## 中国矿业大学计算机学院

**16 级本科生课程设计报告**

课程名称 程序设计综合实践

报告时间 2018.01.18

学生姓名 xxxxxxx

学 号 xxxxxxxxxxx

专 业 计算机科学与技术

任课教师 xxxxxxx

## 任课教师评阅

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 编号 | 课程教学目标 | | | 占比 | | 得分 |
| 1 | **目标1：**掌握一门计算机高级语言，并能使用特定的软件开发工具，设计、开发、调试及运行应用程序。 | | | 20% | |  |
| 2 | **目标2：**针对具体的应用问题，进行功能需求分析，确定设计目标，并能绘制算法流程图。 | | | 20% | |  |
| 3 | **目标3：**在进行需求分析的基础上，设计软件运行界面、关键类、编写代码，调试并正确运行满足需求的应用程序。 | | | 60% | |  |
| 总成绩 | | | | | |  |
| 指导教师 | |  | 评阅日期 | |  | |

## 目 录

中国矿业大学计算机学院

任课教师评阅

实验一 简单计算器

1 系统概述

2 系统设计

设计目标

设计分析与算法流程

界面设计

关键类图

3系统实现（运行调试）

4系统扩展

[5 总结](#_Toc503794892)

实验二 多文档文本编辑器

1 系统概述

2 系统设计

设计目标

设计分析与算法流程

界面设计

关键类图

3系统实现（运行调试）

4系统扩展

[5 总结](#_Toc503794892)

实验三 学生通讯录

1 系统概述

2 系统设计

设计目标

设计分析与算法流程

界面设计

关键类图

3系统实现（运行调试）

4系统扩展

[5 总结](#_Toc503794892)

实验四 拼图游戏

1 系统概述

2 系统设计

设计目标

设计分析与算法流程

界面设计

关键类图

3系统实现（运行调试）

4系统扩展

[5 总结](#_Toc503794892)

## 任课教师评阅

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 编号 | 课程教学目标 | | | 占比 | | 得分 |
| 1 | **目标1：**掌握一门计算机高级语言，并能使用特定的软件开发工具，设计、开发、调试及运行应用程序。 | | | 20% | |  |
| 2 | **目标2：**针对具体的应用问题，进行功能需求分析，确定设计目标，并能绘制算法流程图。 | | | 20% | |  |
| 3 | **目标3：**在进行需求分析的基础上，设计软件运行界面、关键类、编写代码，调试并正确运行满足需求的应用程序。 | | | 60% | |  |
| 总成绩 | | | | | |  |
| 指导教师 | |  | 评阅日期 | |  | |

## 目 录

中国矿业大学计算机学院

任课教师评阅

实验一 简单计算器

1 系统概述

2 系统设计

设计目标

设计分析与算法流程

界面设计

关键类图

3系统实现（运行调试）

4系统扩展

[5 总结](#_Toc503794892)

实验二 多文档文本编辑器

1 系统概述

2 系统设计

设计目标

设计分析与算法流程

界面设计

关键类图

3系统实现（运行调试）

4系统扩展

[5 总结](#_Toc503794892)

实验三 学生通讯录

1 系统概述

2 系统设计

设计目标

设计分析与算法流程

界面设计

关键类图

3系统实现（运行调试）

4系统扩展

[5 总结](#_Toc503794892)

实验四 拼图游戏

1 系统概述

2 系统设计

设计目标

设计分析与算法流程

界面设计

关键类图

3系统实现（运行调试）

4系统扩展

[5 总结](#_Toc503794892)

**实验一 简单计算器**

1 系统概述

该系统是由Python,Tkinter模块编写的支持四则运算连续计算器，拓展了1/x，%，sqrt等功能，并且加入了基本养老保险金的计算拓展

2 系统设计

2.1 设计目标

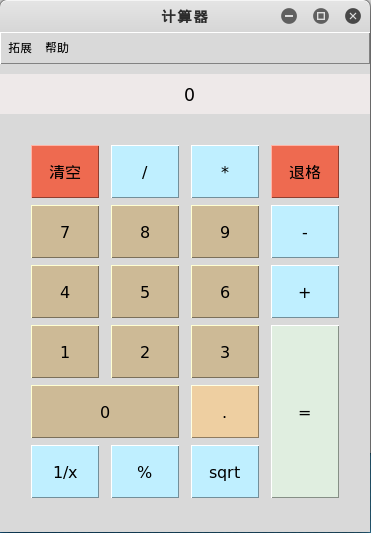
该项目目标是设计开发一个支持连续计算的四则运算计算器，通过单击按钮，输入并完成如4+5+6或5\*8+16类似的连续计算，并将运算结果显示在输出文本框中，同时也具备清空、后退等功能。

2.2 设计分析与算法流程

计算器的主要功能是完成加、减、乘、除四则运算，而且支持连续计算，如4+5+7-8或4\*5-6等运算。连续计算的实现借助于Python中列表的定义实现数据结构Stack（栈），并通过正则等手段限制输入以正常运行

