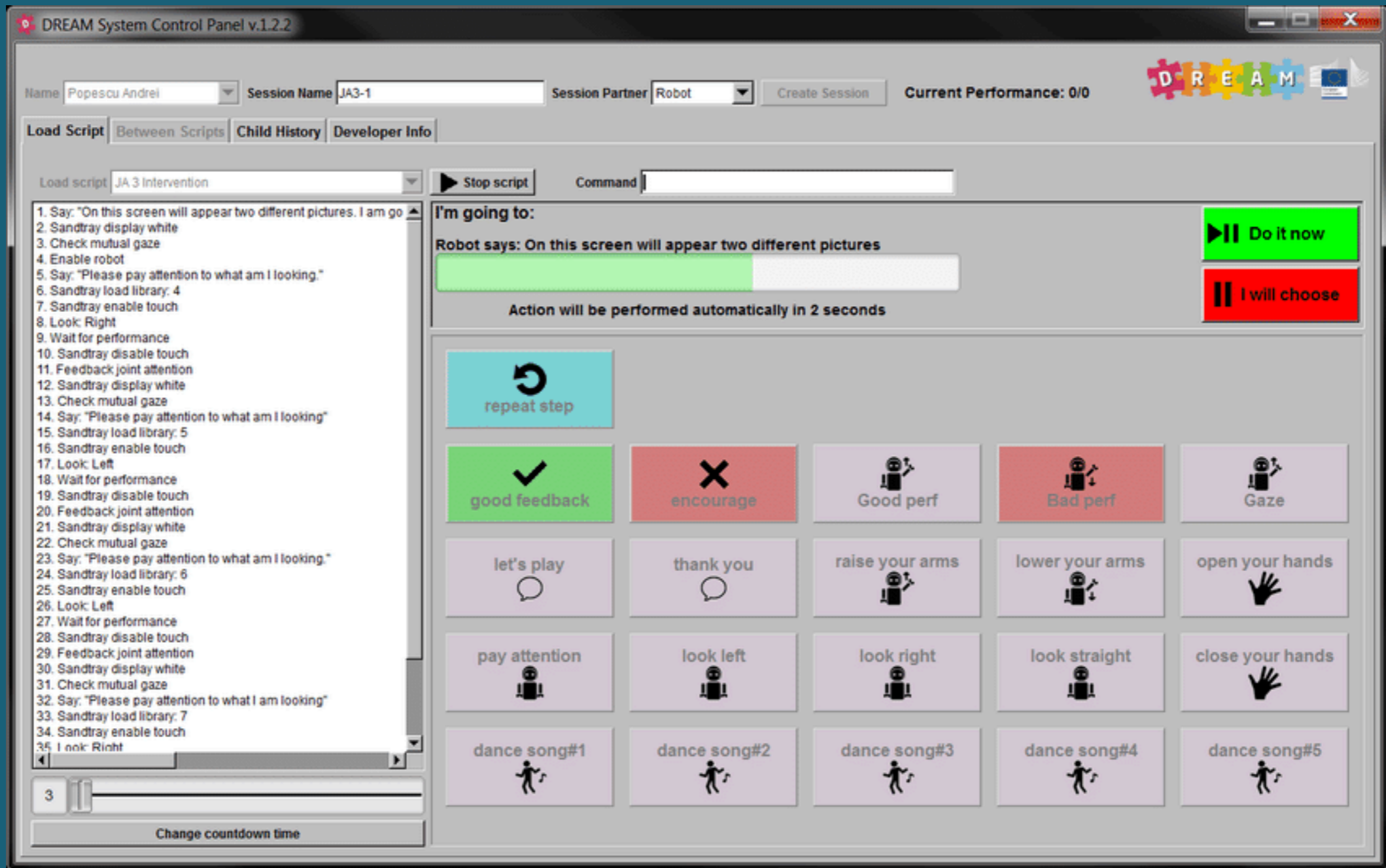


Unix tips and tricks



Most computational science is not done by GUI



GUI : Graphical User Interface

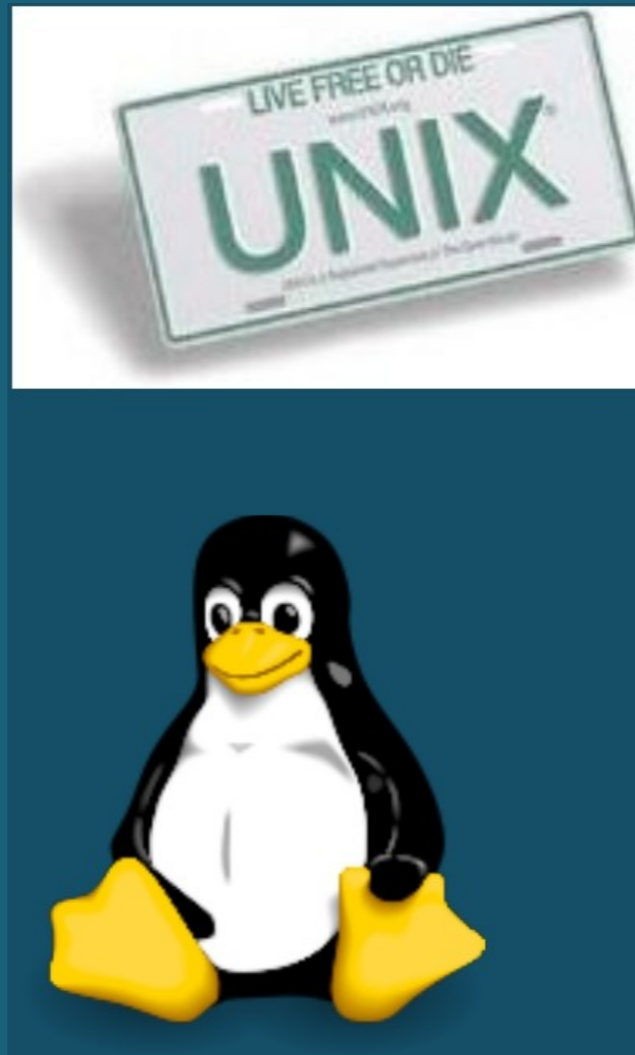
A typical science session

The screenshot displays a Mac OS X desktop environment during a science session. The desktop features several overlapping windows:

- Terminal (top left):** Shows PBS job submission commands and output. The commands include setting environment variables like `nodes=300`, `pn=4000mb`, and `nsn=120000mb`, followed by `mpirun -v -np 32 mpiexec -verbos -n 300 /home/2nd.icv/2lpt.paran.2`. The output shows job statistics such as `Stop= 77`, `total-avr= 10603200512`, and `total 7349428`.
- ImageMagick (top center):** Displays a colorful map of the solar system background, showing the orbits of planets and the Sun. The map is overlaid with a grid of latitude and longitude lines. A color scale on the left ranges from -37.9 to 4.00.
- emacs (bottom center):** Shows a README file for "Solar System Barycenter Background". The file contains metadata such as `AUTHOR: Kelly Holley-Bockelmann` and `DATE: January 5, 2006`, along with an abstract and compile instructions.
- Terminal (bottom right):** Lists a directory of files, including `SS3.f4,ecliptic.dat.gz`, `SS3.f5,ecliptic.dat.gz`, `thickdisk.f1.above.dat.gz`, and many others.

The system clock in the top right corner indicates the time is Sun 6:33 PM.

Scientific workstations use a UNIX-based operating system



Why UNIX?

- Multitasking
- Multiuser
- Access data stored in another computer
- Able to store large data sets

Bash shell

Program whose primary purpose is to read commands and run other programs.

Pros: high action-to-keystroke ratio, automating repetitive tasks, accessing network machines

Cons: Cryptic commands

Bash shell

It all starts with a \$

If the shell can't find a program whose name is the command you typed the error message will read 'command not found'

Bash shell

What it looks like :

```
vivek@nixcraft:/tmp$ vi hello.sh
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ chmod +x hello.sh
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ ls -l hello.sh
-rwxr-xr-x 1 vivek vivek 31 Jan 21 15:08 hello.sh
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ ./hello.sh
Hello World
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ bash hello.sh
Hello World
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ sh hello.sh
Hello World
vivek@nixcraft:/tmp$ cat hello.sh
#!/bin/bash
echo "Hello World"
vivek@nixcraft:/tmp$ █
```


Bash shell online

<https://tinyurl.com/terminalproject2021>

Or

- Open Terminal on your Mac
- Download shell-lesson-data.zip
 - <https://tinyurl.com/shelldata>
- Save it on your Desktop

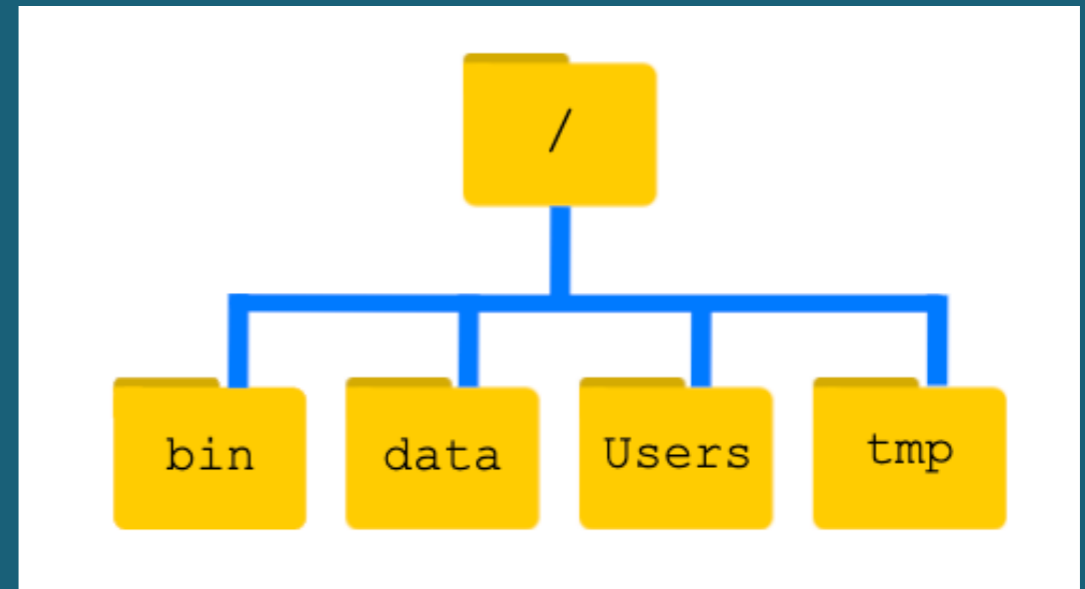
Bash shell

`/` is the root dir

Otherwise it's a path separator (opposite of Windows)

Absolute (from root) vs relative path (from current location)

`..` & `.` are special dirs.



