

### Lesson Module Status

- Slides – draft
- Flash cards – draft
- Properties – done
- Page numbers - done
- 1<sup>st</sup> minute quiz – done
- Web Calendar summary – done
- Web book pages - done
- Commands – done
- Lab tested – done
  
- enlightenment script tested - done
  
- CCC Confer wall paper / quiz - done
- Check that headset is charged – done
- Backup headset charged - done
- Backup slides, CCC info, handouts on flash drive - done



Instructor: **Rich Simms**  
Dial-in: **888-450-4821**  
Passcode: **761867**



Emanuel



Tanner



Merrick



Quinton



Chris



Bobby



Craig



Jeff



Yu-Chen



Terence



Tommy



Eric



Dan M



Geoffrey



Marisol



David



Josh



Gabriel



Jesse



Tajvia



Daniel W



Jason

Email me ([risimms@cabrillo.edu](mailto:risimms@cabrillo.edu)) a relatively current photo of your face for 3 points extra credit

## First Minute Quiz

Please close your books, notes, lesson materials, forum and answer these questions **in the order** shown:

**email answers to: [risimms@cabrillo.edu](mailto:risimms@cabrillo.edu)**



- [ ] Has the phone bridge been added?
- [ ] Is recording on?
- [ ] Does the phone bridge have the mike?
- [ ] Share slides, putty (rsimms, simmsben, roddyduk), Chrome
- [ ] Disable spelling on PowerPoint

# The UNIX/Linux File System

Objectives	Agenda
<ul style="list-style-type: none"><li>• Become familiar with the UNIX file hierarchy.</li><li>• Be able to navigate the hierarchy using cd, ls and pwd commands.</li><li>• Understand the key elements of a file.</li><li>• Be able to distinguish the different UNIX files types.</li><li>• Learn appropriate commands to view file contents.</li></ul>	<ul style="list-style-type: none"><li>• Quiz</li><li>• Questions</li><li>• The UNIX Directory Hierarchy</li><li>• Navigating the file system</li><li>• File types</li><li>• Viewing files</li><li>• Exercise: Enlightenment</li><li>• Wrap up</li></ul>

## Previous material and assignment

1. Questions on previous material?

2. Lab 3 questions?

- I'll use check3 for grading
- bash shell vs mail "shell"
- clean up duplicates before last submittal
- mail \$(ls /home/cis90ol)
- mail -f, mail -f mbox, mail -f uhistory

## Lab 2 Results

1) show shell	(0)
2) type commands	xxxxx (5)
3) echo variables	x (1)
4) set TERM	xxx (3)
5) upper/lower case	(0)
6) who -g	xx (2)
7) number of arguments	xxxxxxxxxx (10)
8) CR and quotes	xxxxxxxx (8)
9) ; to separate commands	xxxxxx (6)
10) change password	(0)
11) uname options	xxxxxxxxxx xxxx (14)
12) banner	(0)
13) finger	xx (2)
14) id	x (1)
15) man	(0)
16) whatis vs man -f	xxxxx (5)
17) Tryme	xxxxxxxxxx (11)
18) who -q	xxxxxxxx (8)
19) man -k vs apropos	xxxxxxxxxx (9)
20) info bash	(0)
21) Google	(0)
22) sqrt	xx (2)
Q1 – input from cmd line	x (1)
Q2 – input from keyboard	xxx (3)
Q3 – input from OS	xx (2)

## Lab 2 Results – Q2

2. Use the following commands as arguments to the type command, to find out where each of the commands resides.

cmd argument

**type man**

**type uname**

**type tryme**

**type echo**

**type type**

## Lab 2 Results – Q2

```
/home/cis90ol/simmsben $ type man
man is /usr/bin/man
```

*The **man** command is in the /usr/bin/ directory*

```
/home/cis90ol/simmsben $ type uname
uname is /bin/uname
```

*The **uname** command is in the /bin/ directory*

```
/home/cis90ol/simmsben $ type tryme
tryme is /home/cis90ol/simmsben/bin/tryme
```

*The **tryme** command is in the bin/ directory of our home directory*

```
/home/cis90ol/simmsben $ type echo
echo is a shell builtin
```

*The **echo** and **type** commands are built into the bash shell*

```
/home/cis90ol/simmsben $ type type
type is a shell builtin
```

## Lab 2 Results – Q7

7. How many arguments do each of the following command lines have?

**echo one two threefour**

**echo "My TERM type is " \$TERM**

**echo one.two.three**

## Lab 2 Results – Q7

```
/home/cis90ol/simmsben $ echo one two threefour  
one two threefour  
(3 arguments)
```

```
/home/cis90ol/simmsben $ echo "My TERM type is " $TERM  
My TERM type is xterm  
(2 arguments)
```

```
/home/cis90ol/simmsben $ echo one.two.three  
one.two.three  
(1 argument)
```

## Lab 2 Results – Q8

8. What is the difference in output between the following two commands? Note, the \$ and > are part of the prompt, you don't need to type them.

```
$ echo red 'white  
> and blue'
```

and

```
$ echo red white \  
> and blue
```

Note: the [enter] key is pressed immediately after the last character of each line

## Lab 2 Results – Q8

*Pressing the Enter (or Return on Macs) key generates an invisible <newline> metacharacter. This signals the shell to stop prompting and process the command line.*

```
/home/cis90ol/simmsben $ echo red 'white <newline>
> and blue'
red white
and blue
```

*The unclosed single quote prevents the <newline> from signaling the end of the command. Instead the shell continues to prompt and the <newline> gets passed to the echo command.*

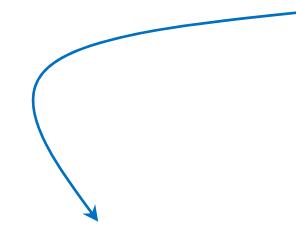
```
/home/cis90ol/simmsben $ echo red white \<newline>
> and blue
red white and blue
```

*The <newline> is escaped in this example. The shell ignores it and continues to prompt the user for the rest of the command. The escaped <newline> and is never passed to the echo command.*

## Lab 2 Results – Q8

*Note: Primary prompt is determined by the value of PS1*

```
/home/cis90ol/simmsben $ echo $PS1  
$PWD $
```



```
/home/cis90ol/simmsben $ echo red 'white  
> and blue'
```

```
red white  
and blue
```

*Note: Secondary prompt is determined by the value of PS2*

```
/home/cis90ol/simmsben $ echo $PS2  
>
```

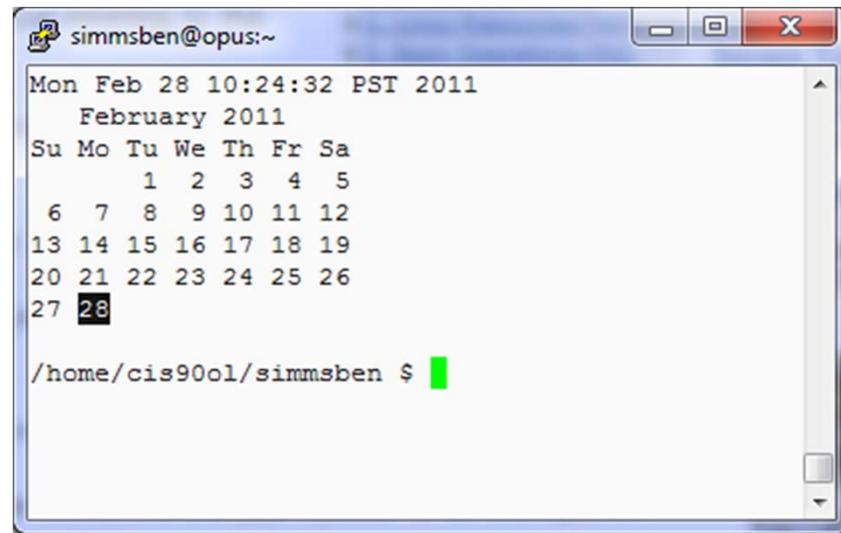
## Lab 2 Results – Q9

9. Use the shell metacharacter ";" to write out a one line command that will clear the screen, print out the date and the current month's calendar.

\$ \_\_\_\_\_

## Lab 2 Results – Q9

```
/home/cis90ol/simmsben $ clear;date;cal
```



The screenshot shows a terminal window titled "simmsben@opus:~". The window displays the following text:

```
Mon Feb 28 10:24:32 PST 2011
February 2011
Su Mo Tu We Th Fr Sa
 1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28

/home/cis90ol/simmsben $
```

The date command shows Monday, February 28, 2011 at 10:24:32 PST. The calendar command shows the month of February 2011. The final prompt shows the user's home directory and the dollar sign.

*The ; metacharacter allows multiple commands on one line*

## Lab 2 Results – Q11

11. Using the **uname** command what options would you use to display just the operating system, it's kernel release numbers and the machine's network node hostname?

(Hint: Use the **man uname** command)

## Lab 2 Results – Q11

```
-a, --all
    print all information, in the following order, ex
    omit -p and -i if unknown:

-s, --kernel-name
    print the kernel name

-n, --nodename
    print the network node hostname

-r, --kernel-release
    print the kernel release

-v, --kernel-version
    print the kernel version

-m, --machine
    print the machine hardware name

-p, --processor
    print the processor type or "un

-i, --hardware-platform
    print the hardware platform or

-o, --operating-system
    print the operating system

--help display this help and exit
```

*Use the man page to determine the options to show just the operating system, it's kernel release numbers and the machine's network node hostname*

```
/home/cis90ol/simmsben $ man uname

/home/cis90ol/simmsben $ uname -orn
opus.cabrillo.edu 2.6.18-164.el5 GNU/Linux

or

/home/cis90ol/simmsben $ uname -o -r -n
opus.cabrillo.edu 2.6.18-164.el5 GNU/Linux
```

*Use q to quit the man page*



*Free Software = Freedom to view and modify the source code*



Richard Stallman started the GNU project in 1983 to create a free UNIX-like OS. He Founded the Free Software Foundation in 1985. In 1989 he wrote the first version of the GNU General Public License

```
/home/cis90ol/simmsben $ uname -orn
opus.cabrillo.edu 2.6.18-164.el5 GNU/Linux
      _____
      node hostname   kernel release   OS
```

*Dan M. didn't like the order the **uname** command printed the information so he downloaded the source code, modified it, recompiled it. He now has his own version of the **uname** command!*

```
cis90@eko-04:~/dan/coreutils-7.4/src$ ./uname -orn
GNU/Linux 2.6.32-27-generic eko-04
      _____
      OS       kernel release   node hostname
```

*See forum post topic "Lab #2...even though 'info uname' output states". This is one of the really cool things about Linux and the GNU General Public License ... if you don't like something about it you can change it!*

## Lab 2 Results – Q16

16. What is the **whatis** command? Use the command with the argument, bc

How does this compare to using the man command with -f option?

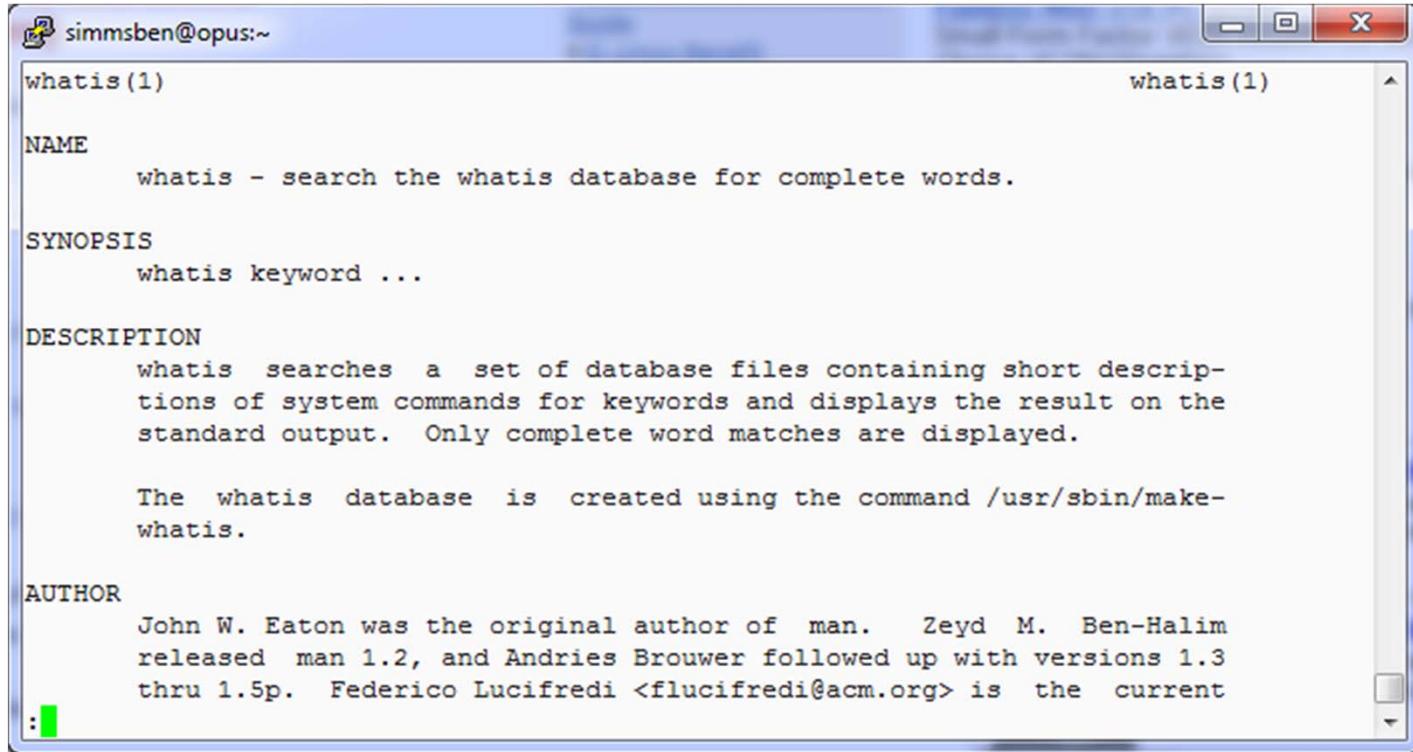
**man -f bc**

## Lab 2 Results – Q16

*Use the **whatis** or **man** command to determine what the **whatis** command does.*

```
/home/cis90ol/simmsben $ whatis whatis
whatis          (1) - search the whatis database for complete words
```

```
/home/cis90ol/simmsben $ man whatis
```



The screenshot shows a terminal window titled "simmsben@opus:~". The window displays the man page for the "whatis" command. The page is organized into sections: NAME, SYNOPSIS, DESCRIPTION, and AUTHOR. The NAME section states "whatis - search the whatis database for complete words." The SYNOPSIS section shows the command "whatis keyword ...". The DESCRIPTION section explains that "whatis" searches a set of database files for keywords and displays the results. The AUTHOR section credits John W. Eaton, Zeyd M. Ben-Halim, Andries Brouwer, and Federico Lucifredi.

```
whatis(1)                                         whatis(1)

NAME
    whatis - search the whatis database for complete words.

SYNOPSIS
    whatis keyword ...

DESCRIPTION
    whatis searches a set of database files containing short descriptions of system commands for keywords and displays the result on the standard output. Only complete word matches are displayed.

    The whatis database is created using the command /usr/sbin/make-whatis.

AUTHOR
    John W. Eaton was the original author of man. Zeyd M. Ben-Halim
    released man 1.2, and Andries Brouwer followed up with versions 1.3
    thru 1.5p. Federico Lucifredi <flucifredi@acm.org> is the current
```

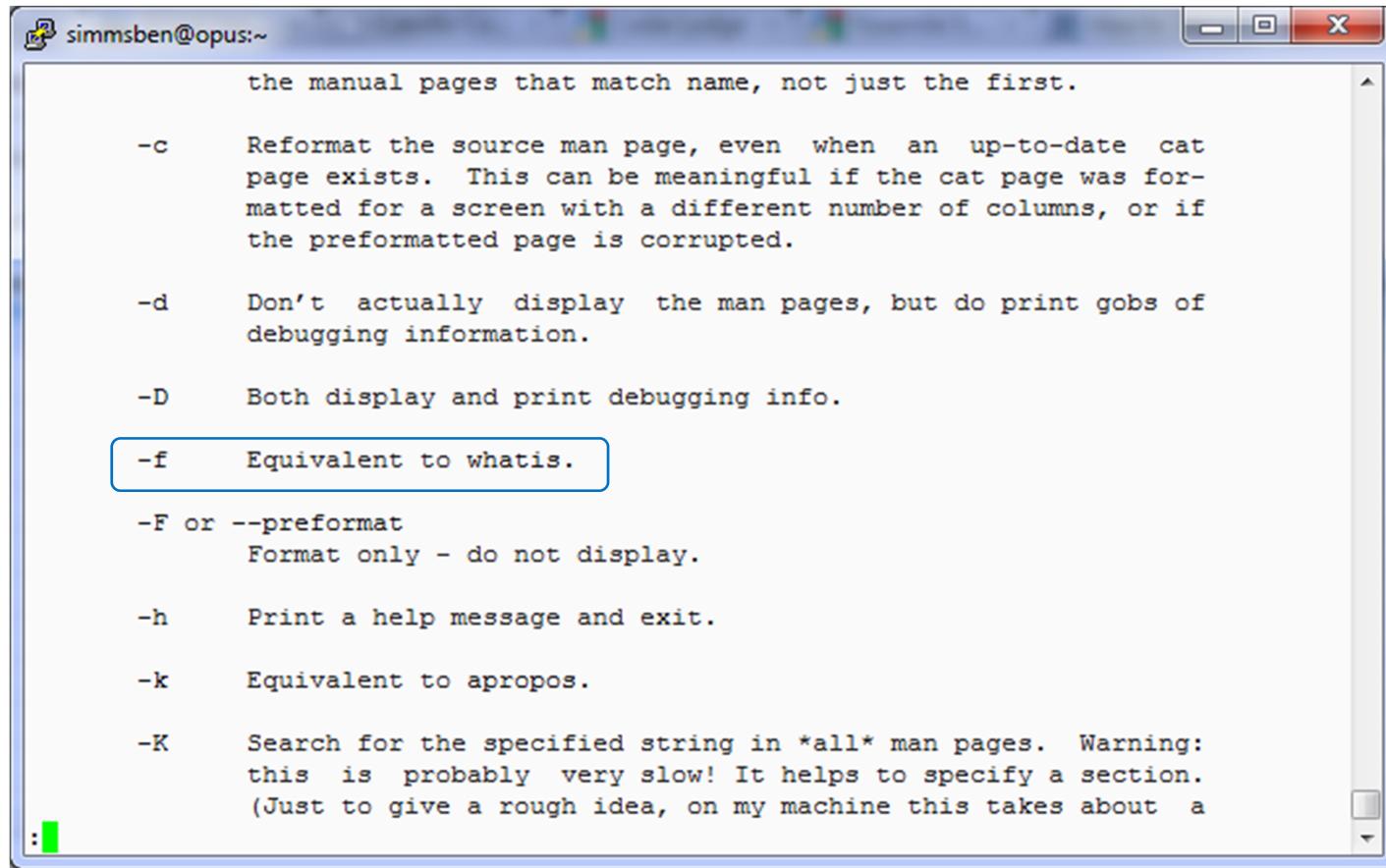
## Lab 2 Results – Q16

*The **whatis** command is the same as the **man -f** command*

```
/home/cis90ol/simmsben $ whatis bc
bc                      (1)  - An arbitrary precision calculator language
bc                      (1p) - arbitrary-precision arithmetic language
bc                      (rpm) - GNU's bc (a numeric processing language)
and dc (a calculator).
```

```
/home/cis90ol/simmsben $ man -f bc
bc                      (1)  - An arbitrary precision calculator language
bc                      (1p) - arbitrary-precision arithmetic language
bc                      (rpm) - GNU's bc (a numeric processing language)
and dc (a calculator).
/home/cis90ol/simmsben $
```

## Lab 2 Results – Q16



simmsben@opus:~

```
the manual pages that match name, not just the first.

-c      Reformat the source man page, even when an up-to-date cat
        page exists. This can be meaningful if the cat page was for-
        matted for a screen with a different number of columns, or if
        the preformatted page is corrupted.

-d      Don't actually display the man pages, but do print gobs of
        debugging information.

-D      Both display and print debugging info.

-f      Equivalent to whatis.

-F or --preformat
        Format only - do not display.

-h      Print a help message and exit.

-k      Equivalent to apropos.

-K      Search for the specified string in *all* man pages. Warning:
        this is probably very slow! It helps to specify a section.
        (Just to give a rough idea, on my machine this takes about a
```

***man man*** will display the manual page for the **man** command and its documented there that the **-f** option is “Equivalent to whatis”

## Lab 2 Results – Q17

17. Is tryme a UNIX command? How do you know?

## Lab 2 Results – Q17

```
/home/cis90ol/simmsben $ tryme
My name is "tryme"
I am pleased to make your acquaintance, Benji Simms
/tmp
```

```
/home/cis90ol/simmsben $ whatis tryme
tryme: nothing appropriate
```

```
/home/cis90ol/simmsben $ man tryme
No manual entry for tryme
```

*UNIX commands are documented with man pages and have entries in the whatis database. **tryme** does not appear in either one so is not a UNIX command*

## Lab 2 Results – Q17

```
/home/cis90ol/simmsben $ type tryme
tryme is /home/cis90ol/simmsben/bin/tryme
```

*type shows **tryme** resides in the bin/ directory of my home directory*

```
/home/cis90ol/simmsben $ file /home/cis90ol/simmsben/bin/tryme
/home/cis90ol/simmsben/bin/tryme: Bourne-Again shell script text executable
```

*file shows **tryme** is a bash shell script*

```
/home/cis90ol/simmsben $ cat /home/cis90ol/simmsben/bin/tryme
#!/bin/bash
```

```
hello()
{
    cd /tmp
}
PATH=/bin
echo My name is `basename $0`"
IFS=:
set `grep $LOGNAME /etc/passwd`
echo I am pleased to make your acquaintance, $5
hello
pwd
```

*cat shows the actual  
**tryme** script itself*

## Lab 2 Results – Q18

18. Use the manual pages, and the **who** command, to find out the number of users logged on.

## Lab 2 Results – Q18

```
--lookup
    attempt to canonicalize hostnames via DNS

-m      only hostname and user associated with stdin

-p, --process
    print active processes spawned by init

-q, --count
    all login names and number of users logged on
    (highlighted with a blue box)

-r, --runlevel
    print current runlevel
```

:

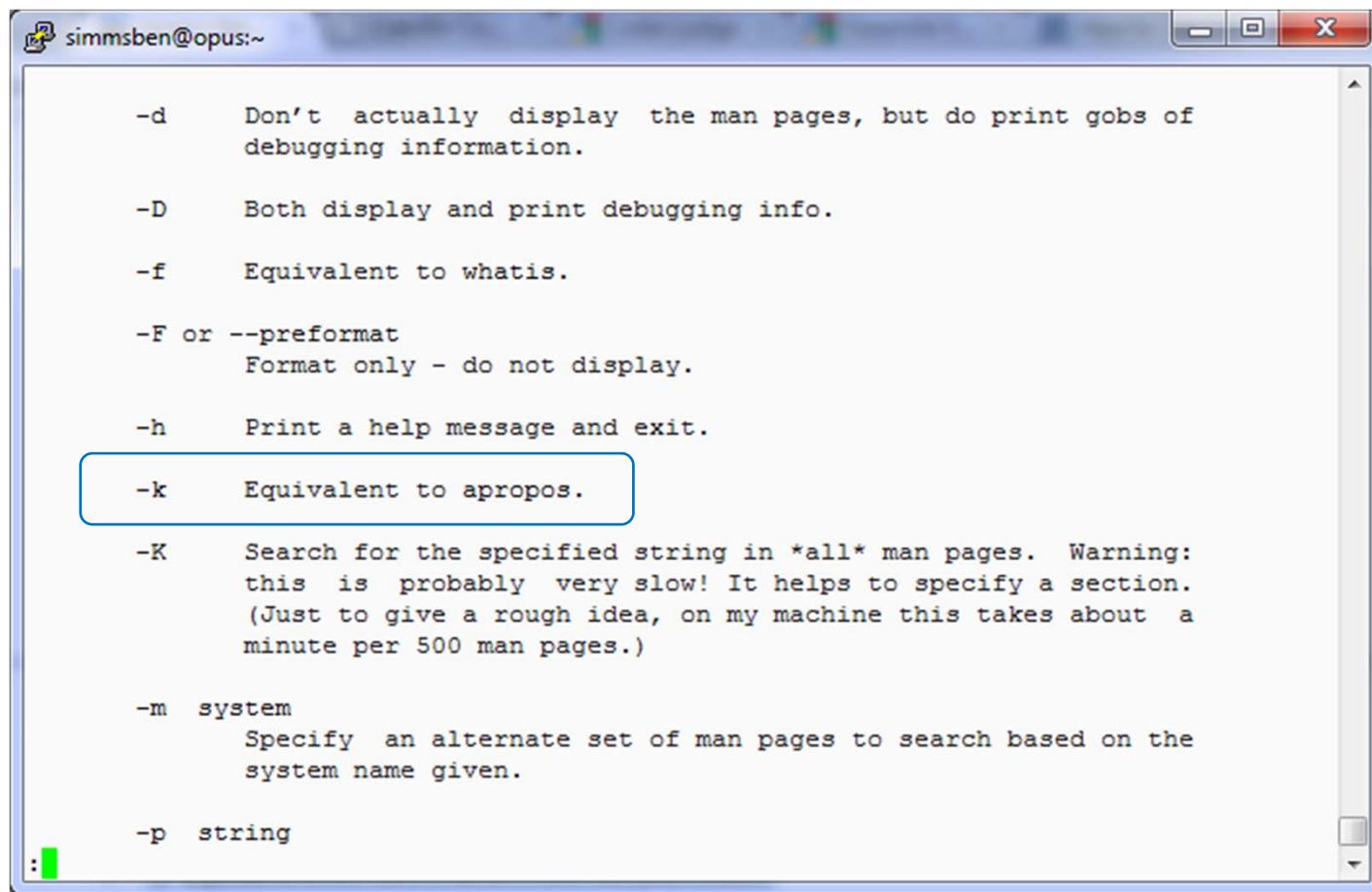
*The man page for **who** shows the **q** option will count the users logged in*

```
/home/cis90ol/simmsben $ who -q
arnsdtha rsimms alvesdes simmsben salinjac wingejas
pirkllau pirkllau wingejas vasqucar
# users=10
/home/cis90ol/simmsben $
```

## Lab 2 Results – Q19

19. Run the command: **man -k boot** Use the manual pages to find out what the -k option does. What command is **man -k** equivalent to? Run the equivalent command and verify.

## Lab 2 Results – Q19



```
simmsben@opus:~
```

```
-d      Don't actually display the man pages, but do print gobs of
       debugging information.

-D      Both display and print debugging info.

-f      Equivalent to whatis.

-F or --preformat
       Format only - do not display.

-h      Print a help message and exit.

-k      Equivalent to apropos.

-K      Search for the specified string in *all* man pages. Warning:
       this is probably very slow! It helps to specify a section.
       (Just to give a rough idea, on my machine this takes about a
       minute per 500 man pages.)

-m  system
       Specify an alternate set of man pages to search based on the
       system name given.

-p  string
```

**Use *man***  
***man* to read**  
***the manual***  
***page for the***  
***man***  
***command***

*the **apropos** command is equivalent to the **man -k** command*

## Lab 2 Results – Q19

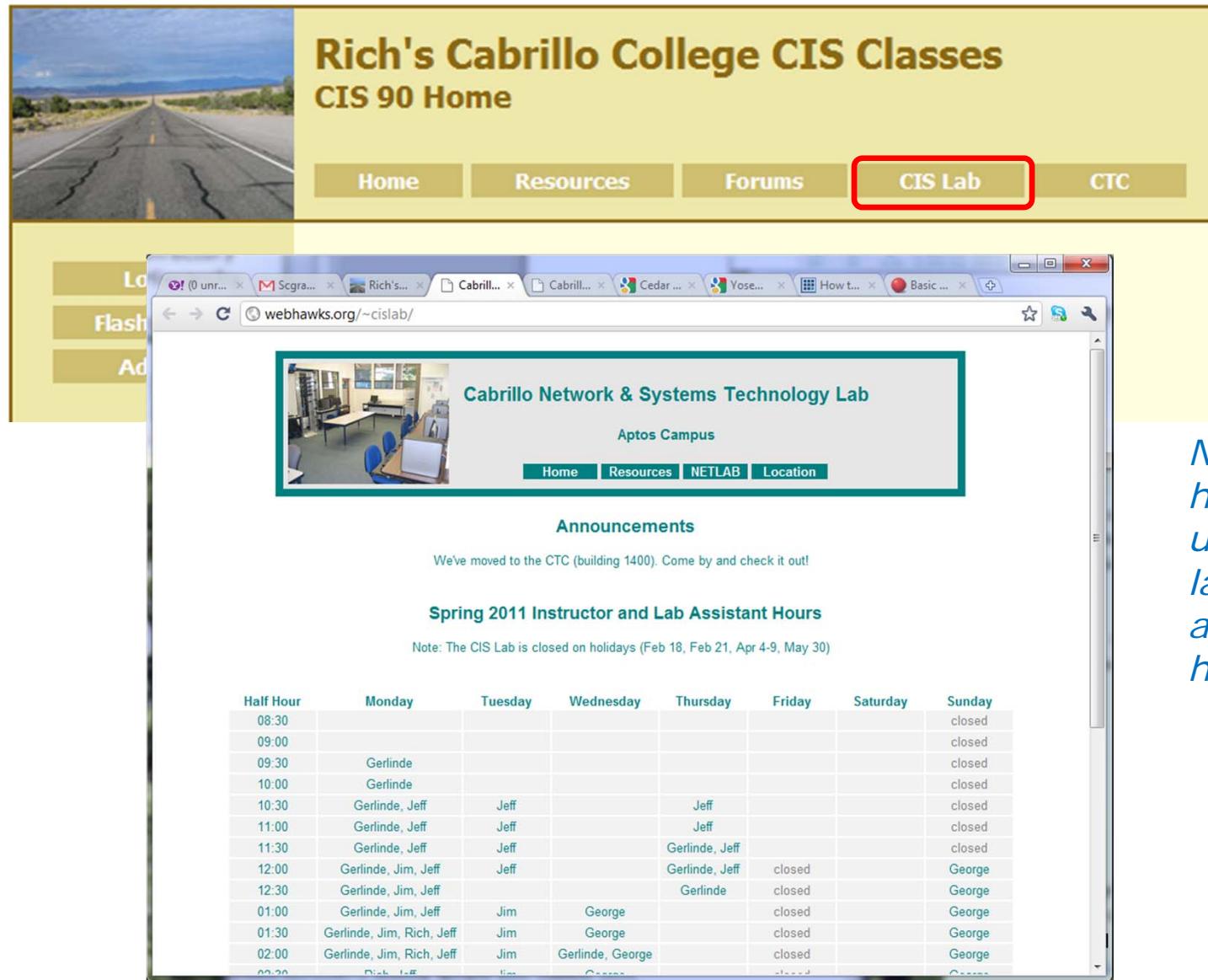
The image shows two terminal windows side-by-side. Both windows have a title bar with the text "simmsben@opus:~". The left window displays the output of the command "/home/cis90ol/simmsben \$ apropos boot". The right window displays the output of the command "/home/cis90ol/simmsben \$ man -k boot". Both outputs list various Linux boot-related commands and packages, such as ExtUtils::Mkbootstrap, boot-scripts, bootparam, firstboot, firstboot-tui, grub, initrd, kexec, mbchk, mkbootdisk, perlboot, pxeboot, pxeos, ty, reboot, reboot [halt], rhgb, sys-unconfig, syslinux, system-config-netboot, and system-config-netboot-c.

```
/home/cis90ol/simmsben $ apropos boot
ExtUtils::Mkbootstrap (3pm) - make a bootstrap file for use by DynaLoader
boot-scripts [boot] (7) - General description of boot sequence
bootparam (7) - Introduction to boot time parameters of the Linux kernel
firstboot (rpm) - Initial system configuration utility
firstboot-tui (rpm)
grub (rpm)
initrd (4)
kexec (8)
mbchk (1)
mkbootdisk (8)
mkbootdisk (rpm)
perlboot (1)
pxeboot (8)
pxeos (8)
ty
reboot (2)
reboot [halt] (8)
rhgb (rpm)
sys-unconfig (8)
syslinux (rpm)
system-config-netboot (8)
system-config-netboot-c (8)
/home/cis90ol/simmsben $ man -k boot
ExtUtils::Mkbootstrap (3pm) - make a bootstrap file for use by DynaLoader
boot-scripts [boot] (7) - General description of boot sequence
bootparam (7) - Introduction to boot time parameters of the Linux kernel
firstboot (rpm) - Initial system configuration utility
firstboot-tui (rpm) - A text interface for firstboot
grub (rpm) - GRUB - the Grand Unified Boot Loader.
initrd (4) - boot loader initialized RAM disk
kexec (8) - directly boot into a new kernel
mbchk (1) - check the format of a Multiboot kernel
mkbootdisk (8) - creates a stand-alone boot floppy for the running system
mkbootdisk (rpm) - Creates a boot floppy disk for booting a system.
perlboot (1) - Beginner(aqs Object-Oriented Tutorial
pxeboot (8) - Network Booting Operating Systems Configuration Utility
pxeos (8) - PXEBoot Operating System description Configuration Utility
ty
reboot (2) - reboot or enable/disable Ctrl-Alt-Del
reboot [halt] (8) - stop the system
rhgb (rpm) - Red Hat Graphical Boot
sys-unconfig (8) - shell script to reconfigure the system upon next boot
syslinux (rpm) - Simple kernel loader which boots from a FAT filesystem
system-config-netboot (8) - Network Booting Configuration Utility
system-config-netboot (rpm) - network booting/install configuration utility (GUI)
system-config-netboot-cmd (rpm) - network booting/install configuration utility
/home/cis90ol/simmsben $
```

*the **apropos** command is equivalent to the **man -k** command*

# Housekeeping

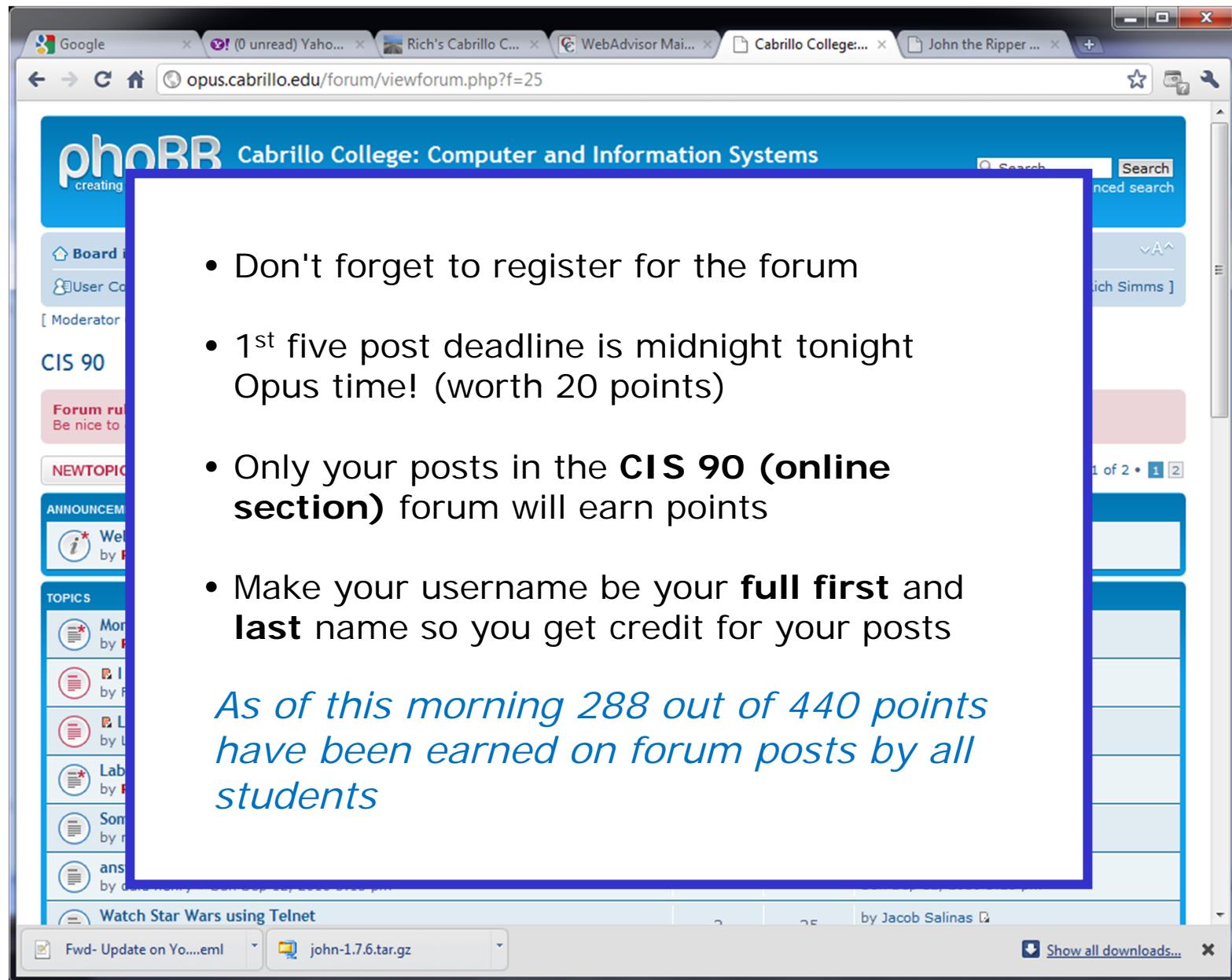
- MSDN AA - is it working?
- Graded labs placed in your home directory
- Answers to labs in /home/cis90ol/answers/ directory
- Lab 3 and five forum posts due tonight



The screenshot shows a website for "Rich's Cabrillo College CIS Classes". The main navigation bar includes "Home", "Resources", "Forums", "CIS Lab" (which is highlighted with a red box), and "CTC". On the left, there are links for "Loc", "Flash", and "Ad". The main content area features a photograph of a long, straight road leading into the distance under a blue sky. Below the image, the text "Rich's Cabrillo College CIS Classes" and "CIS 90 Home" is displayed. A sub-page window titled "Cabrillo Network & Systems Technology Lab" is open in a browser, showing a photo of a lab room with desks and computers, and the text "Aptos Campus". The browser's address bar shows "webhawks.org/~cislabs/". The sub-page also has a "Home" link and a "Resources" link. Below the sub-page, there is an "Announcements" section with the message "We've moved to the CTC (building 1400). Come by and check it out!", and a "Spring 2011 Instructor and Lab Assistant Hours" section. This section includes a note about closed days and a detailed table of hours for various days and times.

