

实 验 报 告

## 实验一《动态链表的设计与应用》

**学生姓名 姚婉欣**

**学号 216001887**

**课程名称 数据结构与算法**

**实验学期 2022 至 2023 学年 第 二 学期**

**所在学院 信息科学与工程学院**

**专业班级 21计算机专业网络技术与工程方向**

**任课教师 赵杰煜**

**实验成绩**

**一、实验题目**

动态链表的设计与应用

**二、实验目的**

1、掌握单链表的基本操作：插入、删除、查找等运算。

2、掌握单向链表的存储特点及其实现。

**三、实验设备与环境**

微型计算机、macOS操作系统 、pyqt6、QT designer、Pycharm系列软件

**四、实验内容**

利用单向循环链表实现一个班级学生信息管理（数据录入、插入、删除、查找，清空链表内容等）。

**五、概要设计（思路、算法、步骤等）**

**六、详细设计（核心代码、算法流程图等）**

UI界面：

*# Form implementation generated from reading ui file 'link.ui'  
#  
# Created by: PyQt6 UI code generator 6.1.0  
#  
# WARNING: Any manual changes made to this file will be lost when pyuic6 is  
# run again. Do not edit this file unless you know what you are doing.*from PyQt6 import QtCore, QtGui, QtWidgets  
  
  
class Ui\_Dialog(object):  
 def setupUi(self, Dialog):  
 Dialog.setObjectName("Dialog")  
 Dialog.setEnabled(True)  
 Dialog.resize(693, 593)  
 self.namebox = QtWidgets.QLineEdit(Dialog)  
 self.namebox.setGeometry(QtCore.QRect(110, 100, 113, 25))  
 self.namebox.setText("")  
 self.namebox.setObjectName("namebox")  
 self.idbox = QtWidgets.QLineEdit(Dialog)  
 self.idbox.setGeometry(QtCore.QRect(110, 140, 113, 25))  
 self.idbox.setText("")  
 self.idbox.setObjectName("idbox")  
 self.hobbybox = QtWidgets.QLineEdit(Dialog)  
 self.hobbybox.setGeometry(QtCore.QRect(110, 220, 113, 25))  
 self.hobbybox.setText("")  
 self.hobbybox.setObjectName("hobbybox")  
 self.label = QtWidgets.QLabel(Dialog)  
 self.label.setGeometry(QtCore.QRect(70, 100, 45, 25))  
 self.label.setObjectName("label")  
 self.label\_2 = QtWidgets.QLabel(Dialog)  
 self.label\_2.setGeometry(QtCore.QRect(70, 140, 45, 25))  
 self.label\_2.setObjectName("label\_2")  
 self.label\_3 = QtWidgets.QLabel(Dialog)  
 self.label\_3.setGeometry(QtCore.QRect(70, 180, 45, 25))  
 self.label\_3.setObjectName("label\_3")  
 self.label\_4 = QtWidgets.QLabel(Dialog)  
 self.label\_4.setGeometry(QtCore.QRect(70, 220, 45, 25))  
 self.label\_4.setObjectName("label\_4")  
 self.label\_5 = QtWidgets.QLabel(Dialog)  
 self.label\_5.setGeometry(QtCore.QRect(270, 100, 45, 25))  
 self.label\_5.setObjectName("label\_5")  
 self.sex = QtWidgets.QComboBox(Dialog)  
 self.sex.setGeometry(QtCore.QRect(310, 100, 71, 25))  
 self.sex.setObjectName("sex")  
 self.sex.addItem("")  
 self.sex.addItem("")  
 self.label\_6 = QtWidgets.QLabel(Dialog)  
 self.label\_6.setGeometry(QtCore.QRect(270, 140, 45, 25))  
 self.label\_6.setObjectName("label\_6")  
 self.nationbox = QtWidgets.QLineEdit(Dialog)  
 self.nationbox.setGeometry(QtCore.QRect(310, 140, 71, 25))  
 self.nationbox.setText("")  
 self.nationbox.setObjectName("nationbox")  
 self.birthdaybox = QtWidgets.QDateEdit(Dialog)  
 self.birthdaybox.setGeometry(QtCore.QRect(110, 180, 131, 25))  
 self.birthdaybox.setObjectName("birthdaybox")  
 self.input = QtWidgets.QPushButton(Dialog)  
 self.input.setEnabled(True)  
 self.input.setGeometry(QtCore.QRect(300, 220, 113, 32))  
 self.input.setMouseTracking(False)  
 self.input.setObjectName("input")  
 self.putphoto = QtWidgets.QPushButton(Dialog)  
 self.putphoto.setGeometry(QtCore.QRect(480, 200, 113, 32))  
 self.putphoto.setObjectName("putphoto")  
 self.delete\_2 = QtWidgets.QPushButton(Dialog)  
 self.delete\_2.setGeometry(QtCore.QRect(360, 440, 81, 32))  
 self.delete\_2.setMouseTracking(False)  
 self.delete\_2.setObjectName("delete\_2")  
 self.changephoto = QtWidgets.QPushButton(Dialog)  
 self.changephoto.setGeometry(QtCore.QRect(480, 460, 113, 32))  
 self.changephoto.setObjectName("changephoto")  
 self.hobby\_2 = QtWidgets.QLineEdit(Dialog)  
 self.hobby\_2.setGeometry(QtCore.QRect(110, 420, 113, 25))  
 self.hobby\_2.setText("")  
 self.hobby\_2.setObjectName("hobby\_2")  
 self.label\_7 = QtWidgets.QLabel(Dialog)  
 self.label\_7.setGeometry(QtCore.QRect(270, 340, 45, 25))  
 self.label\_7.setObjectName("label\_7")  
 self.label\_8 = QtWidgets.QLabel(Dialog)  
 self.label\_8.setGeometry(QtCore.QRect(70, 380, 45, 25))  
 self.label\_8.setObjectName("label\_8")  
 self.label\_9 = QtWidgets.QLabel(Dialog)  
 self.label\_9.setGeometry(QtCore.QRect(70, 420, 45, 25))  
 self.label\_9.setObjectName("label\_9")  
 self.label\_10 = QtWidgets.QLabel(Dialog)  
 self.label\_10.setGeometry(QtCore.QRect(270, 380, 45, 25))  