Getting around the directory tree

- `ls` → lists content of directory
- `pwd` → print working directory
- `cd 'blah'` → change directory to 'blah'
- `mkdir 'blah'` → make directory 'blah'
- `rmdir 'blah'` → remove directory 'blah'

Practice: What directory are you in?

Practice: If on your computer, navigate to your Desktop directory

Practice: make a directory called 'Hello_World'

Practice: Go inside your new directory

**Hint use tab completion **

Feel the power: adding tags to your
command can customize it.

ls → lists content of directory

ls -l long format

ls -a list all files

ls -lhS long format, file size is 'human readable', sorted by file size

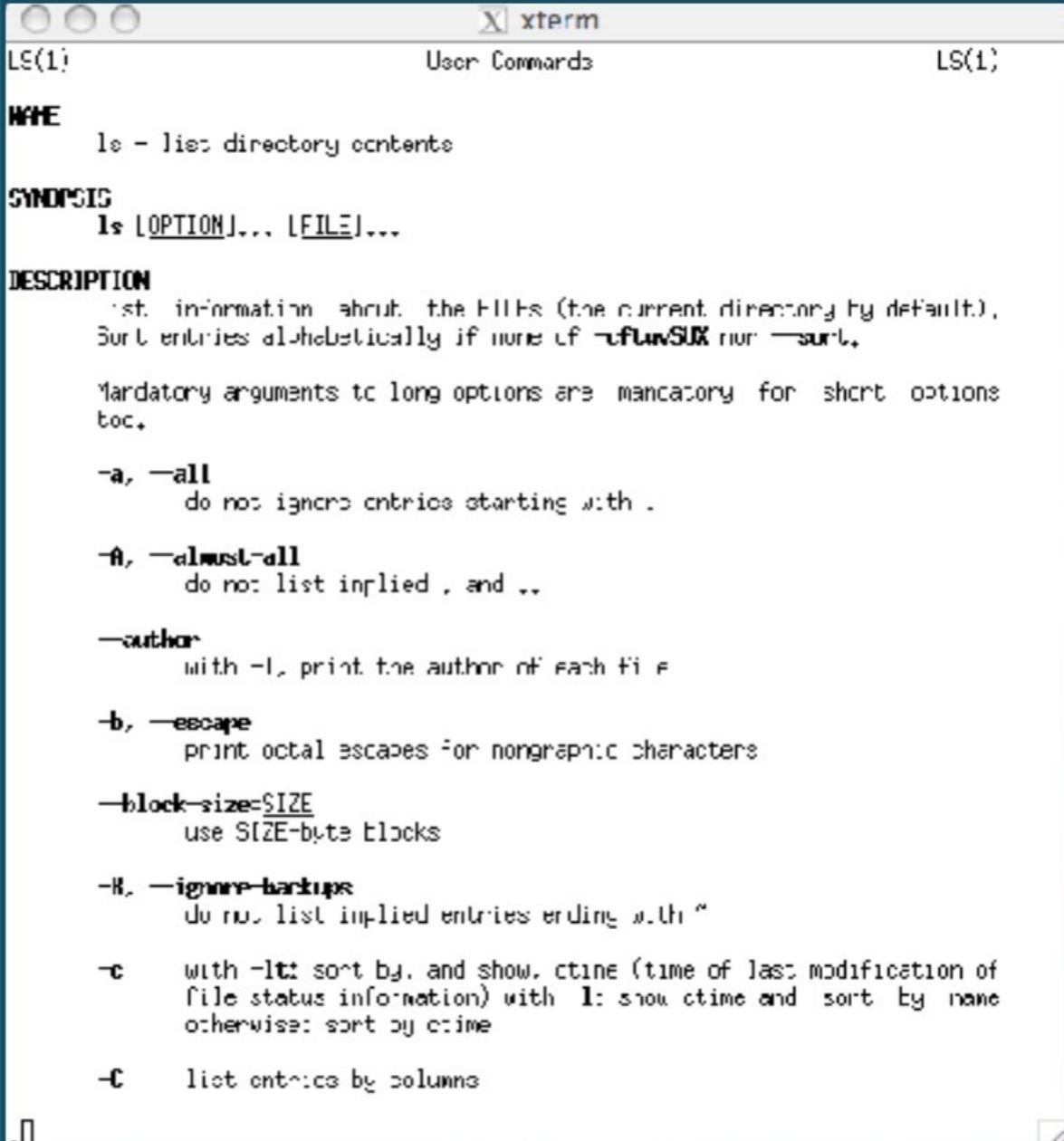
ls -F

```
total 101M
-rw-r--r-- 1 holleyjk astro 58M May 17 10:30 hdf5-1.8.4-patch1.tar
-rw----- 1 holleyjk astro 6.7M May 17 11:11 king_snap.dat
-rw----- 1 holleyjk astro 6.3M Aug 10 2009 mri.pdf
-rw----- 1 holleyjk astro 3.4M Jul 21 18:24 100719_Yaqiongcareerproposal.doc
-rw----- 1 holleyjk astro 2.6M Jan 1 2010 merger_44_20_2366.png
```


But how do I know the secret tags for each command?

man command

man intro is a good
place to start!



```
LS(1)                               User Commands                               LS(1)

NAME
  ls - list directory contents

SYNOPSIS
  ls [OPTION]... [FILE]...

DESCRIPTION
  List information about the FILES (the current directory by default).
  Sort entries alphabetically if none of -ftuvsUX nor --sort.

  Mandatory arguments to long options are mandatory for short options
  too.

  -a, --all
      do not ignore entries starting with .

  -A, --almost-all
      do not list implied . and ..

  --author
      with -l, print the author of each file

  -b, --escape
      print octal escapes for nongraphic characters

  --block-size=SIZE
      use SIZE-byte blocks

  -H, --ignore-backups
      do not list implied entries ending with #

  -c
      with -lt: sort by, and show, ctime (time of last modification of
      file status information) with l: show ctime and sort by name
      otherwise: sort by ctime

  -C
      list entries by columns
```

Key Points

- The file system is responsible for managing information on the disk.
- Information is stored in files, which are stored in directories (folders).
- Directories can also store other directories, which then form a directory tree.
- `cd [path]` changes the current working directory.
- `ls [path]` prints a listing of a specific file or directory; `ls` on its own lists the current working directory.
- `pwd` prints the user's current working directory.
- `/` on its own is the root directory of the whole file system.

Key Points

- Most commands take options (flags) that begin with a -
- A relative path specifies a location starting from the current location.
- An absolute path specifies a location from the root of the file system.
- Directory names in a path are separated with / on Unix, but \ on Windows.
- .. means 'the directory above the current one'
- . on its own means 'the current directory'.

Making and adding files

With Terminal you can write scripts, documents using vim*

Practice: On you bash shell type **vim Hello.txt**

You should have an empty text document

- To write using vim.
- Hit **i** on your keyboard (insert)
- Type your string **Hello world**
- Hit **esc** on your keyboard to stop writing
- Hit **:wq** on your keyboard to write and quite the text file
- Repeat the steps and make a empty text file

- Back in bash (\$) see if your text files are there with **ls**

*Vim is a text editor, people both love it and hate it.

If you have a prefer text editor feel free to use that one.


```
~$ pwd
/home/user      Current directory
~$ mkdir 'Hello_World'  Make directory
~$ ls          List things inside directory. Hello_World is bold because it's a directory
Hello_World  'Welcome to CoCalc.term'
~$ cd Hello_World/      Change to the Hello_World directory
~/Hello_World$ vim hello.txt  Create hello.txt
~/Hello_World$ vim hello.txt
~/Hello_World$ cd ..      Go back one directory
~$ ls
Hello_World  'Welcome to CoCalc.term'
~$ █
```

Manipulating files

- `cp file1 file2` → copy 'file1' to 'file2'
- `mv file1 file2` → move 'file1' to 'file2'
- `rm 'blah'` → remove file 'blah'

*Terminal has no trash bin!

~ means /home/username/

• means the current directory

•• means go back 1 directory

* means wildcard

Copy a file

Practice: Copy a file

- To copy hello_world.
 - `cp hello_world.txt hello_world_2.txt`
 - or
 - `cp hello_world.txt [path/filename]`
- Are the files the same?

Move a file

Practice: Move a file

- To move hello_world up one directory
 - `mv hello_world.txt ..`
 - or
 - `mv hello_world.txt [path/filename]`
- What is the new path for hello_world.txt?

Copy a directory

Practice: Copy a directory

- To copy Hello_world
 - `cp -r Hello_world Hello_world_backup`
- List both directories
 - `ls Hello_world Hello_world_backup`

We can also copy a directory and all its contents by using the recursive option `-r`

Move a directory

Practice: Move a directory

- Make a directory called `Research_Test` inside `Hello_World` folder
 - `mkdir ~/shell-lessons-Crisel-1/Hello_world/Research_Test`
 - Inside `Hello_world` > `mkdir Research_Test`
- To move `Research_Test` to `shell-lessons-Crisel-1`
 - `mv Research_Test/ ~/shell-lessons-Crisel-1/`

Remove a file

Practice: Remove a file/Delete a file (forever)

- Go to your Hello_World folder
- Remove hello_world_2.txt
- `rm hello_world_2.txt`
- List all files
- Is hello_world_2.txt in your directory?

- What happened if you try:
 - `rm -i Hello_world/hello_world.txt`

Remove a directory

Practice: Remove a directory and everything inside it

- Remove your Hello_World directory safely
 - `rm -ri Hello_World/`
- List the items in your directory and make sure you don't have the Hello_World folder

Operations with multiple files and directories

Practice: Go to [shell-lesson-data-YourName/molecules/](#)

- * Wildcard matches characters
- List all the files inside the [molecules/](#) directory

Try it out : Which command(s) will produce this output?

ethane.pdb methane.pdb

ls *t*ane.pdb

ls *t?ne.*

ls *t??ne.pdb

ls ethane.*

Key Points

- **cp** [old] [new] copies a file.
- **mkdir** [path] creates a new directory.
- **mv** [old] [new] moves (renames) a file or directory.
- **rm** [path] removes (deletes) a file.
- ***** matches zero or more characters in a filename, so *.txt matches all files ending in .txt.
- **?** matches any single character in a filename, so ?.txt matches a.txt but not any.txt.
- The shell does not have a trash bin
- Most files' names are something.extension. The extension is used to indicate the type of data in the file.