增加的特殊符号则是用检测输入标志，来区分执行功能

2.3 界面设计



2.4 关键类图

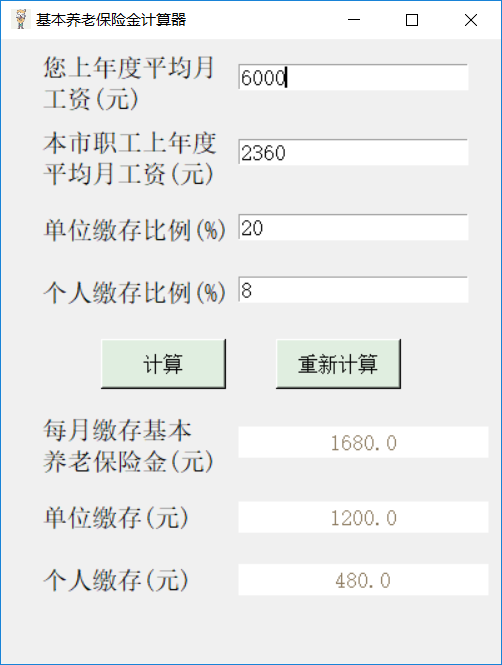
*#按钮事件处理***def** knobDown(self,button):  
 self.inputCheck(button)  
 *#inputlist为空时，检查输入的第一位是否为数字* **if** self.inputlist==[] **and** re.findall(**r'[-+\*/=.%]'**,str(button)):  
 self.count.set(**'符号不能放在第一位哦~'**)  
 *#如果输入算符* **elif** re.findall(**r'[-+\*/]'**,str(button)):  
 *#判断inputlist里面是否有算符* **if** self.signCheck(**'-'**) **or** self.signCheck(**'+'**) \  
 **or** self.signCheck(**'/'**) **or** self.signCheck(**'\*'**):  
 **if** re.findall(**r'[-+\*/]'**,str(self.inputlist[-1])):  
 **if** button==**'1/x' or** button==**'%'**:  
 **pass  
 else**:  
 self.count.set(**'不能连续输入算符'**)  
 **else**:  
 self.addButton(button)  
 **else**:  
 self.addButton(button)  
 *#输入等号时转化为str利用eval函数计算* **elif** button==**'='**:  
 **if** re.findall(**r'[-+\*/%]'**,str(self.inputlist[-1])):  
 self.count.set(**'结尾不能是算符哦~'**)  
 **else**:  
 **for** length **in** range(0,len(self.inputlist)):  
 **if** str(self.inputlist[length]).isdigit():  
 self.midstr+=str(self.inputlist[length])  
 **else**:  
 self.midstr=self.midstr+self.inputlist[length]  
 self.inputCheck(self.midstr) *#eval函数很危险要严格过滤* self.count.set(self.midstr+**'='**+str(round(eval(self.midstr),5)))  
 self.inputlist=[]  
 self.midstr=**''  
 else**:  
 self.addButton(button)

*#添加button***def** addButton(self,button):  
 **if** button==self.ButtonList[18]:  
 listSum,locate=self.checkList()  
 **if** locate==-1:  
 self.inputlist=[str(round(eval(**'1/'**+str(listSum)),5))]  
 **else**:  
 **for** k **in** range(locate+1,len(self.inputlist)):  
 **del** self.inputlist[k]  
 self.inputlist.append(str(round(eval(**'1/'**+str(listSum)),5)))  
 **elif** button==self.ButtonList[19]:  
 listSum,locate=self.checkList()  
 **if** locate==-1:  
 self.inputlist=[str(listSum\*0.01)]  
 **else**:  
 **for** k **in** range(locate+1,len(self.inputlist)):  
 **del** self.inputlist[k]  
 self.inputlist.append(str(listSum\*0.01))  
 **elif** button==self.ButtonList[20]:  
 listSum,locate=self.checkList()  
 **if** locate==-1:  
 self.inputlist=[str(round(listSum\*\*0.5,5))]  
 **else**:  
 **for** k **in** range(locate+1,len(self.inputlist)):  
 **del** self.inputlist[k]  
 self.inputlist.append(str(round(listSum\*\*0.5,5)))  
 **else**:  
 self.inputlist.append(button)  
 self.count.set(self.inputlist)  
*#检查输入***def** inputCheck(self,input):  
 **if** re.findall(**r'[&a-zA-Z<>,?~!@#$";:]'**,str(input)):  
 **if** input==**'1/x' or** input==**'%' or** input==**'sqrt'**:  
 **pass  
 else**:  
 self.count.set(**'非法'**)

3 系统实现（运行调试）



4 系统扩展



5 总结

通过这次计算器实验的学习，让我对栈的使用以及各种功能的设计有了更加深刻地认识，也更加熟练的运用了Python列表的使用

**实验二 多文档编辑器**

1 系统概述

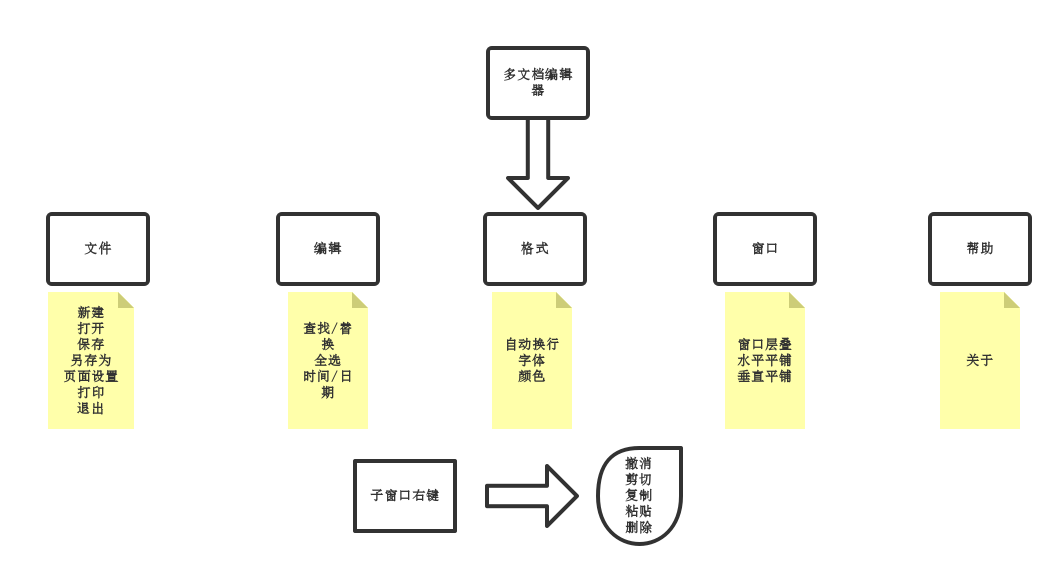
该系统是由C#编写的仿照Windnows记事本的并且实现多文档格式的编辑器，几乎拥有记事本的所有功能，包括剪切复制，另存为，打印，时间日期，查找替换，页面设置等，以及窗口的各种显示方式

