

EE309 Assignment 1

Directory listing - Simplified Is (Is309)

Overview

ls

Lists the content of a folder

```
list all files of current directory | view as list | all files, incl. hidden
dev@dev-machine:~$ ls -lah --human-readable file size
total 15,2M --size of entire directory
directory drwx-----@ 51 dev devs 1632B 22 Sep 16:11 Desktop
           drwx-----@ 7 dev devs 224B 8 Apr 16:02 Documents
           drwx-----@ 56 dev devs 1792B 25 Sep 17:14 Downloads
file      -rw-r--r-- 1 dev devs 980K 19 Sep 16:33 package.json
link      lrwxrwxrwx@ 10 dev devs 690G 20 Sep 16:33 memories.avi
```

Owner: File Owner
Group: Group
Permissions: r=read, w=write, x=execute
Number of Links: User
Size: Byte, K=Kilobyte, M=Megabyte, G=Gigabyte, T=Terabyte
Day-Month-Time: Last modified
Filename

Overview

- ls (list)
 - Display files & directories in a specified location
 - Providing options for display targets / format
 - Hidden files
 - More informations
 - Subdirectories recursion
 - ...

- Goal: Implementing your own simplified ls (ls309)

Getting started

- Download skeleton code into your system (ex. Haedong lounge server)
 - https://teemo.kaist.ac.kr/ee309/2023/assignments/assignment1/ee309_assign1.tar.gz

```
20233083@eelabg1:~$ wget https://teemo.kaist.ac.kr/ee309/2023/assignments/assignment1/ee309_assign1.tar.gz
--2023-09-06 17:41:42-- https://teemo.kaist.ac.kr/ee309/2023/assignments/assignment1/ee309_assign1.tar.gz
Resolving teemo.kaist.ac.kr (teemo.kaist.ac.kr)... 143.248.55.32
Connecting to teemo.kaist.ac.kr (teemo.kaist.ac.kr)|143.248.55.32|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 514039 (502K) [application/octet-stream]
Saving to: 'ee309_assign1.tar.gz'

ee309_assign1.tar.g 100%[=====>] 501.99K  --.-KB/s   in 0.008s

2023-09-06 17:41:42 (61.7 MB/s) - 'ee309_assign1.tar.gz' saved [514039/514039]

20233083@eelabg1:~$ tar -xzvf ee309_assign1.tar.gz > /dev/null
20233083@eelabg1:~$ mv dist/ 20233083_assign1
20233083@eelabg1:~$ ls 20233083_assign1/
Makefile  run_test.py  samplels309  src  tests
```

Getting started

- Compile skeleton code

```
20233083@eelabg1:~$ cd 20233083_assign1/  
20233083@eelabg1:~/20233083_assign1$ make  
mkdir -p obj  
gcc209 -Isrc -Isrc/include -Wall -g -D_GNU_SOURCE -o obj/dnode.o -c src/dnode.c  
mkdir -p obj  
gcc209 -Isrc -Isrc/include -Wall -g -D_GNU_SOURCE -o obj/dir.o -c src/dir.c  
mkdir -p obj  
gcc209 -Isrc -Isrc/include -Wall -g -D_GNU_SOURCE -o obj/main.o -c src/main.c  
gcc209 -o ls309 obj/dnode.o obj/dir.o obj/main.o  
20233083@eelabg1:~/20233083_assign1$ ./ls309  
.
```

Getting started

- Run test script with sample test cases

```
20233083@eelabg1:~/20233083_assign1$ make grade
./run_test.py
[*] Running test test-basic :
  - Running test cases file-single.....O
  - Running test cases file-multiple.....X
  - Running test cases dir-single.....X
  - Running test cases dir-multiple.....X
  - Running test cases dir-cwd.....X
  - Running test cases mixed.....X
[*] Running test test-a :
  - Running test cases dir-hidden-file.....X
  - Running test cases **REDACT**.....? (ModuleNotFoundError)
[*] Running test test-l1 :
  - Running test cases file-single-long.....X
  - Running test cases **REDACT**.....? (ModuleNotFoundError)
  - Running test cases **REDACT**.....? (ModuleNotFoundError)
[*] Running test test-R :
  - Running test cases dir-recursive-one.....X
```

```
***** Grade Result *****
- test-basic (3/20)
  - file-single : Success (3.0/3.0)
  - file-multiple : Fail (0.0/3.0)
  - dir-single : Fail (0.0/3.0)
  - dir-multiple : Fail (0.0/4.0)
  - dir-cwd : Fail (0.0/3.0)
  - mixed : Fail (0.0/4.0)
- test-a (0/10)
  - dir-hidden-file : Fail (0.0/5.0)
  - **REDACT** : Error (0.0/5.0)
- test-l1 (0/15)
  - file-single-long : Fail (0.0/5.0)
  - **REDACT** : Error (0.0/5.0)
  - **REDACT** : Error (0.0/5.0)
- test-R (0/15)
  - dir-recursive-one : Fail (0.0/5.0)
  - **REDACT** : Error (0.0/5.0)
  - **REDACT** : Error (0.0/5.1)
- test-advanced (0/30)
  - **REDACT** : Error (0.0/6.0)
  - **REDACT** : Error (0.0/6.0)
  - **REDACT** : Error (0.0/6.0)
  - **REDACT** : Error (0.0/6.0)
  - **REDACT** : Error (0.0/6.0)
- test-error (0/10)
  - error-eexist : Fail (0.0/3.3)
  - **REDACT** : Error (0.0/3.3)
  - **REDACT** : Error (0.0/3.4)
-----
- total 3/100
```

