Linux/Unix Command Line Cheat Sheet - GettingGeneticsDone.blogspot.com

Command	Description
`pwd`	prints working directory (prints to screen, ie displays the full path, or your location on the filesystem)
`ls`	lists contents of current directory
`ls –l`	lists contents of current directory with extra details
`ls` ***`/home/user/*.txt`***	lists all files in /home/user ending in .txt
`cd`	change directory to your home directory
`cd ~`	change directory to your home directory
`cd` ***`/scratch/user`***	change directory to user on scratch
`cd -`	change directory to the last directory you were in before changing to wherever you are now
`mkdir` ***`mydir`***	makes a directory called mydir
`rmdir` ***`mydir`***	removes directory called mydir. mydir must be empty
`touch` ***`myfile`***	creates a file called myfile. updates the timestamp on the file if it already exists, without modifying its contents
`cp` ***`myfile myfile2`***	copies myfile to myfile2. if myfile2 exists, this will overwrite it!
`rm` ***`myfile`***	removes file called myfile
`rm –f` ***`myfile`***	removes myfile without asking you for confirmation. useful if using wildcards to remove files *******
`cp –r` ***`dir newdir`***	copies the whole directory dir to newdir. –r must be specified to copy directory contents recursively
`rm –rf` ***`mydir`***	this will delete directory mydir along with all its content without asking you for confirmation! *******
`nano`	opens a text editor. see ribbon at bottom for help. ^x means CTRL-x. this will exit nano
`nano` ***`new.txt`***	opens nano editing a file called new.txt
`cat` ***`new.txt`***	displays the contents of new.txt
`more` ***`new.txt`***	displays the contents of new.txt screen by screen. spacebar to pagedown, q to quit
`head` ***`new.txt`***	displays first 10 lines of new.txt
`tail` ***`new.txt`***	displays last 10 lines of new.txt
`tail –f` ***`new.txt`***	displays the contents of a file as it grows, starting with the last 10 lines. ctrl-c to quit.
`mv` ***`myfile newlocdir`***	moves myfile into the destination directory newlocdir
`mv` ***`myfile newname`***	renames file to newname. if a file called newname exists, this will overwrite it!
`mv` ***`dir subdir`***	moves the directory called dir to the directory called subdir
`mv` ***`dir newdirname`***	renames directory dir to newdirname
`top`	displays all the processes running on the machine, and shows available resources
`du –h --max-depth=1`	run this in your home directory to see how much space you are using. don't exceed 5GB
`ssh` ***`servername`***	goes to a different server. this could be queso, brie, or provolone
`grep` ***`pattern files`***	searches for the pattern in files, and displays lines in those files matching the pattern
`date`	shows the current date and time
`anycommand` **`>`** ***`myfile`***	redirects the output of anycommand writing it to a file called myfile
`date >` ***`timestamp`***	redirects the output of the date command to a file in the current directory called timestamp
`anycommand` **`>>`** ***`myfile`***	appends the output of anycommand to a file called myfile
`date >>` ***`timestamp`***	appends the current time and date to a file called timestamp. creates the file if it doesn't exist
`command1` **`|`** ***`command2`***	"pipes" the output of command1 to command2. the pipe is usually shift-backslash key
`date | grep` ***`Tue`***	displays any line in the output of the date command that matches the pattern Tue. (is it Tuesday?)
`tar -zxf` ***`archive.tgz`***	this will extract the contents of the archive called archive.tgz. kind of like unzipping a zipfile. *******
`tar -zcf` ***`dir.tgz dir`***	this creates a compressed archive called dir.tgz that contains all the files and directory structure of dir
`time` ***`anycommand`***	runs anycommand, timing how long it takes, and displays that time to the screen after completing anycommand
`man` ***`anycommand`***	gives you help on anycommand
`cal -y`	free calendar, courtesy unix
`CTRL-c`	kills whatever process you're currently doing
`CTRL-insert`	copies selected text to the windows clipboard (n.b. see above, ctrl-c will kill whatever you're doing)
`SHIFT-insert`	pastes clipboard contents to terminal

******* = use with extreme caution! you can easily delete or overwrite important files with these.

Absolute vs relative paths.

Let's say you are here: /home/turnersd/scripts/. If you wanted to go to /home/turnersd/, you could type: **`cd /home/turnersd/`**. Or you could use a relative path. **`cd ..`** (two periods) will take you one directory "up" to the parent directory of the current directory.

`.`	(a single period) means the current directory
`..`	(two periods) means the parent directory
`~`	means your home directory

A few examples

`mv myfile ..`	moves myfile to the parent directory
`cp myfile ../newname`	copies myfile to the parent directory and names the copy newname
`cp /home/turnersd/scripts/bstrap.pl .`	copies bstrap.pl to "." i.e. to dot, or the current directory you're in
`cp myfile ~/subdir/newname`	copies myfile to subdir in your home, naming the copy newname
`more ../../../myfile`	displays screen by screen the content of myfile, which exists 3 directories "up"

Wildcards (use carefully, especially with rm)

`*`	matches any character. example: **`ls *.pl`** lists any file ending with ".pl" ; **`rm dataset*`** will remove all files beginning with "dataset"
`[xyz]`	matches any character in the brackets (x, y, or z). example: **`cat do[or]m.txt`** will display the contents of either doom.txt or dorm.txt