操作系统课程设计实验报告

实验名称	复制文件		
学号	1120180207	姓名	唐小娟

1. 实验目的

1. 了解 Windows 关于复制文件和目录的系统调用
2. 了解 Linux 关于复制文件和目录的系统调用
3. 了解 Linux 关于符号链接文件的相关操作。

二、实验内容

1. Windows 系统

(1). 文件复制

(2). 复制文件时间

2. Linux 系统

(1). 文件复制

(2). 复制文件时间

(3). 复制符号链接文件

三、实验环境及配置方法

操作系统：Windows 10，Ubuntu 20.04，Linux 5.4.0-42

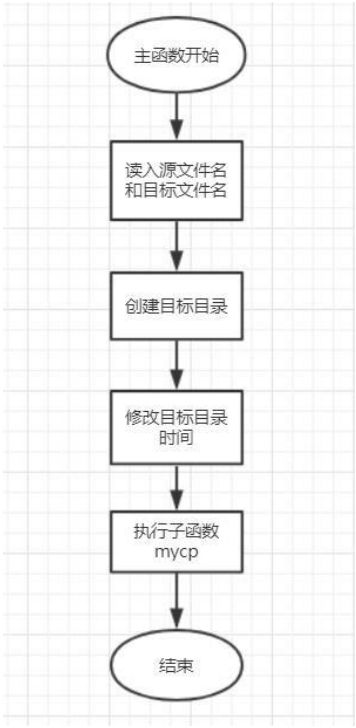
集成开发环境：Microsoft VS Code

编译器：gcc 9.3.0

四、 实验方法和实验步骤（程序设计与实现）

1. Windows 系统

(1). 主函数控制流程图如下：



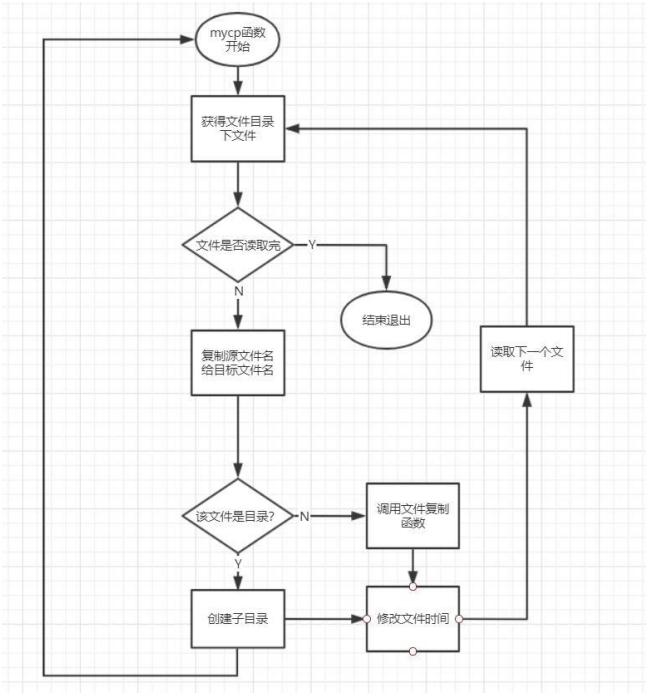
首先判断传入参数是否为 3，再调用 FindFirstFile 看源文件目录是否存在，若不存在，提示错误信息后退出，之后，判断目标目录是否存在，如果不存在则调用 CreateDirectory 创建目标目录，修改目录创建时间，最后调用 mycp 函数进行文件复制操作。

相关函数说明：

```
//查找指定文件路径的文件
HANDLE FindFirstFile(
 LPCTSTR lpFileName, //文件路径名
 LPWIN32_FIND_DATA lpFindFileData //指向一个用于保存文件信息的结构体
);
WIN32_FIND_DATA结构体如下:
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];      // 长文件名
TCHAR cAlternateFileName[14];   // 8.3格式文件名
} WIN32_FIND_DATA, *PWIN32_FIND_DATA;
```

```
//创建一个新的文件夹
BOOL CreateDirectory(
LPCTSTR lpPathName,                             //文件夹名称
LPSECURITY_ATTRIBUTES lpSecurityAttributes      //安全属性，一般设置为NULL即可
);
```

