Fundamentos de Sistemas de Operação

MIEI 2017/2018

Laboratory session 8

Objectives

Input/output (I/O) system [1] – the file system. Implementation of the *ls* command to list a directory's contents.

Listing the contents of a directory

The goal of this work is to implement a *simplified version* of the *ls* command. This command lists orderly the contents of a directory (including hidden files) received as argument or of the *current directory*, if none is given. The path to the directory may either be an *absolute path* starting at the root directory '/' or a *relative path* to *the current directory*. In the latter, the name may either be the current directory './' or one of its sub-directories, ('../' or other). Some examples of the *ls* command with options *–l*, *–a* and *–i* may be:

```
$ ls -ali /home/user/mydir
(...)
  422142   drwxrwx---  4 user user 4096 Nov 12 21:29 .
  371048   drwxrwxr-x 10 user user 4096 Nov 12 21:03 ..
  422152   -rw-rw-r--  1 user user 0 Nov 12 21:29 empy.txt
  422146   drwxrwx---  2 user user 4096 Nov 13 00:11 shared
  395362   drwxrwxr-x  3 user user 4096 Nov 12 21:55 subdir1
(...)
$ ls -al
(...)
drwxrwx---  4 user user 4096 Nov 12 21:29 .
drwxrwxr-x 10 user user 4096 Nov 12 21:03 ..
-rw-rw-r--  1 user user 0 Nov 12 21:29 empy.txt
drwxrwx---  2 user user 4096 Nov 13 00:11 shared
drwxrwxr-x  3 user user 4096 Nov 12 21:55 subdir1
(...)
$ ls -ali subdir1
(...)
  395362   drwxrwxr-x 3 user user 4096 Nov 12 21:55 .
  422142   drwxrwx--- 4 user user 4096 Nov 12 21:29 ..
  395362   drwxrwxr-x 2 user user 4096 Nov 12 21:56 subdir2
  422151   -rwxrwx--- 1 user user 1247 Nov 12 21:29 tmp.c
```

Your program is supposed to show the follow information and meta-information for a directory:

- Lists the names and types of all the files (regular files and sub-directories)
  - In case of a directory, the program prints *"(dir)"*
  - In case of a regular file, the program prints its size *"(size)"*
- Lists the inode number, the owner id, and the time of the last modification of a file

For instance, assuming that your program is named *myls*, the simplified outputs of the examples above are

```
$ ./myls /home/user/mydir
395362: 1000 21:55 subdir1 (dir)
371048: 1000 21:3 .. (dir)
422159: 1000 23:29 myls (7979)
422142: 1000 23:29 . (dir)
422152: 1000 21:29 empy.txt (0)
422146: 1000 0:11 shared (dir)

$ ./myls
395362: 1000 21:55 subdir1 (dir)
371048: 1000 21:3 .. (dir)
422159: 1000 23:29 myls (7979)
422142: 1000 23:29 . (dir)
422152: 1000 21:29 empy.txt (0)
422146: 1000 0:11 shared (dir)
```

```
$ ./myls subdir1
422151: 1000 21:29 tmp.c (1247)
422142: 1000 21:29 .. (dir)
395362: 1000 21:55 . (dir)
422153: 1000 21:56 subdir2 (dir)
```

In order to display this information you have to use the *opendir* , *closedir*, *readdir*, and *localtime* operations and the *stat* system call from the C Standard library.

Showing some extra information

Modify your program to print the file owner's name instead of its *id* number. For this you have to use the *getpwuid* function.

Based on the previous examples, your program is now supposed to print:

```
$ ./myls /usr
266127: root 17:8 lib (dir)
2: root 12:57 .. (dir)
266128: root 22:57 local (dir)
(...)
$ ./myls subdir1
422151: user 21:29 tmp.c (1247)
422142: user 21:29 .. (dir)
395362: user 21:55 . (dir)
422153: user 21:56 subdir2 (dir)
```

You can also include the full date, group name, and sort by file's name before printing.

Following the sub-directories

Extend your program so that your command also shows the contents of any sub-directory within the directory received as argument. This is equivalent to the "*ls –alR*" command. For instance,

```
$ ls -alR ./subdir1
./subdir1/:
(...)
drwxrwxr-x 3 user user 4096 Nov 12 21:55 .
drwxrwx--- 4 user user 4096 Nov 12 21:29 ..
drwxrwxr-x 2 user user 4096 Nov 12 21:56 subdir2
-rwxrwx--- 1 user user 1247 Nov 12 21:29 tmp.c

./subdir1/subdir2:
(...)
drwxrwxr-x 2 user user 4096 Nov 12 21:56 .
drwxrwxr-x 3 user user 4096 Nov 12 21:55 ..
-rwxrwx--- 1 user user 1247 Nov 12 21:56 tmp2.c

$ ./myls ./subdir1
./subdir1:
422151: user 21:29 tmp.c (1247)
422142: user 1:21 .. (dir)
395362: user 21:55 . (dir)
422153: user 21:56 subdir2 (dir)

./subdir1/subdir2:
395362: user 21:55 .. (dir)
422155: user 21:56 tmp2.c (1247)
422153: user 21:56 . (dir)
```

Bibliography

[1] Sections about persistence (chapters 36 and 39) of the recommended book, "Operating Systems: Three Easy Pieces" Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau"

[2] http://pages.cs.wisc.edu/~remzi/OSTEP/file-intro.pdf