2 系统设计

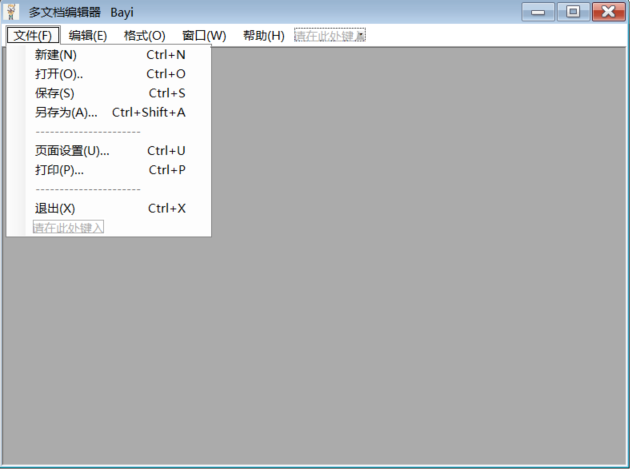
2.1 设计目标

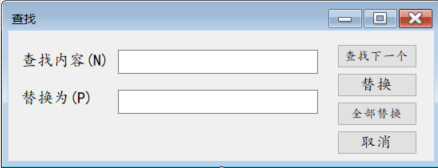
该项目目标是设计开发一个支持多文档并存的编辑器，通过新建按钮，可以连续输入多个新文档窗口，并能进行保存和打开操作，同时也具备字体，颜色修改等拓展功能。

2.2 设计分析与算法流程

编辑器的主要功能是完成多文档的编辑和保存打开操作，而且支持一些字体颜色等的拓展，通过设计主窗口，子窗体，查找子窗口生成相应的UI代码，并进行按钮对应的函数的编写，以完成相应的事件响应和操作，主要是通过richTextBox附件的封装函数，进行一系列功能的实现，具体设计如图

2.3 界面设计







2.4 关键类图

private void 打开OCtrlOToolStripMenuItem\_Click(object sender, EventArgs e)

{

try

{

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

RichTextBoxStreamType fileType = TrunFileType(openFileDialog.FilterIndex);

fileCount++;

file = new FormChild(fileType, openFileDialog.FileName, openFileDialog.FilterIndex);

file.MdiParent = this;

file.Show();

listFormChild.Add(file);

}

}

catch (Exception ex)

{

return;

}

string str = openFileDialog.FileName;

string[] sArray = str.Split('\\');

file.Text = sArray[sArray.Length - 1];

file.WindowState = FormWindowState.Maximized;

}

private void 新建NCtrlNToolStripMenuItem\_Click(object sender, EventArgs e)

{

file = new FormChild();//新建一个子窗体

file.MdiParent = this; //定义此窗体的父窗体，使之成为MDI窗体

++ChildFormCount;

file.Text = "子窗体" + ChildFormCount.ToString();

file.Show();

}

private void buttonFindNext\_Click(object sender, EventArgs e)

{

string str = Convert.ToString(textBoxFind.Text);

try

{

if (file.Sourse.Text.Contains(str))

{

index = file.Sourse.Find(str, index, RichTextBoxFinds.None);

file.Sourse.Select(index, str.Length);

file.Sourse.Focus();

index++;

}

}

catch (Exception ex)

{

if (index == -1)

{

index = 0;

}

}

}

private void buttonReplace\_Click(object sender, EventArgs e)

{

string strOld = Convert.ToString(textBoxFind.Text);

string strNew = Convert.ToString(textBoxReplace.Text);

string str = Convert.ToString(textBoxFind.Text);

try

{

if (file.Sourse.Text.Contains(strOld))

{

Clipboard.Clear();

count = file.Sourse.Find(strOld, index, RichTextBoxFinds.None);

file.Sourse.Select(count, str.Length);

Clipboard.SetText(file.Sourse.SelectedText.Replace(strOld, strNew));

file.Sourse.Paste();

count++;

}

else

return;

}

catch (Exception ex)

{

return;

}

}

private void 另存为ToolStripMenuItem\_Click(object sender, EventArgs e)

{

FormChild df = (FormChild)this.ActiveMdiChild;

try

{

SaveFile(df);

filePath = df.GetFilePath();

MessageBox.Show("保存成功", "温馨提示");

