**中 国 地 质 大 学**

**本科生课程报告**



课程名称： c# 程 序 设 计

教师姓名 杨鸣

学生姓名 单凯

学生班级 191134

学生学号 20131004087

学生专业 计算机科学与技术

所在院系 计算机学院

时间: 2016年 4 月

**目录**

[摘要 1](#_Toc14656)

[一、 题目内容 1](#_Toc9211)

[二、 系统环境 1](#_Toc1784)

[三、 需求分析 1](#_Toc4708)

[1. 最基本功能： 1](#_Toc14629)

[2. 系统目标 1](#_Toc30532)

[3. 功能需求分析 2](#_Toc15728)

[4. 性能需求分析 2](#_Toc25152)

[四、 设计方案 2](#_Toc25349)

[（1） 概要设计 2](#_Toc21383)

[1． 软件部分： 2](#_Toc10306)

[2． 数据库的概念结构设计 2](#_Toc31552)

[（2） 详细设计 3](#_Toc744)

[① 数据库中表的设计 3](#_Toc19834)

[② 用户管理 3](#_Toc24277)

[1) 注册 3](#_Toc22140)

[当用户点击“确定”按钮时： 3](#_Toc12177)

[当用户点击“取消”按钮时： 4](#_Toc23203)

[补充： 4](#_Toc4911)

[2) 登录 4](#_Toc26164)

[当用户点击“登录”按钮时： 4](#_Toc14678)

[当用户点击“取消”按钮时： 5](#_Toc21714)

[当用户点击“退出”按钮时： 5](#_Toc5993)

[3) 注销 5](#_Toc21871)

[4) 查看当前用户 5](#_Toc31534)

[5) 补充细节问题： 5](#_Toc22368)

[a. 用户登录安全性设计： 5](#_Toc27361)

[b. 其它： 5](#_Toc16908)

[③ Excel文件 6](#_Toc22261)

[1) 新建excel文件 6](#_Toc6830)

[2) 打开excel文件 6](#_Toc4882)

[3) 保存excel文件 7](#_Toc15979)

[4) 插入行、列 9](#_Toc8511)

[5) 删除行、列 10](#_Toc8921)

[6) 修改列名 11](#_Toc21949)

[7) 横向求和、求平均 12](#_Toc13354)

[8) 纵向求和、求平均 12](#_Toc6403)

[9) 切换表单 12](#_Toc31177)

[10) 细节问题补充 13](#_Toc14966)

[④ 视频文件 13](#_Toc23707)

[⑤ 文本文件 14](#_Toc25116)

[1) 创建word 14](#_Toc1152)

[2) 保存word 14](#_Toc24720)

[⑥ 网络功能 15](#_Toc11991)

[这个就是一个发送邮件的功能。 15](#_Toc22540)

[算法： 15](#_Toc11781)

[效果图： 15](#_Toc20030)

[⑦ 小组件 16](#_Toc596)

[ 一个用WPF写的简单的秒表程序。 16](#_Toc30273)

[ WPF写的图片效果展示程序 16](#_Toc16033)

[效果示意图如下： 16](#_Toc15096)

[算法流程框图如下： 17](#_Toc16313)

[⑧ 其它 18](#_Toc5527)

[⑨ 软件总的流程 19](#_Toc13134)

[⑩ 压缩文件 19](#_Toc22817)

[压缩文件流程图 19](#_Toc5104)

[效果示意图： 19](#_Toc8991)

[⑪ 在文本文件编辑的窗体中添加“插件” 20](#_Toc10605)

[算法流程框图如下所示： 20](#_Toc25318)

[效果示意图： 21](#_Toc22909)

[⑫ 图像旋转算法 22](#_Toc13343)

[流程框图： 22](#_Toc12983)

[效果示意图： 22](#_Toc28265)

[五、 系统实现 23](#_Toc11279)

[关键部分及代码： 23](#_Toc7001)

[① 图片旋转代码 23](#_Toc9590)

[② 连接数据库 24](#_Toc19743)

[③ 打开excel文件 25](#_Toc23125)

[④ 发送邮件 25](#_Toc25488)

[⑤ 通过xaml编写界面 25](#_Toc15605)

[⑥ 解压文件关键代码 25](#_Toc31350)

[⑦ Xml文件编程 26](#_Toc10771)

[⑧ 在图片上添加文字 26](#_Toc21321)

[⑨ 保存图像 26](#_Toc7733)

[⑩ 进度条控件和多线程 27](#_Toc23996)

[⑪ 验证码： 27](#_Toc8821)

[⑫ 补充说明： 28](#_Toc24015)

[六、 遇到的问题以及解决方案 28](#_Toc5345)

[① 对象已经释放了,程序代码又去访问它 28](#_Toc25404)

[问题现象： 29](#_Toc20643)

[问题原因： 29](#_Toc9593)

[结决方案： 29](#_Toc7300)

[参考材料： 29](#_Toc12458)

[② 邮件发送失败异常 29](#_Toc14997)

[问题现象： 29](#_Toc29487)

[问题原因： 30](#_Toc2554)

[解决方案： 30](#_Toc30913)

[参考材料： 30](#_Toc29737)

[③ Mdi容器默认的灰色背景太丑了。 31](#_Toc18219)

[问题现象： 31](#_Toc26917)

[解决方案： 31](#_Toc12842)

[参考材料： 31](#_Toc2675)

[④ 进程杀不死 31](#_Toc26365)

[问题现象： 31](#_Toc25313)

[问题原因： 31](#_Toc13728)

[解决方案： 31](#_Toc14676)

[参考材料： 31](#_Toc3282)

[⑤ 当窗体放大后，但是空间还是在原来的位置 31](#_Toc16185)

[问题现象： 31](#_Toc1874)

[解决过程： 32](#_Toc8014)

[解决方案： 33](#_Toc22491)

[参考材料： 33](#_Toc20360)

[⑥ 区分anchor属性和dock属性 33](#_Toc25494)

[⑦ 在图片中添加文字时，坐标位置发生偏移 34](#_Toc32406)

[问题现象： 34](#_Toc3785)

[⑧ 进度条不动 34](#_Toc18058)

[⑨ show（）和 showDialog（）的区别 35](#_Toc619)

[⑩ 数据库连接出错 35](#_Toc4299)

[问题现象： 35](#_Toc7506)

[问题原因： 36](#_Toc6078)

[解决方案： 36](#_Toc13223)

[参考材料： 36](#_Toc15572)

[七、 完成的项目中存在的不足 36](#_Toc20289)

[① 有的地方只是简单的抛出了异常，保证程序不会因此而终止，并未做详细的处理。 36](#_Toc16)

[② 有的想做的功能因水平有限，未能实现 36](#_Toc8084)

[③ 用户交互不是很好 37](#_Toc11427)

[④ 存在冗余和bug。 37](#_Toc32340)

[八、 个人小结 37](#_Toc11158)

[① 本次作业所用到的知识点小结： 37](#_Toc22016)

[② C#程序设计个人总结 37](#_Toc13468)

[九、 参考材料 38](#_Toc25071)

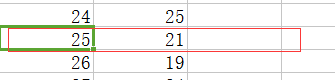
[附录： 38](#_Toc5748)

**摘要**

本文描述的是基于Windows环境的用c#编写的简单的文件处理程序，主要功能模块包括：用户管理，文件处理，其它附加功能,主要工具Microsoft Visual Studio 2015 设计窗体, SQL server2014建立数据库。

关键字：C# SQL winform wpf

1. **题目内容**

****

**本人班号25，做第21题。如有问题，可联系我，tel：18186482897，QQ：1431173103.**

操作EXCEL文件：使用EXCEL.EXE组件将列表控件中显示的数据保存为EXCEL格式文件，并可以一个EXCEL文件的内容显示在列表控件中。



因为做这个题目的时间比较长，所以自己做了相应的扩展，是程序变得更加丰满。我也把许多平时写过的程序组合到这次的作业当中去了。

1. **系统环境**

**所需的开发运行环境:**

软件:*visual studio* 2015 *Microsoft SQL SERVER 2014*

硬件:个人PC（win10+8G内存+500G硬盘）

1. **需求分析**
2. **最基本功能：**

将列表控件中的数据保存为excel文件，可以打开一个二维表格式的excel文件（这个excel文件是最基本的，没有图片，没有合并单元格等复杂功能），将其内容显示到列表控件中。

可以在此基础上进行相应扩展，下面就是做了相应扩展的分析。

1. **系统目标**
2. 打开保存excel文件，可以进行简单编辑：修改单元格内容、增加删除行列、修改列名称、简单排序、简单统计（求和，平均数等）
3. 可以打开保存文本文件、图像、视频等文件，并可以做简单的处理
4. 采用用户管理，当且仅当只有用户在登陆之后，才可以使用软件
5. 其他的组件、插件，通过调用进程，引入dll文件等方式增加功能。
6. **功能需求分析**
7. 文件打开保存：点击菜单后，通过文件浏览对话框，让用户选择相应文件。
8. 其它操作：可以通过菜单或右键菜单进行操作
9. **性能需求分析**

(1) 登录、用户界面需求：简洁、易懂、易用、友好的用户界面。

(2) 安全保密性需求：只有凭借用户名和密码登陆系统，才能进行操作等。

1. **设计方案**
2. 概要设计
3. **软件部分：**

经过对系统的需求分析，将软件主要划分为三个部分：用户管理，几种不同类型的文件的管理，其它功能。如图下图所示：

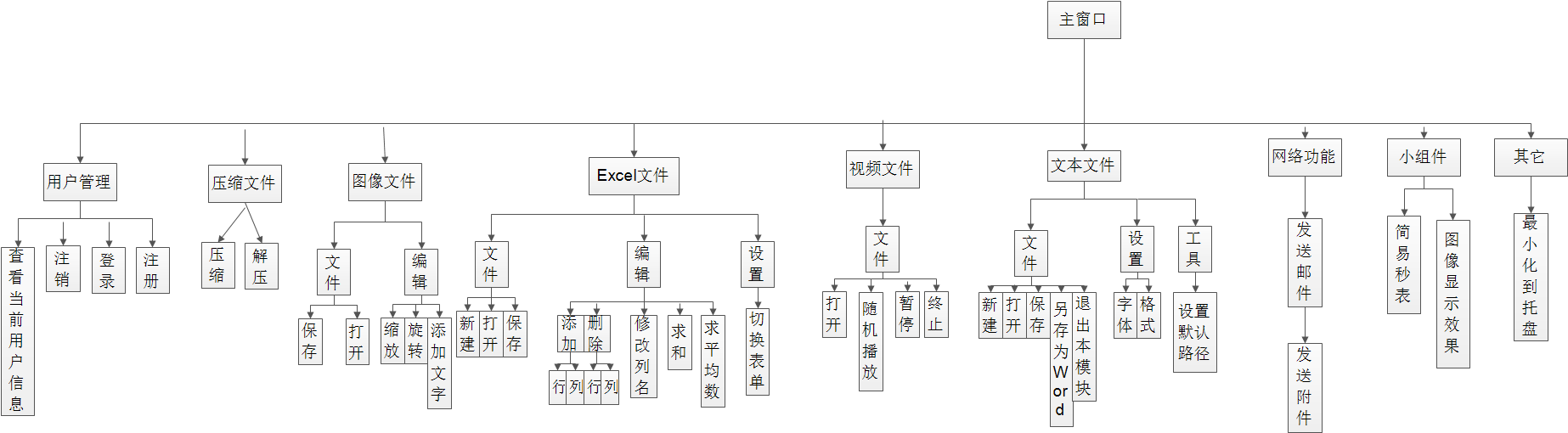


图 1 概要设计图

1. **数据库的概念结构设计**

E-R图如下：



图 2 用户实体E-R图

1. 详细设计
2. **数据库中表的设计**

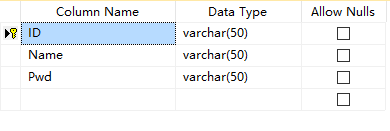
****

图 3 用户表的设计

1. **用户管理**

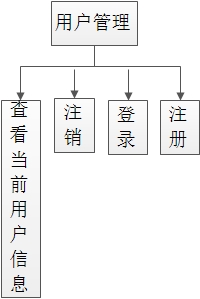
****

图 4 用户管理模块

1. **注册**

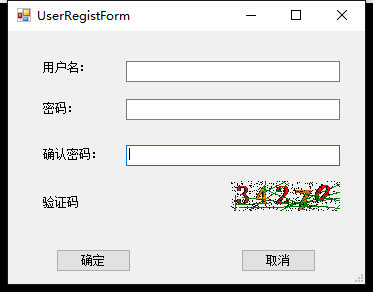


图 5 用户注册界面

**当用户点击“确定”按钮时：**

**算法流程：**

1. 验证密码和确认密码是否一致；如果不一致，提示“密码填写错误”；
2. 如果一致，将密码加密，写到数据库中的tb\_user表当中。
3. 提示“注册成功”

**效果示意图：**

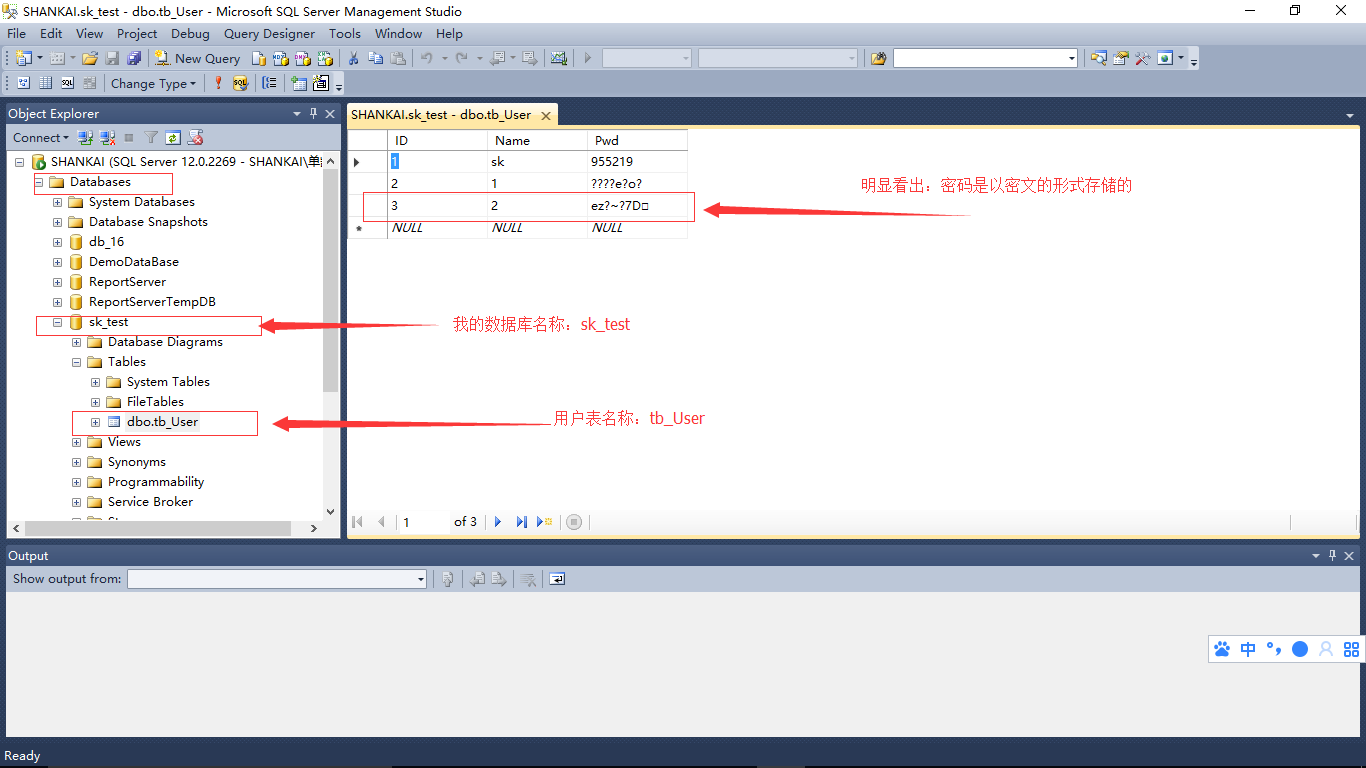


图 6插入到数据库中的用户信息

**当用户点击“取消”按钮时：**

a. 关闭用户注册对话框

**补充：**

那个验证码是个假的，使用GDI绘图+Random函数仿造的，但是是可以点击更换的，所以，注册的时候不需要填写验证码。可在”系统实现——验证码”部分查看代码。不是很复杂，就不写算法了。