 self.label\_10.setObjectName("label\_10")  
 self.name\_2 = QtWidgets.QLineEdit(Dialog)  
 self.name\_2.setGeometry(QtCore.QRect(110, 340, 113, 25))  
 self.name\_2.setText("")  
 self.name\_2.setObjectName("name\_2")  
 self.label\_11 = QtWidgets.QLabel(Dialog)  
 self.label\_11.setGeometry(QtCore.QRect(70, 340, 45, 25))  
 self.label\_11.setObjectName("label\_11")  
 self.nation\_2 = QtWidgets.QLineEdit(Dialog)  
 self.nation\_2.setGeometry(QtCore.QRect(310, 380, 71, 25))  
 self.nation\_2.setText("")  
 self.nation\_2.setObjectName("nation\_2")  
 self.lineEdit\_4 = QtWidgets.QLineEdit(Dialog)  
 self.lineEdit\_4.setGeometry(QtCore.QRect(220, 280, 113, 25))  
 self.lineEdit\_4.setText("")  
 self.lineEdit\_4.setObjectName("lineEdit\_4")  
 self.label\_13 = QtWidgets.QLabel(Dialog)  
 self.label\_13.setGeometry(QtCore.QRect(180, 280, 45, 25))  
 self.label\_13.setObjectName("label\_13")  
 self.find = QtWidgets.QPushButton(Dialog)  
 self.find.setGeometry(QtCore.QRect(350, 278, 113, 32))  
 font = QtGui.QFont()  
 font.setBold(False)  
 font.setUnderline(False)  
 font.setWeight(50)  
 font.setStrikeOut(False)  
 self.find.setFont(font)  
 self.find.setCursor(QtGui.QCursor(QtCore.Qt.CursorShape.PointingHandCursor))  
 self.find.setObjectName("find")  
 self.label\_14 = QtWidgets.QLabel(Dialog)  
 self.label\_14.setGeometry(QtCore.QRect(30, 310, 91, 21))  
 font = QtGui.QFont()  
 font.setFamily("Herculanum")  
 font.setPointSize(18)  
 font.setItalic(True)  
 self.label\_14.setFont(font)  
 self.label\_14.setObjectName("label\_14")  
 self.label\_15 = QtWidgets.QLabel(Dialog)  
 self.label\_15.setGeometry(QtCore.QRect(30, 70, 91, 21))  
 font = QtGui.QFont()  
 font.setFamily("Herculanum")  
 font.setPointSize(18)  
 font.setItalic(True)  
 self.label\_15.setFont(font)  
 self.label\_15.setObjectName("label\_15")  
 self.label\_18 = QtWidgets.QLabel(Dialog)  
 self.label\_18.setGeometry(QtCore.QRect(210, 20, 254, 44))  
 font = QtGui.QFont()  
 font.setPointSize(31)  
 font.setItalic(False)  
 font.setKerning(True)  
 self.label\_18.setFont(font)  
 self.label\_18.setObjectName("label\_18")  
 self.clear = QtWidgets.QPushButton(Dialog)  
 self.clear.setGeometry(QtCore.QRect(210, 530, 113, 32))  
 self.clear.setObjectName("clear")  
 self.Exit = QtWidgets.QPushButton(Dialog)  
 self.Exit.setGeometry(QtCore.QRect(350, 530, 113, 32))  
 self.Exit.setObjectName("Exit")  
 self.label\_12 = QtWidgets.QLabel(Dialog)  
 self.label\_12.setGeometry(QtCore.QRect(270, 180, 61, 25))  
 self.label\_12.setObjectName("label\_12")  
 self.indexbox = QtWidgets.QLineEdit(Dialog)  
 self.indexbox.setGeometry(QtCore.QRect(340, 180, 41, 25))  
 self.indexbox.setText("")  
 self.indexbox.setObjectName("indexbox")  
 self.graph = QtWidgets.QLabel(Dialog)  
 self.graph.setGeometry(QtCore.QRect(490, 90, 101, 101))  
 self.graph.setText("")  
 self.graph.setObjectName("graph")  
 self.graph\_2 = QtWidgets.QLabel(Dialog)  
 self.graph\_2.setGeometry(QtCore.QRect(480, 340, 101, 101))  
 self.graph\_2.setText("")  
 self.graph\_2.setObjectName("graph\_2")  
 self.birthdayy = QtWidgets.QLineEdit(Dialog)  
 self.birthdayy.setGeometry(QtCore.QRect(110, 380, 113, 25))  
 self.birthdayy.setText("")  
 self.birthdayy.setObjectName("birthdayy")  
 self.sexx = QtWidgets.QLineEdit(Dialog)  
 self.sexx.setGeometry(QtCore.QRect(310, 340, 71, 25))  
 self.sexx.setText("")  
 self.sexx.setObjectName("sexx")  
 self.changebtn = QtWidgets.QPushButton(Dialog)  
 self.changebtn.setGeometry(QtCore.QRect(270, 440, 81, 32))  
 self.changebtn.setMouseTracking(False)  
 self.changebtn.setObjectName("changebtn")  
 self.clear\_find = QtWidgets.QPushButton(Dialog)  
 self.clear\_find.setGeometry(QtCore.QRect(300, 470, 121, 32))  
 self.clear\_find.setMouseTracking(False)  
 self.clear\_find.setObjectName("clear\_find")  
 self.label\_16 = QtWidgets.QLabel(Dialog)  
 self.label\_16.setGeometry(QtCore.QRect(40, 530, 60, 16))  
 self.label\_16.setObjectName("label\_16")  
 self.people = QtWidgets.QTextBrowser(Dialog)  
 self.people.setGeometry(QtCore.QRect(100, 530, 41, 21))  
 self.people.setObjectName("people")  
 self.label\_17 = QtWidgets.QLabel(Dialog)  
 self.label\_17.setGeometry(QtCore.QRect(150, 530, 21, 16))  
 self.label\_17.setObjectName("label\_17")  
  