}

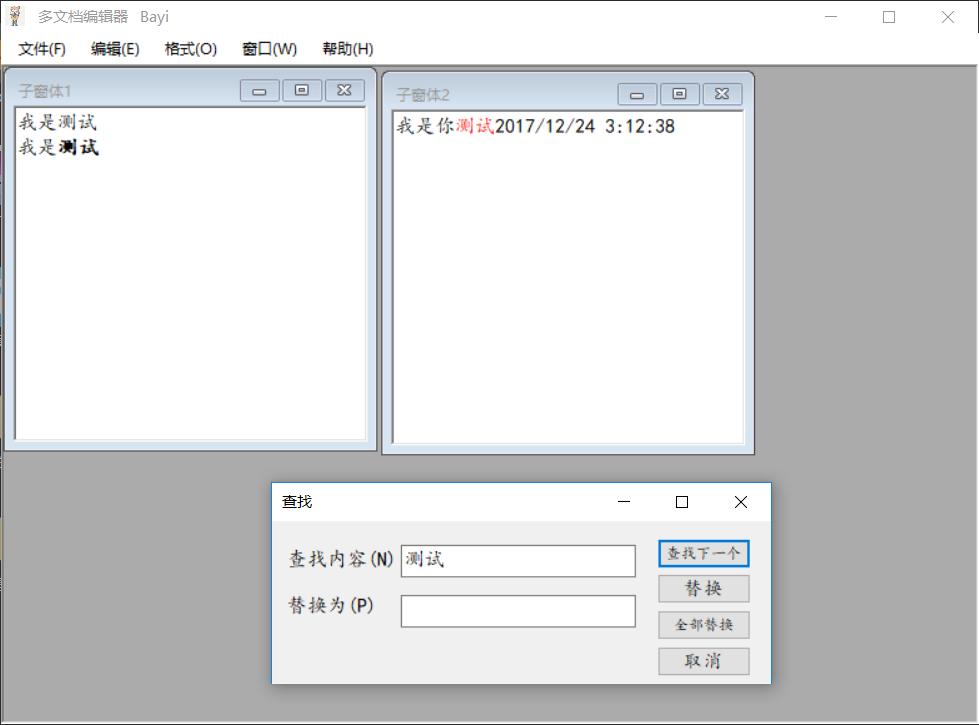
catch (Exception ex)

{

return;

}

}

3 系统调试

4 总结

通过多文档编辑器的程序设计的实现，对C#窗体程序中子窗体和父窗体的交互数据有了更加深层次的理解，还有对各种功能的实现，有些看似很难的功能通过长时间的琢磨也是能够实现的，很感谢这次实践上机的机会，学到了很多设计的思想，和程序整合的理论性实践性的结合

**实验三 通讯录**

1 系统概述

该系统是由C#编写的实现学生管理简单功能的通讯录，拥有学生添加，删除，编辑，查找，备份，恢复等功能，文件是以xml形式保存

2 系统设计

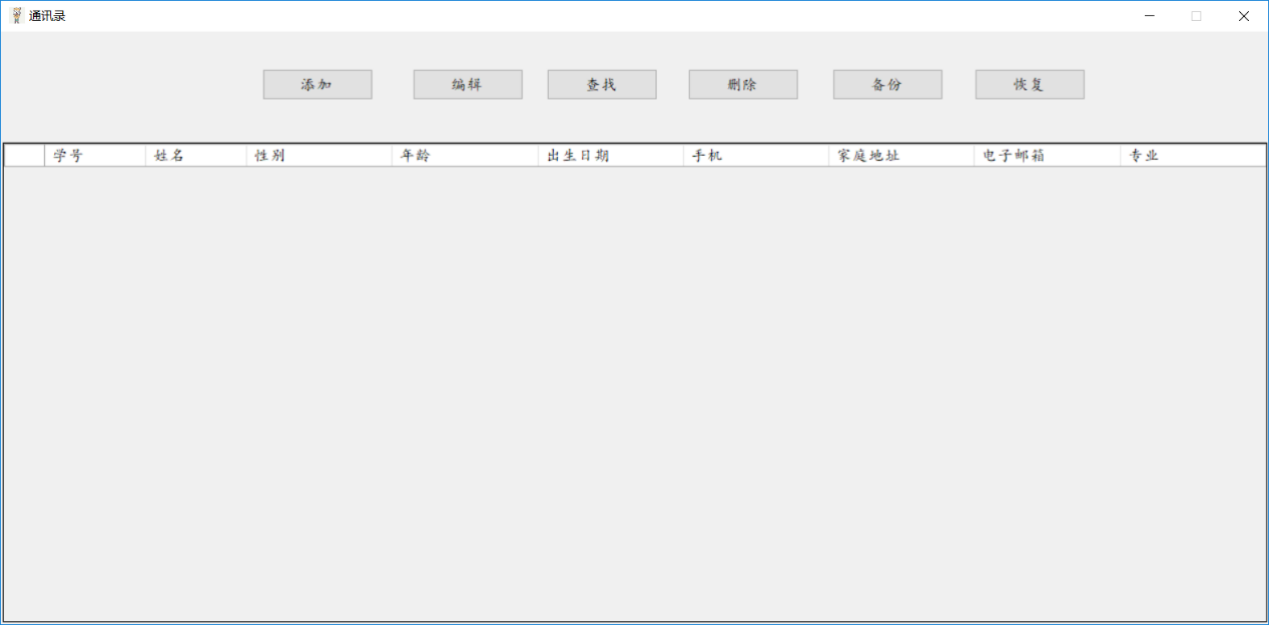
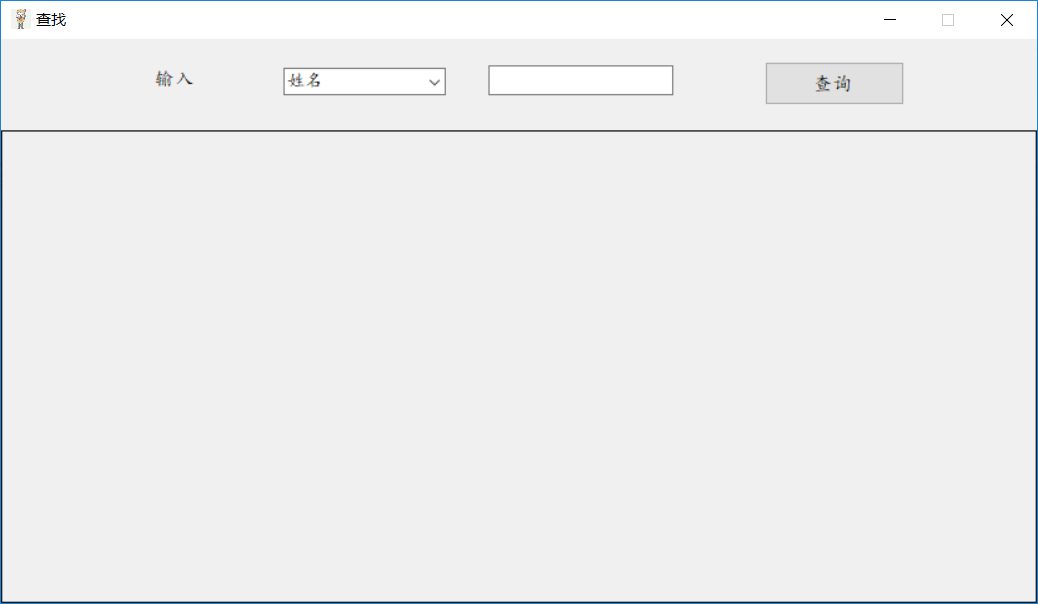
2.1 设计目标