Half Hour	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
08:30							closed
09:00							closed
09:30	Gerlinde						closed
10:00	Gerlinde						closed
10:30	Gerlinde, Jeff	Jeff		Jeff			closed
11:00	Gerlinde, Jeff	Jeff		Jeff			closed
11:30	Gerlinde, Jeff	Jeff		Gerlinde, Jeff			closed
12:00	Gerlinde, Jim, Jeff	Jeff		Gerlinde, Jeff	closed		George
12:30	Gerlinde, Jim, Jeff			Gerlinde	closed		George
01:00	Gerlinde, Jim, Jeff	Jim	George		closed		George
01:30	Gerlinde, Jim, Rich, Jeff	Jim	George		closed		George
02:00	Gerlinde, Jim, Rich, Jeff	Jim	Gerlinde, George		closed		George
02:30	Dick, Jeff	Jim	George		closed		George

*Note: CIS Lab hours have been updated with the latest instructor and lab assistant hours*



Google (0 unread) Yahoo... Rich's Cabrillo C... WebAdvisor Mai... Cabrillo College... John the Ripper ... opus.cabrillo.edu/forum/viewforum.php?f=25

phoBB Cabrillo College: Computer and Information Systems

Board index User Control Panel Moderator CIS 90 Forum rules Be nice to others NEWTOPIC ANNOUNCEMENT Welcome by F. TOPICS More by F. I by F. L by L. Lab by F. Some by n. ans by d. Watch Star Wars using Telnet

Search Advanced search

Rich Simms ] 1 of 2 • 1 [2]

- Don't forget to register for the forum
- 1<sup>st</sup> five post deadline is midnight tonight Opus time! (worth 20 points)
- Only your posts in the **CIS 90 (online section)** forum will earn points
- Make your username be your **full first** and **last** name so you get credit for your posts

*As of this morning 288 out of 440 points have been earned on forum posts by all students*

Fwd- Update on Yo....eml john-1.7.6.tar.gz Show all downloads...

# The UNIX Directory Hierarchy

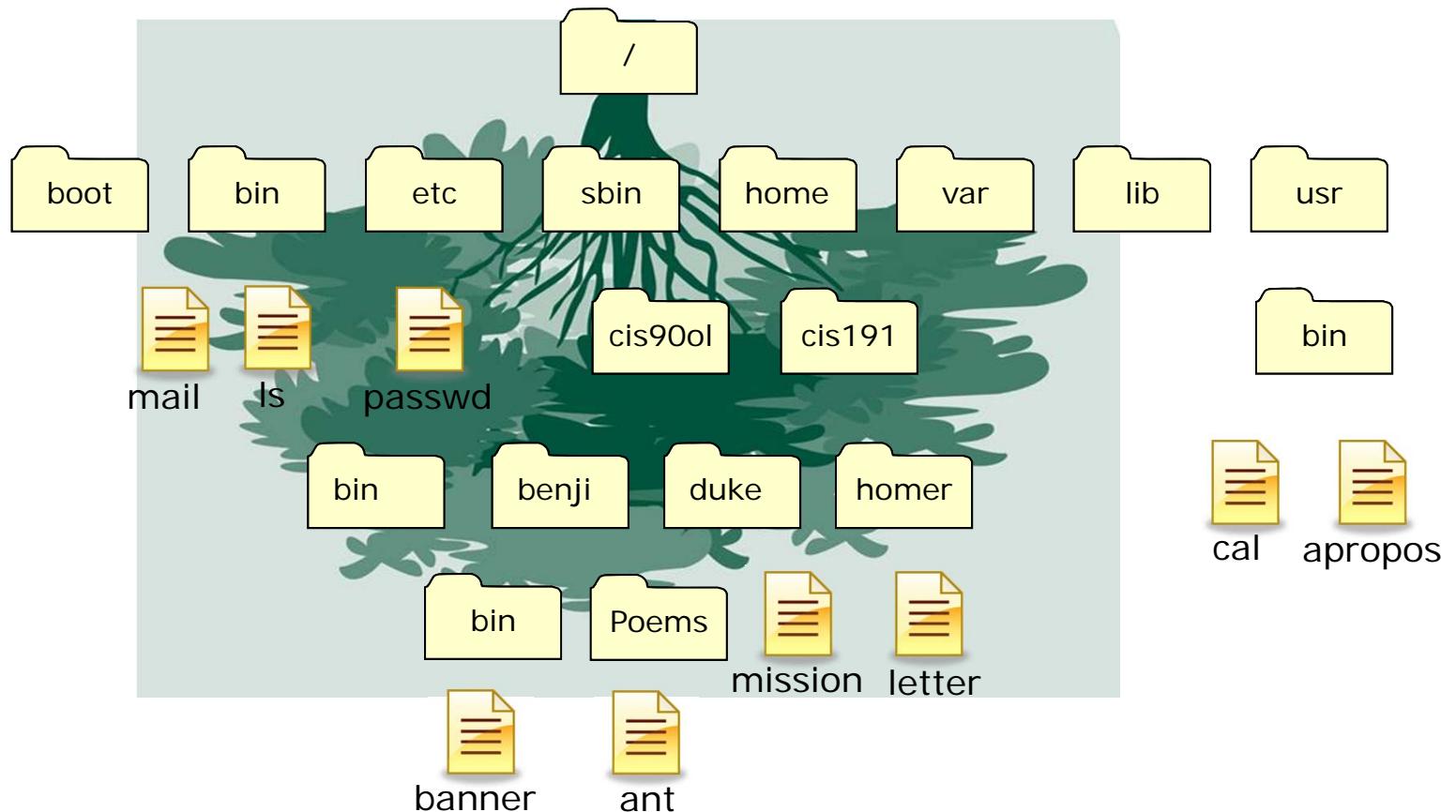
# UNIX File Tree

/ = root of the tree



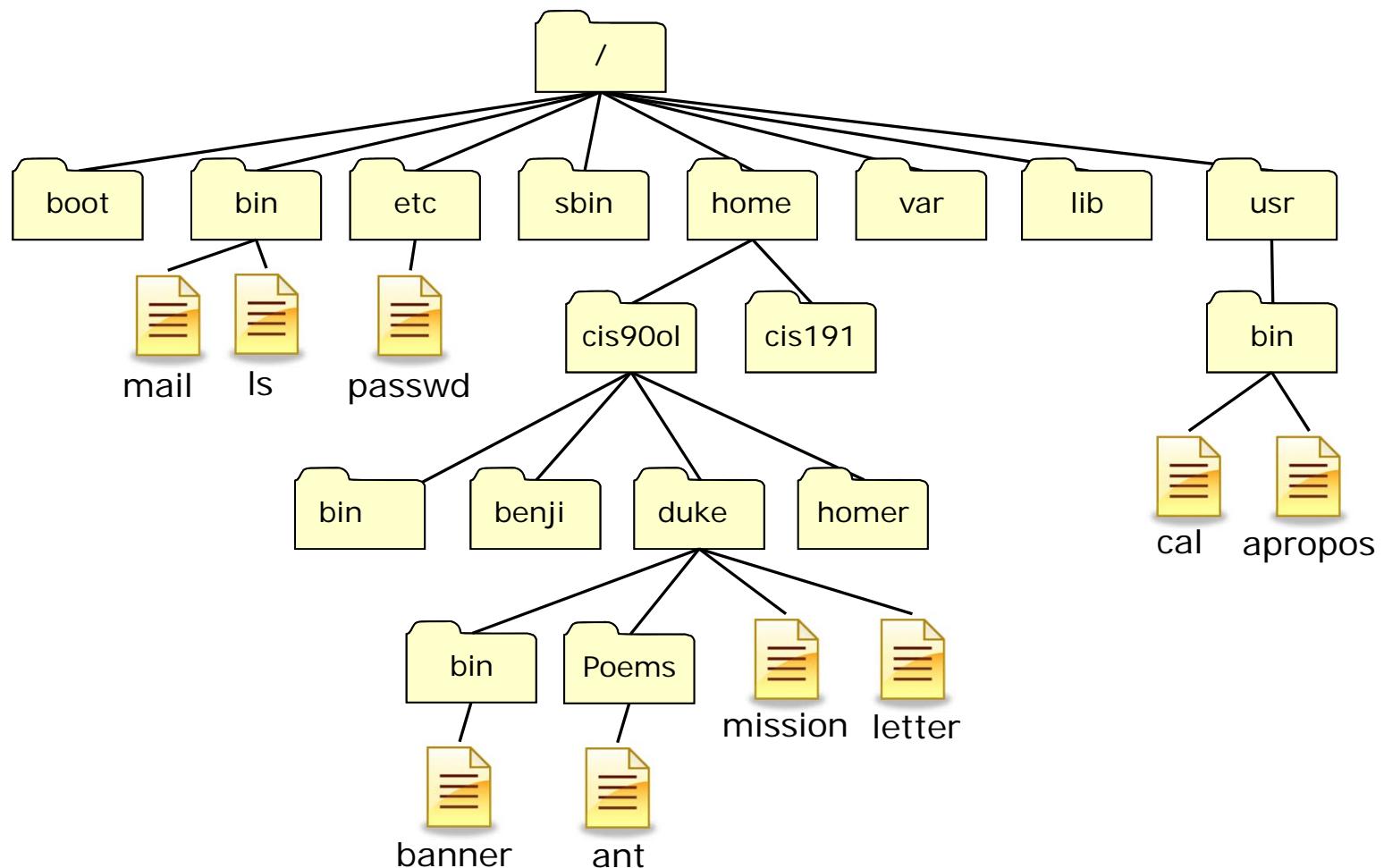
# UNIX File Tree

/ = root of the tree



# UNIX File Tree

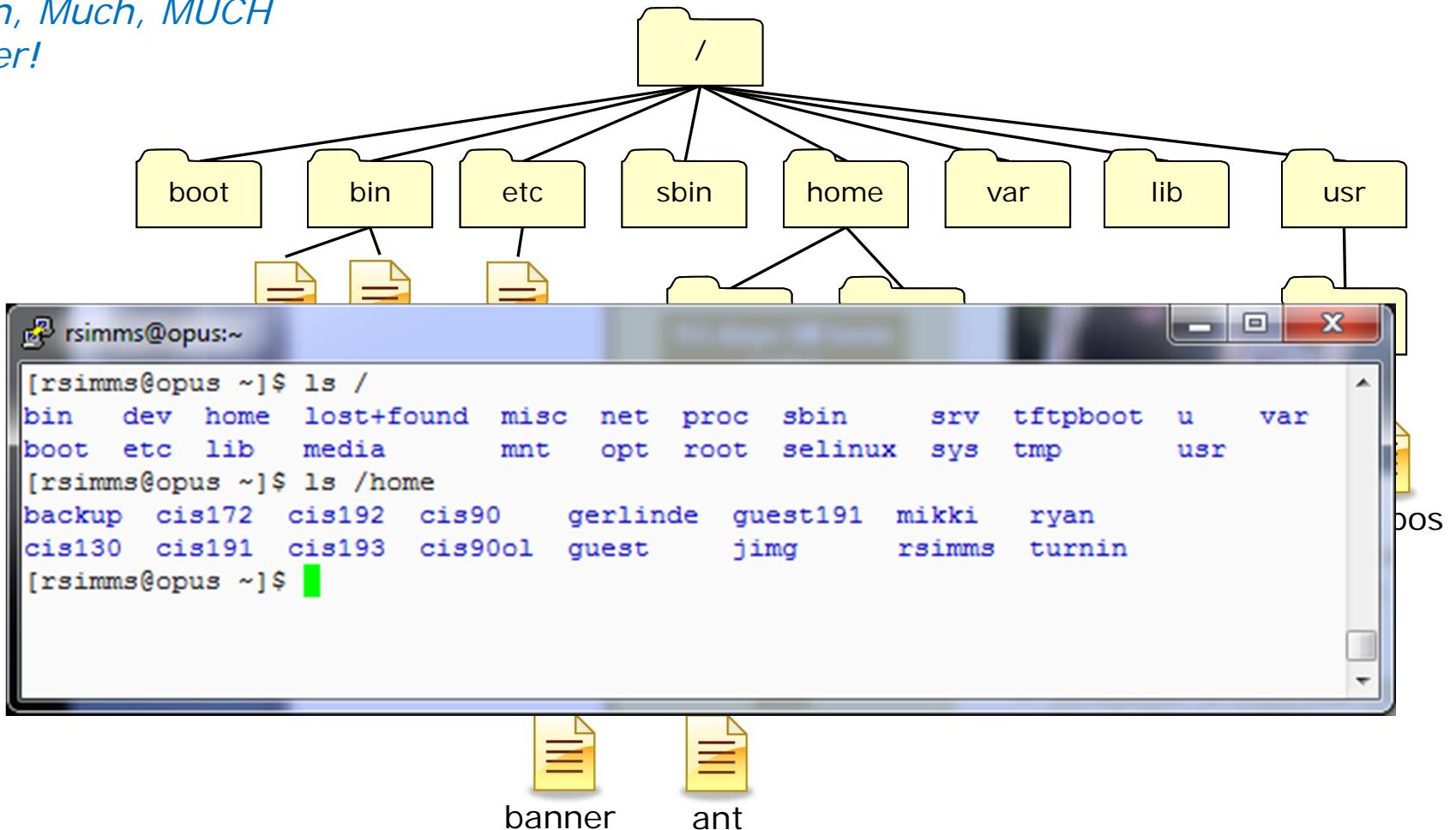
/ = root of the tree



# UNIX File Tree

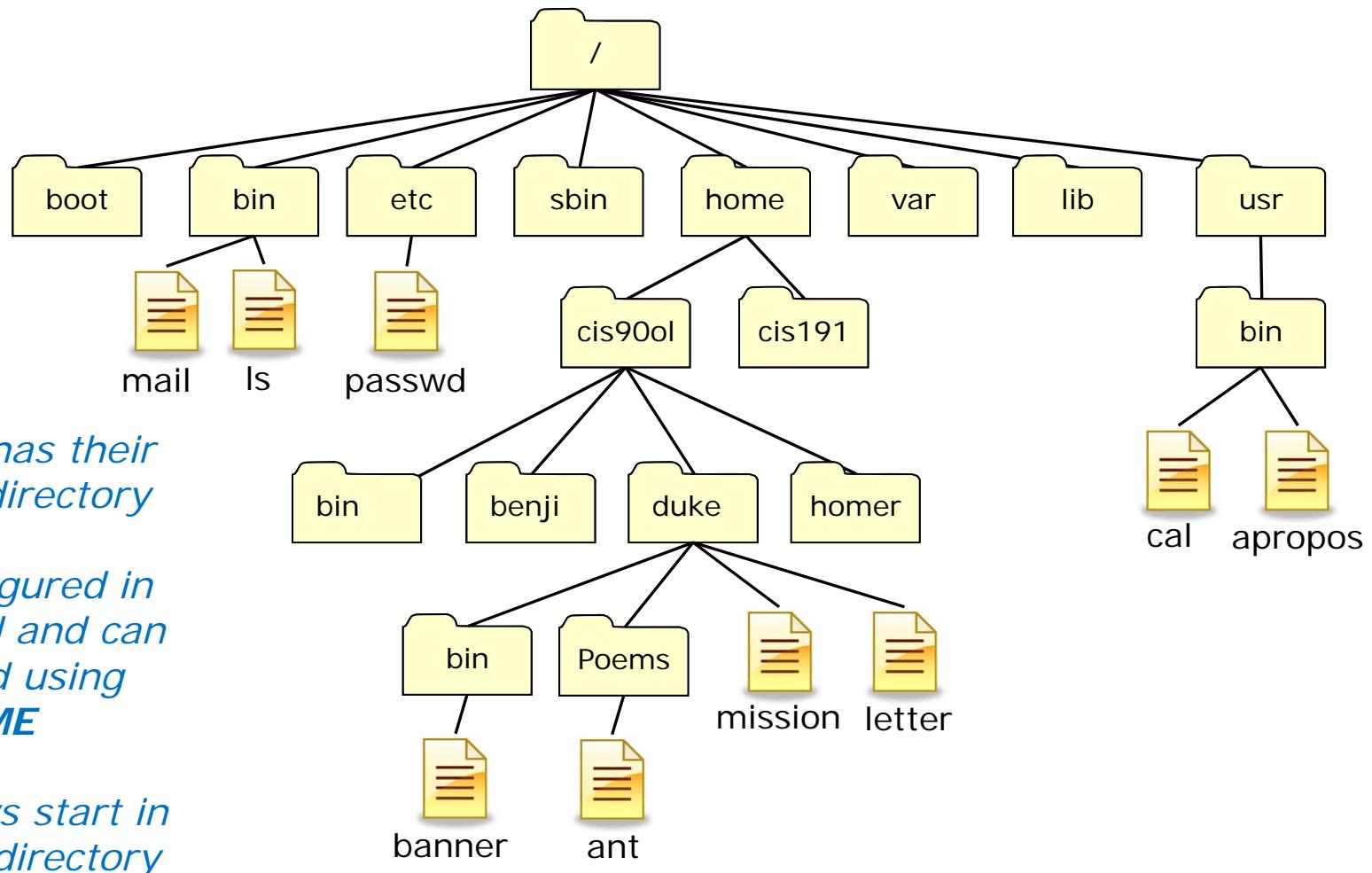
/ = root of the tree

*Note, the actual file tree on Opus is much, Much, MUCH bigger!*



# UNIX File Tree

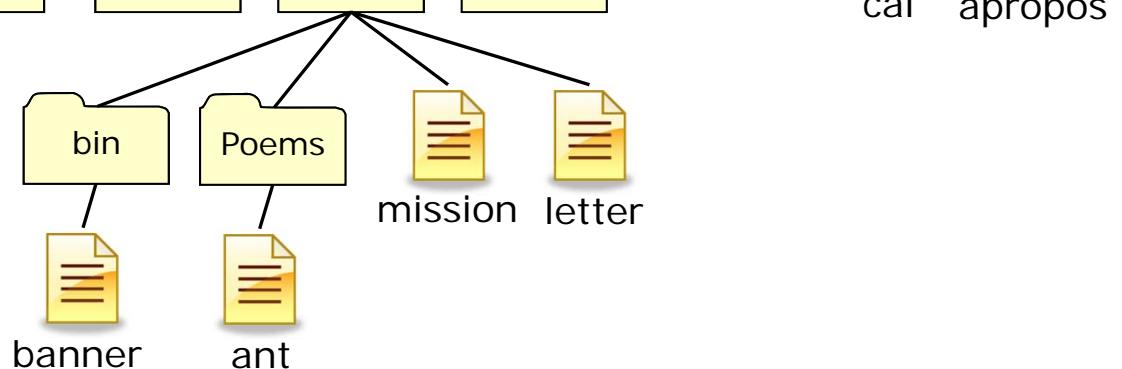
/ = root of the tree



# UNIX File Tree

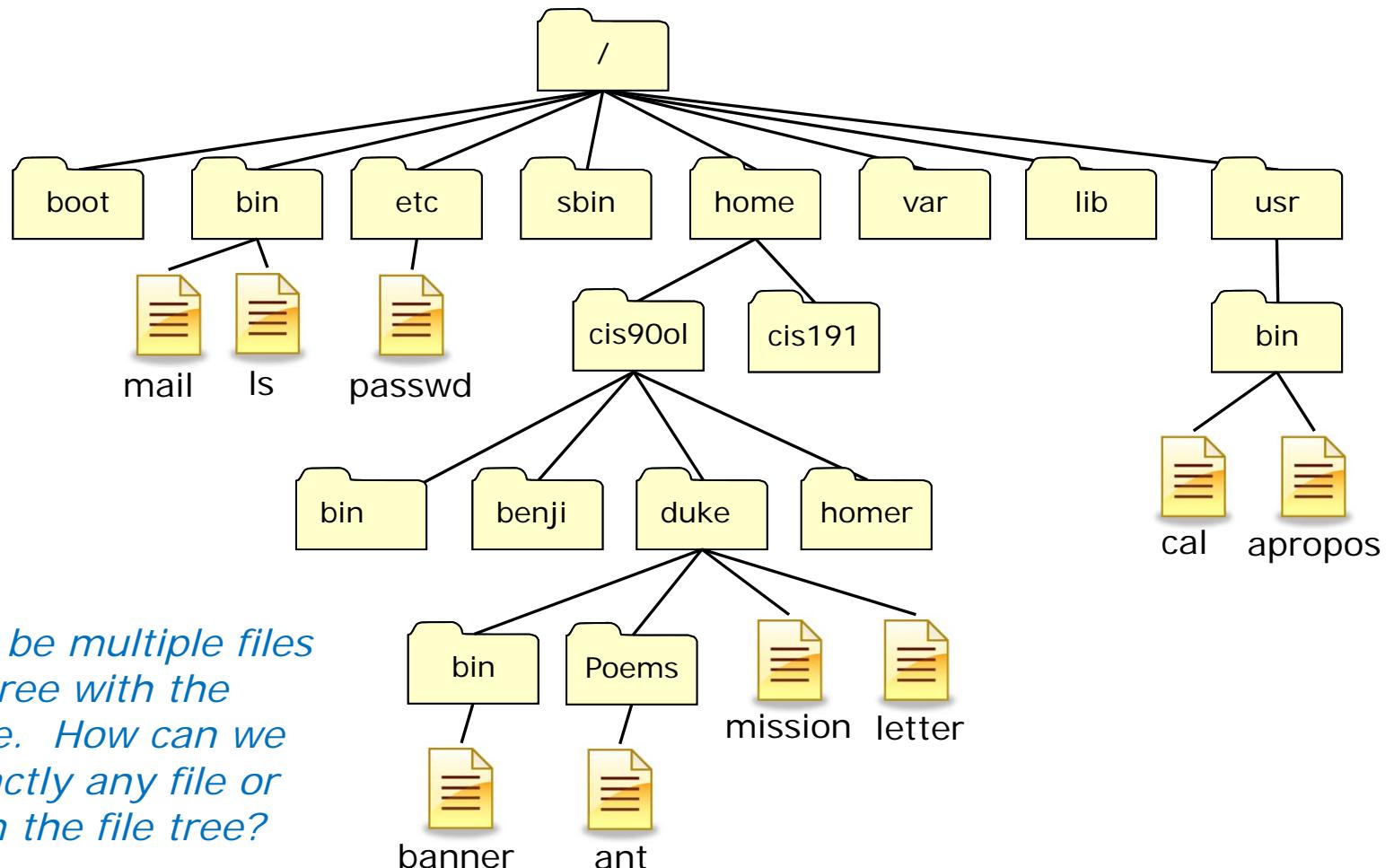
/ = root of the tree

```
simmsben@opus:~  
/home/cis90ol/simmsben $ echo $HOME  
/home/cis90ol/simmsben  
/home/cis90ol/simmsben $ ls  
bigfile      Hidden      log          proposal1  text.err  
bin          lab01.graded mbox         proposal2  text.fxd  
countargs    Lab2.0      Miscellaneous proposal3  timecal  
dead.letter   Lab2.1      mission      small_town uhistory  
empty        letter      Poems        spellk     what_am_i  
/home/cis90ol/simmsben $ cat /etc/passwd | grep simmsben  
simmsben:x:1082:190:Benji Simms:/home/cis90ol/simmsben:/bin/bash  
/home/cis90ol/simmsben $
```



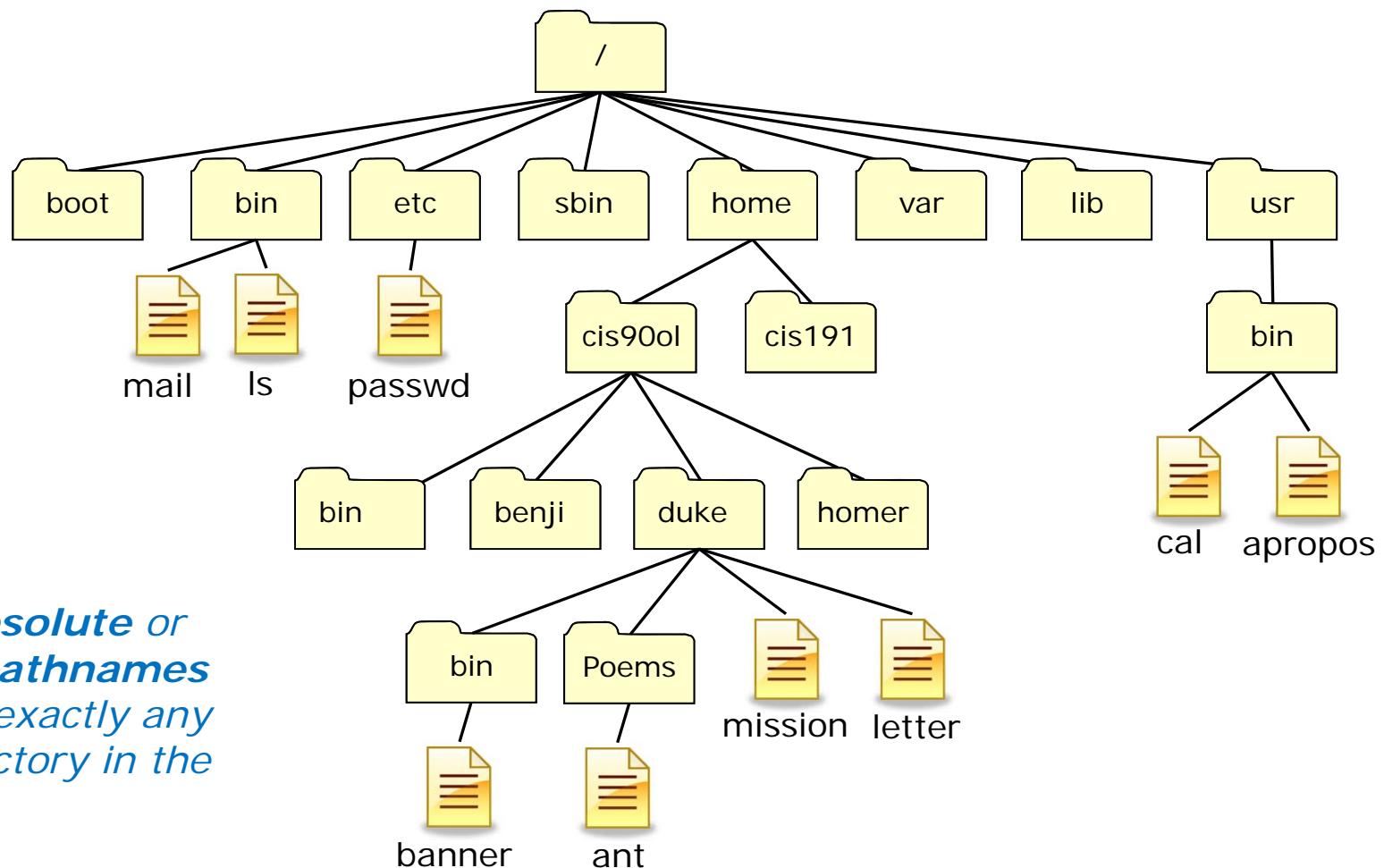
# UNIX File Tree

/ = root of the tree



# UNIX File Tree

/ = root of the tree



We use **absolute or relative pathnames** to specify exactly any file or directory in the tree.

# Pathnames

## What the heck are they?

A pathname is a precise way to specify any file or directory in the file tree.

- An **absolute pathname** specifies the path from the top of the tree to the target directory or file.
- A **relative pathname** specifies the path from your current location to the target directory or file.

*Understanding pathnames is critical because they are used as arguments to all commands that deal with files and directories.*

# Absolute Pathnames

An **absolute pathname** specifies the path from the top of the tree to the target directory or file.