1. **登录**

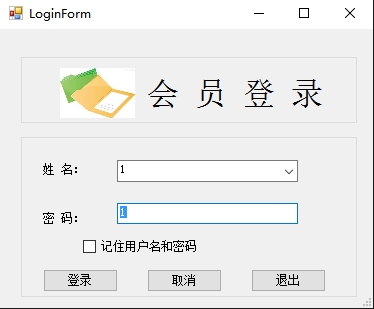
****

图 7 用户登录界面

**当用户点击“登录”按钮时：**

**算法流程：**

1. 连接数据库；
2. 对用户密码进行加密；
3. 查找tb\_User表中内容，如果用户名和密码匹配，登陆成功；菜单和工具栏可用。
4. 如果匹配不成功，提示错误；

**当用户点击“取消”按钮时：**

1. 清空用户名文本框和密码文本框

**当用户点击“退出”按钮时：**

a. 关闭用户登录对话框

1. **注销**

**2**

图 8 注销后结果

1. 清空CurUser变量（该变量用于保存当前用户名）
2. 菜单和工具栏不可用
3. **查看当前用户**

****

图 9 who am I？

1. 弹出消息框，显示当前用户
2. **补充细节问题：**
3. **用户登录安全性设计：**

系统设计了登录界面，每个合法用户有用户名及一个密码，只有当用户输入正确的用户名及密码组合后才能够使用软件的大部分功能。并且在数据库中存储用户密码的时候存储密文（使用DES加密）。

1. **其它：**

当用户名和密码填写正确切勾选保存用户名和密码，系统会将用户信息保存到“UseList.xml”文件中，下次登录时，用户信息会自动显示出来。如下图所示：

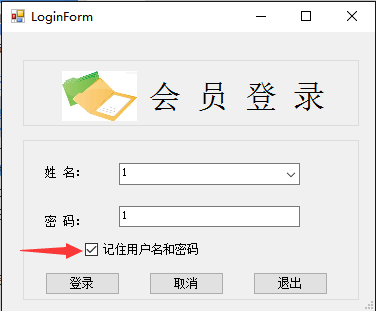
****

图 10 记住用户名和密码

1. **Excel文件**

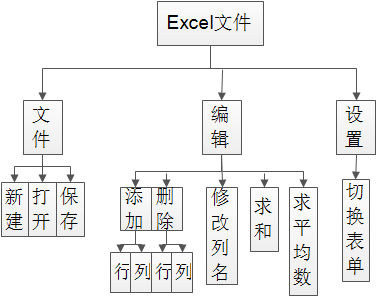
****

图 11 Excel模块设计

1. **新建excel文件**

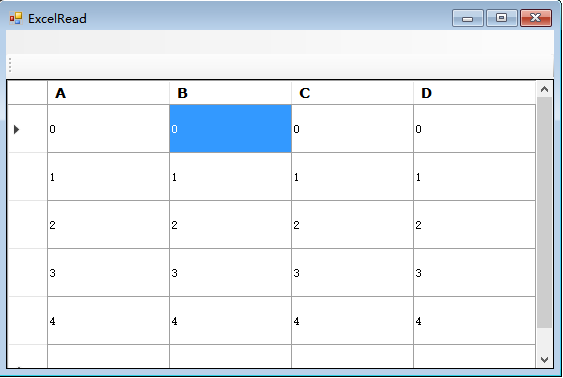
****

图 12 新建excel

当点击“新建”菜单时，显示excel编辑对话框，初始化datagridview控件，效果如上图所示。

1. **打开excel文件**

当点击“打开”菜单时，将用户选中的excel文件作为数据源，将excel文件中的内容显示到datagridview中。效果如下图所示：

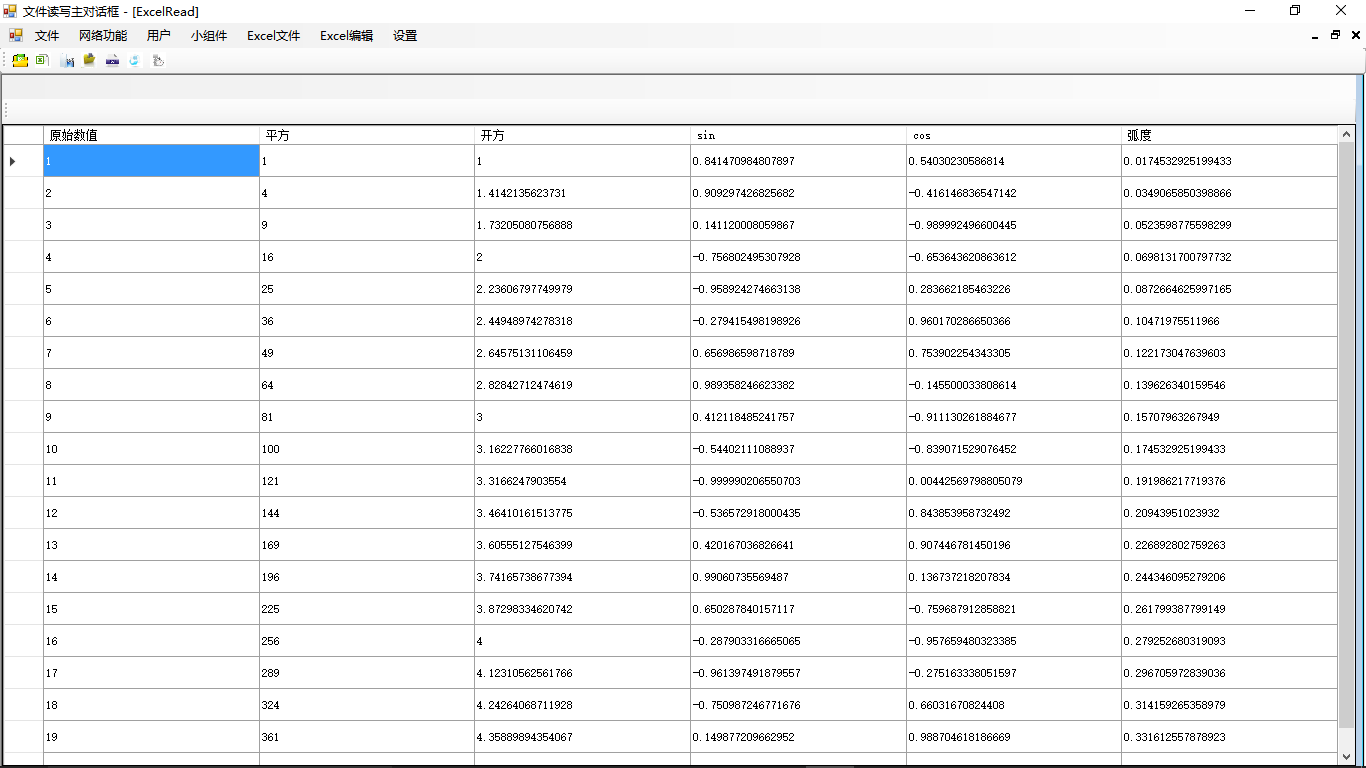
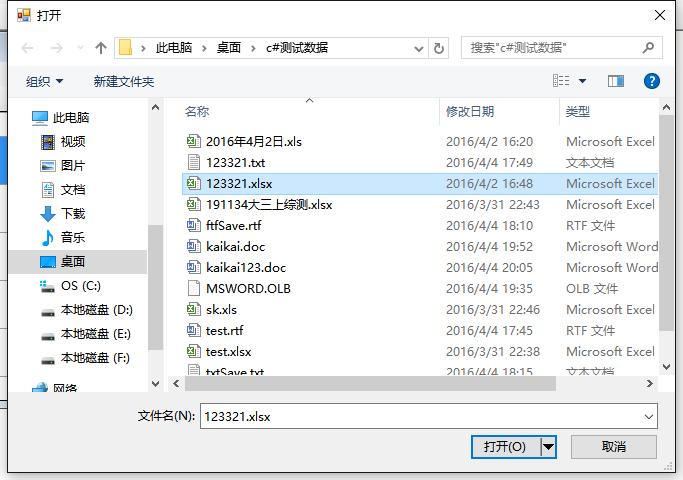
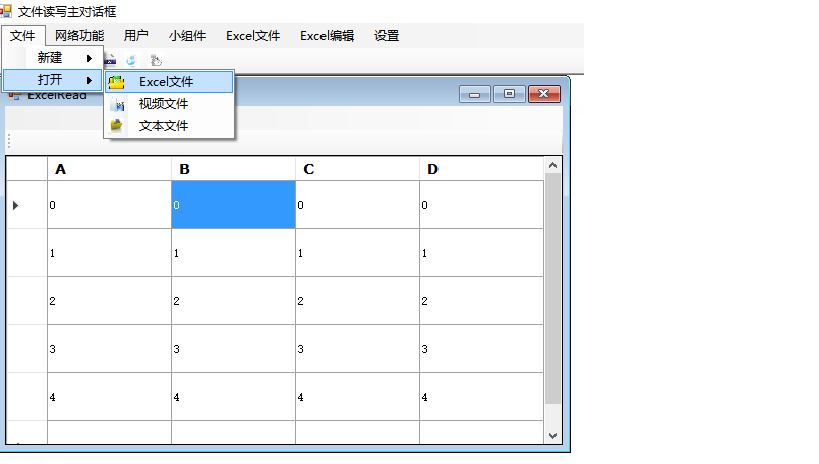
****

图 13 打开excel

算法流程框图如下：

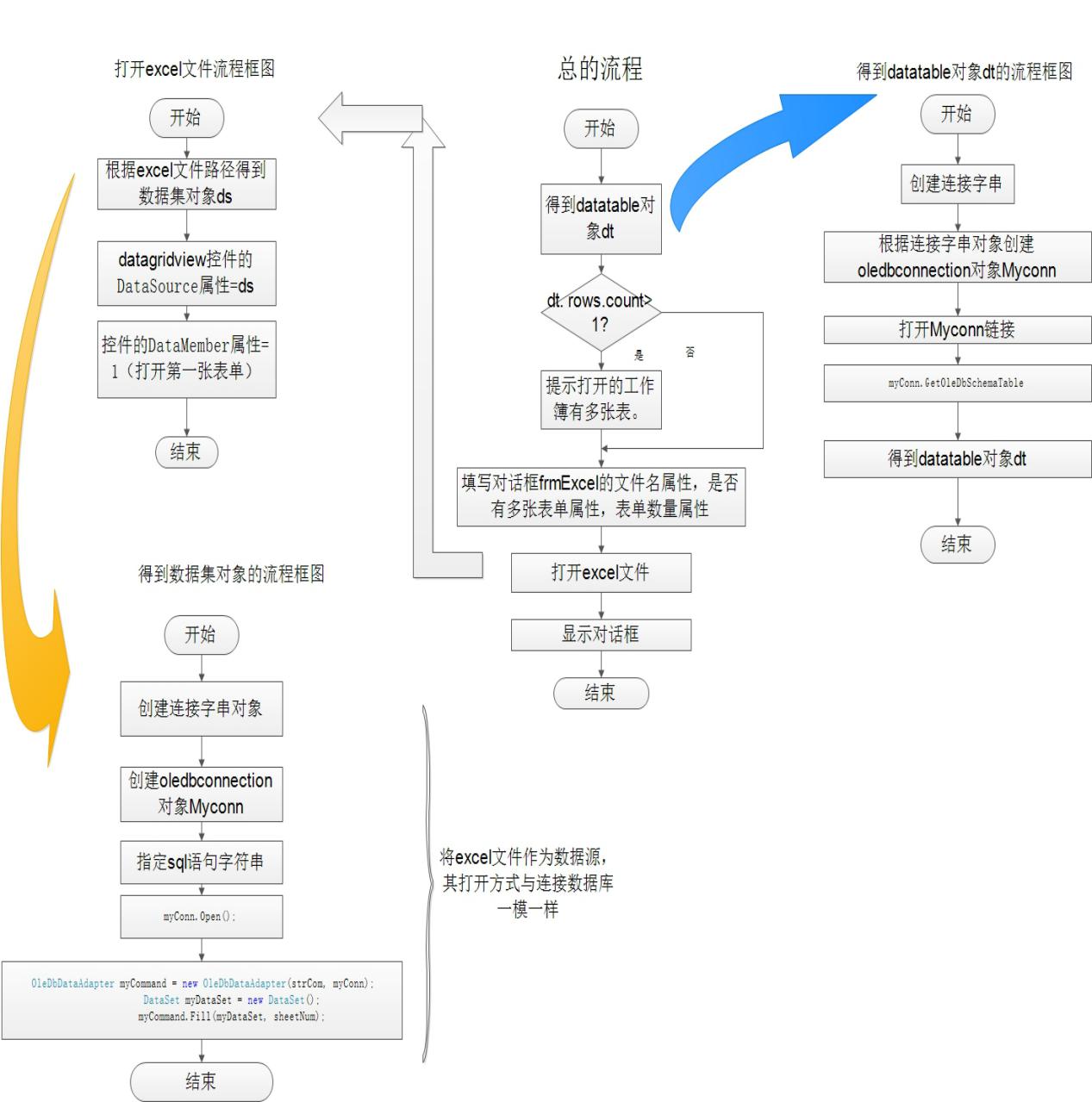


图 14 打开excel文件并显示的算法

1. **保存excel文件**

**将datagridview中每个单元格中的数据写到excel文件当中。使用** Microsoft.Office.Interop.Excel.命名空间下的Application类，Workbook类和Worksheet进行保存。效果如下图所示：

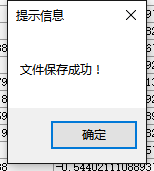
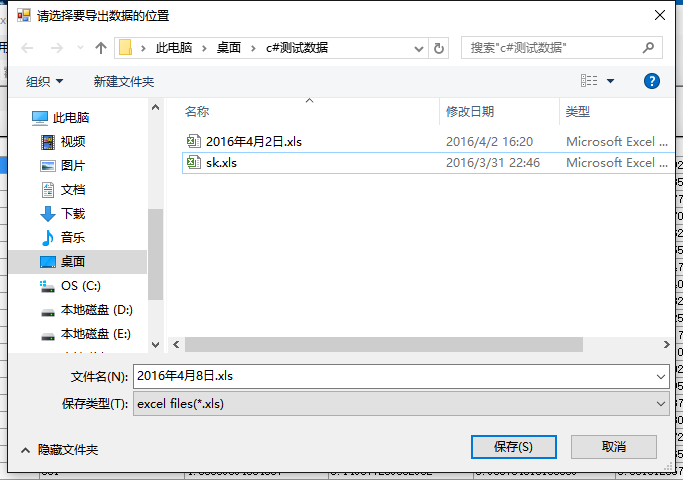
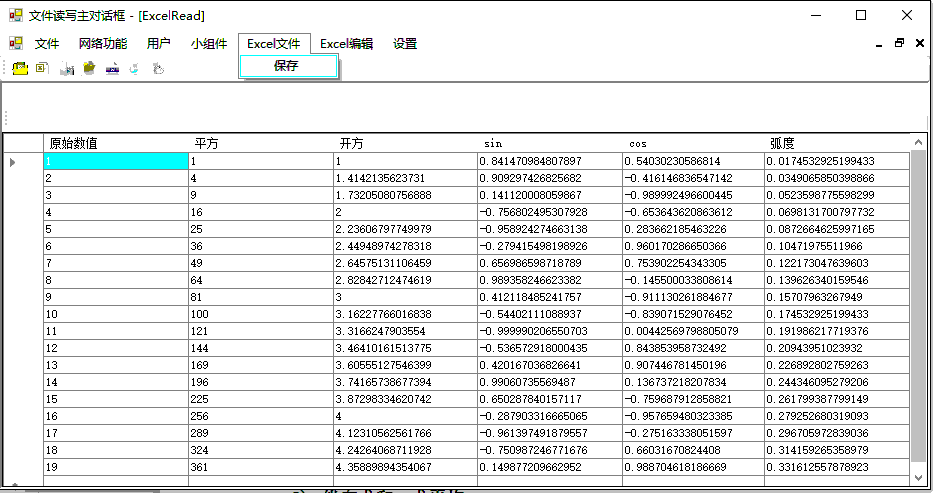
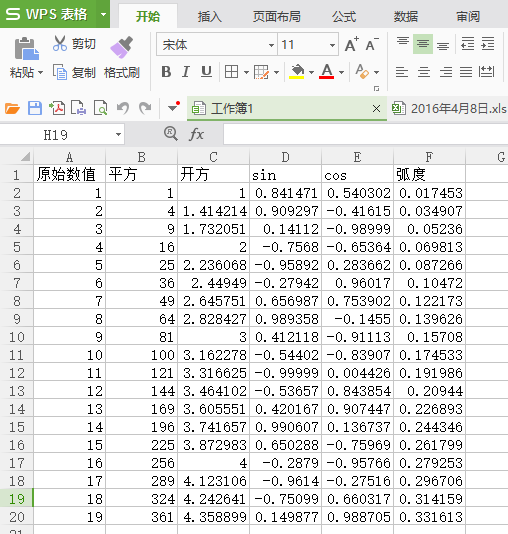
 

