Lesson Module Status
- Slides - draft
- Flash cards – done
- properties – done
- page numbers - done
- 1st minute quiz – done
- Web Calendar summary – done
- Web book pages - none
- Commands – done
- Lab tested – done

- Materials uploaded – 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

[ ] 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

Email me (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:

1) What are two commands you can use to read through long text files?

2) What are the three elements of a UNIX file?

3) How do you distinguish between relative and absolute paths?

email answers to: risimms@cabrillo.edu

Review

Objectives	Agenda
• Review Lessons 1-4 • Practice skills • Learn about filename expansion characters	• Quiz • Questions from last week • Commands (syntax, docs) • Shell • Meta characters • Filename expansion characters • Environment variables • Program to process • OS Architecture • File System • Preparing for Test 1 • Wrap up

❃ = hands on exercise for topic

Questions

Previous material and assignment

- Questions on previous material?
- Questions on any of the labs?
- Note: Lab 4 due today, email it to me at risimms@cabrillo.edu
  - Be sure and read the forum before turning in Lab 4 (or any lab for that matter).
  - Remember, you can re-submit labs as many times as you wish up till the deadline. The most recent submittal gets graded.

Housekeeping

Coming up next week

1. No lab assignment so you can prepare for the test next week

2. Practice test is available.

3. The first half of next week's lesson will be new lesson material. The second half will be the test covering material in Lesson 1-5.

Test next week

- 30 points, plus some extra credit
- 5 flashcard questions
  - Take directly from the flashcards on the web site
- 10 operational questions
  - You can verify your answers using Opus
- Open book, open notes, open computer
- To be taken during the last half of class
- Should take about 60-90 minutes, however if you need extra time, you can turn it in no later than midnight.
- PDF form format. Fill out the form, save it and email to instructor when finished.

Tips

Tips on how to answer questions on lab assignments and tests

What command will do "blah, blah, blah" questions:

Examples:

- What **ls** command-line allows you to see the permissions of your home directory while you are in your home directory?
- What command will give you a prompt showing your current working directory path and a $?
- What command allows you to see hidden files in your current directory?

Tip: Always use Opus to test your answer for these kinds of questions. If your command doesn't work on Opus it won't be the right answer!

Tips on how to answer questions on lab assignments and tests

Absolute / relative pathname questions:

Example:

- What is the relative pathname from your home directory to the **date** command?
- What is the absolute path to the sonnet1 file in your Shakespeare directory?

*Tip: Use the **ls** command with tab completion to check your absolute or relative pathnames*

```
/home/cis90/simmsben $ type date
date is /bin/date
/home/cis90/simmsben $ ls ../../../bin/date
../../../bin/date
/home/cis90/simmsben $
```

Tips on how to answer questions on lab assignments and tests

How many arguments or "parse this command" questions

Example: The shell performs file name expansion during the Parse step. When a user types the command: **file /v*/l??/*o*.[14]** on Opus, how many arguments get passed to the **file** command? What specifically are those arguments?

Tip: Use the echo command to preview how the shell will expand arguments containing metacharacters.

```
/home/cis90ol/simmsben $ echo /v*/l??/*o*.[14]
/var/log/boot.log.1 /var/log/boot.log.4 /var/log/cron.1 /var/log/cron.4
  /var/log/maillog.1 /var/log/maillog.4 /var/log/spooler.1
  /var/log/spooler.4 /var/log/yum.log.1
```

*The shell will expand **/v*/l??/*o*.[14]** into the 9 arguments shown above*

Tips on how to answer questions on lab assignments and tests

```
/home/cis90ol/simmsben $ file /v*/l??/*o*.[14]
/var/log/boot.log.1: empty
/var/log/boot.log.4: empty
/var/log/cron.1:     writable, regular file, no read permission
/var/log/cron.4:     writable, regular file, no read permission
/var/log/maillog.1:  writable, regular file, no read permission
/var/log/maillog.4:  writable, regular file, no read permission
/var/log/spooler.1:  empty
/var/log/spooler.4:  empty
/var/log/yum.log.1:  ASCII text
/home/cis90ol/simmsben $
```

*The shell expands **/v*/l??/*o*.[14]** into 9 arguments, each a specific file pathname, to be processed by the file command.*

The file command never sees the metacharacters typed by the user, it just sees the 9 arguments with are specific file pathnames.

