

Important: Please do all assignments on hoare

Linux System Calls and Library Functions

The goal of this homework is to become familiar with the environment in hoare while practicing system calls. We will be using `getopt` and `perror`.

Do Exercise 5.8: **Traversing Directories** (p. 179) in your text by Robbins/Robbins. You only need to do the Example 5.37, or depth-first traversal.

The programming task requires you to create a utility to traverse a specified directory in depth-first order. Depth-first search explores each branch of a tree to its leaves before looking at other branches. Depth-first search is naturally recursive, as indicated by following pseudocode:

```
depthfirst ( root )
{
    for each node at or below root
    {
        visit node;
        if node is a directory
            depthfirst ( node );
    }
}
```

The indentation of the filenames shows the level in the file system tree. Use the output format specified in Example 5.37, with a default indentation of 4 spaces for each level in the directory. You should be able to change the indentation spaces by using an option on command line followed by a number. The executable should be called `dt`. The program will be invoked by:

```
dt [-h] [-I n] [-L -d -g -i -p -s -t -u | -l] [dirname]
```

The options are to be interpreted as follows:

h Print a help message and exit.

I n Change indentation to *n* spaces for each level.

L Follow symbolic links, if any. Default will be to not follow symbolic links.

t Print information on file type.

p Print permission bits as `rw-rw-rw`.

i Print the number of links to file in inode table.

u Print the UID associated with the file.

g Print the GID associated with the file.

s Print the size of file in bytes. If the size is larger than 1K, indicate the size in KB with a suffix K; if the size is larger than 1M, indicate the size in MB with a suffix M; if the size is larger than 1G, indicate the size in GB with a suffix G.

d Show the time of last modification.

1 This option will be used to print information on the file as if the options `tpiugs` are all specified.

If the user does not specify `dirname`, run the command using current directory and print the tree accordingly. The output will appear as follows:

```
$ ls -l 4 -l proj
proj
  bi_scan      drwx----- 10 sanjiv  faculty  4K Nov 25, 2003
  CVS          drwx-----  3 sanjiv  faculty  4K Jul 06, 2004
    Entries    drwx-----  2 sanjiv  faculty  4K Nov 25, 2003
    Repository -rw-----  1 sanjiv  faculty 336 Nov 25, 2003
    Root       -rw-----  1 sanjiv  faculty  24 Nov 25, 2003
  Makefile     -rw-----  1 sanjiv  faculty  15 Nov 25, 2003
  Makefile.Linux -rw-----  1 sanjiv  faculty 712 Nov 25, 2003
  include      -rw-----  1 sanjiv  faculty  1K Nov 25, 2003
    CVS        drwx-----  3 sanjiv  faculty  4K Nov 25, 2003
      Entries  drwx-----  2 sanjiv  faculty  4K Nov 25, 2003
      Repository -rw-----  1 sanjiv  faculty 650 Nov 25, 2003
      Root     -rw-----  1 sanjiv  faculty  24 Nov 25, 2003
    cluster.h  -rw-----  1 sanjiv  faculty  15 Nov 25, 2003
    config.h   -rw-----  1 sanjiv  faculty  5K Nov 25, 2003
```

With the use of `perror`, I'll like some meaningful error messages. The format for error messages should be:

dt: Error: Detailed error message

where `dt` is actually the name of the executable (`argv[0]`) and should be appropriately modified if the name of executable is changed without recompilation. These error messages should be sent to `stderr` using `perror`.

It is required for this project that you use version control, a `Makefile`, and a `README`. Your `README` file should consist at a minimum of a description of how I should compile and run your project, any outstanding problems that it still has, and any problems you encountered. Your `Makefile` should use suffix-rules or pattern-rules and have an option to clean up object files.

What to handin

Create your programs in a directory called `username.1` where `username` is your user name on hoare. Once you are done with developing and debugging, *remove the executables and object files*, and issue the following commands:

```
% cd
% chmod 755 ~
% ~sanjiv/bin/handin cs4760 1
% chmod 700 ~
```

Do not copy and paste those commands from the PDF of the assignment. Type in the commands.

Do not forget `Makefile` (with suffix or pattern rules), your versioning files, and `README` for the assignment. If you do not use version control, you will lose 10 points. I want to see the log of how the program files are modified. Therefore, you should use some logging mechanism and let me know about it in your `README`. You must check in the files at least once a day while you are working on them. Omission of a `Makefile` (with suffix rules) will result in a loss of another 10 points, while `README` will cost you 5 points. I do not like to see any extensions on `Makefile` and `README` files.

Before the final submission, perform a `make clean` and keep the latest source checked out in your directory.

You do not have to hand in a hard copy of the project. Assignment is due by 11:59pm on the due date.