", *command*=*self*.delStudentInfo037).grid(

*row*=0, *column*=1, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="修改", *command*=*self*.updateStudentInfo037).grid(

*row*=0, *column*=2, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="查找", *command*=*self*.getStudentInfo037).grid(

*row*=0, *column*=3, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="清空", *command*=*self*.studentInputSetNull037).grid(

*row*=0, *column*=4, *ipadx*=10, *padx*=15)

*self*.menuTabF037.pack(*pady*=5)

    def studentShow037(*self*):

*self*.getStudentInfo037()

*pass*

    def getStudentInfo037(*self*):

*# self.connectDb037("search")*

*# 删除表格中原来的所有行*

*for* row037 *in* *self*.tree037.get\_children():

*self*.tree037.delete(row037)

*# 读取数据库中的所有数据*

*with* sqlite3.connect(*self*.path037) *as* conn037:

            cur037 = conn037.cursor()

*try*:

                cur037.execute('SELECT \* FROM studentInfo ORDER BY id ASC')

*except* OperationalError *as* error:

                print(error)

*if* str(error) == "no such table: studentInfo":

                    cur037.execute(

                        'CREATE TABLE "studentInfo" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,"number" varchar,"name" varchar,"sex" varchar,"telephone" integer,"address" varchar)')

                    cur037.execute('SELECT \* FROM studentInfo ORDER BY id ASC')

            temp037 = cur037.fetchall()

*# 把数据插入列表*

*for* i, item *in* enumerate(temp037):

            print(i, item)

*self*.tree037.insert(*parent*='', *index*=i, *iid*=i,

*text*='', *values*=(item[1:]))

**6.实验代码：**

*# 这是一个基于python以及少量数据库语言编写的关于学生信息管理系统的一个文件*

*# 这些代码中集中体现的数据结构是基于数据库中对于sqlite3应用于将数据以列表形式进行读取和存储的编码结构*

*# 该套程序中的可视化界面是基于python自带的thinter库进行设计编写的*

*import* tkinter *as* tk *# 引入python自带的用户登录界面库*

*import* tkinter.ttk

*import* tkinter.messagebox

*import* sqlite3 *# SQLite是一个进程内的库，实现了自给自足的、无服务器的、零配置的、事务性的 SQL 数据库引擎。它是一个零配置的数据库，这意味着与其他数据库不一样，您不需要在系统中配置。*

*# 就像其他数据库，SQLite 引擎不是一个独立的进程，可以按应用程序需求进行静态或动态连接。*

*# SQLite 直接访问其存储文件。*

*from* sqlite3 *import* OperationalError *# SQLite 是一个C语言库，它可以提供一种轻量级的基于磁盘的数据库，这种数据库不需要独立的服务器进程，也允许需要使用一种非标准的 SQL 查询语言来访问它。一些应用程序可以使用 SQLite 作为内部数据存储。*

*# 可以用它来创建一个应用程序原型，然后再迁移到更大的数据库，比如 PostgreSQL 或 Oracle。*

*from* tkinter *import* VERTICAL, Scrollbar, Y *# Scrollbar是一个滚动条库，可以和其它的组件库一起使用*

*# 用户界面菜单栏的设置*

class StudentManageSystem037(object):

    def \_\_init\_\_(*self*):  *# 构造方法初始化变量*

*# print("初始化")*

*# 窗体初始化*

*self*.root037 = tk.Tk()

*self*.menuTabF037 = tk.Frame(*self*.root037)

*self*.studentInputF037 = tk.Frame(*self*.root037)

*self*.studentShowF037 = tk.Frame(*self*.root037)

*# 添加滚动条*

*# 设置主窗体的title*

*self*.root037.title("学生信息管理系统 V1.0")

*# 设置主窗体的大小*

*self*.root037.geometry('500x350')

*self*.root037.resizable(False, False)

*self*.nameToDelete037 = tkinter.StringVar('')

*# self.path037 = "./assets/data.db"*

*self*.path037 = "data.db"