图 15 保存excel

其算法流程框图如下：

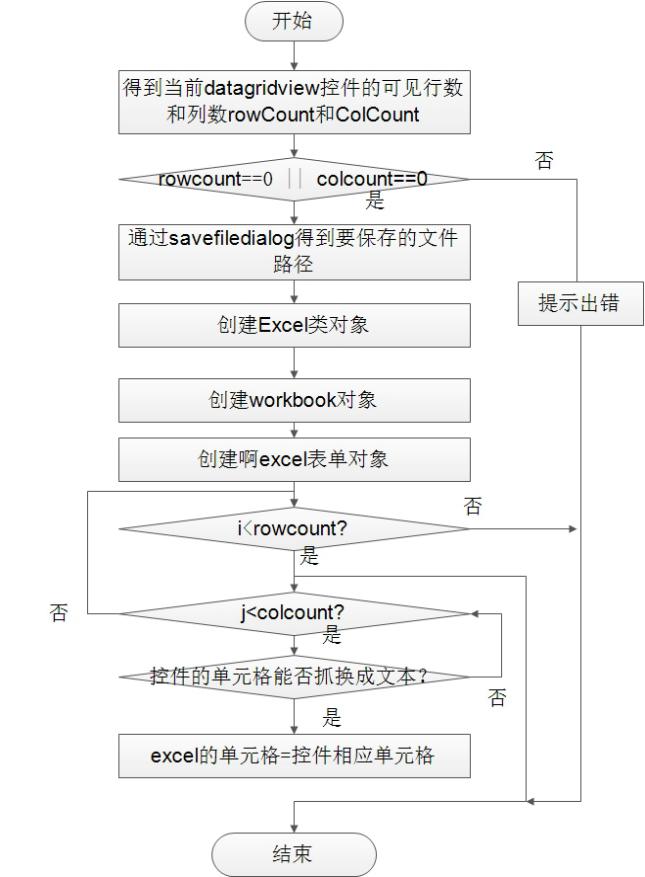


图 16 保存excel文件

1. **插入行、列**

每当点击“插入行”菜单，datagridview控件末尾添加一个空行（此时是新建文件的情况，在打开文件的时候直接编辑即可）；每当点击“插入列”菜单，提示用户输入列名，插入一列。具体流程如下图所示：

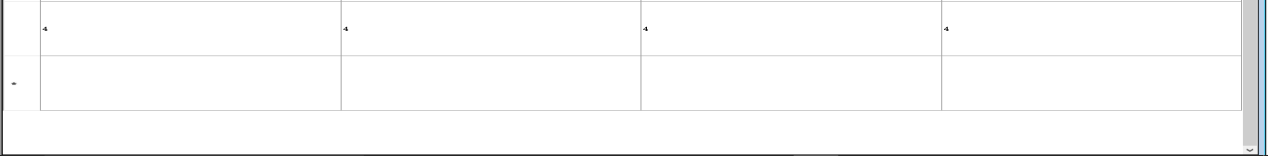


图 17 插入行前

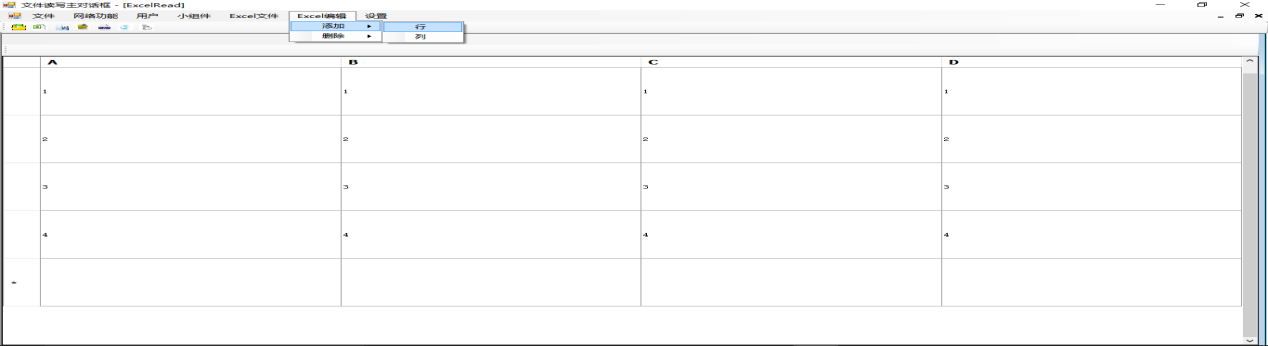


图 18 点击菜单

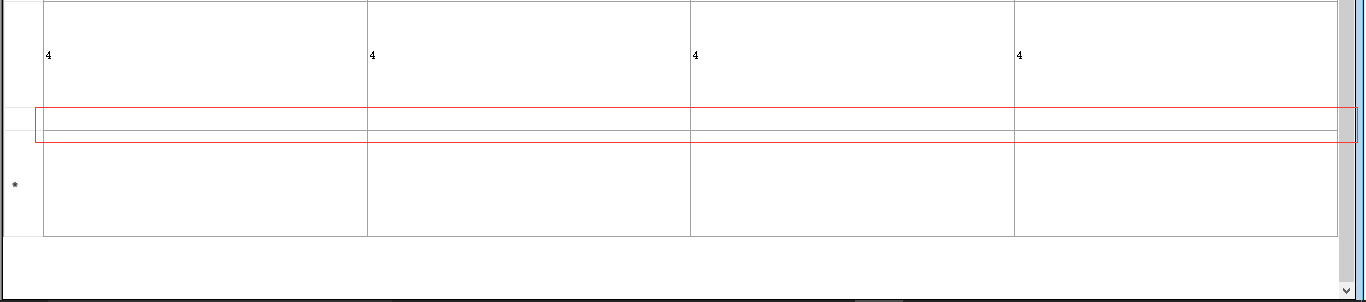


图 19 插入行后

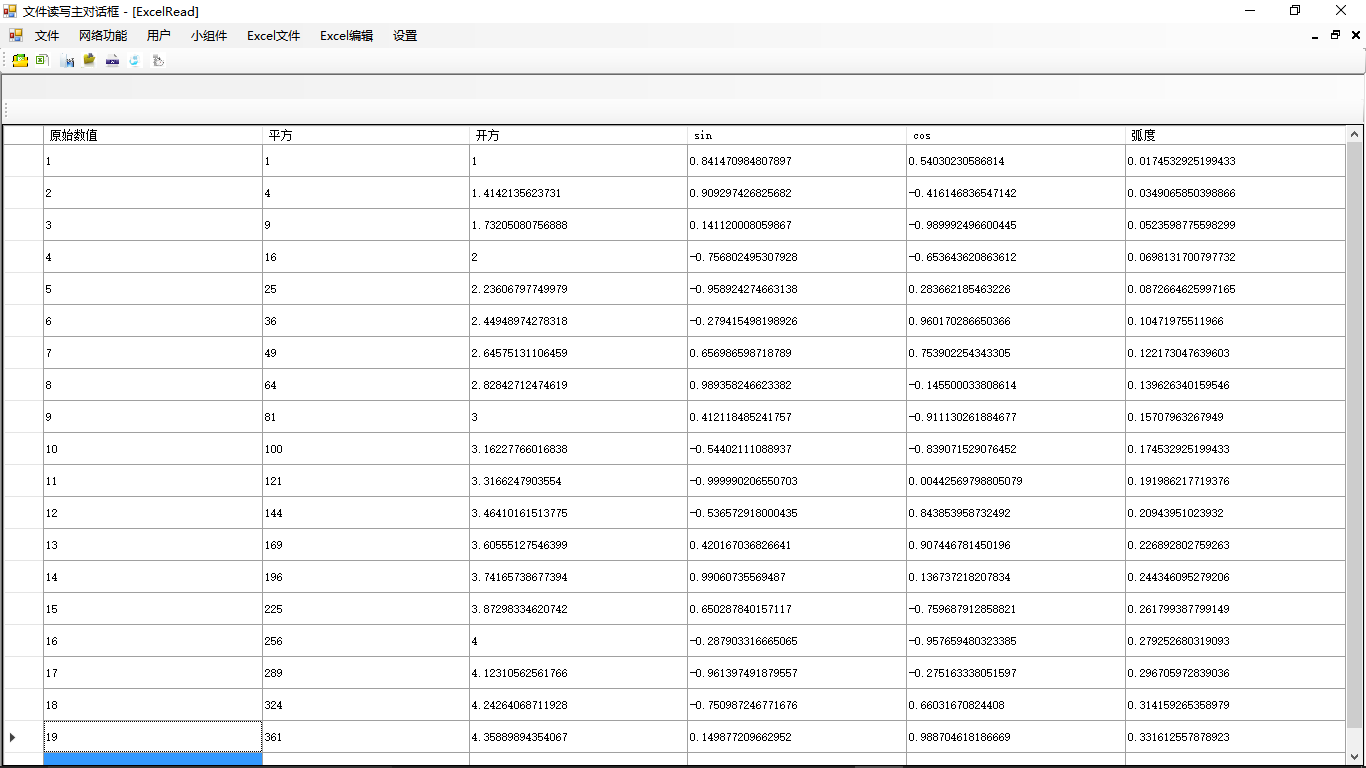


图 20 插入列前

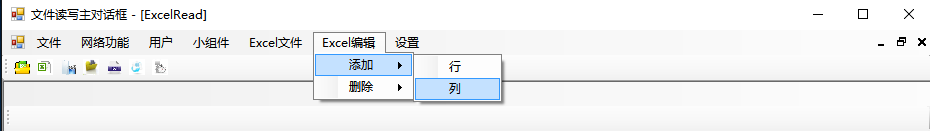


图 21 点击菜单

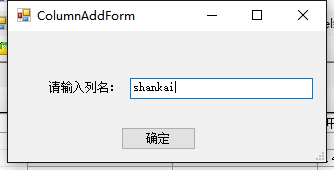


图 22 输入列名

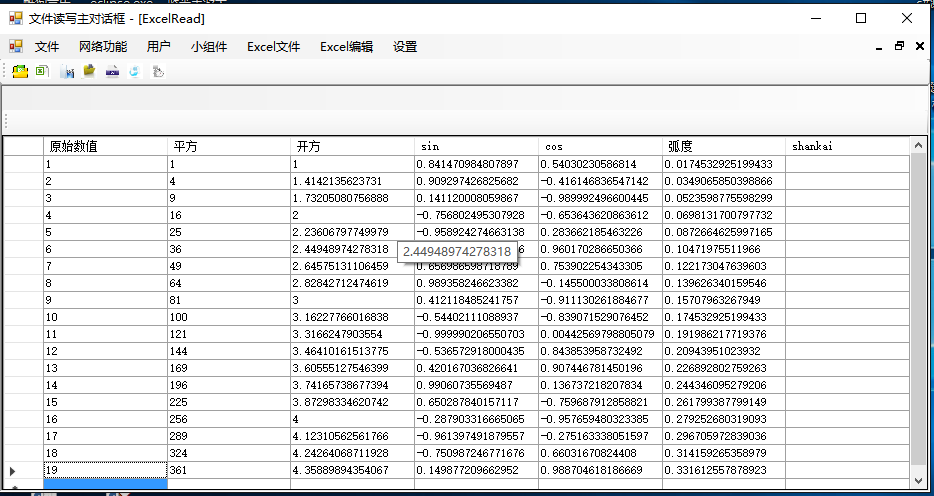


图 23 插入列后

1. **删除行、列**

删除鼠标选中的单元格所在的行或者列。

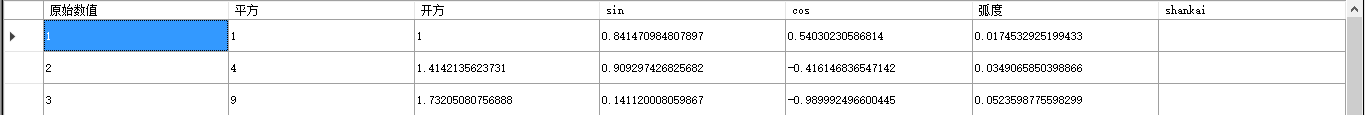


图 24 删除行前



图 25 点击菜单



图 26 删除行后



图 27 删除列前

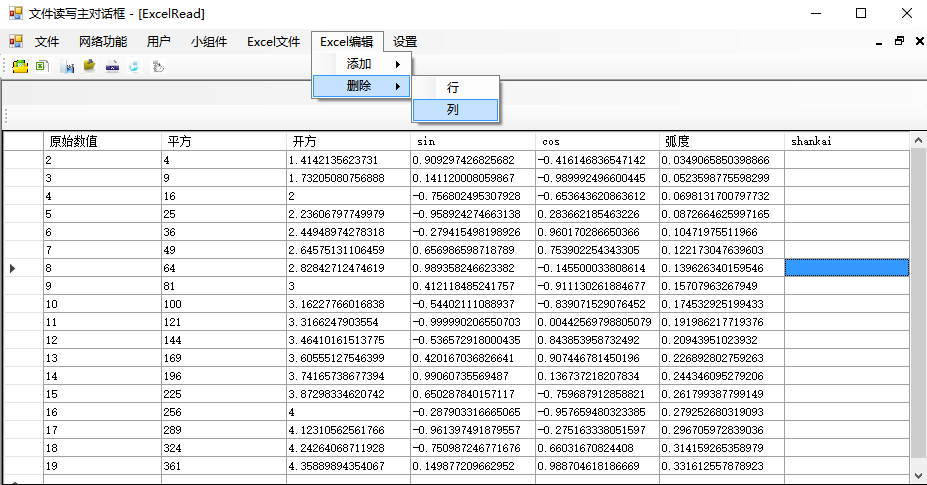


图 28 点击菜单

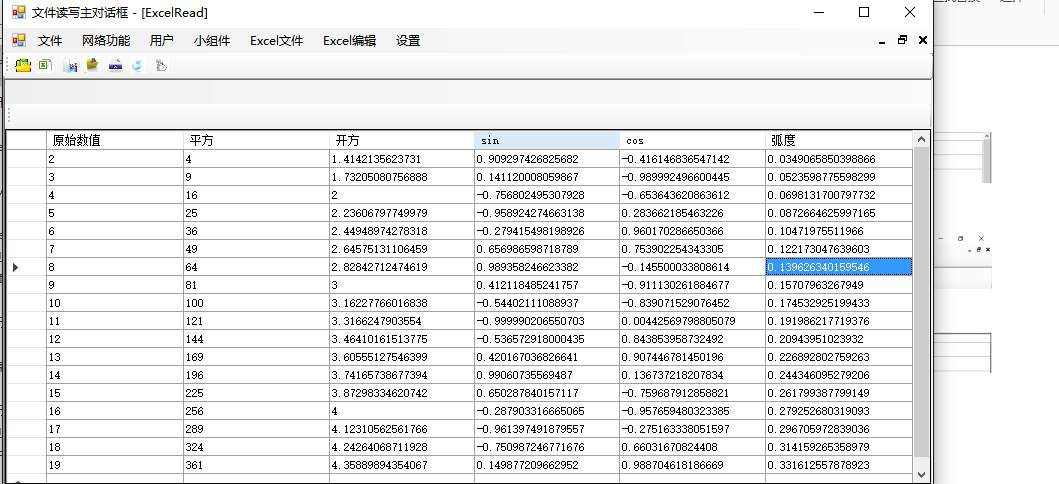


图 29 删除列后

1. **修改列名**

选中一个单元格，在此基础上右键，会出现右键菜单，点击修改列名，弹出修改列名对话框，填写新列名，具体流程如图：

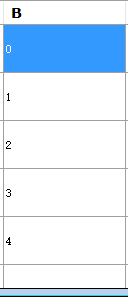
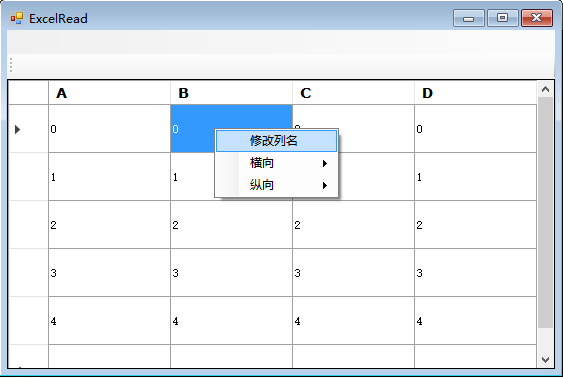
** **

