Housekeeping

- Course Resources
  http://bit.ly/DHSI2017-coding

- Washrooms
  Outside and right ?

- Pedagogical style
  “The hard way”, hands-on, playful, and collaborative

- Outside Resources
  Visual QuickStart to Unix & Linux, Software Carpentry, and *Think Python*

- Sticky Notes
  Help and feedback

http://bit.ly/DHSI2017-coding

Day #1
Getting Comfortable with the *nix Command Line

John Simpson & Alicia Cappello DHSI-2017

Fundamentals of Programming/Coding for Human(s|ists)

What’s Important Today

1. Quick history of *nix
2. Learn basic terminal techniques
    1. Navigation
    2. Creation and Destruction
    3. Plumbing and Searching
3. Practice using those techniques

Four things to keep in mind…

1. Silence is golden (or frustrating)
2. Capitalization matters
3. Spaces matter
4. Everything is a file

The Cathedral

The Bazaar

“Not only is UNIX dead,
it’s starting to smell really bad.”

–Rob Pike circa 1991
(widely attributed but unsourced)

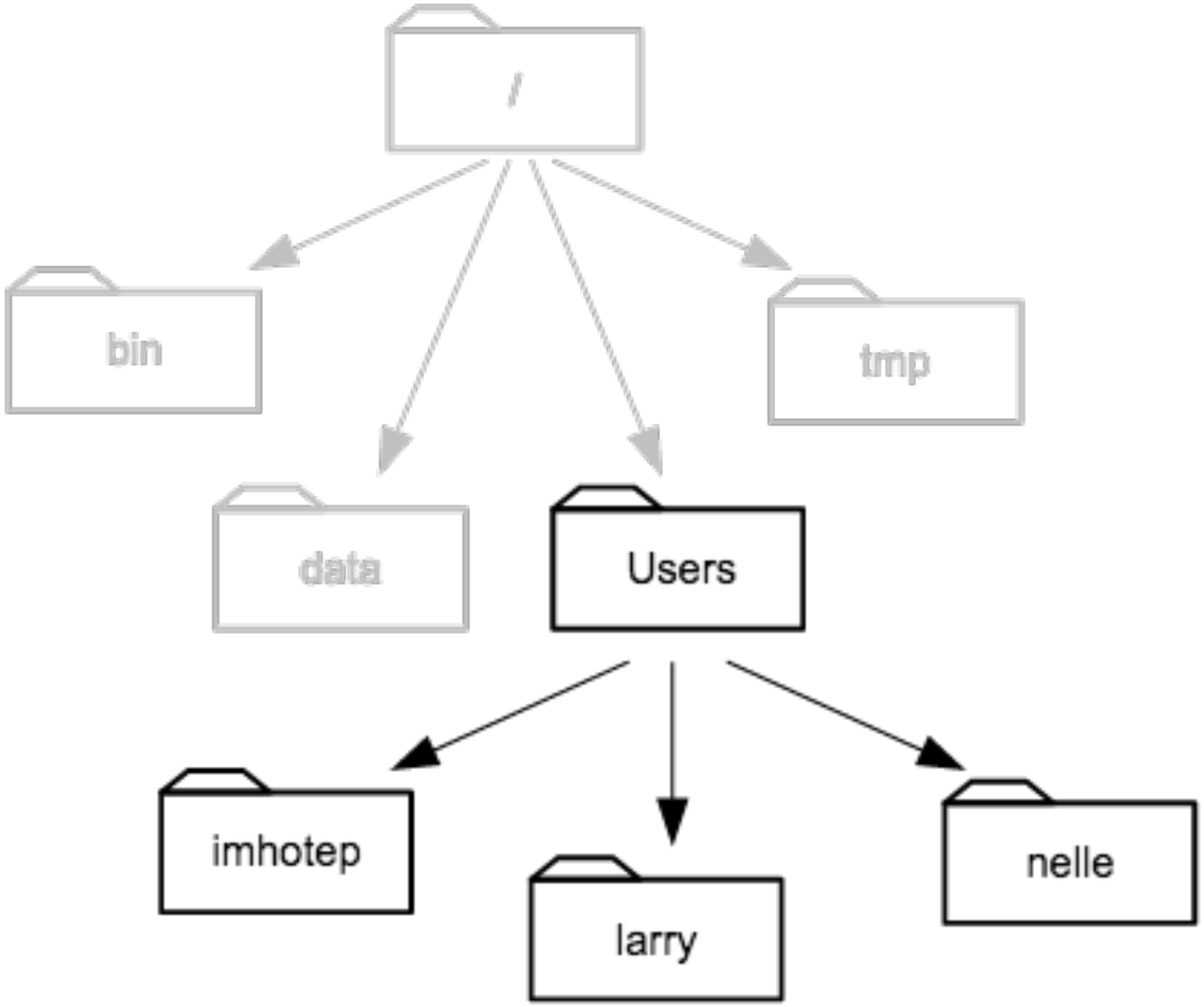
Some of the major flavours of GNU/Linux available today.
It is unfortunate for the history of the free software movement that “GNU” is so often dropped from the name (possibly because it is too hard to say--why is there no free marketing movement to prevent such things from happening?).
Want to see *all* the flavours in distribution? Look **HERE**.