(2). mycp 函数控制流程图：



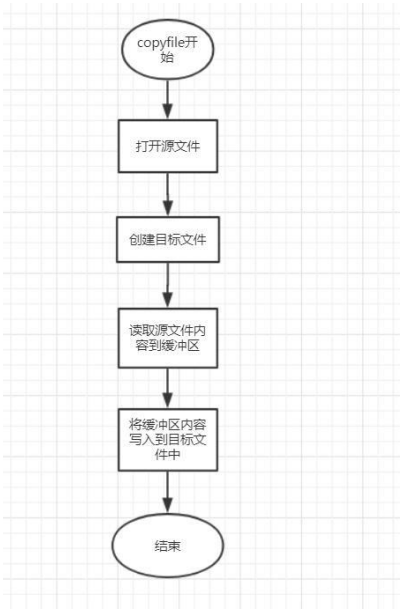
遍历目录下的所有文件，根据文件属性判断该文件是目录文件还是普通文件，如果是普通文件调用 copyfile 函数，若是目录文件，则递归调用 mycp 函数，之后调用 changetime 修改时间。

相关函数说明：

```
//查找下一个文件
BOOL FindNextFile(
HANDLE hFindFile,                   //搜索的文件句柄
LPWIN32_FIND_DATA lpFindFileData //指向一个用于保存文件信息的结构体
);
```

```
BOOL SetFileTime(
 HANDLE hFile,
 const FILETIME *lpCreationTime,    //创建时间
 const FILETIME *lpLastAccessTime, //最近访问时间
 const FILETIME *lpLastWriteTime    //最后修改时间
);
```

(3). copyfile 函数控制流程图



通过调用 CreateFile 打开源文件、创建目标文件。调用 ReadFile 读取源文件的内容到缓冲区，再调用 WriteFile 将缓冲区内容写入目标文件中。

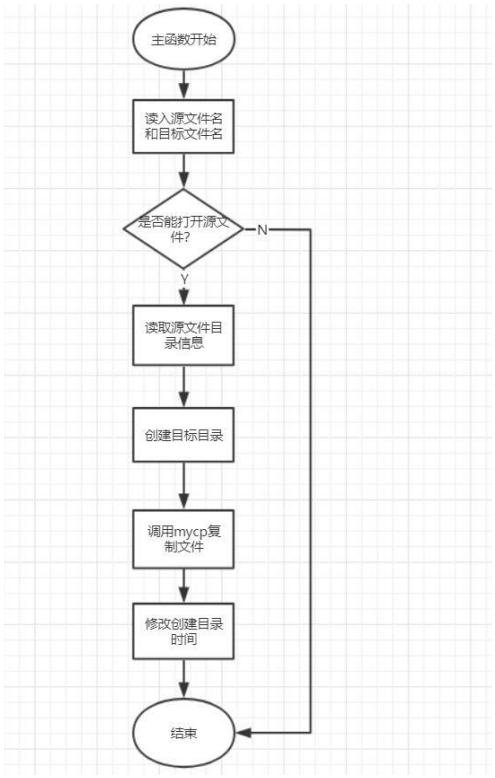
相关函数说明：

```
BOOL ReadFile(
 HANDLE hFile,
 LPVOID lpBuffer,
 DWORD nNumberOfBytesToRead,
 LPDWORD lpNumberOfBytesRead, //指向实际读取字节数的指针
 LPOVERLAPPED lpOverlapped
);
```

```
BOOL WriteFile(
HANDLE hFile,                    //文件句柄
LPCVOID lpBuffer,                //数据缓存区指针
DWORD nNumberOfBytesToWrite,     //要写的字节数
LPDWORD lpNumberOfBytesWritten,  //用于保存实际写入字节数的存储区域的指针
LPOVERLAPPED lpOverlapped        //OVERLAPPED 结构体指针
);
```

