目录

[第一章 题目解读与要求分析 2](#_Toc14753)

[§1 实习题目 2](#_Toc9600)

[§2 功能要求 2](#_Toc11483)

[§3 界面要求 2](#_Toc18965)

[第二章 需求分析 2](#_Toc23545)

[§1 问题描述 2](#_Toc8009)

[§2 系统环境 3](#_Toc2392)

[§3 运行要求 3](#_Toc4701)

[第三章 软件设计与实现 5](#_Toc14372)

[§1 数据结构和存储结构的设计 5](#_Toc250)

[§2 算法设计 6](#_Toc16875)

[ 递归算法 6](#_Toc3822)

[ 计数器累加判断是否已存在的算法 8](#_Toc4558)

[ 读取一个文件夹下的所有文件或文件夹的算法 8](#_Toc16164)

[§3 模块设计 9](#_Toc31285)

[ 树形控件显示模块 9](#_Toc4345)

[ 列表控件显示模块 10](#_Toc309)

[ 返回、转到与显示地址模块 10](#_Toc11677)

[ 创建模块 11](#_Toc8451)

[ 删除模块 11](#_Toc19750)

[ 复制模块 11](#_Toc9517)

[ 粘贴模块 12](#_Toc31093)

[§4 类的函数成员和数据成员设计 12](#_Toc21106)

[ CFileSystemDlg类的定义 12](#_Toc17706)

[§5 界面设计 14](#_Toc13928)

[§6 其它模块设计与实现 15](#_Toc32557)

[ 界面刷新模块 15](#_Toc9559)

[第四章 调试分析 17](#_Toc28572)

[§1 主要问题及解决方案 17](#_Toc24564)

[ 树形控件与列表控件之间的信息传递问题 17](#_Toc20772)

[ 复制、粘贴的实现问题 18](#_Toc26589)

[§2 设计和编码的回顾讨论和分析 19](#_Toc4612)

[§3 算法的时间和空间复杂度的分析 20](#_Toc261)

[§4 算法的进一步改进 21](#_Toc594)

[ 计数器累加判断是否已存在的算法 21](#_Toc3952)

[ 复制粘贴功能利用系统剪贴板实现 21](#_Toc310)

[ 文件或者文件夹重命名 21](#_Toc15349)

[ 在调试的过程中出现了四个警告并且程序最终返回值不为0（为2） 21](#_Toc2360)

[第五章 实习心得 22](#_Toc24538)

# 题目解读与要求分析

## §1 实习题目

实现一个文件系统。

## §2 功能要求

* 界面上显示树形目录结构

a）根节点是“我的电脑”

b）“我的电脑”下有几个盘符（C、D、E等）就有几个子节点，递归显 示文件系统下的所有文件信息（分支可以是目录也可以是文件，叶 子节点都是文件）

* 能够创建目录、创建文件、删除目录、删除文件、复制文件、粘贴文件

## §3 界面要求

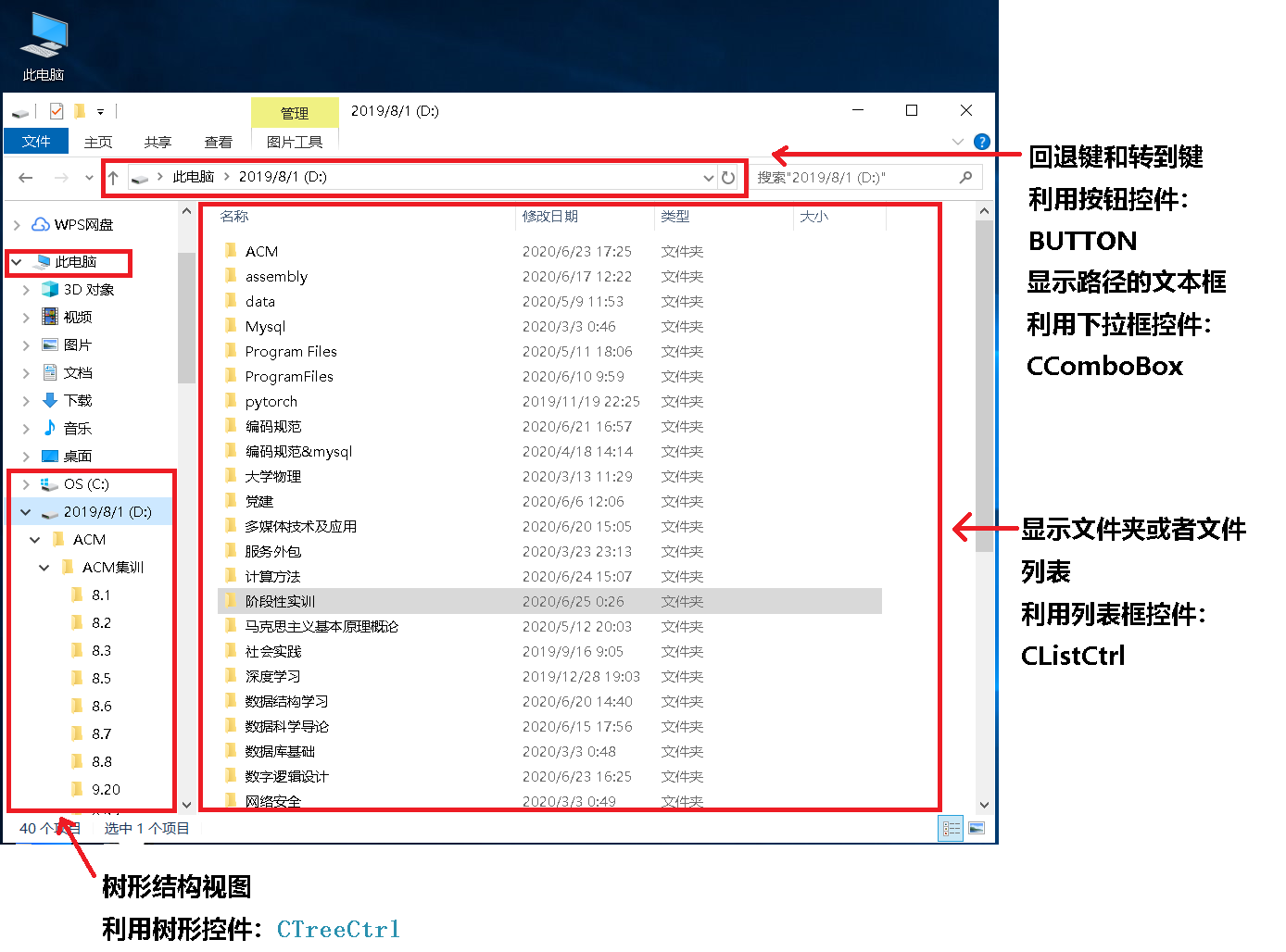
界面要友好，让用户操作使用起来非常方便。

# 需求分析

## §1 问题描述

由题目以及题目功能要求与界面要求可知，这一题首先要求我们能够获取电脑本地的所有盘符；然后利用递归将本地的所有文件夹以及文件都获取到；接着将获取到的所有内容以树形目录结构的形式在UI界面显示，并且任一文件夹中的内容（文件夹、文件）可以以列表的形式在UI界面显示出来，以此来给用户较好的体验；最后在写出的可视化界面上要能够对文件夹或者文件进行创建、删除、复制、粘贴等操作，这些操作应以可视化的形式显示并且要足够简单，让用户操作使用起来足够方便。