What do the flags -S, -h, and -r do when combined with the `ls` command?

1. Nothing
2. Display the help file and nothing else
3. [✓ Display directory content by file size in reverse order with human readable sizes]
4. Flags cannot be combined because of the “one command, one action” policy

From `/home/amanda/data/` which commands could Amanda could use to navigate to her home directory, `/home/amanda/`

```
cd .
cd /
cd /home/amanda
cd ../..
cd ~
cd home
cd ~/data/..
cd
cd ..
```



```
cd Users/larry

cd larry

cd /larry

cd ../larry

cd ../Users/larry
```

If you were asked to describe the structure commands on the command line what would you say?

http://bit.ly/MDxHM

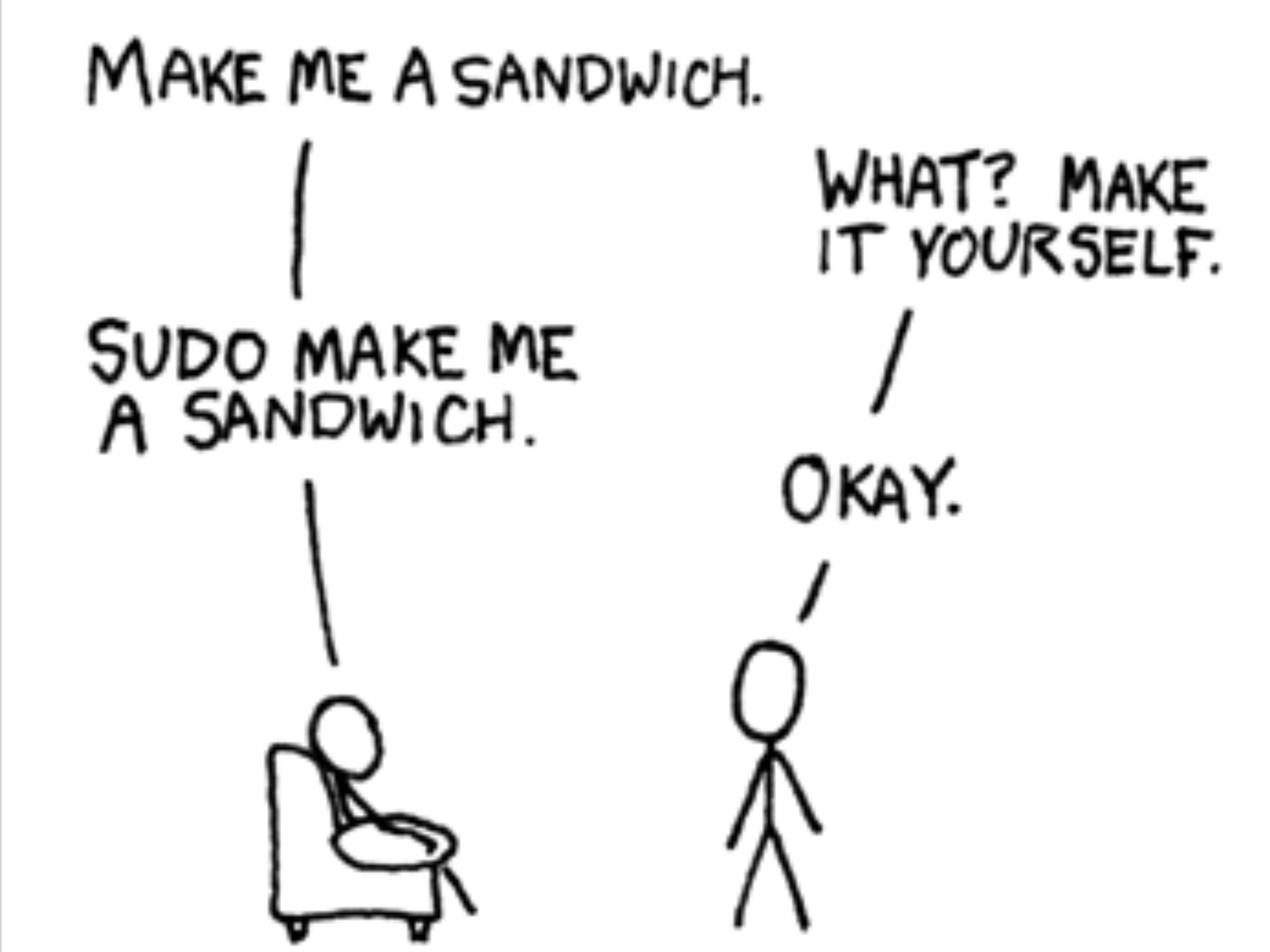
W A R N I N G: DO NOT RUN THIS CODE

What do you ***think*** that running

```
rm -rf /
```

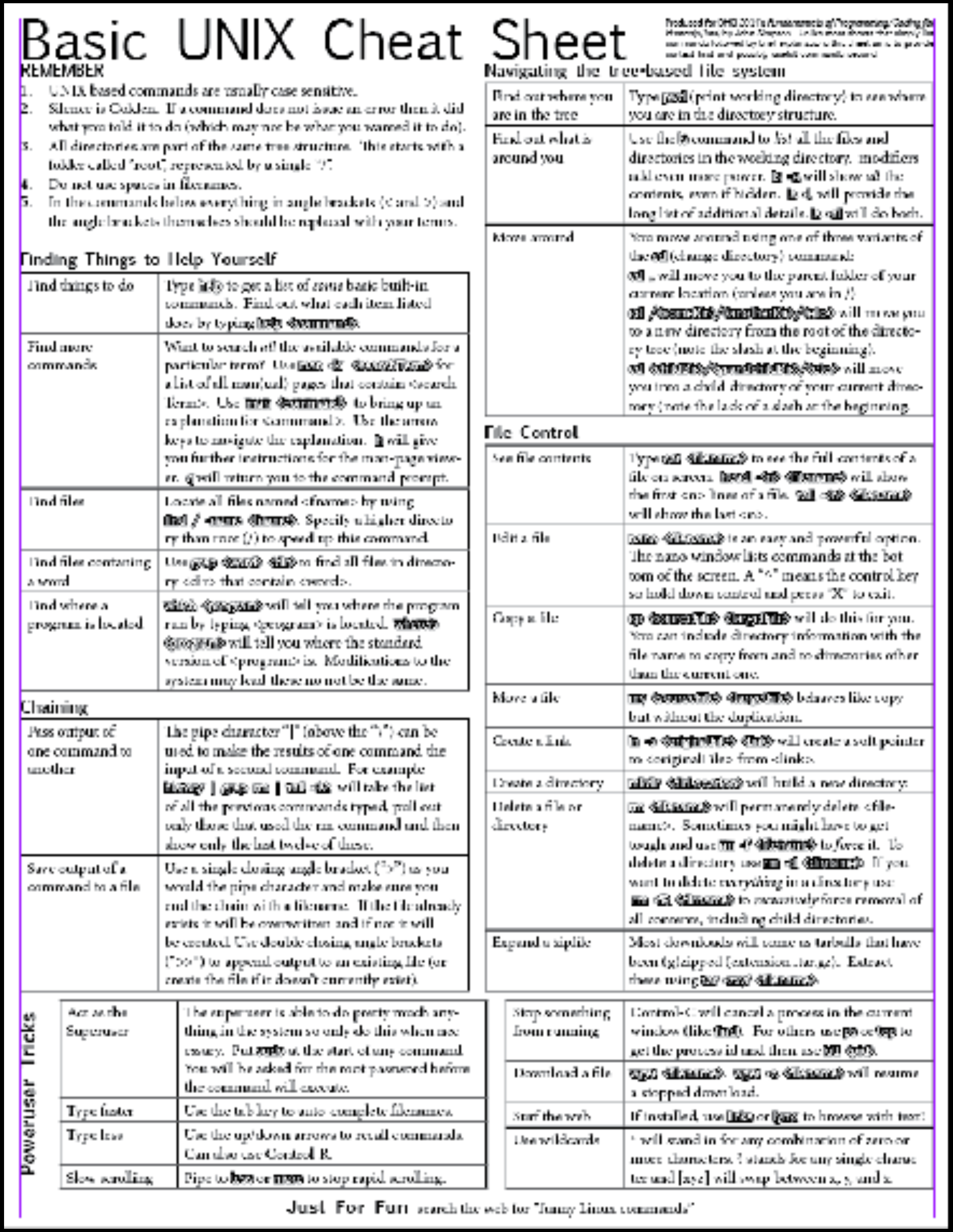
would do?

W A R N I N G: DO NOT RUN THIS CODE


MAKE ME A SANDWICH.
WHAT? MAKE IT YOURSELF.
SUDO MAKE ME A SANDWICH.
OKAY.

Spend a few minutes looking over this cheatsheet to get an idea of what can be done.

You can use it as a reference for what we'll do from here.



Basic UNIX Cheat Sheet