Pipes and Filters

Practice: Go to [shell-lesson-data-YourName/molecules/](#)

- Word Count (**wc**)
- `$ wc cubane.pdb`
 - `20 156 1158 cubane.pdb`
 - Number of lines, words, and characters in files
- *Practice:*
- `wc *.pdb`
- `wc -l *.pdb`
- Save output
 - `wc -l *.pdb > lengths.txt`

Pipes and Filters

Check: Did you saved lengths.txt ?

- Peek inside lengths.txt with:
 - **cat** - 'concatenate' i.e. join together, and it prints the contents of files one after another.
 - **head** – displays only the first 10 lines of the file
 - **less** - screenful of the file, and then stops. You can go forward one screenful by pressing the spacebar, or back one by pressing b. Press q to quit.

Pipes and Filters

Check: Sort

- Sort lengths.txt with:
 - `$ sort -n lengths.txt > sorted-lengths.txt`
 - `$ head -n 1 sorted-lengths.txt`

Pipes and Filters

Check: `echo` prints statement

- `$ echo The echo command prints text`
 - `The echo command prints text`

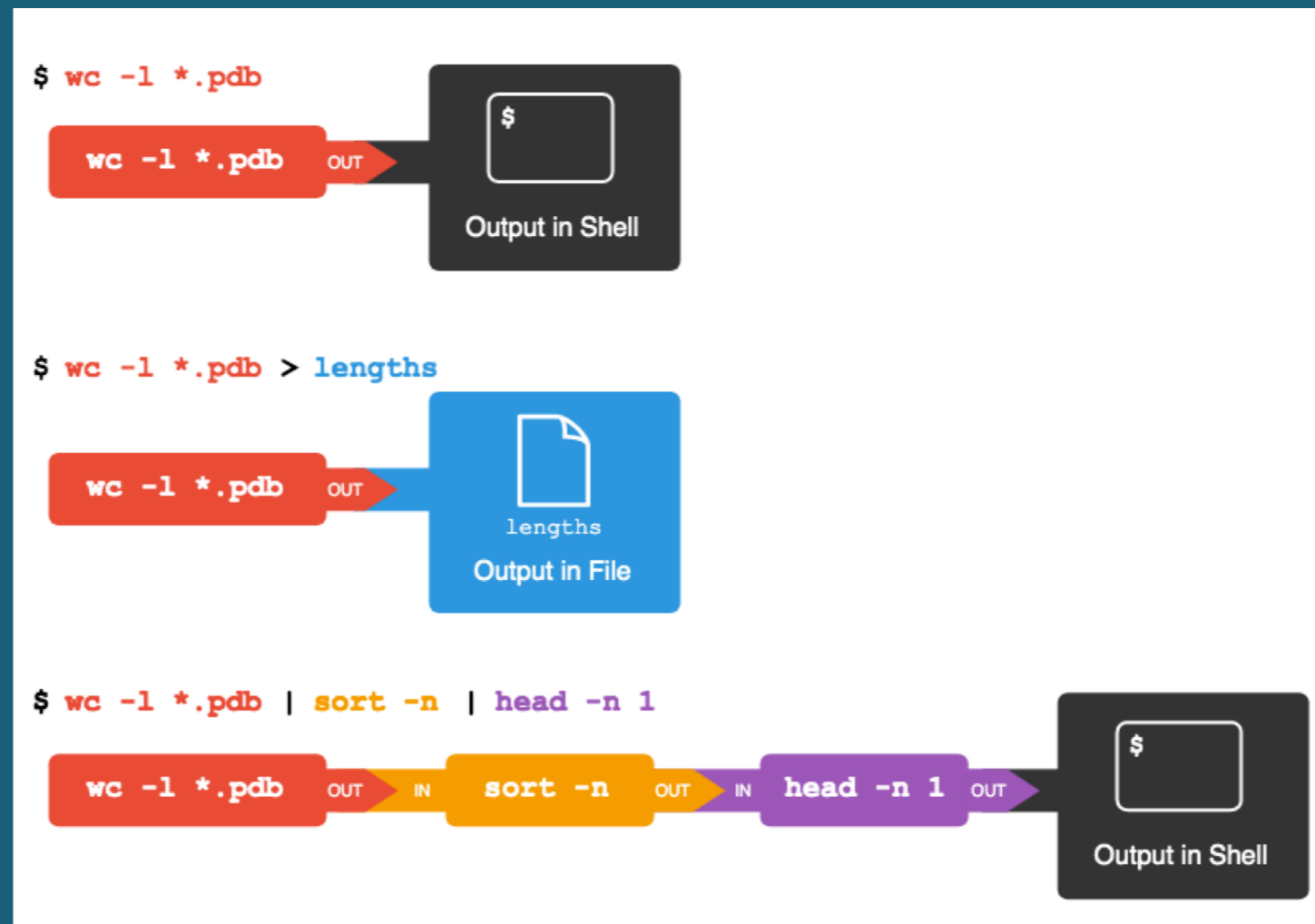
Check: `>` vs `>>`

- `$ echo hello > testfile01.txt`
- `$ echo hello >> testfile02.txt`
- `>` 'hello' is written to testfile01.txt, but the file gets overwritten each time we run the command.
- `>>` 'hello' is written to testfile01.txt, but the file gets overwritten each time we run the command.

Pipes and Filters

Check: More commands in one line | (pipe)

- `$ sort -n lengths.txt | head -n 1`
- `$ wc -l *.pdb | sort -n`
- `$ wc -l *.pdb | sort -n | head -n 1`



Key Points

- **wc** counts lines, words, and characters in its inputs.
- **cat** displays the contents of its inputs.
- **sort** sorts its inputs.
- **head** displays the first 10 lines of its input.
- **tail** displays the last 10 lines of its input.
- **command > [file]** redirects a command's output to a file (overwriting any existing content).

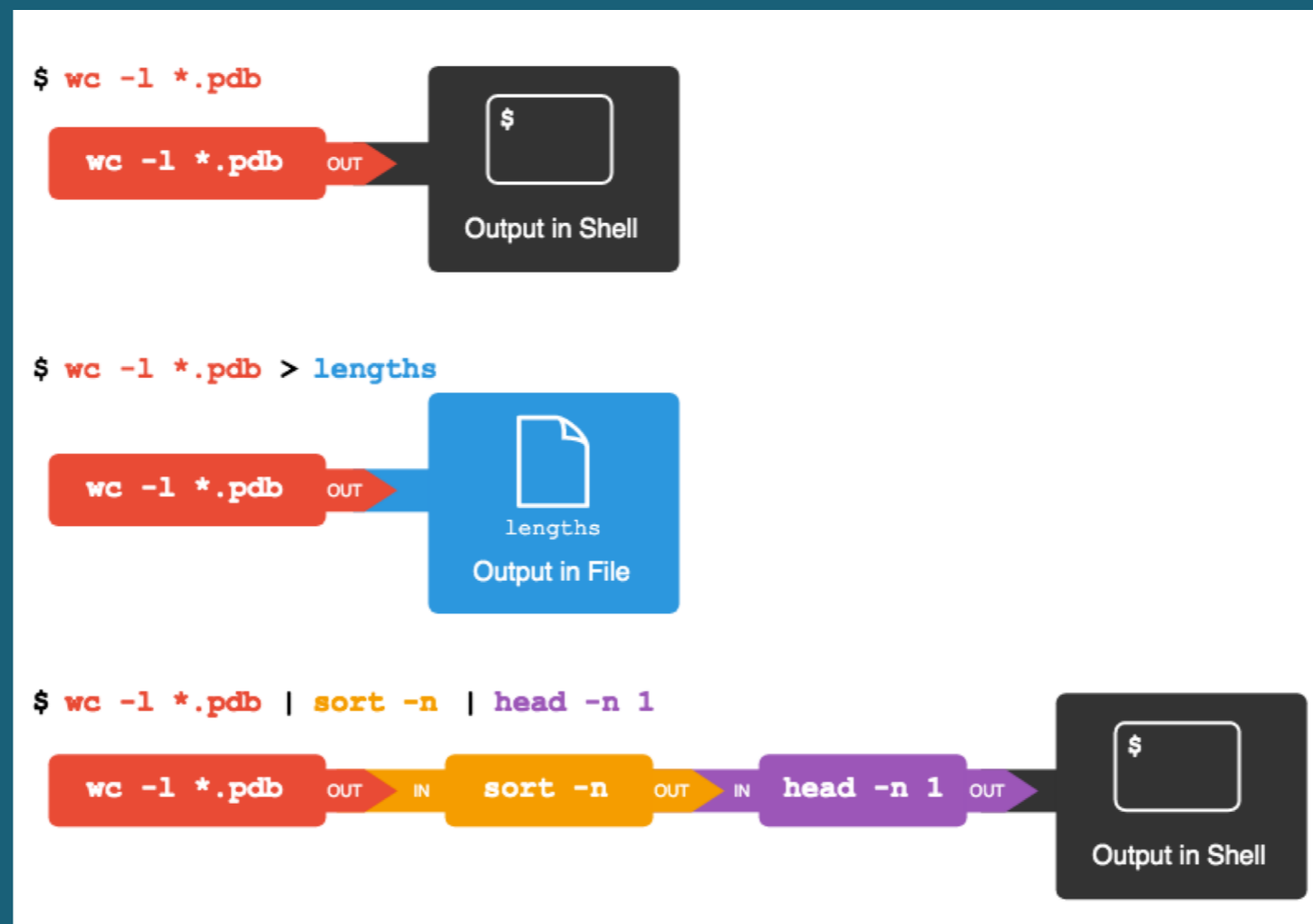
Key Points

- `command >> [file]` appends a command's output to a file.
- `[first] | [second]` is a pipeline: the output of the first command is used as the input to the second.
- The best way to use the shell is to use pipes to combine simple single-purpose programs (filters).