由上述描述分析可知，本题要求做出一个功能类似Windows 10操作系统中的“此电脑”的可视化应用程序，故可以仿照“此电脑”的UI设计及相关功能的操作方法。



## §2 系统环境

操作系统：Windows 10

编程语言：C++(MFC)

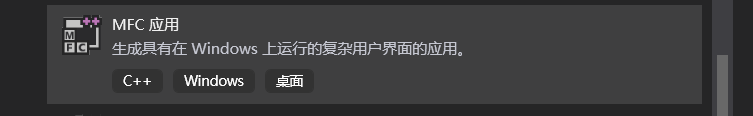
开发环境及编译器：Visual Studio 2019 + MFC(C++的API)

硬件：除分好盘的硬盘空间外，能够流畅运行VS2019的基础硬件即可

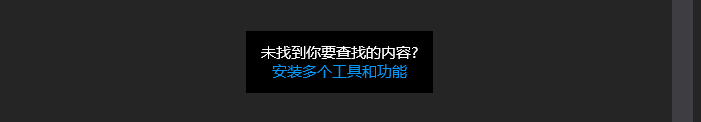
## §3 运行要求

VS2019里面配置MFC

新建MFC项目



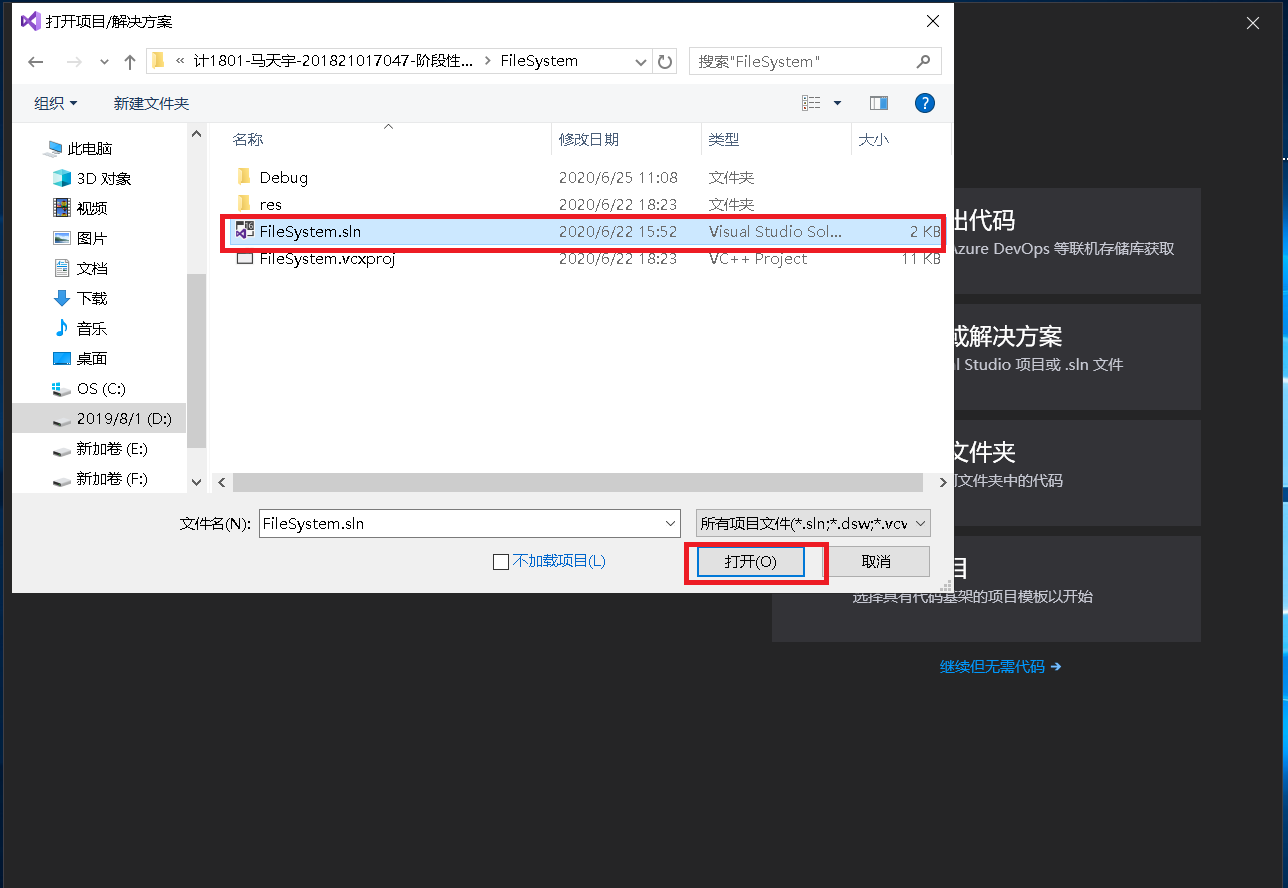
若无此项，拖动到底部安装相关工具和功能







**若已配置好环境，就可以在vs里面直接打开项目**

****

# 软件设计与实现

## §1 数据结构和存储结构的设计

因为这个文件系统的程序需要实时读取当前磁盘空间的存储状态及存储内容并且在C++中具有读取文件或文件夹状态以及路径的API函数，所以在程序中对于文件或文件夹的操作并不需要其他什么特殊的数据结构和存储结构，只要在使用树控件的时候利用现成的API获取相应的信息并将这些信息在树形控件中显示出来即可。

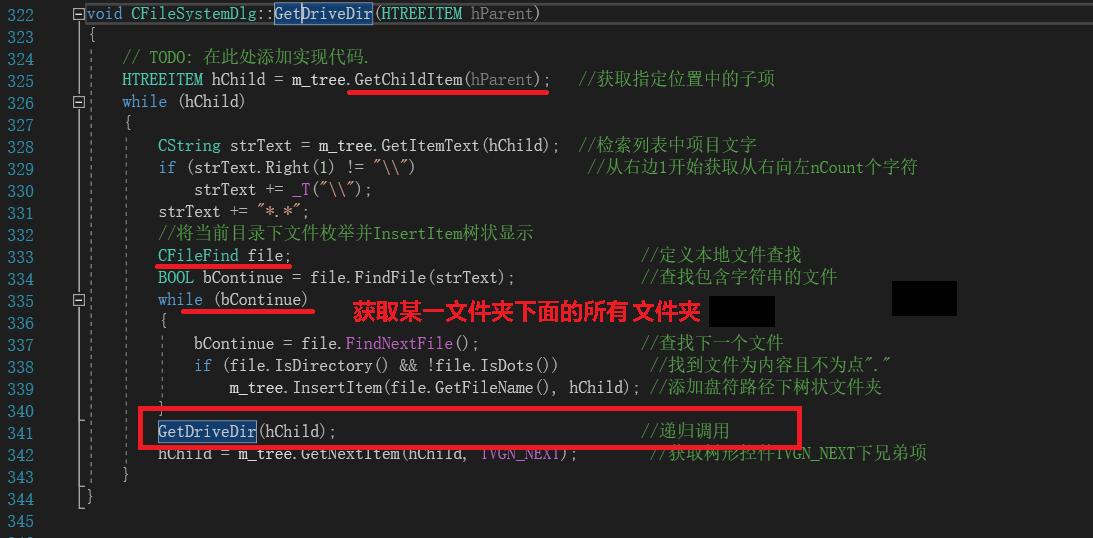
若想存储文件或文件夹的信息，可以采用树结构来存储，一个结点的孩子结点是这个结点所代表的文件夹中的所有文件或者其他文件夹，其中文件一定处在叶子结点上；其次，为了查询的方便、快速以及IO开销尽可能小，可以采用B-树或者B+树的结构存储，可以减小IO开销，提高查询效率。