图 30 修改之前 图 316 右键菜单

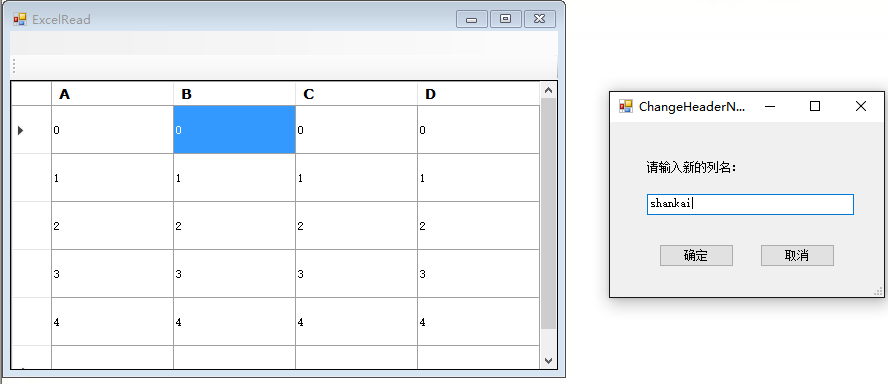
****

图 32 填写列名 图 33 修改之后

**声明：4）-6）较为简单，只是调用datagridview本身的成员函数就可以解决，所以不写出具体的算法。**

1. **横向求和、求平均**
2. 用户选中一行数据，右键菜单，点击“横向”->“求和”
3. 判断用户所选是否在同一行，如果不在，提示出错
4. 如果在同一行，看用户选中的数据是不是int行或double型，如果不是，报错
5. 计算出结果
6. 为控件添加一列名为“和”，用来显示数据，如果已经存在同名列，提示用户是否在该列显示数据，如果用户选择“否”，用消息框显示结果
7. 如果选择“是”，就在该列对应行显示结果。

具体流程如下图所示：

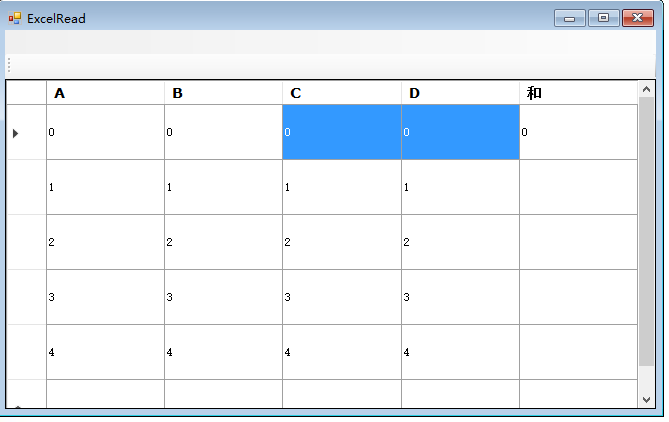
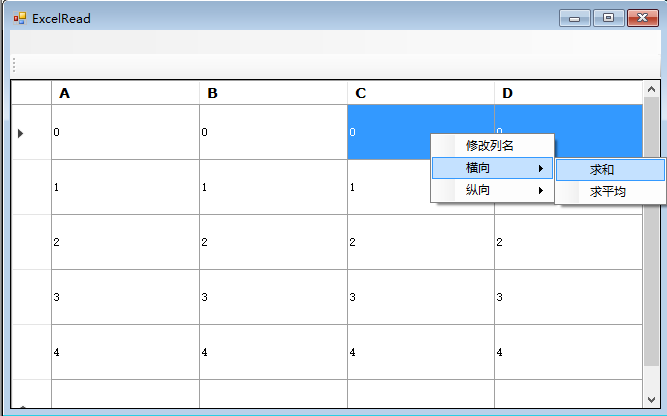


图 34 横向求和菜单 图 35 结果

1. **纵向求和、求平均**
2. 便利所有列，如果该列是数值类型的，那么就对该列求和，求平均数
3. 新添加两行，第一行存放对应列的和，第二行存放对应列的平均数

具体流程如下图所示：

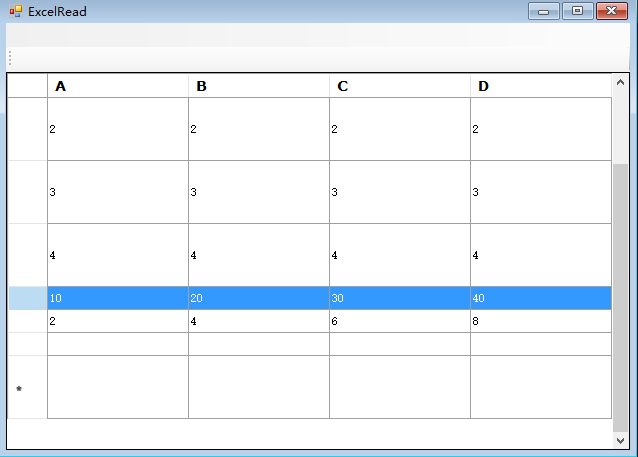
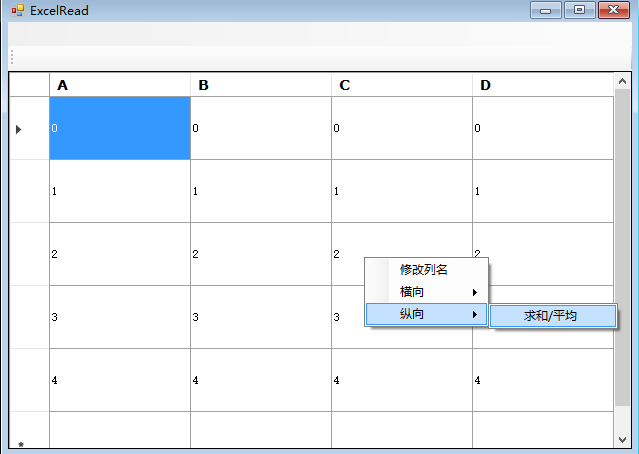


图 36 纵向求和求平均

1. **切换表单**
2. 当用户打开的工作簿中有多张表的时候，提示用户该工作簿有多张表，默认打开第一张；
3. 用户可以选择表单进行切换

具体流程如下图所示：

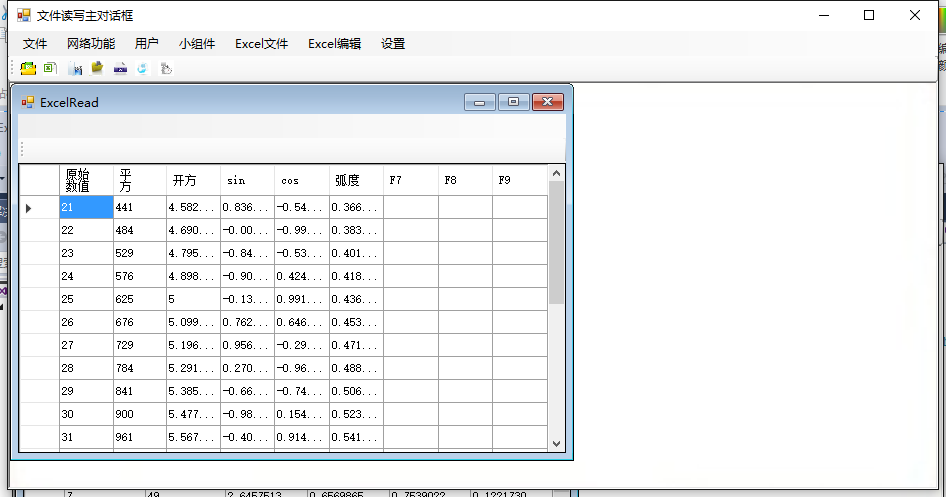
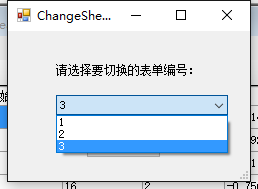
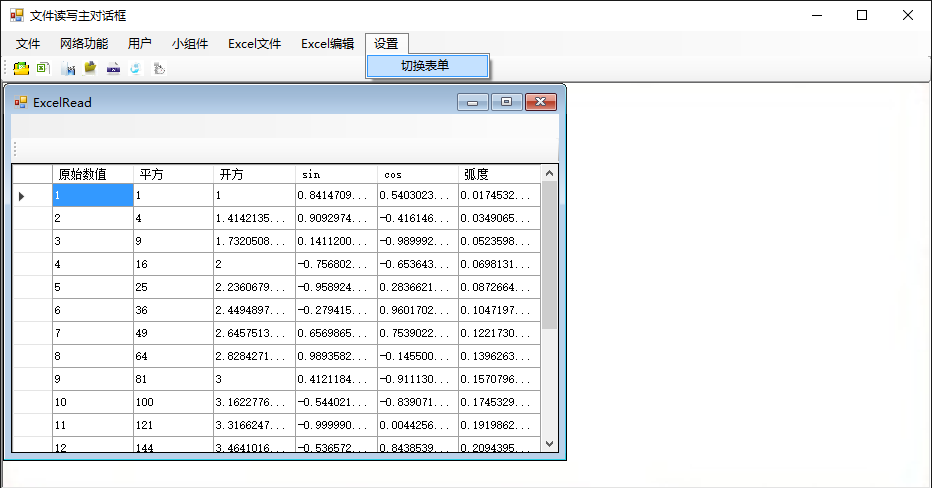
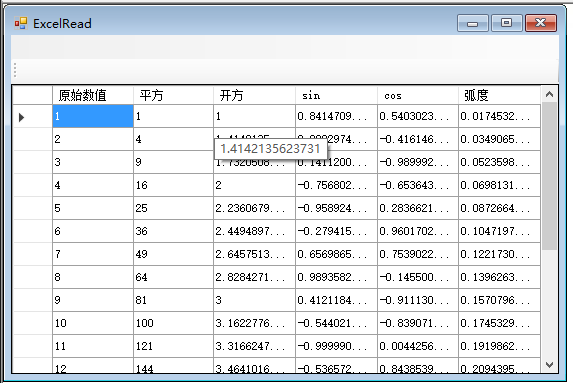
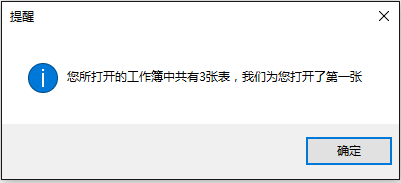
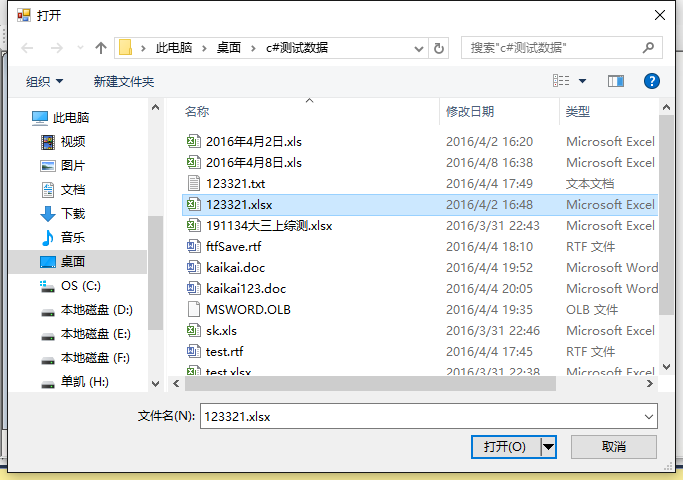
****

图 37 切换表单

1. **细节问题补充**

不足：在纵向求和的时候默认列的类型是一致的，如果不一致的话，很容易就抛异常了。

1. **视频文件**

主要是对windows media player的简单应用，展示如下：

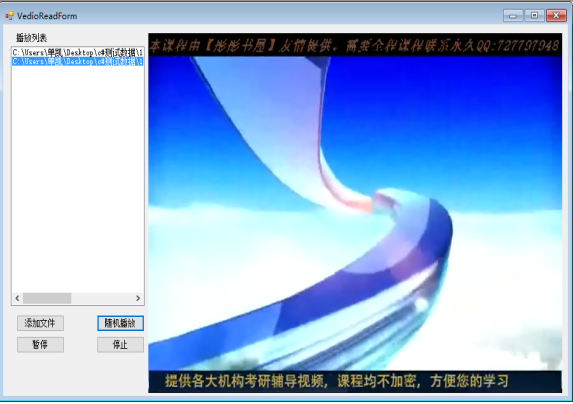
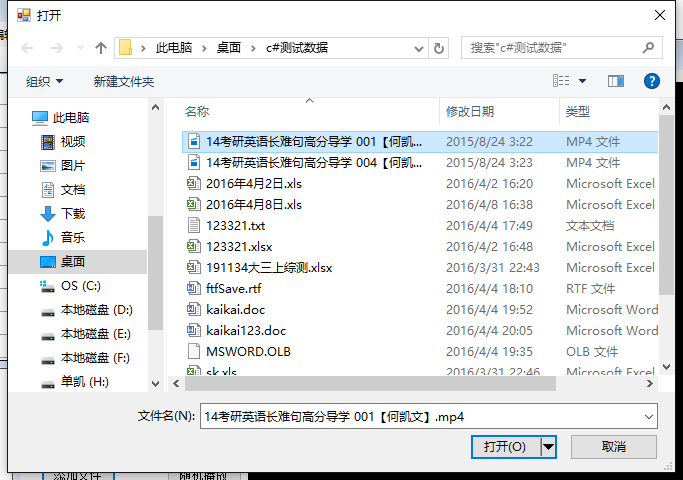
****

