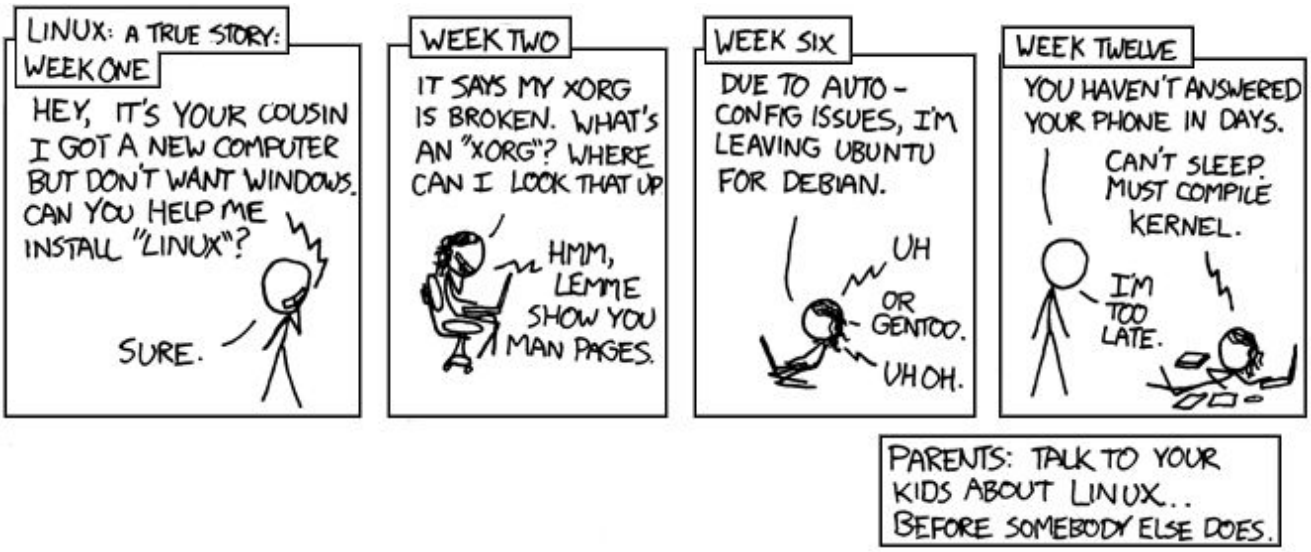
Linux Command Line



Basic Terminal

#	• Any command with a pound sign at the beginning is a no op
[up arrow]	• this will bring up the last command you used in the your terminal ○ if used multiple times will keep going up in your command history • can be useful when using long/ repeated commands • the down arrow will go back through commands in the other direction
history	• this will print out a list of your previous terminal commands • can be useful if you are trying to remember a complex command you did previously but can't quite remember
clear	• this will clear your terminal screen • can be useful when have just run something complex and would like a clean slate

man	• short for **man**ual
	• **usage:** `man [command]`
	  ○ ex: `man history`
	• will give you helpful usage information about certain commands/ system calls
	  ○ is especially useful to look at the flags for commands

echo	• this will basically just print something to your terminal window • **usage:** `echo [words]` • is really useful in bash scripts ○ **NOTE**: bash scripts are a way to run a collection of terminal commands as a single command
[ctrl] + c	• this will stop whatever is currently running in your terminal • can be especially useful when you accidentally run a program with an

	infinite loop ○ Or any other long running program/ script that you want to stop
[ctrl] + [shift] + c	● copy something from you terminal ● useful if trying to Google what an error means ● On mac: [command] + c
[ctrl] + [shift] + v	● paste into the terminal ● On mac: [command] + v ● useful if just Googled a way to do something cool
[ctrl] + r	● This will let you search previously used commands ● This is useful if you used a long command in the past and remember a word from it ● Pressing [ctrl] + r again will traverse previous occurrences of the search query
sudo	● short for **s**uper **u**ser **do** ● will allow you to run commands you normally aren't allowed to ● **usage:** `sudo [command]` ○ **NOTE:** requires you to have password permission
ssh	● short for **s**ecure **sh**ell ● **usage:** `ssh [host]@[IP]` ● works natively for Macs and Linux machines ● for Windows machines will need to use Putty ○ WinSCP is also a great program for Windows ● there is also a secure shell extension for Chrome
exit	● will close out of the terminal window without having the hit the little x in the corner

Directories and Files

ls	● short for **l**i**s**t ● lists all the files/ directories in the current directory ● you might also want to try the sl command on the lab machines
ls -al	● will list files along with their permissions ● permissions: ○ read - can view the stuff ○ write - can edit the stuff ○ execute - can run (for scripts and such) ● 3 sets ○ (owner) (group) (anyone)
pwd	● short for **p**rint **w**orking **d**irectory ● any easy way to know where you are in the file hierarchy if you forget

cd	• short for **c**hange **d**irectory
	• used to navigate between directories in your file structure
	• **usage:** `cd [directory]`
	  ○ can use “cd ..” to go back up the directory structure
	  ○ can also put in a full path instead of just a directory name
	  ■ “/” at front of directory will be an absolute path from your root directory
	  ■ no “/” at front of directory will be a relative path
	  ■ “.” just means current directory so ./hello.txt is the same as hello.txt