Specifications

- Inputs
 - Take the (absolute/relative) paths to the file or directory from command-line arguments
 - `$./ls309 /home/user/some_abs_path`
 - `$./ls309 ./some_rel_path`
 - `$./ls309 ./file1 ./file2 ./file2`
 - `$./ls309 ./some_file1 ./some_dir/ ./some_file2`
 - Support an empty argument -> use the current directory
 - `$./ls309` should give same result with `$./ls309 .`
 - Support three options: -a, -l, and -R
 - `$./ls309 -a`
 - `$./ls309 -a -l`
 - `$./ls309 -aR`
 - `$./ls309 -aIR`
 - the option strings can be located in any position
 - `$./ls309 -aIR ./file`
 - `$./ls309 ./file -aIR`
 - `$./ls309 ./file1 -a ./file2 -IR`

Specifications

- Outputs
 - Print a list separated by newline("\n") into standard output
 - Should be sorted in **lexicographical order based on ASCII character set**
 - Different behavior from coreutils ls

```
$ ls
alpha file samplels309 Zoo
$ ./samplels309
Zoo
alpha
file
samplels309
```


Specifications

- Outputs
 - If multiple display targets are given
 - List the files from targets
 - List the directories with their name and entries.
 - Lists should be separated by newline("\n")
 - Each list and order of directories should be sorted as well.

```
$ ./samples309 file1 dir1 dir2 file2 dir3
file1
file2

dir1:
dir1-file
dir1-file2

dir2:
dir2-file

dir3:
dir3-file
dir3-file2
dir3-file3
```

Specifications

- Options
 - **-a** option: Display hidden files and directories (start with .)

```
$ ./samplels309
file
samplels309
$ ./samplels309 -a
.
..
.hidden-file
file
samplels309
```

Specifications

- Options
 - `-l` option: Display additional information

```
$ ./samplels309
dir
file
samplels309
symlink
$ ./samplels309 -l
total 1184
drwxrwxr-x   2 20233083 20233083      4096 Sep  7 14:52 dir
-rw-rw-r--   1 20233083 20233083         0 Sep  7 14:52 file
-rwxrwxr-x   1 20233083 20233083 1204960 Sep  7 14:51 samplels309
lrwxrwxrwx   1 20233083 20233083         7 Sep  7 14:53 symlink -> /bin/sh
```

Specifications

- Options
 - **-R** option: Display directories recursively

```
$ ./samplels309
dir
file
samplels309
$ ./samplels309 -R
.:
dir
file
samplels309

./dir:
file1
file2
```

Specifications

- Etc
 - Properly handle errors and print textual error messages
 - ex. **No such file or directory**
 - Should be robust from any kind of input
 - i.e. Should not crash unintendedly
 - Will get 0 points for test case in which unexpected crashes happen
 - Should not have any kind of memory leak.
 - i.e. Should free every memory which returned from malloc

Implementation

- Option parsing & Entries sorting
 - Finish implementation of option parsing routine of **main** function
 - Finish implementation of **sort_dnode_entries** function
 - You may want to use **compare_dnode_name** helper function

```
73 v int main(int argc, char *argv[]) {
74     int opt;
75
76     // parse given options and set proper flags
77 v while ((opt = getopt(argc, argv, "alR")) != -1) {
78 v     switch (opt) {
79 v         // TODO: Task 1
80         //
81         //
82         //
83         default: /* unknown options */
84             fprintf(stderr, "Usage: %s [-alR] [file path]\n", argv[0]);
85             exit(EXIT_FAILURE);
86     }
87 }
```

```
// sort_dnode_entries : sort given dnode linked list into dnode array
// @param head : head pointer of dnode linked list
// @param cnt : count of dnodes
// @return : dnode double pointer that points sorted dnode array
struct dnode **sort_dnode_entries(struct dnode *head, size_t cnt) {
    struct dnode **res, **curr;

    if (!(res = calloc(sizeof(struct dnode *), cnt))) {
        fprintf(stderr, "cannot allocate dnode array : %s\n", strerror(errno));
        return NULL; /* could not allocate dnode array */
    }

    curr = res;
    while (head) {
        *curr++ = head;
        head = head->dn_next;
    }

    // TODO: Task 1
    //
    //
    //

    return res;
}
```