Everything is a file

Everything is a file in UNIX
(even a terminal)

- A terminal
- A file
- A hard drive
- A hard drive partition
- A CD
- A partition on a USB flash drive
- Kernel run-time information

Implemented as files in UNIX

Everything is a file in UNIX
(even a terminal)

- A terminal — *e.g. /dev/pts/2*
- A file — *e.g. /home/cis90/simmsben/letter*
- A directory — *e.g /home/cis90/*
- A hard drive — *e.g. /dev/sda*
- A hard drive partition — *e.g. /dev/sda1*
- A CD — *e.g. /dev/cdrom*
- A partition on a USB flash drive — *e.g. /dev/sdb2*
- Kernel run-time information — *e.g. /proc/sys/kernel/hostname*

Everything is a file
(even a terminal)

```
/home/cis90/simmsben $ tty
/dev/pts/1
```

*Use the **tty** command to identify the specific terminal device being used*

Note this device is identified using a pathname

```
/home/cis90/simmsben $ echo $TERM
xterm
```

Use the TERM variable to identify the specific type of terminal being used

Everything is a file
(even a terminal)

```
/home/cis90/simmsben $ tty
/dev/pts/1
```

Show which terminal you are using

```
/home/cis90/simmsben $ echo $TERM
xterm
```

Show what kind of terminal you are using

```
/home/cis90/simmsben $ who
simmsben pts/1        2010-09-29 07:38 (dsl-49-64-10-90.dhcp.cruzio.com)
srecklau pts/2        2010-09-29 06:06 (62.143.60.194)
rsimms   pts/4        2010-09-29 06:47 (dsl-49-64-10-90.dhcp.cruzio.com)
```

Use who to see who is logged in

```
/home/cis90/simmsben $ ls -l /dev/pts/*
crw--w---- 1 simmsben tty 136, 1 Sep 29 07:45 /dev/pts/1
crw--w---- 1 srecklau tty 136, 2 Sep 29 07:44 /dev/pts/2
crw--w---- 1 rsimms   tty 136, 4 Sep 29 06:48 /dev/pts/4
```

Do a long listing to see all the terminal devices in use

Notice the owner is someone who has logged in

Notice the file type is "c" which is a character device file

File Types and Commands

Long listing code (ls –l)	Type	How to make one
d	directory	mkdir
-	regular • Programs • Text • Data (binary)	touch
l	symbolic link	ln -s
c	character device files	mknod
b	block device files	mknod

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

Everything is a file in UNIX
(even a terminal)

