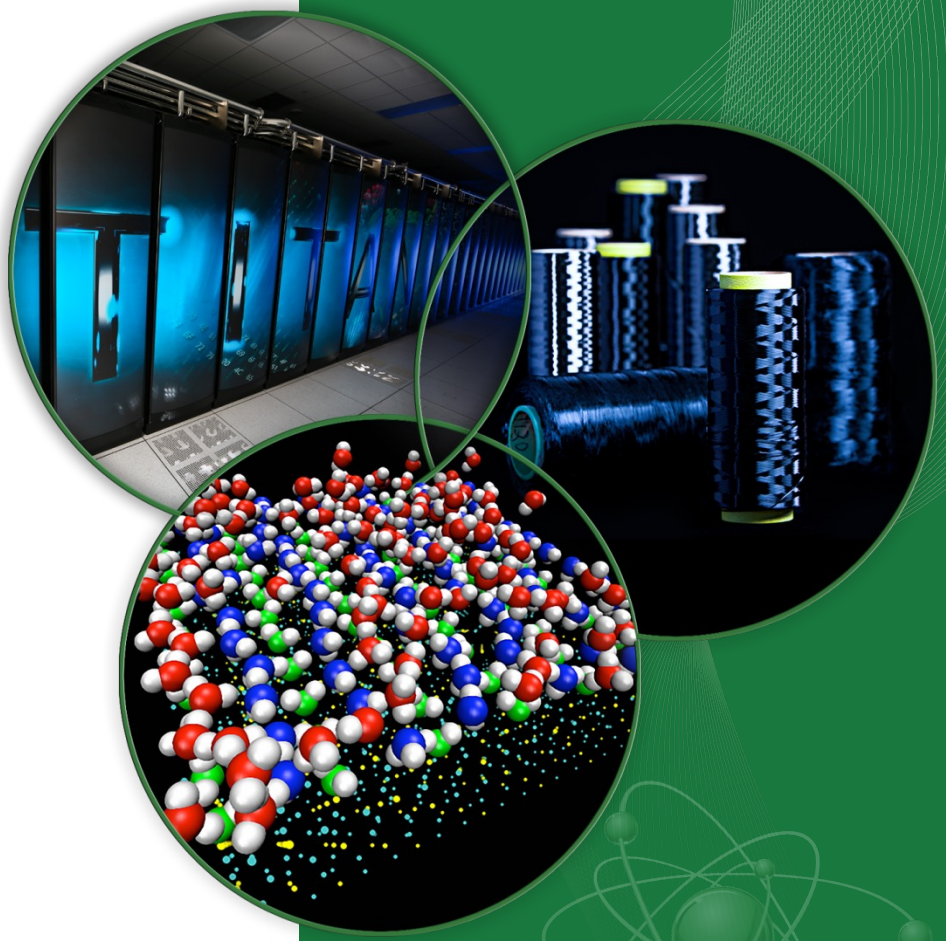
Introduction to Unix/Linux (*nix)

Michael Sandoval, OLCF



ORNL is managed by UT-Battelle
for the US Department of Energy

OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

This Presentation

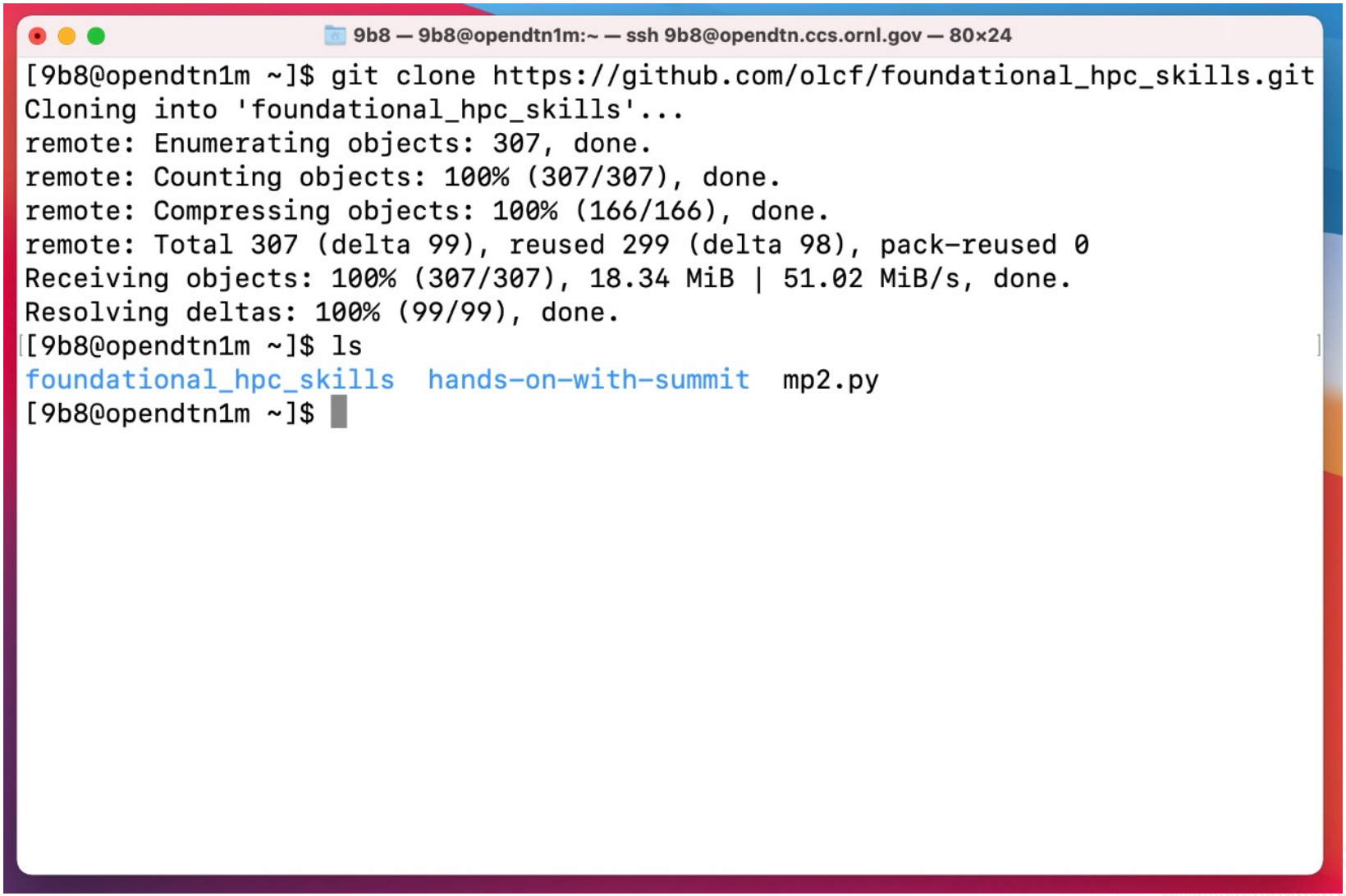
- This presentation will focus on using basic Unix and Linux commands in an HPC environment
  - Assumes you're using a 'remote' system, ie: that you are logged in via terminal, PuTTY, Powershell, etc.
- Cover the basics
  - The terminal window, command line
  - The filesystem structure & how to navigate it
  - Common commands to create, delete, edit, move, copy, etc., directories and files.
- Follow along here:
  - https://github.com/olcf/foundational_hpc_skills/blob/master/intro_to_unix/README.md
- Google is your friend!

