操作系统课程设计 实验报告

姓名：刘天祺

班级：07121502

学号：1320151097

学院：计算机学院

专业：物联网工程

日期：2018 年 4 月 10 日

实验五　复制文件

一、实验要求

完成一个目录复制命令 mycp，包括目录下的文件和子目录，运行结果如下：

```
beta@bugs.com [~/]# ls -l sem
total 56
drwxr-xr-x  3 beta beta 4096 Dec 19 02:53 ./
drwxr-xr-x  8 beta beta 4096 Nov 27 08:49 ../
-rw-r--r--  1 beta beta  128 Nov 27 09:31 Makefile
-rwxr-xr-x  1 beta beta 5705 Nov 27 08:50 consumer*
-rw-r--r--  1 beta beta  349 Nov 27 09:30 consumer.c
drwxr-xr-x  2 beta beta 4096 Dec 19 02:53 subdir/
beta@bugs.com [~/]# mycp sem target
beta@bugs.com [~/]# ls -l target
total 56
drwxr-xr-x  3 beta beta 4096 Dec 19 02:53 ./
drwxr-xr-x  8 beta beta 4096 Nov 27 08:49 ../
-rw-r--r--  1 beta beta  128 Nov 27 09:31 Makefile
-rwxr-xr-x  1 beta beta 5705 Nov 27 08:50 consumer*
-rw-r--r--  1 beta beta  349 Nov 27 09:30 consumer.c
drwxr-xr-x  2 beta beta 4096 Dec 19 02:53 subdir/
```

- 实现 Windows 版本和 Linux 版本
- 目录拷贝时需要支持多级目录（子目录）的拷贝
- 支持 Linux 里的 soft link 和 windows 中的快捷方式拷贝

二、实验环境

2.1 Linux 环境

操作系统：Ubuntu 14.04.5 LTS 64bit

Shell：zsh 5.0.5 (x86_64-pc-linux-gnu)

编译器：gcc 4.8.5 (Ubuntu 4.8.5-2ubuntu1~14.04.1)

2.2 Windows 环境

操作系统：Windows 10 64bit

Shell：cmder 160710

编译器：gcc 3.4.5 (mingw-vista special r3)

三、实验步骤

3.1 在 Linux 环境实现

- 程序流程如下：

```
int main(int argc,char *argv[]) {
    struct stat statbuf;
    DIR *dir;
    // usage err
    if(argc != 3) {
        usage();
    }
    // src file is not a folder or src is a symbolic link
    lstat(argv[1],&statbuf);
    if((dir = opendir(argv[1])) == NULL || S_ISLNK(statbuf.st_mode) ) {
        file_err(argv[1]);
    }
    // if dest folder does not exist, create it
    if((dir = opendir(argv[2])) == NULL) {
        mkdir(argv[2],statbuf.st_mode);
    }
    // call mycp
    cp(argv[1], argv[2]);
    return 0;
}
```

判断程序参数是否合法后，使用 mkdir()创建目标目录，随后调用 cp 函数。

在 cp 函数中 ,使用 opendir()函数打开待拷贝的文件目录 ,并使用 readdir()遍历读取该文件夹下的文件名，代码如下：

```
dir = opendir(src);
while((entry = readdir(dir)) != NULL) {
```

- 对于每一个遍历到的文件：

通过 d_name 属性判断文件类型，首先判断其是否为符号链接文件，其次判断其是否为目录文件。

若为符号链接文件，则调用 readlink()函数读取原符号链接文件指向的文件名 ,然后调用 symlink()函数在新的目录下创建新的符号链接文件 ,代码如下：

```
// cp symlink file
if(entry->d_type == DT_LNK) {
    strcat(src, "/");
    strcat(src, entry->d_name);
    strcat(dest, "/");
    strcat(dest, entry->d_name);

    char buf[512];
    memset(buf, 0, sizeof(buf));        // init buf with '\0'
    readlink(src, buf, sizeof(buf));    // filename of symlink file point to
    symlink(buf, dest);                 // create symlink file

    strcpy(src, src_d);                 // back to previous directory
    strcpy(dest, dest_d);
}
```