[illegible]

REMEMBER

1. UNIX based commands are usually case sensitive.
2. Silence is Golden. If a command does not issue an error then it did what you told it to do (which may not be what you wanted it to do).
3. All directories are part of the same tree structure. This starts with a folder called "root", represented by a single "/".
4. Do not use spaces in filenames.
5. In the commands below everything in angle brackets (< and >) and the angle brackets themselves should be replaced with your terms.

Finding Things to Help Yourself

Find things to do	Type help to get a list of some basic built-in commands. Find out what each item listed does by typing help <command>.
Find more commands	Want to search all the available commands for a particular term? Use man -k <searchterm> for a list of all man(ual) pages that contain <search Term>. Use man <command> to bring up an explanation for <command>. Use the arrow keys to navigate the explanation. h will give you further instructions for the man-page viewer. q will return you to the command prompt.
Find files	Locate all files named <fname> by using find / -name <fname>. Specify a higher directory than root (/) to speed up this command.
Find files containing a word	Use grep <word> <dir> to find all files in directory <dir> that contain <word>.
Find where a program is located	which <program> will tell you where the program run by typing <program> is located. whereis <program> will tell you where the standard version of <program> is. Modifications to the system may lead these to not be the same.

Chaining

Pass output of one command to another	The pipe character "|" (above the "\") can be used to make the results of one command the input of a second command. For example history | grep rm | tail -12 will take the list of all the previous commands typed, pull out only those that used the rm command and then show only the last twelve of those.
Save output of a command to a file	Use a single closing angle bracket (">") as you would the pipe character and make sure you end the chain with a filename. If the file already exists it will be overwritten and if not it will be created. Use double closing angle brackets (">>") to append output to an existing file (or create the file if it doesn't currently exist).