 self.retranslateUi(Dialog)  
 QtCore.QMetaObject.connectSlotsByName(Dialog)  
  
 def retranslateUi(self, Dialog):  
 \_translate = QtCore.QCoreApplication.translate  
 Dialog.setWindowTitle(\_translate("Dialog", "Dialog"))  
 self.label.setText(\_translate("Dialog", "姓名："))  
 self.label\_2.setText(\_translate("Dialog", "学号："))  
 self.label\_3.setText(\_translate("Dialog", "生日："))  
 self.label\_4.setText(\_translate("Dialog", "兴趣："))  
 self.label\_5.setText(\_translate("Dialog", "性别："))  
 self.sex.setItemText(0, \_translate("Dialog", "男"))  
 self.sex.setItemText(1, \_translate("Dialog", "女"))  
 self.label\_6.setText(\_translate("Dialog", "民族："))  
 self.input.setText(\_translate("Dialog", "录入"))  
 self.putphoto.setText(\_translate("Dialog", "上传照片"))  
 self.delete\_2.setText(\_translate("Dialog", "删除此人"))  
 self.changephoto.setText(\_translate("Dialog", "替换照片"))  
 self.label\_7.setText(\_translate("Dialog", "性别："))  
 self.label\_8.setText(\_translate("Dialog", "生日："))  
 self.label\_9.setText(\_translate("Dialog", "兴趣："))  
 self.label\_10.setText(\_translate("Dialog", "民族："))  
 self.label\_11.setText(\_translate("Dialog", "姓名："))  
 self.label\_13.setText(\_translate("Dialog", "学号："))  
 self.find.setText(\_translate("Dialog", "查询"))  
 self.label\_14.setText(\_translate("Dialog", "查询结果"))  
 self.label\_15.setText(\_translate("Dialog", "录入信息"))  
 self.label\_18.setText(\_translate("Dialog", "好友数据管理系统"))  
 self.clear.setText(\_translate("Dialog", "清空好友"))  
 self.Exit.setText(\_translate("Dialog", "退出程序"))  
 self.label\_12.setText(\_translate("Dialog", "插入位置："))  
 self.changebtn.setText(\_translate("Dialog", "修改信息"))  
 self.clear\_find.setText(\_translate("Dialog", "清空查询结果"))  
 self.label\_16.setText(\_translate("Dialog", "现在共有"))  
 self.people.setHtml(\_translate("Dialog", "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0//EN\" \"http://www.w3.org/TR/REC-html40/strict.dtd\">\n"  
"<html><head><meta name=\"qrichtext\" content=\"1\" /><style type=\"text/css\">\n"  
"p, li { white-space: pre-wrap; }\n"  
"</style></head><body style=\" font-family:\'.AppleSystemUIFont\'; font-size:13pt; font-weight:400; font-style:normal;\">\n"  
"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;\">0</p></body></html>"))  
 self.label\_17.setText(\_translate("Dialog", "人"))  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 import sys  
 app = QtWidgets.QApplication(sys.argv)  
 Dialog = QtWidgets.QDialog()  
 ui = Ui\_Dialog()  
 ui.setupUi(Dialog)  
 Dialog.show()  
 sys.exit(app.exec())