Examples:

/home/cis90ol/duke/Poems/ant	(file)
/boot	(directory)
/usr/bin/cal	(file)
/home/cis90ol/bin/	(directory)
/bin/mail	(file)

*Notice they all start with the /*

# Absolute Pathnames

Using absolute pathnames as command arguments

*An **absolute pathname** specifies the path from the top of the tree to the target directory or file.*

Examples of absolute pathnames used as command arguments:

```
ls /bin /sbin /usr/bin /usr/sbin
```

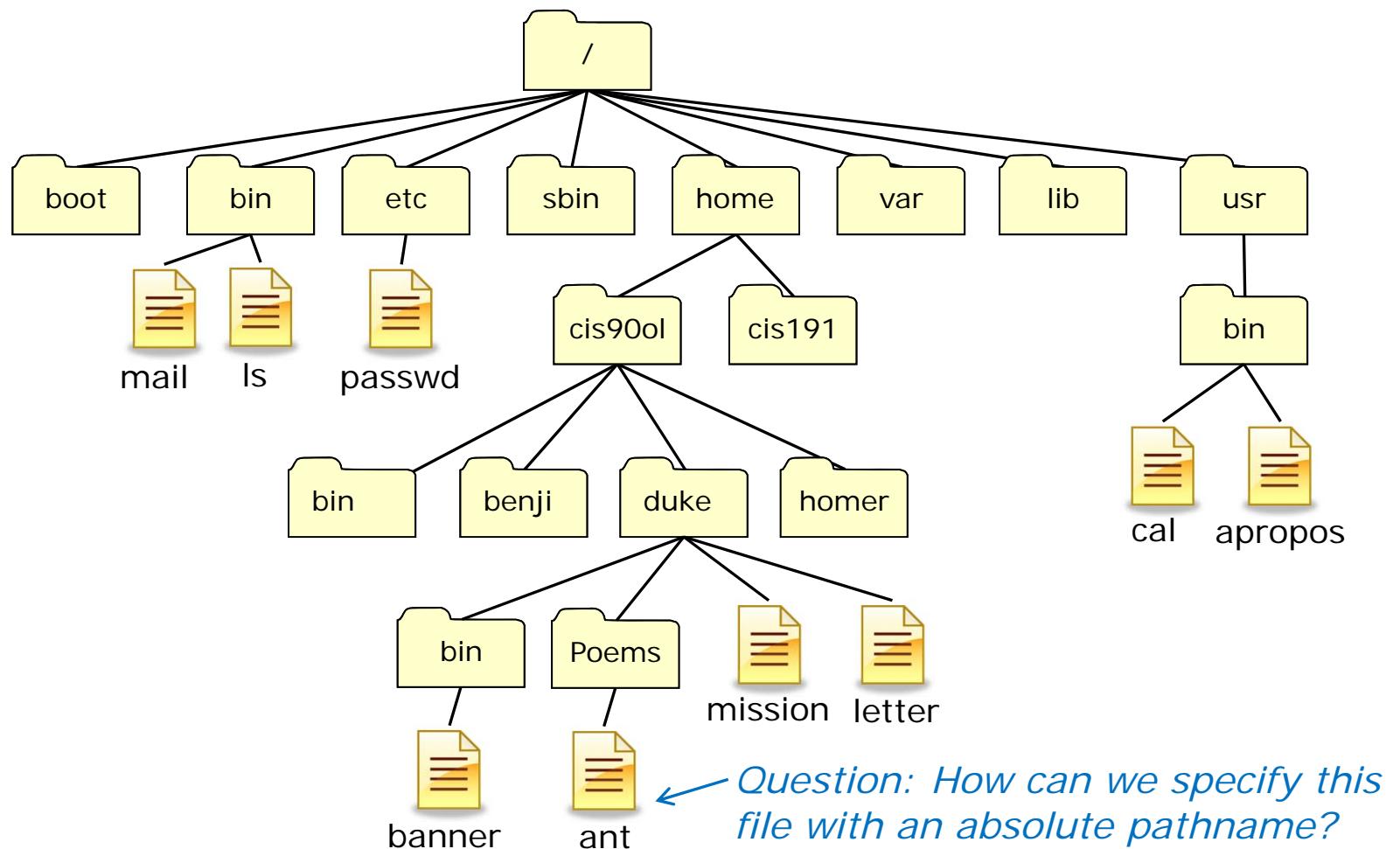
```
file /usr/bin/cal
```

```
cd /home/cis90ol/Poems/Shakespeare
```

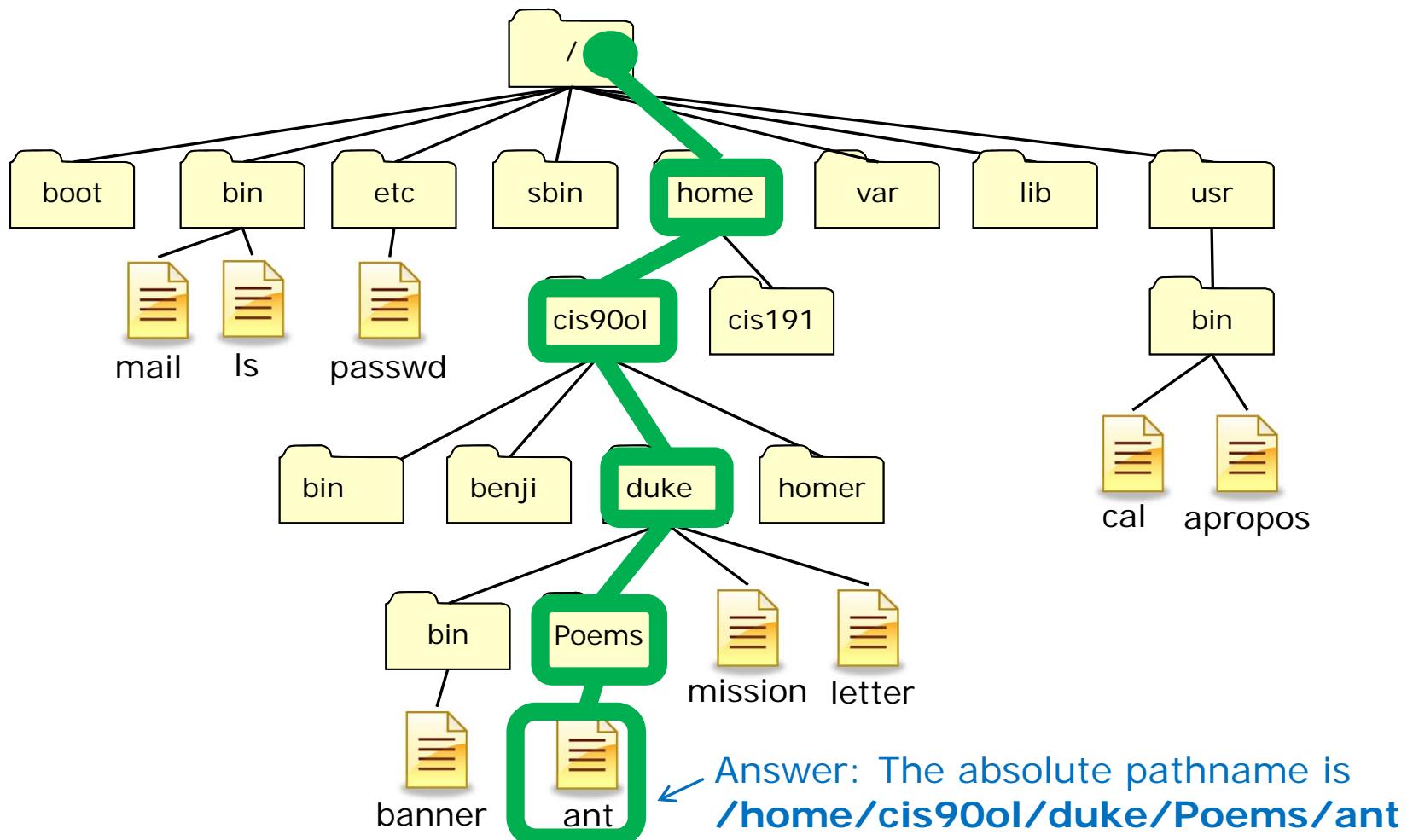
```
ls -l /bin/mail
```

```
cp /etc/passwd /home/cis90ol/simmsben/misc
```

An **absolute pathname** specifies the path from the top of the tree to the target directory or file

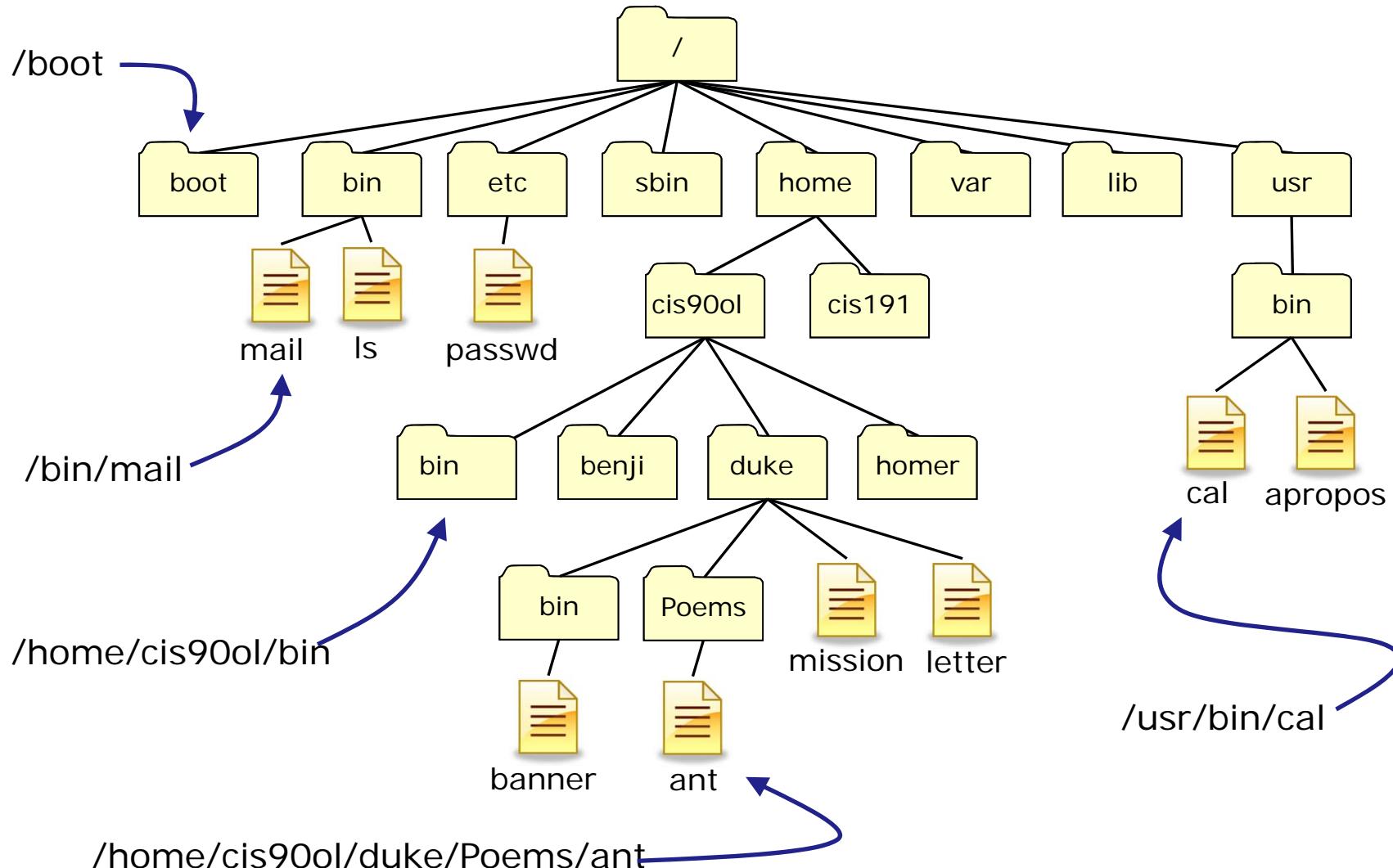


An **absolute pathname** specifies the path from the top of the tree to the target directory or file



# Absolute Pathnames

These are all absolute pathname examples



# Relative Pathnames

A **relative pathname** specifies the path from your current location to the target directory or file.

Examples:

ant (file)

Poems/Shakespeare/sonnet5 (file)

../mission (file)

../bin/ (directory)

../../../../boot/vmlinuz-2.6.18-164.el5 (file)

# Relative Pathnames

Using relative pathnames as command arguments

*A **relative pathname** specifies the path from your current location to the target directory or file.*

Examples of using relative pathnames as command arguments:

ls -l ant

file ../../../../../../bin/mail

cd Poems/Blake

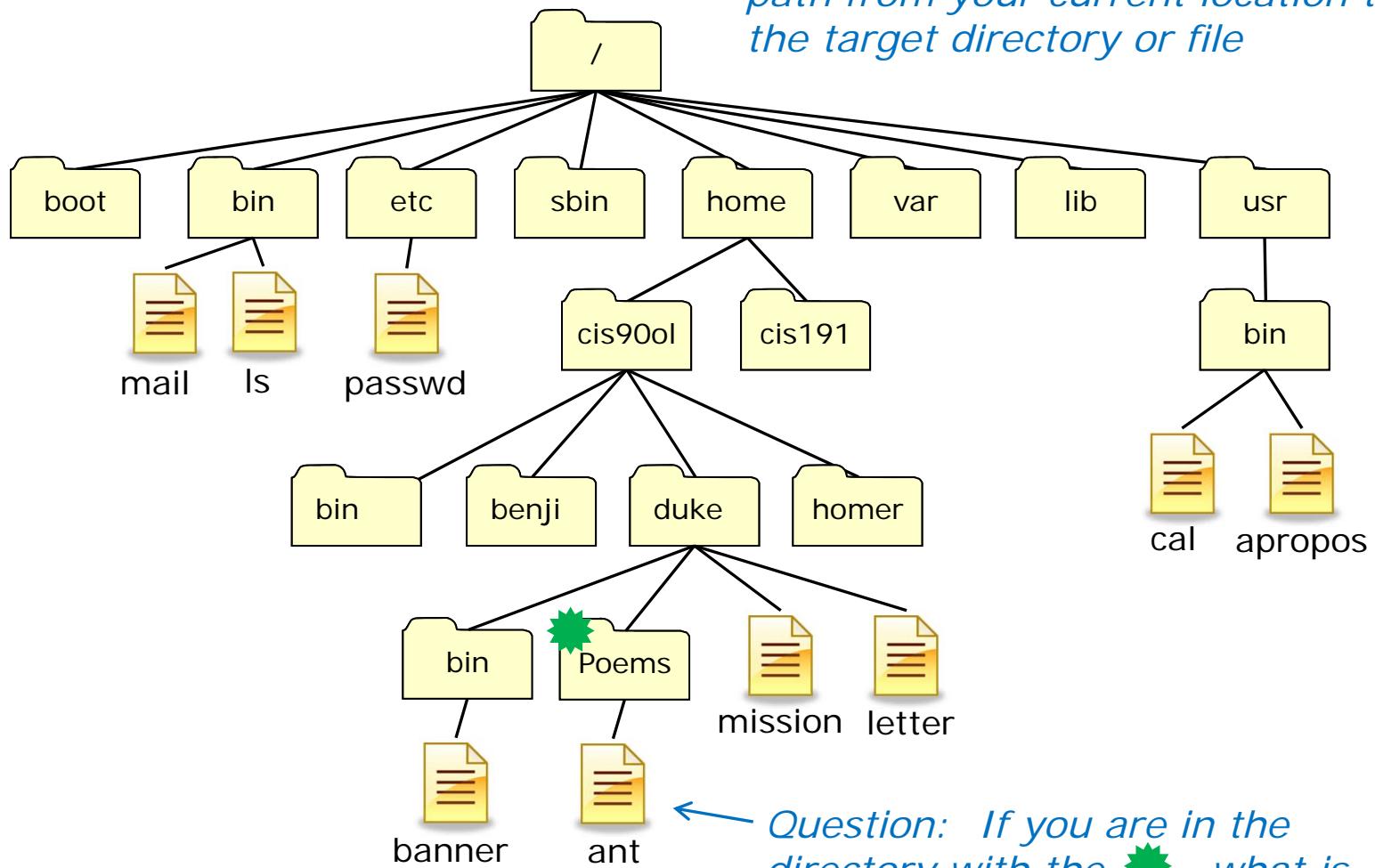
ls -l ../../bin/check3

file Poems/Shakespeare/sonnet4

cd Poems/Shakespeare

## Relative Pathname Example

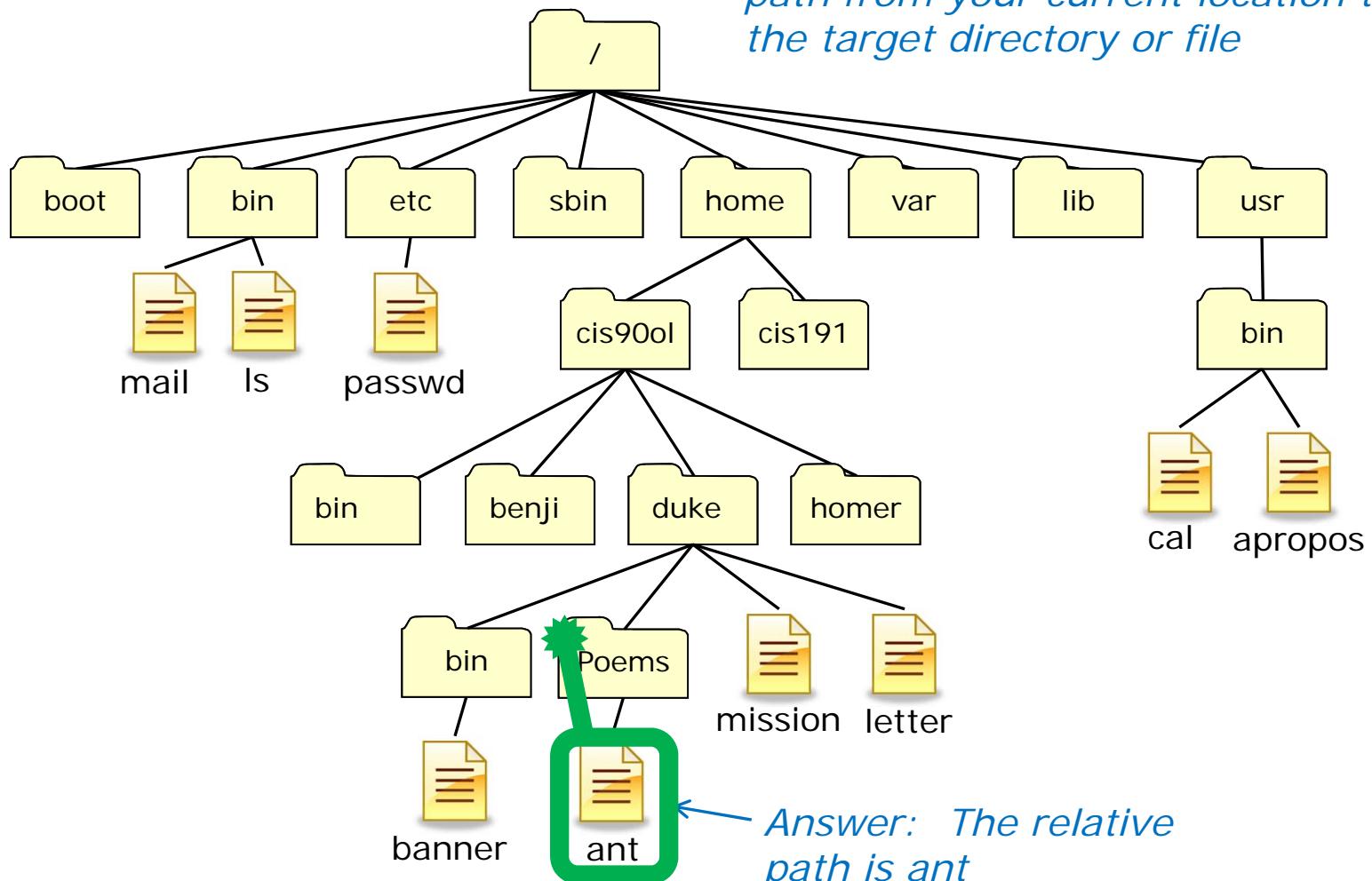
A **relative pathname** specifies the path from your current location to the target directory or file



Question: If you are in the directory with the , what is the relative path to this file?

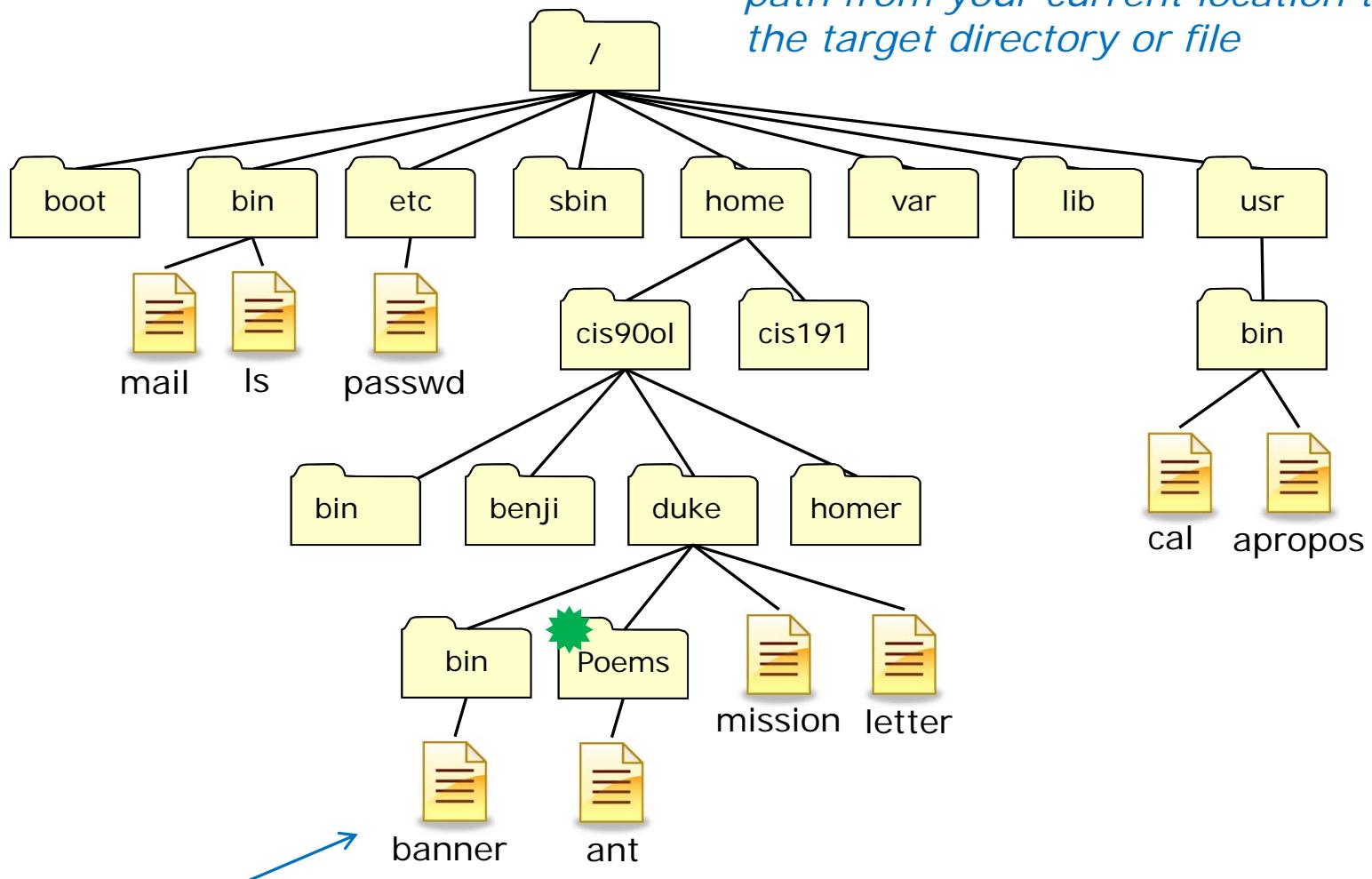
## Relative Pathname Example

A **relative pathname** specifies the path from your current location to the target directory or file



## Relative Pathname Example

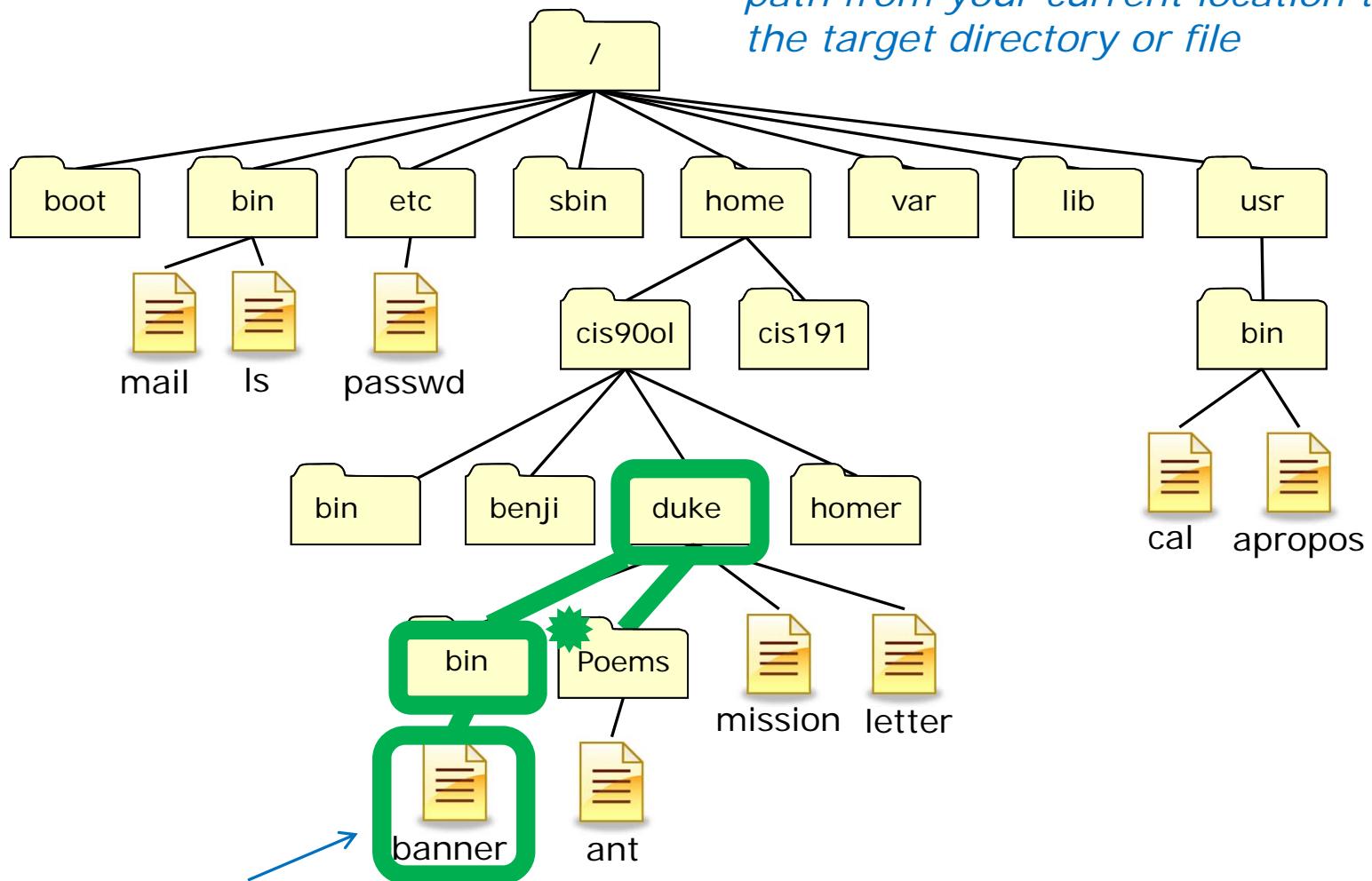
A **relative pathname** specifies the path from your current location to the target directory or file



Question: If you are in the directory with the  , what is the relative path to this file?

## Relative Pathname Example

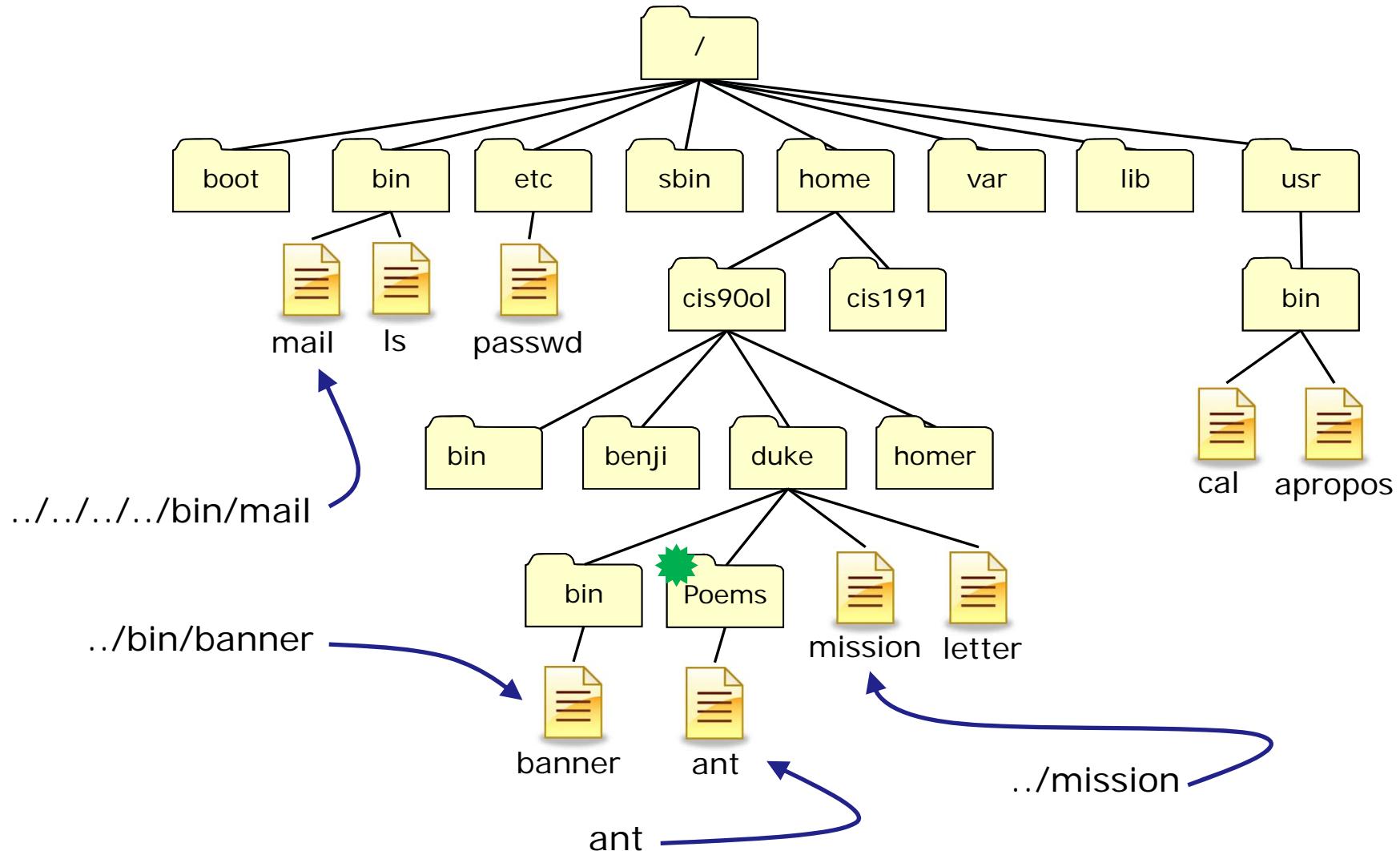
A **relative pathname** specifies the path from your current location to the target directory or file



Answer: The relative path to this file is ../bin/banner

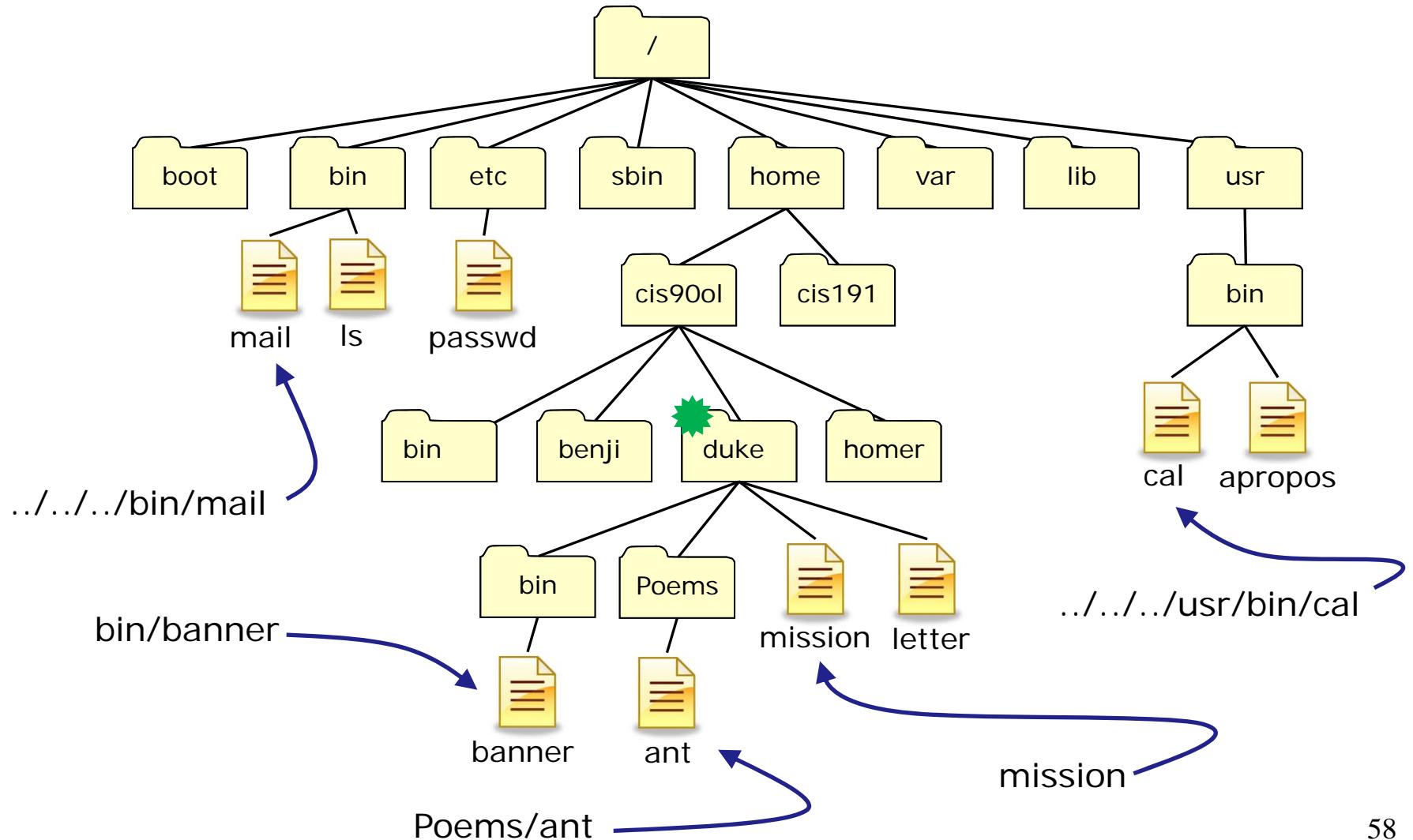
# Relative Pathnames

Names that start relative to the current working directory (★)



# Relative Pathnames

Names that start relative to the current working directory (★)



## Class Exercise

From your home directory:

- List the /etc/passwd/ file using a relative pathname

**ls ../../etc/passwd**

- List the /etc/passwd file using a absolute pathname

**ls /etc/passwd**

- List the letter file using a relative pathname

**ls letter**

- List the letter file using an absolute pathname

**ls /home/cis9001/simmsben/letter**



*user your home directory instead*

## Heads up on a future test question

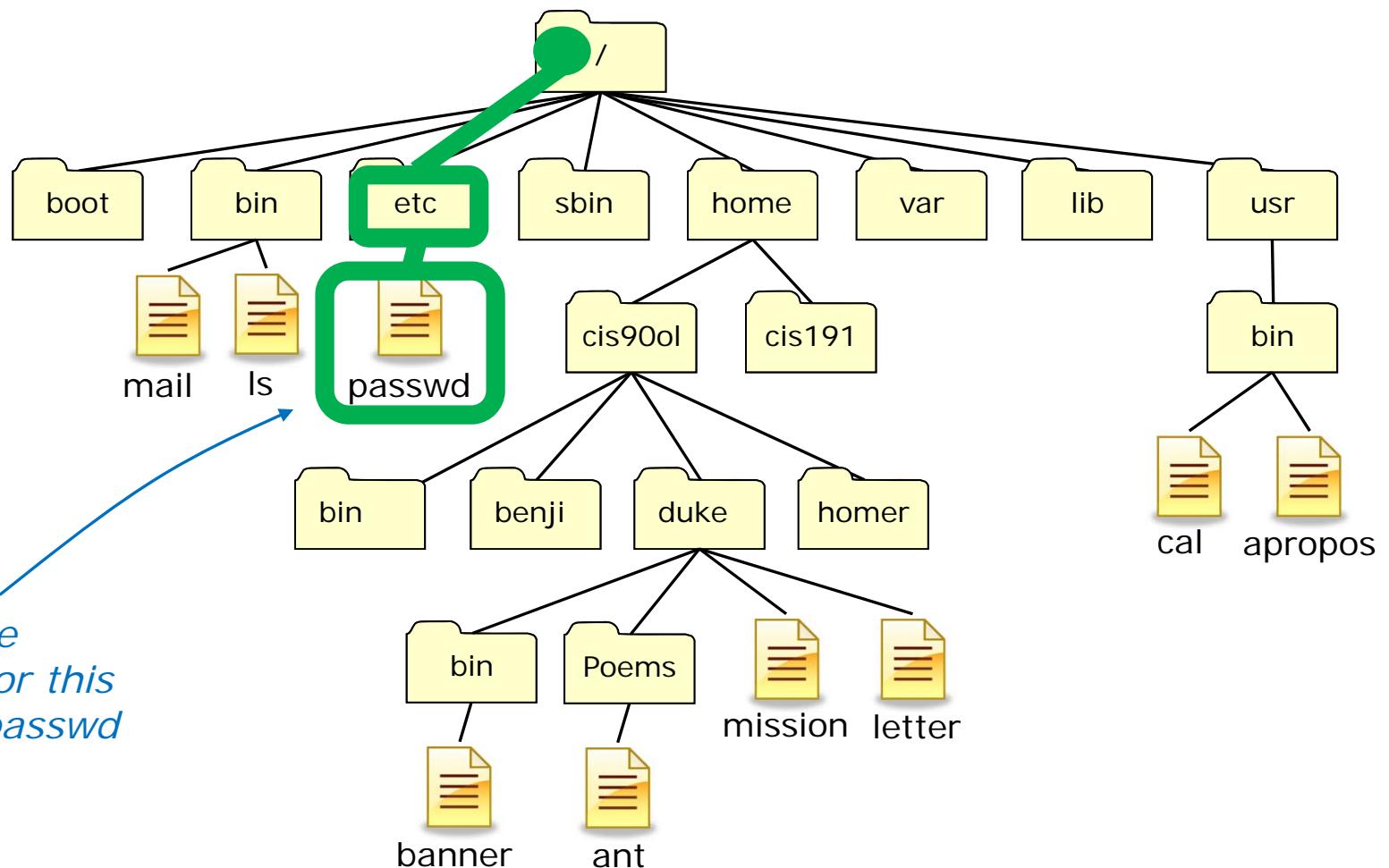
Question: What is the absolute pathname of /etc/passwd?

