cd	change directory: **cd x** change directory to directory **x** **cd** change directory to home directory **cd ..** change directory to parent directory
mkdir x	make directory named **x**
pwd	print working directory (outputs the absolute path to the current directory)
ls	list files and directories **ls** list contents of the current directory **ls a b** list each of **a** and **b**. if either is a directory, list its contents options: **-F** marks directories to distinguish them from files **-l** to show permissions, etc **-R** recursive (from a directory, show all its directories and their directories...)
tree	show a recursive listing in a tree-like format in text. It helps to add the **-F** and/or the **-a** option (see **ls**)
rm f	remove file **f** permanently add option **-r** to remove directories and all their contents (**very dangerous!**) *DO NOT USE* **-r** *unless you need it!*
rmdir d	remove directory **d** permanently. The directory must be empty
cat	take files (or stdin) as input and con***cat***enate them, sending result to stdout **cat** copy stdin to stdout **cat a b ..** copy the contents of **a**, followed by contents of **b**, etc to stdout **cat > xxx** create (or overwrite) file **xxx** and place characters typed at the keyboard into it. The file is terminated with a contol-D character at the beginning of a line.
more	display files page by page
date	display the current date and time
id	display information about the current user **id yyy** display information about user **yyy**
script	record a terminal session in a file named typescript. Use exit to close the file. **script yyy** names the file **yyy** instead DO NOT USE the arrow keys to edit your commands when using script. Examine your output file using cat or more afterwards. If you can't read it, neither can I.
#	comment. If it is at the beginning of the line (or after a space), ignore the rest of the line (used in scripts for documentation)
man	get information from the Unix manual **man x** output the first manual page found that is named **x** **man -k yy** output the names of pages whose terse description contains **yy** **cmd --help** often works to get simple help
wc	options: count lines (**-l**), words (**-w**), and bytes(**-c**). default is **-lwc**
cp	copy files: **cp file1 file2** copy **file1** to **file2** **cp file(s) dir** copy files into a directory. overwrites files that are in the way. add **-r** option to copy directories (again, can be dangerous)
mv	rename or rearrange files or directories **mv file1 file2** (rename **file1** to **file2**. deletes existing **file2** if it exists) (we will cover more uses later)

file	display what type of an object it is: if it is a [regular] file, what kind of file is it, otherwise, is it a directory, or something else
sftp	secure file transfer protocol **sftp [user@]host** connect to **host** as **user** (defaults to the current login). the system you run **sftp** on is the *local system*. **host** is the *remote system*. **sftp** commands: **ls**, **pwd**, **mkdir**, **cd,rm** operate on the remote system. use **l** (lower-case L) prefix to each command to operate on the local system examples: **lpwd, lcd, lrm** **get x** transfer file **x** from the current directory on the remote system to the current directory on the local system **put x** transfer file **x** from the current directory on the local system to the current directory on the remote system
scp	secure copy program. Execute a single copy command between two hosts **scp xxx [user@]hostx:[path]** copy **xxx** from the current host to **hostx**, logging in as **user**, and placing the result at **path**. If **user** is missing, the current login is used. If **path** is missing, the **user's** home directory is used. **scp [user@]hostx:path xxx** login to **hostx** as **user** and copy **path** from that system to **xxx** on the current system. Again, **user** defaults to the current login. NOTE: if specified, **hostx** *must be followed by a colon (:) to be recognized.* options: **-r** must be used if you are copying a directory.

Control of standard input and output:

cmd > file1 send standard output of command **cmd** to file **file1** (overwriting file1**)**

cmd < file1 run command **cmd** and use the input from **file1** (connect standard input to **file1**)

cmd >> file1 append standard output of command **cmd** to file **file1**

cmd1 | cmd2 run commands **cmd1** and **cmd2** in parallel, sending the output of **cmd1** to the input of **cmd2**

Special characters

control-C (**^C**) - abort the program that you are currently running

control-D (**^D**) if you are typing input to a command from the keyboard, use control D to say "I'm done"

NOTE: control-D is a synonym for exit in many applications like the shell, so resist holding the control-D character too long - you don't want it to repeat.