链表中的结点：

class Node(): *#链表中的一个结点* def \_\_init\_\_(self, id=None, name=None, birthday=None, sex=None, photo=None, nation=None, hobby=None, index=None):  
 self.\_id = id  
 self.\_name = name  
 self.\_birthday = birthday  
 self.\_sex = sex  
 self.\_photo = photo  
 self.\_nation = nation  
 self.\_hobby = hobby  
 self.\_index = index  
 self.\_next = None  
 def change\_name(self, name):  
 self.\_name = name  
 def change\_birthday(self, birthday):  
 self.\_birthday = birthday  
 def change\_sex(self, sex):  
 self.\_sex = sex  
 def change\_photo(self, photo):  
 self.\_photo = photo  
 def change\_nation(self, nation):  
 self.\_nation = nation  
 def change\_hobby(self, hobby):  
 self.\_hobby = hobby

节点包含了一些属性，如id（学号）、name（姓名）、birthday（生日）、sex（性别）、photo（照片）、nation（民族）、hobby（兴趣）和index（插入的位置），其中每个属性都有一个默认值为None。节点还包含一个指向下一个节点的指针，该指针的默认值为None。

该类还包含了一些用于更改节点属性的方法，例如change\_name、change\_birthday、change\_sex等等。这些方法接受一个参数，并将节点相应的属性更改为该参数的值。这些方法允许在程序运行时动态地更改节点属性，从而实现对链表的修改，这在修改信息中用到。

链表类，初始化了链表，且定义了一些方法。

class linklist():  
 def \_\_init\_\_(self):  
 node = Node()  
 self.head = node  
  
 def is\_empty(self):  
 return self.head.\_next == None  
  
 def get\_length(self):  
 if self.is\_empty() == False:  
 cnt =0  
 h = self.head.\_next  
 while h != None:  
 cnt += 1  
 h = h.\_next  
 return cnt  
 else:  
 return 0  
  
 def insert(self, index, item):  
 pre = self.head  
 p = self.head.\_next  
 index -=1  
 while index:  
 p = p.\_next  
 pre = pre.\_next  
 index -= 1  
 pre.\_next = item  
 item.\_next = p  
  
 def remove(self, index):  
 pre = self.head  
 p = self.head.\_next  
 index -= 1  
 while index:  
 p = p.\_next  
 pre = pre.\_next  
 index-=1  
 pre.\_next = p.\_next  
 del p  
  
 def find(self, iid):  
 p = self.head  
 while p != None:  
 if p.\_id == iid:  
 return p  
 else:  
 p = p.\_next  
 return 0  
  
 def clear(self):  
 self.head = None

这段代码定义了一个链表类 linklist，包括以下方法：

\_\_init\_\_(self)：初始化链表，创建一个空的头节点，将其赋值给 self.head。

is\_empty(self)：判断链表是否为空，如果头节点的下一个节点为空，返回 True，否则返回 False。

get\_length(self)：获取链表长度，遍历链表计数，返回链表长度。

insert(self, index, item)：在指定位置插入元素， index 表示插入位置， item 表示要插入的元素。

remove(self, index)：删除指定位置的元素， index 表示要删除的位置。

find(self, iid)：查找指定元素， iid 表示要查找的元素的 id。

clear(self)：清空链表，将头节点置为 None。

将链表类实例化。

friends\_link = linklist()

定义MyLink类

class MyLink(QMainWindow, u):  
 signal = pyqtSignal(str)  
 def \_\_init\_\_(self, parent = None):  
 super(MyLink, self).\_\_init\_\_(parent)  
 self.pic\_path = ''  
 self.setupUi(self)  
 self.connecter()  
  
 def connecter(self): *#把函数与对应按钮相连接* self.putphoto.clicked.connect(self.put\_photo)  
 self.input.clicked.connect(self.insert\_item)  
 self.find.clicked.connect(self.find\_item)  
 self.delete\_2.clicked.connect(self.del\_item)  
 self.clear.clicked.connect(self.clear\_items)  
 self.changebtn.clicked.connect(self.change\_item)  
 self.changephoto.clicked.connect(self.change\_photo)  
 self.clear\_find.clicked.connect(self.clear\_finditem)  
 self.Exit.clicked.connect(self.click\_button)  
  
  
 def put\_photo(self): *#上传图片* file, filetype = QFileDialog.getOpenFileName(self,"打开图片", "",  
 "All Pixmaps(\*.bmp;\*.cur;\*.gif;\*.icns;\*.ico;\*.jpeg;\*.jpg;\*.pbm;\*.pgm;\*.png;\*.ppm;\*.svg;\*.svgz;\*.tga;\*.tif;\*.tiff;\*. wbmp;\*.webp;\*.xbm;\*.xpm)")  
 img = QtGui.QPixmap(file).scaled(self.graph.width(), self.graph.height(), Qt.AspectRatioMode.KeepAspectRatio)  
 self.graph.setPixmap(img)  
 self.pic\_path = file *#打开图片* def change\_photo(self): *#修改图片* file, filetype = QFileDialog.getOpenFileName(self, "打开图片", "",  
 "All Pixmaps(\*.bmp;\*.cur;\*.gif;\*.icns;\*.ico;\*.jpeg;\*.jpg;\*.pbm;\*.pgm;\*.png;\*.ppm;\*.svg;\*.svgz;\*.tga;\*.tif;\*.tiff;\*. wbmp;\*.webp;\*.xbm;\*.xpm)")  
 img = QtGui.QPixmap(file).scaled(self.graph.width(), self.graph.height(), Qt.AspectRatioMode.KeepAspectRatio)  
 self.graph\_2.setPixmap(img)  
 self.pic\_path = file *# 打开图片* def insert\_item(self): *#插入信息* if self.namebox.text() == '' or self.idbox.text() =='' or self.hobbybox.text() =='' or self.nationbox.text()=='':  
 QMessageBox.about(self, "提示", "信息未填写完整！")  
 return  
 if self.indexbox.text().isdigit():  
 index = int(self.indexbox.text())  
 else:  
 QMessageBox.about(self, "提示", "插入位置请填写数字！")  
 return  
 name = self.namebox.text()  
 id = self.idbox.text()  
 birthday = self.birthdaybox.text()  
 hobby = self.hobbybox.text()  
 nation = self.nationbox.text()  
 sex = self.sex.currentText()  
  