其他的一些数据结构，例如栈、堆、队列、数组等，用法、设计与其余所有C++程序相同。（比如调用函数，利用了堆栈等）

## §2 算法设计

### 递归算法

1、获取所有盘符的下一级目录



2、删除目录及目录中的所有内容，要删除一个目录首先要删除该目录下的所有文件和文件夹，然后再移除空文件夹，同样删除该文件夹下的文件夹也需要先删除其中的文件和文件夹，就形成了递归。

bool CFileSystemDlg::DeleteFolder(LPCTSTR pstrFolder)

{

// TODO: 在此处添加实现代码.

if ((NULL == pstrFolder))

{

return FALSE;

}

/\*检查输入目录是否是合法目录\*/

if (!IsDirectory(pstrFolder))

{

return FALSE;

}

/\*创建源目录中查找文件的通配符\*/

CString strWildcard(pstrFolder);

if (strWildcard.Right(1) != \_T('\\'))

{

strWildcard += \_T("\\");

}

strWildcard += \_T("\*.\*");

/\*打开文件查找，查看源目录中是否存在匹配的文件\*/

/\*调用FindFile后，必须调用FindNextFile才能获得查找文件的信息\*/

CFileFind finder;

BOOL bWorking = finder.FindFile(strWildcard);

while (bWorking)

{

/\*查找下一个文件\*/

bWorking = finder.FindNextFile();

/\*跳过当前目录“.”和上一级目录“..”\*/

if (finder.IsDots())

{

continue;

}

/\*得到当前目录的子文件的路径\*/

CString strSubFile = finder.GetFilePath();

/\*判断当前文件是否是目录,\*/

/\*如果是目录，递归调用删除目录,\*/

/\*否则，直接删除文件\*/

if (finder.IsDirectory())

{

if (!DeleteFolder(strSubFile))

{

finder.Close();

return FALSE;

}

}

else

{

if (!DeleteFile(strSubFile))

{

finder.Close();

return FALSE;

}

}

} /\*while (bWorking)\*/

/\*关闭文件查找\*/

finder.Close();

/\*删除空目录\*/

return RemoveDirectory(pstrFolder);

}

### 计数器累加判断是否已存在的算法

每一次需要创建文件或者文件夹，都从1开始若文件或文件夹已存在计数器就加1，形成新的文件夹或文件的名称，直到没有重复时就以该名称创建文件或文件夹。

**文件夹：**

str += \_T("新建文件夹1");

while (PathIsDirectory(str)) {

str.Delete(str.GetLength() - 1, 1);

CString chg;

chg.Format(\_T("%d"), f\_cnt);

str += chg;

f\_cnt++;

}

**文本文档：**

str += \_T("新建文本文档1.txt");

while (PathFileExists(str)) {

CString chg;

chg.Format(\_T("%d"), t\_cnt);

str.SetAt(str.Find(\_T(".")) - 1, chg.GetAt(0));

t\_cnt++;

}

CFile file(str, CFile::modeCreate);

file.Close();

**DOCX文档：**

str += \_T("新建DOCX文档1.docx");

while (PathFileExists(str)) {

CString chg;

chg.Format(\_T("%d"), d\_cnt);

str.SetAt(str.Find(\_T(".")) - 1, chg.GetAt(0));

d\_cnt++;

}

CFile file(str, CFile::modeCreate);

file.Close();

### 读取一个文件夹下的所有文件或文件夹的算法

运用MFC中的CFileFind类中的FindFile(str)获取一个文件夹下面的所有文件和文件夹。

CFileFind file;

BOOL bContinue = file.FindFile(str);

while (bContinue) {

bContinue = file.FindNextFileW();

if (!file.IsDots()) { //文件和文件夹

SHFILEINFO info = { 0 };

CString temp = str;

int index = temp.Find(\_T("\*.\*"));

temp.Delete(index, 3);

SHGetFileInfo(temp + file.GetFileName(), 0, &info, sizeof(&info), SHGFI\_DISPLAYNAME | SHGFI\_ICON);

int i = m\_ImageList.Add(info.hIcon);

m\_list.SetImageList(&m\_ImageList, LVSIL\_SMALL); //设置图标

m\_list.InsertItem(i, info.szDisplayName, i); //在列表插入一项

}

}

## §3 模块设计

### 树形控件显示模块

1、获取所有盘符并在树形控件中的根节点“我的电脑”的孩子结点上插入：

void CFileSystemDlg::GetLogicalDrive(HTREEITEM hParent)

2、获取所有盘符下的所有文件夹并在相应盘符结点的孩子结点上插入（仅仅是盘符的下一级）：

void CFileSystemDlg::GetDriveDir(HTREEITEM hParent)

**注：盘符的下一级单独处理的原因是盘符与普通文件夹不同需要单独处理。**

3、返回某一结点的绝对路径（从结点开始向根节点回溯）：

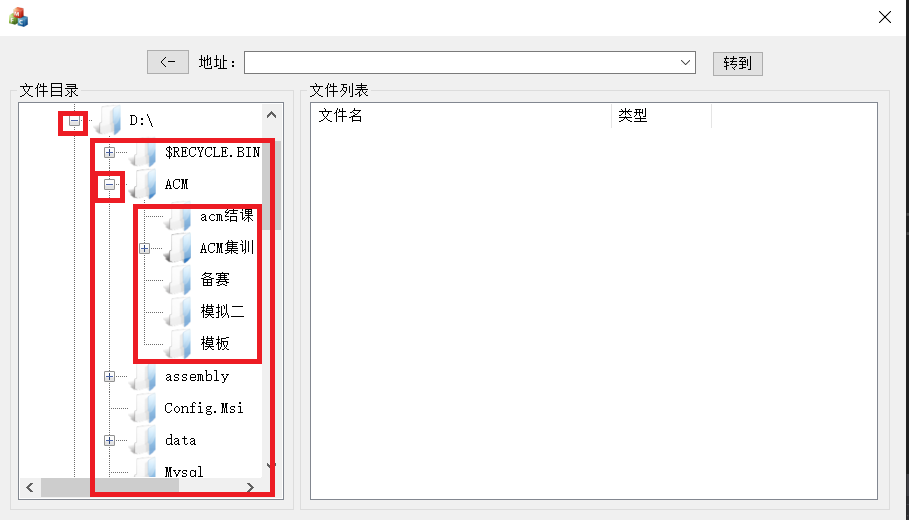
CString CFileSystemDlg::GetFullPath(HTREEITEM hCurrent)

4、获取某一文件夹下的所有文件夹并在代表该文件夹的结点的孩子结点上插入子文件夹：

void CFileSystemDlg::AddSubDir(HTREEITEM hParent)

5、展开树形控件某一结点时触发的事件（显示该节点下的所有文件夹并载入再下一层的文件夹）：

void CFileSystemDlg::OnItemexpandedTree(NMHDR\* pNMHDR, LRESULT\* pResult)