Answer: /etc/passwd

*What is the color of Washington's white horse?*

# UNIX File Tree

/ = root of the tree



.

/

and

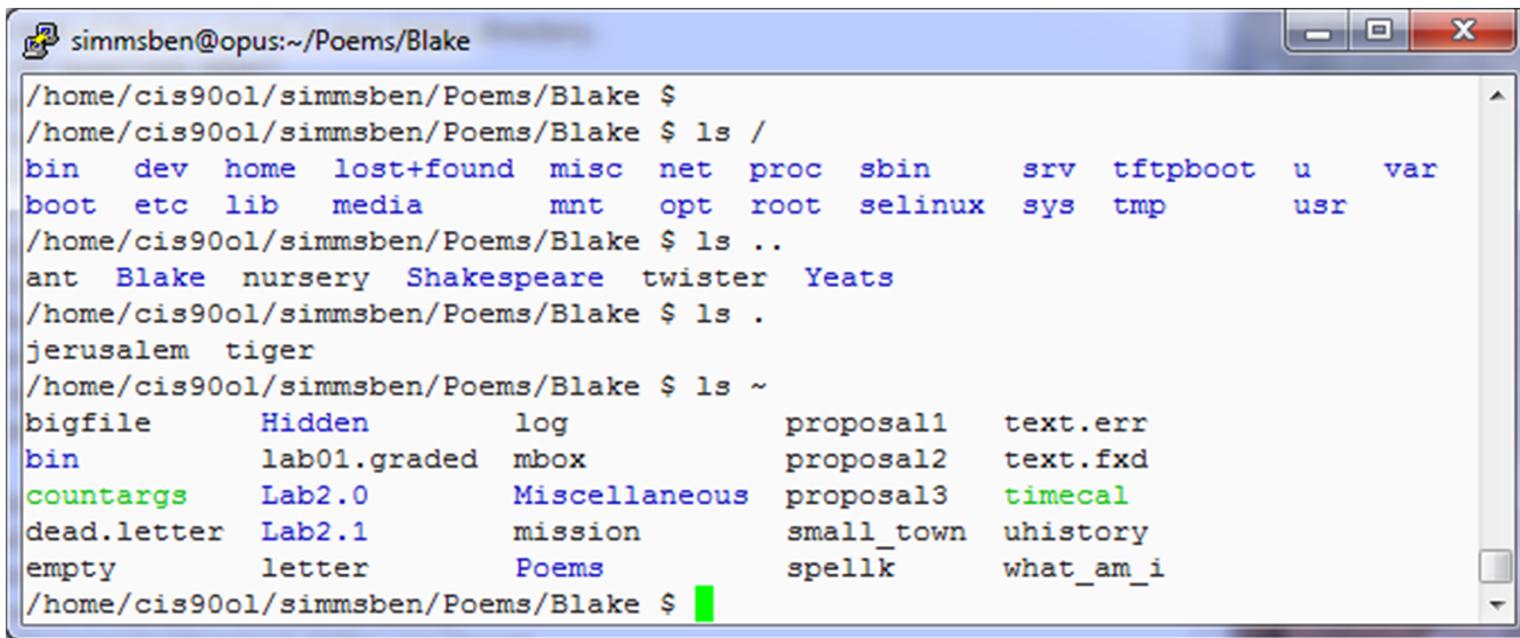
.

.

~

# More on Directories

- / is always used for the root directory of the tree
- .. is shorthand for the current parent directory
- . is shorthand for the absolute path to your current directory -- "here"
- ~ is shorthand for the absolute path to your home directory



A screenshot of a terminal window titled "simmsben@opus:~/Poems/Blake". The window shows the user's command-line session:

```
/home/cis90ol/simmsben/Poems/Blake $  
/home/cis90ol/simmsben/Poems/Blake $ ls /  
bin dev home lost+found misc net proc sbin srv tftpboot u var  
boot etc lib media mnt opt root selinux sys tmp usr  
/home/cis90ol/simmsben/Poems/Blake $ ls ..  
ant Blake nursery Shakespeare twister Yeats  
/home/cis90ol/simmsben/Poems/Blake $ ls .  
jerusalem tiger  
/home/cis90ol/simmsben/Poems/Blake $ ls ~  
bigfile Hidden log proposal1 text.err  
bin lab01.graded mbox proposal2 text.fxd  
countargs Lab2.0 Miscellaneous proposal3 timecal  
dead.letter Lab2.1 mission small_town uhistory  
empty letter Poems spellk what_am_i  
/home/cis90ol/simmsben/Poems/Blake $
```

*. and .. are hidden files, more on hidden files later ...*

# UNIX File Hierarchy

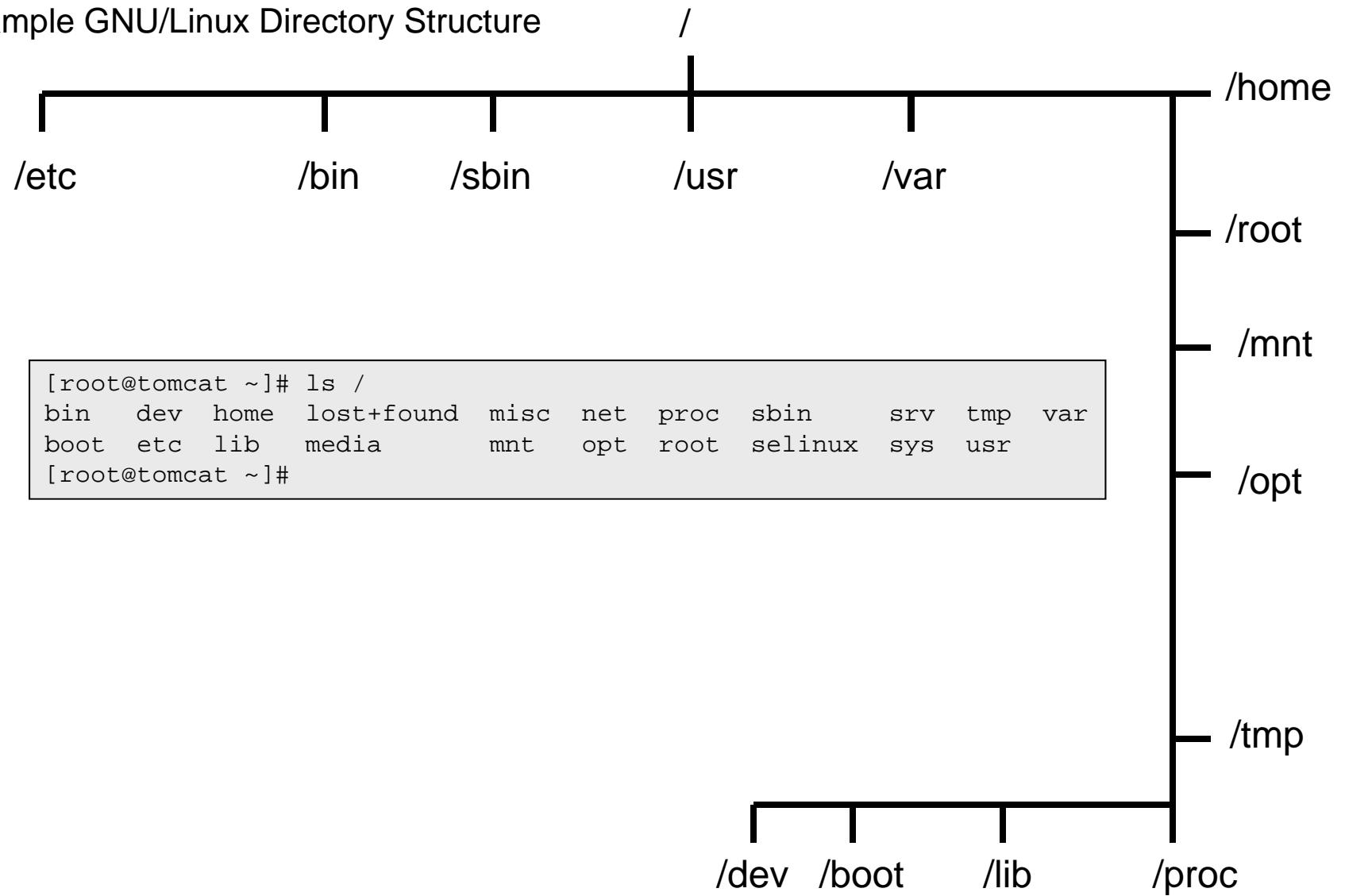
/
/bin
/boot
/dev
/etc
/home
/lib
/lost+found
/mnt
/opt
/proc
/root
/sbin
/tmp
/usr

## The UNIX/Linux File System Hierarchy

*There are standard top level  
directories in every version of  
UNIX/Linux*

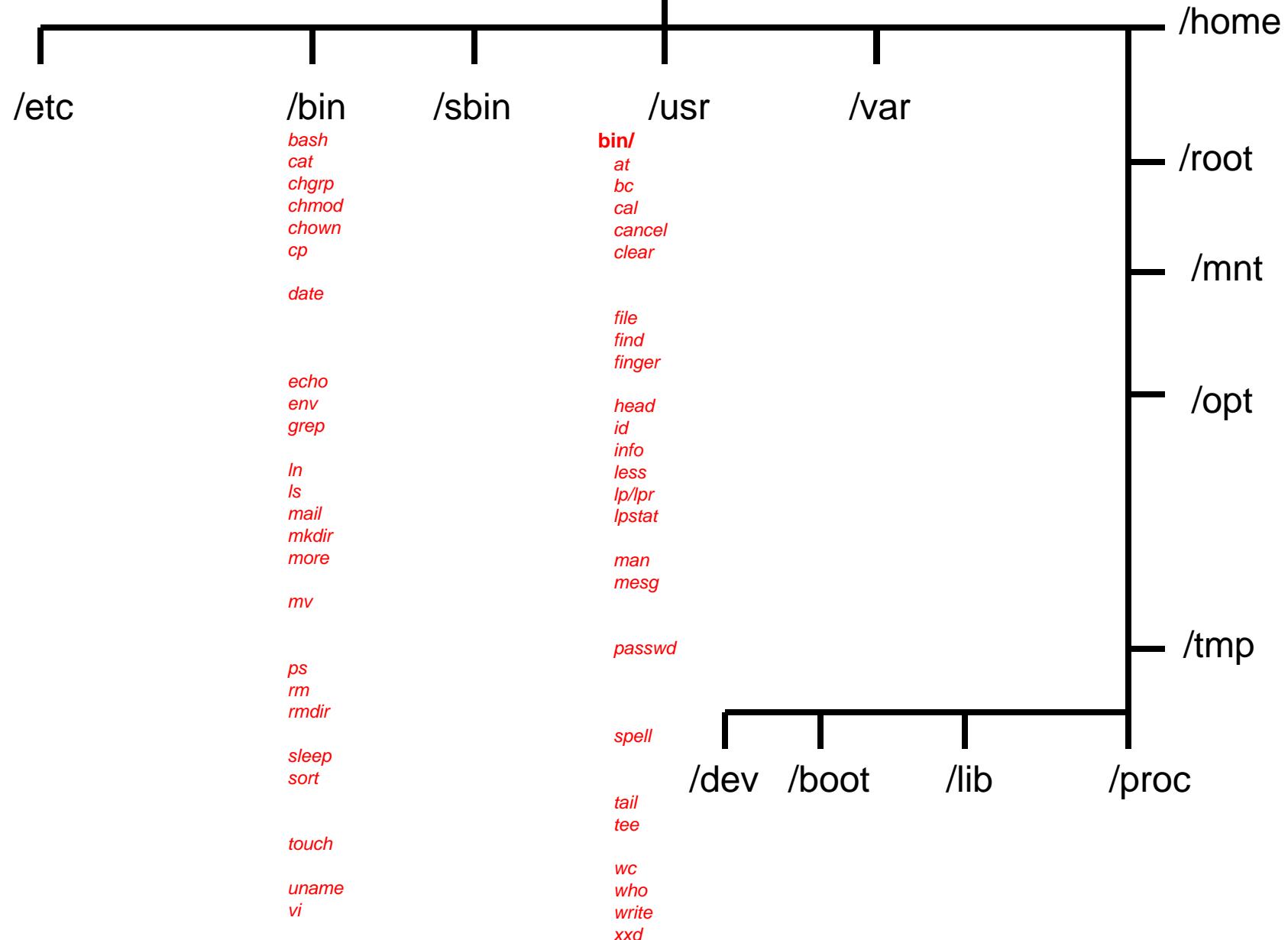
Directory	Contents
/bin	binary files forming the commands and shells used by the system administrator and users
/boot	files used during the initial boot-up process including the kernel
/dev	device files for connected hardware
/etc	system configuration files
/home	individual directories owned by each user
/lib	shared libraries needed to boot the system and run the commands in the root filesystem (i.e. commands in /bin and /sbin)
/lost+found	recovered files that were corrupted by power failures or system crashes
/mnt	mount points for floppies, cds, or other file systems
/opt	add-on software packages and/or commercial applications
/proc	kernel level process information
/root	home directory for the root user
/sbin	system administration commands reserved for the superuser (root)
/tmp	temporary files that are deleted when the system is rebooted or started
/usr	program files and related files for use by all users
/var	log files, print spool files, and mail queues

## Example GNU/Linux Directory Structure



## Example GNU/Linux Directory Structure

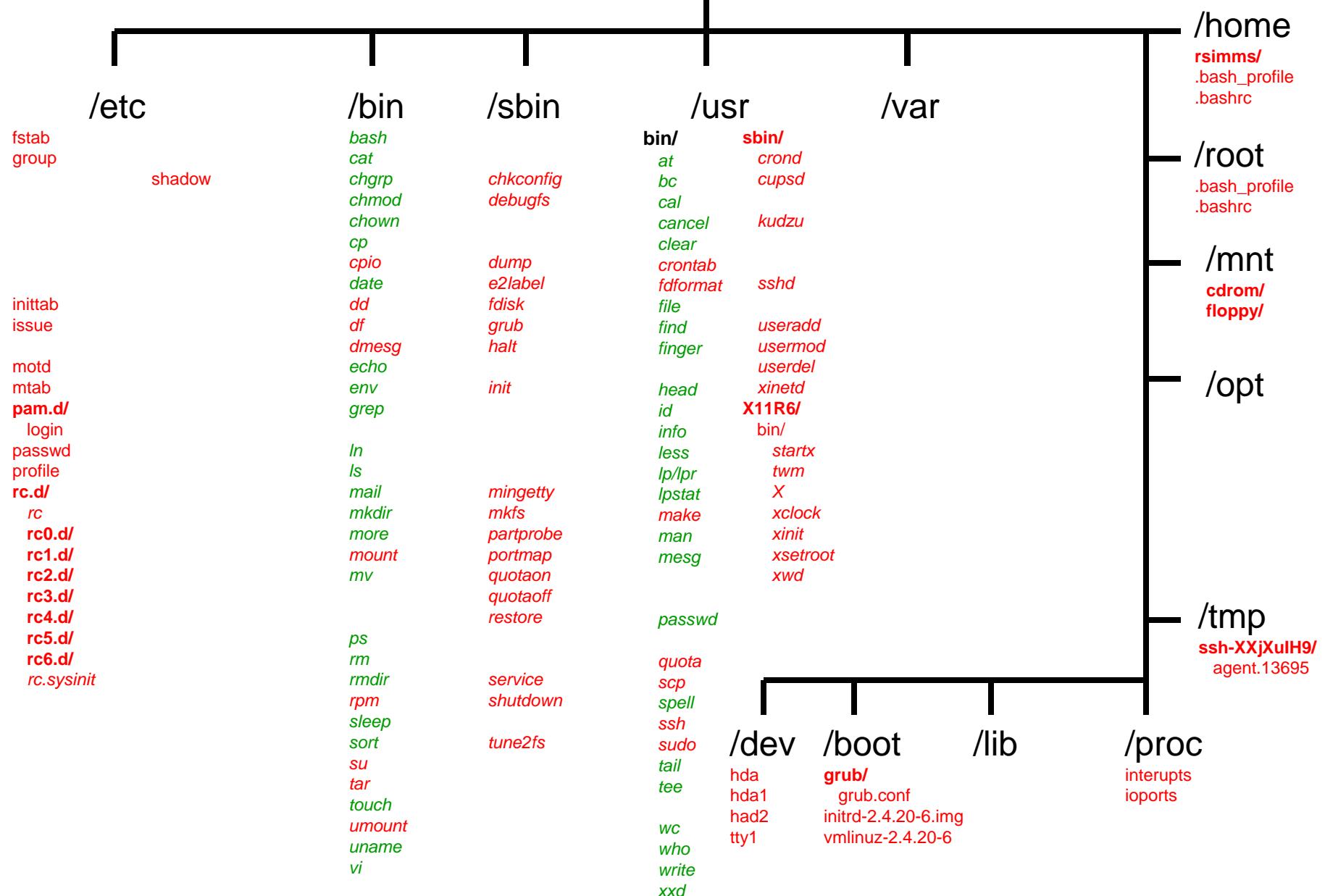
## CIS 90 files, directories, commands



Note: shell builtins = cd, echo, exit, export, history, jobs, kill, pwd, set, type, umask, unset

## Example GNU/Linux Directory Structure

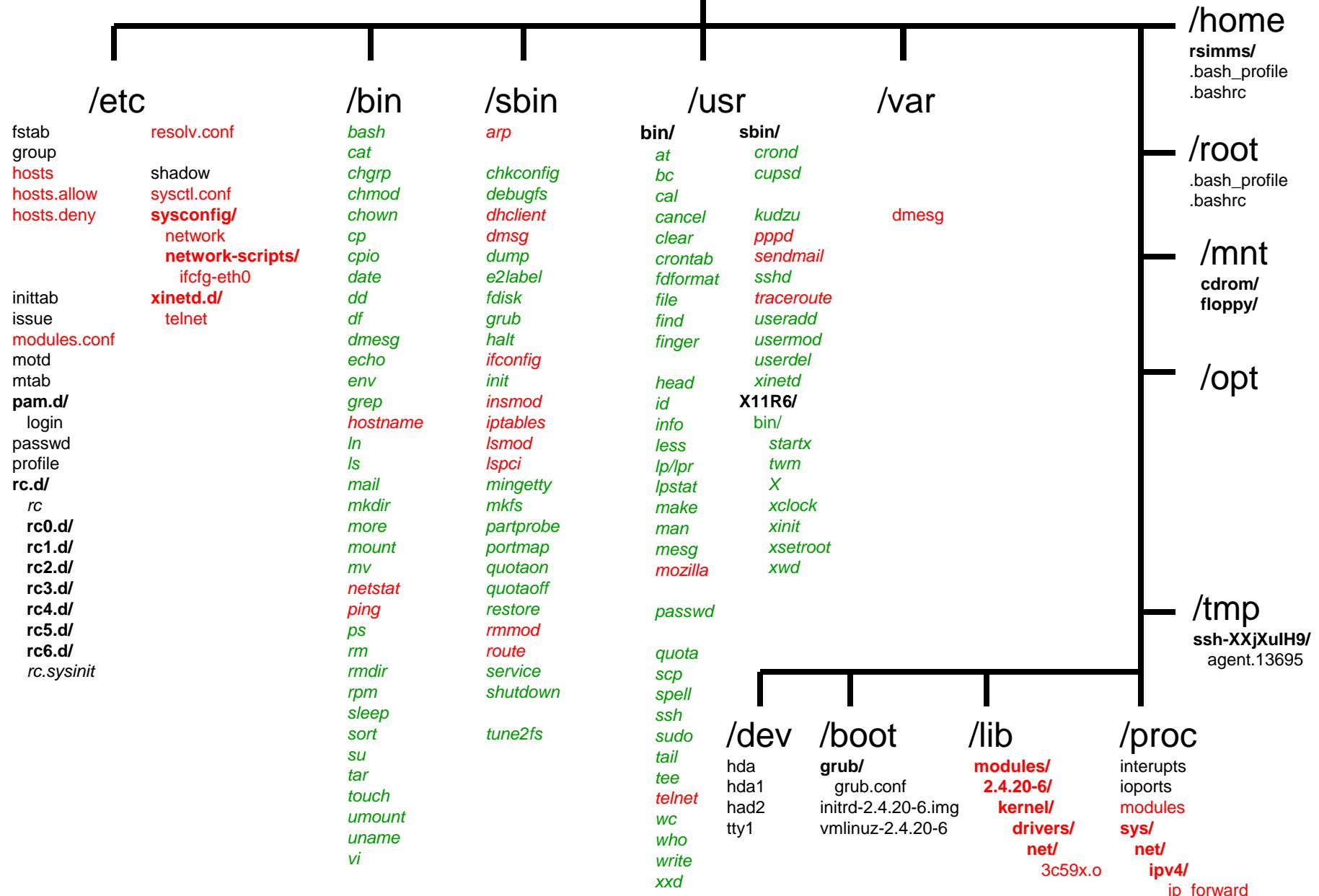
CIS 191 files, directories, commands



Note: shell builtins = cd, echo, exit, export, history, jobs, kill, pwd, set, type, umask, unset

## Example GNU/Linux Directory Structure

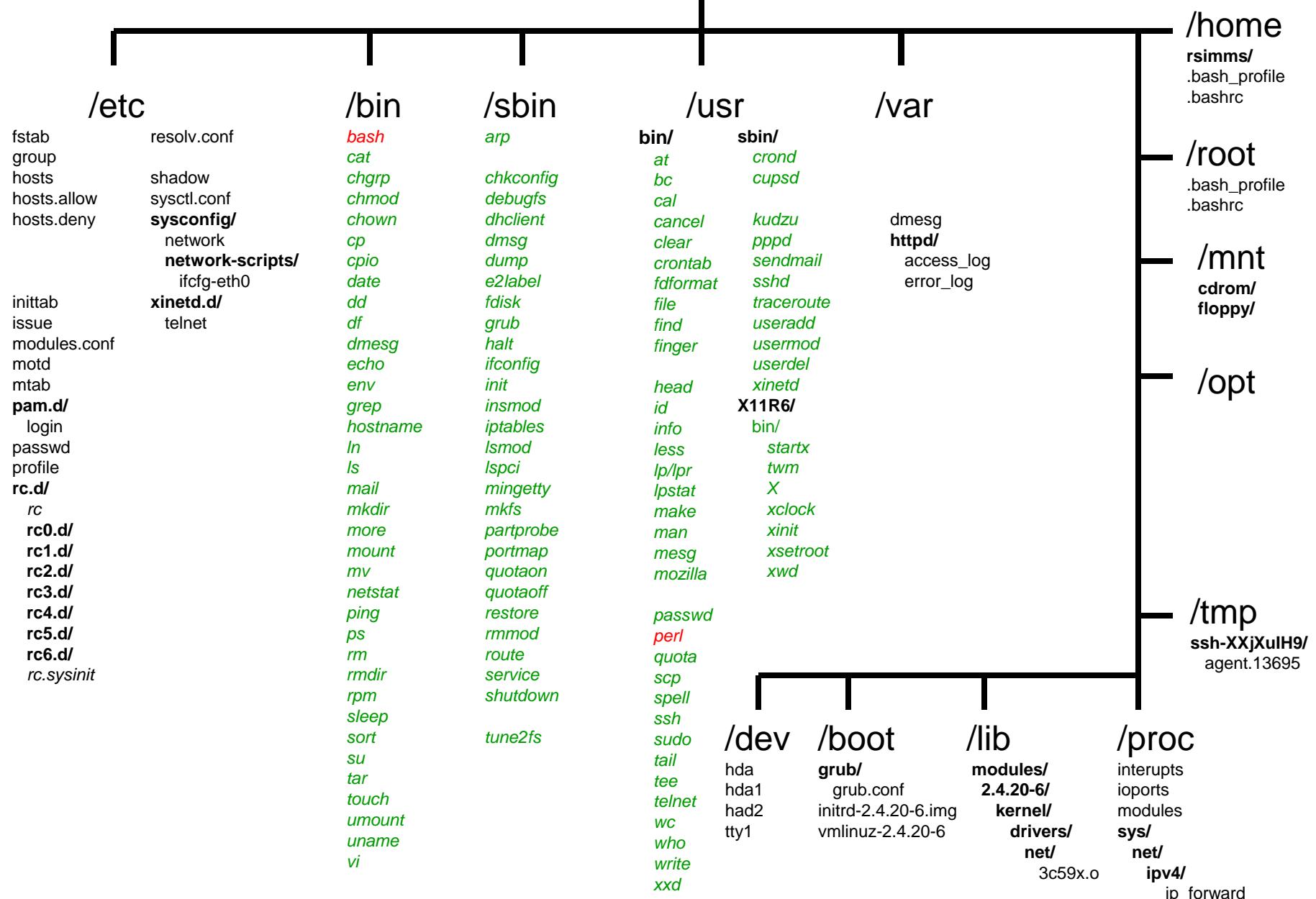
CIS 192 files, directories, commands



Note: shell builtins = cd, echo, exit, export, history, jobs, kill, pwd, set, type, umask, unset

## Example GNU/Linux Directory Structure

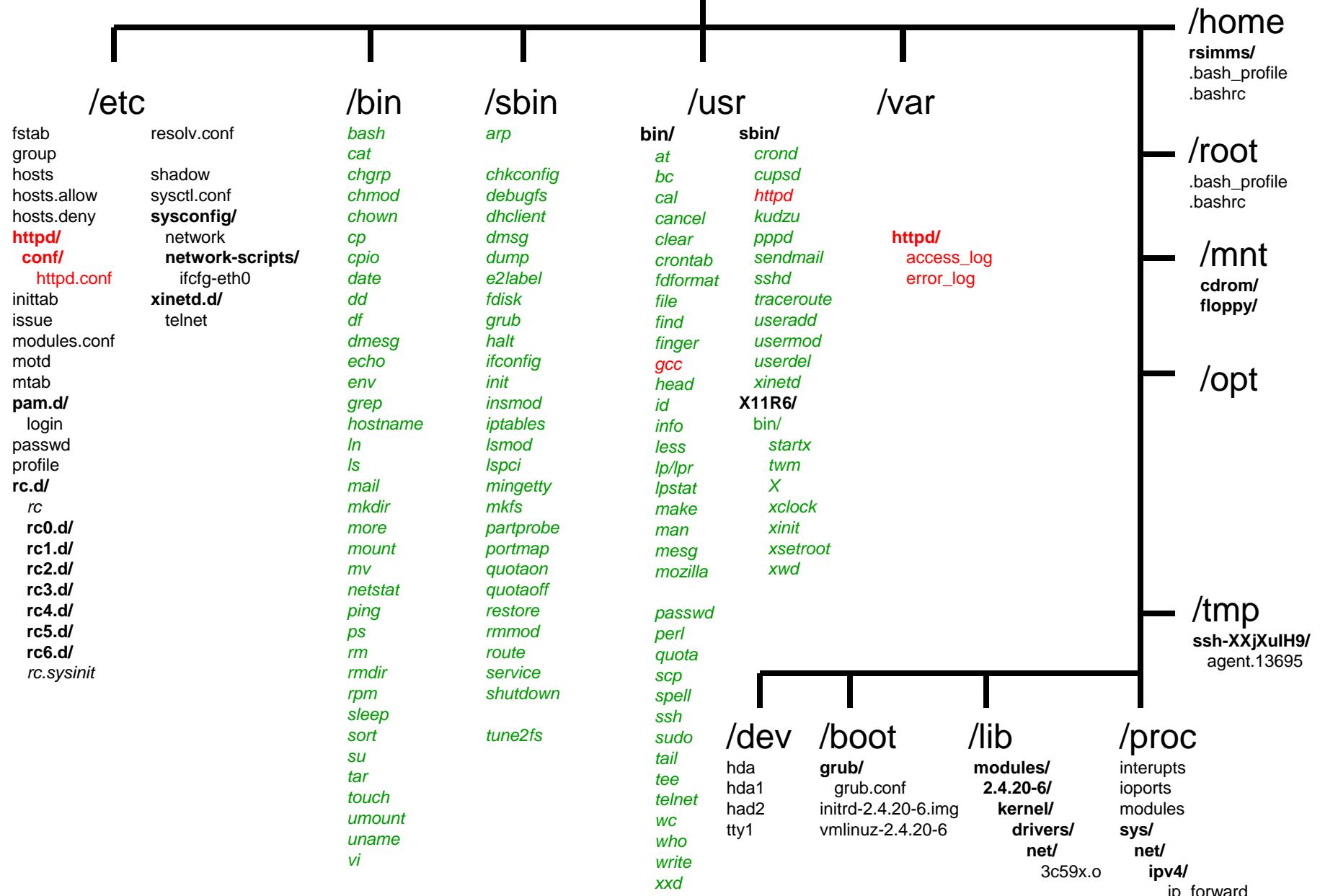
## CIS 130 files, directories, commands



Note: shell builtins = `cd`, `echo`, `exit`, `export`, `history`, `jobs`, `kill`, `pwd`, `set`, `type`, `umask`, `unset` shell keywords = `if`, `then`, `else`, `case`, `for`, `while`

## Example GNU/Linux Directory Structure

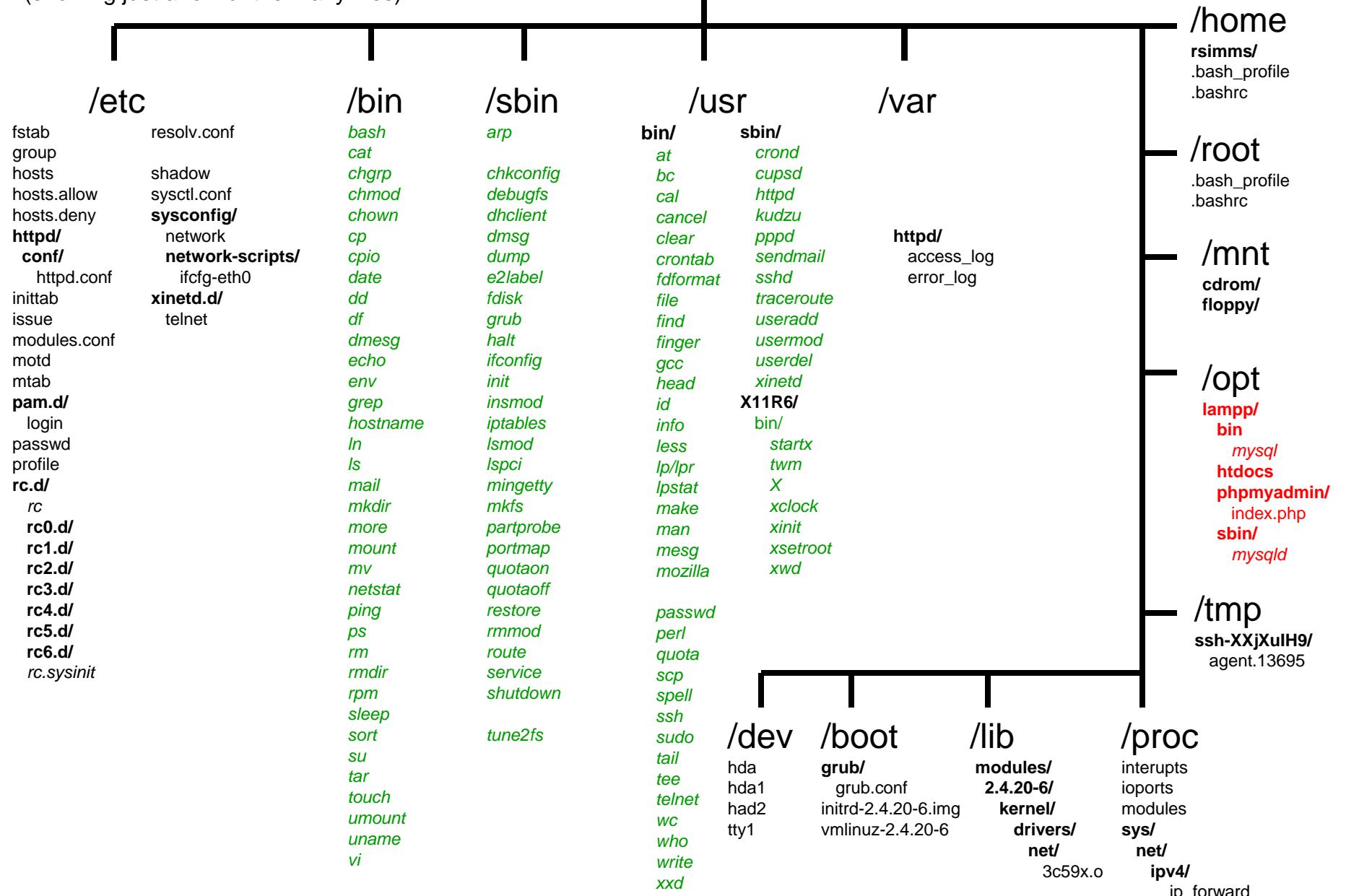
## CIS 164 files, directories, commands



Note: shell builtins = cd, echo, exit, export, history, jobs, kill, pwd, set, type, umask, unset shell keywords = if, then, else, case, for, while

## Example GNU/Linux Directory Structure (showing just a few of the many files)

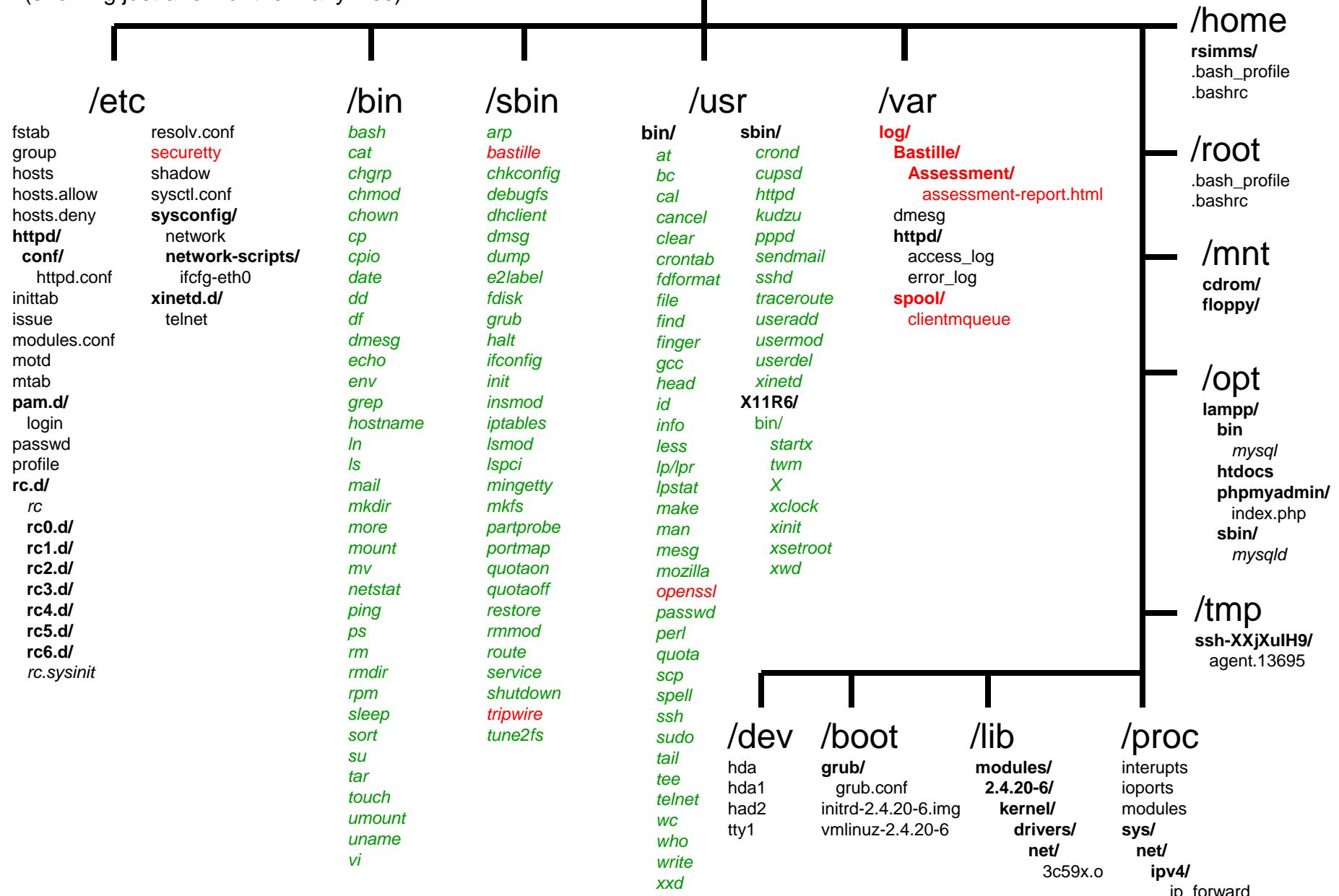
## CIS 165PH files, directories, commands