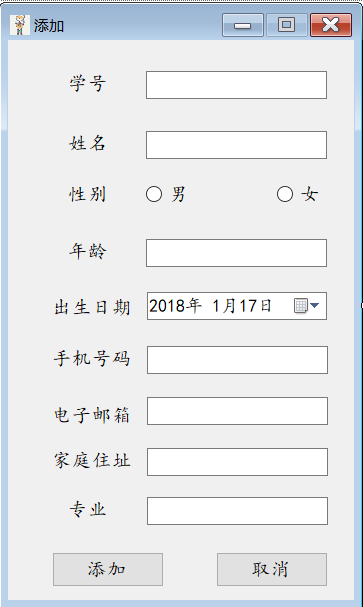
该项目目标是设计开发一个支持学生信息录入编辑的通讯录，通过添加按钮，可以新建学生信息并录入列表，对于录入成功的信息能进行保存和备份操作，同时也支持备份恢复等拓展功能

2.2 设计分析与算法流程

通讯录的主要功能是完成学生信息的添加和编辑，保存等操作，而且支持一些字体颜色等的拓展，通过在主窗口添加列表控件和Button控件，同时设计添加窗体，添加RadioButton,日期控件等实现日期的输入和保存，查找函数则是在已经保存的数据中，Bool是否存在包含查找关键字来判断和显示学生列表，备份和恢复功能则是应用C#中相应的Xml封装函数对文件和列表输入的信息进行一系列的筛选操作，并生成相对应的XML文件的格式，对于恢复功能则是进行相逆的操作，即读取XML文件，然后显示在主窗口的Table控件中

2.3 界面设计





2.4 关键类图

private void buttonBackup\_Click(object sender, EventArgs e)

{

Stream myStream;

SaveFileDialog savefiledialog1 = new SaveFileDialog();

savefiledialog1.Filter = "xml file (\*.xml)|\*.xml";//设置文件格式

savefiledialog1.FilterIndex = 2;//选择系统样式

savefiledialog1.RestoreDirectory = true;

if (savefiledialog1.ShowDialog() == DialogResult.OK)

{

if ((myStream = savefiledialog1.OpenFile()) != null)

{

using (StreamWriter sw = new StreamWriter(myStream))

{

string str = File.ReadAllText(\_bp + @"student.xml");

sw.Write(str);

}

}

myStream.Close();

MessageBox.Show("备份成功!");

}

}

private void buttonRedo\_Click(object sender, EventArgs e)

{

string fName;

OpenFileDialog openFileDialog=new OpenFileDialog();

openFileDialog.InitialDirectory=AppDomain.CurrentDomain.SetupInformation.ApplicationBase;

openFileDialog.Filter="xml file (\*.xml)|\*.xml";;

openFileDialog.RestoreDirectory=true;

openFileDialog.FilterIndex=1;

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

fName = openFileDialog.FileName;

List<StudentInfo> studentList = new List<StudentInfo>();

XElement xml = XElement.Load(fName);

var studentVar = xml.Descendants("student");

studentList = (from student in studentVar

select new StudentInfo

{

StudentID = Int32.Parse(student.Attribute("studentid").Value),

Name = student.Element("name").Value,

Age = Int32.Parse(student.Element("age").Value),

Sex = student.Element("sex").Value,

BirthDate = DateTime.Parse(student.Element("birthdate").Value),

Phone = student.Element("phone").Value,

HomeAddress = student.Element("homeaddress").Value,

Email = student.Element("email").Value,

Profession = student.Element("profession").Value

}).ToList();

dataGridView1.DataSource = studentList;

MessageBox.Show("恢复成功!");

}

}

public static List<StudentInfo>GetStudentInfoList(StudentInfo param)

{

List<StudentInfo> studentList = new List<StudentInfo>();

XElement xml = XElement.Load(\_basePath);

var studentVar = xml.Descendants("student");

if(param.StudentID!=0)

{

studentVar = xml.Descendants("student").Where(a => a.Attribute("studentid").Value == param.StudentID.ToString());