*# self.window = 0*

    def menuMain037(*self*):

        print("menu")

        menubar037 = tk.Menu(*self*.root037)  *# 创建菜单栏*

        file\_menu037 = tk.Menu(menubar037, *tearoff*=False)  *# 创建空菜单*

        file\_menu037.add\_command(*label*="打开")  *# 向file\_menu菜单中添加label*

        file\_menu037.add\_command(*label*="保存")

        file\_menu037.add\_command(*label*="退出", *command*=*self*.root037.quit)

        menubar037.add\_cascade(*label*="文件", *menu*=file\_menu037)  *# 将file\_menu菜单添加到菜单栏*

        do\_menu037 = tk.Menu(menubar037, *tearoff*=False)  *# 创建空菜单*

        do\_menu037.add\_command(*label*="添加学生", *command*=*self*.insertStudentInfo037)

        do2\_menu037 = tk.Menu(do\_menu037)  *# 二级菜单*

        do2\_menu037.add\_command(*label*="修改一个学生", *command*=*self*.updateStudentInfo037)

        do2\_menu037.add\_command(*label*="批量修改学生信息", *command*=*self*.updateStudentInfo037)

        do\_menu037.add\_cascade(*label*='修改学生', *menu*=do2\_menu037)

        do\_menu037.add\_command(*label*="删除学生", *command*=*self*.delStudentInfo037)

        menubar037.add\_cascade(*label*="功能", *menu*=do\_menu037)  *# 将file\_menu菜单添加到菜单栏*

        myself\_menu037 = tk.Menu(menubar037, *tearoff*=False)  *# 创建空菜单*

        myself\_menu037.add\_command(*label*="系统信息", *command*=*self*.showSysInfo037)

        menubar037.add\_cascade(*label*="关于", *menu*=myself\_menu037)  *# 将file\_menu菜单添加到菜单栏*

*self*.root037.config(*menu*=menubar037)  *# display the menu*

    def table037(*self*):

*# 在窗口上放置用来显示通信录信息的表格，使用Treeview组件实现*

        frame037 = tk.Frame(*self*.root037)

        frame037.place(*x*=0, *y*=130, *width*=490, *height*=200)

*# Treeview组件（参考Microsoft关于Treeview的官方文档）*

*# 使用 Windows 窗体 TreeView 控件，可以为用户显示节点层次结构，就像在 Windows 操作系统的 Windows 资源管理器功能的左窗格中显示文件和文件夹一样。*

*# 树视图中的每个节点可能包含其他节点，称为 子节点。*

*# 以按展开或折叠的方式显示父节点或包含子节点的节点。*

*# 还可以通过将树视图的 CheckBoxes 属性设置为 true 来显示在节点旁边带有复选框的树视图。*

*# 然后，通过将节点的 Checked 属性设置为 true 或 false，可以采用编程方式来选中或清除节点。*

*self*.tree037 = tk.ttk.Treeview(frame037, *columns*=(

            'c1', 'c2', 'c3', 'c4', 'c5'), *show*="headings")

*# 滚动条*

*# 使用Scrollbar库*

        scrollBar037 = Scrollbar(frame037, *orient*=VERTICAL)

        scrollBar037.pack(*side*=tkinter.RIGHT, *fill*=Y)

*self*.tree037.config(*yscrollcommand*=scrollBar037.set)

*# Treeview组件与垂直滚动条结合*

        scrollBar037.config(*command*=*self*.tree037.yview)

*# self.tree037.heading('c0', text='id')*

*self*.tree037.heading('c1', *text*='学号')

*self*.tree037.heading('c2', *text*='姓名')

*self*.tree037.heading('c3', *text*='性别')

*self*.tree037.heading('c4', *text*='电话')

*self*.tree037.heading('c5', *text*='成绩')

*# self.tree037.heading('c6', text='QQ')*

*# self.tree037.column('c0', width=70, anchor='center')*

*self*.tree037.column('c1', *width*=90, *anchor*='center')

*self*.tree037.column('c2', *width*=90, *anchor*='center')

*self*.tree037.column('c3', *width*=90, *anchor*='center')

*self*.tree037.column('c4', *width*=90, *anchor*='center')

*self*.tree037.column('c5', *width*=110, *anchor*='center')

*# self.tree037.column('c6', width=70, anchor='center')*

*self*.tree037.pack(*side*=tkinter.LEFT)

*self*.tree037.bind('<ButtonRelease>', *self*.midify\_item037)

*# 学生输入数据函数*

    def studentInput037(*self*):

        tk.Label(*self*.studentInputF037, *text*="学号：").grid(*row*=0, *column*=0)

*self*.stuNum037 = tk.StringVar(*self*.studentInputF037)  *# 学号*

*self*.numberInput037 = tk.Entry(

*self*.studentInputF037, *width*=15, *textvariable*=*self*.stuNum037)