Note: shell builtins = `cd`, `echo`, `exit`, `export`, `history`, `jobs`, `kill`, `pwd`, `set`, `type`, `umask`, `unset` shell keywords = `if`, `then`, `else`, `case`, `for`, `while`

## Example GNU/Linux Directory Structure (showing just a few of the many files)

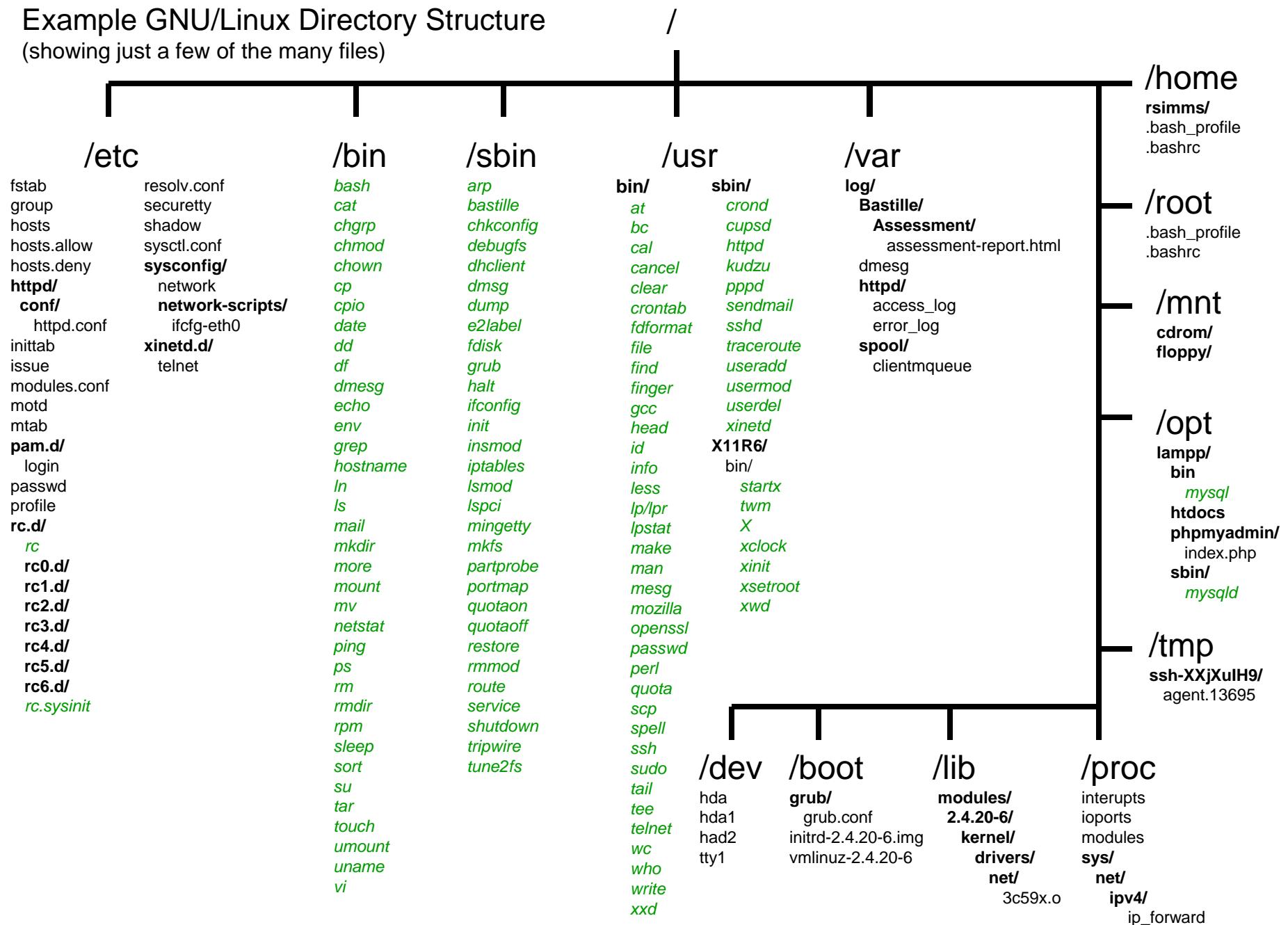
CIS 193 files, directories, commands



Note: shell builtins = `cd`, `echo`, `exit`, `export`, `history`, `jobs`, `kill`, `pwd`, `set`, `type`, `umask`, `unset` shell keywords = `if`, `then`, `else`, `case`, `for`, `while`

# Example GNU/Linux Directory Structure

(showing just a few of the many files)



Note: shell builtins = cd, echo, exit, export, history, jobs, kill, pwd, set, type, umask, unset shell keywords = if, then, else, case, for, while

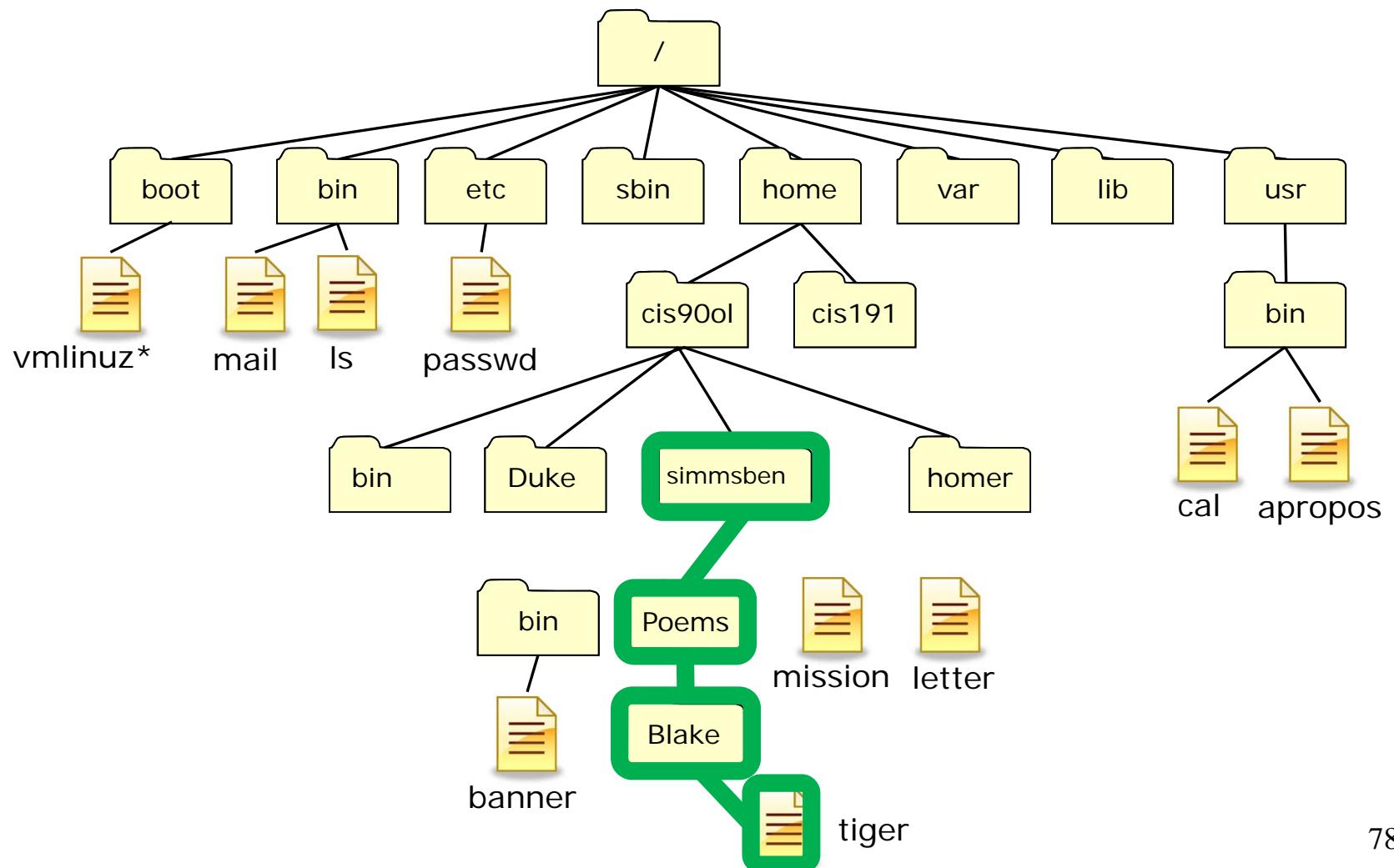
# Navigating the UNIX file tree

# Navigating the tree

- Use the **cd** command to change directories  
*(your legs)*
- Use the **ls** command to list files at your current location  
*(your eyes)*
- Use the **pwd** command to check where you are  
*(your GPS)*

*Note, as CIS 90 students your command prompt has been configured to show what you would normally get with the **pwd** command. As you move around the tree your command prompt will change to show your current location.*

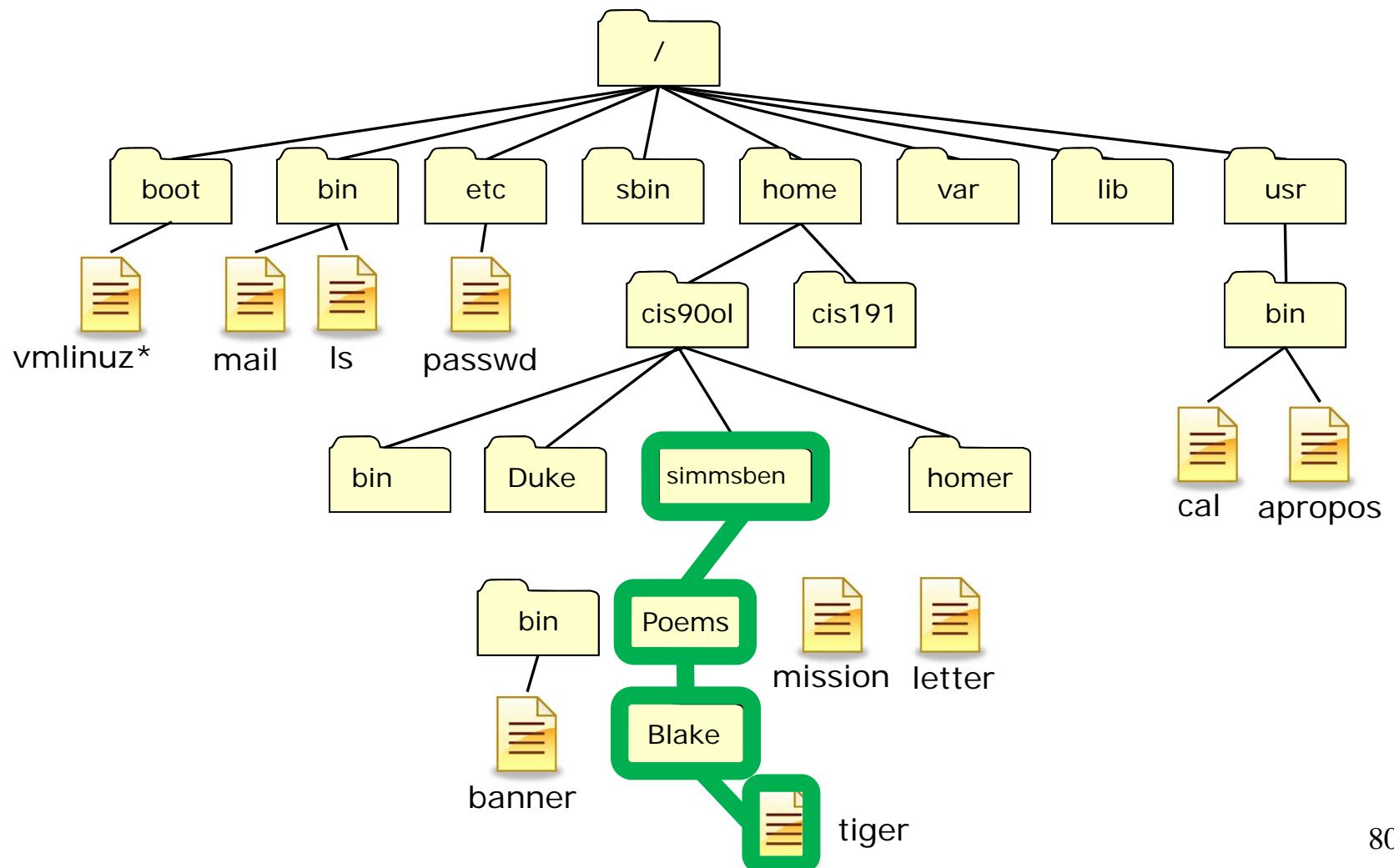
*How do we walk the tree from our home directory to the directory containing the tiger file and print it?*



## Class Exercise

```
/home/cis90ol/simmsben $ cd      start in our home directory
/home/cis90ol/simmsben $ ls      see what's there
bigfile      Hidden          log           proposal1    text.err
bin          lab01.graded   mbox          proposal2    text.fxd
countargs    Lab2.0          Miscellaneous proposal3    timecal
dead.letter   Lab2.1          mission       small_town  uhistory
empty        letter          Poems         spellk      what_am_i
/home/cis90ol/simmsben $ cd Poems/  go down into Poems directory
/home/cis90ol/simmsben/Poems $ ls      see what's there
ant  Blake  nursery  Shakespeare  twister  Yeats
/home/cis90ol/simmsben/Poems $ cd Blake/  go down into Blake directory
/home/cis90ol/simmsben/Poems/Blake $ ls      see what's there
jerusalem  tiger
/home/cis90ol/simmsben/Poems/Blake $ cat tiger  print tiger file
Tiger, Tiger burning bright
In the forest of the night,
What immortal hand or eye
Dare frame thy fearful symmetry?
```

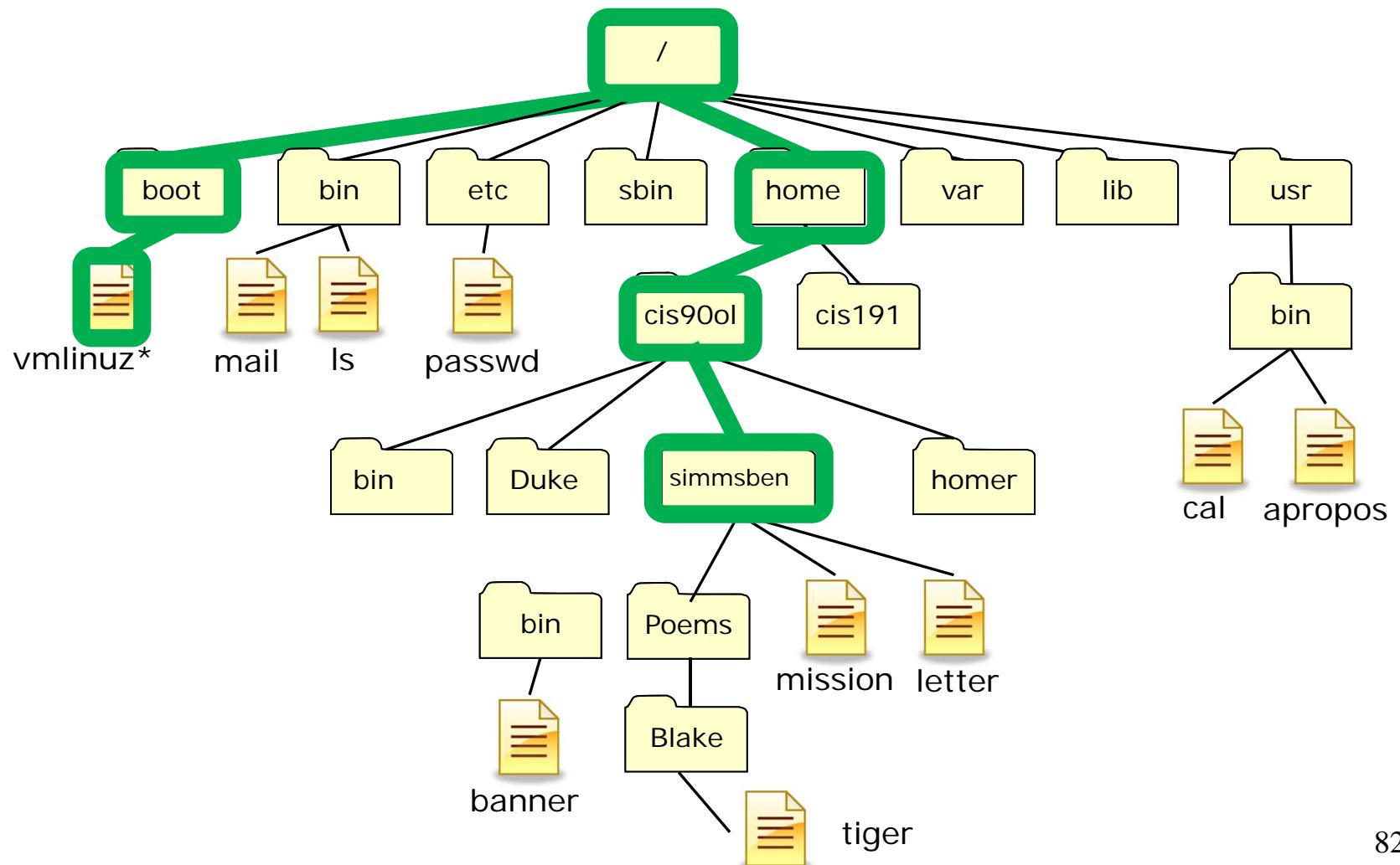
*Alternatively how could we print the tiger file from our home directory without navigating there first?*



### Class Exercise

```
/home/cis90ol/simmsben $ cd  
/home/cis90ol/simmsben $ cat Poems/Blake/tiger    using a relative pathname  
Tiger, Tiger burning bright  
In the forest of the night,  
What immortal hand or eye  
Dare frame thy fearful symmetry?  
/home/cis90ol/simmsben $
```

*How do we walk the tree from our home directory to the directory containing the Linux kernel and do a long listing of it?*



## Class Exercise

```
/home/cis90ol/simmsben/Poems/Blake $ cd      start in your home directory
```

```
/home/cis90ol/simmsben $ cd ..    go up
```

```
/home/cis90ol $ ls    look around
```

answers	christan	dienequi	hillejef	lighttom	paytomar	sylvijos	willitaj	
bin	cis90	elmenchr	hwangyuc	lynbeeri	roddyduk	vieyrleo	wilsodan	
carvaema	clarkric	herodbob	keezeter	mcnamdan	simmsben	vistigab	wingejas	
cheeken	depot	hextcra	lewisgre	montageo	stumbdav	warrejes		

```
/home/cis90ol $ cd ..    go up
```

```
/home $ ls    look around
```

backup	cis172	cis192	cis90	gerlinde	guest191	mikki	ryan	
cis130	cis191	cis193	cis90ol	guest	jimg	rsimms	turnin	

```
/home $ cd ..    go up
```

```
/ $ ls    look around
```

bin	dev	home	lost+found	misc	net	proc	sbin	srv	tftpboot	u	var
boot	etc	lib	media		mnt	opt	root	selinux	sys	tmp	usr

```
/ $ cd boot    go down into boot
```

```
/boot $ ls    look around
```

config-2.6.18-164.el5	lost+found	vmlinuz-2.6.18-164.el5
grub	symvers-2.6.18-164.el5.gz	
initrd-2.6.18-164.el5.img	System.map-2.6.18-164.el5	

```
/boot $ ls -l vmlinuz-2.6.18-164.el5
```

```
-rw-r--r-- 1 root root 1855956 Aug 18 2009 vmlinuz-2.6.18-164.el5
```

*student home  
directories*

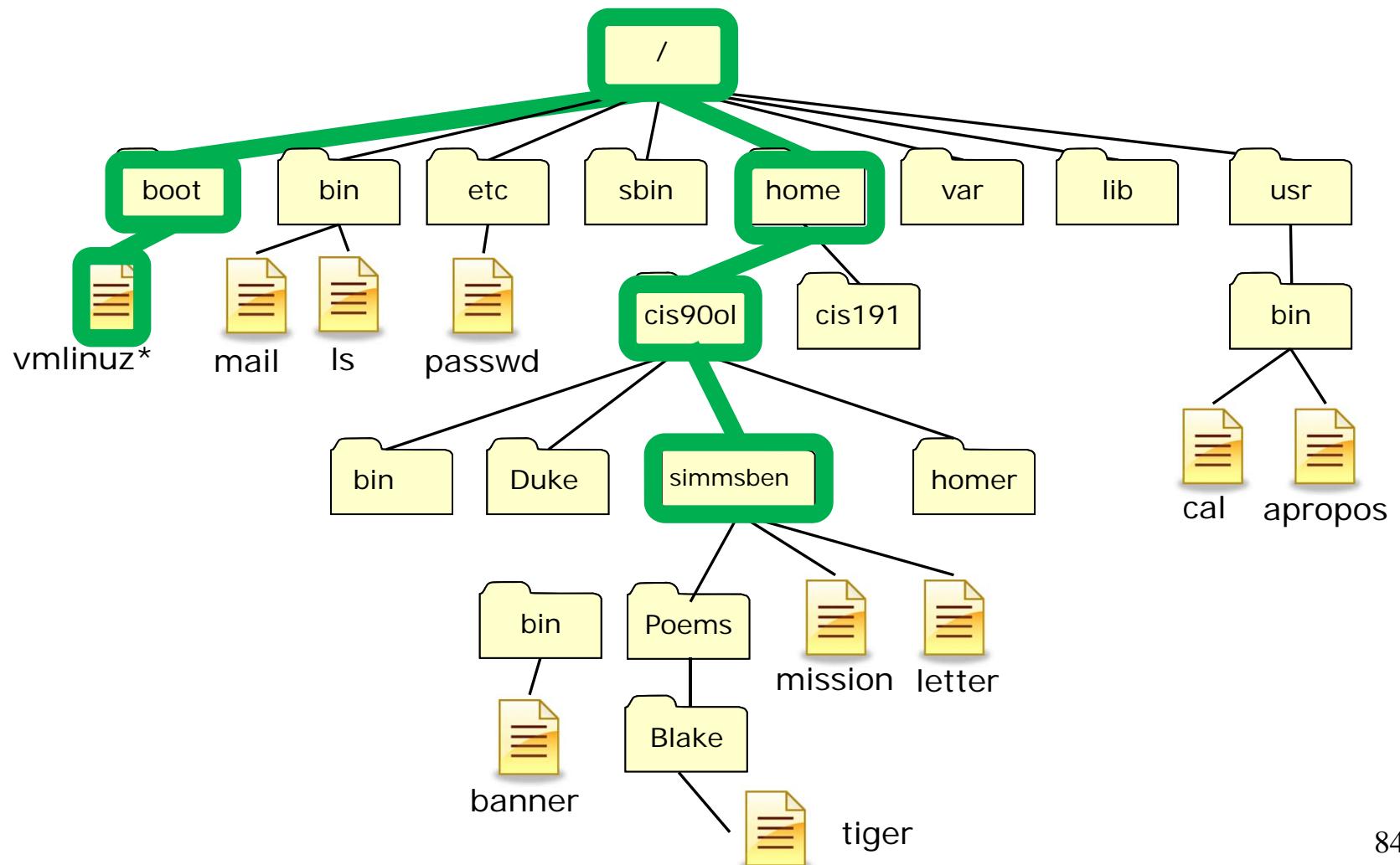
*where labs are  
submitted*

*our class directory*

*my home directory*

*the Linux kernel*

*Alternatively, how could we do the same thing without walking there first?*



## Class Exercise

```
/home/cis90ol/simmsben/Poems/Blake $ cd      start in your home directory
/home/cis90ol/simmsben $ ls -l /boot/vmlinuz-2.6.18-164.el5   using an absolute pathname
-rw-r--r-- 1 root root 1855956 Aug 18  2009 /boot/vmlinuz-2.6.18-164.el5
/home/cis90ol/simmsben $
```



the Linux kernel



# Navigating

## cd command

# cd command

## change directory

- Syntax: **cd [directory]**
- Changes the current working directory to the directory specified.
- Use **cd** with no arguments to return to your home directory.

*Note, users always start in their home directory after logging in.  
Every user's home directory is configured in the /etc/passwd file.*

- The *directory* can be:
  - An absolute pathname, e.g. **cd /home/cis90/duke/Poems/ant**
  - A relative pathname, e.g. **cd Poems**
  - A **..** for the parent of the current working directory, e.g. **cd ..**
- Note, **cd** is a Bash builtin command (part of the shell itself)  
`/home/cis90/simmsben $ type cd`  
`cd is a shell builtin`

## More on .. and .

*Any file that begins with a . is a hidden file.  
There are two hidden files that are quite useful:*

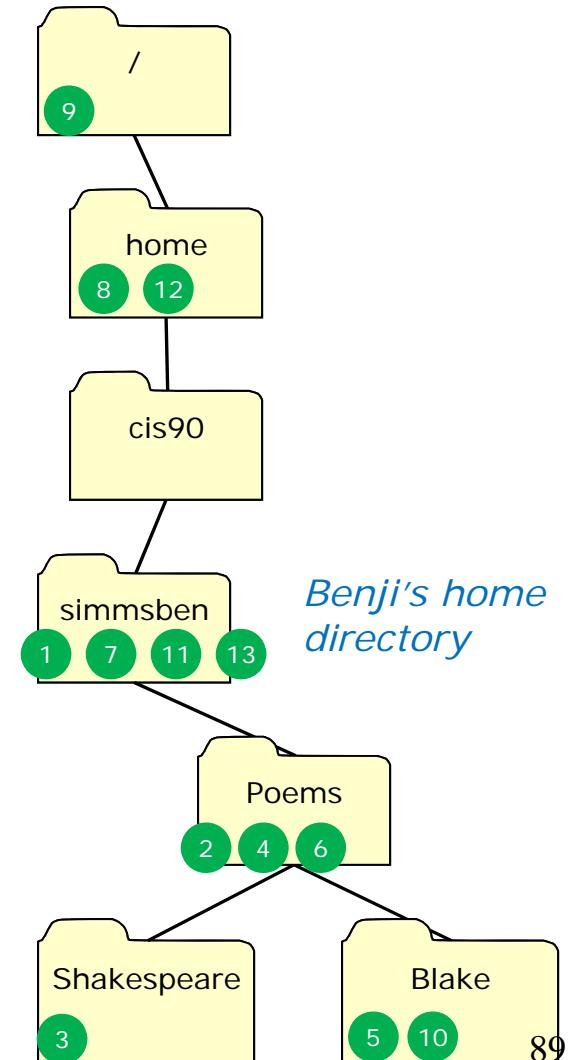
- . . will always refer to the parent directory of the current directory
- . will always refer to “here” or the current directory, it is shorthand for writing out the entire absolute path to your current directory

*More on hidden files later ...*

# cd command

## change directory example

```
/home/cis90/simmsben $ echo $HOME
/home/cis90/simmsben
/home/cis90/simmsben $ echo $PS1
$PWD $
1 /home/cis90/simmsben $ cd Poems/
2 /home/cis90/simmsben/Poems $ cd Shakespeare/
3 /home/cis90/simmsben/Poems/Shakespeare $ cd ..
4 /home/cis90/simmsben/Poems $ cd Blake/
5 /home/cis90/simmsben/Poems/Blake $ cd ..
6 /home/cis90/simmsben/Poems $ cd ..
7 /home/cis90/simmsben $ cd /home
8 /home $ cd ..
9 / $ cd /home/cis90/simmsben/Poems/Blake/
10 /home/cis90/simmsben/Poems/Blake $ cd ..
11 /home/cis90/simmsben $ cd ../../..
12 /home $ cd
13 /home/cis90/simmsben $
```



# Navigating pwd command

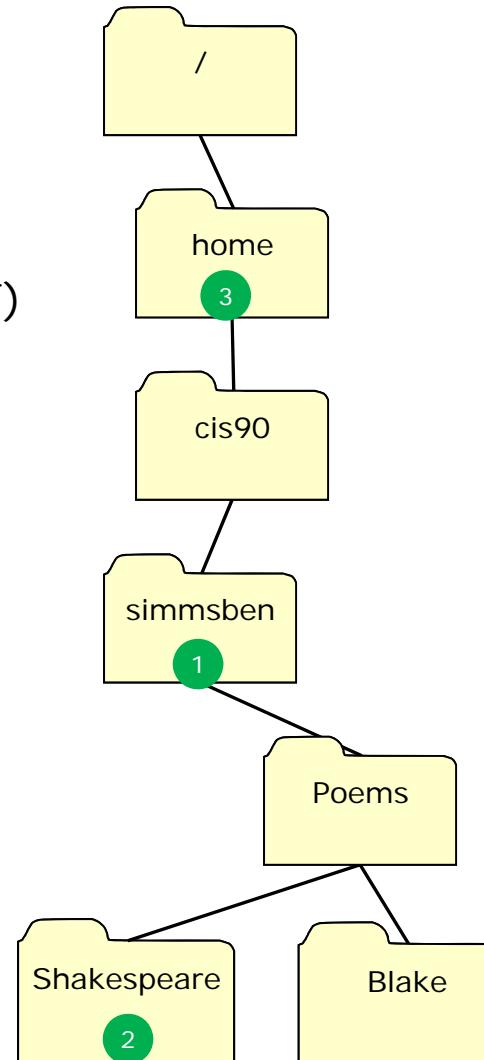
# pwd command

## print working directory

*Note: The shell prompt has been configured for CIS 90 students to always show the current working directory. This example shows the pwd command with a more typical prompt.*

- Syntax: **pwd**
- Prints the current working directory.
- pwd is a BASH builtin command (part of the shell itself)  
`/home/cis90/simmsben $ type pwd`  
pwd is a shell builtin

```
/home/cis90/simmsben $ PS1='[\u@\h \W]\$'  
1 [simmsben@opus ~]$ pwd  
/home/cis90/simmsben  
[simmsben@opus ~]$ cd Poems/Shakespeare/  
2 [simmsben@opus Shakespeare]$ pwd  
/home/cis90/simmsben/Poems/Shakespeare  
[simmsben@opus Shakespeare]$ cd /home/  
3 [simmsben@opus home]$ pwd  
/home  
/home/cis90/simmsben $ PS1='$PWD $ '  
/home/cis90/simmsben $
```





# Navigating

## ls command

# ls command

## lists files

- Syntax: **ls [-a -i -d -l -F -S -R ] [directory]...**

Option	Description
-a	Show all files, even the hidden ones with names starting with "."
-i	Show inode numbers
-d	Show the directory itself rather than the contents of the directory
-l	Long listing (lots of inode information)
-F	Show file types (directory/, program*, link@, socket=)
-S	Sort by size
-R	Recursive (show all sub-directories)

- The *directory* can be:
  - An absolute pathname, e.g. **cd /home/cis90/duke/Poems/**
  - A relative pathname, e.g. **cd /Poems**
  - If no directory is specified, the current working directory is used.
  - Can also be a filename e.g. **ls -l /etc/passwd** to show permissions
  - More than one directory can be specified
- Use **man ls** to see more information.