Implementation

- Directory listing and Support **-a** option
 - Implement **parse_dir** function
 - You may need to use...
 - Directory-related library functions (**opendir**, **readdir**, **closedir**)
 - Other helper functions (**concat_path**, **strdup**)
 - Support **-a** option into your implementation of **parse_dir** function.

```
// parse_dir : parse the directory into dnode linked list
// @param path : path of directory that will be parsed
// @param cnt : size_t pointer which count of dnode entries stored
// @return : head pointer of dnode linked list
struct dnode *parse_dir(const char *path, size_t *cnt) {
    struct dnode *head = NULL;

    // TODO: Task 2
    //
    //
    //

    return head;
}
```

Implementation

- Support `-l` option
 - Finish implementation of `parse_dnode` function.
 - Need to follow or not follow the link based on `follow_link` parameter
 - You may need to use library functions for file information(`stat`, `fstat`, `lstat`)

```
105 ✓ // parse_dnode : parse the info about given file into dnode
106 // @param fullname : (full) file path for parsing
107 // @param follow_link : determine wheter follows symlink or not
108 // @return : dnode pointer that stores info about given file
109 ✓ struct dnode *parse_dnode(const char *fullname, int follow_link) {
110     struct dnode *res;
111
112 ✓     if (!(res = malloc(sizeof(struct dnode)))) {
113         fprintf(stderr, "cannot allocate dnode : %s\n", strerror(errno));
114         return NULL; /* could not allocate dnode */
115     }
116
117 ✓     // TODO: Task 3
118     //
119     //
120     //
```

```
122     res->fullname = strdup(fullname);
123     res->dn_next = NULL;
124
125     // TODO: Task 3
126     //
127     //
128     //
129
130     return res;
131 }
```


Implementation

- Support -l option
 - Finish implementation of `display_dnode_long` function
 - You may want to use `readlink` function to get link target.

```
84 printf("%s", d->name);
85 if (S_ISLNK(d->dn_mode)) {
86     if (!(lpath = malloc(MAX_PATH_SIZE + 1))) {
87         fprintf(stderr, "cannot allocate lpath : %s\n", strerror(errno));
88         return; /* could not allocate lpath */
89     }
90
91     // TODO: Task 3
92     //
93     //
94     //
95
96     lpath[ret] = '\0';
97
98     printf(" -> ");
99     printf("%s", lpath);
100    free(lpath);
101 }
```

Implementation

- Support -R option
 - Finish implementation of `print_dir` function

```
// print_dir : print the entires of given directory with proper flags
// @param path : path of directory that will be printed
void print_dir(const char *path) {
    // TODO for assign
    size_t dnode_cnt;
    struct dnode *dnode_head, **dnode_arr;

    dnode_head = parse_dir(path, &dnode_cnt);
    dnode_arr = sort_dnode_entries(dnode_head, dnode_cnt);

    if (is_print_dir)
        printf("%s:\n", path);
    display_dnode_arr(dnode_arr, dnode_cnt);

    // TODO: Task 4
    //
    //
    //

    free(dnode_arr);
}
```

Implementation

- Error handling & Robustness
 - For error handling...
 - Always check the return values of library functions
 - Print the proper error message into standard error
 - You may want to use the **errno** variable and **perror** / **strerror** functions
 - For the robustness...
 - Always be aware of memory-related bugs
 - You may use [Valgrind](#) or [Address Sanitizer](#) to check

Submission

- Submit your code as tar.gz archive file
 - Use **KAIST KLMS** to submit your project.
 - File name: **<YourStudentID>_assign1.tar.gz**
 - Make sure to run **make clean** before submitting.

```
20233083@eelabg1:~/20233083_assign1$ make clean
rm -rf ls309 obj ./ee309_test
20233083@eelabg1:~/20233083_assign1$ cd ../
20233083@eelabg1:~$ tar -czvf 20233083_assign1.tar.gz ./20233083_assign1/ > /dev/null
20233083@eelabg1:~$ ls ./20233083_assign1.tar.gz
./20233083_assign1.tar.gz
```

Criteria

- Will test codes on ubuntu 20.04.6 LTS (same with Haedong lounge server)
 - Only provide subset of all test cases as sample

Test categories	Weight
test-basic	20%
test-a	10%
test-l	15%
test-R	15%
test-advanced	30%
test-error	10%

Notes

- Linux man page always helpful
 - ex. [man getopt](#)
- Study [general guideline](#) and [course policy](#) carefully
 - Ethics document, Collaboration Policy...
- Feel free to ask questions on [Piazza](#)

Thank you

Any questions?