*self*.numberInput037.grid(*row*=0, *column*=1)

        tk.Label(*self*.studentInputF037, *text*="姓名：").grid(*row*=0, *column*=2)

*self*.stuName037 = tk.StringVar(*self*.studentInputF037)  *# 姓名*

        tk.Entry(*self*.studentInputF037, *width*=15, *textvariable*=*self*.stuName037).grid(

*row*=0, *column*=3, *sticky*='w')

        tk.Label(*self*.studentInputF037, *text*="性别：").grid(*row*=0, *column*=4)

        stuSex037 = tk.StringVar(*self*.studentInputF037)  *# 性别*

*self*.stuSex037 = tk.ttk.Combobox(*self*.studentInputF037, *width*=10,

*values*=('男', '女'))

*self*.stuSex037.grid(*row*=0, *column*=5)

*# tk.Entry(self.studentInputF037, width=10, textvariable=self.stuSex037).grid(row=0, column=5)*

        tk.Label(*self*.studentInputF037, *text*="电话：").grid(*row*=1, *column*=0)

*self*.stuPhone037 = tk.StringVar(*self*.studentInputF037)  *# 电话*

        tk.Entry(*self*.studentInputF037, *width*=15,

*textvariable*=*self*.stuPhone037).grid(*row*=1, *column*=1)

        tk.Label(*self*.studentInputF037, *text*="成绩：").grid(*row*=1, *column*=2)

*self*.stuAdd037 = tk.StringVar(*self*.studentInputF037)  *# 成绩*

        tk.Entry(*self*.studentInputF037, *width*=15,

*textvariable*=*self*.stuAdd037).grid(*row*=1, *column*=3)

*self*.studentInputF037.pack(*pady*=10)

*# 数据清除函数*

    def clearText037(*self*, *target*):

        '''

        清除输入框文本

        '''

*target*.delete(0, len(*target*.get()))

*# 数据清除函数*

    def studentInputSetNull037(*self*):

        '''

        清空输入

        :return:

        '''

*self*.stuNum037.set("")

*self*.stuName037.set("")

*# self.stuSex037.set("")*

*self*.clearText037(*self*.stuSex037)

*self*.stuPhone037.set("")

*self*.stuAdd037.set("")

*self*.results037 = ''

*self*.numberInput037['state'] = 'normal'

    def menuTab037(*self*):

        tk.Button(*self*.menuTabF037, *text*="添加", *command*=*self*.insertStudentInfo037).grid(

*row*=0, *column*=0, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="删除", *command*=*self*.delStudentInfo037).grid(

*row*=0, *column*=1, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="修改", *command*=*self*.updateStudentInfo037).grid(

*row*=0, *column*=2, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="查找", *command*=*self*.getStudentInfo037).grid(

*row*=0, *column*=3, *ipadx*=10, *padx*=15)

        tk.Button(*self*.menuTabF037, *text*="清空", *command*=*self*.studentInputSetNull037).grid(

*row*=0, *column*=4, *ipadx*=10, *padx*=15)

*self*.menuTabF037.pack(*pady*=5)

    def studentShow037(*self*):

*self*.getStudentInfo037()

*pass*

    def getStudentInfo037(*self*):

*# self.connectDb037("search")*

*# 删除表格中原来的所有行*

*for* row037 *in* *self*.tree037.get\_children():

*self*.tree037.delete(row037)

*# 读取数据库中的所有数据*

*with* sqlite3.connect(*self*.path037) *as* conn037:

            cur037 = conn037.cursor()

*try*:

                cur037.execute('SELECT \* FROM studentInfo ORDER BY id ASC')

*except* OperationalError *as* error:

                print(error)

*if* str(error) == "no such table: studentInfo":

                    cur037.execute(

                        'CREATE TABLE "studentInfo" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,"number" varchar,"name" varchar,"sex" varchar,"telephone" integer,"address" varchar)')

                    cur037.execute('SELECT \* FROM studentInfo ORDER BY id ASC')

            temp037 = cur037.fetchall()

*# 把数据插入列表*

*for* i, item *in* enumerate(temp037):

            print(i, item)

*self*.tree037.insert(*parent*='', *index*=i, *iid*=i,

*text*='', *values*=(item[1:]))

*# 在窗口上放置用于添加通信录的按钮，并设置按钮单击事件函数*

*# 对于用户输入的姓名输入框进行一个判断*

    def buttonAddClick037(*self*):