 if index <= 0 or index > friends\_link.get\_length() + 1:  
 QMessageBox.about(self, "Warning", "插入元素下标越界！")  
 return  
  
 node = Node(id, name, birthday, sex, self.pic\_path, nation, hobby, index)  
 friends\_link.insert(index, node)  
 self.people.setText(str(friends\_link.get\_length()))  
 QMessageBox.about(self, "提示", "恭喜你！添加成功！")  
 self.pic\_path = ''  
 self.namebox.clear()  
 self.idbox.clear()  
 self.birthdaybox.clear()  
 self.nationbox.clear()  
 self.hobbybox.clear()  
 self.graph.clear()  
 self.indexbox.clear()  
  
 def find\_item(self): *#查找信息* iid = self.lineEdit\_4.text()  
 node = friends\_link.find(iid)  
 if node == 0:  
 QMessageBox.warning(self, "警告", "查无此人！")  
 self.lineEdit\_4.clear()  
 else:  
 self.name\_2.setText(node.\_name)  
 self.birthdayy.setText(node.\_birthday)  
 self.hobby\_2.setText(node.\_hobby)  
 self.nation\_2.setText(node.\_nation)  
 self.sexx.setText(node.\_sex)  
 img = QtGui.QPixmap(node.\_photo).scaled(self.graph.width(), self.graph.height(), Qt.AspectRatioMode.KeepAspectRatio)  
 self.graph\_2.setPixmap(img)  
  
 def del\_item(self): *#删除信息* iid = self.lineEdit\_4.text()  
 node = friends\_link.find(iid)  
 friends\_link.remove(node.\_index)  
 self.people.setText(str(friends\_link.get\_length()))  
 QMessageBox.about(self, "提示", "删除成功")  
  
 def clear\_items(self): *#清空链表* friends\_link.clear()  
 self.people.setText('0')  
 QMessageBox.about(self, "提示", "已成功清空！")  
  
 def change\_item(self): *#修改信息* iid = self.lineEdit\_4.text()  
 node = friends\_link.find(iid)  
 if self.name\_2.text() == '' or self.hobby\_2.text() =='' or self.nation\_2.text()=='' or self.sexx.text == '' or self.birthdayy.text =='':  
 QMessageBox.about(self, "提示", "信息未填写完整！")  
 return  
 if self.sexx.text() not in ['男', '女']:  
 QMessageBox.about(self, "提示", "请在性别一栏填入\"男\"或\"女\"")  
 return  
 name = self.name\_2.text()  
 node.change\_name(name)  
 birthday = self.birthdayy.text()  
 node.change\_birthday(birthday)  
 hobby = self.hobby\_2.text()  
 node.change\_hobby(hobby)  
 nation = self.nation\_2.text()  
 node.change\_nation(nation)  
 sex = self.sexx.text()  
 node.change\_sex(sex)  
 node.change\_photo(self.pic\_path)  
 QMessageBox.about(self, "提示", "恭喜你！修改成功！")  
  
 def clear\_finditem(self): *#清空查询的信息* self.pic\_path = ''  
 self.name\_2.clear()  
 self.birthdayy.clear()  
 self.nation\_2.clear()  
 self.hobby\_2.clear()  
 self.graph\_2.clear()  
 self.sexx.clear()  
  
  
 def click\_button(self):  
 confirm\_exit = QMessageBox.question(  
 self, "提示", "你确定要退出吗😭",  
 QMessageBox.StandardButton.Yes | QMessageBox.StandardButton.No, QMessageBox.StandardButton.No)  
 if confirm\_exit == QMessageBox.StandardButton.Yes:  
 QApplication.quit()

main函数

if \_\_name\_\_ == "\_\_main\_\_":  
 app = QApplication(sys.argv)  
 win = MyLink()  
 win.show()  
 sys.exit(app.exec())

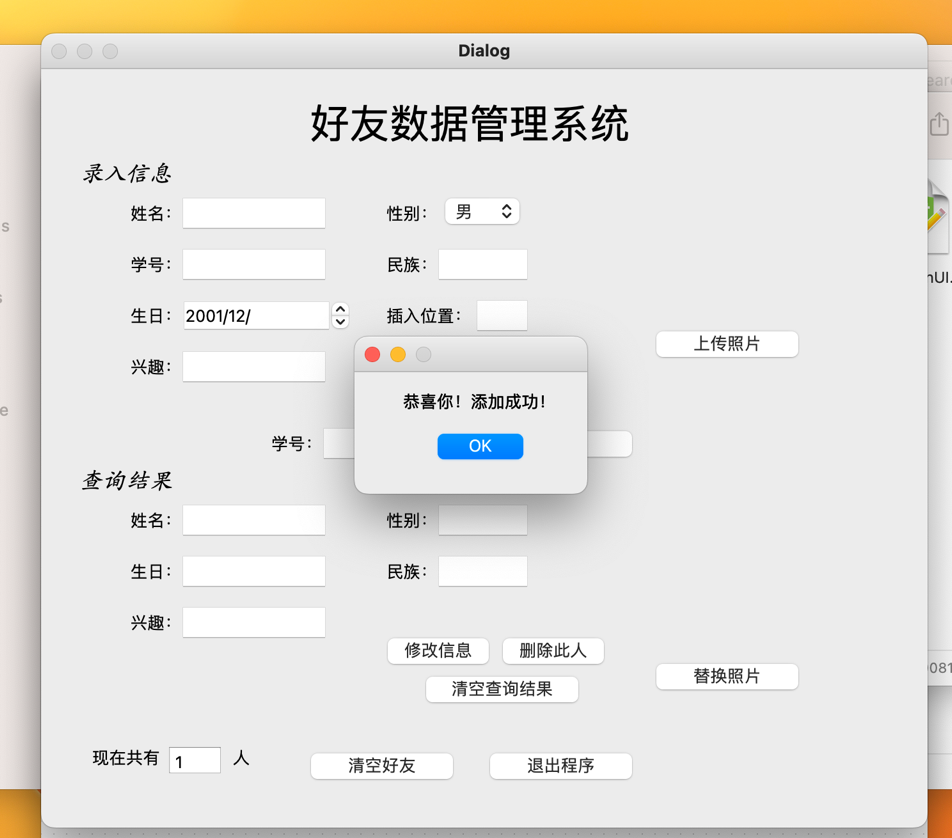
完整代码：

import sys  
from PyQt6.QtCore import Qt  
from PyQt6 import QtCore, QtGui, QtWidgets  
  