First things first

- **Login** to the remote machine using the username and password you used on Day 1. The IP address will be different; check your email!
- **Recall, the syntax is:** username@opendtn#.ccs.ornl.gov
- **Type** the following and hit **enter:**

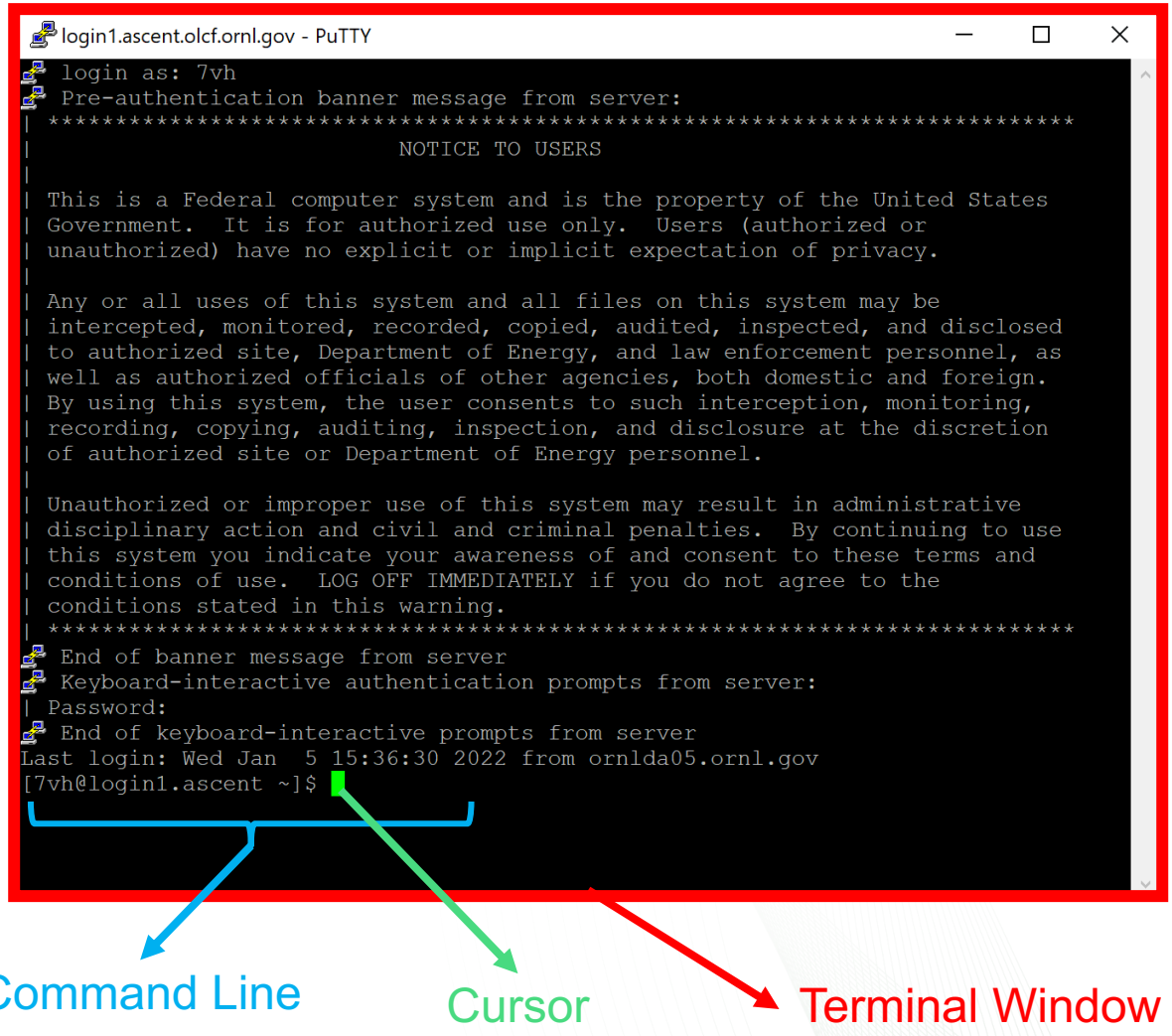
```
git clone https://github.com/olcf/foundational_hpc_skills.git
```