### 列表控件显示模块

1、选择树形控件的某一节点时触发的事件（将该结点代表的文件下的所有文件夹以及文件显示在列表控件中）：

void CFileSystemDlg::OnSelchangedTree(NMHDR\* pNMHDR, LRESULT\* pResult)

2、鼠标左键双击列表控件时触发的事件（文件夹：打开该文件夹在列表控件中显示；文件：调用外部应用程序打开文件）（未选中任何一项无动作）：

void CFileSystemDlg::OnDblclkList(NMHDR\* pNMHDR, LRESULT\* pResult)

3、鼠标右键单击列表控件时触发的事件（选中了某一项时显示操作菜单：打开、复制、删除；未选中某一项时显示操作菜单：刷新、新建、粘贴）：

void CFileSystemDlg::OnRclickList(NMHDR\* pNMHDR, LRESULT\* pResult)

### 返回、转到与显示地址模块

1、单击“返回”按钮时触发的事件（返回上一级目录）：

void CFileSystemDlg::OnClickedBack()

2、单击“转到”按钮时触发的事件（进入下拉列表框中输入的路径所表示的文件夹内）：

void CFileSystemDlg::OnClickedEnter()

3、下拉列表框中所选项发生变化时所触发的事件（进入下拉列表框所选项表示的文件夹内）：

void CFileSystemDlg::OnSelchangeDirpath()

### 创建模块

**注：文件、文件夹的命名均通过计数器来避免名称冲突而创建失败。**

1、鼠标右键单击列表空白处后弹出菜单，再鼠标左键单击菜单的“新建-文件夹”时触发的事件（在操作的目录下新建一个文件夹）：

void CFileSystemDlg::OnNewfile()

2、鼠标右键单击列表空白处后弹出菜单，再鼠标左键单击菜单的“新建-文本文档”时触发的事件（在操作的目录下新建一个文本文档）：

void CFileSystemDlg::OnTxt()

3、鼠标右键单击列表空白处后弹出菜单，再鼠标左键单击菜单的“新建-DOCX文档”时触发的事件（在操作的目录下新建一个DOCX文档）：

void CFileSystemDlg::OnDocx()

### 删除模块

1、判断一个路径是否是已存在的文件夹

bool CFileSystemDlg::IsDirectory(LPCTSTR pstrPath)

2、删除文件夹及文件夹中的所有内容

bool CFileSystemDlg::DeleteFolder(LPCTSTR pstrFolder)

3、选中列表中的某一项后鼠标右键单击后弹出菜单，再鼠标左键单击菜单的“删除”时触发的事件（删除选中的文件夹或者文件）：

void CFileSystemDlg::OnDelete()

**注：如果删除的是文件可以直接调用mfc的函数来删；如果删除的是文件夹，那么就需要先删除该文件夹下所有的文件夹以及文件后再移除该空文件夹（递归思想）。**

### 复制模块

1、将参数（要复制的文件或文件夹的绝对路径）赋给成员变量dir\_path：

void CFileSystemDlg::CopyToClipboard(CString dirPath)

2、选中列表中的某一项后鼠标右键单击后弹出菜单，再鼠标左键单击菜单的“复制”时触发的事件（得到选中项的绝对路径赋给成员变量dir\_path）（在事件处理中调用

CopyToClipboard(dirPath)函数：

void CFileSystemDlg::OnCopy()

**注：将要复制的文件或文件夹的绝对路径赋给成员变量dir\_path是为了在粘贴时调用**

**CopyFile(dir\_path, dirPath, TRUE)函数。**

### 粘贴模块

1、将经过上述复制操作的文件或者文件夹粘贴到参数dirPath目录下:

void CFileSystemDlg::PasteToFile(CString dirPath)

2、鼠标右键单击列表空白处后弹出菜单，再鼠标左键单击菜单的“粘贴”时触发的事件（在操作的目录下粘贴已复制的文件或者文件夹）：

void CFileSystemDlg::OnPaste()

**注：粘贴文件夹时是先创建一个文件夹，然后再在这个新创建的文件夹里面粘贴文件，嵌套（递归）。**

## §4 类的函数成员和数据成员设计

**注：在程序中主要是在CFileSystemDlg类进行编程，所以另外两个自动生成的类这里不做赘述，下面是CFileSystemDlg类的类定义，其中用到的数据成员都给了注释，而用到的成员函数在上一节“模块设计”小节以及本章最后一小节“其他模块设计与实现”小节中已经说明。**

### CFileSystemDlg类的定义

// CFileSystemDlg 对话框

class CFileSystemDlg : public CDialogEx

{

// 构造

public:

CFileSystemDlg(CWnd\* pParent = nullptr); // 标准构造函数

// 对话框数据

#ifdef AFX\_DESIGN\_TIME

enum { IDD = IDD\_FILESYSTEM\_DIALOG };

#endif

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持

// 实现

protected:

HICON m\_hIcon; // 图标

// 生成的消息映射函数

virtual BOOL OnInitDialog(); // 初始化操作

afx\_msg void OnSysCommand(UINT nID, LPARAM lParam);

afx\_msg void OnPaint();

afx\_msg HCURSOR OnQueryDragIcon();

DECLARE\_MESSAGE\_MAP()

public:

CListCtrl m\_list; // 列表控件

CTreeCtrl m\_tree; // 树形控件

CComboBox m\_combo; // 下拉列表框控件（起到桥梁作用，传递路径）

protected:

HTREEITEM m\_hRoot; // CTreeCtrl控件的项句柄，在树中标识唯一一 //个节点是一个DWORD值

CImageList m\_ImageList; //树结点文字内容前面的图标控件

CString dir\_path; //保存需要复制的文件或者文件夹的绝对路径

int f\_cnt; //新建文件夹避免重名用到的计数器

int t\_cnt; //新建文本文档避免重名用到的计数器

int d\_cnt; //新建DOCX文档避免重名用到的计数器

public:

void GetLogicalDrive(HTREEITEM hParent);

void GetDriveDir(HTREEITEM hParent);

CString GetFullPath(HTREEITEM hCurrent);

void AddSubDir(HTREEITEM hParent);

afx\_msg void OnItemexpandedTree(NMHDR\* pNMHDR, LRESULT\* pResult);

afx\_msg void OnSelchangedTree(NMHDR\* pNMHDR, LRESULT\* pResult);

afx\_msg void OnDblclkList(NMHDR\* pNMHDR, LRESULT\* pResult);

afx\_msg void OnOpen();

afx\_msg void OnRclickList(NMHDR\* pNMHDR, LRESULT\* pResult);

afx\_msg void OnCopy();

afx\_msg void OnDelete();

afx\_msg void OnNewfile();

afx\_msg void OnTxt();

// 更新ListCtrl

void Refresh(CString str);

afx\_msg void OnDocx();

afx\_msg void OnRefresh();

afx\_msg void OnPaste();

bool DeleteFolder(LPCTSTR pstrFolder);

bool IsDirectory(LPCTSTR pstrPath);

afx\_msg void OnClickedBack();

afx\_msg void OnClickedEnter();

afx\_msg void OnSelchangeDirpath();