from PyQt6.QtCore import pyqtSignal, Qt  
from PyQt6.QtGui import QImage, QPixmap  
from PyQt6.QtWidgets import QMainWindow, QFileDialog, QMessageBox, QApplication, QGraphicsScene, QGraphicsPixmapItem  
from PyQt6.QtWidgets import QPushButton  
  
from link import Ui\_Dialog as u  
  
class Node(): *#链表中的一个结点* def \_\_init\_\_(self, id=None, name=None, birthday=None, sex=None, photo=None, nation=None, hobby=None, index=None):  
 self.\_id = id  
 self.\_name = name  
 self.\_birthday = birthday  
 self.\_sex = sex  
 self.\_photo = photo  
 self.\_nation = nation  
 self.\_hobby = hobby  
 self.\_index = index  
 self.\_next = None  
 def change\_name(self, name):  
 self.\_name = name  
 def change\_birthday(self, birthday):  
 self.\_birthday = birthday  
 def change\_sex(self, sex):  
 self.\_sex = sex  
 def change\_photo(self, photo):  
 self.\_photo = photo  
 def change\_nation(self, nation):  
 self.\_nation = nation  
 def change\_hobby(self, hobby):  
 self.\_hobby = hobby  
  
class linklist():  
 def \_\_init\_\_(self):  
 node = Node()  
 self.head = node  
  
 def is\_empty(self):  
 return self.head.\_next == None  
  
 def get\_length(self):  
 if self.is\_empty() == False:  
 cnt =0  
 h = self.head.\_next  
 while h != None:  
 cnt += 1  
 h = h.\_next  
 return cnt  
 else:  
 return 0  
  
 def insert(self, index, item):  
 pre = self.head  
 p = self.head.\_next  
 index -=1  
 while index:  
 p = p.\_next  
 pre = pre.\_next  
 index -= 1  
 pre.\_next = item  
 item.\_next = p  
  
 def remove(self, index):  
 pre = self.head  
 p = self.head.\_next  
 index -= 1  
 while index:  
 p = p.\_next  
 pre = pre.\_next  
 index-=1  
 pre.\_next = p.\_next  
 del p  
  
 def find(self, iid):  
 p = self.head  
 while p != None:  
 if p.\_id == iid:  
 return p  
 else:  
 p = p.\_next  
 return 0  
  
 def clear(self):  
 self.head = None  
  
friends\_link = linklist()  
  
class MyLink(QMainWindow, u):  
 signal = pyqtSignal(str)  
 def \_\_init\_\_(self, parent = None):  
 super(MyLink, self).\_\_init\_\_(parent)  
 self.pic\_path = ''  
 self.setupUi(self)  
 self.connecter()  
  
 def connecter(self): *#把函数与对应按钮相连接* self.putphoto.clicked.connect(self.put\_photo)  
 self.input.clicked.connect(self.insert\_item)  
 self.find.clicked.connect(self.find\_item)  
 self.delete\_2.clicked.connect(self.del\_item)  
 self.clear.clicked.connect(self.clear\_items)  
 self.changebtn.clicked.connect(self.change\_item)  
 self.changephoto.clicked.connect(self.change\_photo)  
 self.clear\_find.clicked.connect(self.clear\_finditem)  
 self.Exit.clicked.connect(self.click\_button)  
  
  
 def put\_photo(self): *#上传图片* file, filetype = QFileDialog.getOpenFileName(self,"打开图片", "",  
 "All Pixmaps(\*.bmp;\*.cur;\*.gif;\*.icns;\*.ico;\*.jpeg;\*.jpg;\*.pbm;\*.pgm;\*.png;\*.ppm;\*.svg;\*.svgz;\*.tga;\*.tif;\*.tiff;\*. wbmp;\*.webp;\*.xbm;\*.xpm)")  
 img = QtGui.QPixmap(file).scaled(self.graph.width(), self.graph.height(), Qt.AspectRatioMode.KeepAspectRatio)  
 self.graph.setPixmap(img)  
 self.pic\_path = file *#打开图片* def change\_photo(self): *#修改图片* file, filetype = QFileDialog.getOpenFileName(self, "打开图片", "",  
 "All Pixmaps(\*.bmp;\*.cur;\*.gif;\*.icns;\*.ico;\*.jpeg;\*.jpg;\*.pbm;\*.pgm;\*.png;\*.ppm;\*.svg;\*.svgz;\*.tga;\*.tif;\*.tiff;\*. wbmp;\*.webp;\*.xbm;\*.xpm)")  
 img = QtGui.QPixmap(file).scaled(self.graph.width(), self.graph.height(), Qt.AspectRatioMode.KeepAspectRatio)  
 self.graph\_2.setPixmap(img)  
 self.pic\_path = file *# 打开图片* def insert\_item(self): *#插入信息* if self.namebox.text() == '' or self.idbox.text() =='' or self.hobbybox.text() =='' or self.nationbox.text()=='':  
 QMessageBox.about(self, "提示", "信息未填写完整！")  
 return  
 if self.indexbox.text().isdigit():  
 index = int(self.indexbox.text())  
 else:  
 QMessageBox.about(self, "提示", "插入位置请填写数字！")  
 return  
 name = self.namebox.text()  
 id = self.idbox.text()  
 birthday = self.birthdaybox.text()  
 hobby = self.hobbybox.text()  
 nation = self.nationbox.text()  
 sex = self.sex.currentText()  
  
 if index <= 0 or index > friends\_link.get\_length() + 1:  
 QMessageBox.about(self, "Warning", "插入元素下标越界！")  
 return  
  
 node = Node(id, name, birthday, sex, self.pic\_path, nation, hobby, index)  
 friends\_link.insert(index, node)  
 self.people.setText(str(friends\_link.get\_length()))  
 QMessageBox.about(self, "提示", "恭喜你！添加成功！")  
 self.pic\_path = ''  
 self.namebox.clear()  
 self.idbox.clear()  
 self.birthdaybox.clear()  
 self.nationbox.clear()  
 self.hobbybox.clear()  
 self.graph.clear()  
 self.indexbox.clear()  
  
 def find\_item(self): *#查找信息* iid = self.lineEdit\_4.text()  
 node = friends\_link.find(iid)  
 if node == 0:  
 QMessageBox.warning(self, "警告", "查无此人！")  
 self.lineEdit\_4.clear()  
 else:  
 self.name\_2.setText(node.\_name)  
 self.birthdayy.setText(node.\_birthday)  
 self.hobby\_2.setText(node.\_hobby)  
 self.nation\_2.setText(node.\_nation)  
 self.sexx.setText(node.\_sex)  
 img = QtGui.QPixmap(node.\_photo).scaled(self.graph.width(), self.graph.height(), Qt.AspectRatioMode.KeepAspectRatio)  
 self.graph\_2.setPixmap(img)  
  
 def del\_item(self): *#删除信息* iid = self.lineEdit\_4.text()  
 node = friends\_link.find(iid)  
 friends\_link.remove(node.\_index)  
 self.people.setText(str(friends\_link.get\_length()))  
 QMessageBox.about(self, "提示", "删除成功")  
  
