**程 序 综 合 实 验**

**实 验 报 告**

**班级： 软件172**

**学号： 174402**

**姓名： 吴帅龙**

**实验二 成绩管理程序设计与实现**

**一、实验目的与要求**

**实验目的：**锻炼面向对象编程与文件操作能力

**实验要求**：

1、熟悉VC++、Visual Studio c#或者Java编程环境。

2、运用面向对象知识对实验题目进行分析和设计。

3、进行程序编写和调试工作。

**二、实验内容**

 设计、实现一个具有通讯信息插入、修改、删除、显示、查询和统计功能的通讯录管理程序。

**三、实验设计**

1、实验分析（功能分析以及实现方案）

插入：再弹出一个Form,在新的Form中插入数据到txt文件中

删除：直接点击Delete按钮,删除txt文件中对应行的数据

查询：可以根据学号和姓名进行查询，找到后返回给前端页面

统计：用算法对txt文件中的各行信息(总成绩)进行排序，并对各科成绩做记录

2、数据结构设计和算法设计

查询：直接对txt文件进行循环遍历即可

统计：其中要对信息进行排序。可以使用冒泡排序，选择排序等算法。我使用了冒泡排序。

3、界面设计

和实验指导书中界面相似。不过我增加了一个统计的按钮和几个其他的Form.

**四、核心程序代码**

**//删除txt中的记录条目**

private void Delete(object sender, EventArgs e)

{

try

{

int index = this.listView1.SelectedItems[0].Index;

this.listView1.Items.RemoveAt(this.listView1.SelectedItems[0].Index);

string file = File.ReadAllText("G:\\scoremanagenment.txt", UTF8Encoding.Default);

string[] records = Regex.Split(file, "\r\n");

for(int i=0; i<records.Length; i++)

{

if(i != index)

{

continue;

}

records[i] = " ";

}

File.Delete("G:\\scoremanagenment.txt");

for (int i = 0; i < records.Length; i++)

{

if (records[i] != "" && records[i] != " ")

{

records[i] += "\r\n";

File.AppendAllText("G:\\scoremanagenment.txt", records[i], UTF8Encoding.Default);

}

}

}

catch

{

Form4 form4 = new Form4();

form4.Show();

}

}

//Display 显示

private void Display(object sender, EventArgs e)

{

if(!File.Exists("G:\\scoremanagenment.txt"))

{

File.WriteAllText("G:\\scoremanagenment.txt","");

}

string file = File.ReadAllText("G:\\scoremanagenment.txt", UTF8Encoding.Default);

string[] records = Regex.Split(file, "\r\n");

//开始更新视图

this.listView1.BeginUpdate();

//清空原有视图

this.listView1.Items.Clear();

// records.Length为数组的元素个数

for (int index = 0; index < records.Length; index++)

{ //分割每行记录的各个字段

string[] components = Regex.Split(records[index], "\t");

if(records[index].Trim().Length == 0)

{

continue;

}

//生成listview的一行

ListViewItem lvi = new ListViewItem(components);

//添加背景色

lvi.SubItems[0].BackColor = Color.Red;

//把新生成的行加入到listview中

this.listView1.Items.Add(lvi);

}

//视图更新结束

this.listView1.EndUpdate();

}

//Insert 添加数据

private void Insert(object sender, EventArgs e)

{

//利用string的加法操作构造新纪录，作为一行，写入txt文件中

string newrecord = this.textBox1.Text + "\t" + this.textBox2.Text + "\t" + this.textBox3.Text + "\t" + this.textBox4.Text + "\t" + this.textBox5.Text + "\t";

double Allgrade = (Convert.ToDouble(this.textBox3.Text) + Convert.ToDouble(this.textBox4.Text)+ Convert.ToDouble(this.textBox5.Text));

double Avgrade = Allgrade / 3;

//保留一位数字

newrecord += Allgrade.ToString("0.#") + "\t" + Avgrade.ToString("0.#") + "\r\n";

File.AppendAllText("G:\\scoremanagenment.txt", newrecord, UTF8Encoding.Default);

//结束form2的调用

this.Dispose(false);

//统计

private void Count(object sender, EventArgs e)

{

//先进行排序

Sort(sender, e);

string temp = "";

int[,] array = new int[4,3] { {0,0,0}, { 0, 0, 0 }, { 0, 0, 0 }, { 0, 0, 0 } };

double[] avg\_array = new double[3] { 0,0,0};

string[] range = new string[5] { "<60", "60-69", "70-79", "80-89", "平均分" };

int peopleNum = 0;