Nice things about files

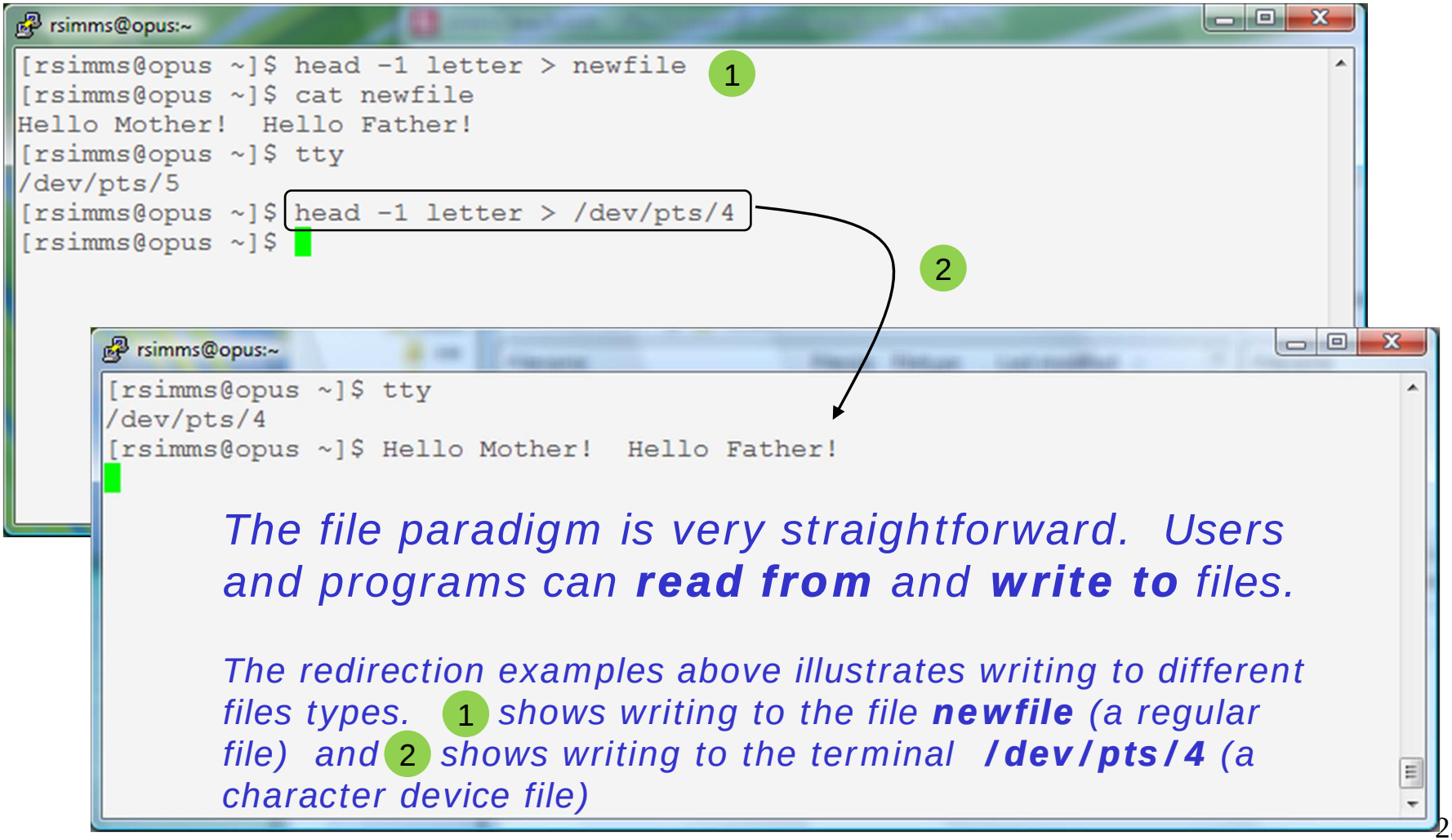
- you can write to them

```
[rsimms@opus ~]$ echo "Rich was here" > myfile
```