图 38 视频文件

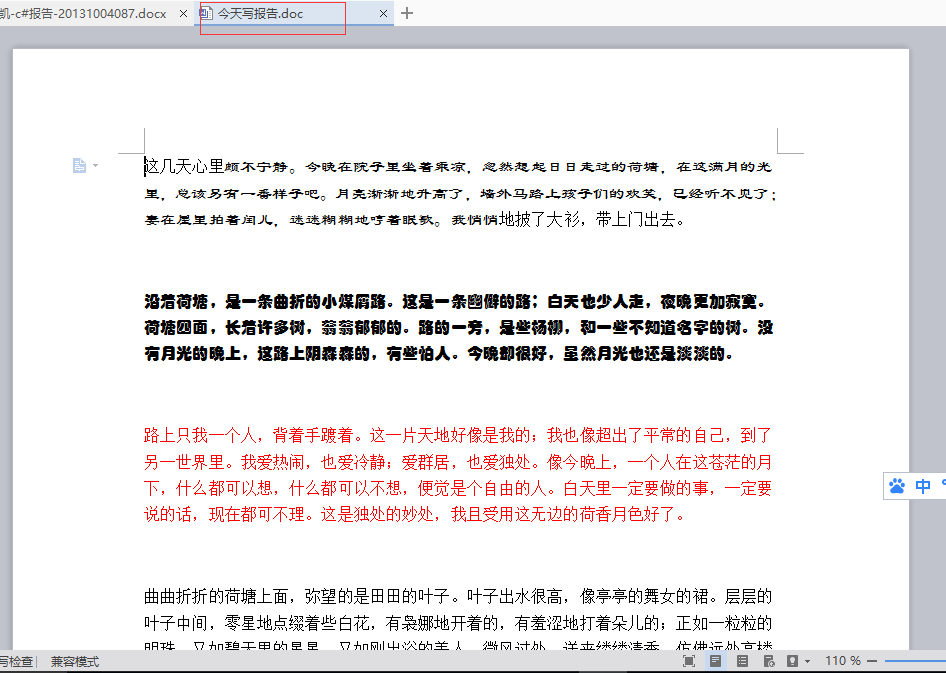
1. **文本文件**

主要是对Richtextbox，colordialog，和fontdialog的使用。



图 39 文本文件

1. **创建word**
2. 创建Application类对象
3. 创建Doc类对象
4. 调用worddoc.SaveAs函数保存word文档
5. 关闭doc对象
6. Application对象退出
7. **保存word**
8. 创建Application类对象
9. 创建Doc类对象
10. 调用Open方法打开文件
11. 将richtextbox中的数据复制到剪贴板
12. 粘贴到Word文档
13. 调用Save方法保存。



**图 40 另存为doc结果**

1. **用反射技术写一个插件**

要实现的功能很简单，就是在文本文件编辑的窗体上根据所导入的dll文件动态的在“插件”。插件算法流程图见**⑪。**

1. **网络功能**

**这个就是一个发送邮件的功能。**

**算法：**

这个c#有现成mail命名空间，只是练习一下使用。算法如下：

1. MailAddress from = new MailAddress(txtUserName.Text);
2. MailAddress to = new MailAddress(txtReceive.Text);
3. MailMessage message = new MailMessage(from, to);
4. 填写message的各个属性值

//主题

message.Subject = txtSubject.Text.Trim();

//主题的编码方式

message.SubjectEncoding = Encoding.UTF8;

//邮件内容

message.Body = txtBody.Text;

//邮件内容的编码方式

message.BodyEncoding = Encoding.UTF8;

1. SmtpClient client = new SmtpClient("smtp." + from.Host);
2. client.Send(message);

效果图：

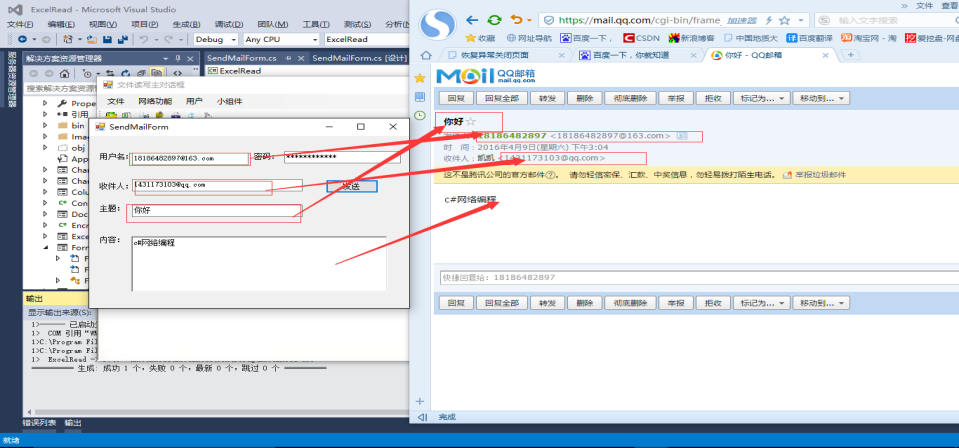
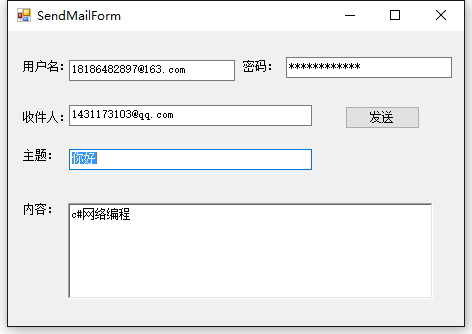


图 41 发送邮件效果

1. **小组件**

* **一个用WPF写的简单的秒表程序。**

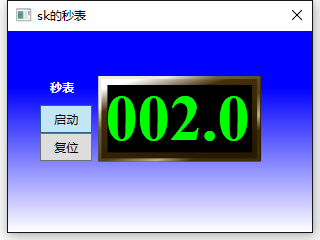


图 42 秒表

当鼠标悬停在“启动”按钮上是，触发click时间，时间++；

* **WPF写的图片效果展示程序**

图片有四种显示效果：圆形展开、随机方块、交错扫描和水帘效果。

**效果示意图如下：**

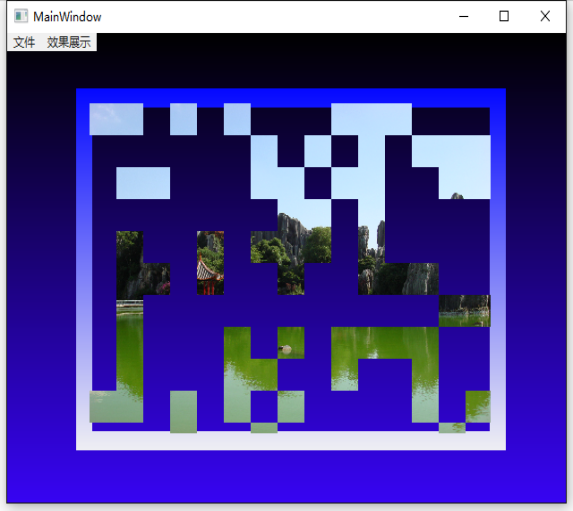
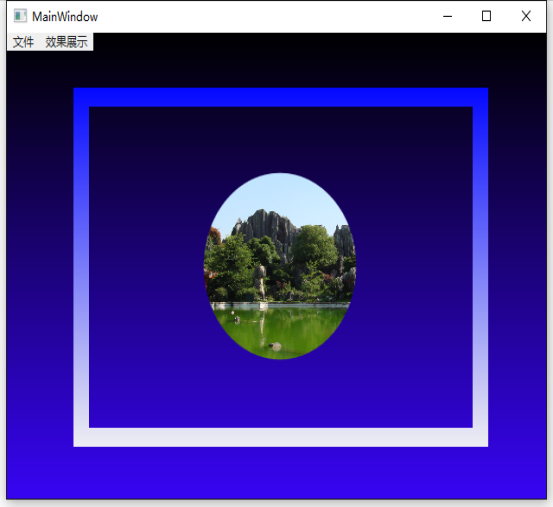


图 43 圆形展开 图 44 随机方块

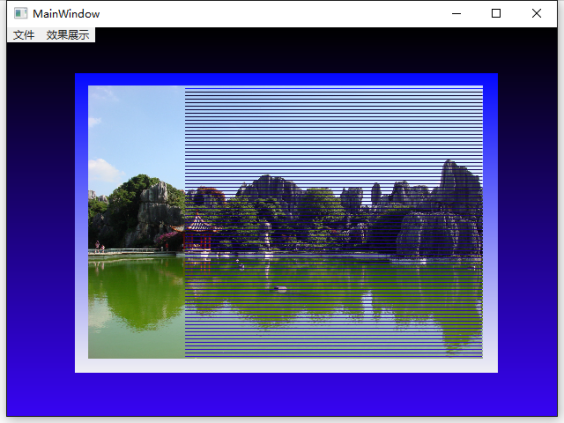
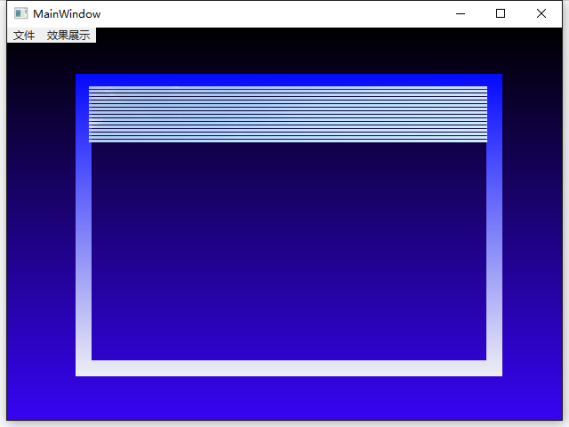


图 45 交错扫描

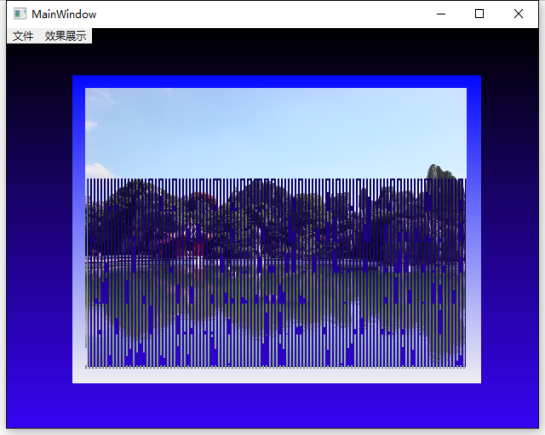
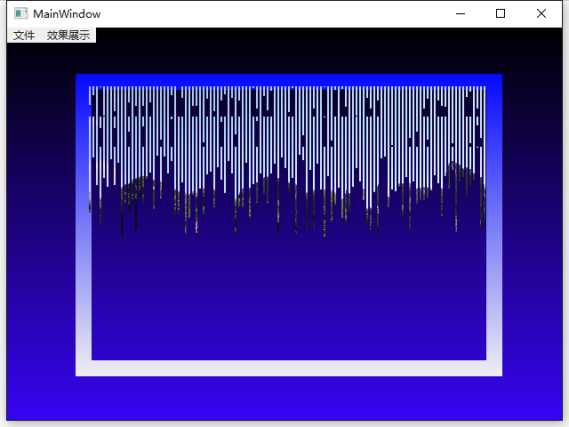


图 46 水帘效果

**算法流程框图如下：**

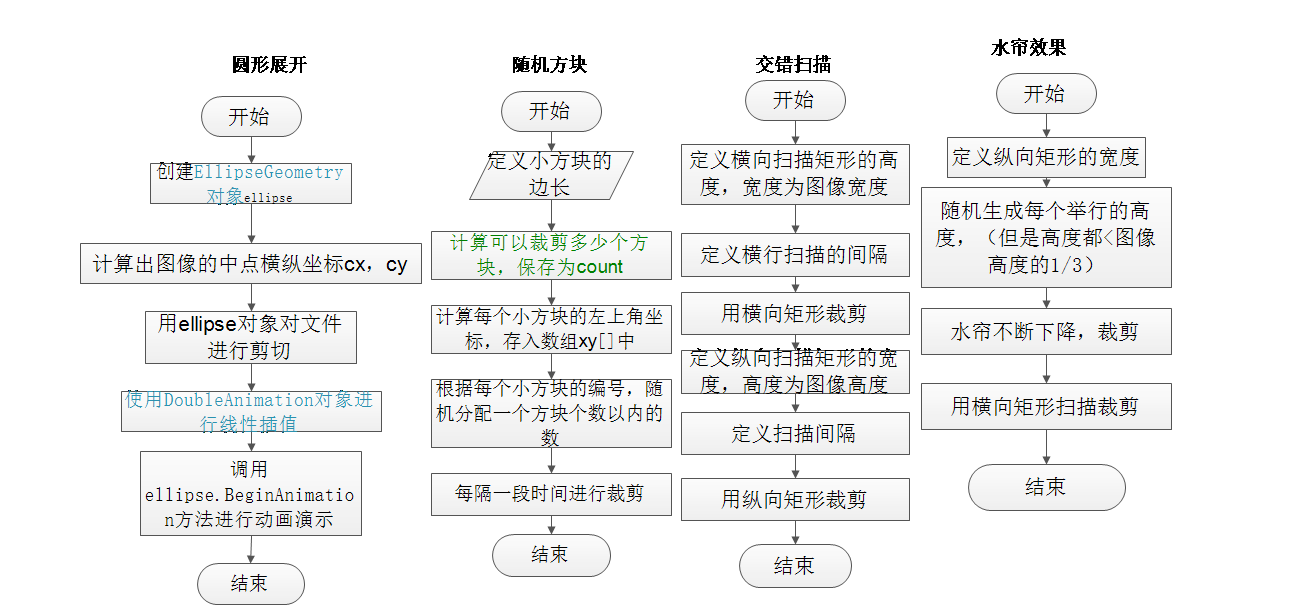


图 47 效果框图

**声明：以上两个wpf程序参考书籍《WPF与silverlight》**

* **文件管理模块**

这部分在树状图中显示本地磁盘信息，在列表视图控件中显示选择的磁盘中的文件（夹）信息，提供搜索文件（夹）功能，还实现了文件和文件夹的新建、复制、粘贴，删除等功能。最后还实现了图标、平铺等四种显示效果和清理垃圾文件的功能。

声明：这部分完全是参照《c#典型模块开发》中的一个例题，仅供扩展学习之用，自己有没有做改动。有些地方也没有搞懂。但是也确实学到了一些c#处理文件和文件夹的方法和技巧。