 def clear\_items(self): *#清空链表* friends\_link.clear()  
 self.people.setText('0')  
 QMessageBox.about(self, "提示", "已成功清空！")  
  
 def change\_item(self): *#修改信息* iid = self.lineEdit\_4.text()  
 node = friends\_link.find(iid)  
 if self.name\_2.text() == '' or self.hobby\_2.text() =='' or self.nation\_2.text()=='' or self.sexx.text == '' or self.birthdayy.text =='':  
 QMessageBox.about(self, "提示", "信息未填写完整！")  
 return  
 if self.sexx.text() not in ['男', '女']:  
 QMessageBox.about(self, "提示", "请在性别一栏填入\"男\"或\"女\"")  
 return  
 name = self.name\_2.text()  
 node.change\_name(name)  
 birthday = self.birthdayy.text()  
 node.change\_birthday(birthday)  
 hobby = self.hobby\_2.text()  
 node.change\_hobby(hobby)  
 nation = self.nation\_2.text()  
 node.change\_nation(nation)  
 sex = self.sexx.text()  
 node.change\_sex(sex)  
 node.change\_photo(self.pic\_path)  
 QMessageBox.about(self, "提示", "恭喜你！修改成功！")  
  
 def clear\_finditem(self): *#清空查询的信息* self.pic\_path = ''  
 self.name\_2.clear()  
 self.birthdayy.clear()  
 self.nation\_2.clear()  
 self.hobby\_2.clear()  
 self.graph\_2.clear()  
 self.sexx.clear()  
  
  
 def click\_button(self):  
 confirm\_exit = QMessageBox.question(  
 self, "提示", "你确定要退出吗😭",  
 QMessageBox.StandardButton.Yes | QMessageBox.StandardButton.No, QMessageBox.StandardButton.No)  
 if confirm\_exit == QMessageBox.StandardButton.Yes:  
 QApplication.quit()  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app = QApplication(sys.argv)  
 win = MyLink()  
 win.show()  
 sys.exit(app.exec())

**七、测试结果及分析**

**软件主页面**

**添加信息**



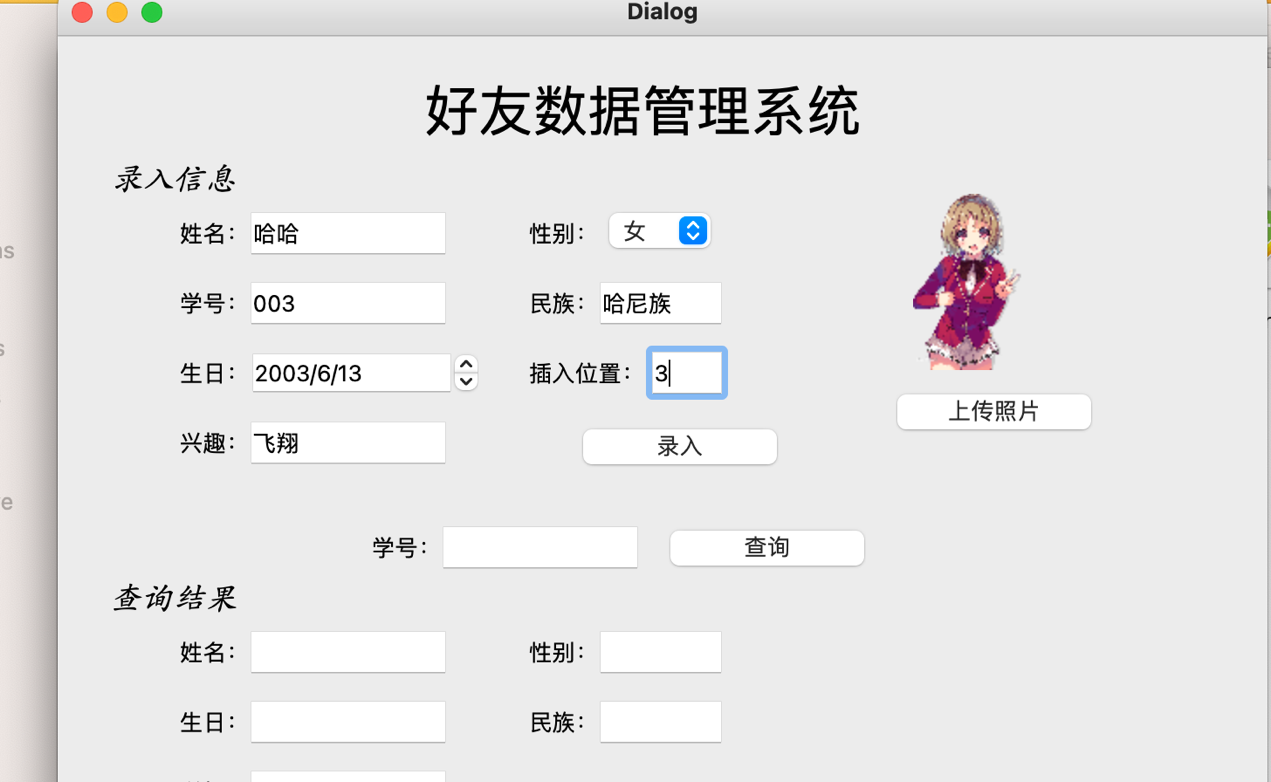
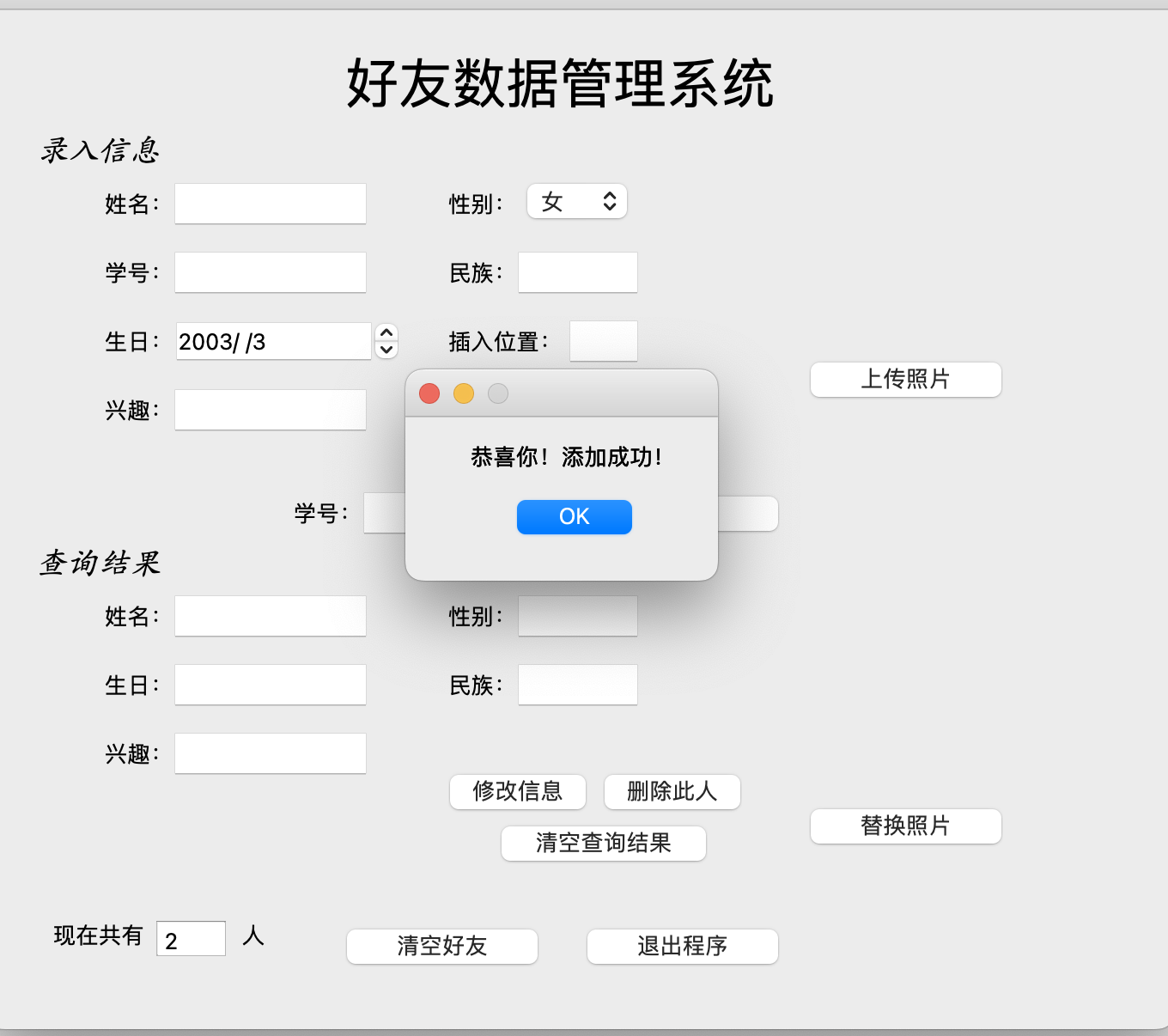
若信息不完整，跳出提示：