Loops

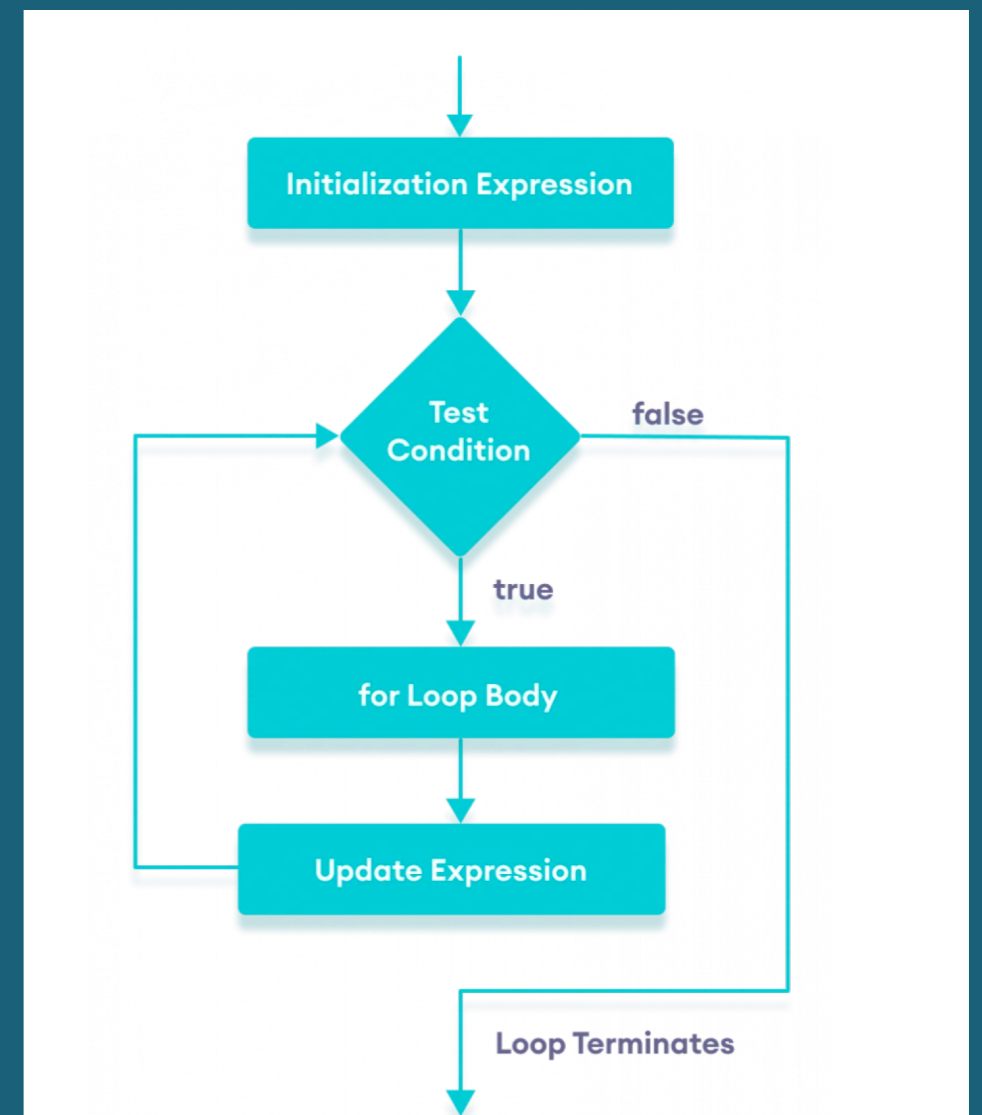
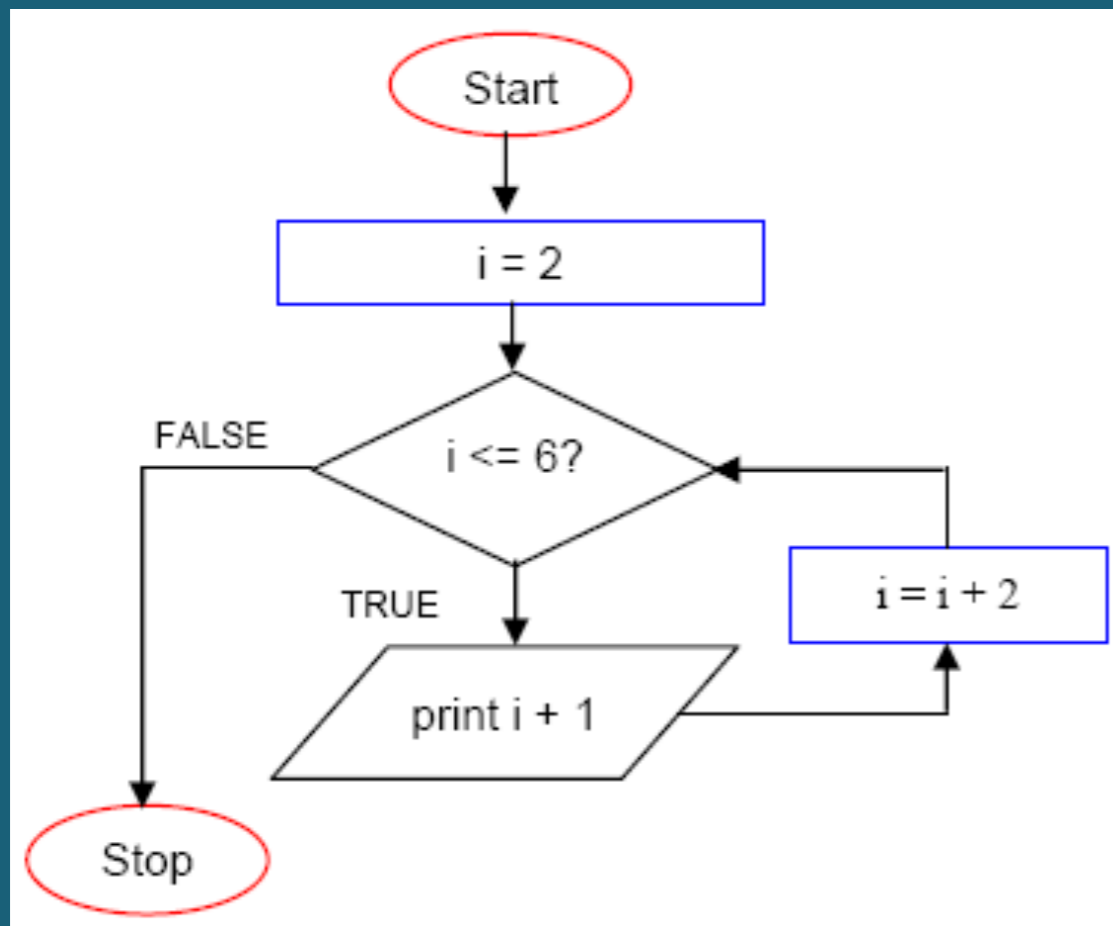
Check: More commands in one line | (pipe)

- `$ sort -n lengths.txt | head -n 1`
- `$ wc -l *.pdb | sort -n`
- `$ wc -l *.pdb | sort -n | head -n 1`



Loops

Loops are a programming construct which allow us to repeat a command or set of commands for each item in a list. As such they are key to productivity improvements through automation



We will cover this also with Python

Loops

```
$for thing in list_of_things
do
    operation_using $thing
done
```

Practice: Go to [shell-lesson-data-YourName/creatures/](#)

```
$ for name in basilisk.dat minotaur.dat unicorn.dat
➤ do
    ➤ head -n 2 $name | tail -n 1
➤ > done
```

COMMON NAME: basilisk

COMMON NAME: minotaur

COMMON NAME: unicorn

Loops

```
$for thing in list_of_things  
do  
    operation_using $thing  
done
```

Practice: Go to [shell-lesson-data-YourName/creatures/](#)

```
$ for filename in *.dat  
➤ do  
    ➤ cp $filename original-$filename  
➤ done
```

Key Points

- A for loop repeats commands once for every thing in a list.
- Every for loop needs a variable to refer to the thing it is currently operating on.
- Use `$name` to expand a variable (i.e., get its value).
- Do not use spaces, quotes, or wildcard characters such as `*` or `?` in filenames, as it complicates variable expansion.
- Give files consistent names that are easy to match with wildcard patterns to make it easy to select them for looping.

Bash scripts

How can I save and re-use commands?

Practice: Go to [shell-lesson-data-YourName/molecules/](#)

With Terminal you can write scripts, documents using vim*

Practice: Type `vim middle.sh`

You should have an empty text document

- To write using vim.
- Hit **i** on your keyboard (insert)
- Type your code `head -n 15 octane.pdb | tail -n 5`
- Hit **esc** on your keyboard to stop writing
- Hit `:wq` on your keyboard to write and quite the text file

- Back in bash (\$) `bash middle.sh`

Bash scripts

How can I save and re-use commands?

Practice: Type more general script
`vim middle.sh`

- To write using vim.
- Hit **i** on your keyboard (insert)
- Type your code `head -n 15 "$1" | tail -n 5`
 - Previously (`head -n 15 octane.pdb | tail -n 5`)
- Hit **esc** on your keyboard to stop writing
- Hit **:wq** on your keyboard to write and quite the text file

- Back in bash - `$ bash middle.sh octane.pdb`
- Back in bash - `$ bash middle.sh pentane.pdb`

Bash scripts

How can I save and re-use commands?

Practice: Type more general script
`vim script.sh`

- To write using vim.
- Hit **i** on your keyboard (insert)
- Type your code
 - `head -n $2 $1`
 - `tail -n $3 $1`
- Hit **esc** on your keyboard to stop writing
- Hit **:wq** on your keyboard to write and quite the text file
- Back in bash - `$ bash script.sh '*.pdb' 1 1`

Key Points

- Save commands in files (usually called shell scripts) for re-use.
- `bash [filename]` runs the commands saved in a file.
- `$1`, `$2`, etc., refer to the first command-line argument, the second command-line argument, etc.