*# 检查姓名*

        name037 = *self*.stuName037.get().strip()

*if* name037 == '':

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='必须输入姓名')

*return*

*# if btn\_text.get() == '添加':*

*# 获取选择的性别*

        sex037 = *self*.stuSex037.get()

*if* sex037 not in ('男', '女'):

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='性别不合法')

*return*

*# 检查学号*

        number037 = *self*.stuNum037.get().strip()

*# 学号不能重复*

*with* sqlite3.connect(*self*.path037) *as* conn:

            cur037 = conn.cursor()

            cur037.execute(

                'SELECT COUNT(id) from studentInfo where number="' + number037 + '"')

            c = cur037.fetchone()[0]

*if* c != 0:

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='学号不能重复')

*return*

*# 检查成绩*

        address037 = *self*.stuAdd037.get().strip()

*if* address037 == '':

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='必须输入成绩')

*return*

*# 检查电话号码*

        telephone037 = *self*.stuPhone037.get().strip()

*if* telephone037 == '' or (not telephone037.isdigit()):

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='电话号码必须是数字')

*return*

*# 所有输入都通过检查，插入数据库*

*# 对要存入的数据进行匹配检查*

        sql037 = 'INSERT INTO studentInfo(name,sex,number,address,telephone) VALUES("' \

                 + name037 + '","' + sex037 + '","' + number037 + '","' + address037 + '","' \

                 + telephone037 + '")'

*self*.connectDb037(sql037)

*self*.studentInputSetNull037()

*self*.bindData037()

        tkinter.messagebox.showinfo(*title*='成功', *message*='添加成功')

*pass*

    def bindData037(*self*):

*self*.getStudentInfo037()

*pass*

    def get\_select037(*self*):

        selected = *self*.tree037.focus()

*if* selected:

*self*.nameToDelete037.set(*self*.tree037.item(selected, 'values')[0])

*return* *self*.tree037.item(selected, 'values')

    def midify\_item037(*self*, *event*):

*# print(event)*

*# print(event.widget)*

        item = *self*.get\_select037()

*if* item is None:

            item = *self*.get\_select037()

*self*.stuNum037.set(item[0])

*self*.stuName037.set(item[1])

*self*.clearText037(*self*.stuSex037)

*self*.stuSex037.insert(0, item[2])

*self*.stuPhone037.set(item[3])

*self*.stuAdd037.set(item[4])

*self*.numberInput037['state'] = 'readonly'

*# entryName.insert(0, item[1])*

*# comboSex.insert(0, item[2])*

*# entryAge.insert(0, item[3])*

*# entryDepartment.insert(0, item[4])*

*# entryTelephone.insert(0, item[5])*

*# entryQQ.insert(0, item[6])*

        print("修改")

    def insertStudentInfo037(*self*):

        '''

        添加学生

        :return:

        '''

*self*.buttonAddClick037()

*pass*

    def delStudentInfo037(*self*):

        '''

        删除学生

        :return:

        '''

        number = *self*.stuNum037.get()

*if* number == '':

            tk.messagebox.showinfo('提示', "请先输入信息再进行对应功能操作")

*return*

*#     # 如果已经选择了一条通信录，执行SQL语句将其删除*

        sql = 'DELETE FROM studentInfo WHERE number="' + number + '"'

*self*.connectDb037(sql)

*self*.studentInputSetNull037()

*self*.getStudentInfo037()

        tkinter.messagebox.showinfo('恭喜', '删除成功')

    def buttonModifyClick037(*self*):

        '''

        修改

        :return:

        '''

*# 从数据库读取数据*

*# 检查姓名*

        name037 = *self*.stuName037.get().strip()

*if* name037 == '':

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='必须输入姓名')

*return*

*# if btn\_text.get() == '添加':*

*# 获取选择的性别*

        sex037 = *self*.stuSex037.get()

*if* sex037 not in ('男', '女'):

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='性别不合法')

*return*

*# 检查学号*

        number037 = *self*.stuNum037.get().strip()

*# 检查成绩*

        address037 = *self*.stuAdd037.get().strip()

*if* address037 == '':

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='必须输入成绩')

*return*

*# 检查电话号码*

        telephone037 = *self*.stuPhone037.get().strip()

*if* telephone037 == '' or (not telephone037.isdigit()):

            tkinter.messagebox.showerror(*title*='很抱歉', *message*='电话号码必须是数字')