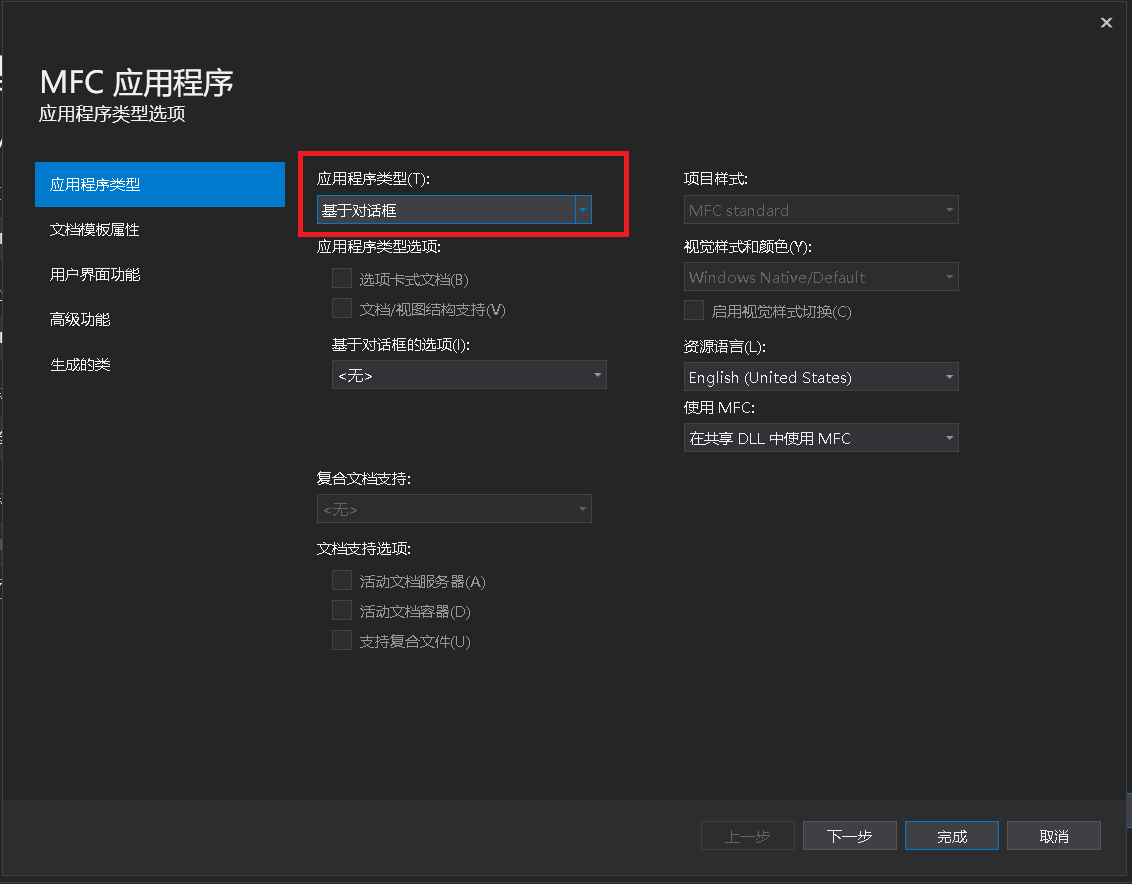
void CopyToClipboard(CString dirPath);

void PasteToFile(CString dirPath);

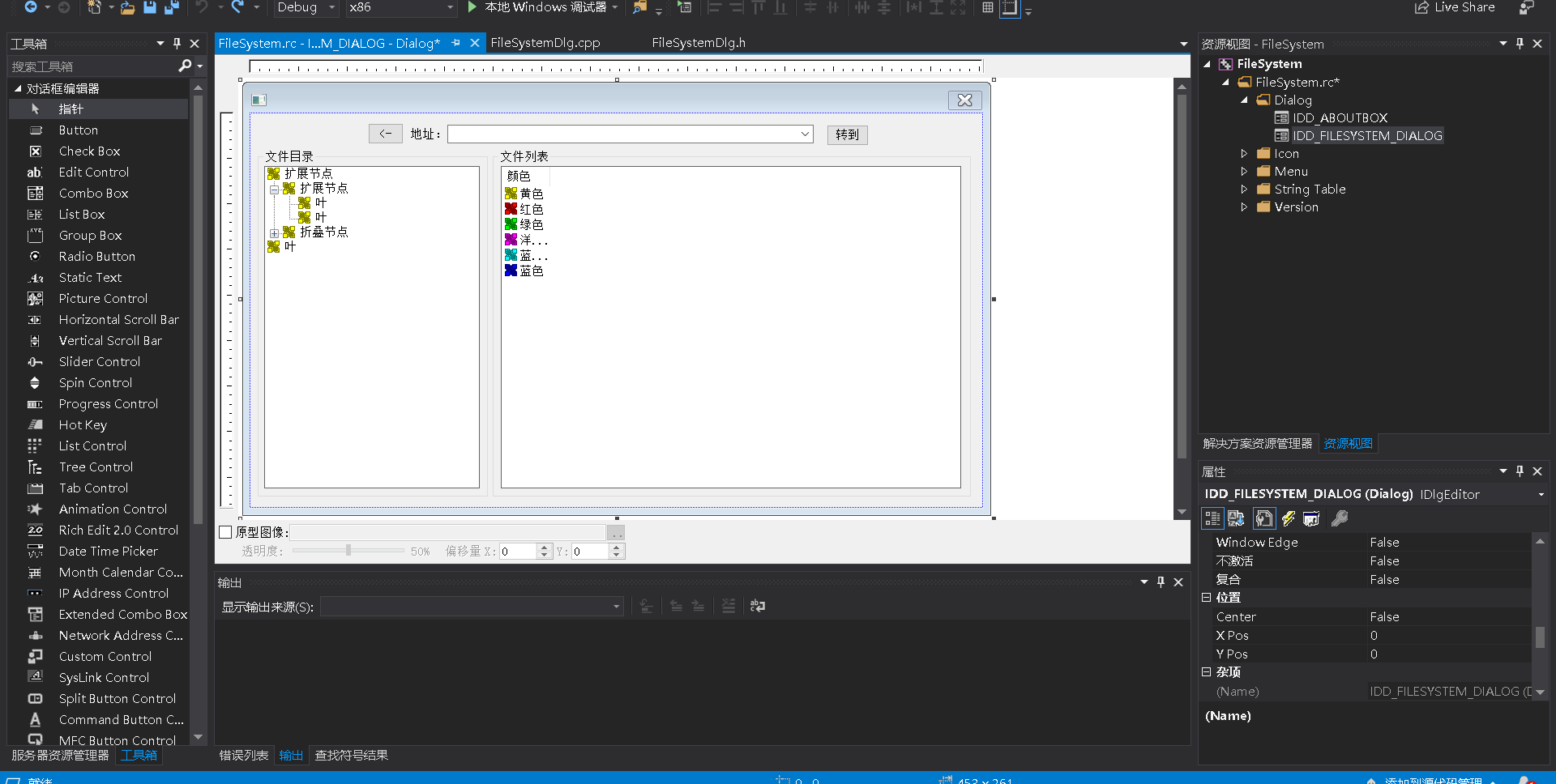
};