Finding files and things inside files

Practice: Go to [shell-lesson-data-YourName/writing/](#)

- Look inside [haiku.txt](#)
- Which lines have contain 'not' ?
- **grep** (find)
- `$ grep not haiku.txt`
 - Is not the true Tao, until
 - "My Thesis" not found
 - Today it is not working

Finding files and things inside files

Practice: Go to [shell-lesson-data-YourName/writing/](#)

- \$ `grep The haiku.txt`
- \$ `grep -w The haiku.txt`

- \$ `grep -w "is not" haiku.txt`

- \$ `grep -i => case insensitive`
- \$ `grep -n => number lines`
- \$ `grep -v => invert search`
- \$ `grep -r => recursive search`
- \$ `grep --help` (no space)

Finding files and things inside files

Practice: Lets search recursively through the whole directory `shell-lesson-data-YourName/writing/`

- `$ grep -r Yesterday .`
 - `data/LittleWomen.txt:Yesterday, when Aunt was asleep and I was trying to be as still as a`
 - `data/LittleWomen.txt:Yesterday at dinner, when an Austrian officer stared at us and then`
 - `data/LittleWomen.txt:Yesterday was a quiet day spent in teaching, sewing, and writing in my`
 - `haiku.txt:Yesterday it worked`

Finding files and things inside files

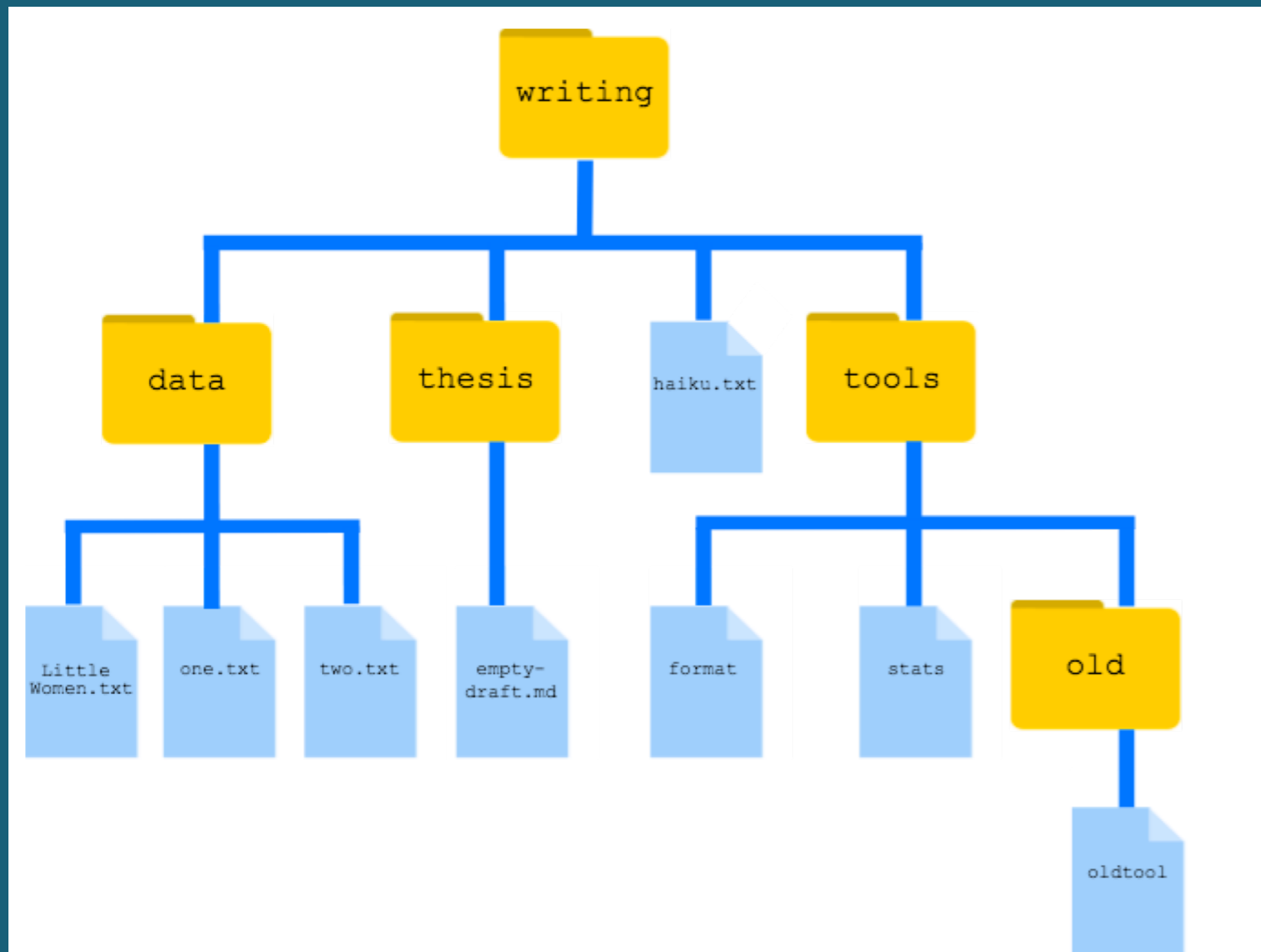
Practice: Go to [shell-lesson-data-YourName/writing/](#)

- `$ grep -E "^o" haiku.txt`
- We use the `-E` option and put the pattern in quotes to prevent the shell from trying to interpret it.
- The `^` in the pattern anchors the match to the start of the line.
- The `.` matches a single character
- `o` matches an actual 'o'.



Finding files and things inside files

Practice: Go to [shell-lesson-data-YourName/writing/](#)



```
$ find .
```

```
$ find . -type d
```

```
$ find . -type f
```

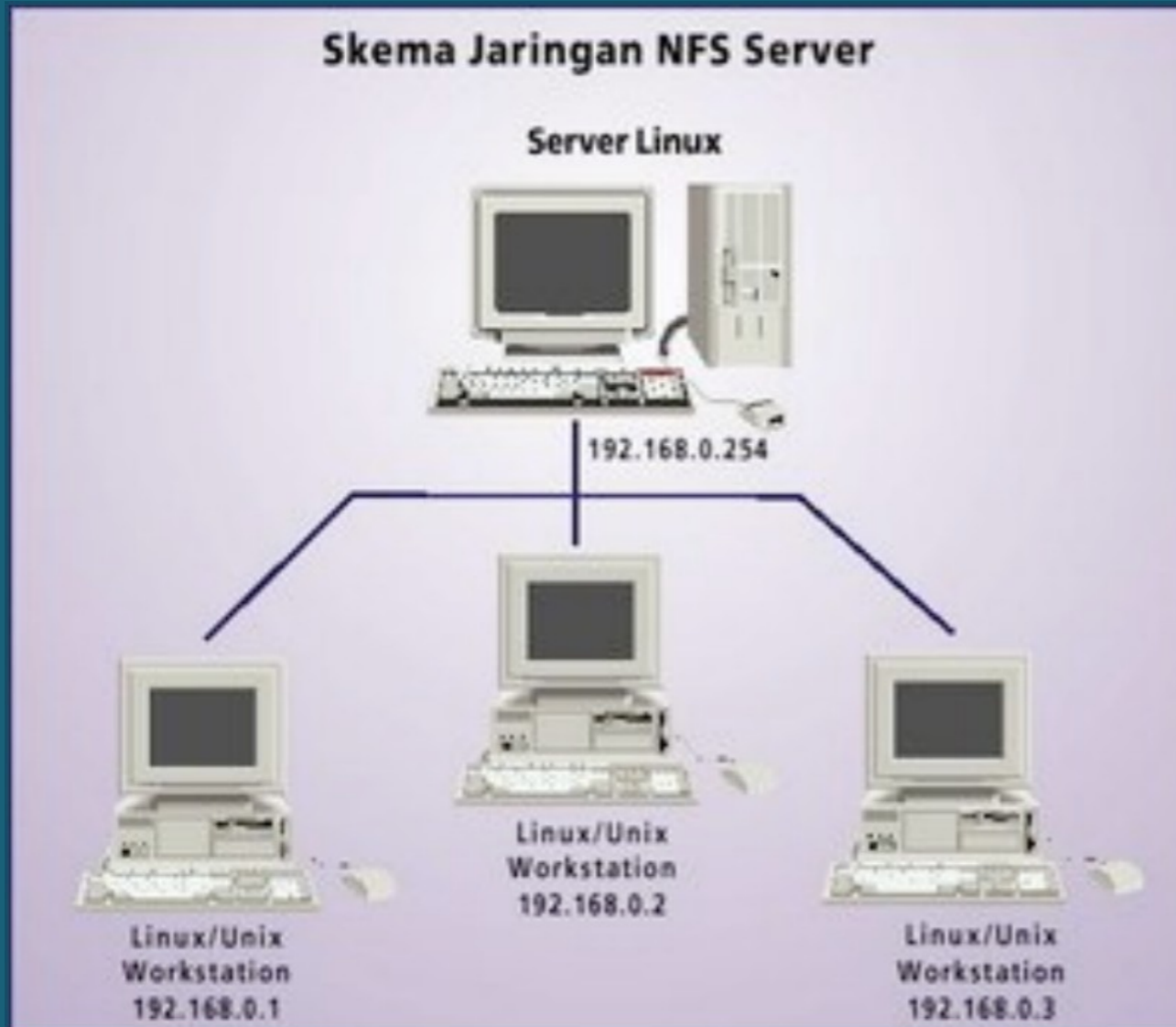
```
$ find . -name "*.txt"
```


Some more commands

- `rm -rf` for directories (careful!)
- Maybe make `rm` interactive using `rm -i` in `~/.aliases`
- `history`
- Unpack files from arXiv : `tar -xvf <filename>`

Translate this:

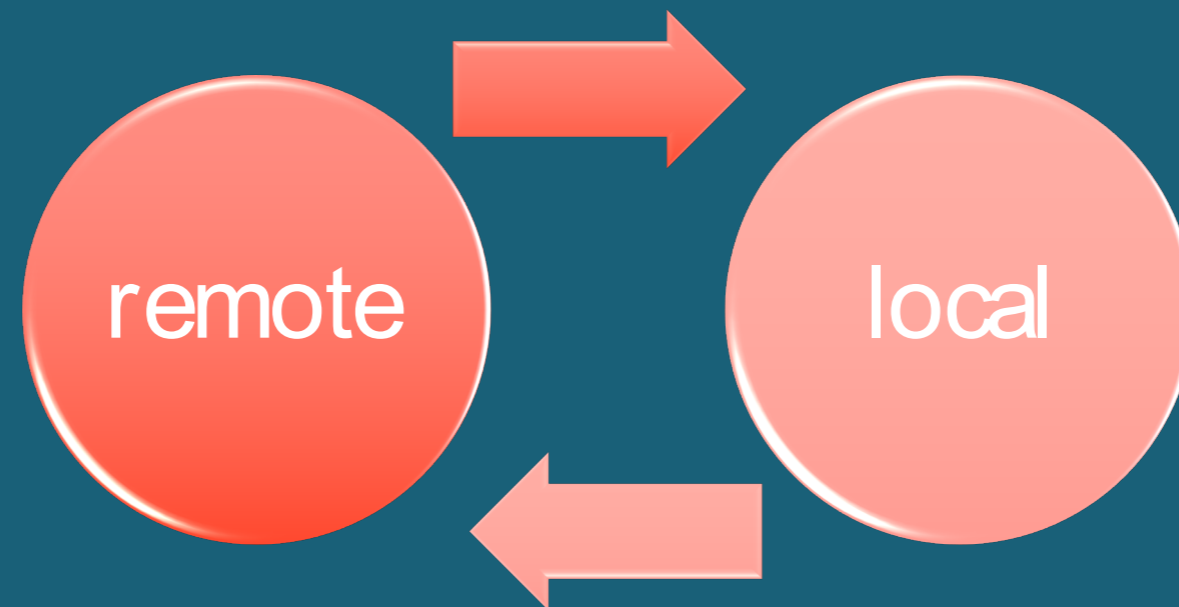
- `cp ~/*.sm ../.`
- `mv ../density.sm ~/bootcamp/dense.s`
- `rm *`



When you ssh into the vpac/accre network, you go to your home directory. Your directory can be accessed through any of these network machines. And, you can (theoretically) access data on other user's accounts, too.

scp

secure copy protocol



~ stands for home dir (make sure you're in the right place!)