File.Delete("G:\\data.txt");

//向data.txt中输入列头

for(int i=0; i<this.listView1.Columns.Count; i++)

{

temp += this.listView1.Columns[i].Text + "\t";

}

temp = temp.TrimEnd() + "\r\n";

File.AppendAllText("G:\\data.txt", temp, UTF8Encoding.Default);

//向data.txt中输入排名信息

if (!File.Exists("G:\\scoremanagenment.txt"))

{

File.WriteAllText("G:\\scoremanagenment.txt", "");

}

string file = File.ReadAllText("G:\\scoremanagenment.txt", UTF8Encoding.Default);

string[] records = Regex.Split(file, "\r\n");

for(int i=0; i<records.Length; i++)

{

if(records[i].Trim().Length != 0)

{

//统计各个分数段的人数

peopleNum += 1;

string[] components = Regex.Split(records[i], "\t");

for(int index=2; index<components.Length-3; index++)

{

double grade = Convert.ToDouble(components[index]);

avg\_array[index - 2] += grade;

switch ((int)(grade / 10))

{

case 10:

break;

case 9:

break;

case 8:

array[3,index - 2] += 1;

break;

case 7:

array[2, index - 2] += 1;

break;

case 6:

array[1, index - 2] += 1;

break;

default:

array[0, index - 2] += 1;

break;

}

}

File.AppendAllText("G:\\data.txt", records[i] + "\r\n", UTF8Encoding.Default);

}

}

//向data.txt中输入统计信息

temp = "";

for(int i=2; i<this.listView1.Columns.Count-3; i++)

{

temp += this.listView1.Columns[i].Text + "\t";

}

temp = "\r\n\r\n\r\n" + "分数段\t" + temp.Trim() + "\r\n";

File.AppendAllText("G:\\data.txt", temp, UTF8Encoding.Default);

//输入具体信息

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

{

temp = range[i] + "\t";

if (i == 4)

{

for(int index=0; index<avg\_array.Length; index++)

{

temp += (avg\_array[index] \* 1.0 / peopleNum).ToString("0.#") + "\t";

}

File.AppendAllText("G:\\data.txt", temp.TrimEnd()+"\r\n", UTF8Encoding.Default);

}

else

{

for(int index=0; index<3; index++)

{

temp += array[i,index] + "\t";

}

File.AppendAllText("G:\\data.txt", temp.TrimEnd() + "\r\n", UTF8Encoding.Default);

}

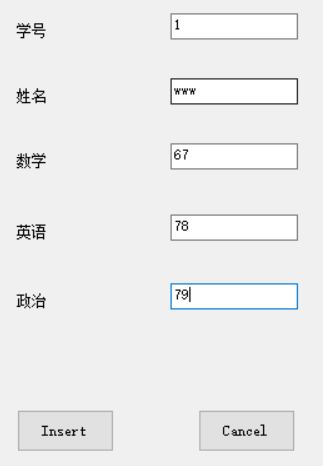
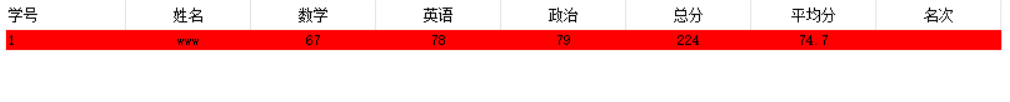
}