若为目录文件，则调用 stat()函数获取待拷贝文件的信息(st_mode)，并调用 mkdir()函数创建新的目录文件，然后对新的目录(子目录)递归调用拷贝函数，代码如下：

```
// cp directory file
else if(entry->d_type == DT_DIR) {
    strcat(src, "/");
    strcat(src, entry->d_name);     // append filename of subdir
    strcat(dest, "/");
    strcat(dest, entry->d_name);

    stat(src, &statbuf);
    mkdir(dest, statbuf.st_mode);
    cp(src, dest);                  // cp subdir (recursive)

    strcpy(src, src_d);
    strcpy(dest, dest_d);
}
```

若文件既非符号链接文件也非目录文件，则调用 cp_file()函数进行文件拷贝

```
// cp other type file
else {
    strcat(src, "/");
    strcat(src, entry->d_name);     // append filename
    strcat(dest, "/");
    strcat(dest, entry->d_name);

    cp_file(src, dest);

    strcpy(src, src_d);             // back to previous directory
    strcpy(dest, dest_d);
}
```

cp_file()函数中通过 open()打开待拷贝文件，通过 stat()获取待拷贝文件信息(st_mode)，通过 creat()函数依据待拷贝文件的 st_mode 创建新文件，以保证二者权限的一致性，打开/创建完毕后，调用 read()，write()函数读

取待拷贝文件的内容，并写入新文件，代码如下：

```
void cp_file(char *src,char *dest) {
    int fd = open(src, O_RDONLY);
    int fd2; // fd of dest file
    char buf[1024];
    int len;
    struct stat statbuf;
    stat(src, &statbuf);
    fd2 = creat(dest, statbuf.st_mode);
    while((len = read(fd, buf, 1024)) > 0) {
        write(fd2, buf, len);
    }
    close(fd);
    close(fd2);
}
```

3.2 在 Windows 环境实现

- 程序流程大体上与 linux 相似，代码如下：

```
int main(int argc, char *argv[]) {
    WIN32_FIND_DATA lpFindFileData;
    if (argc != 3) {
        Usage();
    }
    else {
        if (FindFirstFile(argv[1], &lpFindFileData) == INVALID_HANDLE_VALUE) {
            FileError(argv[1]);
        }
        if (FindFirstFile(argv[2], &lpFindFileData) == INVALID_HANDLE_VALUE) {
            CreateDirectory(argv[2], NULL);
            //printf("CreateDirectory: %s\n", argv[2]);
        }
        Mycp(argv[1], argv[2]);
    }
    return 0;
}
```

判断程序参数是否合法后，使用 CreateDirectort()函数创建目标目录，随后调用 Mycp 函数。

在 Mycp 函数中，通过 WIN32_FIND_DATA 类型结构 lpFindFileData 存储文件信息，通过 FindFirstFile()打开待拷贝文件目录，并通过 FindNextFile()遍历目录下的文件，代码如下：

```
HANDLE hfind = FindFirstFile(src, &lpFindFileData);
if (hfind != INVALID_HANDLE_VALUE) {
    while (FindNextFile(hfind, &lpFindFileData) != 0) {
```

- 对于每一个遍历到的文件：

  通过 dwFileAttributes 属性判断其是否为目录文件。

  若为目录文件，则调用 CreateDirectory()函数创建新的目录文件，然后对新的目录(子目录)递归调用 Mycp()函数，代码如下：

```
// cp directory
if (lpFindFileData.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY) {
    if ((strcmp(lpFindFileData.cFileName, ".") == 0) ||
         (strcmp(lpFindFileData.cFileName, "..") == 0)) {
        continue;
    }
    else{
        memset(src, '0', sizeof(src));  // append filename
        strcpy(src, src_d);
        strcat(src, "\\");
        strcat(src, lpFindFileData.cFileName);
        strcat(dest, lpFindFileData.cFileName);

        CreateDirectory(dest, NULL);
        Mycp(src, dest);                // cp subdir (recursive)

        strcpy(src, src_d);             // back to previous directory
        strcat(src, "\\");
        strcpy(dest, dest_d);
        strcat(dest, "\\");
    }
}
```