}

else if(!String.IsNullOrEmpty(param.Name))

{

studentVar = xml.Descendants("student").Where(a => a.Element("name").Value == param.Name);

}

studentList = (from student in studentVar

select new StudentInfo

{

StudentID = Int32.Parse(student.Attribute("studentid").Value),

Name = student.Element("name").Value,

Age = Int32.Parse(student.Element("age").Value),

Sex = student.Element("sex").Value,

BirthDate = DateTime.Parse(student.Element("birthdate").Value),

Phone = student.Element("phone").Value,

HomeAddress = student.Element("homeaddress").Value,

Email = student.Element("email").Value,

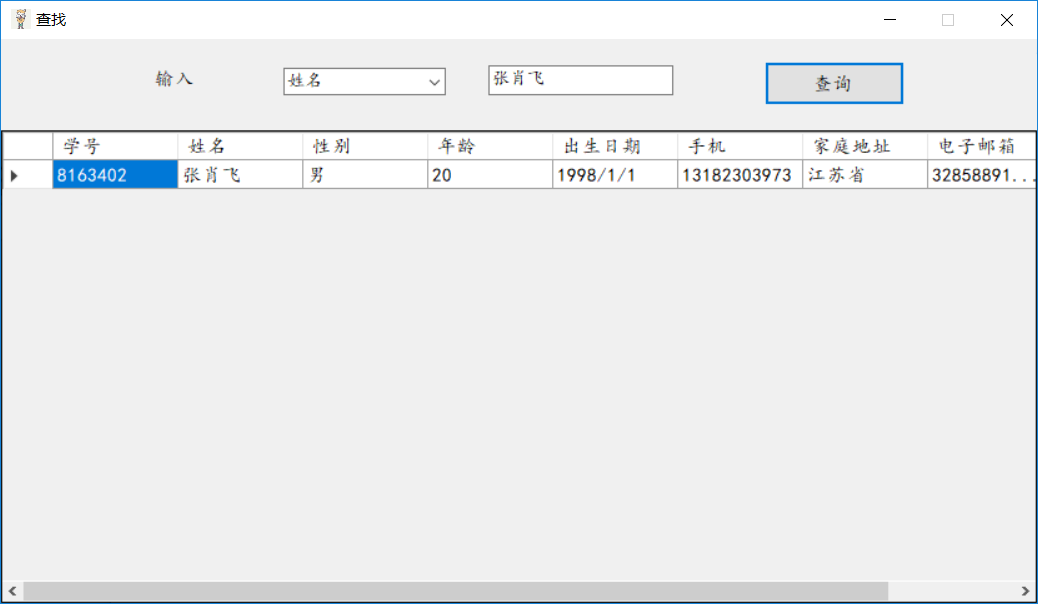
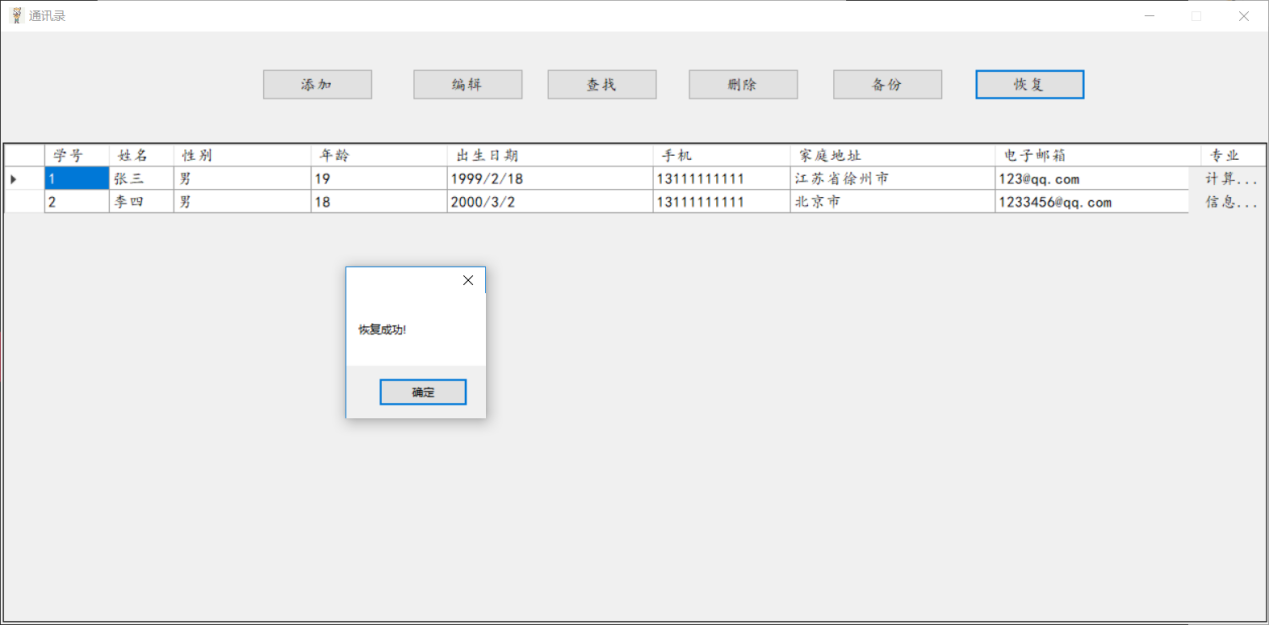
Profession = student.Element("profession").Value

}).ToList();

return studentList;

}

3 系统调试



4 总结

通过通讯录的程序设计的实现，C#窗体程序对Table控件的使用增加了理解，同时对于C#操作XML文件的相互转换有了新的学习和掌握，以前学习web知识对于数据库操作还是比较熟悉的，这次的实践感受到文件保存和操作对于信息转移以及查看的方便性，很不错的经历