# ls command example

is used to list file and directory information

*FYI ...*

- **ls** is in /bin and has been aliased to use color on tty's (not pipes)

```
[simmsben@opus ~]$type -a ls
ls is aliased to `ls --color=tty'
ls is /bin/ls
```

- Note: the `--color=tty` added by the alias is what enables the color classifications

*We will learn about aliases later in the course*

# ls command example

*no options*

```
/home/cis90/simmsben $ ls
bigfile  Hidden  letter          Poems      proposal3  text.err  what_am_i
bin       Lab2.0  Miscellaneous   proposal1  small_town  text.fxd
empty     Lab2.1  mission        proposal2  spelllk    timecal
```



*Directories in blue*



*Executables  
(programs or scripts)  
in green*

*Typing the ls command in your home directory displays the files and directories using colors for different file types*

# ls command example

*use the **-F** option to show file types with symbols*

```
/home/cis90/simmsben $ ls -F
bigfile  Hidden/  letter
bin/     Lab2.0/   Miscellaneous/
empty    Lab2.1/   mission
                    Poems/      proposal3   text.err   what_am_i
                                proposal1   small_town  text.fxd
                                proposal2   spellk     timecal*
```

↑  
*Directories end with /*

Regular files

↑  
*Executables  
(programs or scripts)  
end with \**

*Showing file types with colors doesn't work if you are color blind. Use **ls** with the **-F** option to show file types with symbols*

# ls command example

*use the -a option to show hidden files*

```
/home/cis90/simmsben $ cd
```

*cd with no arguments takes you to your home directory*

```
/home/cis90/simmsben $ ls -a
```

.	.bashrc	Hidden	Miscellaneous	proposal1	text.err
..	bigfile	Lab2.0	mission	proposal2	text.fxd
.bash_history	bin	Lab2.1	.mozilla	proposal3	timecal
.bash_logout	.emacs	.lessht	.plan	small_town	what_am_i
.bash_profile	empty	letter	Poems	spellk	.zshrc

```
/home/cis90/simmsben $
```

*Note, all hidden files begin with a .*

*.. is the parent directory and is hidden*

*. is "this directory" or "here" and is hidden*

# ls command example

*use the -i option to show inode numbers*

```
/home/cis90/simmsben $ cd
```

*cd with no arguments take you to your home directory*

```
/home/cis90/simmsben $ ls -i
```

105056	bigfile	102566	Lab2.1	102608	proposal1	102613	text.err
102542	bin	102576	letter	102609	proposal2	102614	text.fxd
102551	empty	102577	Miscellaneous	102610	proposal3	102615	timecal
102552	Hidden	102582	mission	102611	small_town	102616	what_am_i
102555	Lab2.0	102584	Poems	102612	spellk		

*ls with -i option shows inode numbers*

*More on inode numbers later ...*

## Class Exercise

- **cd** (*takes you home*)
- Use the ls command with different option combinations

**ls**                    (*simple listing*)

**ls -l**                (*long listing*)

**ls -a**                (*include hidden files in list*)

**ls -i**                (*show inode numbers for each file*)

**ls -F**                (*use symbols instead of color to show types*)

**ls -aiF**              (*use multiple options at once*)

## Practice test questions

*Which program generated the error message?*

*ls command gave this error*

```
/home/cis90ol/simmsben $ ls -3
ls: invalid option -- 3
Try `ls --help' for more information.
```

*bash shell gave this error*

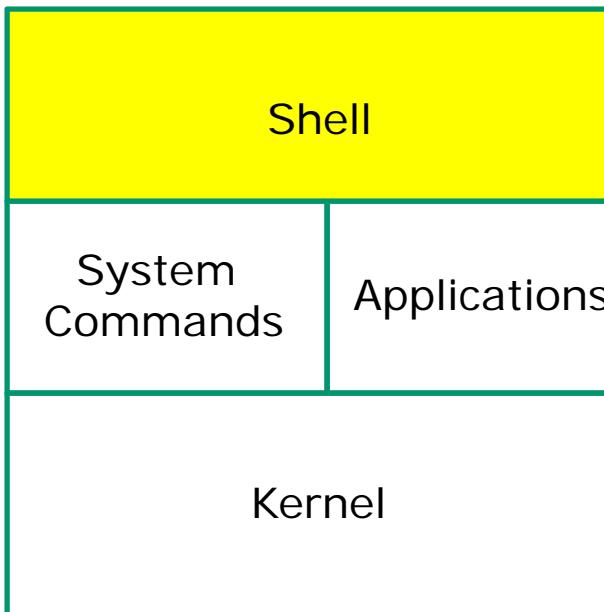
```
/home/cis90ol/simmsben $ lx
-bash: lx: command not found
```

# Navigating

- \* metacharacter



# Life of the Shell



- 1) Prompt
- 2) Parse
- 3) Search
- 4) Execute
- 5) Nap
- 6) Repeat

*Metacharacters, like the \*, are processed during the Parse step (before the selected command is even run)*

## Metacharacters

\* (filename expansion character)

Matches:

- all non-hidden files in the current directory when used alone
- zero or more characters when used as **prefix**, infix or postfix

```
/home/cis90/simmsben $ ls
bigfile  Lab2.0          mission    proposal3   text.fxd
bin      Lab2.1          Poems      small_town  timecal
empty    letter          proposal1  spellk     what_am_i
Hidden   Miscellaneous  proposal2  text.err
```

```
/home/cis90/simmsben $ ls *.err
text.err
```

*\*.err matches all file names ending with ".err"*

*Shell operation question: Does the ls command see the "\*" typed by the user?*

## Metacharacters

\* (filename expansion character)

Matches:

- all non-hidden files in the current directory when used alone
- zero or more characters when used as prefix, **infix** or postfix

```
/home/cis90/simmsben $ ls
bigfile  Lab2.0          mission    proposal3   text.fxd
bin      Lab2.1          Poems      small_town  timecal
empty    letter          proposal1  spell1k     what_am_i
Hidden   Miscellaneous  proposal2  text.err
```

```
/home/cis90/simmsben $ ls *am*
what_am_i
```

**\*am\*** matches all file names containing "am"

Answer to the question on previous slide: NO! The shell replaced the ".err" with the string "text.err" and that's what the **ls** command received as an argument.

## Metacharacters

\* (filename expansion character)

Matches:

- all non-hidden files in the current directory when used alone
- zero or more characters when used as prefix, infix or **postfix**

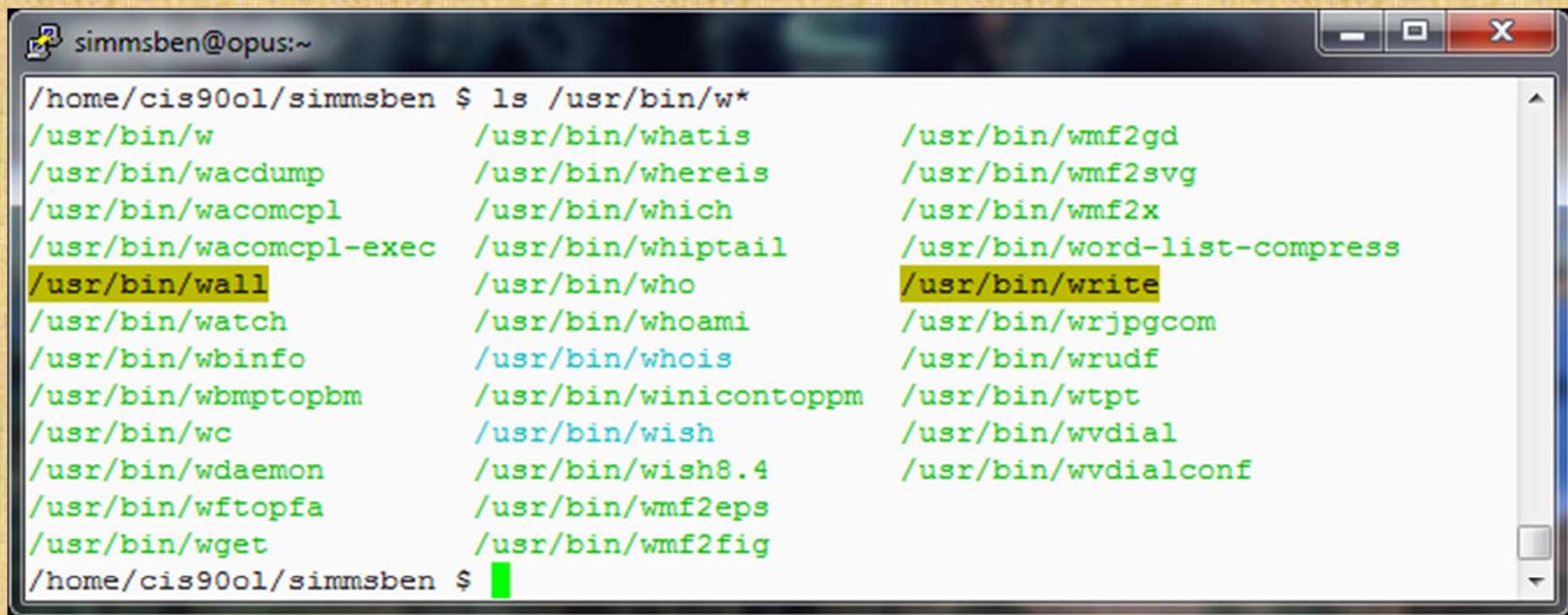
```
/home/cis90/simmsben $ ls
bigfile  Lab2.0          mission    proposal3   text.fxd
bin      Lab2.1          Poems      small_town  timecal
empty    letter          proposal1  spell1k     what_am_i
Hidden   Miscellaneous  proposal2  text.err
```

```
/home/cis90/simmsben $ ls p*
proposal1  proposal2  proposal3
```

*p\* matches all file names starting with a "p"*

## Activity

*What commands in the /usr/bin directory starts with the letter w?*



```
simmsben@opus:~$ ls /usr/bin/w*
/usr/bin/w          /usr/bin/whatis      /usr/bin/wmf2gd
/usr/bin/wacdmp    /usr/bin/whereis    /usr/bin/wmf2svg
/usr/bin/wacomcpl   /usr/bin/which      /usr/bin/wmf2x
/usr/bin/wacomcpl-exec /usr/bin/whiptail /usr/bin/word-list-compress
/usr/bin/wall       /usr/bin/who        /usr/bin/write
/usr/bin/watch      /usr/bin/whoami     /usr/bin/wrjpgcom
/usr/bin/wbinfo     /usr/bin/whois      /usr/bin/wrudf
/usr/bin/wbmptopbm  /usr/bin/winicontoppm /usr/bin/wtpt
/usr/bin/wc         /usr/bin/wish      /usr/bin/wvdial
/usr/bin/wdaemon    /usr/bin/wish8.4   /usr/bin/wvdialconf
/usr/bin/wftopfa    /usr/bin/wmf2eps
/usr/bin/wget       /usr/bin/wmf2fig
```



# Navigating

## More on ls command

# ls command

## Using files and directories as arguments

```
/home/cis90/simmsben $ ls  
bigfile  Lab2.0  
bin      Lab2.1  
empty    letter  
Hidden   Miscellaneous
```

*With no arguments specified, ls will show current directory contents*

```
mission  proposal3  text.fxd  
Poems    small_town  timecal  
proposal1 spellk     what_am_i  
proposal2 text.err
```

```
/home/cis90/simmsben $ ls bigfile  
bigfile
```

*With a filename specified as an argument, the file will be displayed (if it exists)*

```
/home/cis90/simmsben $ ls Poems/  
ant  Blake  nursery  Shakespeare  twister  Yeats
```

*With a directory specified as an argument, the contents of the directory will be displayed (if it exists)*

# ls command

## specifying multiple directories

*The ls command can take more than one argument*

```
/home/cis90/simmsben $ ls Poems/ bin/ Miscellaneous/  
bin/: ←  
app banner enlightenment hi I treed tryme zoom  
  
Miscellaneous/: ←  
better_town file.dos fruit manpage mystery salad  
  
Poems/:  
ant Blake nursery Shakespeare twister Yeats
```

*Note the mystery file is **light blue** ... it is a symbolic link to another file*

# ls command example

The \* is expanded by the shell and replaced with the names of all files and directories in the current directory

```
/home/cis90/simmsben $ ls *
```

bigfile letter proposal1 proposal3 spellk text.fxd what\_am\_i *Files listed first*  
empty mission proposal2 small\_town text.err timecal

bin:

```
app banner enlightenment hi I treed tryme zoom Followed by each directory expanded to show that directory's files  
ls: Hidden: Permission denied
```

Lab2.0:

```
386 A_long_name file.9 READNAME this_years_annual_report  
afile annual report junk.old.bak sTrAnGeNeSS
```

Lab2.1:

```
1.1 filename junk letter more old Proposal3 Proposal.old xyz
```

Miscellaneous:

```
better_town file.dos fruit manpage mystery salad
```

Poems:

```
ant Blake nursery Shakespeare twister Yeats
```

*Do you see the error message? Which directory in the home directory could not be listed?*

# ls command

*Use the `-l` option for a “long listing”*

```

simmsben@opus:~]$ ls -l
total 204
-rw-rw-r-- 1 simmsben cis90      56 Jul  8 17:22 bcommands
-rw-r--r-- 2 simmsben cis90 10576 Jul 20 2001 bigfile
drwxr-xr-x 2 simmsben cis90    4096 Sep 11 2005 bin
-rw-r--r-- 1 simmsben cis90       0 Jul 20 2001 empty
d----- 2 simmsben cis90    4096 Feb  1 2002 Hidden
drwxr-xr-x 2 simmsben cis90    4096 Feb 17 2001 Lab2.0
drwxr-xr-x 3 simmsben cis90    4096 Feb 17 2001 Lab2.1
-rw-r--r-- 1 simmsben cis90   1044 Jul 20 2001 letter
-rw----- 1 simmsben cis90   5799 Jul 24 21:08 mbox
drwxr-xr-x 2 simmsben cis90    4096 Sep 11 2005 Miscellaneous
-rw-r--r-- 1 simmsben cis90     759 Jun  6 2002 mission
drwxr-xr-x 5 simmsben cis90    4096 Jul  9 14:24 Poems
-rw-r--r-- 1 simmsben cis90   1074 Aug 26 2003 proposal1
-rw-r--r-- 1 simmsben cis90   2175 Jul 20 2001 proposal2
-rw-r--r-- 1 simmsben cis90   2054 Sep 14 2003 proposal3
-rw-r--r-- 1 simmsben cis90   5467 Jul  6 13:41 results-e1
-rw-r--r-- 1 simmsben cis90   1286 Jul  6 12:20 results-e1a
-rw-rw-r-- 1 simmsben cis90     688 Jul 24 15:35 salsa
-rw-r--r-- 1 simmsben cis90   1580 Nov 16 2004 small_town
-rw-r--r-- 1 simmsben cis90     485 Aug 26 2003 spellk
-rw-r--r-- 1 simmsben cis90     250 Jul 20 2001 text.err
-rw-r--r-- 1 simmsben cis90    231 Jul 20 2001 text.fxd
-rwxr-xr-x 1 simmsben cis90    509 Jun  6 2002 timecal
-rw-r--r-- 1 simmsben cis90    352 Jul 20 2001 what_am_i

```

total size of all files in blocks

1. file type  
- = regular  
d = directory  
l = link
2. permissions
3. Number of hard links
4. owner
5. group
6. size (in bytes)
7. last modified
8. file name

# ls command example

```
simmsben@opus:~]$ ls -ls
total 204
-rw-r--r-- 2 simmsben cis90 10576 Jul 20 2001 bigfile
-rw----- 1 simmsben cis90 5799 Jul 24 21:08 mbox
-rw-r--r-- 1 simmsben cis90 5467 Jul 6 13:41 results-e1
drwxr-xr-x 2 simmsben cis90 4096 Sep 11 2005 bin
d----- 2 simmsben cis90 4096 Feb 1 2002 Hidden
drwxr-xr-x 2 simmsben cis90 4096 Feb 17 2001 Lab2.0
drwxr-xr-x 3 simmsben cis90 4096 Feb 17 2001 Lab2.1
drwxr-xr-x 2 simmsben cis90 4096 Sep 11 2005 Miscellaneous
drwxr-xr-x 5 simmsben cis90 4096 Jul 9 14:24 Poems
-rw-r--r-- 1 simmsben cis90 2175 Jul 20 2001 proposal2
-rw-r--r-- 1 simmsben cis90 2054 Sep 14 2003 proposal3
-rw-r--r-- 1 simmsben cis90 1580 Nov 16 2004 small_town
-rw-r--r-- 1 simmsben cis90 1286 Jul 6 12:20 results-e1a
-rw-r--r-- 1 simmsben cis90 1074 Aug 26 2003 proposal1
-rw-r--r-- 1 simmsben cis90 1044 Jul 20 2001 letter
-rw-r--r-- 1 simmsben cis90 759 Jun 6 2002 mission
-rw-rw-r-- 1 simmsben cis90 688 Jul 24 15:35 salsa
-rwxr-xr-x 1 simmsben cis90 509 Jun 6 2002 timecal
-rw-r--r-- 1 simmsben cis90 485 Aug 26 2003 spellk
-rw-r--r-- 1 simmsben cis90 352 Jul 20 2001 what_am_i
-rw-r--r-- 1 simmsben cis90 250 Jul 20 2001 text.err
-rw-r--r-- 1 simmsben cis90 231 Jul 20 2001 text.fxd
-rw-rw-r-- 1 simmsben cis90 56 Jul 8 17:22 bcommands
-rw-r--r-- 1 simmsben cis90 0 Jul 20 2001 empty
[simmsben@opus ~]$
```

*Long listing (-l)  
sorted by file  
size (-S) of  
home directory*

# ls command

*the directory or the contents of a directory*

```
/home/cis90ol/simmsben $ ls bin  
app banner enlightenment hi I treed tryme zoom
```

*The contents of the directory  
are shown*

```
/home/cis90ol/simmsben $ ls -d bin  
bin
```

*The directory itself is shown  
with the -d option*

*Use the **d** option to list the directory itself. Without the **d** the directory contents are listed instead.*

# ls command

*the directory or the contents of a directory*

```
/home/cis90/simmsben $ ls -l bin      The contents of the directory  
are shown  
total 68  
-rwxr-xr-x 1 simmsben cis90 220 Apr 22 2004 app  
-rwxr-xr-x 1 simmsben cis90 6160 Aug 28 2003 banner  
-rwxr-xr-x 1 simmsben cis90 3388 Sep 11 2005 enlightenment  
-rwxr-xr-x 1 simmsben cis90 107 Jul 20 2001 hi  
-rwxr-x--x 1 simmsben cis90 375 Oct 20 2003 I  
-rwxr-xr-x 1 simmsben cis90 190 Jul 20 2001 treed  
-rwxr-xr-x 1 simmsben cis90 174 Mar 4 2004 tryme  
-rwxr-xr-x 1 simmsben cis90 74 Jul 20 2001 zoom
```

```
/home/cis90/simmsben $ ls -ld bin      The directory itself is shown  
with the -d option  
drwxr-xr-x 2 simmsben cis90 4096 Sep 11 2005 bin  
/home/cis90/simmsben $
```

*Use the **d** option to get long listing information about the directory  
itself. Without the **d** the directory contents are listed instead.*

*What are some different ways to get the inode number of your home directory?*

# ls command

*different ways to determine the inode number of your home directory*

```
/home/cis90ol/simmsben $ ls -id /home/cis90ol/simmsben  
2329740 /home/cis90ol/simmsben
```

*using an absolute pathname*

```
/home/cis90ol/simmsben $ ls -i /home/cis90ol  
2523721 answers      2329975 dienequi    2330007 lighttom   2329935 sylvijos  
2395395 bin          2329756 elmenchr    2329839 lynbeeri   2329951 vieyrlleo  
2329831 carvaema    2329748 herodbob    2329780 mcnamdan   2329847 vistigab  
2329943 cheeken     2329764 hextcra    2329855 montageo   2329927 warrejes  
2329991 christan    2329919 hillejef    2329967 paytomar   2329983 willitaj  
1902656 cis90        2330015 hwangyuc   2329823 roddyduk   2329772 wilsodan  
2329657 clarkric    2329999 keezeter    2329740 simmsben   2329911 wingejas  
2524651 depot       2329871 lewisgre    2329811 stumbdav
```

*using contents of the parent directory*

```
/home/cis90ol/simmsben $ ls -id .  
2329740 .
```

*. in this context is an absolute pathname to here.*

```
/home/cis90ol/simmsben $ ls -idl .  
2329740 drwxr-xr-x 9 simmsben cis90ol 4096 Feb 28 09:25 .
```

# ls command

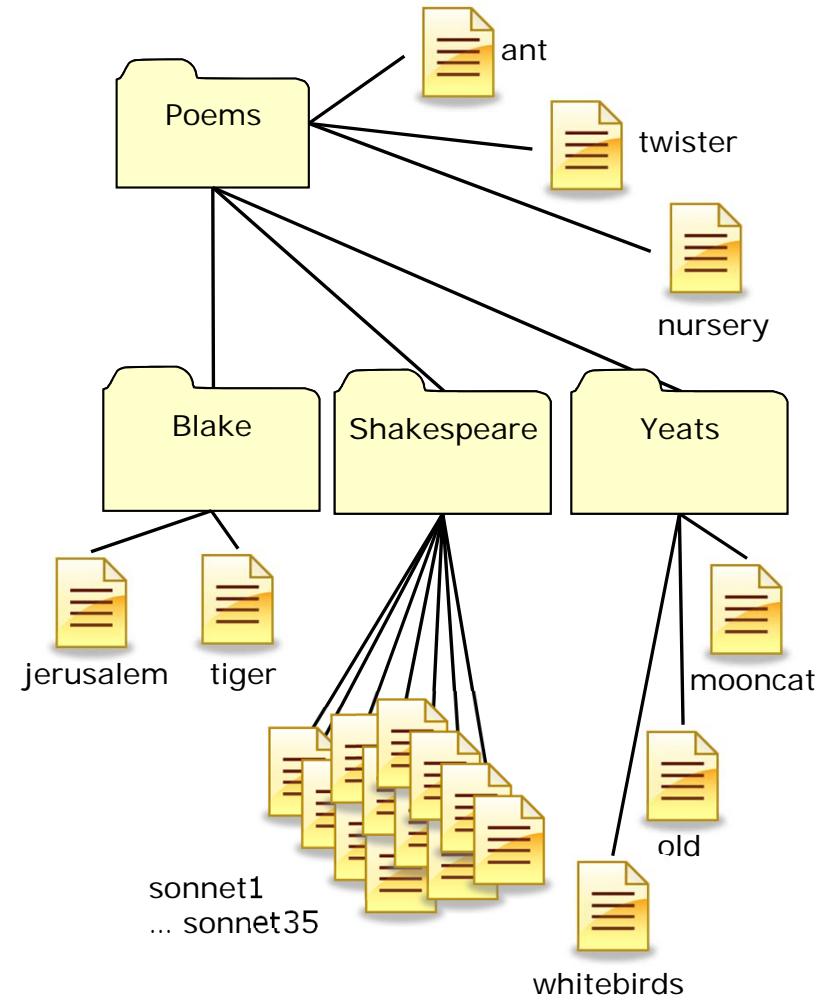
## long listing (-l), recursively list subdirectories (-R)

```
simmsben@opus:~/Poems
[simmsben@opus Poems]$ ls -lR
.:
total 48
-rw-r--r-- 1 simmsben cis90 237 Aug 26 2003 ant
drwxr-xr-x 2 simmsben cis90 4096 Jul 20 2001 Blake
-rw-r--r-- 1 simmsben cis90 779 Oct 12 2003 nursery
drwxr-xr-x 2 simmsben cis90 4096 Oct 31 2004 Shakespeare
-rw-r--r-- 1 simmsben cis90 151 Jul 20 2001 twister
drwxr-xr-x 2 simmsben cis90 4096 Jul 20 2001 Yeats

./Blake:
total 16
-rw-r--r-- 1 simmsben cis90 582 Jul 20 2001 jerusalem
-rw-r--r-- 1 simmsben cis90 115 Jul 20 2001 tiger

./Shakespeare:
total 104
-rw-r--r-- 1 simmsben cis90 614 Jul 20 2001 sonnet1
-rw-r--r-- 1 simmsben cis90 620 Jul 20 2001 sonnet10
-rw-r--r-- 1 simmsben cis90 689 Oct 31 2004 sonnet11
-rw-r--r-- 1 simmsben cis90 618 Jul 20 2001 sonnet15
-rw-r--r-- 1 simmsben cis90 647 Jul 20 2001 sonnet17
-rw-r--r-- 1 simmsben cis90 631 Jul 20 2001 sonnet2
-rw-r--r-- 1 simmsben cis90 601 Jul 20 2001 sonnet26
-rw-r--r-- 1 simmsben cis90 615 Jul 20 2001 sonnet3
-rw-r--r-- 1 simmsben cis90 598 Jul 20 2001 sonnet35
-rw-r--r-- 1 simmsben cis90 588 Jul 20 2001 sonnet4
-rw-r--r-- 1 simmsben cis90 622 Jul 20 2001 sonnet5
-rw-r--r-- 1 simmsben cis90 581 Jul 20 2001 sonnet7
-rw-r--r-- 1 simmsben cis90 620 Jul 20 2001 sonnet9

./Yeats:
total 24
-rw-r--r-- 1 simmsben cis90 855 Jul 20 2001 mooncat
-rw-r--r-- 1 simmsben cis90 520 Jul 20 2001 old
-rw-r--r-- 1 simmsben cis90 863 Jul 20 2001 whitebirds
[simmsben@opus Poems]$
```



## Class Exercise

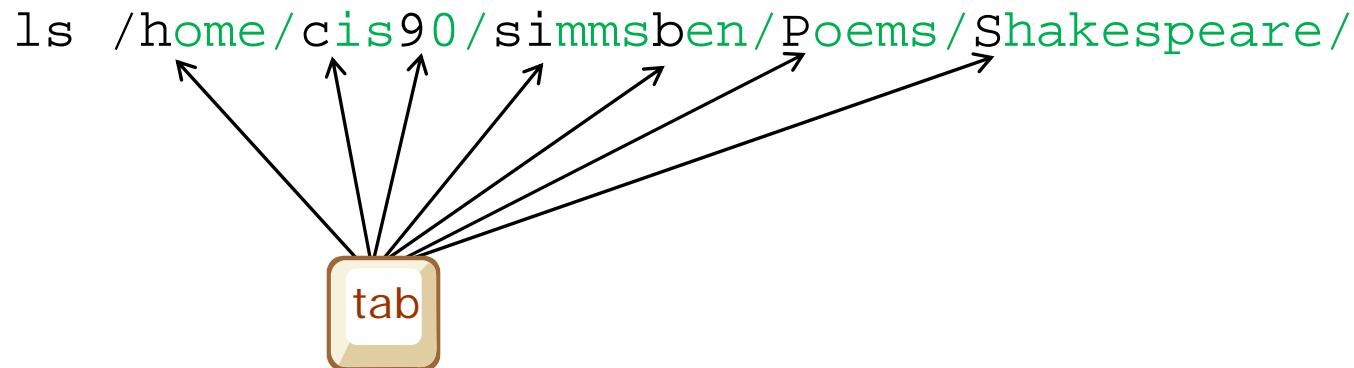
- Go to your home directory, type: **cd**
- Use the **ls** command with different arguments:
  - ls –S *(sort by size)*
  - ls –Sr *(sort by size in reverse order)*
  - ls –id / *(get the inode number of the root directory)*
  - ls .. *(list contents of parent directory)*
  - ls –id .. *(get inode number of parent directory)*
  - ls –id /home/cis90ol/ *(get inode number of parent directory)*
  - ls bin Poems *(short listings of bin and Poems directories)*
  - ls –R *(recursive simple listing of home directory)*
  - ls –IR *(recursive long listing of home directory)*

# Shell tips

# bash shell tip

## tab completes

- It can be tedious typing in long pathnames.
- Since bash knows the names of the files you only have to type just enough characters to uniquely specify a name and then the tab key can be pressed to complete them.
- Example: the black characters were typed by the user, the green ones were typed by bash:



# bash shell tip

## command history and editing

- It can be tedious re-typing a long command to fix a typo.
- Since bash knows the commands you have previously entered, just use the up and down arrows to re-type a previous command.
- When the command you want appears, use the home, right or left arrow keys to go where you want to make the correction. New text can be inserted and old text deleted or backspaced over.
- Example: The ls command was mis-typed as la:

```
/home/cis90/simmsben $ la /home/cis90/simmsben/Poems/Shakespeare/  
-bash: la: command not found
```

  then fix typo

```
/home/cis90/simmsben $ ls /home/cis90/simmsben/Poems/Shakespeare/  
sonnet1    sonnet11   sonnet17   sonnet26   sonnet35   sonnet5   sonnet9  
sonnet10   sonnet15   sonnet2     sonnet3     sonnet4     sonnet7  
/home/cis90/simmsben $
```

# UNIX Files

# File Systems

## Linux

*A typical hard drive*

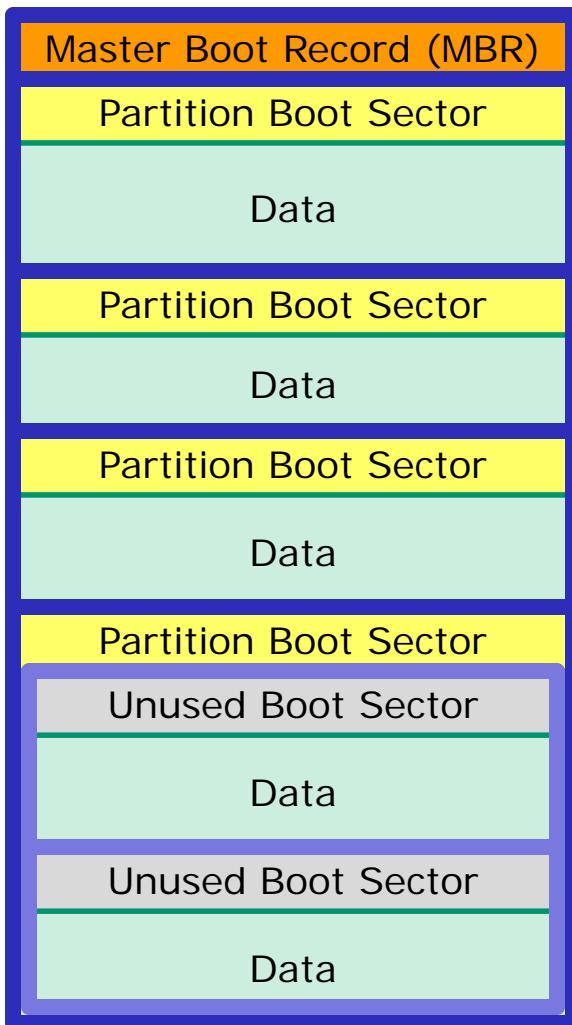


*This is where your files actually reside*

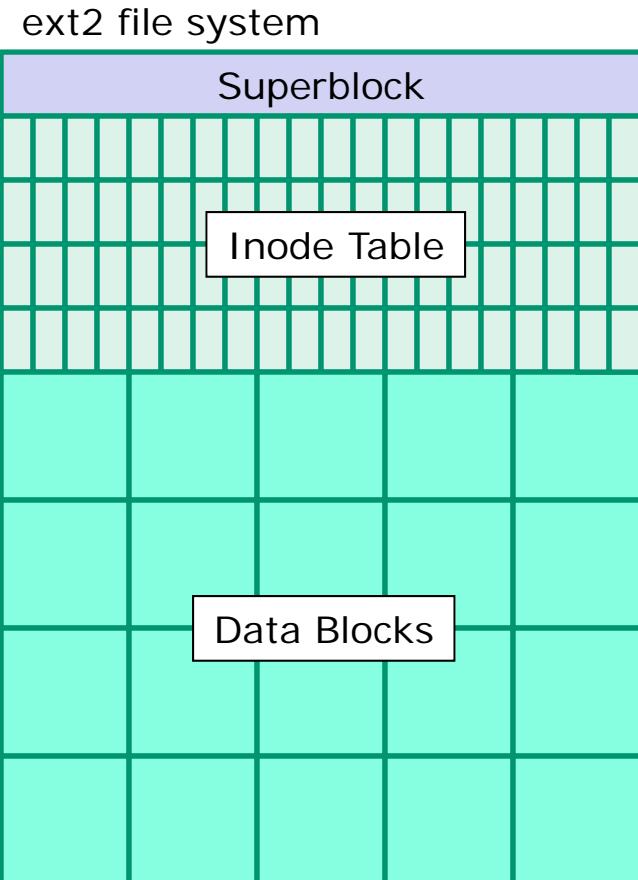


# File Systems

## Linux



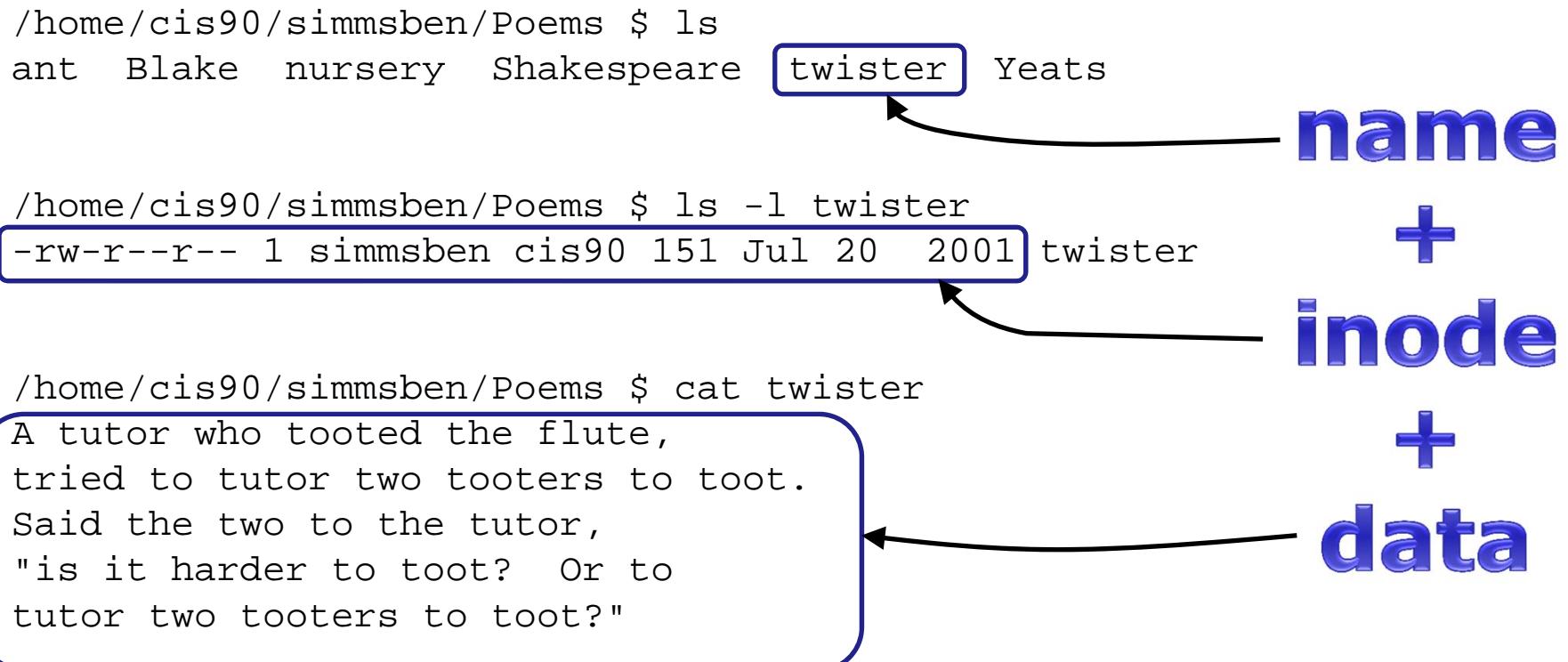
*The hard drive is partitioned and the data areas can be formatted as a file system. Linux typically uses ext2 and ext3 file systems. Windows uses FAT32 and NTFS file systems.*



# UNIX Files

## The three elements of a file

```
/home/cis90/simmsben/Poems $ ls  
ant Blake nursery Shakespeare twister Yeats
```



twister

**name**

+

**inode**

+

**data**

```
/home/cis90/simmsben/Poems $ ls -l twister  
-rw-r--r-- 1 simmsben cis90 151 Jul 20 2001 twister
```

```
A tutor who tooted the flute,  
tried to tutor two tooters to toot.  
Said the two to the tutor,  
"is it harder to toot? Or to  
tutor two tooters to toot?"
```

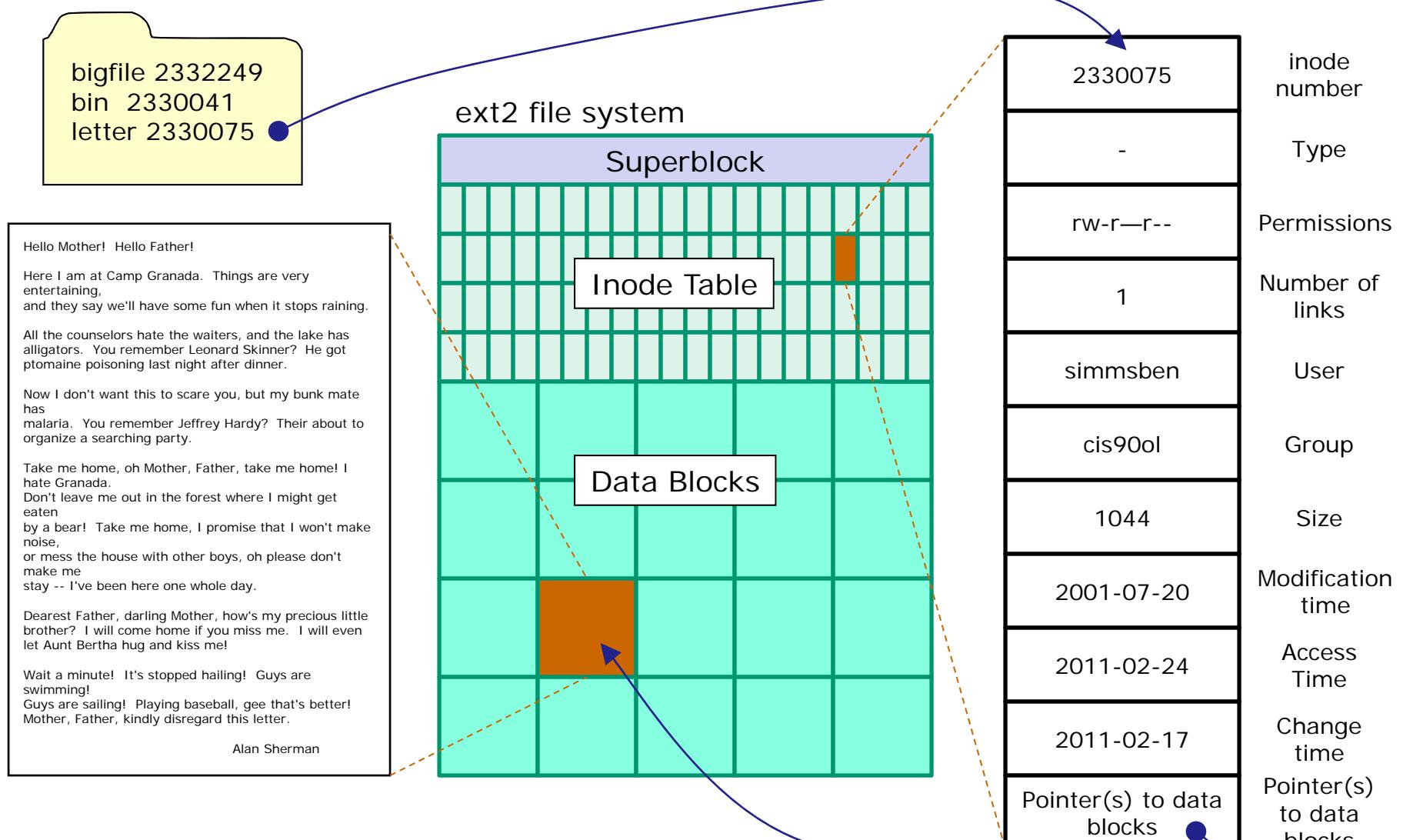
## UNIX File names conventions

Any combination of the following:

- Upper and lower case letters: A-Z and a-z
- Numbers: 0-9
- Periods, underscores, hyphens: . \_ -

*Don't use metacharacters, blanks or /'s as part of your filenames because the shell will treat those characters differently!*

Note: filenames are stored in directories, **not** in inodes



```
/home/cis90ol/simmsben $ ls -il letter
2330075 -rw-r--r-- 1 simmsben cis90ol 1044 Jul 20 2001 letter
```

# File Types

# File Types and Commands

Long listing code (ls -l)	Type	How to make one
d	directory	mkdir
-	regular <ul style="list-style-type: none"><li>• Programs</li><li>• Text</li><li>• Data (binary)</li></ul>	touch
l	symbolic link	ln -s
c	character special device files	mknod
b	block special device files	mknod

*Note: Other file types includes sockets (s) and named pipes (p)*

*Note: Use the **file** command to distinguish between text and date files*

## Various Types of Files (found in /etc)

The terminal window shows a list of files in the /boot directory. The files are color-coded based on their type:

- Regular files (black):** popularity-contest.conf, power, ppp, profile, profile.d, protocols, pulse, purple, python, python2.5.
- Directories (blue):** rc0.d, rc1.d, rc2.d, rc3.d, rc4.d, rc5.d, rc6.d, rc.local, rcS.d, readline, resolvconf.
- Regular files with execute bit set (green):** resolv.conf, rmt, rpc, samba, sane.d, scim, screenrc.