Connecting and Cloning

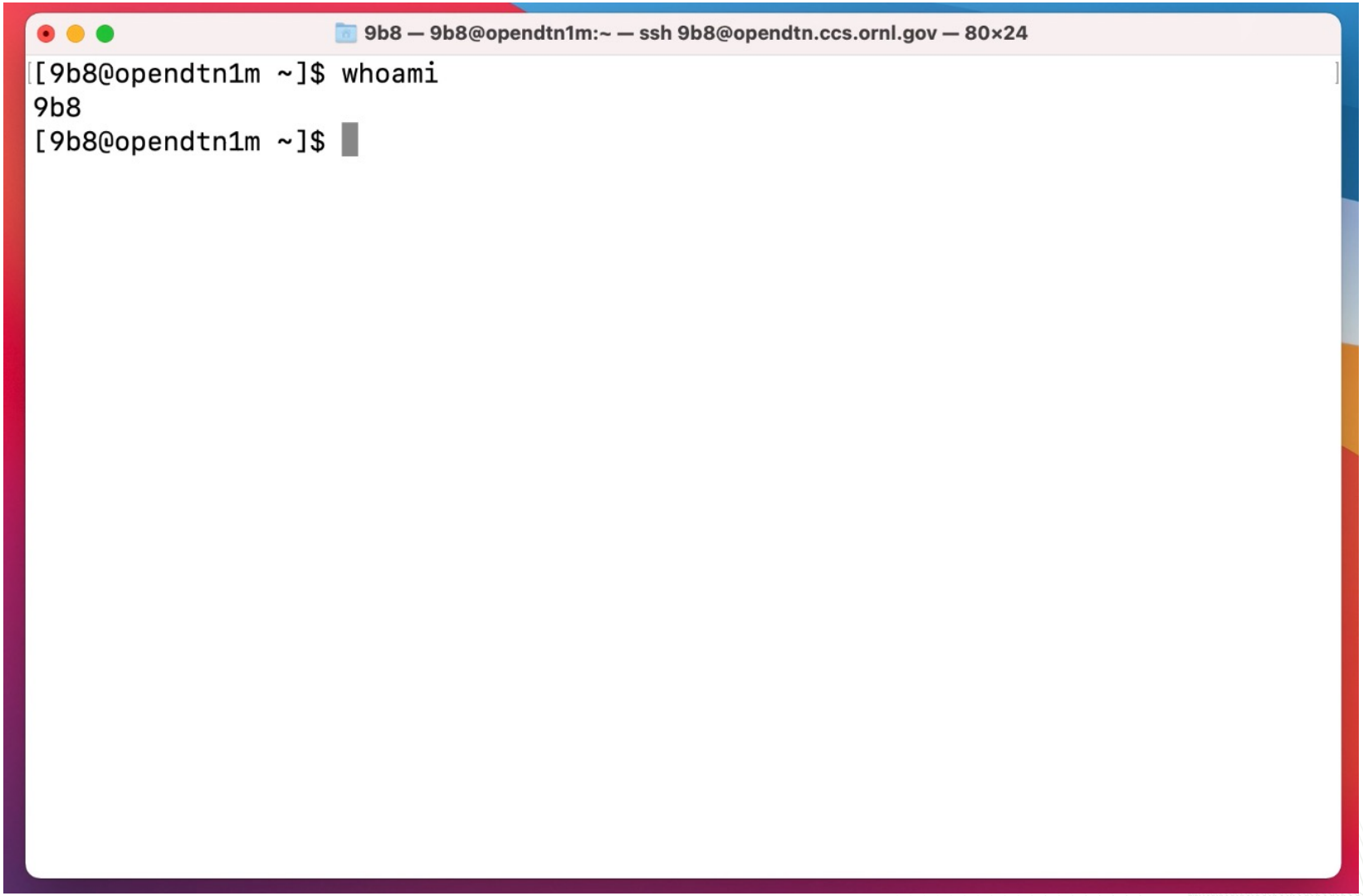


The Command Line

- Commands are typed as text on the command line and executed by pressing “enter”
- Many examples show the “$” symbol from the command line before the actual command; you don’t need to type this symbol
- The command line often contains login and username info.
- Try typing **`$ whoami`** and pressing enter

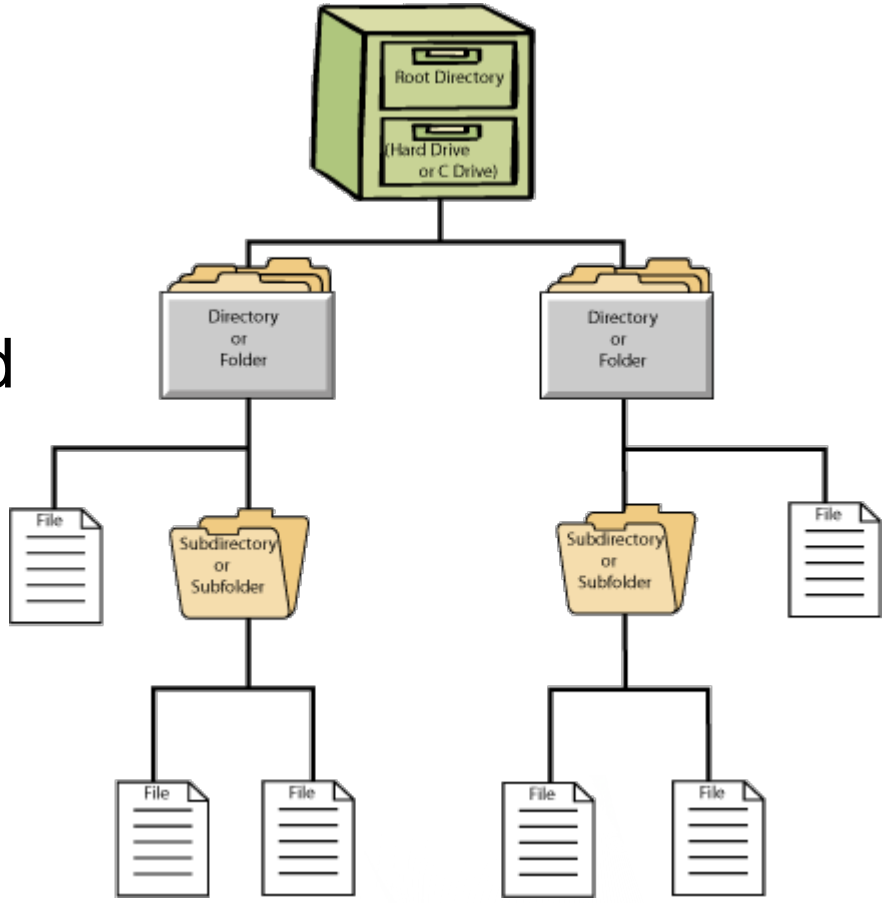


Who am I? (really though)



The Filesystem 1

- Think of the filesystem as a tree. Instead of branches, there are “directories. Instead of leaves, there are “files”
- It starts at /, which is called the “root” directory
- The slash (/) is the directory separator; thus when we move into subdirectories it’s used to separate things (i.e. /home/user1/src)

