### 实验4 文件复制

班级： 07152001 学号： 1120201198 姓名： 史桠彬

一、实验目的

了解Windows的文件系统时如何管理保存在磁盘、光盘等存储介质上的信息。并通过文件系统提供的各种API，对文件进行操作，深入理解Windows文件系统的功能和作用，理解文件系统的系统调用的功能。

二、实验内容

1. 完成一个目录复制命令mycp，包括目录下的文件和子目录。
2. 在Windows下：调用CreateFile(), ReadFile(), WriteFile(), CloseHandle()等函数，完成文件的复制。

（要求每个文件不仅读写权限一致，而且时间属性一致。）

三、程序设计与实现

**1. 实验环境**  
操作系统：Windows10  
开发环境：Dev-Cpp 小熊猫.C++.0.14.2.64位.10.3.安装版

1. **设计思路**

① 结构体解析

* WIN32\_FIND\_DATA结构体：

typedef struct \_WIN32\_FIND\_DATA {

　　 DWORD dwFileAttributes; //文件属性

　　 FILETIME ftCreationTime; // 文件创建时间

　　 FILETIME ftLastAccessTime; // 文件最后一次访问时间

　　 FILETIME ftLastWriteTime; // 文件最后一次修改时间

　　 DWORD nFileSizeHigh; // 文件长度高32位

　　 DWORD nFileSizeLow; // 文件长度低32位

　　 DWORD dwReserved0; // 系统保留

　　 DWORD dwReserved1; // 系统保留

　　 TCHAR cFileName[ MAX\_PATH ]; // 文件名

　　 TCHAR cAlternateFileName[ 14 ]; // 格式文件名

} WIN32\_FIND\_DATA, \*PWIN32\_FIND\_DATA;

② 函数功能

* FindFirstFile()函数

HANDLE FindFirstFile(

LPCTSTR lpFileName,// 目录名

LPWIN32\_FIND\_DATA lpFindFileData// 文件信息结构体);

* FindNextFile()函数

BOOL CreateDirectory(

LPCTSTR lpPathName,//文件路径

LPSECURITY\_ATTRIBUTES lpSecurityAttributes//常为NULL);

* CreateDirectory()函数

BOOL CreateDirectory(

LPCTSTR lpPathName,//文件路径

LPSECURITY\_ATTRIBUTES lpSecurityAttributes//常为NULL);

* HANDLE CreateFile()函数

HANDLE CreateFile(

LPCTSTR lpFileName, // 指向文件名的指针

DWORD dwDesiredAccess, // 访问模式（写 / 读）

DWORD dwShareMode, // 共享模式

LPSECURITY\_ATTRIBUTES lpSecurityAttributes,//安全属性

DWORD dwCreationDisposition,//创建方式

DWORD dwFlagsAndAttributes,//文件属性

HANDLE hTemplateFile//用于复制文件句柄

);

* SetFileTime()函数

Long SetFileTime(

Long hFile,//文件句柄

FILETIME lpCreationTime,//文件创建时间

FILETIME lpLastAccessTime,//文件最后一次访问时间

FILETIME lpLastWriteTime//文件最后一次修改时间

);

* SetFileAttributes()函数

BOOL SetFileAttributes(

LPCTSTR lpFileName, //文件路径

DWORD dwFileAttributes // 文件属性);

* BOOL WriteFile()函数

BOOL WriteFile(

　　HANDLE hFile, // 文件句柄

　 LPCVOID lpBuffer, // 数据缓存区指针

　　DWORD nNumberOfBytesToWrite, // 写入字节数

　　LPDWORD lpNumberOfBytesWritten, // 指向实际写入字节数

　　LPOVERLAPPED lpOverlapped // 结构体指针，一般为NULL);

* BOOL ReadFile()函数

BOOL ReadFile(

HANDLE hFile, //文件的句柄

　　LPVOID lpBuffer, //用于保存读入数据的一个缓冲区

　　DWORD nNumberOfBytesToRead, //要读入的字节数

　　LPDWORD lpNumberOfBytesRead, //实际读取字节数的指针

　　LPOVERLAPPED lpOverlapped //一般置为NULL);

③ 程序的设计和实现

* 主函数的设计与实现

在主函数中，将从命令行中接收三个参数，第一个(argv[0])是将要执行的可执行程序路径，第二个(argv[1])是等待被复制的文件路径(此后统一称为旧路径)，第三个(argv[2])是将把旧文件复制到的目标路径(此后同一称为新路径)。

主函数(main函数)设计流程如下：

(1)在主函数中，首先判断输入的参数是否为3个，如果是则转(2)，否则退出执行。

(2)调用FindFIrstFile(argv[1], &lpFindData)打开argv[1]路径下的旧文件，返回句柄hFindFile，并将其信息保存到lpFindNewData结构体中。如果成功打开则转(3)，否则退出执行。

(3)调用FindFIrstFile(argv[2], &lpFindNewData)检查是否存在路径为argv[2]的文件，如果存在则退出执行，否则调用CreateDirectory(argv[2],NULL)在argv[2]路径上创建新文件。

(4)调用MyCp()递归函数复制就文件夹中的子文件。

(5)递归函数返回后，调用CreateFile()函数打开argv[2]路径上的新文件，返回句柄hDirFile，并调用SetFileTime()函数修改其时间属性，调用SetFileAttributes()函数设置文件属性。