Navigating the tree-based file system

Find out where you are in the tree	Type pwd (print working directory) to see where you are in the directory structure.
Find out what is around you	Use the ls command to list all the files and directories in the working directory. modifiers add even more power. ls -a will show all the contents, even if hidden. ls -l will provide the long list of additional details. ls -al will do both.
Move around	You move around using one of three variants of the cd (change directory) command: cd .. will move you to the parent folder of your current location (unless you are in /). cd /someDir/anotherDir/child will move you to a new directory from the root of the directory tree (note the slash at the beginning). cd childDir/grandchildDir/child will move you into a child directory of your current directory (note the lack of a slash at the beginning.

File Control

See file contents	Type cat <filename> to see the full contents of a file on screen. head -n <filename> will show the first <n> lines of a file. tail -n <filename> will show the last <n>.
Edit a file	nano <filename> is an easy and powerful option. The nano window lists commands at the bottom of the screen. A "^" means the control key so hold down control and press "X" to exit.
Copy a file	cp <sourceFile> <targetFile> will do this for you. You can include directory information with the file name to copy from and to directories other than the current one.
Move a file	mv <sourceFile> <targetFile> behaves like copy but without the duplication.
Create a link	ln -s <originalFile> <link> will create a soft pointer to <original> files from <link>.
Create a directory	mkdir <dirname> will build a new directory.
Delete a file or directory	rm <filename> will permanently delete <filename>. Sometimes you might have to get tough and use rm -f <filename> to force it. To delete a directory use rm -r <dirname>. If you want to delete everything in a directory use rm -rf <dirname> to recursively force removal of all contents, including child directories.
Expand a zipfile	Most downloads will come as tarballs that have been (g)zipped (extension .tar.gz). Extract these using tar -xzvf <filename>.