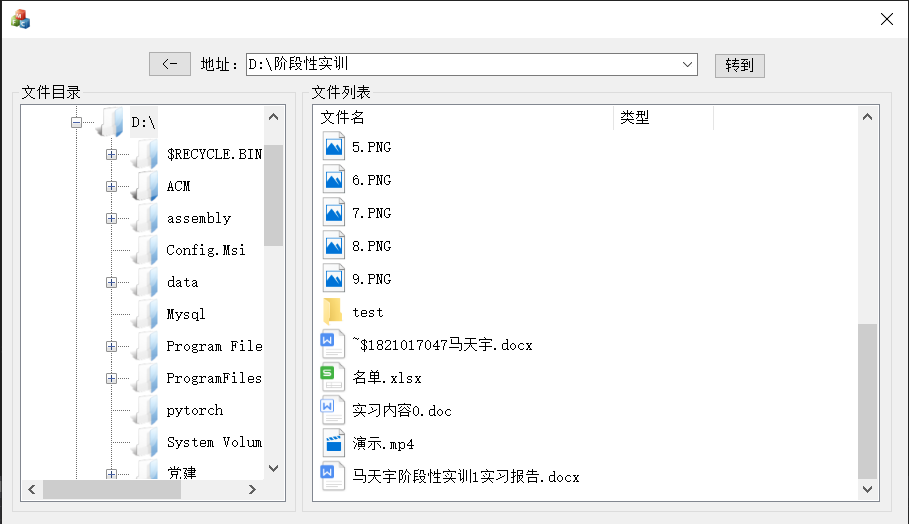
## §5 界面设计

采用MFC的基于对话框的界面设计



* 将默认生成的确定和取消按钮删除
* 调整好对话框的大小
* 在对话框中添加两个GroupBox控件，并将他们的Caption属性分别改为“文件目录”和“文件列表”
* 添加树形控件Tree Control,设置相关属性
* 添加列表控件ListBox，设置相关属性
* 添加两个按钮控件Button，分别实现返回和转到的功能
* 添加下拉列表框控件Combo Box，设置相关属性





## §6 其它模块设计与实现

### 界面刷新模块

1、更新参数strTemp（绝对路径）目录在列表控件中的显示（在删除、新建、粘贴事件中均会调用这个函数）：

// 更新ListCtrl

void CFileSystemDlg::Refresh(CString strTemp)

{

// TODO: 在此处添加实现代码.

if (GetFileAttributes(strTemp) & FILE\_ATTRIBUTE\_DIRECTORY) {

//m\_combo.DeleteString(0);

m\_combo.InsertString(0, strTemp);

m\_combo.SetCurSel(0);

if (strTemp.Right(1) != "\\") {

strTemp += "\\";

}

strTemp += "\*.\*";

CFileFind file;

BOOL bContinue = file.FindFile(strTemp);

m\_list.DeleteAllItems();

while (bContinue) {

bContinue = file.FindNextFileW();

if (!file.IsDots()) {

SHFILEINFO info = { 0 };

CString temp = strTemp;

int index = temp.Find(\_T("\*.\*"));

temp.Delete(index, 3);

SHGetFileInfo(temp + file.GetFileName(), 0, &info, sizeof(&info), SHGFI\_DISPLAYNAME | SHGFI\_ICON);

int i = m\_ImageList.Add(info.hIcon);

m\_list.SetImageList(&m\_ImageList, LVSIL\_SMALL);

m\_list.InsertItem(i, info.szDisplayName, i);

}

}

}

else {

if (PathFileExists(strTemp))

ShellExecute(NULL, TEXT("OPEN"), strTemp, NULL, NULL, SW\_SHOWNORMAL); //调用外部程序打开文件

}

}

2、鼠标右键单击列表空白处后弹出菜单，再鼠标左键单击菜单的“刷新”时触发的事件（在操作的目录下更新显示的列表控件中的内容）：

void CFileSystemDlg::OnRefresh()

{

// TODO: 在此添加命令处理程序代码

CString strTemp;

m\_combo.GetWindowText(strTemp);

if (strTemp.Right(1) != "\\") {

strTemp += "\\";

}

Refresh(strTemp);

}

# 调试分析

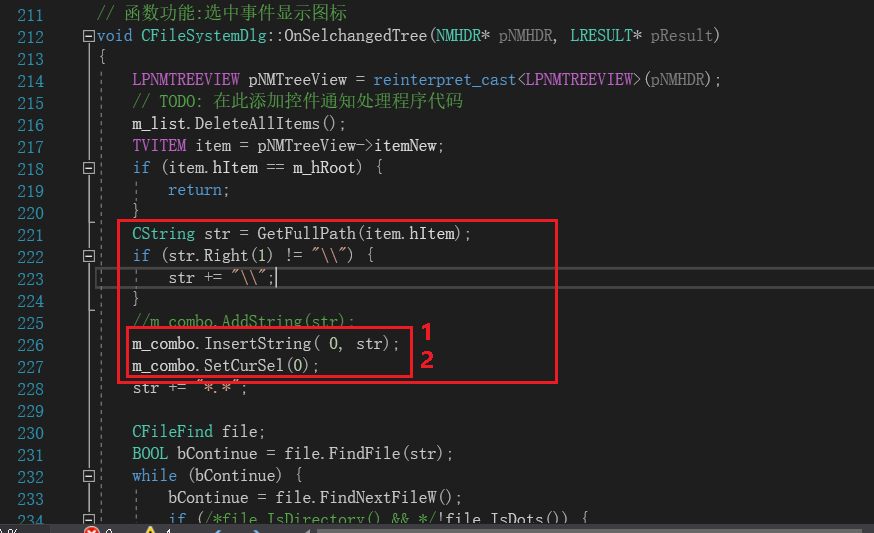
## §1 主要问题及解决方案

### 树形控件与列表控件之间的信息传递问题

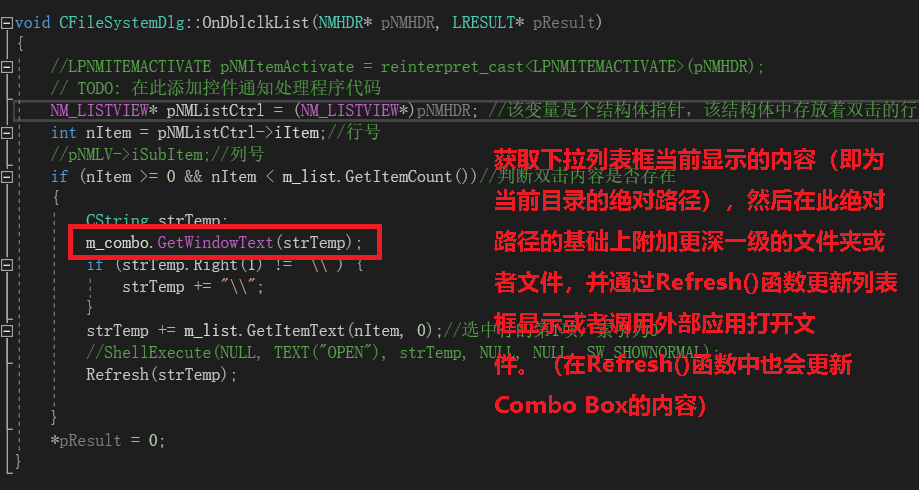
**问题：**在列表控件的事件处理中如何获得操作的绝对路径，并在此基础上进行更复杂的操作？

**分析：**选择树形控件的某一节点之后，会在列表控件中显示该节点所表示的文件夹内的所有内容。此时，如果不保留所选节点的绝对路径，那么只能得到该节点所代表的文件夹内的所有内容的名称，而丢失他们的绝对路径，堆在列表控件中要进行的新建、删除、复制、粘贴等操作就无从下手（这些操作都需要相应文件夹或者文件的绝对路径）。