*return*

*# 所有输入都通过检查，插入数据库*

        sql037 = 'UPDATE studentInfo SET name = "' + name037 + '",sex = "' + sex037 + '",number = "' + number037 + \

                 '",address = "' + address037 + '",telephone = "' + \

                 telephone037 + '" WHERE number=' + number037

*self*.connectDb037(sql037)

*self*.studentInputSetNull037()

*self*.numberInput037['state'] = 'normal'

        tkinter.messagebox.showinfo(*title*='成功', *message*='修改成功')

*self*.bindData037()

*pass*

    def updateStudentInfo037(*self*):

        '''

        修改学生信息

        :return:

        '''

*self*.buttonModifyClick037()

    def connectDb037(*self*, *do*):

        '''用来执行SQL语句，尤其是INSERT和DELETE语句'''

*with* sqlite3.connect(*self*.path037) *as* conn:

            cur037 = conn.cursor()

            cur037.execute(*do*)

            conn.commit()

*pass*

    def showSysInfo037(*self*):

        tk.messagebox.showinfo(

            '提示信息', "学生信息管理系统 V1.0\n单击表格项目可以修改，修改后即可添加，和删除。")

    def gui037(*self*):

*# print("ui 渲染")*

*self*.menuMain037()

*self*.studentInput037()

*self*.menuTab037()

*self*.table037()

*self*.studentShow037()

*# self.window = 1*

*# 调用上述函数*

def main():

    xt = StudentManageSystem037()

    xt.gui037()

    tk.mainloop()

*if* \_\_name\_\_ == "\_\_main\_\_":

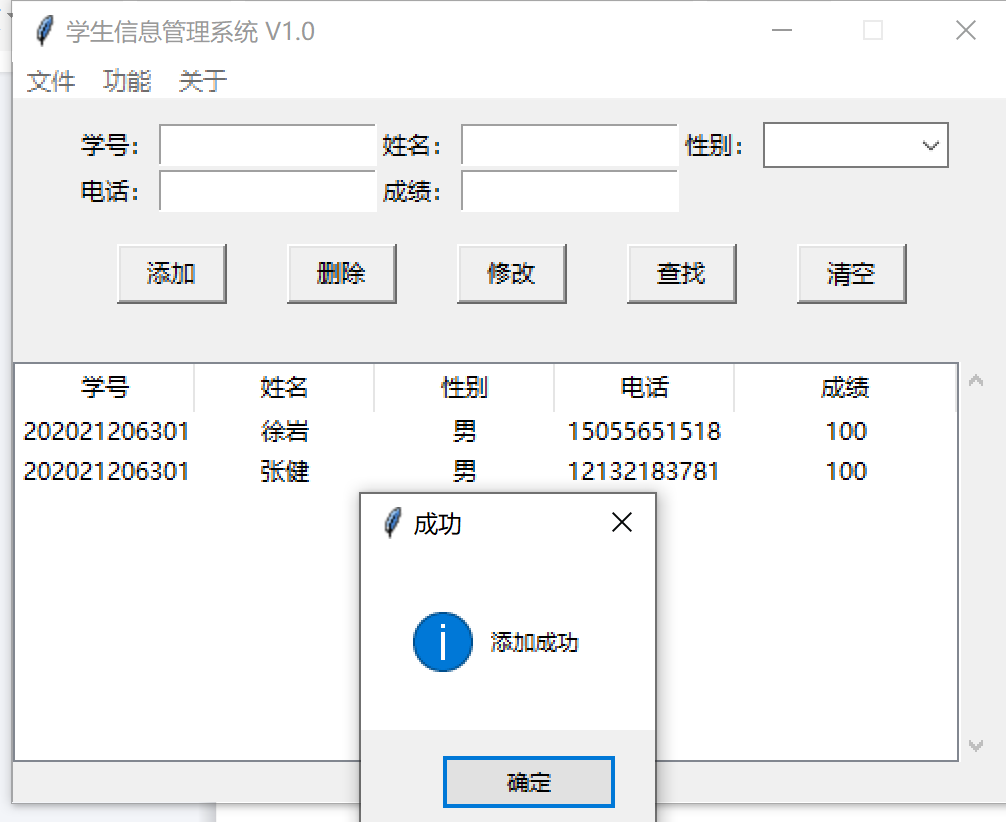
    main()