Use * wildcard to grab multiple files that meet some condition



To log into
another machine:

`ssh -YC username@computer's whole address`
example: `ssh -YC holleyjk@vpac03.phy.vanderbilt.edu`



To transfer data
between machines:

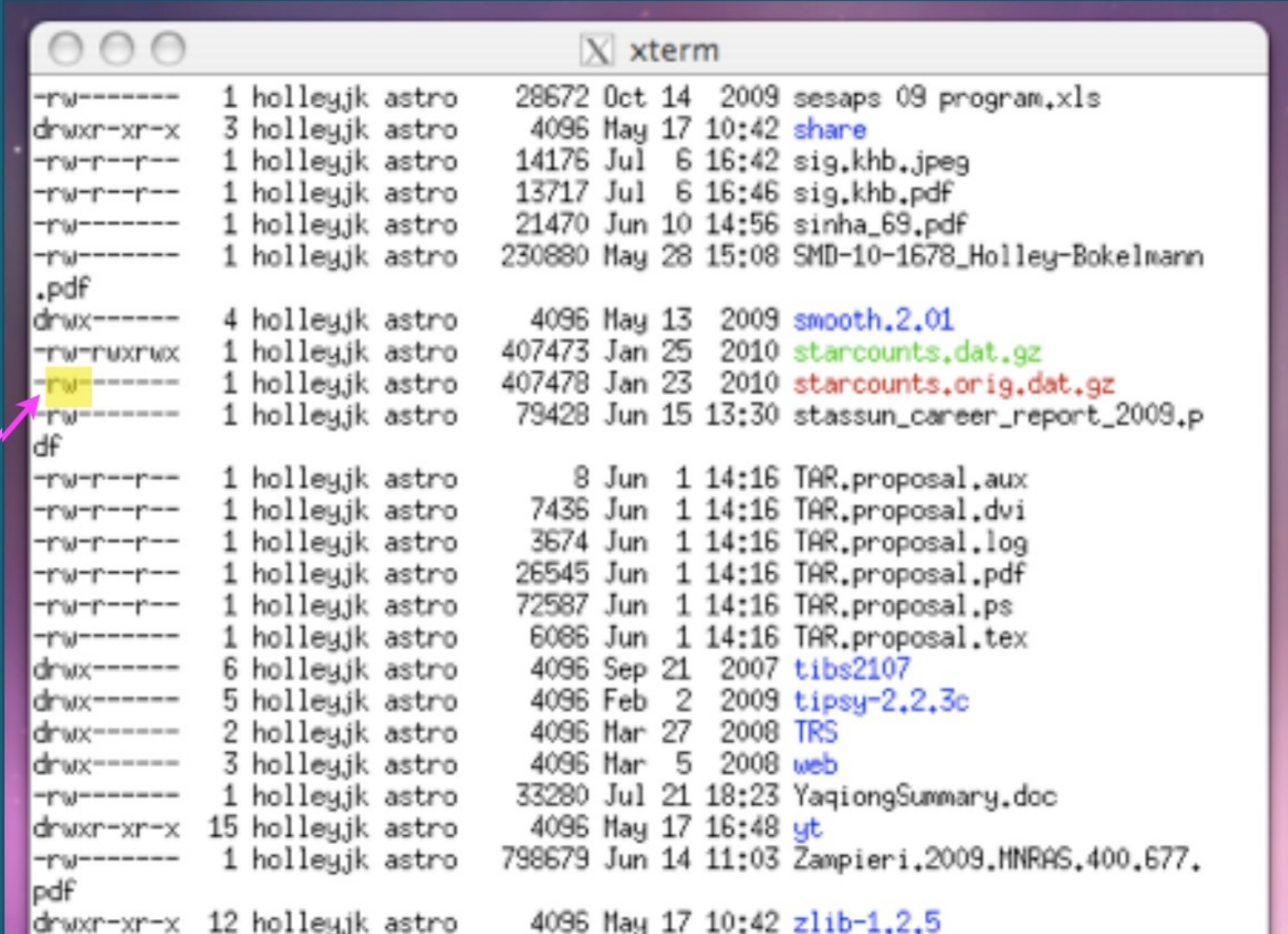
`scp -pr complete path of file you
want to move complete path
describing where you want to put it`

example 1: `scp -pr holleyjk@vpac03.phy.vanderbilt.edu:/home/holleyjk/superfile
/Users/kelly/Desktop/superfile.moo`

example 2: `scp -pr /Users/kelly/awesomefile
holleyjk@vmplogin.accre.vanderbilt.edu:/home/holleyjk/.`

Your data is protected by file permissions

Aside: How did I get this list?



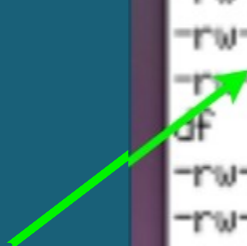
```
xterm
-rw----- 1 holleyjk astro 28672 Oct 14 2009 sesaps_09_program.xls
drwxr-xr-x 3 holleyjk astro 4096 May 17 10:42 share
-rw-r--r-- 1 holleyjk astro 14176 Jul 6 16:42 sig.khb.jpeg
-rw-r--r-- 1 holleyjk astro 13717 Jul 6 16:46 sig.khb.pdf
-rw----- 1 holleyjk astro 21470 Jun 10 14:56 sinha_69.pdf
-rw----- 1 holleyjk astro 230880 May 28 15:08 SMD-10-1678_Holley-Bokelmann
.pdf
drwx----- 4 holleyjk astro 4096 May 13 2009 smooth.2.01
-rw-rwxrwx 1 holleyjk astro 407473 Jan 25 2010 starcounts.dat.gz
-rw----- 1 holleyjk astro 407478 Jan 23 2010 starcounts.orig.dat.gz
-rw----- 1 holleyjk astro 79428 Jun 15 13:30 stassun_career_report_2009,p
df
-rw-r--r-- 1 holleyjk astro 8 Jun 1 14:16 TAR.proposal.aux
-rw-r--r-- 1 holleyjk astro 7436 Jun 1 14:16 TAR.proposal.dvi
-rw-r--r-- 1 holleyjk astro 3674 Jun 1 14:16 TAR.proposal.log
-rw-r--r-- 1 holleyjk astro 26545 Jun 1 14:16 TAR.proposal.pdf
-rw-r--r-- 1 holleyjk astro 72587 Jun 1 14:16 TAR.proposal.ps
-rw----- 1 holleyjk astro 6086 Jun 1 14:16 TAR.proposal.tex
drwx----- 6 holleyjk astro 4096 Sep 21 2007 tibs2107
drwx----- 5 holleyjk astro 4096 Feb 2 2009 tipsy-2.2.3c
drwx----- 2 holleyjk astro 4096 Mar 27 2008 TRS
drwx----- 3 holleyjk astro 4096 Mar 5 2008 web
-rw----- 1 holleyjk astro 33280 Jul 21 18:23 YaqiongSummary.doc
drwxr-xr-x 15 holleyjk astro 4096 May 17 16:48 yt
-rw----- 1 holleyjk astro 798679 Jun 14 11:03 Zampieri,2009,MNRAS,400,677,
pdf
drwxr-xr-x 12 holleyjk astro 4096 May 17 10:42 zlib-1.2.5
```