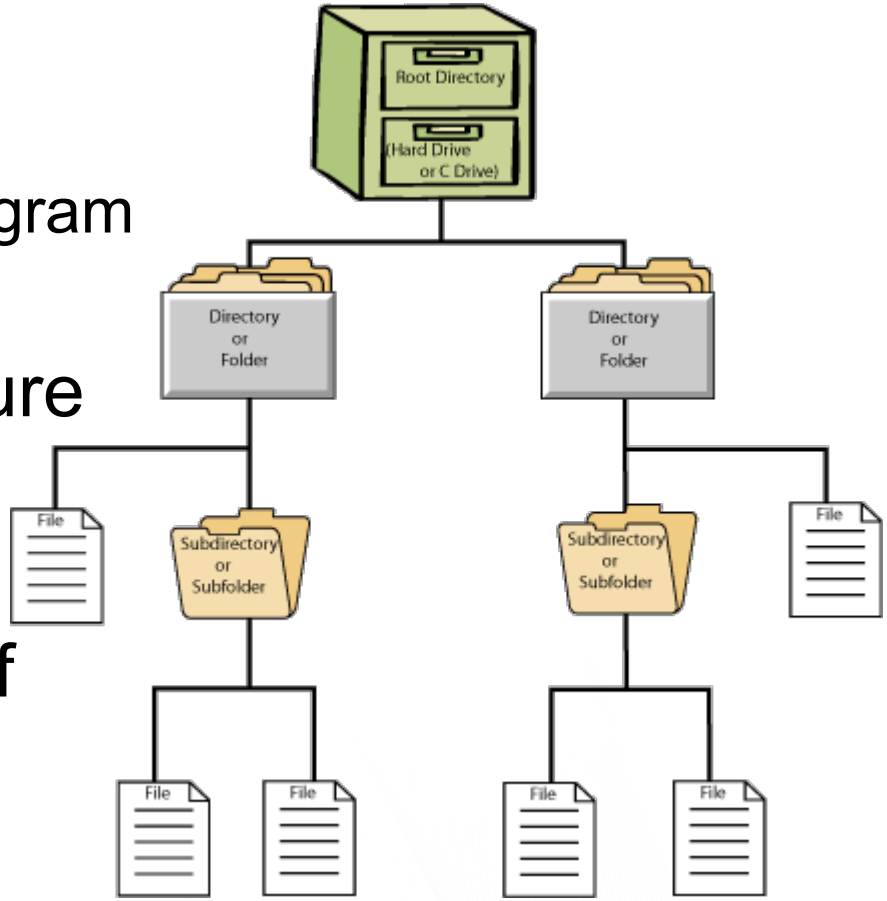


If you are ever “lost”, use the `pwd` command. It prints your location.

```
$ pwd
/ccsopen/home/your_username
```

The Filesystem 2

- **File** – A collection of data
  - Plain text, Input/output, a program (executable), an image, etc.
- **Directory** – A logical structure to help organize files (think "folder")
- **Filesystem** – A collection of files and directories
- As a user, you land in your "home" directory when you log in, and can create directories and files there, or move elsewhere to do that.



Basic Commands

- Commands are usually abbreviations of words (or a series of words)
  - cp for “Copy”, rm for “Remove”, cd for “change directory”
- Commands tend to follow the following syntax:

```
[command] [option flags] [file/directory/object to act upon]
```

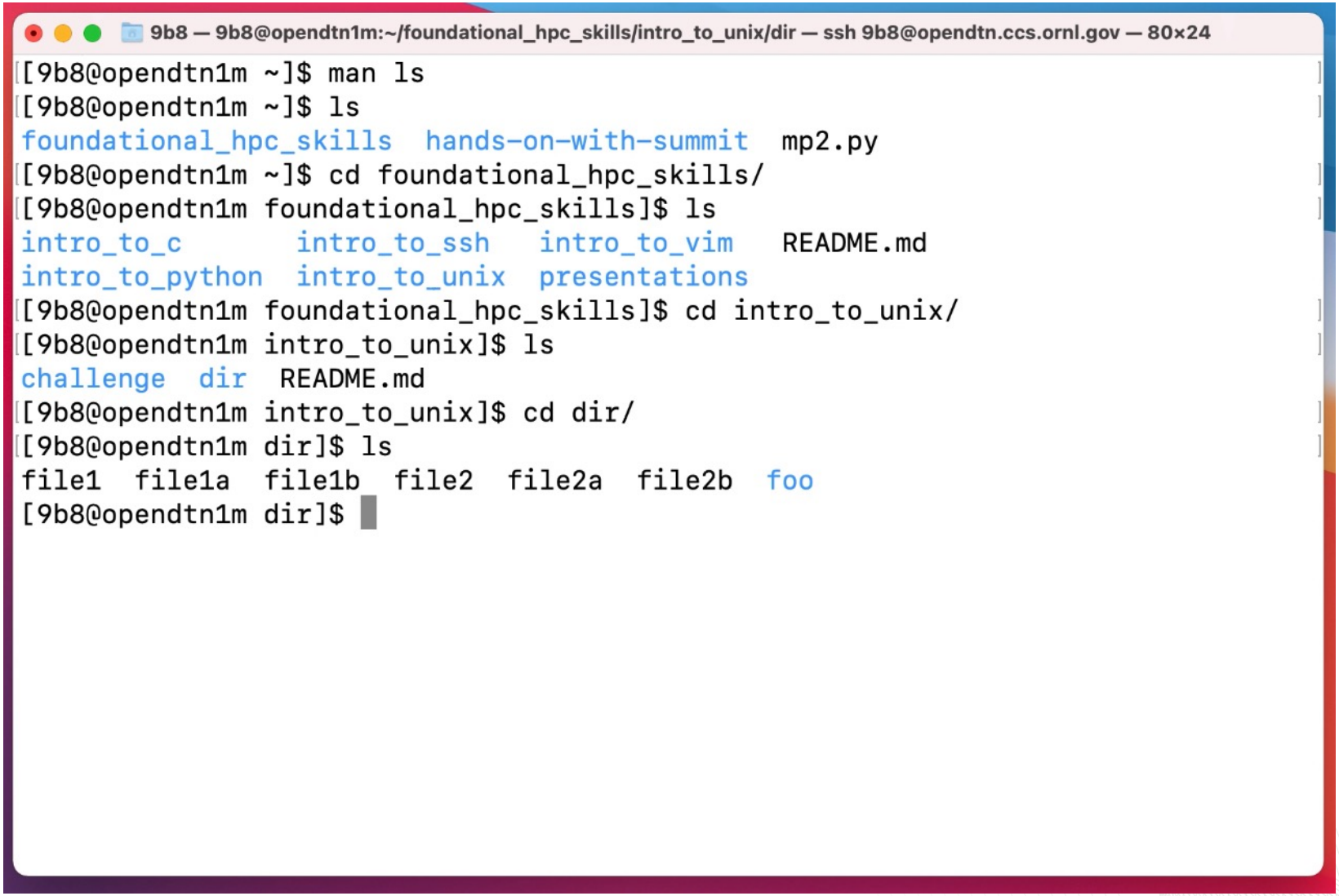
- Almost all commands take various options to control what they do.
- Some help is available via an online manual
  - The `man` command
  - Example: Want info about `ls`? Type the following:

```
man ls
```

Basic File/Directory Commands

Command	Description
`pwd`	**P**rint **w**orking (i.e. current) **d**irectory
`man`	Display the **Man**ual for a command
`whoami`	Display the current user's username
`mkdir`	Create a directory (**M**a**K**e **DIR**ectory)
`rmdir`	Delete a directory (**R**e**M**ove **DIR**ectory)
`cd`	**C**hange (into a) **d**irectory
`ls`	**Ls**t files
`cp`	**Cp**y a file
`mv`	**Mv**e a file (also used to rename a file)
`rm`	**Rm**ove (delete) a file
`cat`	Display the contents (con**cat**enate) a (hopefully text) file
`less`	Display the contents of a text file in a viewing mode
`find`	**Find** files by filename
`grep`	Search for text within files
`diff`	Identify **diff**erences in the contents of files

