

CS 306 – Linux/UNIX Programming – Spring 2012

Solutions to Homework #1

©2012 Norman Carver, Computer Science Dept., Southern Illinois University Carbondale. This material is provided for *personal use by students enrolled in CS 306 Spring 2012 only!* Any other use represents infringement of the author's exclusive rights under copyright law.

1. Username, default group, UID, and GID:

Obviously depends on user account, but for cs majors should have: username based on actual name, group will be `ugrad`, UID will vary, and GID will be 110. For non-majors, username will be something like `siucs011`, group will be `siucs`, UID will vary, and GID will be 150.

The answers to this question and the next are to be from your CS Dept. Linux account—not a personal Linux machine!

2. Path of your *home* directory:

Obviously depends on account username, but will be under `/home`, something like `/home/carver` (where `carver` is your username).

3. Four commands for resetting the *current working directory* to home directory:

- (1) `cd` (which is equivalent to doing `cd $HOME`)
- (2) `cd ~`
- (3) `cd ~carver` (replaced with your username)
- (4) `cd /home/carver` (replaced with your username)

4. Path of the `ls` command/program:

To find use: `which ls`
Path: `/bin/ls`

5. Two commands to change directories:

- (1) Absolute path: `cd ~/courses/cs306`
or could use `cd /home/carver/courses/cs306`
- (2) Relative path: `cd ../../../../courses/cs306`

6. Listing `~/ .bashrc`:

The file is not listed with a normal `ls` because it is considered a “hidden file” since it starts with a “.” and `ls` does not display hidden files by default. Must use either `-a` or `-A` option to display hidden files with `ls` (could also do `ls .*` but then this won't list non-hidden files).

7. `ls` options to get a long listing of a directory itself:

The `l` and `d` options would be required:
`ls -dl dir`

8. How to move C source files:

```
mv ~/cs306/*.c ~/cs306/labs
```

or

```
cd ~/cs306; mv *.c labs
```

(GCC assumes that C source files have a “c” extension, so these are “C source files”)

9. How to get prompted by `cp`:

You need to use the `-i` or `--interactive` option, so command: `cp -i test ~`

10. How to always be prompted when copying files:

You need to create an **alias** for **cp** that includes the option mentioned in the previous problem. To get this alias to always automatically be enabled, you need to place the **alias** command into an initialization file for bash, usually `~/.bashrc` is best (on CS Dept. accounts, students must use `~/mybash` as your `.bashrc` is locked).

There are two possible approaches to creating an alias:

- (1) Redefine `cp`: `alias cp="cp -i"`
- (2) Define alternative command: `alias cpi="cp -i"`

11. Default search path:

Two main ways to determine: (1) `printenv PATH` or (2) `echo $PATH`

Search path (on CS machines) may be something like:

```
/usr/kerberos/bin:/usr/X11R6/bin:/usr/local/sbin:/usr/local/bin...
```

12. CWD in search path:

You need to look for “.” (i.e., *dot*, which denotes the CWD) as one of the directories specified in the command search path stored in the *environment variable* `PATH` (using one of the commands given for the previous problem to look at the search path). If `PATH` contains the dot, it is best if it come at the very end so that it is the last directory to be searched. Some Linuxes (and some versions of Ubuntu) have it while others do not. A system is slightly more secure without it, as standard commands cannot be overridden by mistake as easily. If it is not in `PATH`, you may have to run a personal command as: `./command` rather than just `command`.

Note that this question is asking whether the CWD is *always* in the command search path (`PATH`), no matter what the CWD is. Thus, the correct answer is not a matter of looking for a *particular* CWD in `PATH`—you must find “.”!

13. Command to print the *number* of files in current directory:

```
ls -A | wc -l
```

(Need `A` option to list hidden but not `.` and `..`, and need `l` option with `wc` to just count lines—if use `w` option to count words, will be off if filenames contain spaces.)

