Bash Shell Reference (manual)

category	command	description
basic shell	`clear`	clear all previous commands' output text from the terminal
	`exit` (or `logout`)	quits the shell
	`alias`, `unalias`	give a pseudonym to another command (you may need to enclose the command in quotes if it contains spaces or operators)
	`history`	show a list of all past commands you have typed into this shell
directories	`ls`	list files in a directory
	`pwd`	displays the shell's current working directory
	`cd`	changes the shell's working directory to the given directory; can be a relative or absolute path
	`mkdir`	creates a new directory with the given name
	`rmdir`	removes the directory with the given name (the directory must be empty)
file operations	`cp`	copies a file/directory
	`mv`	moves (or renames) a file/directory
	`rm`	deletes a file
	`touch`	update the last-modified time of a file (or create an empty file)
file examination	`cat`	output the contents of a file
	`more` (or `less`)	output the contents of a file, one page at a time
	`head`, `tail`	output the beginning or ending of a file
	`wc`	output a count of the number of characters, lines, words, etc. in a file
	`du`	report disk space used by a file/directory
	`diff`	output differences between two files
file permissions	`chmod`	change the permissions on a file or group of files
	`chown`	change the owner of a file
	`chgrp`	change the group associated with a file
	`umask`	change the default permissions given to newly created files
searching and sorting	`grep`	search a file for a given string or expression
	`sort`	convert an input into a sorted output
	`uniq`	strip duplicate lines
	`find`	search for files by name within a given directory
	`xargs`	launch a command over each of a set of lines of input (often used with `find`)
	`locate`	search for files by name on the entire system
	`which`	shows the complete path of a command or file
compression	`zip`, `unzip`	create a .zip archive or extract its contents
	`tar`	Unix archiving/de-archiving program
	`gzip`, `gunzip`	GNU compression/decompression programs
	`bzip2`, `bunzip2`	improved compression/decompression programs
system information	`date`	outputs the current date/time
	`cal`	outputs an ASCII calendar
	`uname`	print information about the system
	`time`	measure how long a program takes to run
process	`ps`, `jobs`	list the processes you are running; every process has a unique

management		integer id number (PID)
	`top`	see what processes are using the most CPU/memory, and show system memory/CPU stats
	`kill`	terminate a process
	`killall`	terminate a group of processes by name
	`^C` or `^\`	(hotkey) terminates (kills) the currently running process
	`^Z`	(hotkey) suspends the currently running process
	`&`	(special character) when `&` is placed at the end of a command, that command is run in the background (shell does not wait for the command to finish before returning to the input prompt)
	`bg`, `fg`	starts a suspended process running in the background or foreground
users and groups	`whoami`	outputs your user name
	`passwd`	changes your password
	`groups`	list the groups to which a user belongs
	`sudo`	execute a single command as the super-user
	`su`	log in to a shell as the super-user
multi-user environments	`hostname`	outputs the name of the current computer/server
	`w`, `finger`	see who is logged in to this computer
	`write`	sends a message to another user logged in to this computer
	`wall`	broadcasts a message to all other users logged in to this computer
	`.plan`	(filename) a special hidden file you can create in your home directory, whose contents will be displayed when other users run `finger` on you. Was originally intended to be used to tell others what you are up to right now. (the Twitter of the 1970s!)
network	`links`, `lynx`	text-only web browsers (yes, really)
	`ssh`, `sftp`, `scp`	connect to a remote Unix server; open a shell on it or send/receive files from it
	`wget`	download from a URL and save it to a file on the local hard drive
	`curl`	download from a URL and output its contents to the console
	`pine`, `mail`	text-only email programs
text editors	`pico`, `nano`	crappy but simple text editors (recommended)
	`emacs`	a complicated text editor (not recommended)
	`vi`, `vim`	another complicated text editor (not recommended)
regular expressions	`sed`	stream editor; find/replace based on regular expressions
	`egrep`	extended version of `grep` that matches regular expressions
programming	`javac`, `java`	compile or run a Java program
	`python`, `perl`, `ruby`, `gcc`, `sml`, ...	compile or run programs in various other languages
shell scripting	`echo`, `printf`	like `println` for the shell; outputs a message or value
	`read`	reads a value from standard input
	`set`, `unset`	give values to a variable, or delete a variable
	`export`	sets a variable that any sub-programs launched by this shell can see
	`let`	for computing integer variable values
	`source`	executes commands/statements stored in another file (useful for re-loading `.bash_profile` without logging out)
	`if`, `[`, `for`,	bash control statements

	`while`	
	`seq`	outputs a sequence of integers (used with `for` loops)
miscellaneous	`yes`	output `"y"` (or another string) over and over
	`sleep`, `usleep`	pause for a given number of seconds or ms
	`~stepp/cowsay`	displays a talking ASCII cow (on `attu` only, though you could install it if you are using Linux on a PC)
	`xeyes`	googley eyes that follow your mouse cursor
build management	`make`	determine which parts of a system must be recompiled, and compile them
	`svn`, `cvs`	Subversion and CVS version-control systems

- References of common commands: UW (alphabetical), New Mexico Tech (by functionality), ss64 (alphabetical), Tennessee-Knoxville (by functionality)
- notes about command-line arguments:
  - Most options are a `-` followed by a letter such as `-c` . Some options are longer words/phrases preceded by two `-` signs, such as `--count` .
  - Many parameters can be combined; for example, `ls -l -a -r` can be written as `ls -lar` .
  - Many programs accept a `--help` or `-help` parameter to give more information about that command (in addition to `man` or `info` pages).
  - For many programs that require parameters, if you run the program and omit those parameters, it will print information about how to use the program. Therefore one good way to learn about commands is to run them with no arguments.
  - For many commands that accept a file name parameter, if you omit the parameter, it will read from standard input (your keyboard). Note that this can conflict with the previous tip.

Keyboard shortcuts and special characters: (`^X` means hold `Ctrl` key and press `X`)

key / character	description
`Up arrow`	repeat previous command(s)
`Home`/`End` or `^A`/`^E`	move cursor to start/end of line
`*`	"wildcard", matches any file(s)
`Tab`	auto-completes a partially typed file/directory/command name
`^C` or `^\`	kills the currently running process
`^D`	end-of-input; press this if a program is reading input from your keyboard and you want to notify it that you are finished
`^Z`	suspends (pauses) the currently running process; use `fg` or `bg` to resume it
`^S`	never ever press this; worst hotkey ever; totally locks up your shell until you press `^Q`

attu

- connect to our shared student Linux server by typing: `ssh attu.cs.washington.edu`
- there are actually severall `attus` (to spread the load), so if you want to be on the same physical machine as your friend, you may need to connect to `attu2`, `attu3`, etc.