(6)关闭句柄hFindFile以及hDirFile，将其值设置为INVALID\_HANDEL\_VALUE。

* MyCp(char\* OldPath, char\* NewPath)函数设计与实现

在此函数中，传入的参数OldPath为旧文件路径(源文件)，NewPath为新文件路径(目标文件)。此外，设置char\* 型的参数new\_path和old\_path，用于拼接目录中的子文件路径。WIN32\_FIND\_DATA型结构体lpFindData，句柄hFindData，BOOL型参数hFound。MyCp()函数流程如下：

(1)定义参数old\_path，new\_path，结构体lpFindData，句柄hFindData，hFound。

(2)通过lstrcpy(str1,str2)接口对old\_path，new\_path赋值，其值分别为OldPath，NewPath。

(3)调用FinFirstFile(old\_path,&lpFindData)查找old\_path路径下的文件并返回句柄hFindData，将文件信息存于lpFindData之中。

(4)如果hFindData值为INVALID\_HANDLE\_VALUE，则表示查找文件失败，程序退出执行；否则转(5)。

(5)调用hFIndData(hFinData,&lpFindData)查找OldPath路径上的文件夹中的下一个文件，并返回hFound(表示是否成功查找)，若成功，则将文件信息存入lpFindData且转(6)，否则说明已遍历当前文件夹中的最后一个文件。

(6)lpFindData.cFilename中存放当前将复制文件的相对路径，调用lstrcpy()。

(7)调用IsChildDir(lpFindData)判断当前文件是否为子目录文件。

(8)如果不为目录文件，则为其他文件，调用CopyFile()函数完成文件的复制。

(9)关闭句柄hFindData，设置其值为INVALID\_HANDLE\_VALUE，执行结束并返回。

* CopyFile(char\* OldPaht, char\* NewPath)的设计与实现

(1)定义句柄hOldFile(指向旧文件的句柄)、hNewFile(指向新文件的句柄)，WIN32\_FIND\_DATA型参数lpFindData(存放文件信息)。

(2)调用CreateFile()打开OldPath路径下的旧文件，返回句柄hOldFile。

(3)调用CreateFile()在NewPath路径下创建新文件，返回句柄hNewFile。

(4)如果hOldFile和hNewFile不是无效值，转(5)，否则退出执行。

(5)循环调用ReadFile()从旧文件中读并且调用WriteFile()函数往新文件中写，设置每次读写的大小BUFSIZE为1024，记录每次实际读写的大小dwxfer，当BUFSIZE不等于dwxfer时，说明复制结束，退出循环。

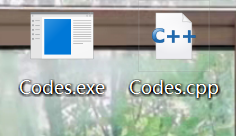
(6)调用SetFileTime()函数设置文件时间属性，调用SetFileAttributes()函数设置文件属性。

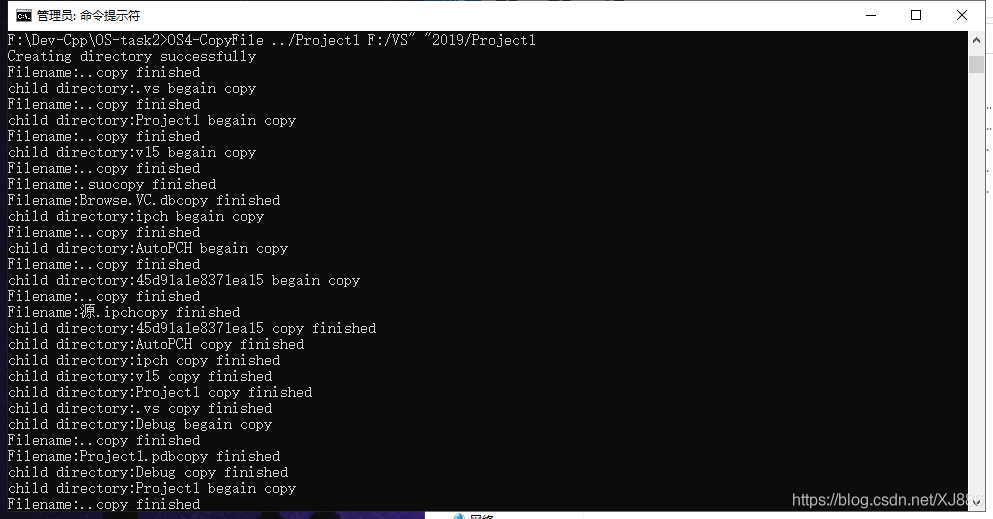
(7)关闭文件句柄hNewFile、hOldFile，设置其值为INVALID\_HANDLE\_VALUE。

1. **编写、编译代码**

在Dev-Cpp 小熊猫.C++.0.14.2.64位.10.3.安装版中编写C++代码，实现相应的文件复制功能。

1. **运行得到结果**

****

****

四、实验结果及分析

通过本次实验，我掌握了Windows操作系统下对文件和目录操作的系统调用的使用，理解了其原理，成功实现了文件复制功能。理解和掌握了文件系统的实现功能。

在完成实验的过程中，有一个需要额外注意的地方，那就是在获得新文件后，我们需要调用SetFileAttributes()函数完成对文件存取权限的设置，修改新文件的权限，以便后续操作。

五、实验收获与体会

“文件复制”是操作系统知识里非常重要的部分，通过本次实验，我进一步加深了对操作系统中文件系统和系统调用相关内容的理解，深刻了解到文件复制的过程。

通过本实验，我掌握了关于Windows的文件系统的相关知识，学习了文件系统的系统调用功能的使用，同时对文件系统如何管理保存在磁盘、光盘等存储介质上的信息有了更加深入的了解。

在亲自动手实验的过程中，我对本章知识点有了更深刻的理解，同时也复习了程序设计方面的知识，这次实验让我收获颇多！

附录：程序清单及说明