添加后页面清空：









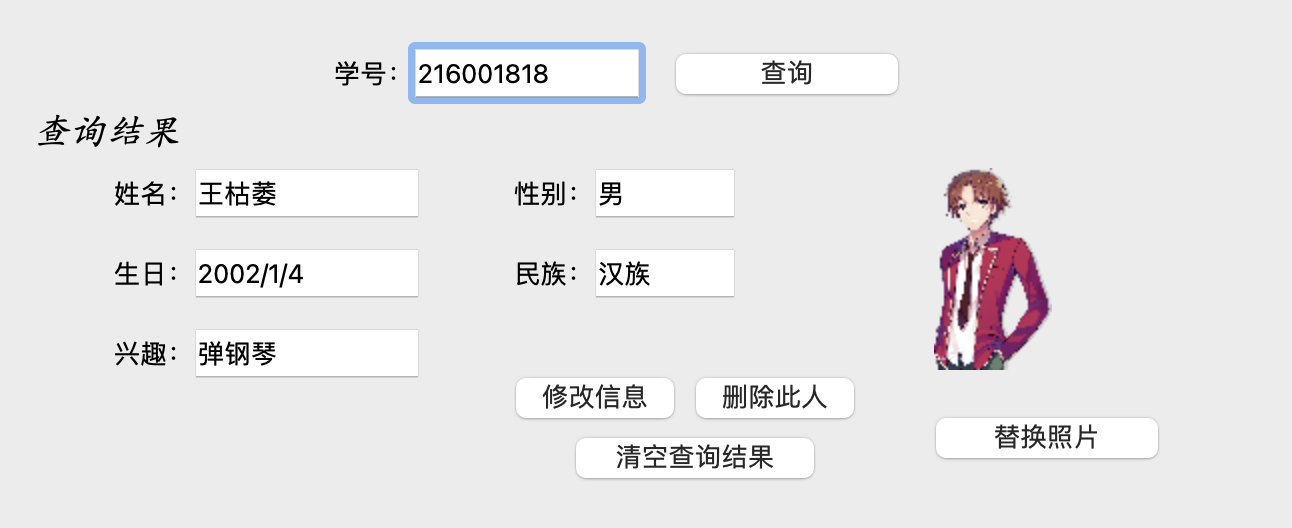
如果插入位置不对，也会有提示：





**查询操作**

输入学号后，在查询结果页面显示查询结果：

****

****

**修改信息操作**

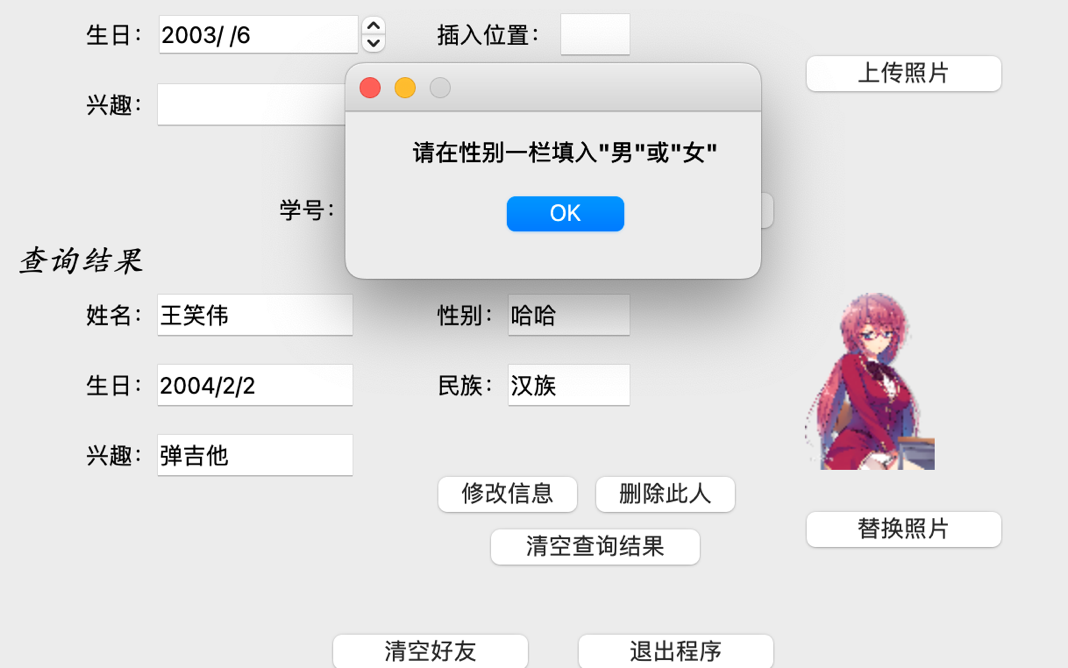
直接在查询结果处修改，修改完后点击“修改信息”。

****

也可以替换照片

****

如果在性别栏填了其它字，会跳出提示：

****

点击清空查询结果，查询结果被清空：

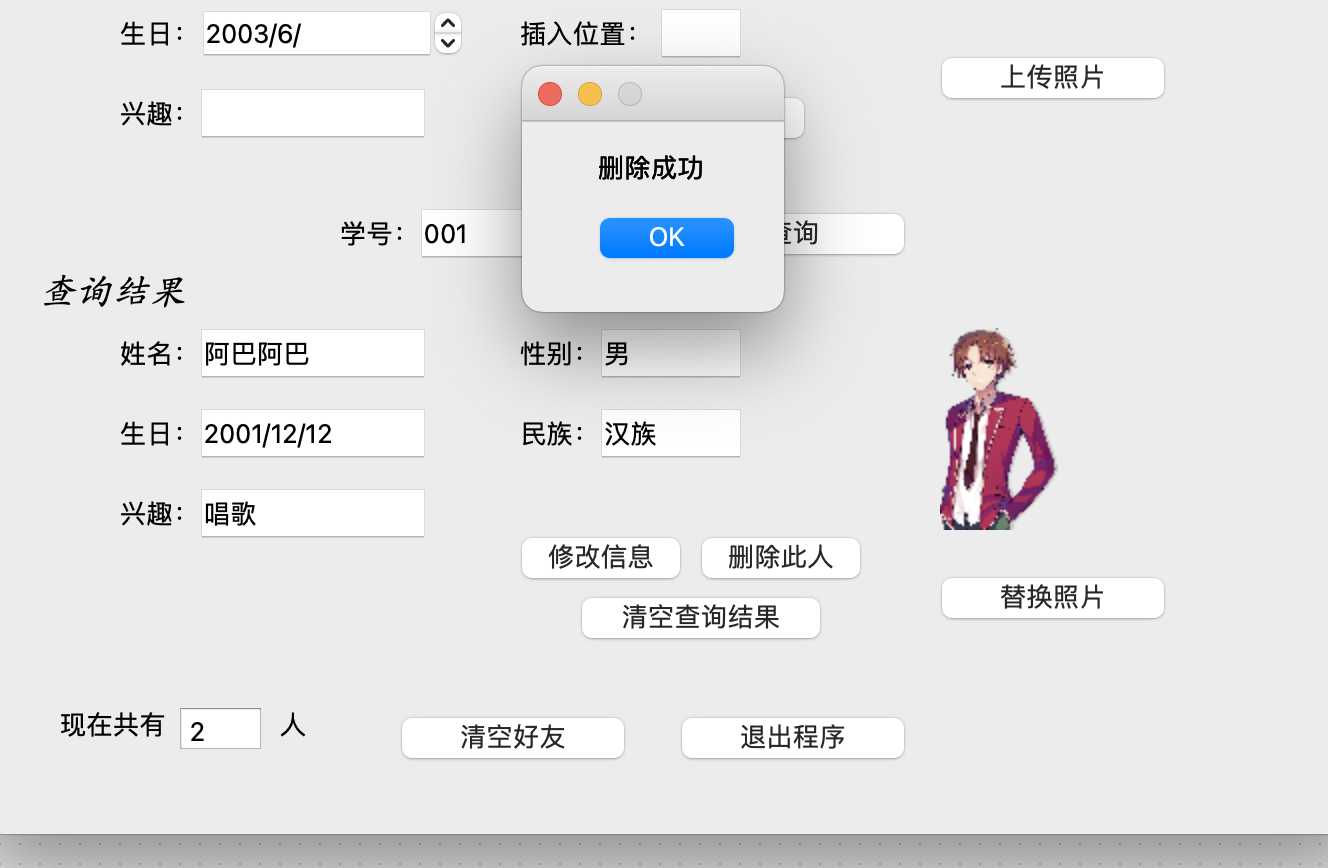
****

再次输入学号216001818，结果已经改变了



**删除好友**

点击删除此人

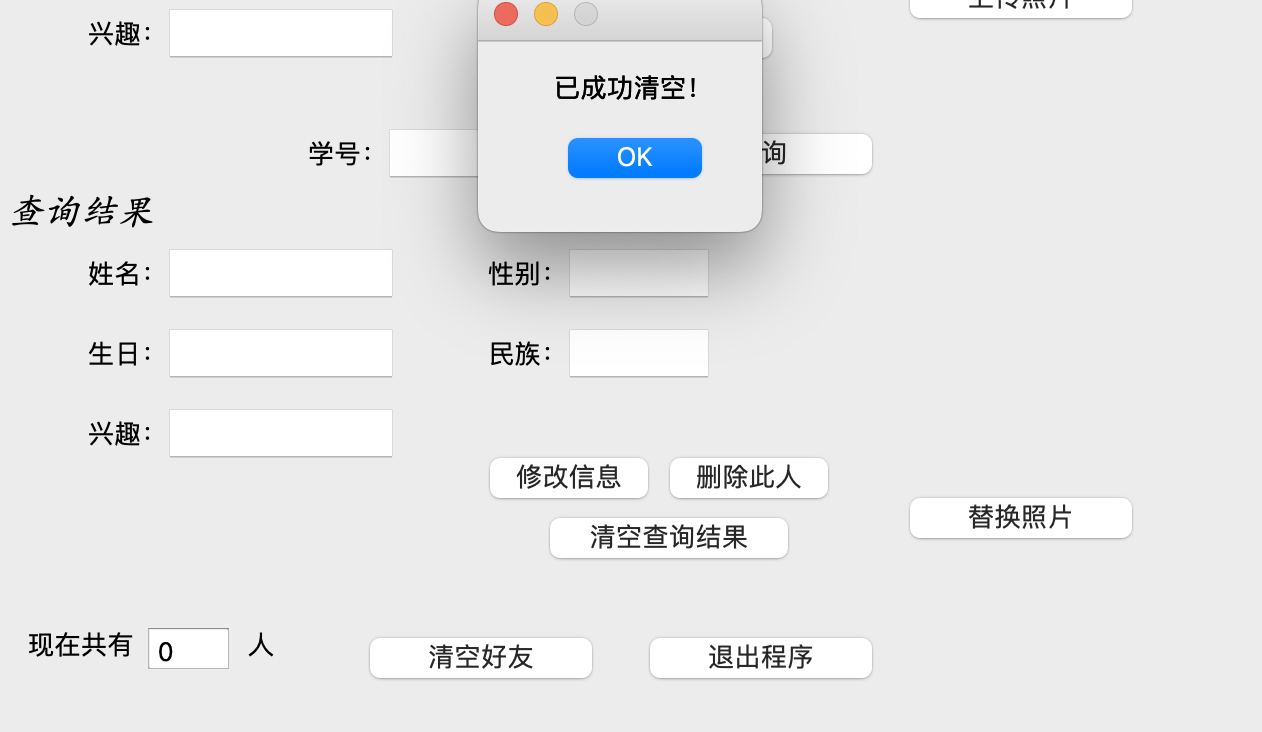


再次查询，跳出“查无此人”



清空好友：

点击清空后，就没有好友了。

****

退出这个界面：



点击yes即可退出。

**八、总结**