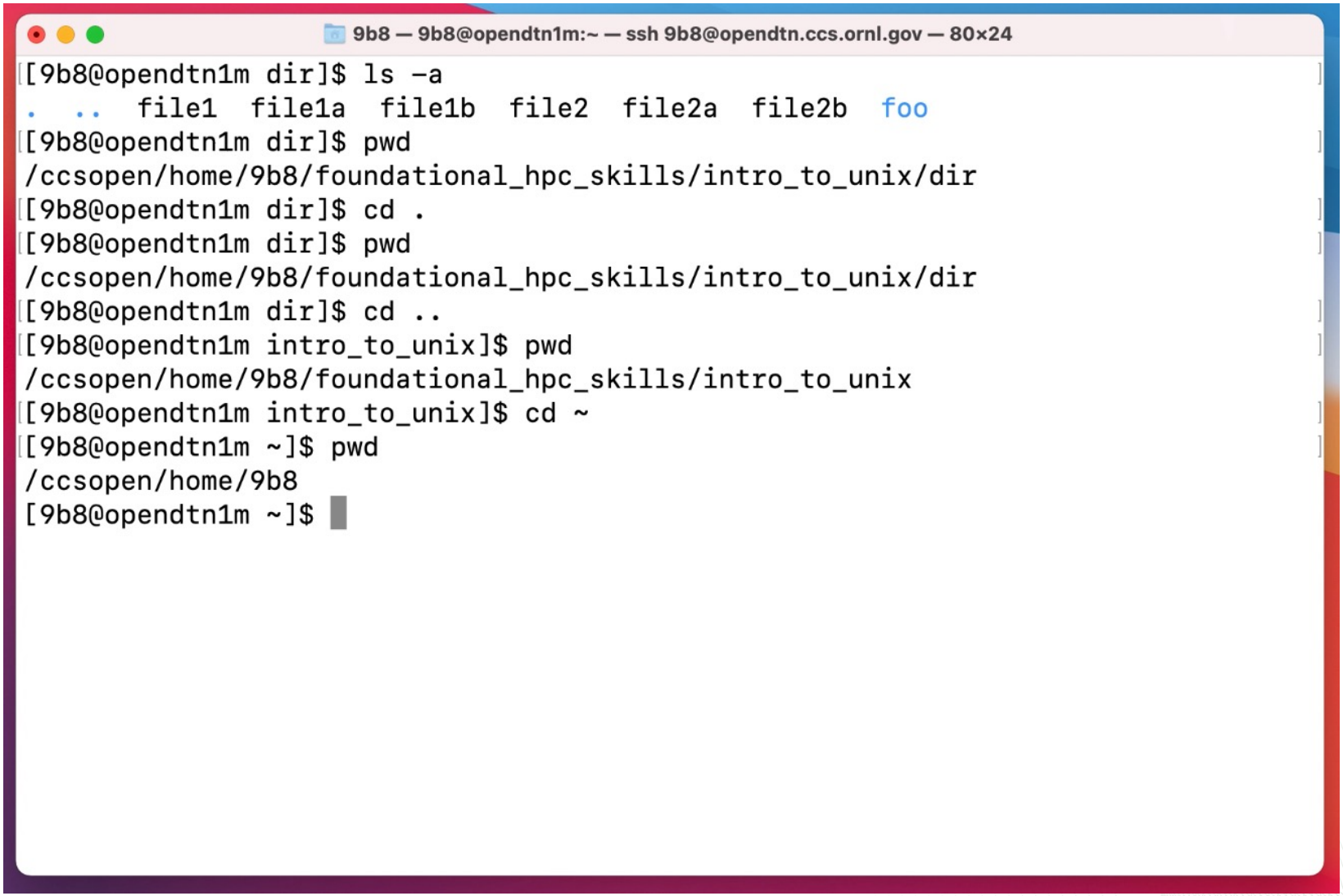
“ls” and “cd” In Action 1



Special Directories

- Every directory contains two special directory entries: ”.” and “..”
- . is a reference to the current directory
- .. is a reference to the parent directory (so we can do things like `cd ..`)
- ~ can be a reference to home directories
  - ~/ is yours
  - ~user1/ is user1’s
- You’ll see how these are useful later