**解决方案：**首先可以想到在类的成员变量中设置一个存放当前列表控件所显示的目录的绝对路径，并在各种操作中获取或更新这个成员变量，这种方案是可行的；但是还有一种更加方便快捷的方案：**利用Combo Box下拉列表控件**，在选择树形控件的某一节点进行事件处理时，在Combo Box中插入当前展示的目录的绝对路径并展示出来，这样以后在列表控件中需要使用用绝对路径的时候，只要在Combo Box控件中获取即可。



**1处将路径插入下拉列表框的列表中，2处将刚刚插入的内容显示出来。**

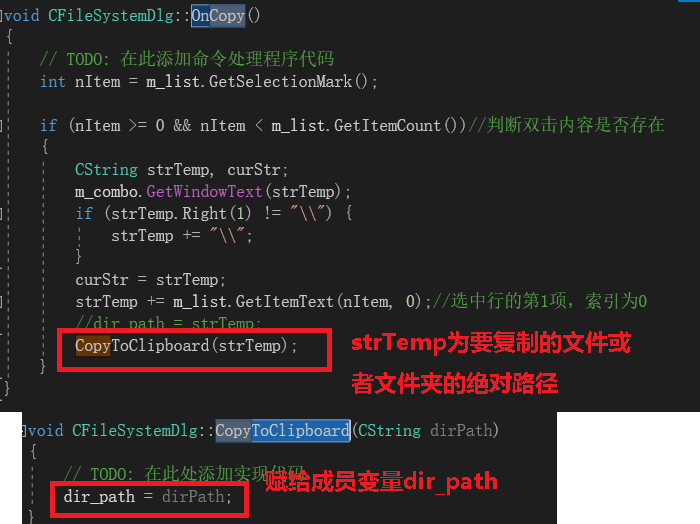
****

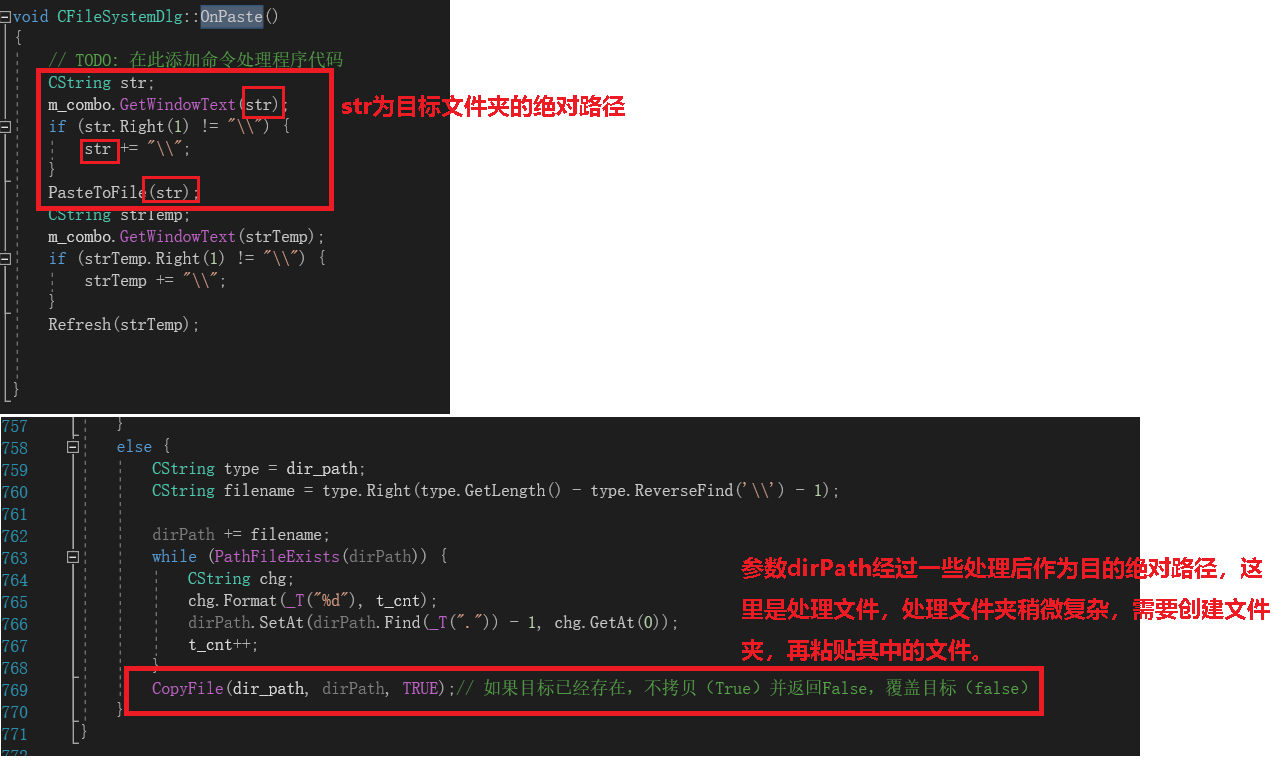
### 复制、粘贴的实现问题

**问题：**如何复制粘贴？利用剪贴板和直接采用MFC的API哪个好？

**分析：**复制粘贴有两种思路。第一，利用系统的剪贴板，将要复制的文件放到剪贴板上取，然后在粘贴的时候，从剪贴板上获取之后粘贴（**看似**比较可行）；第二，利用MFC的API，直接通过需要复制的文件的绝对路径和要复制到的绝对路径，调用CopyFile(source, destination, TRUE)直接复制（**看似**笨重，很不灵活）。开始死钻第一种思路，结果发现文件的类型多样并且文件夹嵌套较多，处理起来十分困难，网上也没有找到相关的教程，似乎走进了死胡同。这时，回头看一看第二种思路，发现只要稍微变通一下：在复制的时候只将文件或者文件夹的绝对路径保存下来（**这里采用数据成员的方式进行保存**），然后在粘贴的时候，可以获取要粘贴到文件夹的绝对路径，再进行一些处理，即可利用CopyFile(source, destination, TRUE)直接将文件复制粘贴过去。

**解决方案：**在复制的时候只将文件或者文件夹的绝对路径保存下来（**这里采用数据成员的方式进行保存**），然后在粘贴的时候，可以获取要粘贴到文件夹的绝对路径，再进行一些处理，即可利用CopyFile(source, destination, TRUE)直接将文件复制粘贴过去。





## §2 设计和编码的回顾讨论和分析

整个程序是以mfc的可视化对话框为中心来设计编码的，通过对控件控件成员的操作以及一些适当的辅助成员的利用，可以完美实现题目中要求的大部分功能。

在编码时，结合帮助文档以及一些博客的讲解，熟悉掌握CFile、CString、CFileFind等类的基本使用方法，对绝对路径的字符串进行操作以及对文件和文件夹的操作，能够写出这样的程序。

熟悉MFC的时间处理机制，与QT的信号与槽进行对比（虽然大一下学期做大作业的时候学的QT知识已经忘的差不多了）。

## §3 算法的时间和空间复杂度的分析

主要的算法都是直接调用MFC中封装好的API，故算法的时间复杂度和空间复杂度基本上就是API函数的复杂度。

同样也没有特别的空间复杂度。

以下为最坏时间复杂度：

void GetLogicalDrive(HTREEITEM hParent);//O(盘符个数)

void GetDriveDir(HTREEITEM hParent);//O(盘符个数×平均盘符下的文件和文 //件夹的数量)