代码可在源程序中查看，下面是一些运行的结果图片：

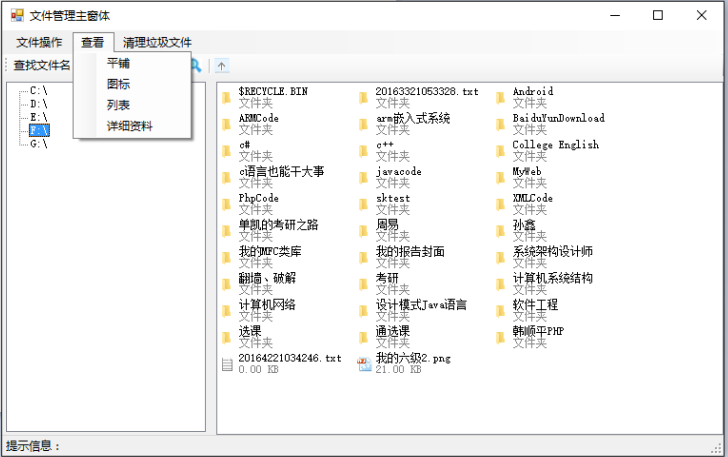


图 48 4种显示效果

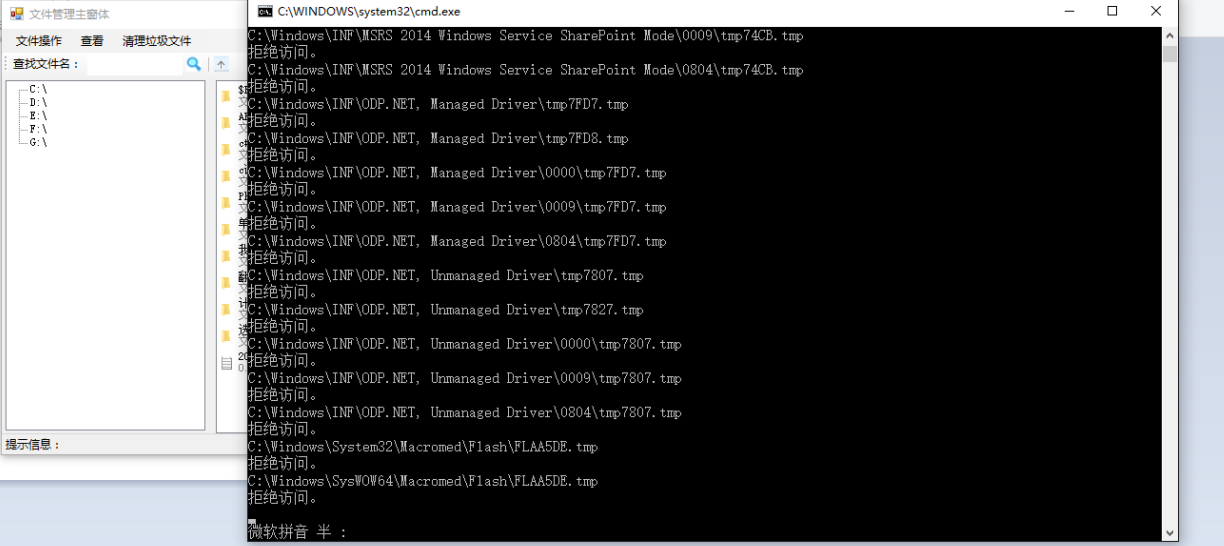


图 49 调用bat文清理垃圾

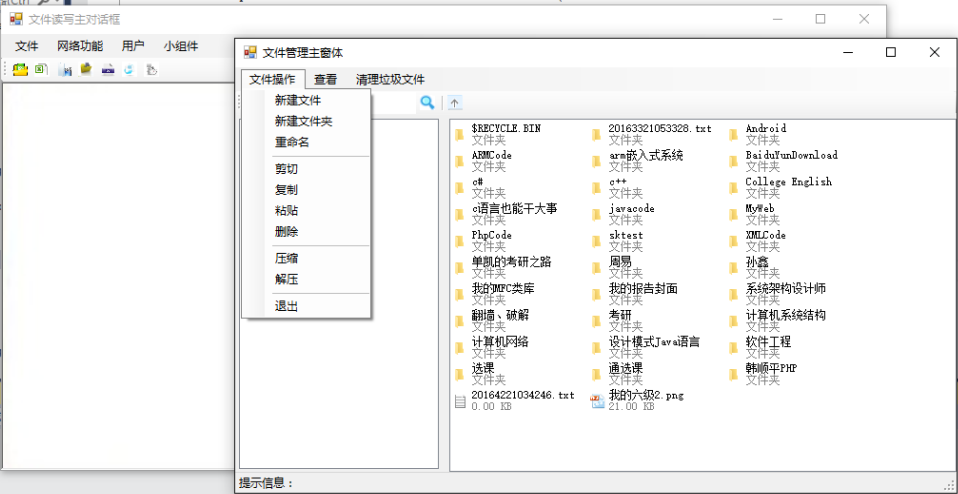


图 50 一些基本操作

1. **其它**

最小化托盘控件



图 51 最小化到托盘

双击图标，会还原为正常状态，右键可以弹出菜单。只要按照控件要求编写代码即可。

1. **软件总的流程**

****

图 52 软件流程

1. **压缩文件**

**压缩文件流程图**



图 53 压缩文件流程

解压缩的流程与此类似，可直接看代码（在“系统实现”中的“⑥”）

**效果示意图：**

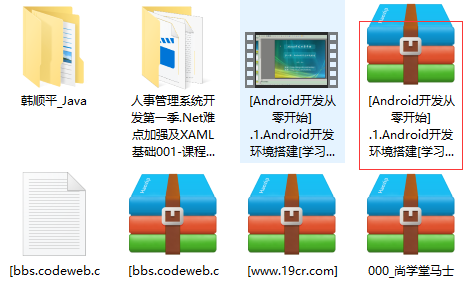
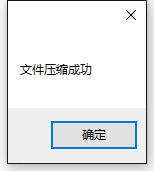
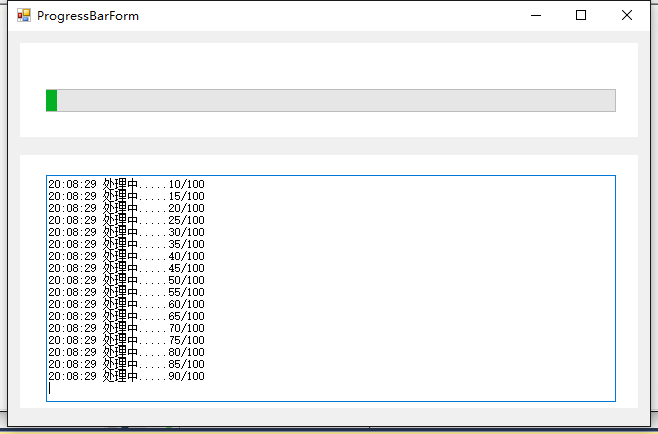
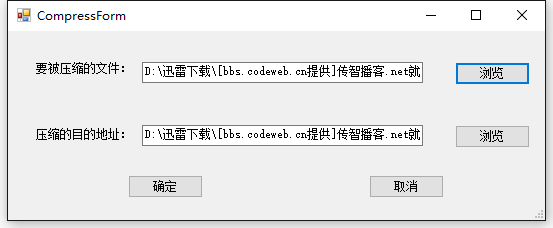


图 54 压缩文件示意图

**⑪ 在文本文件编辑的窗体中添加“插件”**

**算法流程框图如下所示：**

****

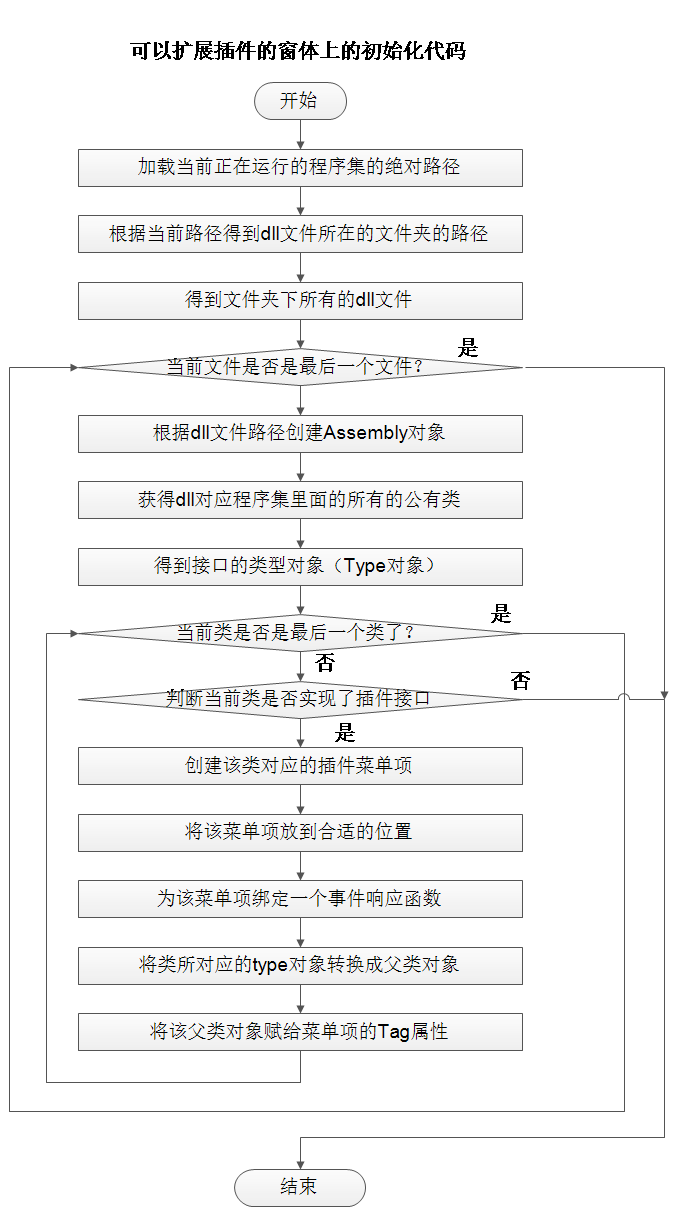
****

图 55 插件算法

**效果示意图：**

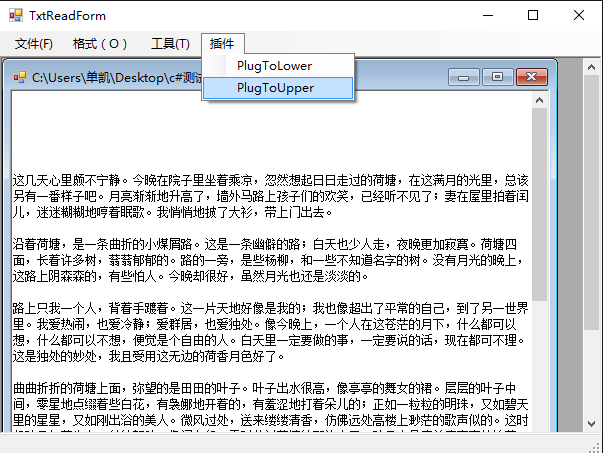
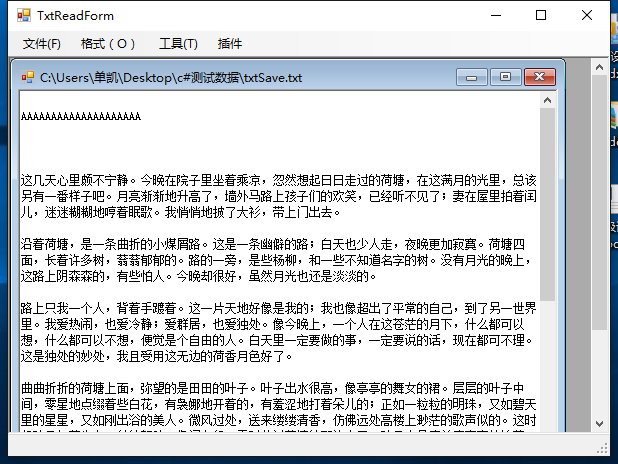
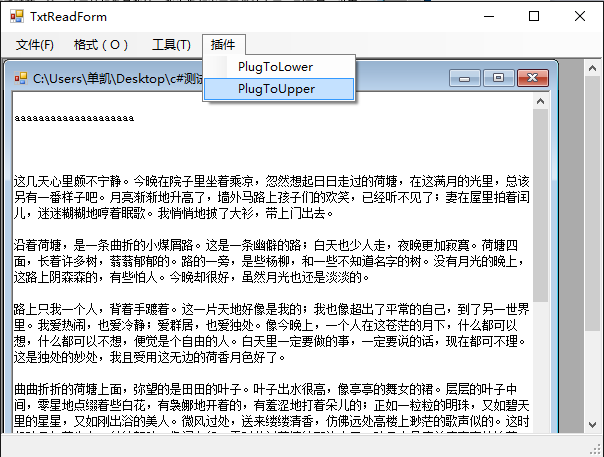


图 56 动态生成了菜单并演示了效果

**⑫ 图像旋转算法**

**流程框图：**

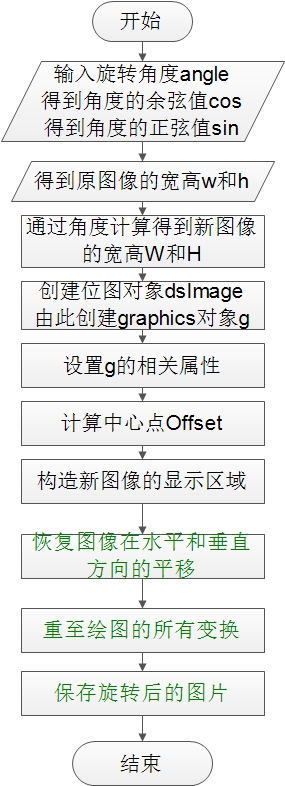
****

图 57 图象旋转流程图

**效果示意图：**

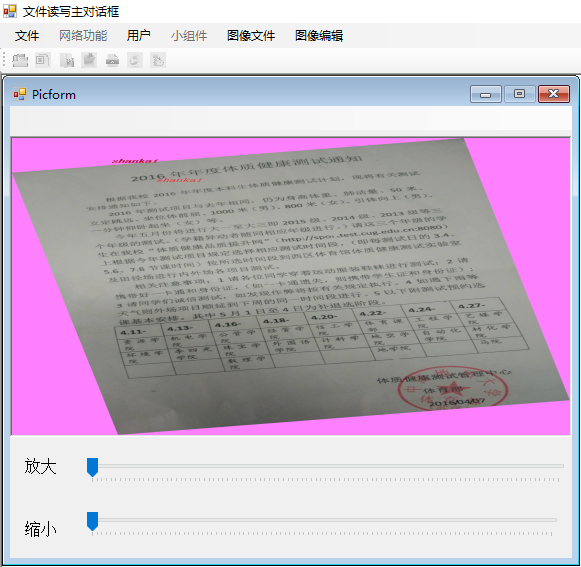
****

图 58 图像旋转效果

1. **系统实现**

**关键部分及代码：**

1. **图片旋转代码**

/// <summary>

/// 以逆时针为方向对图像进行旋转

/// </summary>

/// <param name="b">位图流</param>

/// <param name="angle">旋转角度[0,360](前台给的)</param>

/// <returns></returns>

public Image RotateImg(Image b, int angle)

{

angle = angle % 360;

//弧度转换

double radian = angle \* Math.PI / 180.0;

double cos = Math.Cos(radian);

double sin = Math.Sin(radian);

//原图的宽和高

int w = b.Width;

int h = b.Height;

int W = (int)(Math.Max(Math.Abs(w \* cos - h \* sin), Math.Abs(w \* cos + h \* sin)));

int H = (int)(Math.Max(Math.Abs(w \* sin - h \* cos), Math.Abs(w \* sin + h \* cos)));

//目标位图

Bitmap dsImage = new Bitmap(W, H);

System.Drawing.Graphics g = System.Drawing.Graphics.FromImage(dsImage);

g.InterpolationMode = System.Drawing.Drawing2D.InterpolationMode.Bilinear;

g.SmoothingMode = System.Drawing.Drawing2D.SmoothingMode.HighQuality;

//计算偏移量

Point Offset = new Point((W - w) / 2, (H - h) / 2);

//构造图像显示区域：让图像的中心与窗口的中心点一致

Rectangle rect = new Rectangle(Offset.X, Offset.Y, w, h);

Point center = new Point(rect.X + rect.Width / 2, rect.Y + rect.Height / 2);

g.TranslateTransform(center.X, center.Y);

g.RotateTransform(360 - angle);

//恢复图像在水平和垂直方向的平移

g.TranslateTransform(-center.X, -center.Y);

g.DrawImage(b, rect);

//重至绘图的所有变换

g.ResetTransform();

g.Save();

g.Dispose();

//保存旋转后的图片

b.Dispose();

dsImage.Save("FocusPoint.jpg", System.Drawing.Imaging.ImageFormat.Jpeg);

return dsImage;

}

1. **连接数据库**

string sql = string.Format("select count(\*) from [tb\_User] where name='{0}'and pwd='{1}'",

userName, MiPwd);

SqlConnection connection = ConnDataBase.connection;

try

{

connection.Open();

SqlCommand command = new SqlCommand(sql, connection);

//执行查询语句，发挥匹配的行数

int num = (int)command.ExecuteScalar();

if (num > 0)

{

frmLogin.Close();

//获得相应权限

HavePermission();

//保存到当前用户

CurUser = userName;

if (!frmLogin.DicNameCode.ContainsKey(frmLogin.UserName)

&& frmLogin.CkRememberNameCode.Checked == true)

{

frmLogin.WriteNameCodeToXml(frmLogin.UserName, frmLogin.PassWord);

}

MessageBox.Show("恭喜您，成功登录", "信息提示",

MessageBoxButtons.OKCancel,

MessageBoxIcon.Information);

}

1. **打开excel文件**

**//将excel文件当做数据源，与数据库连接类似**

private DataSet GetExcelData(string str, string sheetNum)

{

string strCon = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + str + ";Extended Properties='Excel 12.0;HDR=YES;IMEX=1;'";

OleDbConnection myConn = new OleDbConnection(strCon);

string strCom = "SELECT \* FROM " + sheetNum;

myConn.Open();

OleDbDataAdapter myCommand = new OleDbDataAdapter(strCom, myConn);

DataSet myDataSet = new DataSet();

myCommand.Fill(myDataSet, sheetNum);

myConn.Close();

return myDataSet;

}

1. **发送邮件**

//不使用默认凭证

client.UseDefaultCredentials = false;

//指定用户名和密码

client.Credentials = new NetworkCredential(from.Address, password);

//邮件通过网络发送到服务器

client.DeliveryMethod = SmtpDeliveryMethod.Network;

try

{

client.Send(message);