**实验四 拼图游戏**

1 系统概述

该系统是由Python,wxPython,及Pygame模块编写的分割拼图小游戏，拥有自定义图片，自定义分割数，时间限制等功能，成功拼图时加入了音效提示

2 系统设计

2.1 设计目标

本项目目标是设计开发一个拼图的游戏软件，软件能够自动对加载的图片进行分割，并打乱顺序后放置在不同的图片框中，用户使用鼠标点击图片框中的图片进行拼图，拼图成功后，系统会自动进行提示

2.2 设计分析与算法流程

图片尺寸自适应：为方便用户能够对不同尺寸的图片进行拼图，软件加入了图片尺寸自动调整功能，能对不同尺寸的图片进行自动调整以满足拼图游戏软件的要求

图片动态分割：能自动将图片分割为不同的大小，如3×3矩阵或其它大小的矩阵，便于控制拼图游戏的难易程度

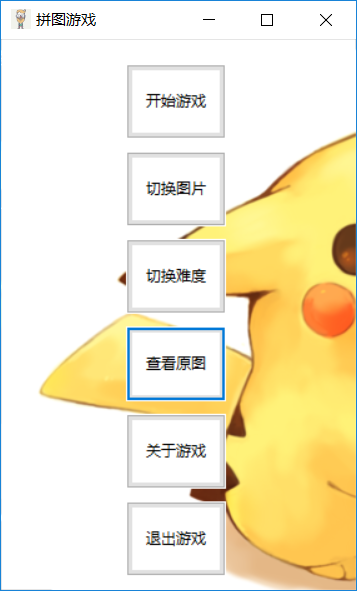
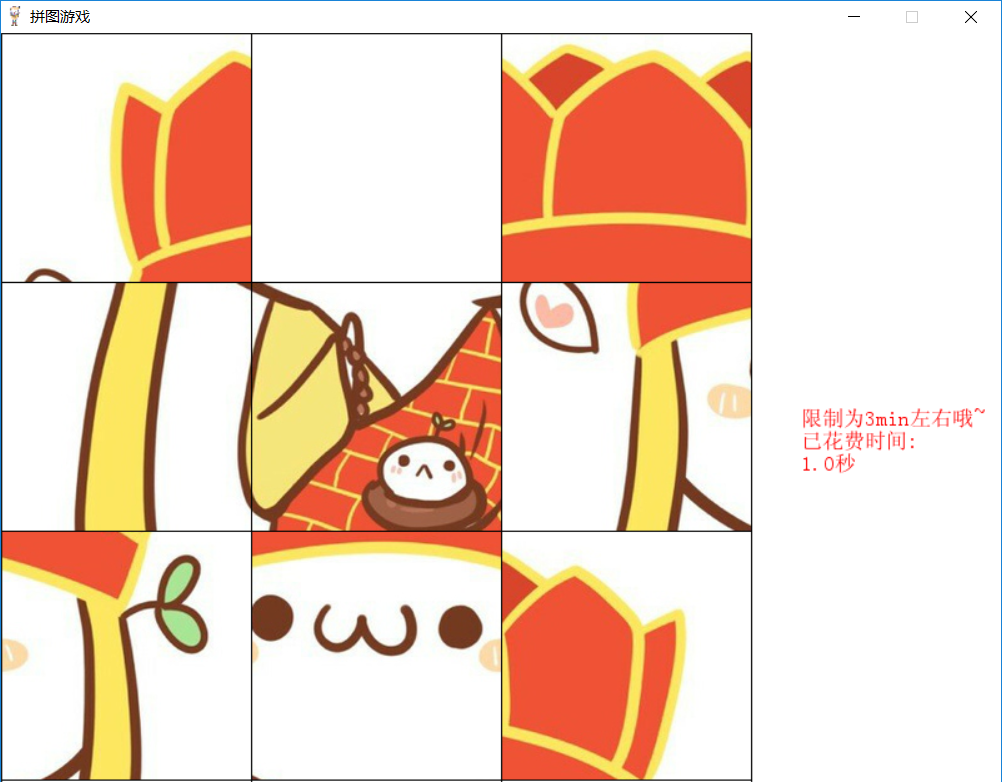
查看原图：为方便拼图，允许用户随时查看原图

随机切换图片：能在软件自身携带的图片中进行随机切换，也可以另外选择用户自己喜欢的图片进行拼图，增加趣味性

自动判断拼图成功：软件自动记忆图片切割后的原始状态，能够对用户拼图是否成功自动做出判断进行音效小提示

时间限制：计算游戏开始的时间，并且同步进行计时，如果超时自动结束游戏(由于点击以及程序运行误差，显示的时间会和设定的时间相差1~2s)