File Type	File Name	Owner	Last Modified	Permissions
Regular File	popularity-contest.conf	root	2008-06-20 11:10	-rw-r--r--
Directory	power	root	2008-04-22 13:52	drwxr-xr-x
Regular File	ppp	dip	2008-04-22 14:01	drwxr-xr-x
Regular File	profile	root	2008-04-22 13:49	-rw-r--r--
Regular File	profile.d	root	2008-04-15 01:53	drwxr-xr-x
Regular File	protocols	root	2007-12-03 17:04	-rw-r--r--
Regular File	pulse	root	2008-04-22 14:03	drwxr-xr-x
Regular File	purple	root	2008-04-22 14:03	drwxr-xr-x
Regular File	python	root	2008-04-22 13:49	drwxr-xr-x
Regular File	python2.5	root	2008-04-22 13:49	drwxr-xr-x
Regular File	rc0.d	root	2008-06-20 11:12	drwxr-xr-x
Regular File	rc1.d	root	2008-04-22 14:07	drwxr-xr-x
Regular File	rc2.d	root	2008-06-20 11:12	drwxr-xr-x
Regular File	rc3.d	root	2008-06-20 11:12	drwxr-xr-x
Regular File	rc4.d	root	2008-06-20 11:12	drwxr-xr-x
Regular File	rc5.d	root	2008-06-20 11:12	drwxr-xr-x
Regular File	rc6.d	root	2008-06-20 11:12	drwxr-xr-x
Regular File	rc.local	root	2008-04-22 13:49	-rwxr-xr-x
Regular File	rcS.d	root	2008-04-22 14:05	drwxr-xr-x
Regular File	readahead	root	2008-04-22 14:03	drwxr-xr-x
Regular File	resolvconf	root	2008-04-22 13:53	drwxr-xr-x
Regular File	resolv.conf	root	2008-06-24 10:44	-rw-r--r--
Regular File	rmt	root	2008-04-04 07:07	-rwxr-xr-x
Regular File	rpc	root	2007-12-03 17:04	-rw-r--r--
Regular File	samba	root	2008-06-20 11:15	drwxr-xr-x
Regular File	sane.d	root	2008-04-22 13:59	drwxr-xr-x
Regular File	scim	root	2008-04-22 14:05	drwxr-xr-x
Regular File	screenrc	root	2007-10-23 12:02	-rw-r--r--

## Various Types of Files (found in /bin)

```
rsimms@ulysses: /bin
File Edit View Terminal Tabs Help
rsimms@ulysses:/bin$ ls -l s* z*
-rwxr-xr-x 1 root root 40724 2007-12-04 07:50 sed
lrwxrwxrwx 1 root root    15 2008-06-20 11:03 setpci -> /usr/bin/setpci
-rwxr-xr-x 1 root root  8431 2008-04-22 01:59 setupcon
lrwxrwxrwx 1 root root     4 2008-06-20 11:03 sh -> dash
lrwxrwxrwx 1 root root     4 2008-06-20 11:03 sh.distrib -> bash
-rwxr-xr-x 1 root root 24488 2008-04-04 02:42 sleep
-rwxr-xr-x 1 root root 48932 2008-04-04 02:42 stty
-rwsr-xr-x 1 root root 25540 2008-04-02 21:08 su
-rwxr-xr-x 1 root root 22312 2008-04-04 02:42 sync
-rwxr-xr-x 1 root root   64 2007-11-15 06:49 zcat
-rwxr-xr-x 1 root root   69 2007-11-15 06:49 zcmp
-rwxr-xr-x 1 root root  4424 2007-11-15 06:49 zdiff
-rwxr-xr-x 1 root root   64 2007-11-15 06:49 zegrep
-rwxr-xr-x 1 root root   64 2007-11-15 06:49 zfgrep
-rwxr-xr-x 1 root root  2015 2007-11-15 06:49 zforce
-rwxr-xr-x 1 root root  4893 2007-11-15 06:49 zgrep
-rwxr-xr-x 1 root root  1733 2007-11-15 06:49 zless
-rwxr-xr-x 1 root root  2416 2007-11-15 06:49 zmore
-rwxr-xr-x 1 root root  4952 2007-11-15 06:49 znew
rsimms@ulysses:/bin$ file sed
sed: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux 2.6
.8, dynamically linked (uses shared libs), stripped
rsimms@ulysses:/bin$ file sh
sh: symbolic link to `dash'
rsimms@ulysses:/bin$ file znew
znew: Bourne-Again shell script text executable
rsimms@ulysses:/bin$
```

Long listing of files with names starting with s or z

Symbolic links (light blue) with arrow to real file

Regular file with setuid bit (red background)

Rest are regular files with execute bit set (green)

Use **file** command to show additional file type information

## Various Types of Files (found in /dev)

The terminal window shows the following output from the command `ls -l /dev/*`:

```
rsimms@ulysses:~$ ls -l /dev/sda
brw-rw---- 1 root disk 8, 0 2008-06-24 10:43 /dev/sda
rsimms@ulysses:~$ ls -l /dev/sda1
brw-rw---- 1 root disk 8, 1 2008-06-24 10:44 /dev/sda1
rsimms@ulysses:~$ ls -l /dev/tty1
crw----- 1 root root 4, 1 2008-06-24 10:44 /dev/tty1
rsimms@ulysses:~$ ls -l /dev/pts/0
crw----- 1 rsimms tty 136, 0 2008-06-24 10:53 /dev/pts/0
rsimms@ulysses:~$ clear
```

Annotations in the image:

- A blue box labeled "Block" has an arrow pointing to the line `brw-rw---- 1 root disk 8, 0 2008-06-24 10:43 /dev/sda`.
- A blue box labeled "Character" has an arrow pointing to the line `crw----- 1 root root 4, 1 2008-06-24 10:44 /dev/tty1`.
- A blue box labeled "Special files (yellow with black background)" has an arrow pointing to the line `brw-rw---- 1 root disk 8, 1 2008-06-24 10:44 /dev/sda1`.

*Hard drives are block devices (data is transferred in large chunks for efficiency). Terminals are character devices where data is transferred one character at a time.*

## /boot (Red Hat 9)

```
[root@frida root]# ls -l /boot
total 5127
-rw-r--r--  1 root    root        5824 Jan 24  2003 boot.b
-rw-r--r--  1 root    root       612 Jan 24  2003 chain.b
-rw-r--r--  1 root    root      44309 Feb 27  2003 config-2.4.20-6
drwxr-xr-x  2 root    root      1024 Jun  5 19:10 grub
-rw-r--r--  1 root    root    254430 Jun  5 18:47 initrd-2.4.20-6.img
-rw-r--r--  1 root    root      473 Jun  5 18:47 kernel.h
drwx----- 2 root    root    12288 Jun  5 11:45 lost+found
-rw-r--r--  1 root    root    23108 Feb 24  2003 message
-rw-r--r--  1 root    root    21282 Feb 24  2003 message.ja
lrwxrwxrwx  1 root    root        20 Jun  5 18:47 module-info -> module-info-2.4.20-6
-rw-r--r--  1 root    root    15436 Feb 27  2003 module-info-2.4.20-6
-rw-r--r--  1 root    root      640 Jan 24  2003 os2_d.b
lrwxrwxrwx  1 root    root        19 Jun  5 18:47 System.map -> System.map-2.4.20-6
-rw-r--r--  1 root    root    520099 Feb 27  2003 System.map-2.4.20-6
-rw-r--r--  1 root    root  3193468 Feb 27  2003 vmlinuz-2.4.20-6
lrwxrwxrwx  1 root    root        16 Jun  5 18:47 vmlinuz -> vmlinuz-2.4.20-6
-rw-r--r--  1 root    root  1122363 Feb 27  2003 vmlinuz-2.4.20-6
[root@frida root]#
```

The kernel

The kernel (compressed)

Symbolic link to kernel

### Class Exercise

- Do a long listing of the /bin directory
  - *Who owns the vi command?*
  - *What size is the sleep command?*
  - *What file does the symbolic link tcptraceroute point to?*
  - *When was the file touch last modified?*
- Do a long listing of the /etc directory
  - *Is yum a directory or a file?*
  - *What are two ways you can tell if yum is a directory or a file?*

# file command

Provides expanded information about files

- There are many different types of regular files:
  - Programs (binary)
  - Scripts (text)
  - Text files
  - Data files (binary)
- The **file** command attempts to classify files and give you more detailed information as to what type they are.

*Use the **file** command to determine if a file is a text file and can be viewed with **cat**, **more**, **less**, **tail** ... etc commands.*

# file command examples

*The file command can take multiple arguments*

```
/home/cis90ol/simmsben $ file Poems/ proposal2 timecal empty
Poems/:      directory
proposal2:  ASCII English text
timecal:    shell archive or script for antique kernel text
empty:      empty
/home/cis90ol/simmsben $
```

*Note, this example is from Lab 4*

# file command examples

```
[rsimms@opus:~/work/examples/filetypes]$ ls -l
total 156
-rw-r--r-- 1 rsimms cis191 8983 Aug  5 07:57 Adjective.frm
-rw-rw-rw- 1 rsimms cis191 5976 Aug  5 07:57 Adjective.MYD
-rw-rw-rw- 1 rsimms cis191 2048 Aug  5 07:57 Adjective.MYI
-rw-r--r-- 1 rsimms cis191 10240 Aug  4 18:13 backup.tar
-rw----- 1 rsimms users    191 Aug  5 08:10 bash.profile
crw-r--r-- 1 rsimms cis191 5, 1 Aug  5 08:03 console
-rwxr-xr-x 1 rsimms cis191 4846 Aug  4 18:08 cprog
lrwxrwxrwx 1 rsimms users    5 Aug  5 08:07 go-cprog -> cprog
-rw-r--r-- 1 rsimms cis191 119 Aug  4 17:55 letter
-rw----- 1 rsimms users    2968 Aug  5 08:08 mbox
-rw-r--r-- 1 rsimms cis191 34611 Aug  5 07:59 rich-260x216.jpg
-rwxr-xr-x 1 rsimms cis191   445 Aug  4 17:56 runit
brw-r--r-- 1 rsimms cis191  8, 0 Aug  5 08:04 sda
drwxr-xr-x 2 rsimms cis191 4096 Aug  4 17:57 travel
[rsimms@opus filetypes]$
```

*Not all regular files are text files*

```
[rsimms@opus:~/work/examples/filetypes]$ file *
Adjective.frm:      MySQL table definition file Version 9
Adjective.MYD:      DBase 3 data file (33517822 records)
Adjective.MYI:      MySQL MISAM compressed data file Version 1
backup.tar:         POSIX tar archive
bash.profile:       ASCII English text
console:            character special (5/1)
cprog:              ELF 32-bit LSB executable, Intel 80386, version
                   1 (SYSV), for GNU/Linux 2.2.5, dynamically linked (uses shared l
                   ibs), for GNU/Linux 2.2.5, not stripped
go-cprog:           symbolic link to `cprog'
letter:             ASCII English text
mbox:               ASCII mail text
rich-260x216.jpg:  JPEG image data, JFIF standard 1.02
runit:              Bourne shell script text executable
sda:                block special (8/0)
travel:             directory
[rsimms@opus filetypes]$
```

*Use the **file** command to identify text files*

# WC command

## count words, lines, and bytes

```
simmsben@opus:~/Poems/Blake
/home/cis90/simmsben/Poems/Blake $ cat tiger
Tiger, Tiger burning bright
In the forest of the night,
What immortal hand or eye
Dare frame thy fearful symmetry?
/home/cis90/simmsben/Poems/Blake $ wc -l tiger Number of lines
4 tiger
/home/cis90/simmsben/Poems/Blake $ wc -w tiger Number of words
20 tiger
/home/cis90/simmsben/Poems/Blake $ wc tiger Number of lines,
        words, bytes
    4 20 115 tiger
/home/cis90/simmsben/Poems/Blake $ ls -l tiger
-rw-r--r-- 1 simmsben cis90 115 Jul 20 2001 tiger
/home/cis90/simmsben/Poems/Blake $ wc -l *
27 jerusalem
    4 tiger
31 total
/home/cis90/simmsben/Poems/Blake $
```

tiger file has 4 lines, 20 words and 115 bytes

## Class Exercise

Navigate with cd, pwd, and ls

- Navigate to your Blake directory
- Use the file command on tiger: **file tiger**
  - *Is tiger a binary or ASCII text file?*
- Using one command only, with a relative path, get expanded type information on the sonnet files in your Shakespeare directory  
**file ..//Shakespeare/\***
- Navigate to your Shakespeare directory
- Print the sonnet3 files with **cat sonnet3**

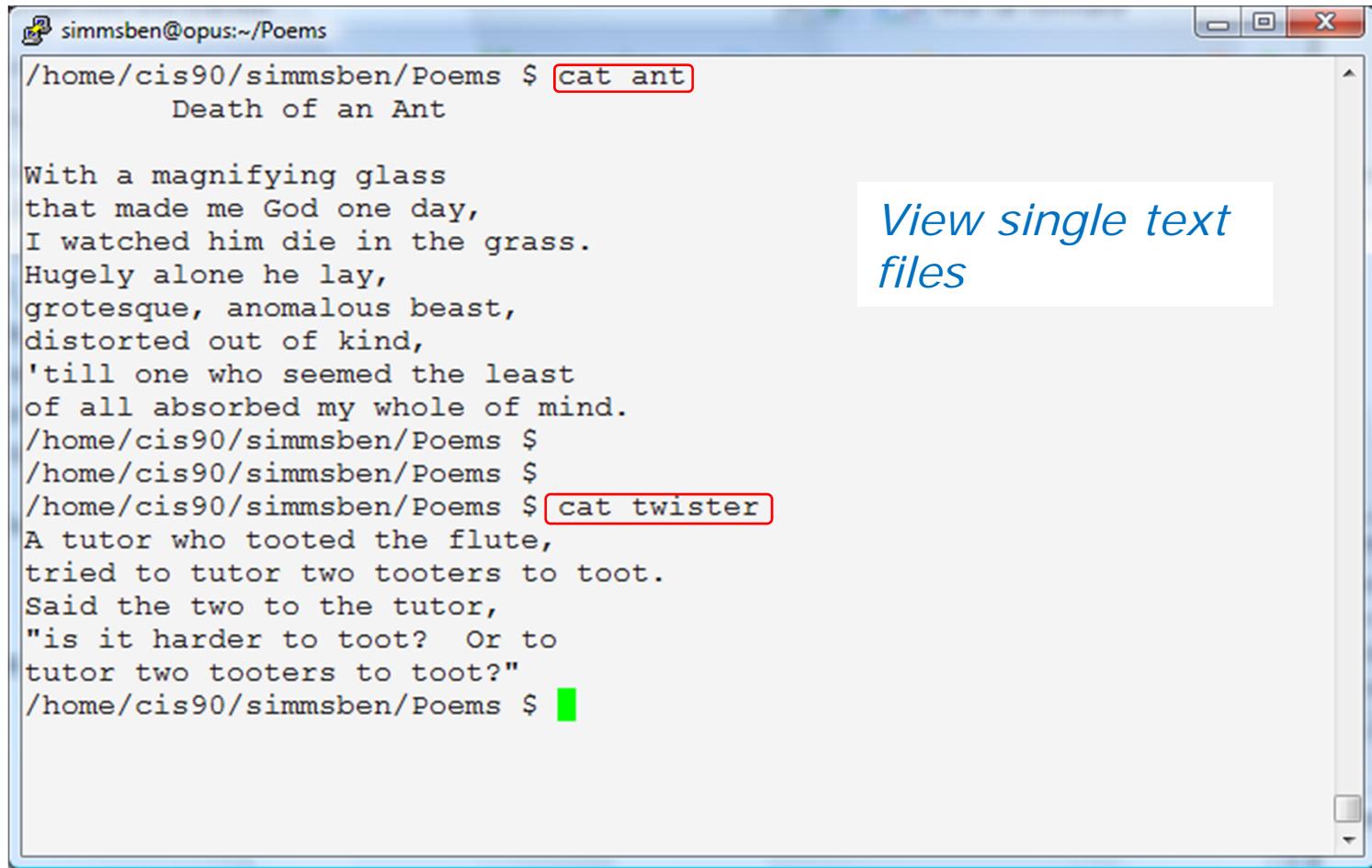
*How many lines in the sonnet3 file? Hint: Use wc -l sonnet3*

*How many words in the sonnet3 file?*

# Viewing Files

## cat command

concatenate or view text files



simmsben@opus:~/Poems

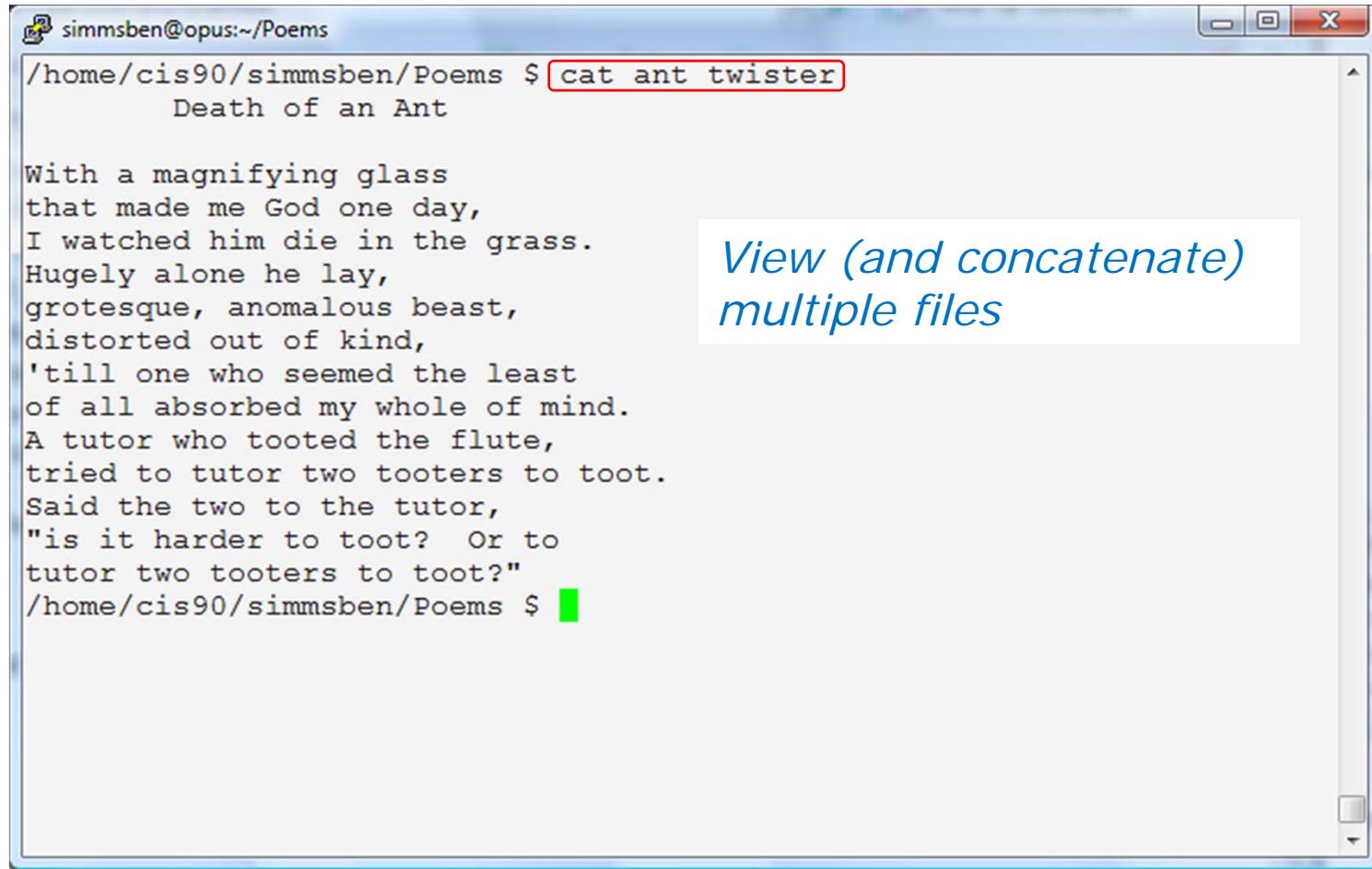
```
/home/cis90/simmsben/Poems $ cat ant
    Death of an Ant

With a magnifying glass
that made me God one day,
I watched him die in the grass.
Hugely alone he lay,
grotesque, anomalous beast,
distorted out of kind,
'till one who seemed the least
of all absorbed my whole of mind.
/home/cis90/simmsben/Poems $
/home/cis90/simmsben/Poems $
/home/cis90/simmsben/Poems $ cat twister
A tutor who tooted the flute,
tried to tutor two tootlers to toot.
Said the two to the tutor,
"is it harder to toot? Or to
tutor two tootlers to toot?"
/home/cis90/simmsben/Poems $
```

*View single text files*

# cat command

## concatenate or view text files



A screenshot of a terminal window titled "simmsben@opus:~/Poems". The command entered is "cat ant twister". The output displays two poems: "Death of an Ant" and "ant twister".

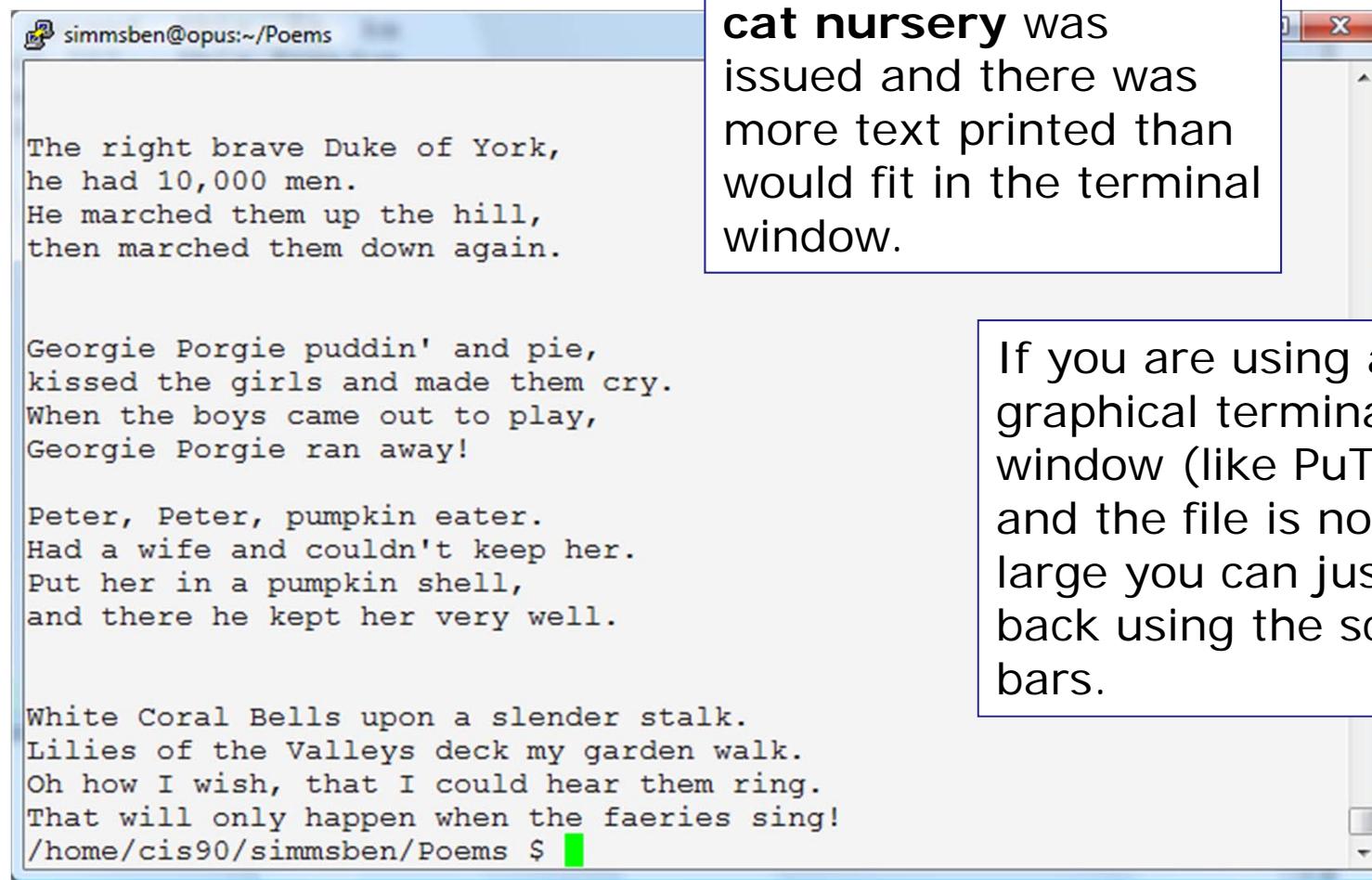
```
simmsben@opus:~/Poems
/home/cis90/simmsben/Poems $ cat ant twister
Death of an Ant

With a magnifying glass
that made me God one day,
I watched him die in the grass.
Hugely alone he lay,
grotesque, anomalous beast,
distorted out of kind,
'till one who seemed the least
of all absorbed my whole of mind.
A tutor who tooted the flute,
tried to tutor two tooters to toot.
Said the two to the tutor,
"is it harder to toot? Or to
tutor two tooters to toot?"
/home/cis90/simmsben/Poems $
```

*View (and concatenate)  
multiple files*

## cat command

concatenate or view text files



```
simmsben@opus:~/Poems
The right brave Duke of York,
he had 10,000 men.
He marched them up the hill,
then marched them down again.

Georgie Porgie puddin' and pie,
kissed the girls and made them cry.
When the boys came out to play,
Georgie Porgie ran away!

Peter, Peter, pumpkin eater.
Had a wife and couldn't keep her.
Put her in a pumpkin shell,
and there he kept her very well.

White Coral Bells upon a slender stalk.
Lilies of the Valleys deck my garden walk.
Oh how I wish, that I could hear them ring.
That will only happen when the faeries sing!
/home/cis90/simmsben/Poems $
```

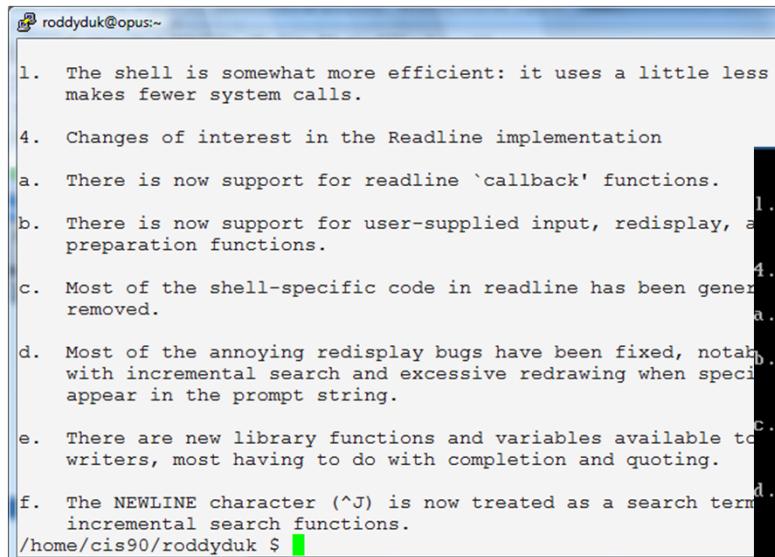
**cat nursery** was issued and there was more text printed than would fit in the terminal window.

If you are using a graphical terminal window (like PuTTY) and the file is not too large you can just scroll back using the scroll bars.

# cat command

## concatenate or view text files

- Problem - if you **cat** really long files the text at the beginning is scrolled off and cannot be read.
- For example: **cat /usr/share/doc/bash-3.2/NEWS**

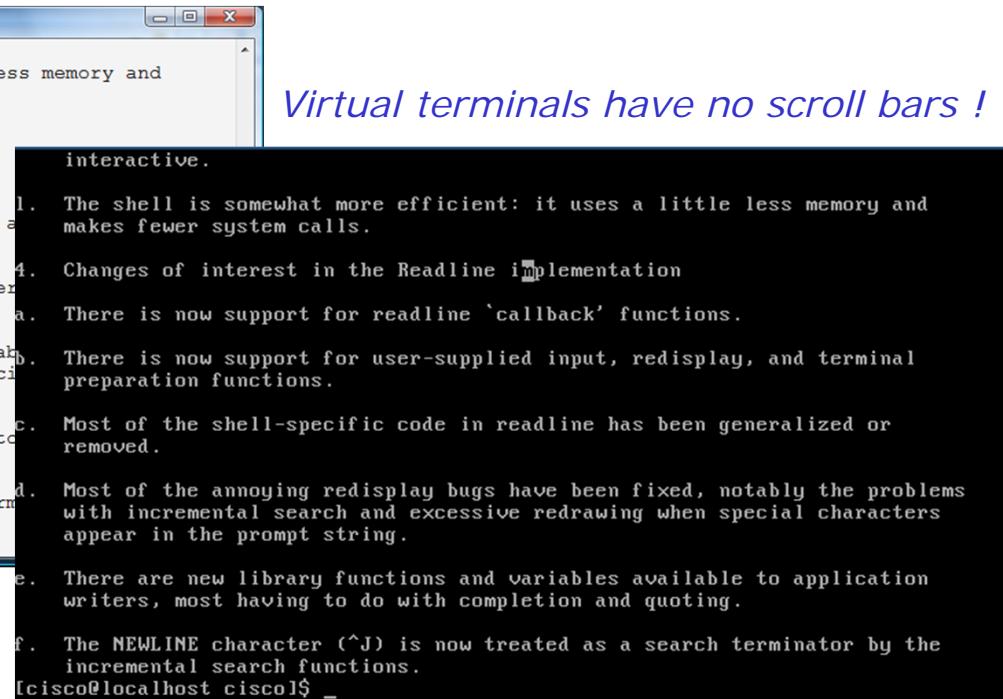


```
roddyduk@opus:~$ cat /usr/share/doc/bash-3.2/NEWS
1. The shell is somewhat more efficient: it uses a little less memory and
   makes fewer system calls.

4. Changes of interest in the Readline implementation
a. There is now support for readline 'callback' functions.
b. There is now support for user-supplied input, redisplay, and
   preparation functions.
c. Most of the shell-specific code in readline has been generalized
   or removed.
d. Most of the annoying redisplay bugs have been fixed, notably the
   problems with incremental search and excessive redrawing when special
   characters appear in the prompt string.
e. There are new library functions and variables available to application
   writers, most having to do with completion and quoting.
f. The NEWLINE character (^J) is now treated as a search terminator by the
   incremental search functions.

/home/cis90/roddyduk $
```

Terminal windows (like PuTTY) have scroll bars but the number of lines they buffer can be exceeded.