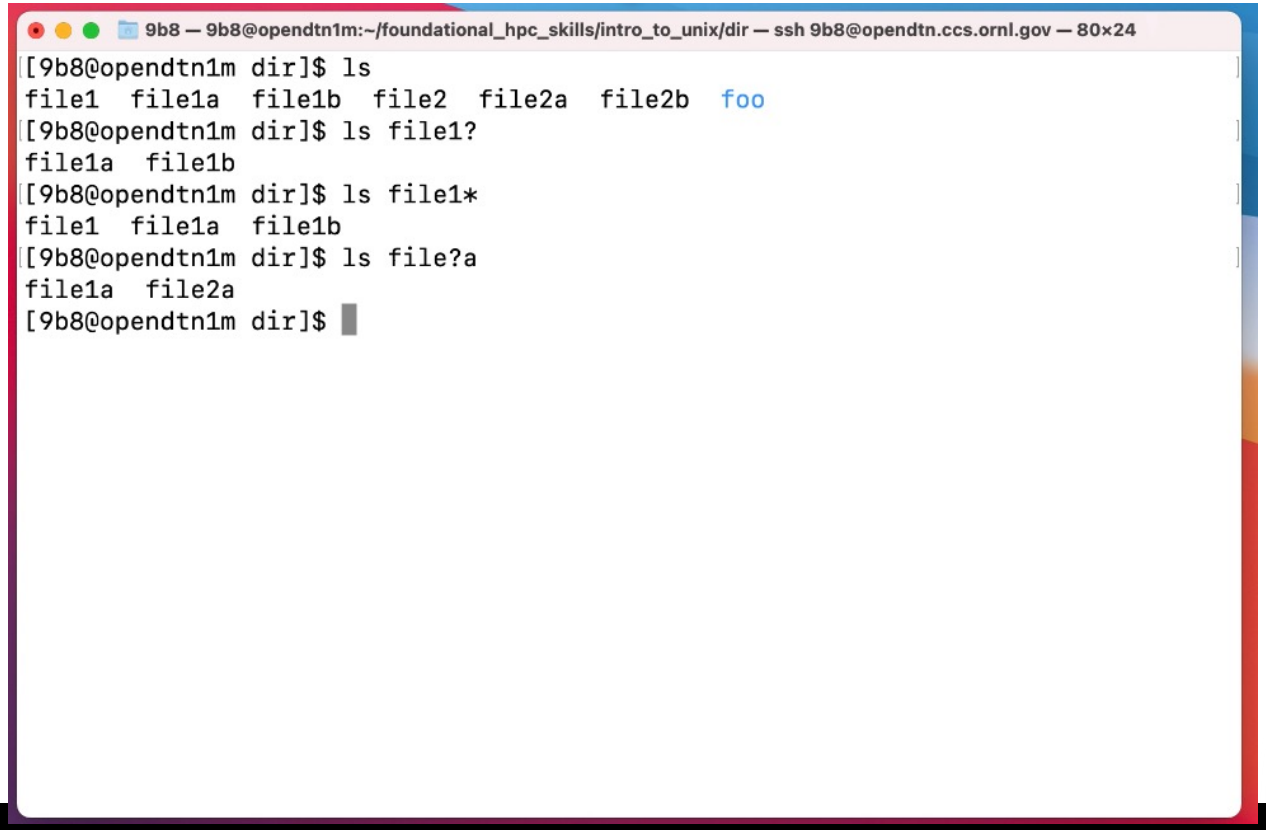
“ls” and “cd” In Action 2



Wildcards

- When dealing with multiple files, it's nice to type a single command for all files at once vs. typing many separate commands for each file
- Wildcards help with this: They are generic characters that "fill in" for other characters
  - `*` means match zero or more character
  - `?` Matches 1 character
  - Example follows (the `ls` command lists files in a directory, we'll worry about specific later)

Wildcards (example)

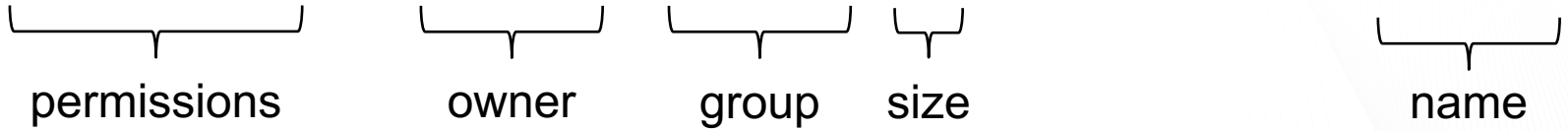



```
$ ls
file1   file1a  file1b  file2   file2a  file2b  file3   file3a  file3b

$ ls file1?
file1a  file1b

$ ls file2*
file2   file2a  file2b

$ ls file?a
file1a  file2a  file3a
```

More About "ls"

- Lists directory contents
- Helpful option: -l (shows many file attributes)

```
$ ls
filea   fileb

$ ls –l
total 0
-rw-r--r--  1 user1  group1 50 Jun 20 14:15 filea
-rw-r--r--  1 user1  group1  0 Jun 20 14:15 fileb
```



Even More "ls" Info (Other Useful Options)

Option	Meaning
`-1`	Show one file per line (helpful in scripting)
`-F`	Show file types (directories, links, etc)
`-a`	Show all files (including hidden files)
`-r`	Reverse the order of the listing
`-t`	Sort files by timestamp
`-d`	List the (attributes of) the directory itself rather than listing its contents
…And many, many (many) more	

- You can combine options: `ls -altr` is the same as `ls -a -l -t -r` (but more concise & w/less typing)

Directories

Use `pwd, cd, mkdir, rmdir` commands to navigate the filesystem and manipulate directories

```
$ pwd
/home/user1

$ mkdir dir1

$ ls
dir1

$ cd dir1

$ pwd
/home/user1/dir1

$ cd ..

$ rmdir dir1
```

Typing just `cd` will always take you back to home no matter where you are.

Directories must be empty in order to delete them with `rmdir`

Manipulating Files 1

- The `rm` command is used to delete a file.
- The `mv` command is used to move and rename files
- There are multiple ways to create and view text files. In the challenge, we will look at various ways to use `cat` and `less` commands to do this.
- Utilities such as `less` and `cat` are intended only for text files. The system will not stop you from running them on a non-text file
  - If you do, you'll get a screenful of unintelligible characters
  - You might get a recognizable prompt (you might not)
  - There's no shame in closing that session's window & re-connecting

Manipulating Files 2

The `cp` command is used to copy a file, and the `mv` command is used to move and rename files.

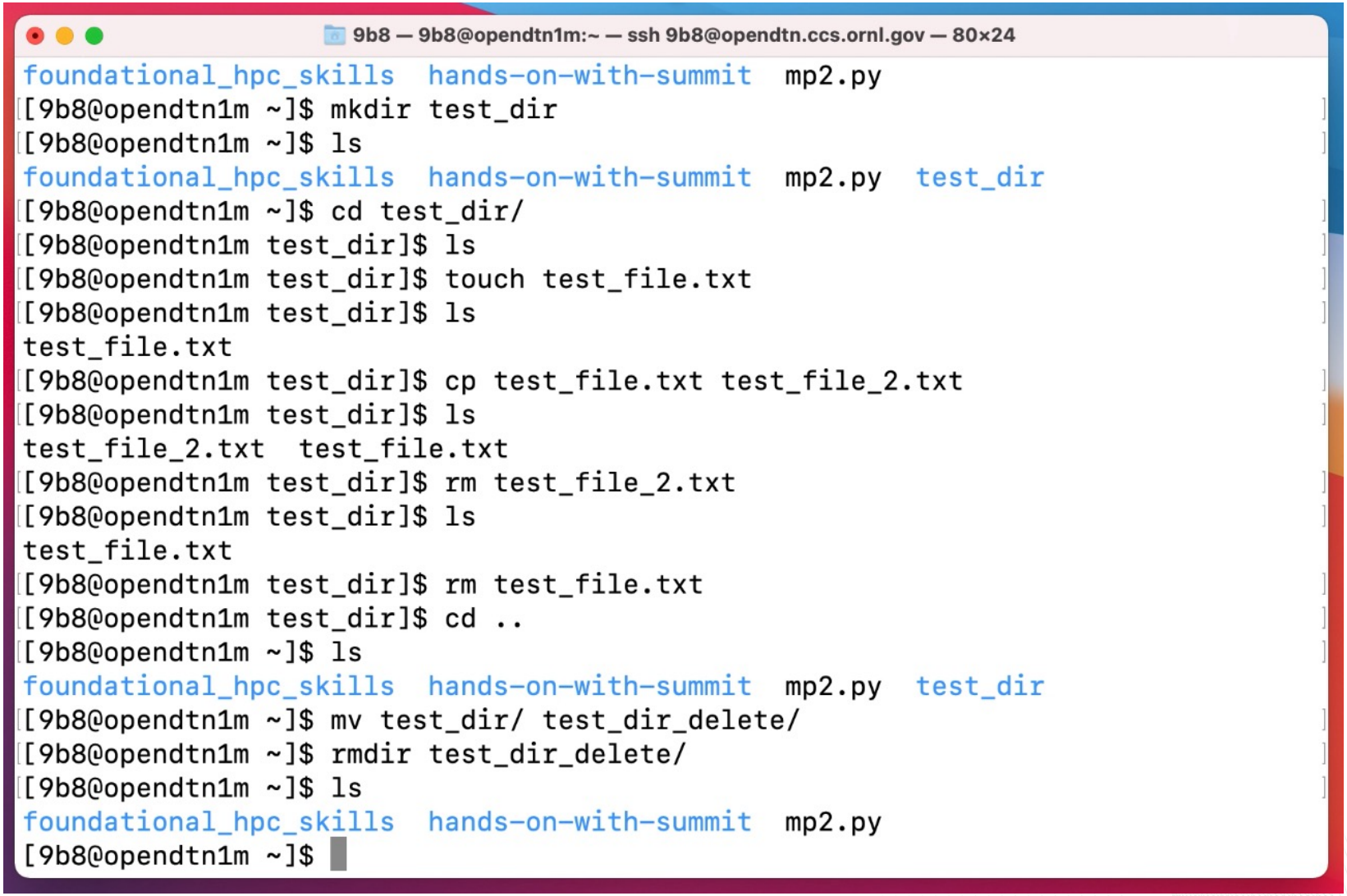
```
$ ls
dir1    filea   fileb

$ cat filea
This is a file that
contains three lines
of text.

$ cp filea filea1

$ ls
dir1    filea   filea1  fileb

$ mv filea1 filec

$ ls
dir1    filea   fileb   filec
```