2.3 界面设计



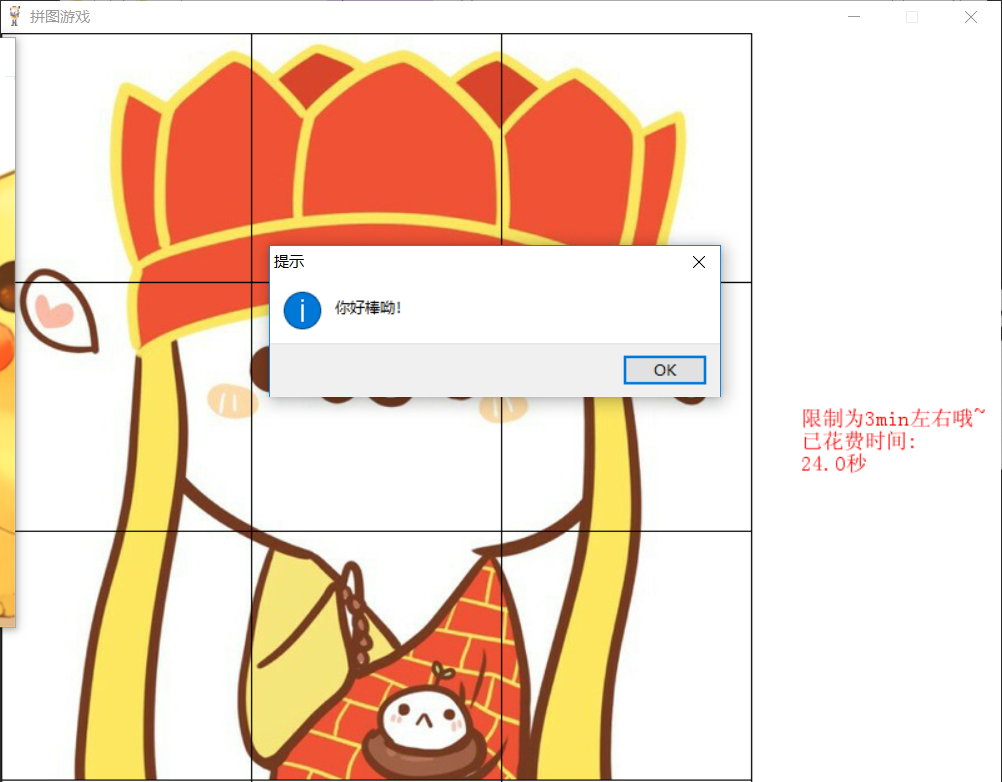
2.4 关键类图

**def** changePicButton\_click(self):  
 dlg = wx.FileDialog(self, **u"选择新图片"**,os.getcwd(),style=wx.DD\_DEFAULT\_STYLE,wildcard= **"(\*.jpg)|\*.jpg|" "(\*.png)|\*.png"**)  
 **if** dlg.ShowModal() == wx.ID\_OK:  
 self.fileAddress=dlg.GetPath()  
 dlg.Destroy()  
  
**def** changeLevelButton\_click(self):  
 levelEntry = wx.TextEntryDialog(None, **u"请输入要设置的分割数:"**, **u"修改图片分割数"**,**u'3'**)  
 **if** levelEntry.ShowModal() == wx.ID\_OK:  
 message = levelEntry.GetValue()  
 **if** str(message).isdigit() **and** int(message)>=3 **and** int(message)<=9:  
 self.VHNUMS=int(message)  
 **else**:  
 wx.MessageBox(**"请检查输入!"**,**"提示"**)

**def** draw\_info(surfacetodraw,minutes,seconds,font):  
 surfacetodraw.fill(BACKGROUNDCOLOR)  
 now = time.localtime()  
 now\_min,now\_sec = now.tm\_min,now.tm\_sec  
 time\_passed = now\_min \* 60.0 + now\_sec - (minutes \* 60.0 + seconds)  
 str\_time\_limit=**u"限制为3min左右哦~"** str\_time\_sign = **u"已花费时间:"** str\_time\_passed=str(time\_passed) + **u"秒"** limit\_surface=font.render(str\_time\_limit,True,red)  
 sign\_surface = font.render(str\_time\_sign,True,red)  
 time\_surface = font.render(str\_time\_passed,True,red)  
 surfacetodraw.blit(limit\_surface,(0,0))  
 surfacetodraw.blit(sign\_surface,(0,limit\_surface.get\_height()))  
 surfacetodraw.blit(time\_surface,(0,limit\_surface.get\_height()+sign\_surface.get\_height()))

**for** event **in** pygame.event.get():  
 **if** event.type == QUIT:  
 terminate()  
 temp = False  
 **if** finish:  
 success.play()  
 successEnd()  
 terminate()  
 temp = False  
 finish=False  
 **if** passtime:  
 falseEnd()  
 terminate()  
 temp=False  
 passtime=False  
 **if** event.type == KEYDOWN:  
 **if** event.key == K\_LEFT **or** event.key == ord(**'a'**):  
 blackCell = moveLeft(gameBoard, blackCell,VHNUMS,CELLNUMS)  
 **if** event.key == K\_RIGHT **or** event.key == ord(**'d'**):  
 blackCell = moveRight(gameBoard, blackCell,VHNUMS,CELLNUMS)  
 **if** event.key == K\_UP **or** event.key == ord(**'w'**):  
 blackCell = moveUp(gameBoard, blackCell,VHNUMS,CELLNUMS)  
 **if** event.key == K\_DOWN **or** event.key == ord(**'s'**):  
 blackCell = moveDown(gameBoard, blackCell,VHNUMS,CELLNUMS)  
 **if** event.type == MOUSEBUTTONDOWN **and** event.button == 1:  
 x, y = pygame.mouse.get\_pos()  
 col = int(x / cellWidth)  
 row = int(y / cellHeight)  
 index = col + row \* VHNUMS  
 **if** (index == blackCell - 1 **or** index == blackCell + 1 **or** index == blackCell - VHNUMS **or** index == blackCell + VHNUMS):  
 gameBoard[blackCell], gameBoard[index] = gameBoard[index], gameBoard[blackCell]  
 blackCell = index

3 系统实现（运行调试）



4 总结

通过这次拼图游戏实验的学习，学习了wxPython，Pygame的使用，体验到了做小游戏的成就感，也感受到了程序和Python的魅力