- and read from them

```
[rsimms@opus ~]$ cat myfile
Rich was here
```

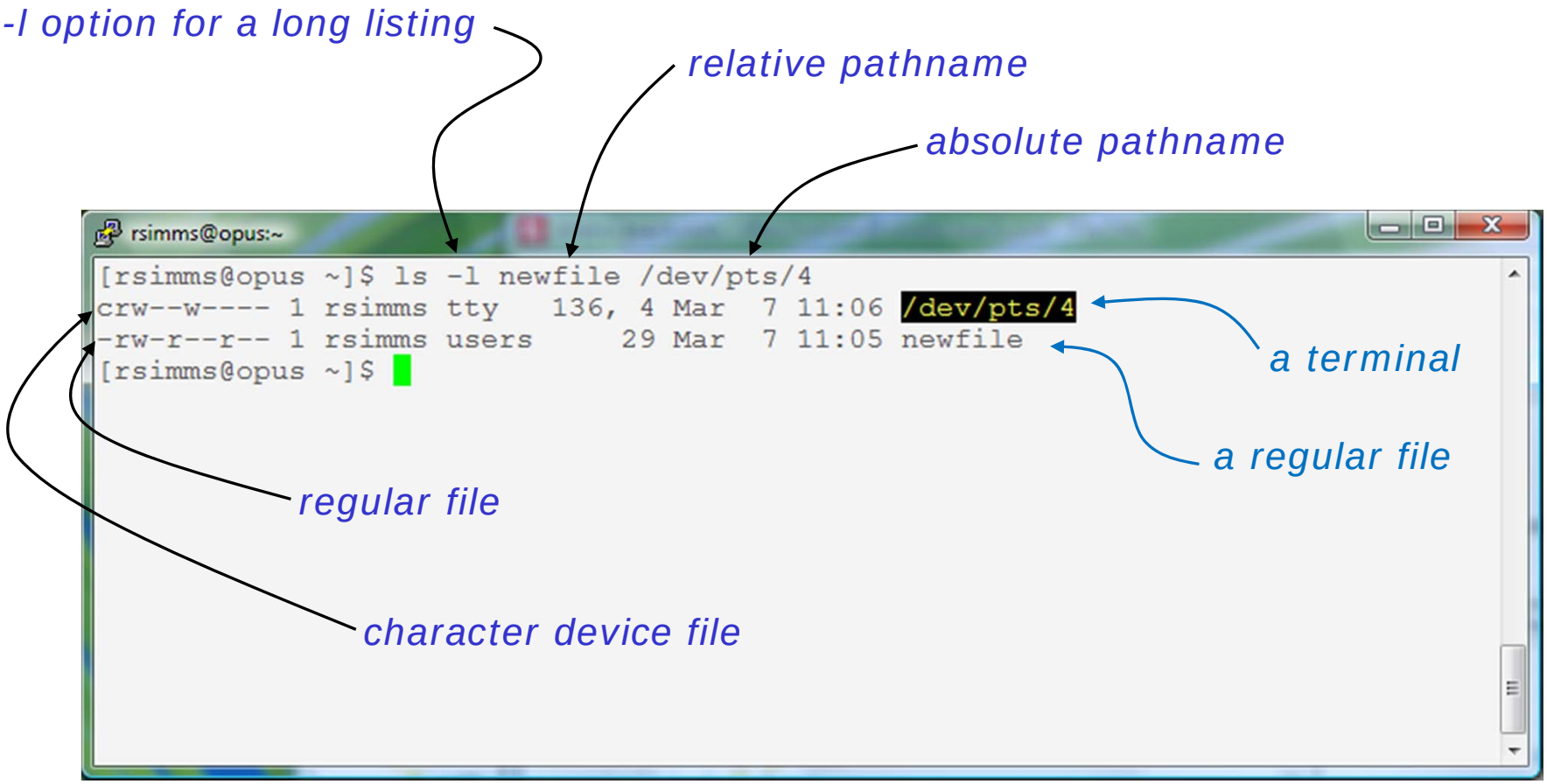
Everything is a file in UNIX
(even a terminal)



*The file paradigm is very straightforward. Users and programs can **read from** and **write to** files.*

*The redirection examples above illustrates writing to different files types. 1 shows writing to the file **newfile** (a regular file) and 2 shows writing to the terminal **/dev/pts/4** (a character device file)*

Everything is a file
(even a terminal)



Class Exercise

- Login into Opus using Putty
- Use **echo "Hello Hugo" > myfile**
- Print your new file with **cat myfile**
- Open a second Putty session and login into Opus
- You should have two terminals now (two Putty windows)
- Use **tty** to identify your terminals
- In one terminal use **echo "Hello Hugo" > /dev/pts/xx** where xx is your other terminal

Command Review

*Use the **man** command or google for the details*

New commands:

cal	- show calendars
clear	- clear the terminal screen
exit	- terminate your shell and log off
history	- show previous commands
hostname	- show the name of the computer being accessed
id	- show user and group id information
ps	- show processes (loaded programs) being run
ssh	- secure login to a remote system
uname	- show OS name
tty	- show terminal information
who	- show who else is logged on
Ctrl-Alt-F1 to Ctrl-Alt-F7	- Change between terminals and X windows (graphics)

New Files and Directories:

VMware:

Ctrl-Alt	- to move mouse cursor out of VM

*Use the **man** command or google for the details*

New commands:

apropos	- search for string in whatis database
bc	- binary calculator
cat	- print file(s)
cd	- change directory
echo	- print text
env	- show shell environment variables
info	- online documentation with hot links
file	- show file information
ls	- show directory contents
passwd	- change password
set	- show (or set) shell variables
type	- show command location in path
man	- manual page for a command
whatis	- command summary

