CSCI4061 Lab5

directories, (l)stat, and links

Lab overview

In this lab exercise:

1. In main(), create a new text file, and two corresponding hard and symbolic link of the newly created file (Check Makefile).
2. Traverse D1 directory recursively, print out all filenames and folders/directories. If a symbolic link is found, print “Symlink found: path/to/your/symlink.txt”
3. Copy output to output.txt

You are going to learn:

opendir(), readdir(), link(), symlink(), lstat()

opendir(), closedir()

When we want to take a look at directory entry, we need to open a directory first.
Don't forget to close the directory before program return.

```
#include <sys/types.h>
#include <dirent.h>
/* Open a directory stream given directory name.
   Return an pointer to directory stream, or NULL if it could not be opened.*/
extern DIR *opendir (const char *dir_name);

/* This is the data type of directory stream objects.
   The actual structure is opaque to users.*/
typedef struct __dirstream DIR;

/*return 0 for success close, -1 for error*/
int closedir(DIR *dirp);
```

readdir()

- After we get a directory stream, we use readdir(). It returns a pointer to dirent struct which contains information about directory/files and directory entry.
- Calling readdir recursively on a same directory stream could give us information about each files or sub directories
- Check p1.c

```
#include <dirent.h>

struct dirent* readdir(DIR* dirp)

struct dirent {
    ino_t          d_ino;       /* inode number */
    off_t          d_off;       /* offset to the next dirent */
    unsigned short d_reclen;    /* length of this record */
    unsigned char  d_type;      /* type of file; not supported
                                   by all file system types */
    char           d_name[256]; /* filename */
};
```

getcwd() and chdir()

- Dirent entry will only contain directory name and not the entire path. Use getcwd() to get the absolute path of current directory
  - Useful when switching between directories

- Sometimes, it may be necessary to change directories while code execution. chdir() can be used for this. Ensure to return back to original directory.

- Both functions return the path on success and NULL on failure

- Check p2.c

```
#include <unistd.h>
/*Get absolute path of current directory in buf*/
char *getcwd(char *buf, int bufsize);

/*Change directory to path*/
char *chdir(char *path);
```

link(), symlink()

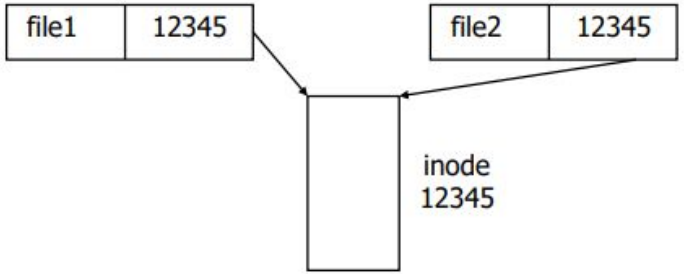
- The link()/symlink() function shall create a new link (directory entry) for the existing target file.
- Upon successful completion, it shall return 0. Otherwise, return -1 and set errno to indicate the error.

```
#include <unistd.h>
/*Generates hard link*/
int link(const char *target_file_path, const char *hard_link_path);
/*Generates symbolic link*/
int symlink(const char *target_file_path, const char *symb_link_path);
```

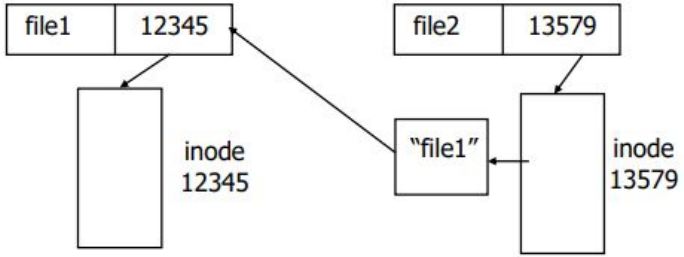
link(), symlink()

1. Run p3.c using make lab5_p3 to generate hard link and symbolic link file for D2/b.txt
2. Try to modify each file and see what happened between them
3. Delete b.txt in the D2 folder, then revert delete action.
4. Do step 2, then see what happened.

- File is deleted only when all hard links are deleted



- A symbolic link is different from original file
  - If original file is moved or deleted, the link becomes hanging



lstat()

Gets attributes about file/directories and store it into buffer. The stat structure contains a lot of information about a file/directory. Run p4 to see those of D1/a.txt

You can also change path to any file you want to lookup.

Using buf.st_mode, one can identify is a directory entry is symlink (next page)

```c
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
int lstat(const char *path, struct stat *buf);
struct stat {
    mode_t st_mode;    // File type and permissions
    ino_t st_ino;      // Inode number
    dev_t st_dev;      // Device ID of filesystem
    // ... (other fields)
};
```

lstat()

```
#include <sys/stat.h>
struct stat {
    mode_t st_mode;    // File type and permissions
    ino_t st_ino;      // Inode number
    dev_t st_dev;      // Device ID of filesystem
    // ... (other fields)
};
// assume m is st_mode
S_ISBLK(m)  // True for a block special file.
S_ISCHR(m)  // True for a character special file.
S_ISDIR(m)  // True for a directory.
S_ISFIFO(m) // True for a pipe or FIFO special file.
S_ISREG(m)  // True for a regular file.
S_ISLNK(m)  // True for a symbolic link.
S_ISSOCK(m) // True for a socket.
```

Expected output

```
$ ./main D1/f1.txt D1/D2/f1_hlink.txt D1/D4/f1_slink.txt D1
/home/csci4061/Lab5/D1/f1.txt
/home/csci4061/Lab5/D1/D2
/home/csci4061/Lab5/D1/D2/f1_hlink.txt
/home/csci4061/Lab5/D1/D2/D3
/home/csci4061/Lab5/D1/D2/D3/f3.txt
/home/csci4061/Lab5/D1/D2/f2.txt
/home/csci4061/Lab5/D1/D4
Symlink found: /home/csci4061/Lab5/D1/D4/f1_slink.txt
/home/csci4061/Lab5/D1/D4/D6
/home/csci4061/Lab5/D1/D4/D6/f6.txt
```

Submission

1. PA1 final submission:
    a. Due at Oct 11 at 11:59PM.
    b. Late Due Date: Oct 12 at 11:59PM

2. Lab5 submission:
    a. Due at Oct 10 at 11:59PM
    b. Upload your dir_trav.c, and output.txt **only** to Gradescope