Manipulating Files 3

The `rm` command is used to delete a file. `cat` prints contents of a file to the screen. `less` displays the contents of a file in a viewing mode. (Press “q” to exit).

```
$  cat filec
This is a file that
contains three lines
of text.

$ less filec

$ mv filec dir1

$ ls dir1
filec

$ rm dir1/filec

$ ls
dir1    filea   fileb
```

Manipulating Files In Action



“cat” and “less” In Action

Many uses of cat

Command	Explanation
cat file1.txt	Display contents of file
cat file1.txt file2.txt	Concatenate two text files and display the result in the terminal
cat file1.txt file2.txt > newcombinedfile.txt	Concatenate two text files and write them to a new file
cat >newfile.txt	Create a file called newfile.txt. Type the desired input and press CTRL+D to finish. The text will be in file newfile.txt.
cat -n file1.txt file2.txt > newnumberedfile.txt	Some implementations of cat, with option -n, can also number lines
cat file1.txt > file2.txt	Copy the contents of file1.txt into file2.txt
cat file1.txt >> file2.txt	Append the contents of file1.txt to file2.txt
cat file1.txt file2.txt | less	Run the program "less" with the concatenation of file1 and file2 as its input

Searching Within Files – grep 1

- Sometimes you want to search for patterns/strings in a file. As with other commands, `grep` takes many options. The grep command searches for “regular expressions”…strings that contain characters with special meaning
- Simple case: find lines with the string ‘user’ in file1

```
grep “user” file1
```

- More complex: show lines ending with ‘user’ in file1

```
grep “user$” file1
```

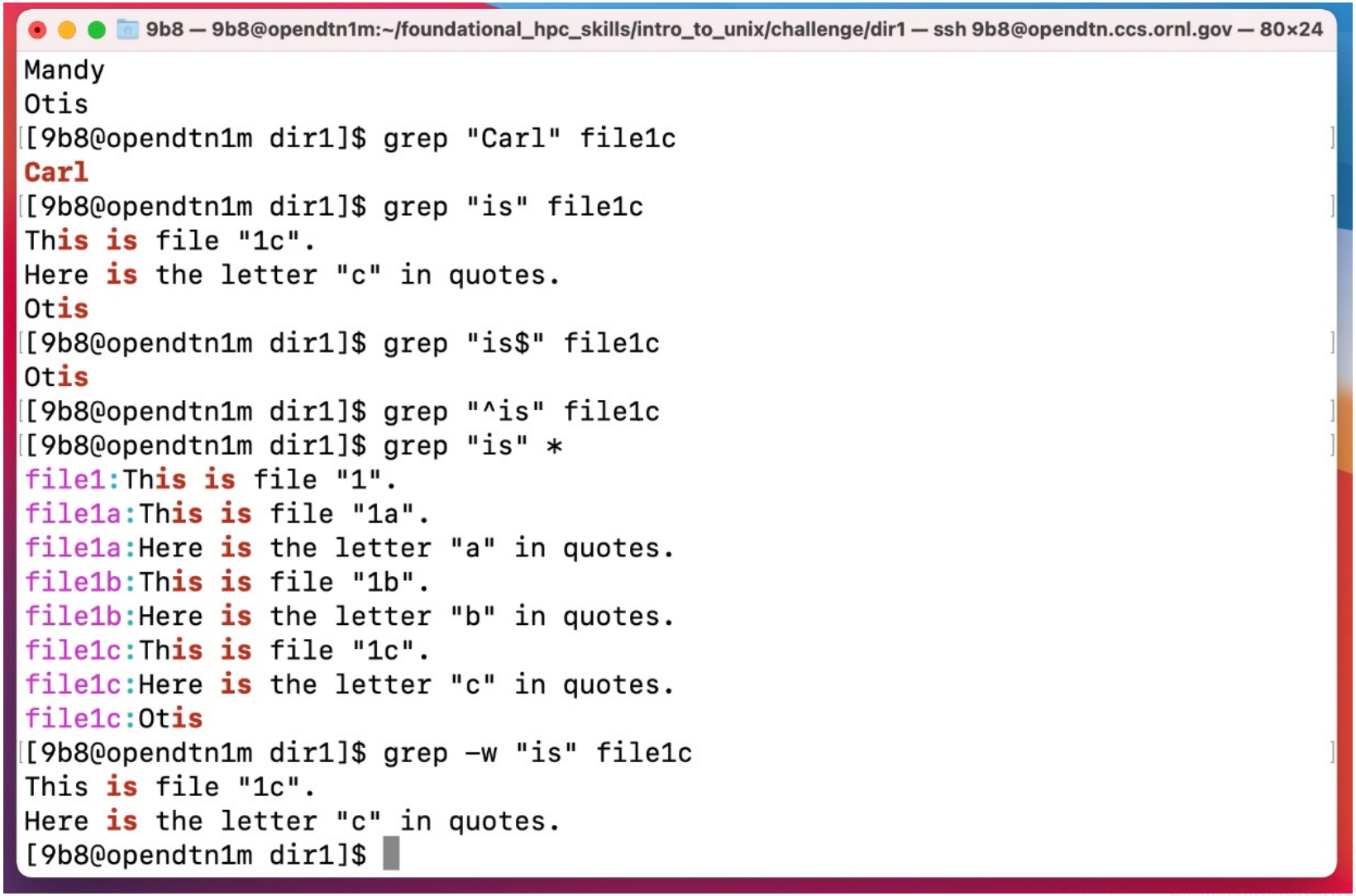
- …or perhaps lines beginning with ‘user’

```
grep “^user” file1
```