CString GetFullPath(HTREEITEM hCurrent);//O(树结点所在的层数)

void AddSubDir(HTREEITEM hParent);//O(文件夹下的文件夹和文件之和)

afx\_msg void OnItemexpandedTree(NMHDR\* pNMHDR, LRESULT\* pResult);

//O(所展开节点下一层目录中的所有文件夹及其子文件夹的个数)

afx\_msg void OnSelchangedTree(NMHDR\* pNMHDR, LRESULT\* pResult);

//O(选择的节点所代表的文件夹下的所有文件和文件夹之和)

afx\_msg void OnDblclkList(NMHDR\* pNMHDR, LRESULT\* pResult);

//O(选择的文件夹下的所有文件和文件夹之和)

afx\_msg void OnOpen();//O(选择的文件夹下的所有文件和文件夹之和)

afx\_msg void OnRclickList(NMHDR\* pNMHDR, LRESULT\* pResult);

//O(展示弹出菜单的时间复杂度)

afx\_msg void OnCopy();//O(1)

afx\_msg void OnDelete();//O(要删除的文件夹下的所有文件和 //文件夹之和与删除的文件夹所在文件夹内文件 //和文件夹之和的最大值)

afx\_msg void OnNewfile();//O(新建文件夹所在文件夹下的所有文件和文件夹 //之和)

afx\_msg void OnTxt();//O(新建文本文档所在文件夹下的所有文件和文件夹 //之和)

void Refresh(CString str);//O(刷新所在文件夹下的所有文件和文件夹之和)

afx\_msg void OnDocx();//O(新建DOCX文档所在文件夹下的所有文件和文件夹 //之和)

afx\_msg void OnRefresh();//O(刷新所在文件夹下的所有文件和文件夹之和)

afx\_msg void OnPaste();//O(需要复制的文件夹下的所有文件和文 //件夹之和与粘贴到的文件夹内所有文件和文件夹之 //和的最大值)

bool DeleteFolder(LPCTSTR pstrFolder);//O(要删除的文件夹下的所有文件和 //文件夹之和)

bool IsDirectory(LPCTSTR pstrPath);//O(1)

afx\_msg void OnClickedBack();//O(上一级目录中文件和文件夹的和)

afx\_msg void OnClickedEnter();//O(所输入目录中文件和文件夹的和)

afx\_msg void OnSelchangeDirpath();//O(所选择目录中文件和文件夹的和)

void CopyToClipboard(CString dirPath);//O(1)

void PasteToFile(CString dirPath);//O(需要复制的文件夹下的所有文件和文 //件夹之和，一直到叶节点)

## §4 算法的进一步改进

### 计数器累加判断是否已存在的算法

原本的算法中采用了计数器来避免重复的文件或者文件夹的名称，但是计数器仍然存在一些问题，在原先的程序中，计数器只有递增而没有递减，所以在删除掉一些文件或者文件夹之后，会出现有的下标没有用到，造成顺序问题。

其实可以改进一下，不用计数器。每次新建都直接从1开始判断，都从1开始递增，通过循环，找到第一个空缺的下标并把它利用起来即可。

str += \_T("新建DOCX文档1.docx");

while (PathFileExists(str)) {

int n = 2;

CString chg;

chg.Format(\_T("%d"), n);

str.SetAt(str.Find(\_T(".")) - 1, chg.GetAt(0));

n++;

}

CFile file(str, CFile::modeCreate);

file.Close();

### 复制粘贴功能利用系统剪贴板实现

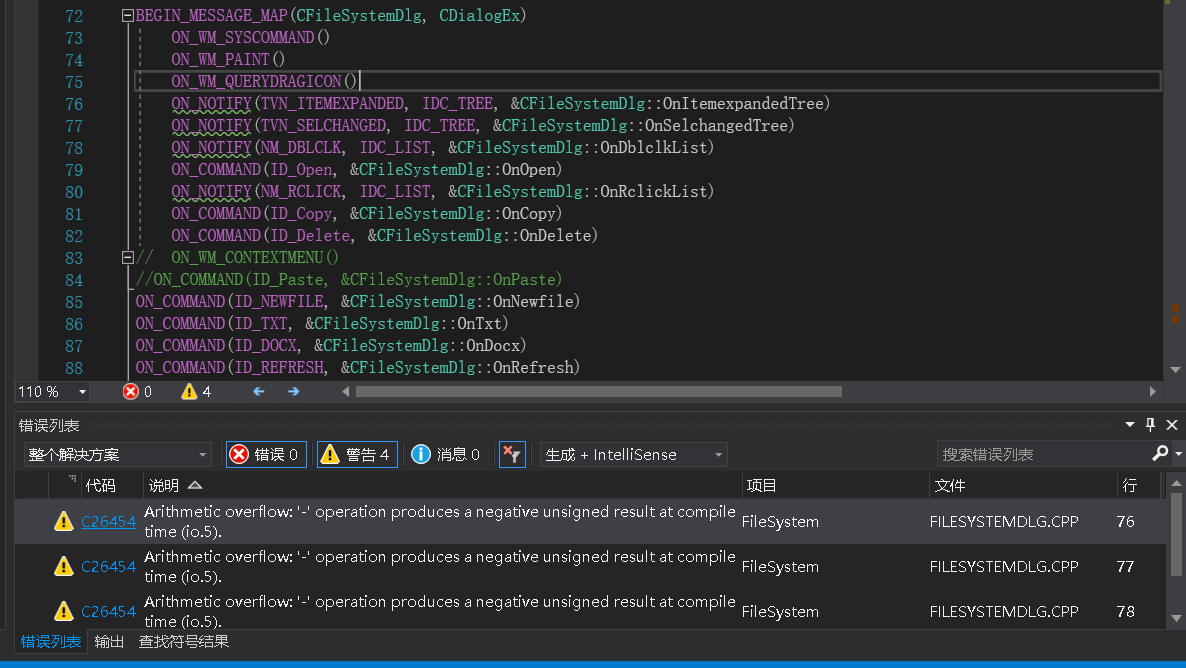
本程序中的复制粘贴功能利用CopyFile(dir\_path, dirPath, TRUE);函数实现，若采用系统剪贴板，自定义类型可能会更好。

### 文件或者文件夹重命名

本程序不能进行文件或者文件夹的重命名，若能够进行重命名将能够更加方便用户使用。

### 在调试的过程中出现了四个警告并且程序最终返回值不为0（为2）

四个警告是数据溢出，可能是空间的ID号设置的太大溢出了宏定义的范围。



开始通过修改控件的ID号可以解决这个问题，然而再次打开时又出现了该警告。

返回值不为0，应该是哪里出问题了。

# 实习心得

在为期一周的实训中，我学到了很多。从拿到题目的那一刻的模模糊糊到开始着手设计、深入的愈渐明朗到最终完成开发，这其中有成功的喜悦，也有失败的泄气，恰恰也正是在这样一个过程中，我学到了很多。

在开发一个程序的过程中遇到各种各样的问题是在所难免的，这就需要我们不断地去学习，去找博客，去吸收前辈们的经验。这次我使用的MFC是我这之前从未接触过的一个框架，但是在边学边写的过程中，我能够基本熟练使用这个框架。

当然，在这次实训中，我更深一步地了解和学习了软件开发的一般过程，从题目解读到需求分析再到设计和编码实现，经历了一个完整的开发过程，在以后的开发中我相信我能够更加熟练。