14. Command to print out `printf` calls in file:

```
grep printf ~/cs03/lab1/lab1.c
```

or

```
grep 'printf(' ~/cs03/lab1/lab1.c
```

(including open paren helps ensure that only function calls are picked out, but requires quoting since paren is Bash metachar)

15. Command to create file listing files modified in July:

This is a deceptively hard problem to solve perfectly. The `ls` long listing option (`-l`) will cause the *modification time* (`mtime`) to be printed (along with other info), which you can pipe through `grep` to select files modified in July, redirecting output to a file:

```
ls -l | grep Jul > ~/jul-files
```

This is all we were expecting, but it is by no means a perfect solution. First, it puts the entire “long” output line for each file into the output—not just the filenames. Second, it will also put files with names like “Jules” into the list, whether they were modified in July or not.

It turns out that reliably getting exactly what was asked for is somewhat “tricky.” Ideally, you would like to think of each piece of information in the long output listing

as being in a particular whitespace-separated column, then choose lines where column 6 is “Jul”, then select columns 9 and on for output (filenames containing whitespace will appear as multiple columns). This can be done with a utility like *AWK*:

```
ls -l | awk '$6 ~ /Jul/ {for (i=9;i<=NF;i++) print $i}'
```

While it seems as though you should be able to do something similar with *cut* (which is designed to pick out columns), *cut* assumes a *single separator character* but *ls -l* uses variable numbers of spaces due to different sized fields. Before we can design a solution without using *AWK*, we must discuss *ls*' time formats:

ls can give mtime's in several formats, and different Linuxes may have different defaults. The answers above have been given assuming the mtime format will be like “Jul 12 16:26” or “Jul 12 2011”. Some distros may use “long-iso” format, though, which looks like: “2011-07-12 16:26”.

This format requires filtering on “-07-”, but searching for dashes causes problems with *grep* since dashes introduce options, requiring that the first dash be “doubly quoted” (single quote plus escape) like so:

```
ls -l | grep '\-07-' > ~/jul-files
```

The advantage of the long-iso format is that using it will guarantee that we first encounter a sequence like dash-digit-digit-dash for the month. A clever approach suggested by former CS 306 student Tim Hanson makes use of this fact along with *regular expressions* and the *-o* option in *grep* to solve the problem perfectly (the *-o* option causes just the matched portion of a line to be output). Here is his solution:

```
ls -l --time-style=long-iso | grep -o '\-[0-9][0-9]-.*' |
  grep '^-07' | cut -d' ' -f3-
```

Regular expressions are a language for expressing patterns using a set of *metacharacters* to express quite complex patterns. While regular expressions appear similar to shell filename expansion (globbing) expressions, they are not the same! The regular expression *-[0-9][0-9]-.** matches the month and all and the remaining portion of each line, while the regular expression *^-07* would match July months but not *-07* in filenames. Finally *cut* is applied to leave only the filenames.

16. Files deleted by command:

```
dab-2.data, eabc-2.data, eefg-3.data, iabh7-4.data, icd2-5-3.data,
jcd2ef-3.data, pabcdf3-4.data, pefcd-3.data.
```

17. Moving files:

```
mv ~/tmp/306*info*.{txt,text} ~/save
```

Technically this will move files with a final extension of *txt/text*, so would move a file like *306info.orig.text* that maybe should not be moved given the naming pattern spec, but it would be hard to limit the pattern to a single extension.

18. *ls* related questions:

- Owner is *carver* and the first *rw-* are the permissions: read and write, but not execute.
- The file's group is *faculty*, with permissions *r-x*, which means read and execute, but not write permissions.
- The owner *carver* and other than members of the group *faculty*.
- The file is 1732 *bytes* in size (must be clear on units—cannot just say 1732).
- Last time the file was *modified*: Jan 23 during the last year, at 8:39pm.

19. Setting all permissions:
Numeric chmod: `chmod 640 test`
Symbolic chmod: `chmod u=rw,g=r,o= test`
(Note: must have `o=` or any current setting won't be changed)
20. Adding write for everyone:
`chmod a+w test` or even just `chmod +w test`
21. Two different versions of the `ps` command:
The standard options for this are:
`ps aux`
`ps -ef`
`ps -eF`
22. Command to terminate process whose *pid* is 4513:
To definitely terminate use:
`kill -9 4513`
or equivalently:
`kill -SIGKILL 4513`