- Search all files in a dir1 for the string ‘user’

```
grep –r “user” dir1/
```

“grep” In Action



Searching Within Files – grep 2

- Normally, `grep` will treat anything beginning with a hyphen as an option…even if it's in quotes
- The workaround is the -- option, which tells grep that you're done giving it options (and therefore any other hyphen is meant as an actual hyphen)

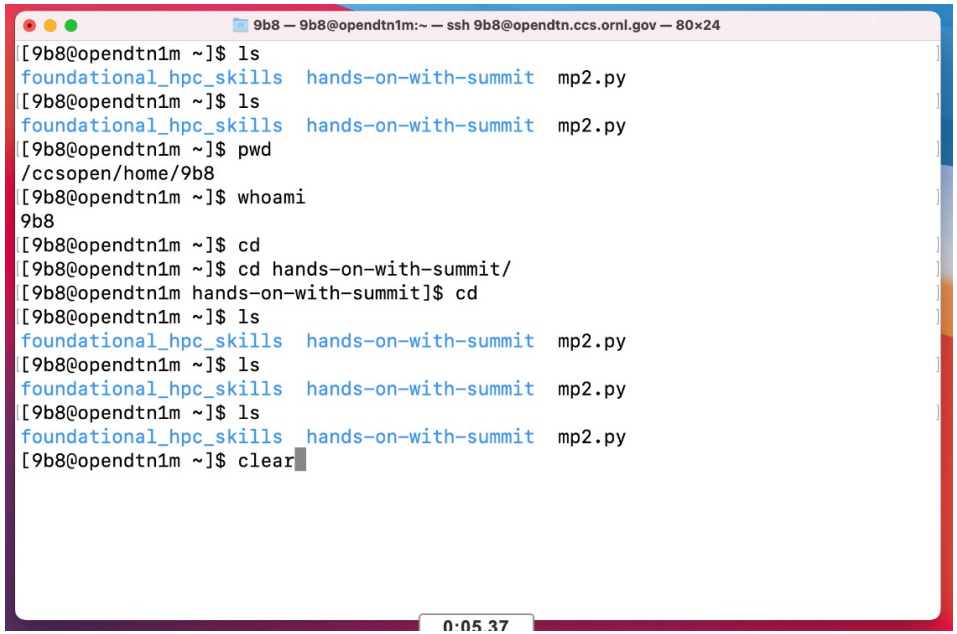
```
grep -- "-2" file1
```

- Grep does NOT recognize wildcards in the quoted string to search for. Use "." instead:

```
grep "user." file1
```

Bonus Tips 1

- “clear” is your friend:



Bonus Tips 2

- Up/down arrows are also your friends

Bonus Tips 3

- Tab-complete is your best friend

Basic File/Directory Commands

Command	Description
`pwd`	**P**rint **w**orking (i.e. current) **d**irectory
`man`	Display the **Man**ual for a command
`whoami`	Display the current user's username
`mkdir`	Create a directory (**M**a**K**e **DIR**ectory)
`rmdir`	Delete a directory (**R**e**M**ove **DIR**ectory)
`cd`	**C**hange (into a) **d**irectory
`ls`	**Ls**t files
`cp`	**Cp**y a file
`mv`	**Mv**e a file (also used to rename a file)
`rm`	**Rm**ove (delete) a file
`cat`	Display the contents (con**cat**enate) a (hopefully text) file
`less`	Display the contents of a text file in a viewing mode
`find`	**Find** files by filename
`grep`	Search for text within files
`diff`	Identify **diff**erences in the contents of files

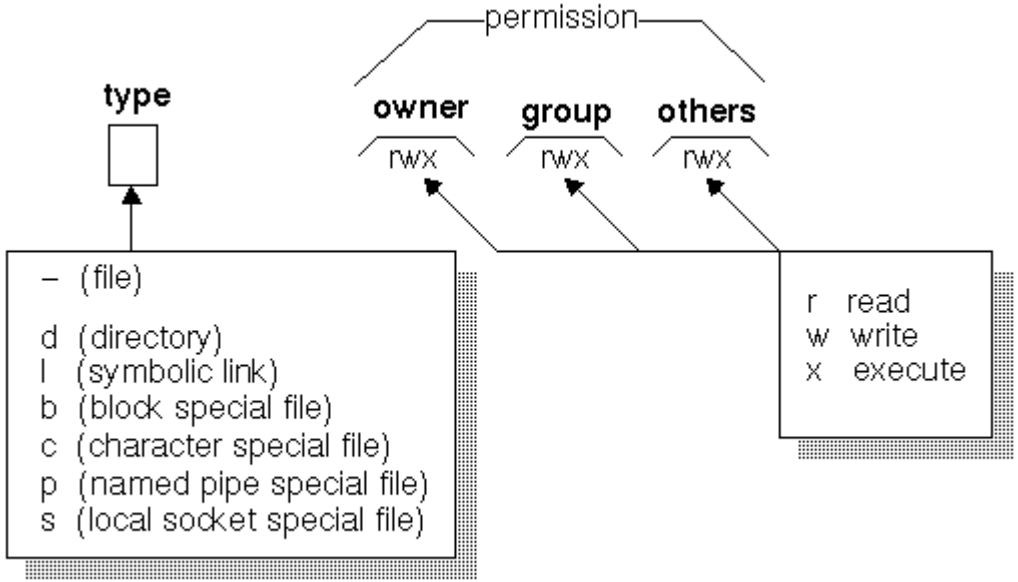
Extra Slides

Files

- Files are the basic entity for storing data
- Might contain a program, data, configuration info, an image, etc.
- Files have several attributes
  - Permissions: who can do what to/with a file
  - Owner: whose file is this
  - Group: to which group does this file belong

Permissions

- In the permissions string, the characters r, w, and x mean read, write, and execute permission is granted
- A - means the permission is not granted
- The permission groups always show read, write, execute in that order



Searching for files

- The `find` command lets you search for files on a huge variety of criteria
- It can also run commands on those files; this makes it one of the most powerful commands available
- General Syntax: `find [path/paths] [expression]`

```
$ find . -name "*data*" -print

$ find . -newer some_file

$ find /home/user1 ! -user user1

$ find . -group users -exec chgrp staff {} \;
```