1. **实验结果**

**初始化界面：**



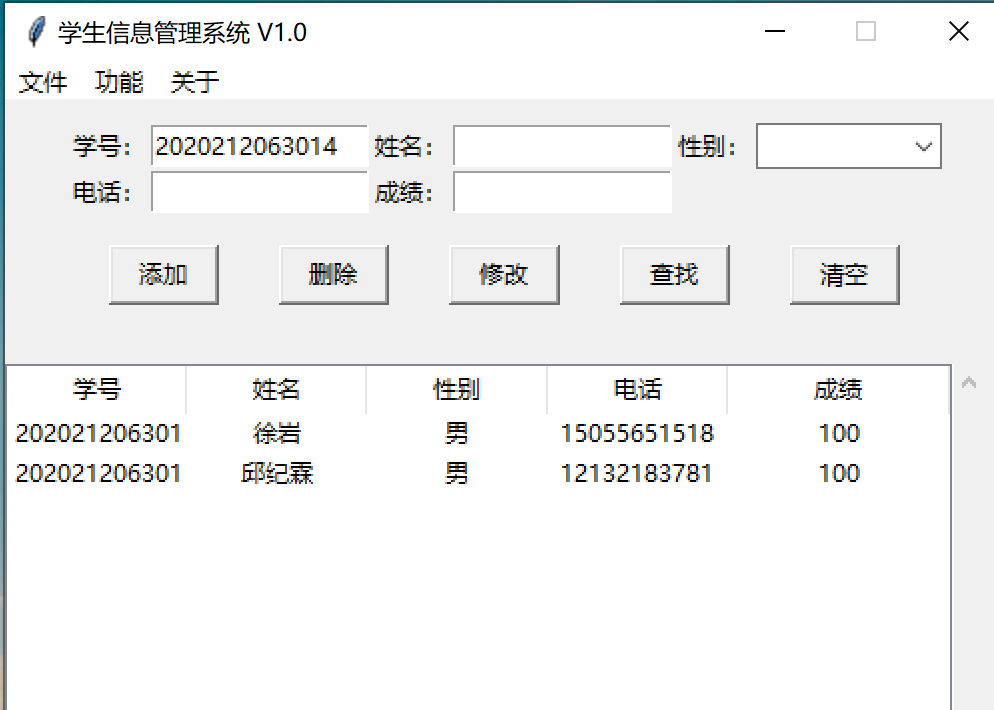
**录入学生信息：**



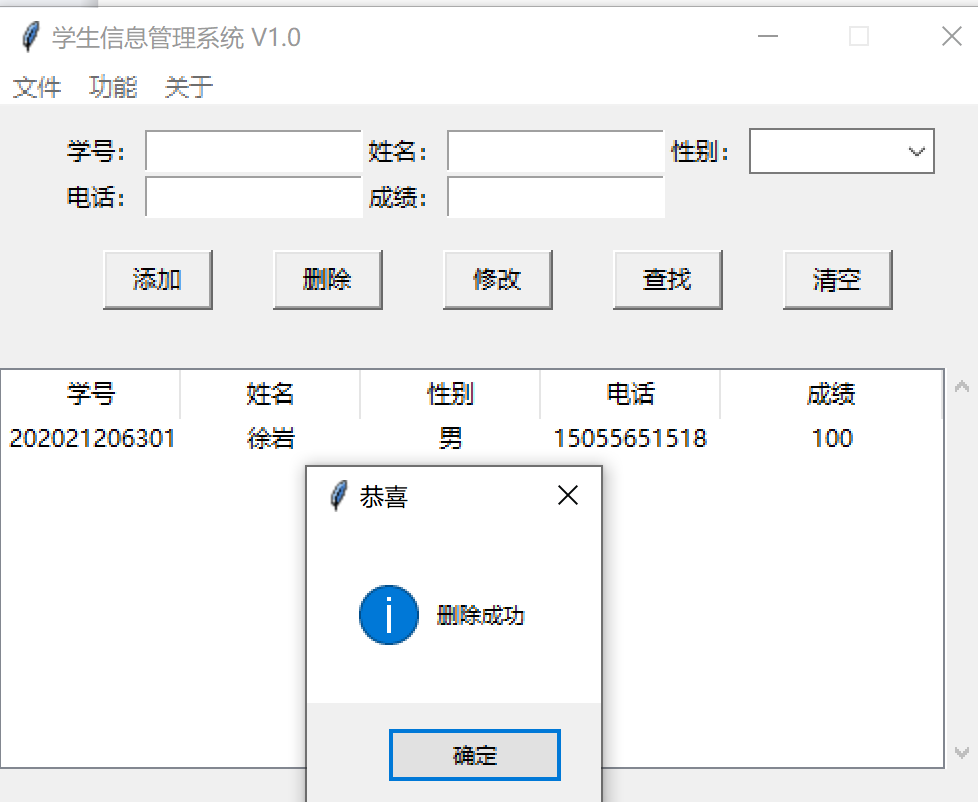
**修改学生信息：**



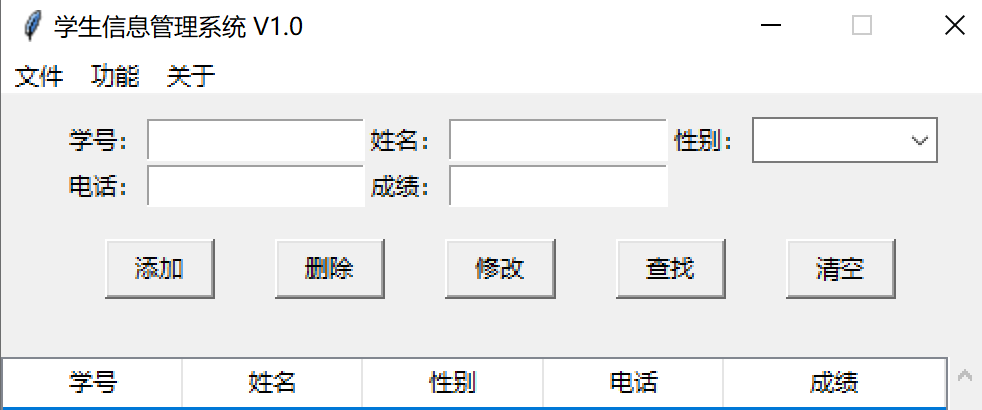
**查找学生信息：**



**删除学生信息：**



**清空学生信息：**

****

**8.程序分析**

**（该程序设计的优点，编写中你们遇到的问题，以及是如何解决的）**

**项目的优点：**

1. 完美实现了本次实验的要求，能够方便简洁的增加学生个人信息包括姓名、学号和成绩。并在此基础上增加了查找、修改、删除、清空等功能，最大化的模拟了信息管理系统所具备的基础职能。
2. 充分利用了python语言在数据结构中的灵活性和便捷性，利用了多方库，使用了可视化面板的操作界面，使得该信息系统使用起来更加方便快捷，对于信息的处理和操作更加灵活。

3、采用了数据库的功能特性，在存储和检查数据方面趋向于功能化的形式。充分利用了信息管理系统的功能属性和特点。真正意义上实现了对于数据的存储和修改，而不是一个简单的模型架构。使得数据结构的代码思想能够进行复现。

**遇到的问题：**

1. **如何解决对于tkinter库制作的菜单栏的按钮布局和整体框架的搭建出现误差和功能的不对等？**