}

**五、实验结果**

程序运行结果截图及各模块功能说明。

插入：

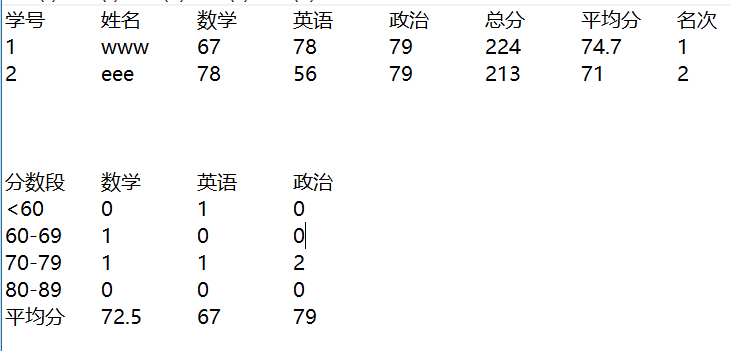
查找：

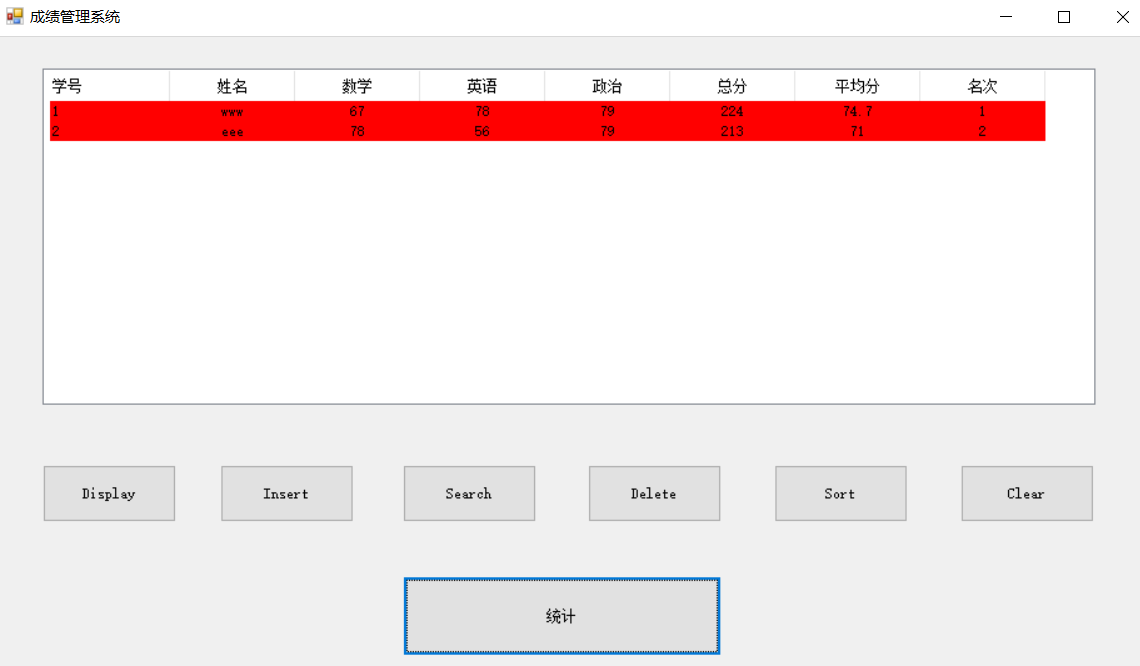
排序：



统计：



前端界面：



**六、实验总结**

总结实验过程及取得实验成果等收获，分析不足之处。

1通过这个实验我感觉自己的动手能力有很大提高，对c#也更加熟悉。在这个实验中我从开始编程以来真正开始使用调试工具，调试工具可以帮助我们更快的解决bug，更加了解程序的执行步骤。

2实验中我遇到了几个问题。我在写排序算法时要比较每个同学的总成绩大小，但是获取这个字符串时一直报索引越界的错误，在百度上查了很多资料都没解决

但是用断点调试工具很容易就解决了(没错，吹爆断点调试)，原来是从txt文件中读取的字符进行正则匹配后可能只有一个空串，对他进行过大下标的访问当然会报错。还有一个小问题，txt文件中的每行数据的item之间我用了一个” ”分开，不过统计出数据后我发现字和字之间都是挨着的(很乱), 改为\t之后就变得很整齐了。

3另外，我发现了c#和我之前学习的语言有很多地方不一样。比如说c#中的二维数组

array[R，C], 而我的惯性思维则是array[R][C], 当时困恼了好久。我在写switch时条件是把一个double类型转为int类型，我试了Convert.ToInt16/32/64() 但是我发现一个很奇怪的问题，比如说6.8，我觉得会被转换为6，结果他转换为了7，所以开始的时候统计出的数据很多错误, (int)num\_double 即可正确转换，但我开始做时直接想到了int(num\_double)和利用Convert来做(前者报错，后者转换不准确). 说明了计算机底层还是要补得。

不足：

1很多地方的数据是写死的，如果以后的界面发生变化又要重写。

2一部分代码并没有考虑算法复杂度。代码的可优化度很高很高。