user
permissions
r=Read
w=Write
x=eXecute

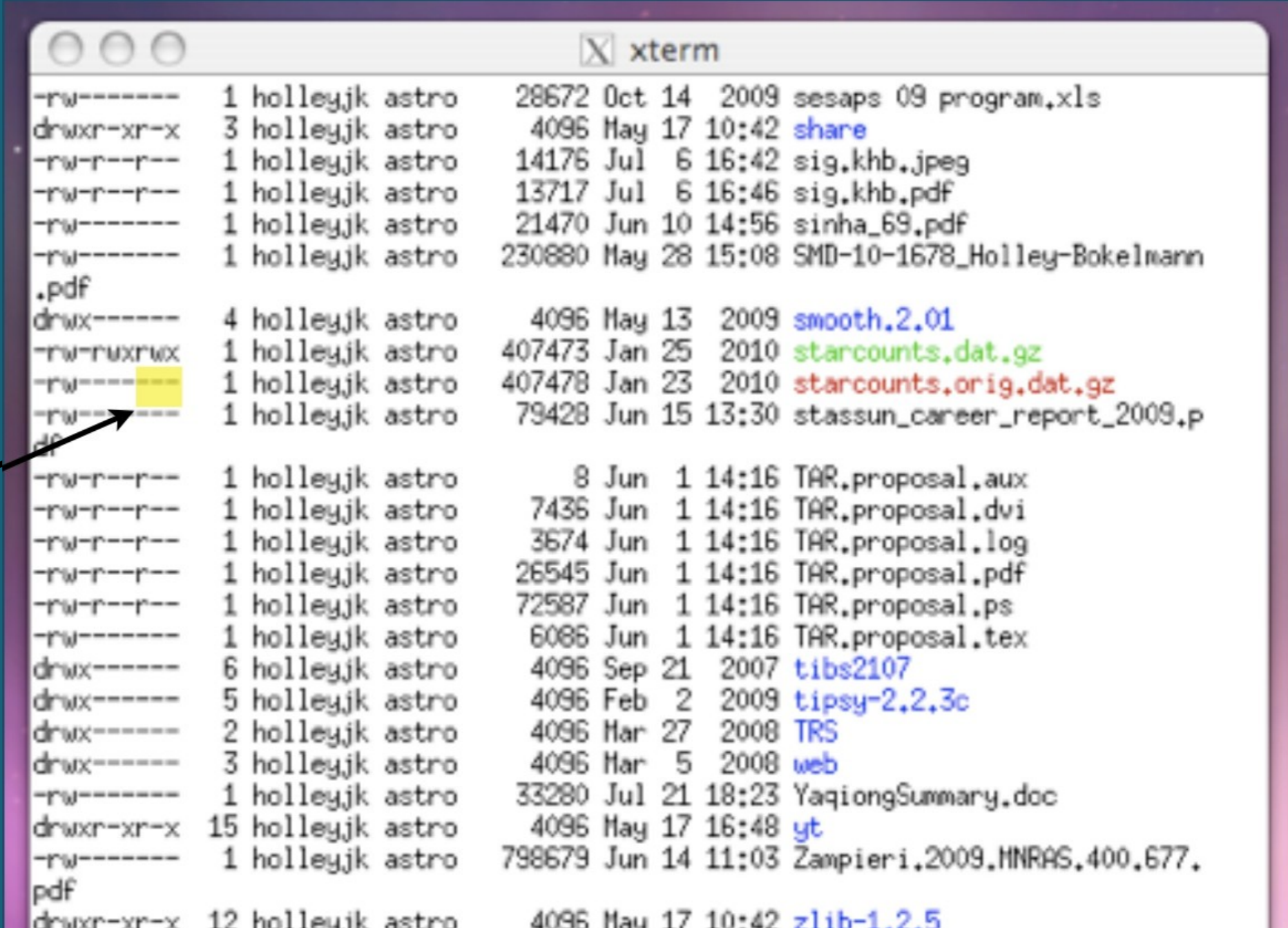
Your data is protected by file permissions

```
xterm
-rw----- 1 holleyjk astro 28672 Oct 14 2009 sesaps_09_program.xls
drwxr-xr-x 3 holleyjk astro 4096 May 17 10:42 share
-rw-r--r-- 1 holleyjk astro 14176 Jul 6 16:42 sig.khb.jpeg
-rw-r--r-- 1 holleyjk astro 13717 Jul 6 16:46 sig.khb.pdf
-rw----- 1 holleyjk astro 21470 Jun 10 14:56 sinha_69.pdf
-rw----- 1 holleyjk astro 230880 May 28 15:08 SMD-10-1678_Holley-Bokelmann
.pdf
drwx----- 4 holleyjk astro 4096 May 13 2009 smooth.2.01
-rw-ruxrwx 1 holleyjk astro 407473 Jan 25 2010 starcounts.dat.gz
-rw----- 1 holleyjk astro 407478 Jan 23 2010 starcounts.orig.dat.gz
-r----- 1 holleyjk astro 79428 Jun 15 13:30 stassun_career_report_2009.p
df
-rw-r--r-- 1 holleyjk astro 8 Jun 1 14:16 TAR.proposal.aux
-rw-r--r-- 1 holleyjk astro 7436 Jun 1 14:16 TAR.proposal.dvi
-rw-r--r-- 1 holleyjk astro 3674 Jun 1 14:16 TAR.proposal.log
-rw-r--r-- 1 holleyjk astro 26545 Jun 1 14:16 TAR.proposal.pdf
-rw-r--r-- 1 holleyjk astro 72587 Jun 1 14:16 TAR.proposal.ps
-rw----- 1 holleyjk astro 6086 Jun 1 14:16 TAR.proposal.tex
drwx----- 6 holleyjk astro 4096 Sep 21 2007 tibs2107
drwx----- 5 holleyjk astro 4096 Feb 2 2009 tipsy-2.2.3c
drwx----- 2 holleyjk astro 4096 Mar 27 2008 TRS
drwx----- 3 holleyjk astro 4096 Mar 5 2008 web
-rw----- 1 holleyjk astro 33280 Jul 21 18:23 YaqiongSummary.doc
drwxr-xr-x 15 holleyjk astro 4096 May 17 16:48 yt
-rw----- 1 holleyjk astro 798679 Jun 14 11:03 Zampieri,2009,MNRAS,400,677.
pdf
drwxr-xr-x 12 holleyjk astro 4096 May 17 10:42 zlib-1.2.5
```

group
permissions
r=Read
w=Write
x=eXecute



Your data is protected by file permissions



```
xterm
-rw----- 1 holleyjk astro 28672 Oct 14 2009 sesaps_09_program.xls
drwxr-xr-x 3 holleyjk astro 4096 May 17 10:42 share
-rw-r--r-- 1 holleyjk astro 14176 Jul 6 16:42 sig.khb.jpeg
-rw-r--r-- 1 holleyjk astro 13717 Jul 6 16:46 sig.khb.pdf
-rw----- 1 holleyjk astro 21470 Jun 10 14:56 sinha_69.pdf
-rw----- 1 holleyjk astro 230880 May 28 15:08 SMD-10-1678_Holley-Bokelmann
.pdf
drwx----- 4 holleyjk astro 4096 May 13 2009 smooth.2.01
-rw-ruxrwx 1 holleyjk astro 407473 Jan 25 2010 starcounts.dat.gz
-rw----- 1 holleyjk astro 407478 Jan 23 2010 starcounts.orig.dat.gz
-rw----- 1 holleyjk astro 79428 Jun 15 13:30 stassun_career_report_2009.p
df
-rw-r--r-- 1 holleyjk astro 8 Jun 1 14:16 TAR.proposal.aux
-rw-r--r-- 1 holleyjk astro 7436 Jun 1 14:16 TAR.proposal.dvi
-rw-r--r-- 1 holleyjk astro 3674 Jun 1 14:16 TAR.proposal.log
-rw-r--r-- 1 holleyjk astro 26545 Jun 1 14:16 TAR.proposal.pdf
-rw-r--r-- 1 holleyjk astro 72587 Jun 1 14:16 TAR.proposal.ps
-rw----- 1 holleyjk astro 6086 Jun 1 14:16 TAR.proposal.tex
drwx----- 6 holleyjk astro 4096 Sep 21 2007 tibs2107
drwx----- 5 holleyjk astro 4096 Feb 2 2009 tipsy-2.2.3c
drwx----- 2 holleyjk astro 4096 Mar 27 2008 TRS
drwx----- 3 holleyjk astro 4096 Mar 5 2008 web
-rw----- 1 holleyjk astro 33280 Jul 21 18:23 YaqiongSummary.doc
drwxr-xr-x 15 holleyjk astro 4096 May 17 16:48 yt
-rw----- 1 holleyjk astro 798679 Jun 14 11:03 Zampieri,2009,MNRAS,400,677.
pdf
drwxr-xr-x 12 holleyjk astro 4096 May 17 10:42 zlib-1.2.5
```

other
permissions
r=Read
w=Write
x=eXecute

To change permissions:

```
chmod ugo+/-rwx filename
```

```
Example: chmod go+r pro*.dat
```

```
Advanced: chgrp -R <group> <directory>
```

changes the group ownership of a directory (R=recursively).

Have to be admin or be member of both groups

Taking a peek at a file:

- `head -n50 file` → displays first 50 lines of file
- `tail -n12 file` → displays last 12 lines of file
- `more file` → scrolls through file page by page
Enter to scroll line by line or Spacebar to scroll "page by page"
- `less file` → a better version of more

*Practice: figure out what is the last line in
~/alias.holleyjk*

Is the file compressed?

`gunzip file`

Looking for a file? Try

`find . -iname file`

What jobs are running in your shell?
...on your machine?

`ps`

`htop`

Need to kill a job?

`kill -9 PID`

Need to count lines in a file?

`wc -l file`

Smooosh files into 1

`cat file1 file2 > smoooshfile`

Split file into many of N bytes

`split -b N file`

String commands together with pipes :

try these:

```
ls -l /home/holleyjk |less
```

```
ps aux |grep username
```

the possibilities are huge!

Q. List the first 5 most recently modified files in a directory in human readable format

Steps:

1. list all the files in the directory
2. list them by modification time, most recent first
3. display that list in human readable format
4. display only the top 5 in that list

Q. List the first 5 most recently modified files in a directory in human readable format

Steps:

1. list all the files in the directory `ls -al`
2. list them by modification time, most recent first
3. display that list in human readable format
4. display only the top 5 in that list

Q. List the first 5 most recently modified files in a directory in human readable format

Steps:

1. list all the files in the directory `ls -al`
2. list them by modification time, most recent first `ls -alt`
3. display that list in human readable format
4. display only the top 5 in that list

Q. List the first 5 most recently modified files in a directory in human readable format

Steps:

1. list all the files in the directory `ls -al`
2. list them by modification time, most recent first `ls -alt`
3. display that list in human readable format `ls -halt`
4. display only the top 5 in that list

Q. List the first 5 most recently modified files in a directory in human readable format

Steps:

1. list all the files in the directory `ls -al`
2. list them by modification time, most recent first `ls -alt`
3. display that list in human readable format `ls -halt`
4. display only the top 5 in that list `ls -halt |head -n5`

Q. List the first 5 most recently modified files in a directory in human readable format

Steps:

1. list all the files in the directory `ls -al`
2. list them by modification time, most recent first `ls -alt`
3. display that list in human readable format `ls -halt`
4. display only the top 5 in that list `ls -halt |head -n6 |tail -n5`

Q. List the first 10 most space-hogging files in a directory in human readable format

Q. List the first 10 most space-hogging files in a directory in human readable format

```
ls -halS |head -n11 |tail -n10
```

`last` - tells you the people who logged in, arranged chronologically

`w` - shows you who are logged in currently

`date` - gives you the current date (time-stamp)

`du` - shows you the disk usage of given directory (and sub-dirs)

`sort` - sorts data file/piped output based on specified column

`diff` - finds differences between two files

`>` - redirects result to a file

`>>` - appends data to a file

`history` - returns history of commands executed in terminal

`ping` - attempts to open a line of communication with a network host

`wget` - download a file from the Internet

Exercises:

1. Who are the last 5 people that logged in to vpac38 today ?
2. How many times did 'hoffmare' login to vpac01 ?
3. Copy wang.merritt.dat from the 2015 bootcamp directory to your laptop and sort on the black hole mass. Dump results to a file.
4. How many people are on vpac02 right now ?
5. How many bytes are you using on your home directory ?
6. How many files does Kelly have in her \$HOME/bootcamp dir ?