Virtual terminals have no scroll bars !

```
interactive.
1. The shell is somewhat more efficient: it uses a little less memory and
   makes fewer system calls.

4. Changes of interest in the Readline implementation
a. There is now support for readline 'callback' functions.
b. There is now support for user-supplied input, redisplay, and terminal
   preparation functions.
c. Most of the shell-specific code in readline has been generalized or
   removed.
d. Most of the annoying redisplay bugs have been fixed, notably the problems
   with incremental search and excessive redrawing when special characters
   appear in the prompt string.

e. There are new library functions and variables available to application
   writers, most having to do with completion and quoting.

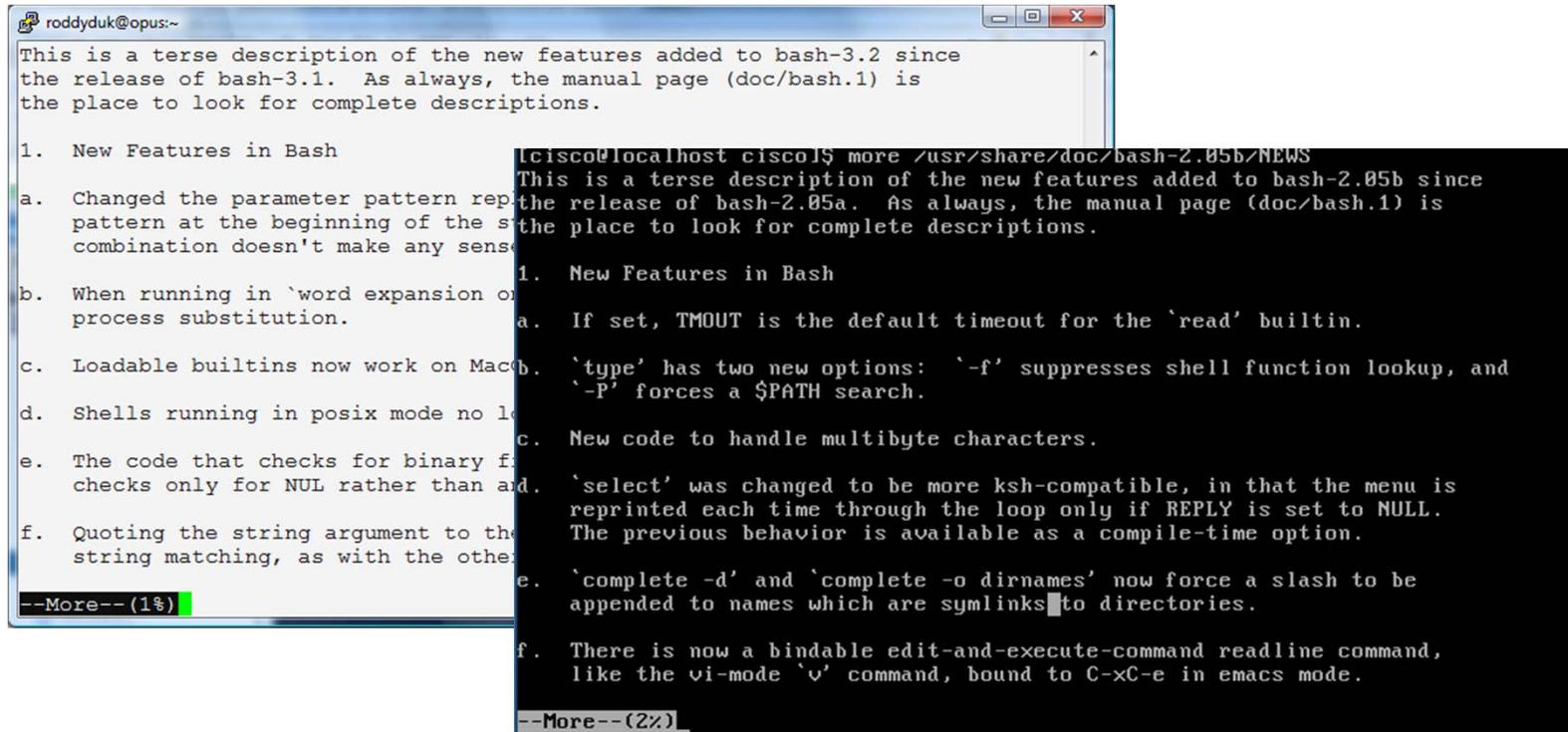
f. The NEWLINE character (^J) is now treated as a search terminator by the
   incremental search functions.

[cisco@localhost cisco]$ _
```

# more command

*For printing really big files*

- Use the **more** command for paging through really long text files
- For example: **more /usr/share/doc/bash-3.2/NEWS**



The image shows two side-by-side terminal windows. Both windows are displaying the same text from the file `/usr/share/doc/bash-3.2/NEWS`. The text describes new features in bash-3.2 compared to bash-3.1, mentioning changes in parameter pattern replacement, word expansion, loadable builtins, shells running in posix mode, binary file handling, quoting, and various command improvements like `select`, `complete`, and readline commands.

The left terminal window has a status bar at the bottom that says `--More-- (1%)`. The right terminal window also has a status bar at the bottom that says `--More-- (2%)`.

```
This is a terse description of the new features added to bash-3.2 since
the release of bash-3.1. As always, the manual page (doc/bash.1) is
the place to look for complete descriptions.

1. New Features in Bash
   a. Changed the parameter pattern replacement at the beginning of the string combination doesn't make any sense.
      b. When running in 'word expansion or process substitution.
         c. Loadable builtins now work on Mac OS X.
            d. Shells running in posix mode no longer check for binary files.
               e. The code that checks for binary files now checks only for NUL rather than aid.
                  f. Quoting the string argument to the string matching, as with the other
                     g. New code to handle multibyte characters.
                        h. 'select' was changed to be more ksh-compatible, in that the menu is reprinted each time through the loop only if REPLY is set to NULL. The previous behavior is available as a compile-time option.
                           i. 'complete -d' and 'complete -o dirnames' now force a slash to be appended to names which are symlinks to directories.
                              j. There is now a bindable edit-and-execute-command readline command, like the vi-mode 'v' command, bound to C-xC-e in emacs mode.

[cisco@localhost cisco]$ more /usr/share/doc/bash-2.05b/NEWS
This is a terse description of the new features added to bash-2.05b since
the release of bash-2.05a. As always, the manual page (doc/bash.1) is
the place to look for complete descriptions.

1. New Features in Bash
   a. If set, TMOUT is the default timeout for the 'read' builtin.
      b. 'type' has two new options: '-f' suppresses shell function lookup, and
         '-P' forces a $PATH search.
            c. New code to handle multibyte characters.
               d. 'select' was changed to be more ksh-compatible, in that the menu is reprinted each time through the loop only if REPLY is set to NULL. The previous behavior is available as a compile-time option.
                  e. 'complete -d' and 'complete -o dirnames' now force a slash to be appended to names which are symlinks to directories.
                     f. There is now a bindable edit-and-execute-command readline command, like the vi-mode 'v' command, bound to C-xC-e in emacs mode.
```

*Use the space key to page forward and q to quit*

# more command

*Printing multiple files with one command*

- Use the **more** command can take multiple arguments

```
/home/cis90ol/simmsben $ more Poems/Blake/tiger Poems/Blake/jerusalem
::::::::::::::::::
Poems/Blake/tiger ←
::::::::::::::::::
Tiger, Tiger burning bright
In the forest of the night,
What immortal hand or eye
Dare frame thy fearful symmetry?
::::::::::::::::::
Poems/Blake/jerusalem ←
::::::::::::::::::
Jerusalem
```

*Notice with multiple files as arguments, each file has a header to separate it from the other files*

And did those feet in ancient times,  
Walk upon England's mountains green?  
And was the holy lamb of God,  
On England's pleasant pastures seen?

And did the countenance divine  
Shine forth upon our darkened hills?  
And was Jerusalem builded here,  
< snipped >

## more command

*Printing multiple files using \* metacharacter*

- Use the **more** command can take multiple arguments

```
/home/cis90ol/simmsben $ more Poems/Blake/*
```

```
:::::::  
Poems/Blake/jerusalem  
:::::::  
Jerusalem
```

And did those feet in ancient times,  
Walk upon England's mountains green?  
And was the holy lamb of God,  
On England's pleasant pastures seen?

And did the countenance divine  
Shine forth upon our darkened hills?  
And was Jerusalem builded here,  
Among these dark satanic mills.

Bring me my bow of burning gold.  
Bring me my arrows of desire.  
Bring me my spear, Oh clouds unfold!  
Bring me my chariot of fire!

I will not cease from endless fight!  
Nor shall my sword sleep in my hand,  
'til we have built Jerusalem  
On England's green and pleasant land.

*The previous example using the \* metacharacter instead*

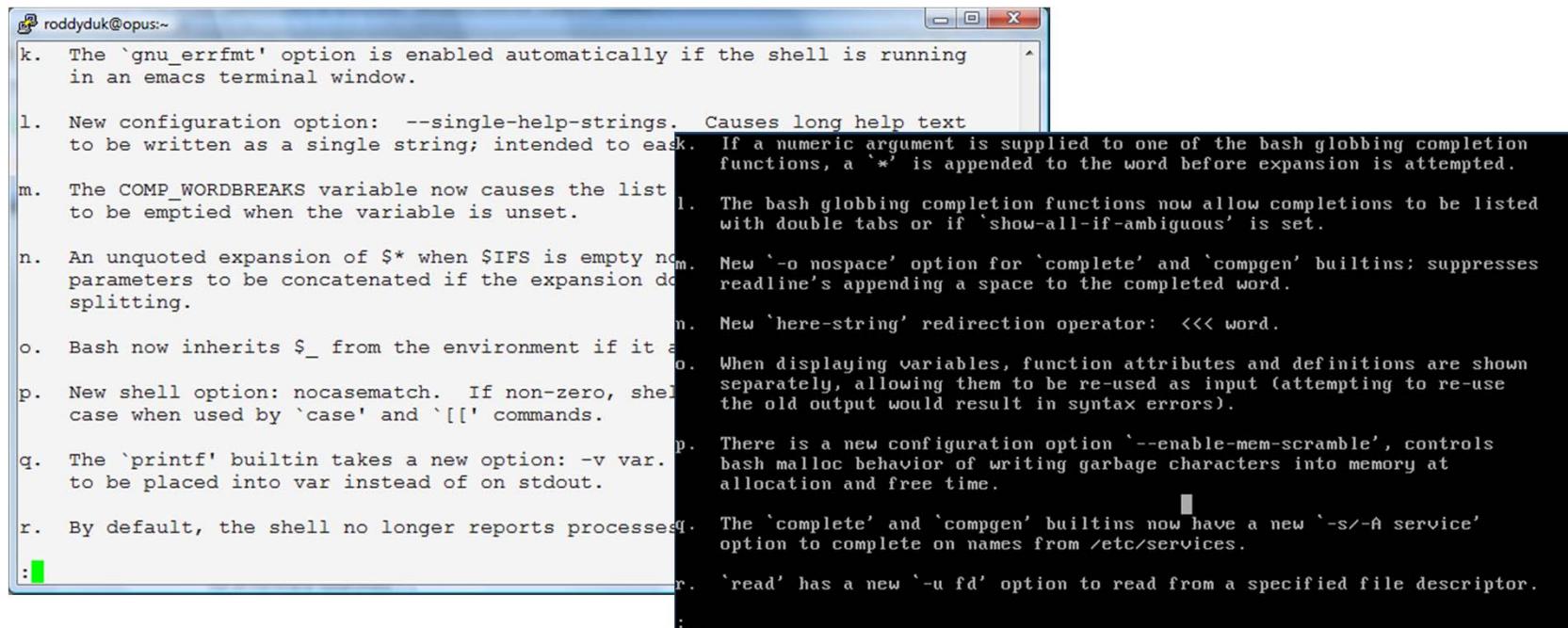
William Blake

```
:::::::  
Poems/Blake/tiger  
:::::::  
< snipped >
```

# less command

*An alternative for printing really big files*

- Use the **less** command to page forward and backward through really long text files. (just like the man command works)
- For example: **less /usr/share/doc/bash-3.2/NEWS**



A screenshot of a terminal window titled "roddyduk@opus:~". The window displays the contents of the "/usr/share/doc/bash-3.2/NEWS" file. The text is presented in a single column, with some lines wrapped to fit the screen. The terminal has a standard Windows-style interface with a title bar and a scroll bar.

```
k. The `gnu_errfmt' option is enabled automatically if the shell is running
   in an emacs terminal window.

l. New configuration option: --single-help-strings. Causes long help text
   to be written as a single string; intended to ease. If a numeric argument is supplied to one of the bash globbing completion
   functions, a '*' is appended to the word before expansion is attempted.

m. The COMP_WORDBREAKS variable now causes the list
   to be emptied when the variable is unset. l. The bash globbing completion functions now allow completions to be listed
   with double tabs or if 'show-all-if-ambiguous' is set.

n. An unquoted expansion of $* when $IFS is empty no. New '-o nospace' option for 'complete' and 'compgen' builtins; suppresses
   parameters to be concatenated if the expansion do readline's appending a space to the completed word.
   splitting.

o. Bash now inherits $_ from the environment if it a. New 'here-string' redirection operator: <<< word.

p. New shell option: nocasematch. If non-zero, shell o. When displaying variables, function attributes and definitions are shown
   case when used by 'case' and '[[[' commands. separately, allowing them to be re-used as input (attempting to re-use
   the old output would result in syntax errors).

q. The 'printf' builtin takes a new option: -v var. p. There is a new configuration option '--enable-mem-scramble', controls
   to be placed into var instead of on stdout. bash malloc behavior of writing garbage characters into memory at
   allocation and free time.

r. By default, the shell no longer reports processes q. The 'complete' and 'compgen' builtins now have a new '-s/-A service'
   option to complete on names from /etc/services.

: r. 'read' has a new '-u fd' option to read from a specified file descriptor.

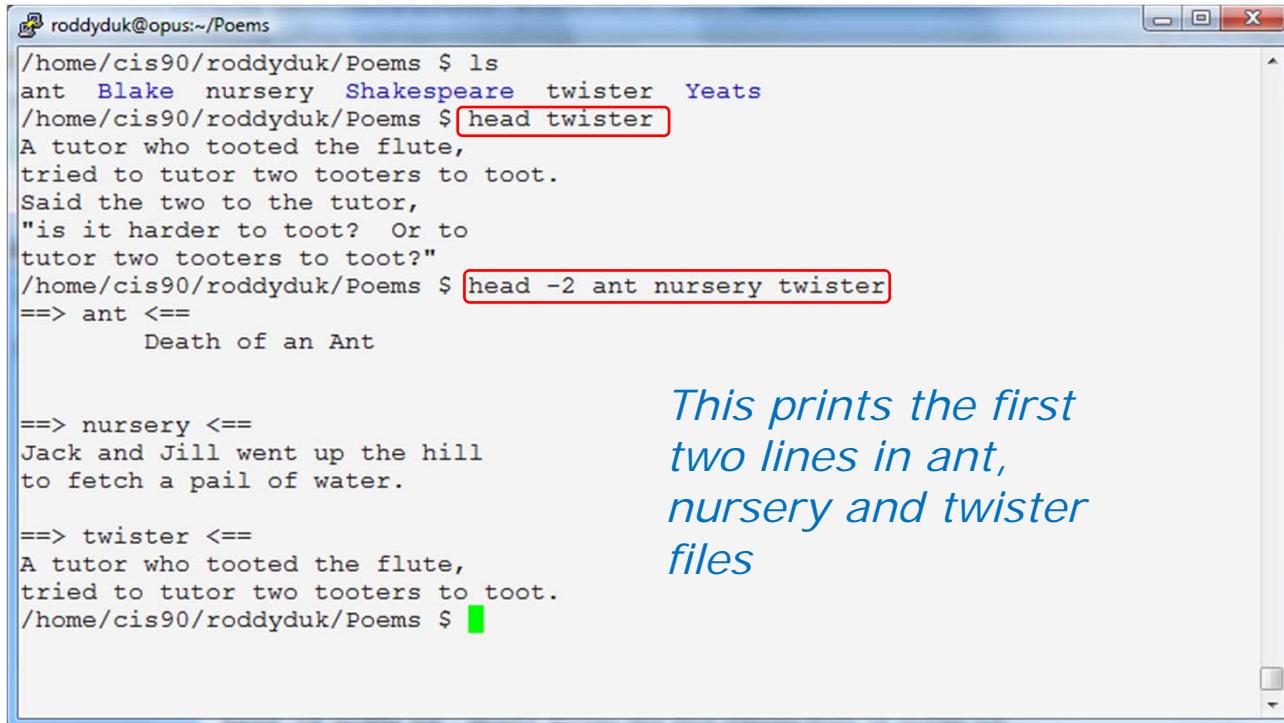
:
```

*Use the pg up/dn and up/down arrows to move through text file. Use q to quit  
(See the man page for many more options like searching)*

## head command

*print just the beginning of a text file*

- Use the **head** command to show the first several lines of a file. Use the –number option to control the number of lines printed.
- For example:



```
roddyduk@opus:~/Poems
/home/cis90/roddyduk/Poems $ ls
ant  Blake  nursery  Shakespeare  twister  Yeats
/home/cis90/roddyduk/Poems $ head twister
A tutor who tooted the flute,
tried to tutor two tootlers to toot.
Said the two to the tutor,
"is it harder to toot? Or to
tutor two tootlers to toot?"
/home/cis90/roddyduk/Poems $ head -2 ant nursery twister
==> ant <==
      Death of an Ant

==> nursery <==
Jack and Jill went up the hill
to fetch a pail of water.

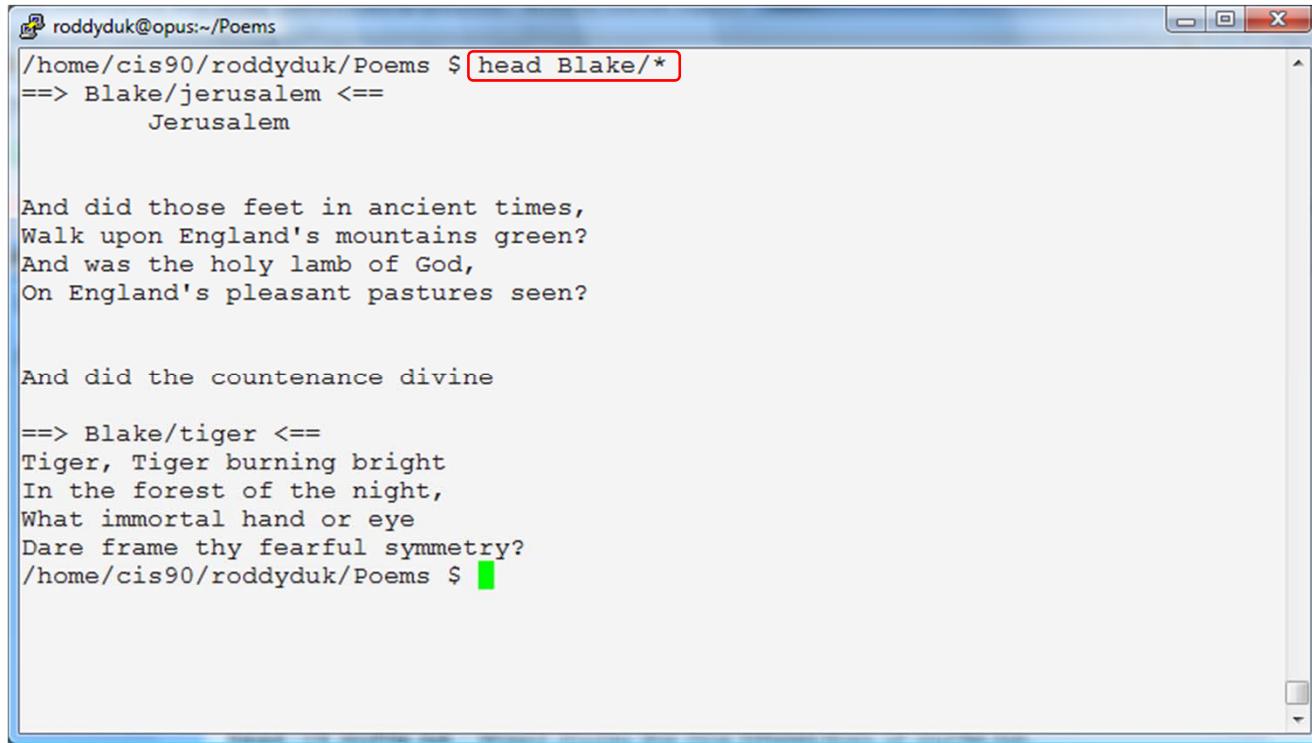
==> twister <==
A tutor who tooted the flute,
tried to tutor two tootlers to toot.
/home/cis90/roddyduk/Poems $
```

*This prints the first two lines in ant, nursery and twister files*

# head command

*print just the beginning of multiple files*

- Another example: **head Blake/\*** to print headings of all the files in the Blake directory:



```
roddyduk@opus:~/Poems
/home/cis90/roddyduk/Poems $ head Blake/*
==> Blake/jerusalem <==
    Jerusalem

And did those feet in ancient times,
Walk upon England's mountains green?
And was the holy lamb of God,
On England's pleasant pastures seen?

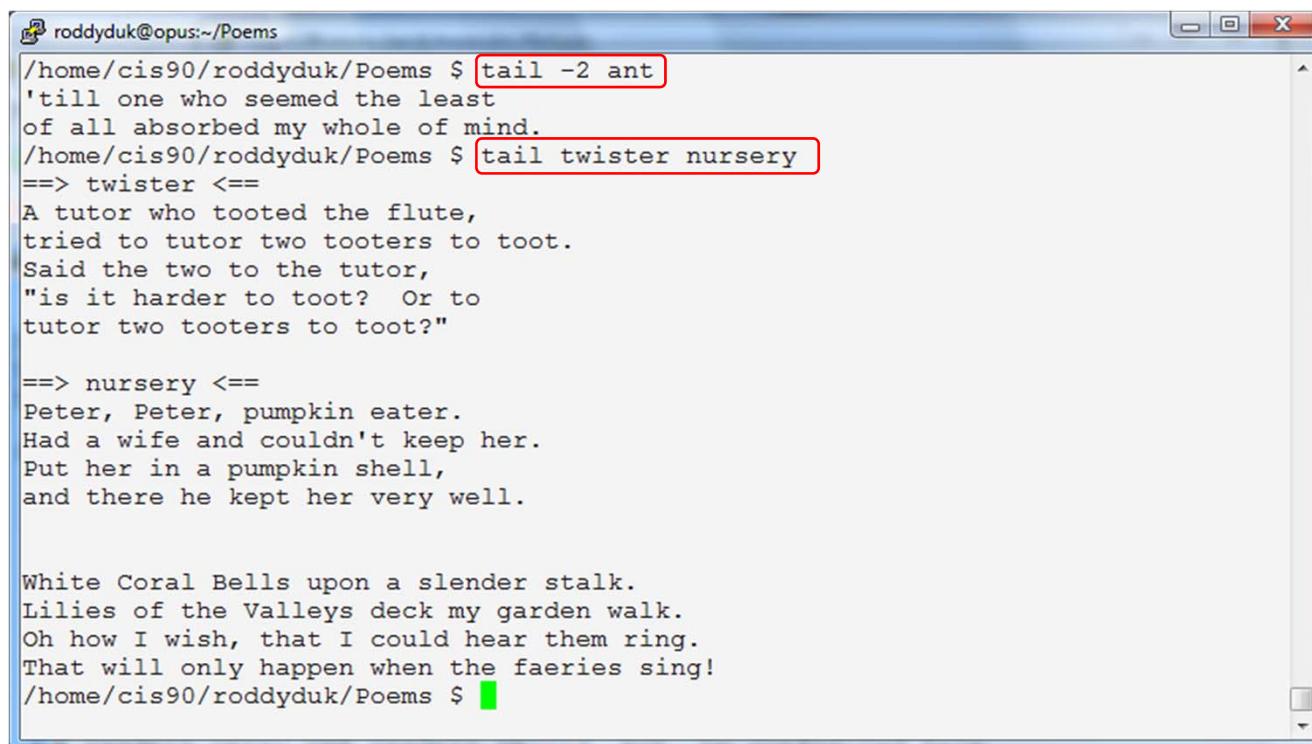
And did the countenance divine

==> Blake/tiger <==
Tiger, Tiger burning bright
In the forest of the night,
What immortal hand or eye
Dare frame thy fearful symmetry?
/home/cis90/roddyduk/Poems $
```

# tail

*print just the end of a text file*

- Use the **tail** command to print the last several lines of a file. Use the –number option to control the number of lines printed.
- For example:



```
roddyduk@opus:~/Poems
/home/cis90/roddyduk/Poems $ tail -2 ant
'till one who seemed the least
of all absorbed my whole of mind.
/home/cis90/roddyduk/Poems $ tail twister nursery
==> twister <=
A tutor who tooted the flute,
tried to tutor two tooters to toot.
Said the two to the tutor,
"is it harder to toot? Or to
tutor two tooters to toot?"

==> nursery <=
Peter, Peter, pumpkin eater.
Had a wife and couldn't keep her.
Put her in a pumpkin shell,
and there he kept her very well.

White Coral Bells upon a slender stalk.
Lilies of the Valleys deck my garden walk.
Oh how I wish, that I could hear them ring.
That will only happen when the faeries sing!
/home/cis90/roddyduk/Poems $
```

## Class Exercise

Navigate with cd, pwd, and ls

- Navigate to your Yeats directory
- Print the first line of the mooncat file with **head -1 mooncat**
- With one command print the first line of all files in the Yeats directory with **head -1 \*** or **head -n 1 \***
- Use **tail -1 mooncat** to see the last line there
- Try **tail -1 \*** to print the last lines in all Yeats poems. What happened?
- Try use **man tail**, review the n option, then try **tail -n 1 \***
- Navigate to your home directory use the **more** and **less** command to view bigfile

# binary data files

**cannot** be viewed with cat, less, head, etc.

```
rsimms@opus:~/work/examples/filetypes
[rsimms@opus filetypes]$ ls Adjective frm
Adjective.frm
[rsimms@opus filetypes]$ ls -l Adjective.frm
-rw-r--r-- 1 rsimms cis191 8983 Aug  5 07:57 Adjective.frm
[rsimms@opus filetypes]$ file Adjective.frm
Adjective.frm: MySQL table definition file Version 9
[rsimms@opus filetypes]$ cat Adjective.frm
```

Note: Adjective.frm is  
**not** a text file

```
rsimms@opus:~/work/examples/filetypes
[rsimms@opus filetypes]$ , AdjectiveID Material English Spanish HasGender AudioEng AudioSp Picture Ref
erence Example 5A 5B 6A 6B 7A 7B 8A 8B 9A Other Merida Mug X1 X2 X3 X4 X5 X6 Y
es No cat: n: No such file or directory
[rsimms@opus filetypes]$ PuTTYPuTTYPuTTY
```

Tip: Use **reset** command to fix terminal if it gets really "sick"

## xxd command

view hex dump of binary files

Example: **xxd Adjective.frm** (a MySql database schema file)

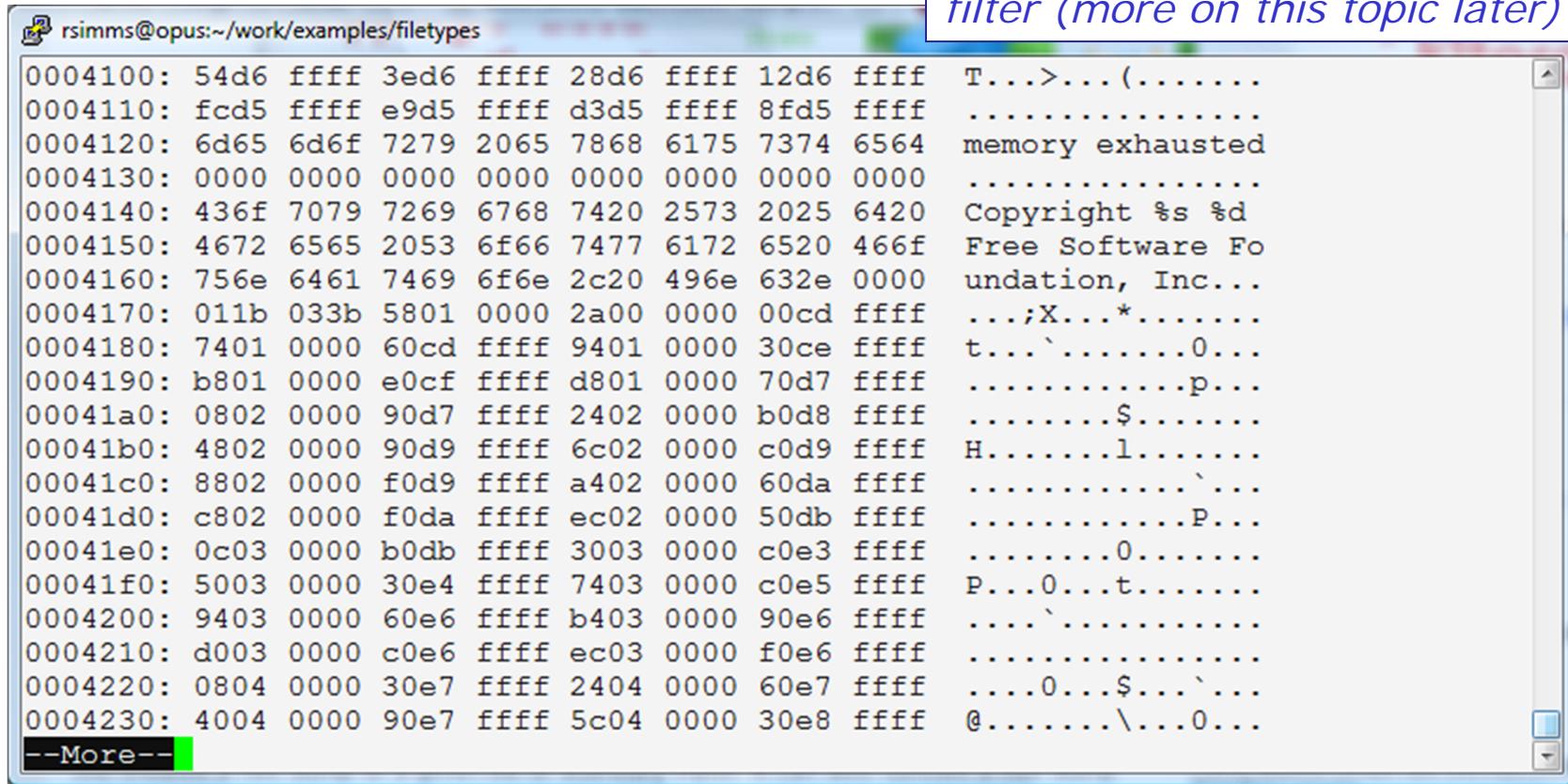
```
rsimms@opus:~/work/examples/filetypes
00021e0: 0008 8110 0001 f7c0 0000 0608 472c 0108 .....G,..
00021f0: 0000 0080 0000 00fd c000 0007 0847 2c01 .....G,.
0002200: 3401 0000 8000 0000 fdc0 0000 080a 0909 4.....
0002210: 0060 0200 0881 1000 02f7 c000 0009 0946 .`.....F
0002220: 2c01 6102 0000 8000 0000 fdc0 0000 0a08 ,.a.....
0002230: 472c 018d 0300 0080 0000 00fd c000 000b G,.....
0002240: 0847 2c01 b904 0000 8000 0000 fdc0 0000 .G,.....
0002250: 0c0a 452c 01e5 0500 0080 0000 00fd c000 ..E,.....
0002260: 000d 0847 2c01 1107 0000 8000 0000 fdc0 ...G,.....
0002270: 0000 ff41 646a 6563 7469 7665 4944 ff4d ...AdjectiveID.M
0002280: 6174 6572 6961 6cff 456e 676c 6973 68ff aterial.English.
0002290: 5370 616e 6973 68ff 4861 7347 656e 6465 Spanish.HasGende
00022a0: 72ff 4175 6469 6f45 6e67 ff41 7564 696f r.AudioEng.Audio
00022b0: 5370 ff50 6963 7475 7265 ff52 6566 6572 Sp.Picture.Refer
00022c0: 656e 6365 ff45 7861 6d70 6c65 ff00 ff35 ence.Example...5
00022d0: 41ff 3542 ff36 41ff 3642 ff37 41ff 3742 A.5B.6A.6B.7A.7B
00022e0: ff38 41ff 3842 ff39 41ff 4f74 6865 72ff .8A.8B.9A.Other.
00022f0: 4d65 7269 6461 ff4d 7567 ff58 31ff 5832 Merida.Mug.X1.X2
0002300: ff58 33ff 5834 ff58 35ff 5836 ff00 ff59 .X3.X4.X5.X6...Y
0002310: 6573 ff4e 6fff 00 es.No..
[rsimms@opus filetypes]$
```

# xxd command

## view hex dump of binary files

Example: **xxd /bin/pwd | more**

*For long files, the output of xxd can be "piped" into the more filter (more on this topic later)*



```
rsimms@opus:~/work/examples/filetypes
0004100: 54d6 ffff 3ed6 ffff 28d6 ffff 12d6 ffff T...>....(.....
0004110: fcd5 ffff e9d5 ffff d3d5 ffff 8fd5 ffff .....
0004120: 6d65 6d6f 7279 2065 7868 6175 7374 6564 memory exhausted
0004130: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0004140: 436f 7079 7269 6768 7420 2573 2025 6420 Copyright %s %d
0004150: 4672 6565 2053 6f66 7477 6172 6520 466f Free Software Fo
0004160: 756e 6461 7469 6f6e 2c20 496e 632e 0000 undation, Inc...
0004170: 011b 033b 5801 0000 2a00 0000 00cd ffff ...;X...*.....
0004180: 7401 0000 60cd ffff 9401 0000 30ce ffff t...`.....0...
0004190: b801 0000 e0cf ffff d801 0000 70d7 ffff .....p...
00041a0: 0802 0000 90d7 ffff 2402 0000 b0d8 ffff .....$.....
00041b0: 4802 0000 90d9 ffff 6c02 0000 c0d9 ffff H.....l.....
00041c0: 8802 0000 f0d9 ffff a402 0000 60da ffff .....`...
00041d0: c802 0000 f0da ffff ec02 0000 50db ffff .....P...
00041e0: 0c03 0000 b0db ffff 3003 0000 c0e3 ffff .....0.....
00041f0: 5003 0000 30e4 ffff 7403 0000 c0e5 ffff P...0...t.....
0004200: 9403 0000 60e6 ffff b403 0000 90e6 ffff .....`.....
0004210: d003 0000 c0e6 ffff ec03 0000 f0e6 ffff .....0.....
0004220: 0804 0000 30e7 ffff 2404 0000 60e7 ffff ....0...$...`...
0004230: 4004 0000 90e7 ffff 5c04 0000 30e8 ffff @.....\....0...
--More--
```

## Class Exercise

- **cd /home/cis90ol/depot/filetypes/**
- **xxd Adjective.frm**
- **xxd Adjective.frm | more**

*Using xxd to dump contents of binary file*

### Class Exercise Enlightenment

- **cd** to your home directory on Opus
- Run the enlightenment program: **enlightenment**
- Write down each magic word as you learn them.

# Wrap up

## Commands:

cat	Print a file on the screen
cd	Change directory
file	Classify a file
head	View first several lines of a file
less	Scroll up and down long files
ls	List files
more	Scroll down long files
pwd	Print working directory
reset	Use to reset terminal window
tail	View last several lines of a file
wc	Count the words, lines or characters in a file
xxd	Hex dump of a binary file

## New Files and Directories:

/	Root of the file tree
/home	Opus home directories
/home/cis90	CIS 90 class home directories
/home/cis90/ <i>username</i>	The home directory for CIS 90 student <i>username</i>

## Next Class

Assignment: Check Calendar Page on web site to see what is due next week.

Quiz questions for next class:

- 1) What are two commands you can use to read through long text files?
- 2) How do you distinguish between relative and absolute paths?
- 3) What are the three elements of a UNIX file?

# Backup

## Lab 2 Results

4. Set the TERM environment variable to "dumb", and execute the **clear** command. What does it do? Use **echo \$TERM** to see the new setting. Set TERM back to "vt100" or "ansi" What happens?

**TERM="dumb"**  
**TERM="ansi"**

Set the TERM environment variable back to "xterm" which is what it was when you logged in.

## Lab 2 Results

12. What is the difference in output between the following two commands?

**banner I am fine**

**banner "I am fine"**