  若文件非目录文件，则调用 CpFile()函数进行文件拷贝，代码如下：

```
// cp file
else {
    memset(src, '0', sizeof(src));
    strcpy(src, src_d);
    strcat(src, "\\");
    strcat(src, lpFindFileData.cFileName);
    strcat(dest, lpFindFileData.cFileName);

    CpFile(src, dest);
    strcpy(src, src_d);
    strcat(src, "\\");
    strcpy(dest, dest_d);
    strcat(dest, "\\");
}
```

  CpFile()函数中通过 CreateFile()打开待拷贝文件、创建新文件，打开/创建完毕后，调用 ReadFile()，WriteFile()函数读取待拷贝文件的内容，并写入新文件。

四、实验结果

4.1 在 Linux 环境下的实验结果

```
$ ls -l sem target
sem:
total 52
lrwxrwxrwx 1 taqini taqini    8 Apr  9 22:49 cons -> consumer
-rwxr-xr-x 1 taqini taqini 8512 Apr  9 22:14 consumer
-rw-r--r-- 1 taqini taqini   13 Apr  9 22:13 consumer.c
-rw-r--r-- 1 taqini taqini   44 Apr  9 22:13 Makefile
drwxr-xr-x 2 taqini taqini 4096 Apr  9 22:14 subdir

target:
total 52
lrwxrwxrwx 1 taqini taqini    8 Apr 10 16:15 cons -> consumer
-rwxr-xr-x 1 taqini taqini 8512 Apr 10 16:15 consumer
-rw-r--r-- 1 taqini taqini   13 Apr 10 16:15 consumer.c
-rw-r--r-- 1 taqini taqini   44 Apr 10 16:15 Makefile
drwxr-xr-x 2 taqini taqini 4096 Apr 10 16:15 subdir
```

```
$ tree sem target
sem
├── cons -> consumer
├── consumer
├── consumer.c
├── Makefile
└── subdir
    ├── 12
    ├── 3
    └── 45
target
├── cons -> consumer
├── consumer
├── consumer.c
├── Makefile
└── subdir
    ├── 12
    ├── 3
    └── 45

2 directories, 14 files
```

4.2 在 Windows 环境下的实验结果

```
H:\os_ep\ep5\windows (master)
λ dir sem target
 Volume in drive H is Boot-Repair
 Volume Serial Number is B4FE-5315

 Directory of H:\os_ep\ep5\windows\sem

04/10/2018  05:52 PM    <DIR>          .
04/10/2018  05:52 PM    <DIR>          ..
04/10/2018  05:43 PM                13 consumer.c
04/10/2018  05:43 PM                44 Makefile
04/10/2018  05:43 PM             8,512 consumer
04/10/2018  05:43 PM    <DIR>          subdir
               3 File(s)          8,569 bytes

 Directory of H:\os_ep\ep5\windows\target

04/10/2018  06:02 PM    <DIR>          .
04/10/2018  06:02 PM    <DIR>          ..
04/10/2018  06:02 PM                13 consumer.c
04/10/2018  06:02 PM                44 Makefile
04/10/2018  06:02 PM             8,512 consumer
04/10/2018  06:02 PM    <DIR>          subdir
               3 File(s)          8,569 bytes
               3 Dir(s)  40,986,492,928 bytes free
```

五、实验总结

通过实验理解了 linux 和 windows 操作系统如何对文件进行操作。在 linux 操作系统中，符号链接文件比较特殊，若将其视作常规文件进行拷贝，则会拷贝其所指向的文件，而不是符号链接文件本身。同时若符号链接文件指向的是一个目录文件，则在判断其类型的时候，它既是符号链接文件，又是目录文件，所以在实验中，为了避免符号链接文件被当作目录文件处理，在判断文件是目录文件之前先判断它是否为符号链接文件。