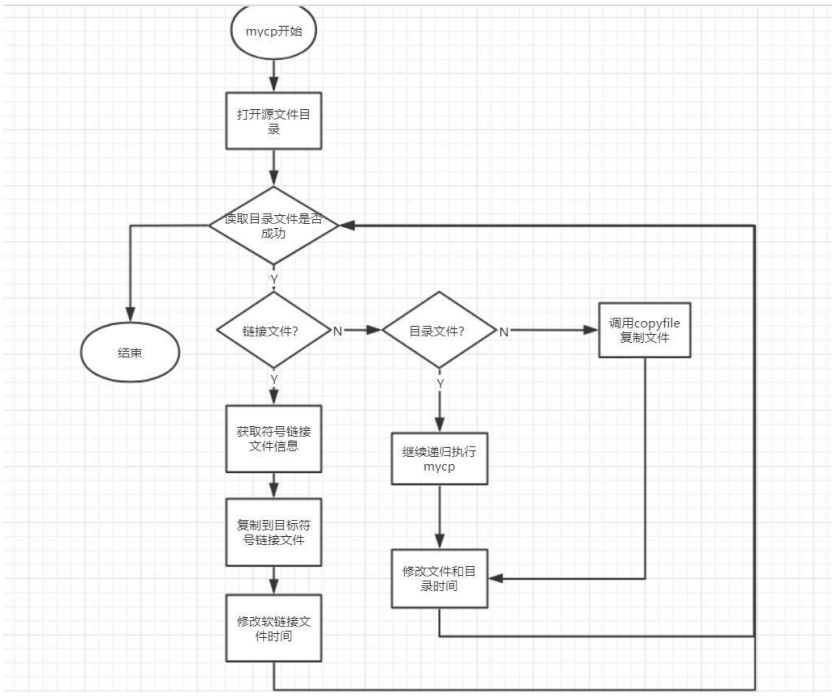
2. Linux 系统

(1). 主函数流程



先判断参数读入个数是否正确，再看源文件是否能成功打开。之后调用 lstat 函数得到源文件信息，根据目标文件路径和源文件信息调用 create 函数创建目标文件目录，接着调用 mycp()进行文件复制，最后修改创建目录时间。

(2)．mycp()流程



打开源文件，循环读取目录下的所有文件。根据文件属性分别执行不同的代码：

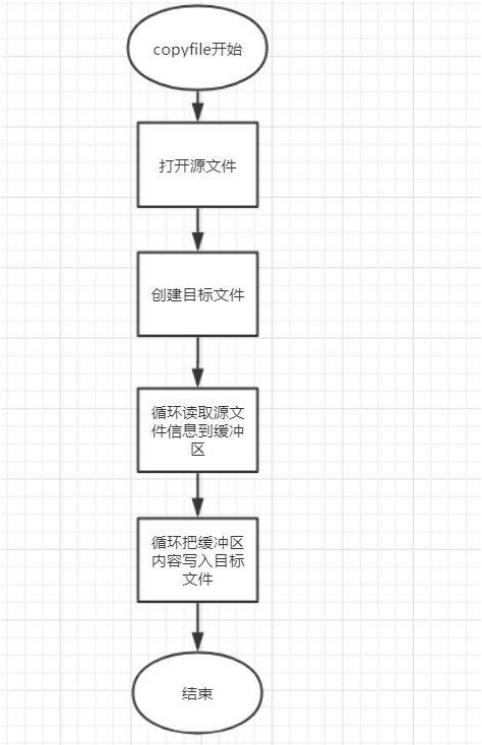
如果是链接文件，调用 readlink 读取源软链接文件中的数据，调用 symlink 创建新的软链接文件，之后调用 lutimes 修改软链接文件的时间戳；

如果是目录文件，创建子目录后，递归调用 mycp 函数继续复制；

如果都不是以上文件，则直接调用 copyfile 函数复制文件信息。

最后，修改文件的时间戳，结束。

(3). copyfile 流程



首先调用 lstat 得到源文件的信息到结构体 stat 中，接着打开源文件，创建目标文件，将源文件的信息以 4096 个单位循环 read 到缓冲区 buffer 中，再将缓冲区的内容循环写入到目标文件中，关闭文件句柄后，copyfile 结束。

相关结构体说明：

```
struct stat {
dev_t st_dev; /* 文件所在设备的标识 */
ino_t st_ino; /* 文件结点号 */
mode_t st_mode; /* 文件保护模式 */
nlink_t st_nlink; /* 硬连接数 */
uid_t st_uid; /* 文件用户标识 */
gid_t st_gid; /* 文件用户组标识 */
dev_t st_rdev; /* 文件所表示的特殊设备文件的设备标识 */
off_t st_size; /* 总大小，单位为字节*/
blksize_t st_blksize; /* 文件系统的块大小 */
blkcnt_t st_blocks; /* 分配给文件的块的数量，512 字节为单元 */
time_t st_atime; /* 最后访问时间 */
time_t st_mtime; /* 最后修改时间 */
time_t st_ctime; /* 最后状态改变时间 */
};
```

```
struct dirent
{
   long d_ino;               /* inode number 索引节点号 */
   off_t d_off;              /* offset to this dirent 在目录文件中的偏移 */
   unsigned short d_reclen;  /* length of this d_name 文件名长 */
   unsigned char d_type;     /* the type of d_name 文件类型 */
   char d_name [NAME_MAX+1]; /* 文件名，最长255字符 */
 }
```