}

1. **通过xaml编写界面**

<Grid.Background>

<!--渐变画刷-->

<!--渐变的二维终止坐标, 起始坐标-->

<LinearGradientBrush EndPoint="1,1" StartPoint="0,0">

<!--此渐变停止点沿渐变向量的相对位置。默认值为 0.0。-->

<GradientStop Color="Blue" Offset="0.279"/>

<GradientStop Color="White" Offset="1"/>

</LinearGradientBrush>

</Grid.Background>

1. **解压文件关键代码**

FileStream sourceFile = File.OpenRead(path);

FileStream destinationFile = File.Create(dest);

//因为压缩文件大小未知，使用任意大小的缓冲区

byte[] buffer = new byte[4096];

int n;

using (GZipStream input = new GZipStream(sourceFile,

CompressionMode.Decompress))

{

n = input.Read(buffer, 0, buffer.Length);

destinationFile.Write(buffer, 0, n);

}

1. **Xml文件编程**

XmlDocument xmlDoc = new XmlDocument();

xmlDoc.Load(@"..\..\UserList.xml");

XmlNode rootNode = xmlDoc.SelectSingleNode("UserList");

XmlElement xe = xmlDoc.CreateElement("User");

XmlElement nameNode = xmlDoc.CreateElement("Name");

nameNode.InnerText = name;

XmlElement codeNode = xmlDoc.CreateElement("Pwd");

codeNode.InnerText = pwd;

xe.AppendChild(nameNode);

xe.AppendChild(codeNode);

rootNode.AppendChild(xe);

xmlDoc.Save(@"..\..\UserList.xml");

**⑧ 在图片上添加文字**

private void pictureBox1\_MouseDoubleClick(object sender, MouseEventArgs e)

{

if (this.pictureBox1.Image.Width < 1)

{//基本的判断还是要的

return;

}

//要进行处理的图片对象

Graphics gra = Graphics.FromImage(this.pictureBox1.Image);

//初始化画笔

SolidBrush brush = new SolidBrush(Color.Red);

gra.DrawString(this.strAddToPic,this.font, brush, e.X,

e.Y);//处理图片

//对显示图片的容器里面的内容进行刷新，以便及时显示添加的文字

this.pictureBox1.Refresh();

}

**⑨ 保存图像**

/// <summary>

/// 保存图像文件

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

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

{

saveFileDialog.Filter = "JPeg Image|\*.jpg|Bitmap Image|\*.bmp|Gif Image|\*.gif|PnG Image|\*.png|Wmf Image|\*.wmf";

saveFileDialog.FilterIndex = 0;

if (pictureBox1.Image == null)

{

MessageBox.Show("没有预览图片");

}

else if (saveFileDialog.ShowDialog() == DialogResult.OK)

{

if (pictureBox1.Image != null)

{

pictureBox1.Image.Save(saveFileDialog.FileName,

System.Drawing.Imaging.ImageFormat.Jpeg);

}

}

}

**⑩ 进度条控件和多线程**

//进度条显示线程

progressThread = new Thread(dispayProgress);

progressThread.IsBackground = true;

progressThread.Start();

//进度条控件的相关设置

frmBar.MyProgressBar.Minimum = 0;

frmBar.MyProgressBar.Maximum = 100;

frmBar.MyProgressBar.Value = 0;

frmBar.output("处理中....."+ frmBar.MyProgressBar.Value.ToString()+"/100");

frmBar.MyProgressBar.Value += 5;

frmBar.MyProgressBar.Invalidate();

**⑪ 验证码：**

/// <summary>

/// //画验证码

/// </summary>

private void DrawVerificationCode()

{

Random r = new Random();

string str = null;

for (int i = 0; i < 5; i++)

{

int rNum = r.Next(0, 10);

str += rNum;

}

//创建GDI对象

Bitmap bmp = new Bitmap(120, 30);

Graphics g = Graphics.FromImage(bmp);

for (int i = 0; i < 5; i++)

{

Point p = new Point(i \* 20, 0);

string[] fonts = { "微软雅黑", "宋体", "仿宋", "华文行楷", "隶书" };

Color[] colors = { Color.Blue, Color.Brown, Color.Chocolate, Color.DarkOrange, Color.Red };

g.DrawString(str[i].ToString(),

new Font(fonts[r.Next(0, 5)], 20, FontStyle.Bold),

new SolidBrush(colors[r.Next(0, 5)]),

p);

}

//画一些杂乱的线条

for (int i = 0; i < 15; i++)

{

Point p1 = new Point(r.Next(0, bmp.Width), r.Next(0, bmp.Height));

Point p2 = new Point(r.Next(0, bmp.Width), r.Next(0, bmp.Height));

g.DrawLine(new Pen(Brushes.Green), p1, p2);

}

//画一些杂乱的点

for (int i = 0; i < 500; i++)

{

Point p = new Point(r.Next(0, bmp.Width), r.Next(0, bmp.Height));

bmp.SetPixel(p.X, p.Y, Color.Black);

}

//将图片镶嵌到picturebox种

pbVerificationCode.Image = bmp;

}

**⑫ 补充说明：**

报告上的代码只是关键部分，和程序流程框图互补，画了流程框图的基本没有粘贴代码，可以到源程序ExcelRead中查看代码；太简单的也没有画流程框图，直接粘贴代码。

1. **遇到的问题以及解决方案**
2. **对象已经释放了,程序代码又去访问它**

**问题现象：**

抛出的异常如下图所示：

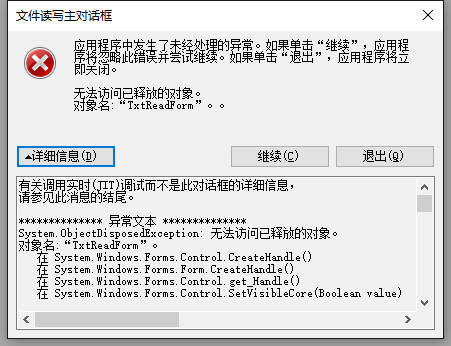


图 59 访问已释放对象异常

**问题原因：**

//单身模式

tatic TxtReadForm mySelf = null;

public static TxtReadForm GetInstance()

{

if(mySelf==null)

{

mySelf = new TxtReadForm();

}

return mySelf;

}

此处我采用单身模式，只允许创建一个对话框的实例。但是在对话框关闭的时候mySelf并未置空，所以GetInstance总是得到同一个引用，所以会出现如上错误。

**结决方案：**

private void TxtReadForm\_FormClosing(object sender, FormClosingEventArgs e)

{

mySelf = null;

}

**参考材料：**

[1] http://q.cnblogs.com/q/3054/

1. **邮件发送失败异常**

**问题现象：**

异常情况如下图所示：

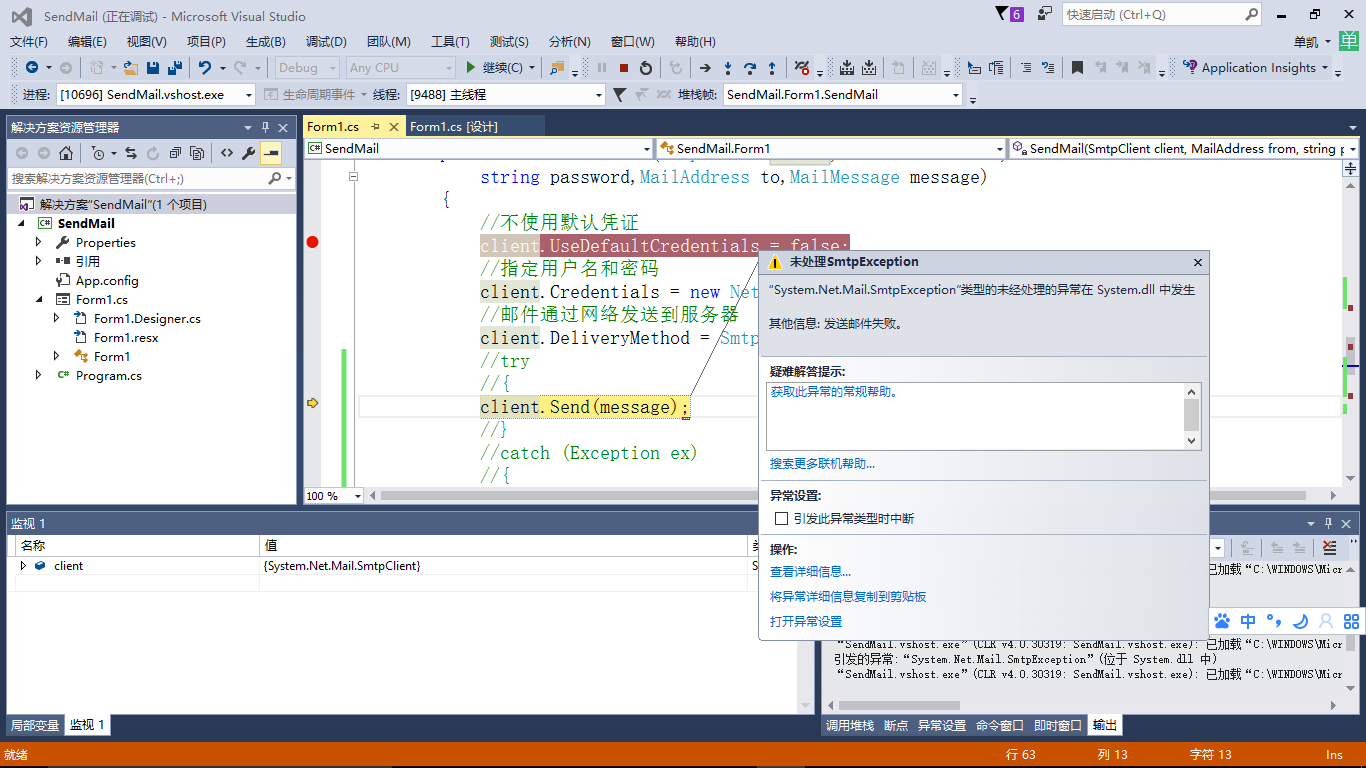


图 60 邮件返送失败异常

**问题原因：**

所用的163邮箱不允用第三方的客户端。

**解决方案：**

****

图 61 设置163邮箱

****

图 62 开启SMTP服务

**参考材料：**

老师帮助。

1. **Mdi容器默认的灰色背景太丑了。**

**问题现象：**

如下图所示：



图 63 MDI背景

**解决方案：**

将窗体的BackgroundImage设置为你想要的背景图片即可。 如果窗体是Mdi容器，则会自动加载背景图片为Mdi容器背景。

**参考材料：**

[1] <http://blog.csdn.net/yunhaic/article/details/6460997>

1. **进程杀不死**

**问题现象：**

当主窗口退出时，调用Process.Kill()函数来结束进程，但是，并没有结束进程。

**问题原因：**

我把Kill进程的代码写在主窗体的Leave事件响应函数中，但是经过调试得知，程序并没有执行到Leave函数。

**解决方案：**

将杀死进程的函数写在FormClosing事件响应函数当中。formClosing实在窗体关闭之前执行，但是Leave是在窗体关闭之后执行。

**参考材料：**

Msdn

1. **当窗体放大后，但是空间还是在原来的位置**

**问题现象：**

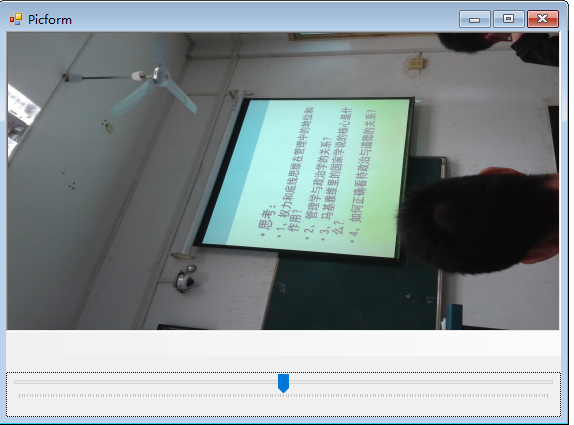


图 64 原始对话框大小

**解决过程：**

设置两个空间的anchor属性或dock属性，但是结果不能令人满意。

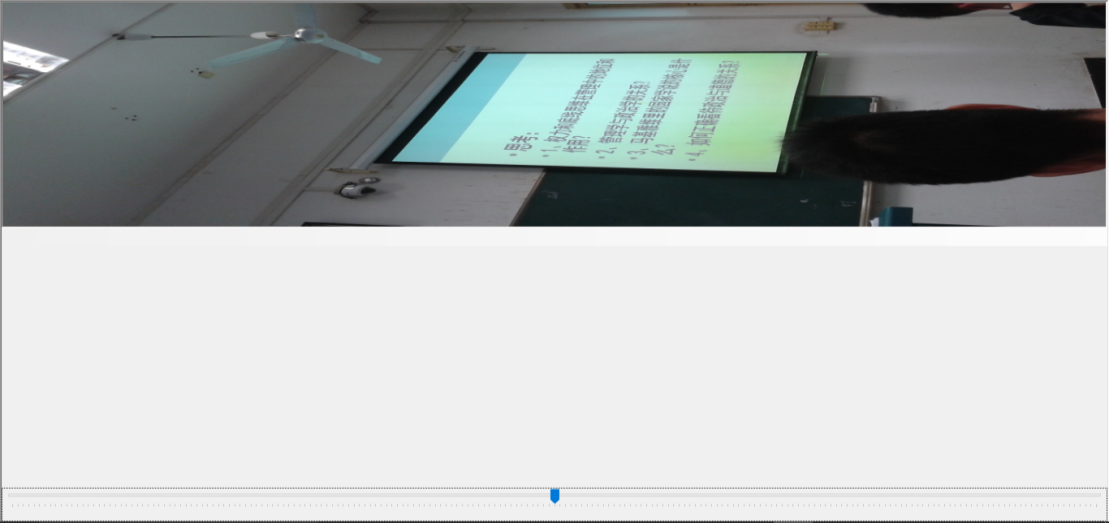


图 65 anchor属性：上中下

如上图所示：中间有大量留白，结果不能说是人满意。

如果将anchor设置为上、下、左、右的话，滑块控件会浮在图片的上面，效果也不能尽如人意。效果如下图所示：



图 66 属性上下左右

**解决方案：**

将窗体最大化属性设为false，不允许窗体大小发生改变。效果如下：

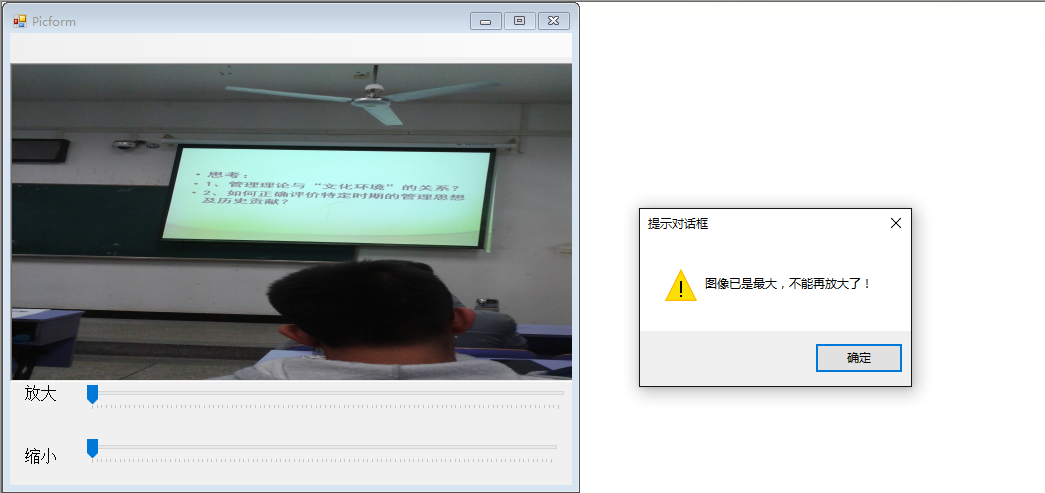


图 67 定死

**参考材料：**

**http://www.cnblogs.com/slyzly/articles/1965965.html**

1. **区分anchor属性和dock属性**

Anchor这个属性迫使控件将其自身定位在父窗体或父控件中的某个相对或绝对位置。这个属性有四个可以开启或关闭的值：

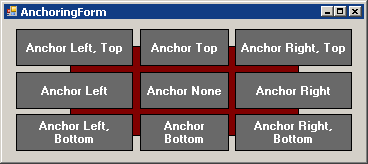
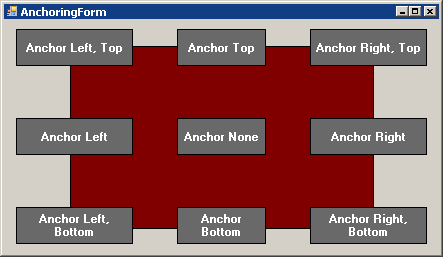
 