[tab]	• this will autocomplete whatever you are currently doing in the terminal • ex: `cd Doc + [tab]` would autocomplete Doc to Documents without you having to type out the whole thing
mkdir	• short for **m**a**k**e **dir**ectory • will make a new directory for you • **usage:** `mkdir [directory name]` ○ **NOTE:** can also use relative vs. absolute paths instead of just a directory name

cp	• short for **c**o**p**y
	• a way to make a copy of something in a different directory
	• **usage:** `cp [source/file name] [destination]`
	  ○ again can use relative or absolute paths for the source and destination
	  ○ **NOTE:** Copies to destinate and keeps the original in source as well
	  ○ **NOTE:** When copying directories you will need to use the -r flag

scp	• short for **s**ecure **c**o**p**y
	• a way to copy files between computers
	• **usage:** `scp [source] [destination]`
	  ○ from other computer: `scp [host]@[ip]:[source/file name] [destination on your computer]`
	  ○ to other computer: `scp [source/file name] [host]@[ip]:[destination on other computer]`

mv	• short for **m**o**v**e
	• a way to actually move files/directories around on your computer
	  ○ also an easy way to rename directories
	• **usage:** `mv [source] [destination]`
	  ○ as usual you can use either a relative or absolute path for the source and destination

rm	• short for r**e**m**o**ve • deletes a file • **usage:** `rm [file name]` • helpful things:

<table>
<tr><td></td><td>

<code>rm -rf [directory name]</code>

will delete a directory and everything inside it
use with caution, if you don’t give a destination for this it will delete EVERYTHING from your current directory down

</td></tr>
<tr><td>touch</td><td>

will either create a new file or update the last modified date on a file to the current date
usage: <code>touch [file]</code>

</td></tr>
<tr><td>cat</td><td>

short for catenate
will print a file’s contents to the terminal
usage: <code>cat [file]</code>

</td></tr>
<tr><td>chmod</td><td>

used to change permissions
usage: <code>chmod [new settings] [file]</code>
new setting options

<table>
<tr><td>Reference</td><td>Operator</td><td>Mode</td></tr>
<tr><td>u - user
g - group
o - others
a - all (everybody)</td><td>+ add
- remove
= set exactly</td><td>r - read
w - write
x - execute</td></tr>
</table>
</td></tr>
<tr><td>grep</td><td>

a way to search through file(s)
usage: <code>grep [search query] [file]</code>

can search for things using regex

helpful flags:

-n lists the line number next to matches
-r search recursively
* instead of a file name will search the whole directory

</td></tr>
<tr><td>find</td><td>

used to find out where a file lives in your file hierarchy
usage: <code>find [path] -name [file]</code>

if path is not given then will search the current directory and every directory it contains

</td></tr>
<tr><td>diff</td><td>

short for difference
shows the difference between 2 files
usage: <code>diff [file 1] [file 2]</code>
helpful flags:

-b ignore whitespace differences
-i ignore case
--side-by-side - see differences next to each other

</td></tr>
<tr><td>head</td><td>

Shows only the the beginning of a file, the number of lines based on input

</td></tr>
</table>

	• **usage: head -[number] [file]**
tail	• Shows only the end of a file, the number of lines based on input • **usage: tail -[number] [file]**

Redirection Input/ Output

|	• **usage:** `[command 1]	[command 2]` • will make the output from command on the left the input for the command on the right, can chain together as many commands as you need • ex: `man history	grep history` ○ will search for the word hello in the man pages for hello
> and >>	• **usage:** `[command] > [file]` • will redirect output on left into the file on the right • single > will will replace the contents of the file with the given output and double >> will append to the file • ex: `echo "hello" > hello.txt` • ex: `cat [file1] [file2] > [file3]`
<	• **usage: [command] < [file]** • will redirect thing on the right to be the input for the thing on the left • ex: `ProgramTakesInAge < 12` • is really good for testing projects that take in user input

Java Specific

java -version	• will tell you what version of Java is currently installed on your machine • will also tell you if java is not installed on your machine at all
javac	• used to compile a java program • **usage:** `javac [file]` ○ file must have the .java extension • if successful will create a .class file with the same name as the original Java file

java	• used to run a compiled Java file
	• **usage:** `java [name of .class file without extension]`
	  ○ ex: java Test
	  ■ don't put .class at the end of the file name
	  ■ this would have come from compiling a file called Test.java

Fun Stuff

cal	• will give you a little ASCII calendar of the current month with the current day highlighted

	• can be useful when you are having a tired moment and forget what your life looks like
date	• will give you the current date and time as a string • again useful if you are having a tired moment and just need to know time still works properly
yes	• will print the same phrase repeatedly in your terminal until you hit `[ctrl] + c` • **usage:** `yes [some words]`

cowsay	• will take a phrase and print a little ASCII art cow saying that phrase
	• **usage:** `cowsay [some words]`
	• can also pipe things into cowsay
	  ○ you could have a cow tell you your grep output
	• there are also many other animals you could do
	  ○ for a list do cowsay -l
	  ○ **usage for different animal:** `cowsay -f [animal file] [some words]`
	  ■ ex: `cowsay -f dragon-and-cow hello`