Basic Linux (Bash) Commands

- secure shell login: `ssh user@hostname` logs `user` into `hostname`. e.g.

  `ssh jgillett@public03.stat.wisc.edu`

- web get: `wget URL` downloads the file at `URL`. e.g.

  `wget https://pages.stat.wisc.edu/~jgillett/DSCP/linux/junk.tar`

- `tar`: ("tape archive") write a directory of files to a `.tar` file. e.g. `tar -cvf DIR.tar DIR` creates `DIR.tar` from `DIR`, and `tar -xvf DIR.tar` extracts `DIR` from `DIR.tar`. e.g.

  `tar -xvf junk.tar`

- directories:
  - `mkdir DIRECTORY`: make `DIRECTORY`, e.g. `mkdir ~/linux`
  - `cd [dir]`: change directory to (optional) `dir`, which defaults to your home directory. Shorthand includes: `~` ("tilde"): your home directory, "`.`" ("dot"): current directory, "`..`" ("dot dot"): parent directory, `~user` ("tilde user"): `user`'s home directory.
  - `pwd`: print working directory.
  - `rmdir DIRECTORY`: remove empty directory `DIRECTORY`
  - `ls`: list directory (see `ls -ltr` below)

- `man name`: display manual page for `name`. e.g. Run `ls -ltr`. Then run `man ls` to learn what the `-l`, `-t`, and `-r` options of `ls` do. Hint: Run `man` in emacs via `M-x man Enter ls Enter` to get emacs page navigation and search features within the manual page.

- files
  - `cp SOURCE DEST` or `cp SOURCE DIRECTORY`: copy; or, for copying between computers, use `scp` ("secure copy"): `scp [[user@]host:]file1 [[user@]host2:]file2]`
  - `mv SOURCE DEST` or `mv SOURCE DIRECTORY`: move (or rename)
  - `cat FILE(S)`: concatenate file(s) and print on standard output. e.g. `cat FILE_1 FILE_2`
  - `rm FILE`: remove
  - `chmod MODE FILE`: change file mode (NFS permission) bits. e.g. We will need `chmod u+x hello.sh` ("give `u`=user `x`=execute permission on `hello.sh`)," later.

- `grep PATTERN [FILE]`: ("global regular expression print") print lines matching `PATTERN`

- `diff FILES` shows differences between two `FILES` or directories; e.g.

  (in the `junk/fruits` directory) `diff fruits.csv fruits.csv~`

- `head` prints the first part of a file and `tail` prints the last part; e.g.

  ```
  head -n 3 mtcars.csv
  tail -n 3 mtcars.csv
  tail -n +2 mtcars.csv # "+2" says "start on line 2" (to omit header)
  ```

- `wc [FILE]` prints newline, word, and character counts for `FILE`; e.g. `wc fruits.R`
- A *pipeline* of the form `A | B` makes the output of `A` the input of `B`. e.g. `ls -l | wc`
- `sort [FILE]` sorts lines of `FILE`; options include `-t 'S'` for field separator `S` instead of blank, `-n` for "numeric," `-r` for "reverse," `-k F` to use field (column) number `F` as the key; e.g.

  ```
  tail -n +2 fruits.csv | sort -t ',' -n -k 2 -r # try without -n
  ```

- `uniq FILE` omits repeated lines from `FILE`; options `-c` includes counts; e.g.

  (in the `junk/a_words` directory) `cat a3.txt | sort | uniq -c`
- `find [path] [expression]`: find files in directory hierarchy. e.g. `find ~ -name "*.R"`

  The option `-exec COMMAND {} ";"` runs `COMMAND` (terminated by `";"`) on each pathname (represented by `{}`). e.g. `find ~ -name "*.R" -exec grep "rm(list" {} ";" -print` finds each file whose name ends ".R" and runs `grep "rm(list"` on each file (`{}`); the `";"` ends the input to `grep`; `-print` prints the names of the matching files
- `sed`: stream editor; e.g. search and replace in each line: `sed 's/PATTERN/REPLACEMENT/' [FILE]`; e.g.

  (in the `junk/a_words` directory) `sed 's/awestruck/lovestruck/' a3.txt`
- `cut` removes sections from each line; e.g.

  ```
  cat mtcars.csv | cut -d ',' -f 2 # use delimiter ',' and cut field 2
  ```

- `awk`: extract and summarize data (complex; breaks line into fields). Option `-F` indicates field separator. A simple use is to select rows based on a column value. e.g.

  ```
  cat mtcars.csv | awk -F ',' '{ if ($10 == "0") {print $0} }' # select "am == 0" rows
  ```

- Others: `echo`, `exit`, `hostname`, `kill`, `ps`, `time`, `top`

Command-line editing: `C-p` previous command, `C-n` next command; `C-r` reverse search and `C-s` search through command history; cursor motion (like emacs): `C-f` forward, `C-b` back, `C-a` start of line, `C-e` end of line, `C-d` delete character

Hint: Running commands in the emacs shell (`emacs -nw`, then `M-x shell Enter`) instead of the terminal eases searching for and revising commands and navigating and copying/pasting output.

Exercises

1. Use `cd` and `ls` several times each to find the number of files are in the `junk` folder.
2. How many words are in `README.txt`?
3. Use `cd`, `ls`, and `rm` several times each to remove each (backup) file ending with `~`.
4. Excluding its header, sort the lines of `mtcars.csv` by its `cyl` field (the third field).
5. Use a pipeline with `cat`, `awk`, `cut`, `sort`, and `head` to read from `mtcars.csv`, select the 3-speed rows (gear is 3), retain only the weight (wt) column, and print the weight of the heaviest 3-speed car. Hint: Write and test pipeline by adding one command at a time.