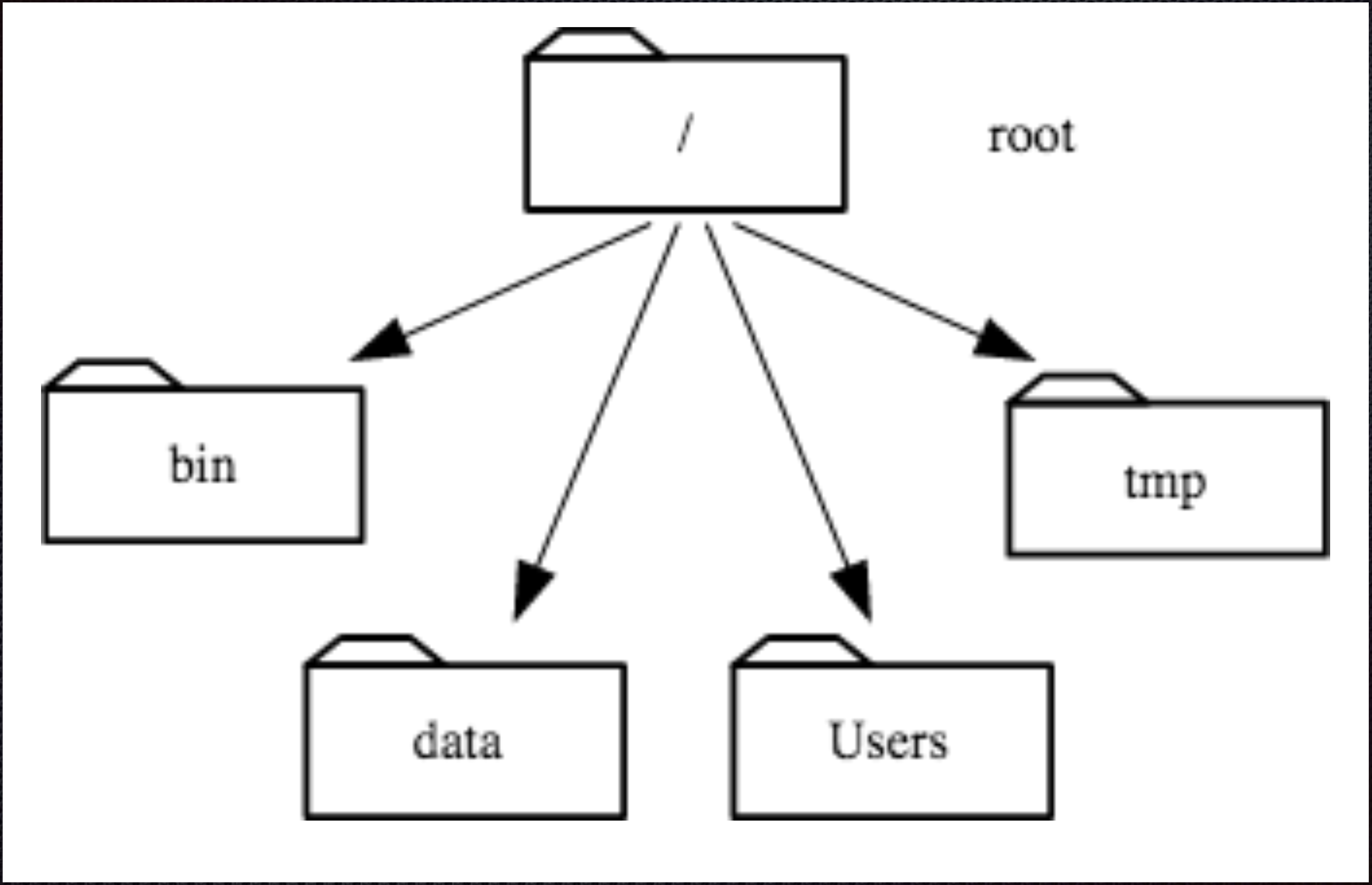
Poweruser Tricks

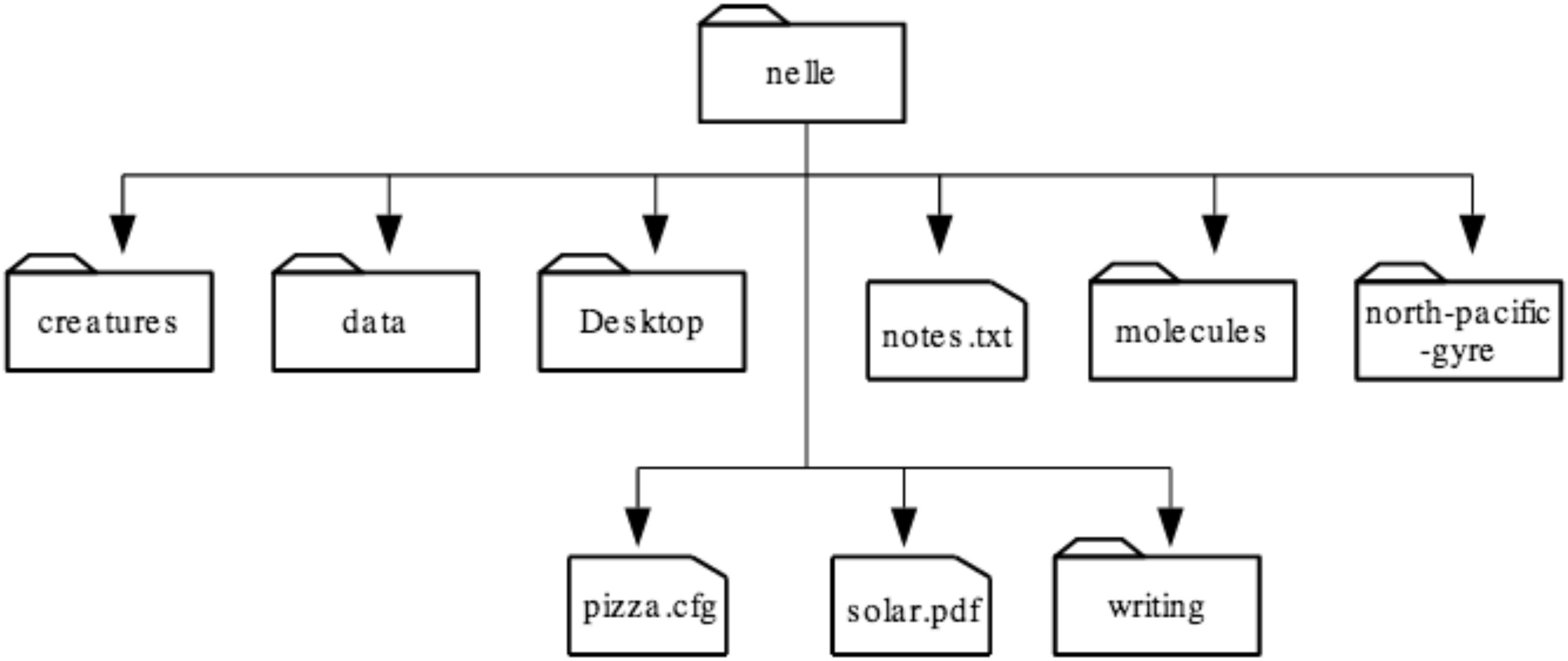
Act as the Superuser	The superuser is able to do pretty much anything in the system so only do this when necessary. Put sudo at the start of any command. You will be asked for the root password before the command will execute.
Type faster	Use the tab key to auto-complete filenames.
Type less	Use the up/down arrows to recall commands. Can also use Control-R.
Slow scrolling	Pipe to less or more to stop rapid scrolling.
Stop something from running	Control-C will cancel a process in the current window (like find). For others use ps or top to get the process id and then use kill <pid>.
Download a file	wget <filename>. wget -c <filename> will resume a stopped download.
Surf the web	If installed, use lynx or links to browse with text!
Use wildcards	* will stand in for any combination of zero or more characters. ? stands for any single character and [xyz] will swap between x, y, and z.

Just For Fun search the web for "funny Linux commands"

Skills to Master: Navigation

- Translate an absolute path into a relative path and vice versa.
- Construct absolute and relative paths that identify specific files and directories and directory content
- Identify the actual command, flags, and filenames in a command-line call.
- Demonstrate the use of tab completion and up arrow review
- Demonstrate an ability to move around the file system


/
root
bin
data
Users
tmp


nelle
creatures
data
Desktop
notes.txt
molecules
north-pacific-gyre
pizza.cfg
solar.pdf
writing

Skills to Master: Navigation

- Explain the similarities and differences between a file and a directory.
- Translate an absolute path into a relative path and vice versa.
- Construct absolute and relative paths that identify specific files and directories and directory content
- Explain the steps in the shell's read-run-print cycle.
- Identify the actual command, flags, and filenames in a command-line call.
- Demonstrate the use of tab completion, and explain its advantages.

Skills to Master: Creation & Destruction

- Create a directory hierarchy that matches a given diagram.
- Create files in that hierarchy using an editor or by copying and renaming existing files.
- Display the contents of a directory using the command line.
- Delete specified files and/or directories.

Skills to Master: Plumbing

- Redirect a command’s output to a file.
- Process a file instead of keyboard input using redirection.
- Construct command pipelines with two or more stages.
- Explain what usually happens if a program or pipeline isn’t given any input to process.
- Explain Unix’s “small pieces, loosely joined” philosophy.

Skills to Master: Searching

- Use grep to select lines from text files that match simple patterns.
- Use find to find files whose names match simple patterns.
- Use the output of one command as the command-line parameters to another command.
- Explain what is meant by “text” and “binary” files, and why many common tools don’t handle the latter well.