图 68 anchor属性示意图

Dock属性迫使控件紧贴父窗体（或控件）的某个边缘。虽然Anchor属性也可以实现这一点，但是dock属性使得你能够在父窗体中让子窗体可以在上方（或旁边）互相“堆叠”。如果某个子窗体改变了大小，其它停驻在它旁边的子窗体也会随之改变。

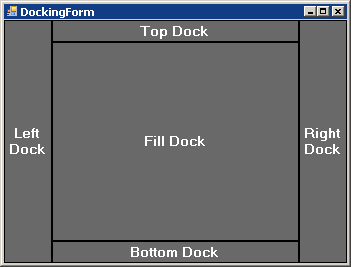
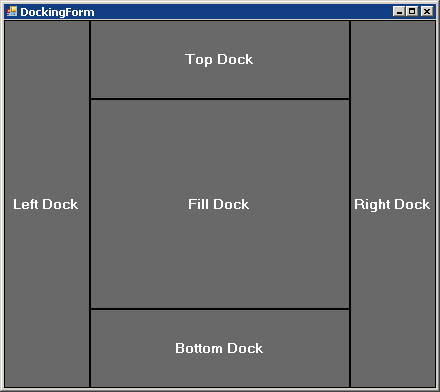
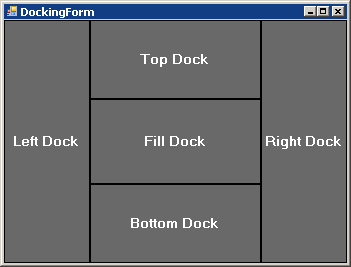


图 69 dock属性示意图

**⑦ 在图片中添加文字时，坐标位置发生偏移**

**问题现象：**

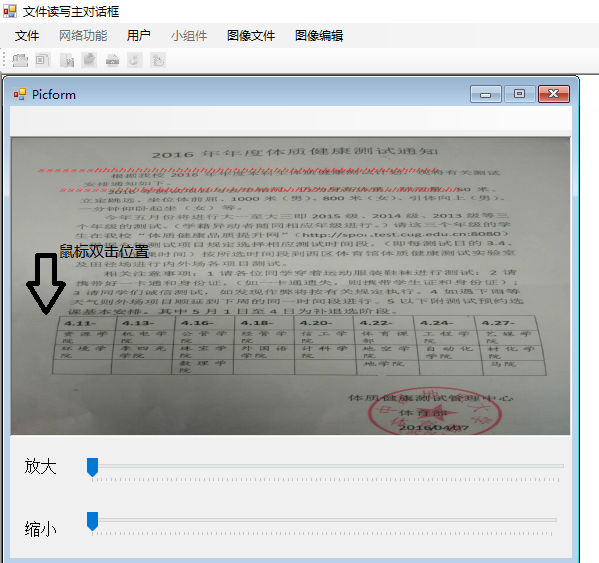


图 70 坐标偏移现象

解决方案：未解决

**⑧ 进度条不动**

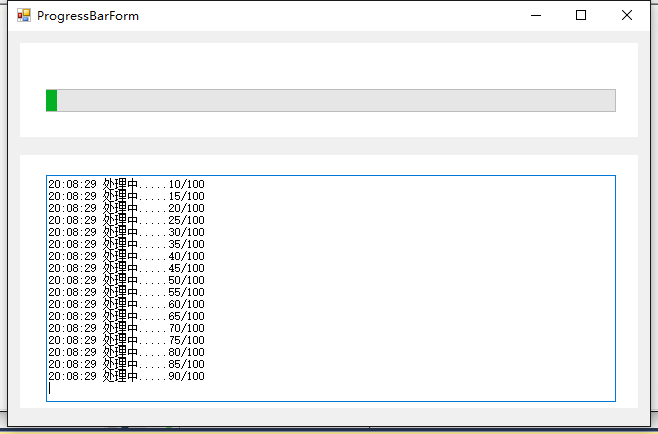
****

图 71 进度条问题

解决方案：未解决

**⑨ show（）和 showDialog（）的区别**

A.WinForm中窗体显示

显示窗体可以有以下2种方法：

Form.ShowDialog方法 （窗体显示为模式窗体）

Form.Show方法 （窗体显示为无模式窗体）

2者具体区别如下：

1.在调用Form.Show方法后,Show方法后面的代码会立即执行

2.在调用Form.ShowDialog方法后,直到关闭对话框后，才执行此方法后面的代码

3.当窗体显示为模式窗体时，单击“关闭”按钮会隐藏窗体，并将DialogResult属性设置为DialogResult.Cancel

与无模式窗体不同，当用户单击对话框的关闭窗体按钮或设置DialogResult属性的值时,不调用窗体的Close方法

实际上是把窗体的Visible属性赋值为false,隐藏窗体了

这样隐藏的窗体是可以重新显示，而不用创建该对话框的新实例

因为未关闭窗体,所以在应用程序不再需要该窗体时,请调用该窗体的Dispose方法

**参考材料：**

<http://blog.csdn.net/haibinglong/article/details/7311175>

**⑩ 数据库连接出错**

**问题现象：**

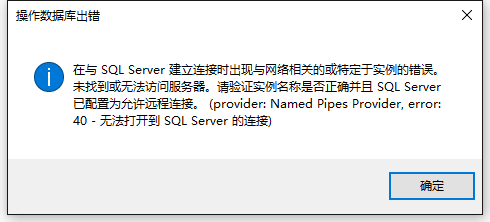
****

图 72 数据库连接出错

**问题原因：**

**Sqlserver数据库服务没开**

**解决方案：**

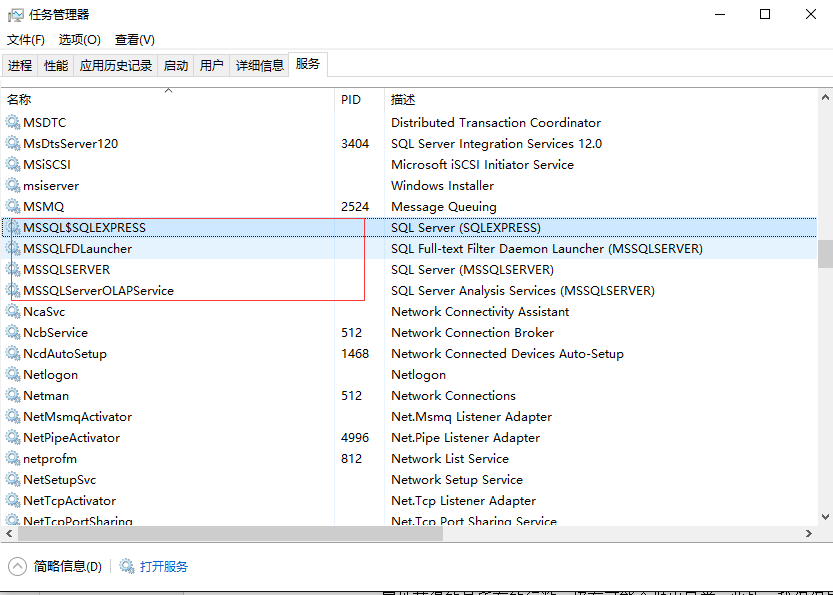
****

图 73 打开数据库的服务

鼠标右键，开始运行上图框出的服务即可。

**参考材料：**

将异常内容百度一下，很容易找到答案。

1. **完成的项目中存在的不足**
2. **有的地方只是简单的抛出了异常，保证程序不会因此而终止，并未做详细的处理。**

例如：在纵向求和求平均的时候，因为表格有值的行数不定，而程序

rows.Count属性获得的是所有的行数，极有可能会抛出异常，此处，我仅仅是捕获异常，并未好好解决。

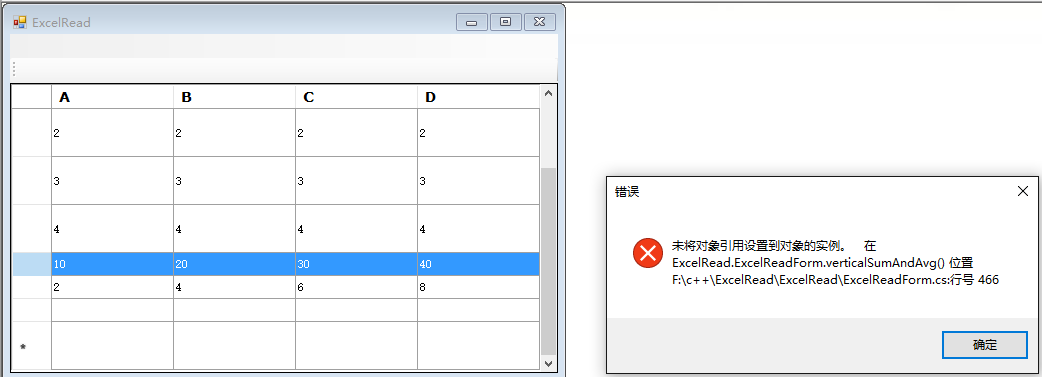
****

图 74 合计表格

1. **有的想做的功能因水平有限，未能实现**

例如：

想实现点击“叉号”就删除以前登陆过的联系人，但是并没有搞出来。

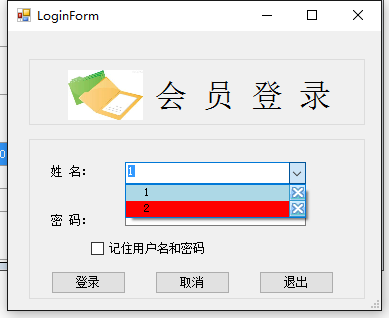


图 75 删除联系人

**③ 用户交互不是很好**

比如图象旋转的时候我直接指定了一次旋转10°，而这应该提供界面，让用户输入。

**④ 存在冗余和bug。**

1. **个人小结**
2. **本次作业所用到的知识点小结：**

* Ado.Net数据库连接
* c#的基本的类：string、file、path等
* Xml文件的创建、修改等
* 几种基本的对话框：openfiledialog，savefiledialog，fontdialog，colordial，Messagebox（作业22）等。
* 几种常件的控件：datagridview、combobox、listview、windows media player、

Toolbar（想当于作业15：使用工具栏控件（ToolBar）：工具栏上的铵钮同时显示图标和文件，这些铵钮可以执行新建、打开和保存文本文件，并可以对文本文件的内容进行剪切、复制和粘贴操作）、notifyicon（作业17）等。

* 反射
* 利用Office组件编程.word（作业20）、excel（作业21）。
* WPF
* DES加解密算法（相当于作业13.加解密文件：以文件流的方式用DES算法对文件进行加密、解密。）
* 调用和杀死其它进程（作业23、24）

**② C#程序设计个人总结**

5个星期的课程结束了,虽然忙碌但是过得还是很充实的,最重要的是自己对于编程语言又有了一个新的看法,总之,这次课程受益非浅.

在此次课程期间学到了好多知识,而这又不仅仅是专业知识的积累,更多是的那一种学习方法与态度的认识提高.

专业知识方面:

1. .系统学习和总结了数据库开发基础知识,如数据的绑定,数据的处理
2. .基本成功完成了作业,并且在csdn上得到了好多的参考学习
3. .对于c#这一门语言的熟透和认识提高了许多
4. .学到了开发中代码的重用技术,并且学会了自主学习使用新控件功能的学习方法.
5. 培养了我选用参考书，查阅手册及文献资料的能力。培养独立思考，深入研究，分析问题、解决问题的能力。

学习态度方面:

1. .学习是一具连续的过程,而不应该边学边玩,这样会降低学习的效率.

2).做事就要做好,就要坚持,这一点我是欠缺的,其中对此我已经得到了一具深刻切身的认识

3). 写程序的过程中要考虑周到，严密。在做设计的时候要有信心，有耐心，切勿浮躁。认真的学习课本知识，掌握课本中的知识点，并在此基础上学会灵活运用。

总之,这次做作业期间又让我体会到了充实的快乐,和学习的一种乐趣.但是这个过程也很痛苦，看到自己解决不了的问题，还是感到很是心塞。但是我坚信：I do my best，god will do the rest。自己尽力就好。还有一点，平时的积累很重要，这次作业我就把很多平时写过的程序直接拷过来了，非常地省时省力。

1. **参考材料**

[1].http://bbs.csdn.net/topics/390123642

[2].http://www.tuicool.com/articles/ZRvMNjF

[3].http://blog.csdn.net/makenothing/article/details/26022855

[4].http://blog.csdn.net/yuekunge/article/details/9171301

[5].http://blog.csdn.net/liuxiuming/article/details/5199879

[6] 《深入浅出WPF》 刘铁猛著

[7] 《c#典型模块开发》中国铁道出版社 周峰、王征编著

[8] 课程PPT

等

**附录：**

**如有问题，请联系：tel：18186482897 QQ：1431173103.**

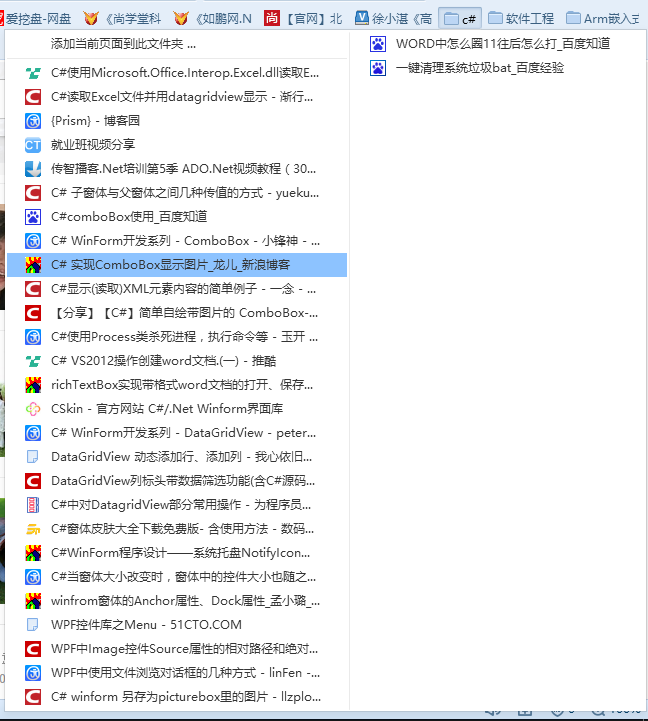
****

图 76 参考的网址截图

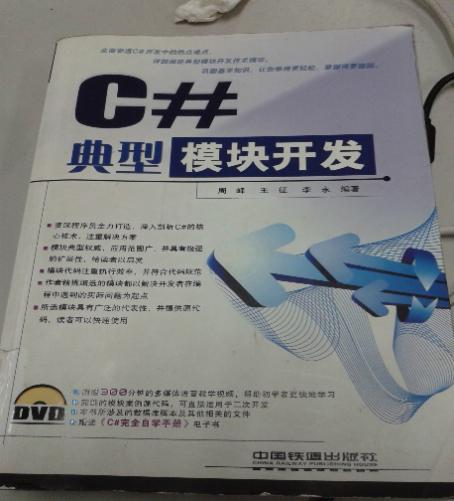
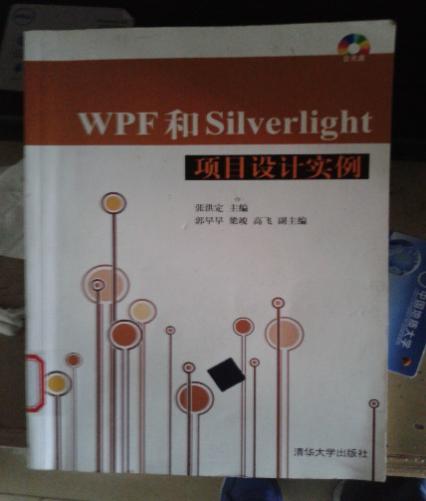
****

图 77 参考书籍截图