New Files and Directories:

/etc/passwd	- user accounts
/etc/shadow	- encrypted passwords
/bin	- directory of commands
/sbin	- directory of superuser commands
/usr/bin	- directory of commands, tools and utilities
/usr/sbin	- directory of superuser commands, tools and utilities

New commands:

mail	- UNIX mail
?	print these commands
p <message list>	print messages
n	goto and print next message
e <message list>	edit messages
d <message list>	delete messages
s <message list> file	save (append) messages to file
u <message list>	undelete messages
R <message list>	reply to sender(s)
r <message list>	reply to all
m <user list>	mail to specific users
q	quit, saving read messages to local mbox file
x	quit, mark all mail as unread and undeleted.
h	print out active message headers
mesg	- Enable or disable writes to your terminal
write	- Write message to another user

New Files and Directories:

/var/mail	- Message store for mail
/var/mail/*username*	- Incoming mailbox for *username*
mbox	- File in users home directory where read messages are archived to

*Use the **man** command or google for the details*

*Use the **man** command or google for the details*

Commands:

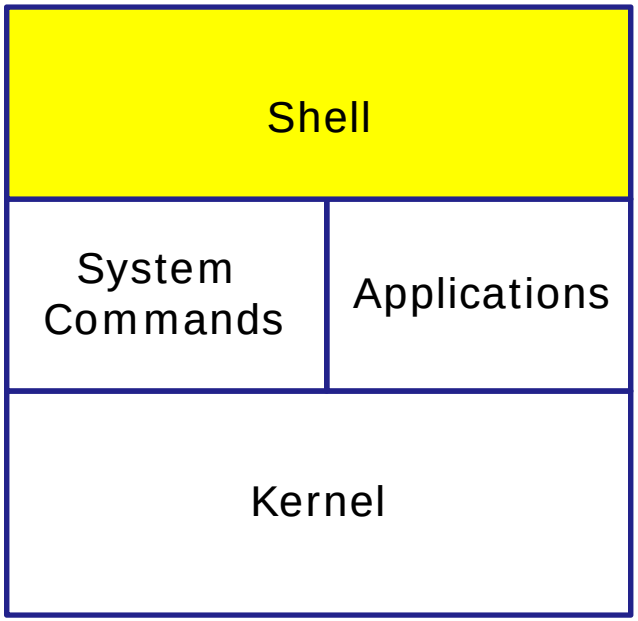
cat	Print a file on the screen
cd	Change directory
file	Classify a file
head	View first several lines of a file
less	Scroll up and down long files
ls	List files
more	Scroll down long files
pwd	Print working directory
reset	Use to reset terminal window
tail	View last several lines of a file
wc	Count the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

New Files and Directories:

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

Command line Prompt Parse

Life of the Shell



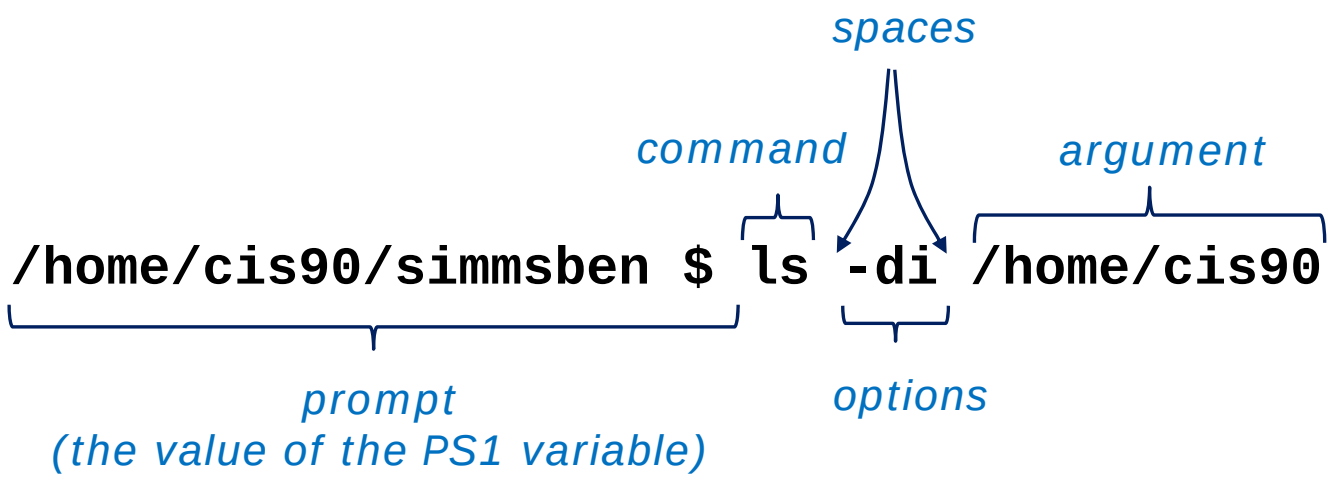
1) **Prompt** for a command
2) **Parse** (interpret metacharacters, expand file names and dissect command line into options and arguments)
3) **Search** for program (along the path)
4) **Execute** program by loading into memory (becomes a process), hookup input and outputs, and pass along command line options and arguments.
5) **Nap** (wait till process is done)
6) **Repeat**