解决方案：查阅官方文档，反复进行调试，譬如对于菜单栏中按钮的布局设置根据文档使用了grid的布局方式，grid采用行列确定位置，行列号只是指定放置的相对位置，而不是实际的像素坐标。行列交汇处为一个单元格。每一列中，列宽由这一列中最宽的单元格确定。每一行中，行高由这一行中最高的单元格决定。最后使得菜单栏的整体设置布局完成。

1. **如何实现对于存入数据库的数据进行检查和匹配？**

解决方案：可以通过sql语句查询和确认数据入库情况。例如“ conn = sqlite3.connect("customers.db") cursor = conn.execute("select id,name,age from cname") for row in cursor: print ('ID = ', row[0],' NAME = ',row[1],' AGE = ', row[2]) print ('Operation done successfully') conn.close”(至此，便将Json格式的数据存储到SQLite3数据库中了，可以进行后续的分析和操作。

1. **如何使得输入的数据以顺序表的形式进行存储？**

解决方案：在python的官方实现中，list就是一种采用****分离式技术实现的动态顺序表****。“顺序表”表示数据的连续存储，而一个列表里面可以存放不同类型的数据元素。“分离”表示了内存中不再存放了具体的数据，而是存放了数据的地址即引用。由于int、char等数据类型所需的内存大小不一致，而系统分配的地址所需内存恰好一样，都是4个字节，所以这样给顺序表中不同数据类型的地址计算带来了方便。