五、实验结果和分析

1. Windows 系统文件复制

```
D:\>mycp test target
copying from test to target
copy success!

D:\>dir test
 驱动器 D 中的卷是 Data
 卷的序列号是 88C9-A2F6

 D:\test 的目录

2021/01/06  13:44    <DIR>          .
2021/01/06  13:44    <DIR>          ..
2021/01/06  13:43                20 hello.txt
2020/11/05  22:18            29,568 picture.jpg
2021/01/06  13:44    <DIR>          txx
               2 个文件         29,588 字节
               3 个目录 293,607,895,040 可用字节

D:\>dir target
 驱动器 D 中的卷是 Data
 卷的序列号是 88C9-A2F6

 D:\target 的目录

2021/01/06  13:44    <DIR>          .
2021/01/06  13:44    <DIR>          ..
2021/01/06  13:43                20 hello.txt
2020/11/05  22:18            29,568 picture.jpg
2021/01/06  13:44    <DIR>          txx
               2 个文件         29,588 字节
               3 个目录 293,607,895,040 可用字节
```

```
D:\>dir test\txx
 驱动器 D 中的卷是 Data
 卷的序列号是 88C9-A2F6

 D:\test\txx 的目录

2021/01/06  13:44    <DIR>          .
2021/01/06  13:44    <DIR>          ..
2020/11/20  14:34         1,724,804 思想汇报2019年第四季度.docx
2020/11/24  16:24            16,876 思想汇报2020第四季度.docx
2020/12/23  12:58            14,603 睿信书院2020年学生荣誉统计（2020.1-2020.12）.xlsx
               3 个文件      1,756,283 字节
               2 个目录 293,607,841,792 可用字节

D:\>dir target\txx
 驱动器 D 中的卷是 Data
 卷的序列号是 88C9-A2F6

 D:\target\txx 的目录

2021/01/06  13:44    <DIR>          .
2021/01/06  13:44    <DIR>          ..
2020/11/20  14:34         1,724,804 思想汇报2019年第四季度.docx
2020/11/24  16:24            16,876 思想汇报2020第四季度.docx
2020/12/23  12:58            14,603 睿信书院2020年学生荣誉统计（2020.1-2020.12）.xlsx
               3 个文件      1,756,283 字节
               2 个目录 293,607,841,792 可用字节
```

由上图可以看到调用 mycp 将源文件 test 复制到了 target，显示 copy success! 之后输入 dir test 和 dir target 可以看到二者目录下的文件内容与时间完全一致，调用 dir test\txx 和 dir target\txx 可以看到子目录下的内容和时间也完全一致，该文件复制成功。

2. Linux 系统文件复制

```
ostxj@ostxj-virtual-machine:~/studycodes/vscode$ ./mycp test target
Start copyingcopy from test/txx/思想汇报2019年第四季度.docx to target/txx/思想汇报2019年第四季度.docx
copy from test/txx/思想汇报2020第四季度.docx to target/txx/思想汇报2020第四季度.docx
copy from test/txx/睿信书院2020年学生荣誉统计（2020.1-2020.12）.xlsx to target/txx/睿信书院2020年学生荣誉统计（2020.1-2020.12）.xlsx
copy from test/txx to target/txx
copy from test/hellolink to target/hellolink
copy from test/hello.txt to target/hello.txt
copy from test/picture.jpg to target/picture.jpg
Copy success!
ostxj@ostxj-virtual-machine:~/studycodes/vscode$ ls -l test
total 40
lrwxrwxrwx 1 ostxj ostxj     9 1月   6 21:48 hellolink -> hello.txt
-rwxrw-rw- 1 ostxj ostxj    20 1月   6 13:43 hello.txt
-rwxrw-rw- 1 ostxj ostxj 29568 11月  5 22:18 picture.jpg
drwxrwxrwx 2 ostxj ostxj  4096 1月   6 13:44 txx
ostxj@ostxj-virtual-machine:~/studycodes/vscode$ ls -l target
total 40
lrwxrwxrwx 1 ostxj ostxj     9 1月   6 21:48 hellolink -> hello.txt
-rwxrw-r-- 1 ostxj ostxj    20 1月   6 13:43 hello.txt
-rwxrw-r-- 1 ostxj ostxj 29568 11月  5 22:18 picture.jpg
drwxrwxr-x 2 ostxj ostxj  4096 1月   6 13:44 txx
```

由上图可以得知，源文件 test 中的所有文件都已经被复制到了目标文件 target 中，子目录的复制已经成功，同时符号链接文件 hellolink 也成功复制到了目标目录中成为一个新的符号链接文件，这些所有文件的时间也一致相同。

六、讨论、心得

通过这次试验，我明白了怎么样在 Linux 系统和 Windows 系统中获取文件信息，了解了大量操作文件的系统调用，在以后的文件管理中，我认为我可以学以致用，写一个小小的程序批量处理大量不需要的文件，节省人力。

在 Linux 实验过程中，我调用 utime 直接修改所有文件的时间戳，在实验结果中发现软链接的创建时间并没有修改成功，于是我经过翻阅资料，得知应该调用 lutimes 去修改。