Exercises:

1. Who are the last 5 people that logged in to vpac38 today ?

first, ssh into vpac38 then... `last |head -n5`

2. How many times did 'hoffmare' login to vpac01 ?

3. Copy wang.merritt.dat from the 2015 bootcamp directory to your laptop and sort on the black hole mass. Dump results to a file.

4. How many people are on vpac02 right now ?

5. How many bytes are you using on your home directory ?

6. How many files does Kelly have in her \$HOME/bootcamp dir ?

Exercises:

1. Who are the last 5 people that logged in to vpac38 today ?
2. How many times did 'hoffmare' login to vpac01 ?
first, ssh into vpac01 , then...
`last |grep hoffmare > dummy ;wc -l dummy`
3. Copy wang.merritt.dat from the 2015 bootcamp directory to your laptop and sort on the black hole mass. Dump results to a file.
4. How many people are on vpac02 right now ?
5. How many bytes are you using on your home directory ?
6. How many files does Kelly have in her \$HOME/bootcamp dir ?

Exercises:

1. Who are the last 5 people that logged in to vpac38 today ?
2. How many times did 'hoffmare' login to vpac01 ?
3. Copy wang.merritt.dat from the 2015 bootcamp directory to your laptop and sort on the black hole mass. Dump results to a file.

```
sort -n -k2,2 < wang.merritt.dat > sorted.wang.merritt.dat
```
4. How many people are on vpac02 right now ?
5. How many bytes are you using on your home directory ?
6. How many files does Kelly have in her \$HOME/bootcamp dir ?

Exercises:

1. Who are the last 5 people that logged in to vpac38 today ?
2. How many times did 'hoffmare' login to vpac01 ?
3. Copy wang.merritt.dat from the 2015 bootcamp directory to your laptop and sort on the black hole mass. Dump results to a file.
4. How many people are on vpac02 right now ?
first, ssh into vpac02 then...
`w |sort`
5. How many bytes are you using on your home directory ?
6. How many files does Kelly have in her \$HOME/bootcamp dir ?

Exercises:

1. Who are the last 5 people that logged in to vpac38 today ?
2. How many times did 'hoffmare' login to vpac01 ?
3. Copy wang.merritt.dat from the 2015 bootcamp directory to your laptop and sort on the black hole mass. Dump results to a file.
4. How many people are on vpac02 right now ?
5. How many bytes are you using on your home directory ?

du -h

6. How many files does Kelly have in her \$HOME/bootcamp dir ?

Exercises:

1. Who are the last 5 people that logged in to vpac38 today ?
2. How many times did 'hoffmare' login to vpac01 ?
3. Copy wang.merritt.dat from the 2015 bootcamp directory to your laptop and sort on the black hole mass. Dump results to a file.
4. How many people are on vpac02 right now ?
5. How many bytes are you using on your home directory ?
6. How many files does Kelly have in her \$HOME/bootcamp dir ?

```
ls -a | wc -l
```

Exercises:

1. Who are the last 5 people that logged in to vpac38 today ?

first, ssh into vpac38 then...

```
last |head -n6 |tail -n5
```

2. How many times did 'hoffmare' login to vpac01 ?

first, ssh into vpac01 , then...

```
last |grep hoffmare > dummy ;wc -l dummy
```

3. Copy wang.merritt.dat from the 2015 bootcamp directory to your laptop and sort on the black hole mass. Dump results to a file.

```
sort -n -k2,2 < wang.merritt.dat > sorted.wang.merritt.dat
```

4. How many people are on vpac02 right now ?

first, ssh into vpac02 then...

```
w |sort
```

5. How many bytes are you using on your home directory ?

```
du -h
```

6. How many files does Kelly have in her \$HOME/bootcamp dir ?

```
ls -a |wc -l
```

Setting your shell (how you interface with your operating system)

do a less on
cshrc.holleyjk

execute an alias file that
nicknames unix commands

tells your system where to look
for external commands, like
graphics programs

```
xterm
# .cshrc

set listjobs
limit coredumpsize 0
umask 022
unset autologout
if ( -e ~/.alias )    source ~/.alias
if( $?REMOTEHOST && ! $?DISPLAY ) then
    setenv DISPLAY ${REMOTEHOST}:0
endif

#set path=(/usr/local/products/mpich2/bin $path)

# pattans 20080123
set osv = `uname -a | awk '{print $1 substr ($3,1,1)}'`
set arch = `uname -m`
setenv MPICH_HOME /usr/local/mpich2/${osv}/1.0.6p1/${arch}
setenv MPI_HOME /usr/local/mpich2/${osv}/1.0.6p1/${arch}

#setenv MPICH_HOME /usr/local/products/mpich2
#setenv MPI_HOME /usr/local/products/mpich2

setenv PATH /usr/local/bin:/usr/bin:/home/holleyjk/bin:/usr/local/python2.6/x86_64/
bin:/usr/local/jdk:/usr/local/lib/jdk/jdk1.6.0_01/bin:/usr/local/mpich2/Linux2/1.0.
6p1/x86_64/bin:/usr/lib64/qt-3.3/bin:/usr/kerberos/bin:/bin:/usr/X11R6/bin
setenv CLASSPATH /home/holleyjk/lstore/accre-loci-1.0-1802.jar:/home/holleyjk/lstor
e/aspectjrt.jar:/home/holleyjk/lstore/jargs.jar:/home/holleyjk/lstore/log4j-1.2.12.
jar
setenv LD_LIBRARY_PATH /usr/lib:${HOME}/lib:/usr/local/lib:/usr/local/python2.6/x86
_64/lib:/usr/local/jdk/lib:/usr/local/lib/jdk/jdk1.6.0_01/lib:/usr/local/mpich2/Lin
ux2/1.0.6p1/x86_64/lib:/usr/include/kde/arts/gsl:/usr/lib64/qt-3.3/lib:/usr/kerbero
s/lib:/lib64:/usr/X11R6/lib:/lib

setenv IDL_DIR /usr/local/itt/idl
--More--(91%)
```

Setting your shell (how you interface with your operating system)

Open a web browser to github.com/djsissom

Go to bootcamp repository

Click "Clone or Download" button, and choose "Download Zip"

Scp the zip file to your vpac home directory

Unzip the file on your laptop (Mac) and on VPAC

Setting your shell (how you interface with your operating system)

What does ``echo $SHELL`` return?

`Mv bashrc to ~/.bashrc` (if it doesn't exist yet...)

`Less .bashrc` to read it

`source .bashrc`
will run the file and install your commands

To change your `.bashrc` ,you need a text editor

`emacs <filename> &`

- Ctl-v -- page up
 - Esc-v -- page down
 - Ctl-n -- next line
 - Ctl-p -- previous line
 - Esc-< -- top of file
 - Esc -> -- bottom of file
 - Esc-Esc-Esc -- get out of hotkey
 - Ctl-X-S -- save file
 - Ctl-X-C -- quit
 - Ctl-k -- kill a line
 - Ctl-y -- paste line
 - Esc -% -- query replace
 - Ctl-x (-- start macro
 - Ctl-x) -- end macro
- Example: Ctl-x(Ctl-n Ctl-n Ctl-x
- Ctl-x e -- execute macro
 - Esc-num -- do num times
 - Ctl-/ -- undo

Practice: Open a new file called 'motivation' and write 'I love emacs with my whole heart!' 2^{16} times. Then, replace 'heart' with 'mind' on every other line.

Editing text files with Vim

Set up vim files (dark terminal recommended)

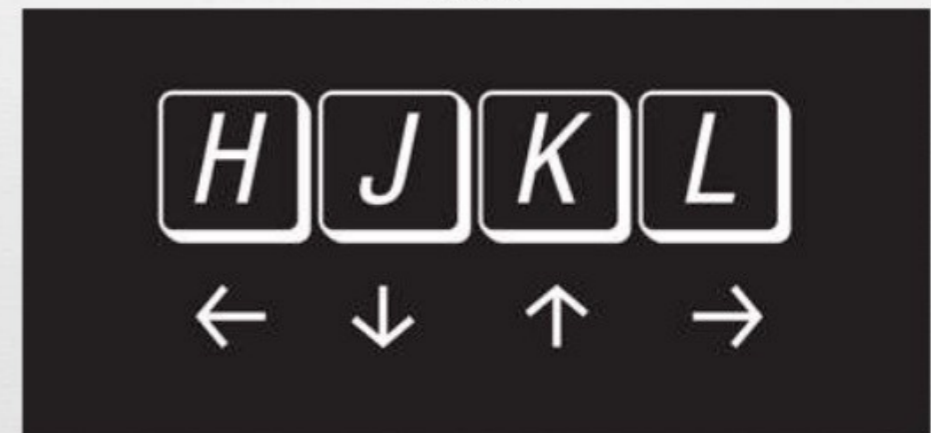
```
$ vim .bashrc
```

More slides at <https://www.slideshare.net/brandonliu/introduction-to-vim>

[brandonliu/introduction-to-vim](https://www.slideshare.net/brandonliu/introduction-to-vim) →

/- to search and n/N to
navigate search results

Surviving in Vim



- ⌘ i – Gets you into insertion mode
- ⌘ <ESC> – Always gets you back to normal mode
- ⌘ :wq<Enter> – Saves and quits Vim

Vim walkthrough

<https://www.openvim.com/>

Remember the cheat sheet

<http://cheatsheetworld.com/programming/unix-linux-cheat-sheet/>

For homework:

Add your own nicknames for unix commands -- most useful aliases win a prize!

Add aliases to make rm, mv, and cp safer (check the man pages for the interactive option for each)

Practice with your text editor of choice

Unix Challenge!

-- Find a file with cereal in its name

(hint: it's in a vpac directory
involving /home/holleyjk...)

-- copy it to your home directory, add your name to the filename, and change permissions so that you can read and write to it, but group/other can only read it

-- write the name of your favorite cereal in the file

-- copy it to Kelly's bootcamp/2017 directory and tell me when you've done it. First one wins a prize!!