Command Syntax

Command	Options	Arguments	Redirection

Command – is the name of an executable program file.
Options – various options which control how the program will operate.
Arguments – the objects the command is directed to work upon.
Redirection – The default input stream (stdin) is from the console keyboard, the default output (stdout) and error (stderr) streams go to the console screen. Redirection can modify these streams to other files or devices.

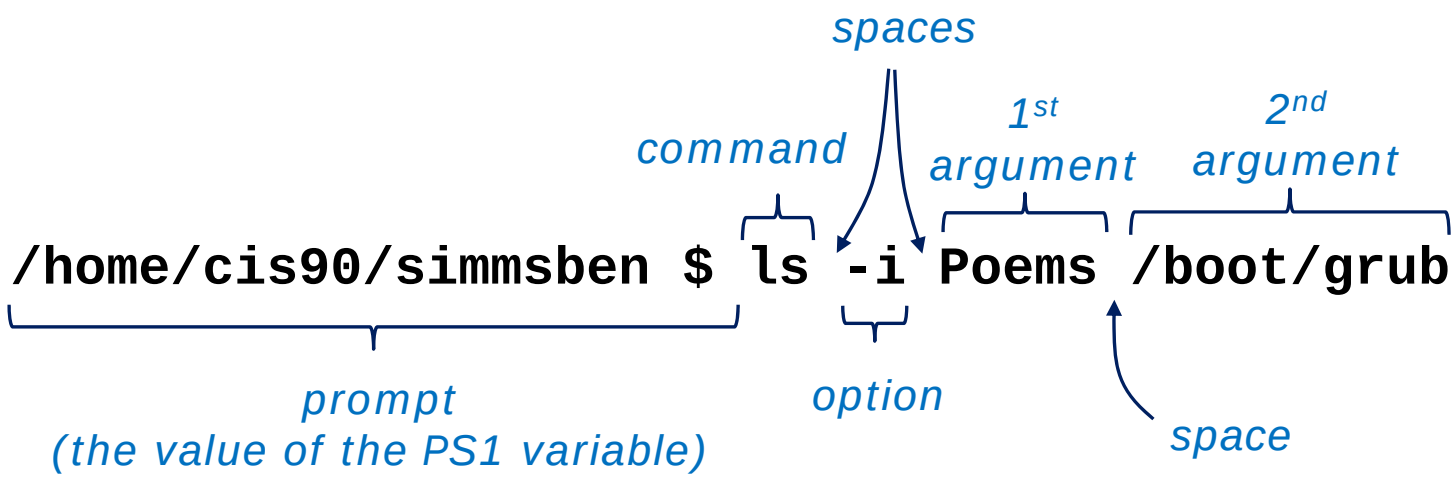
Command Line Syntax Review



Parsing the command line above yields:

One command: **ls**
Two options: **d** and **i**
One argument: **/home/cis90** (an absolute pathname to a directory)

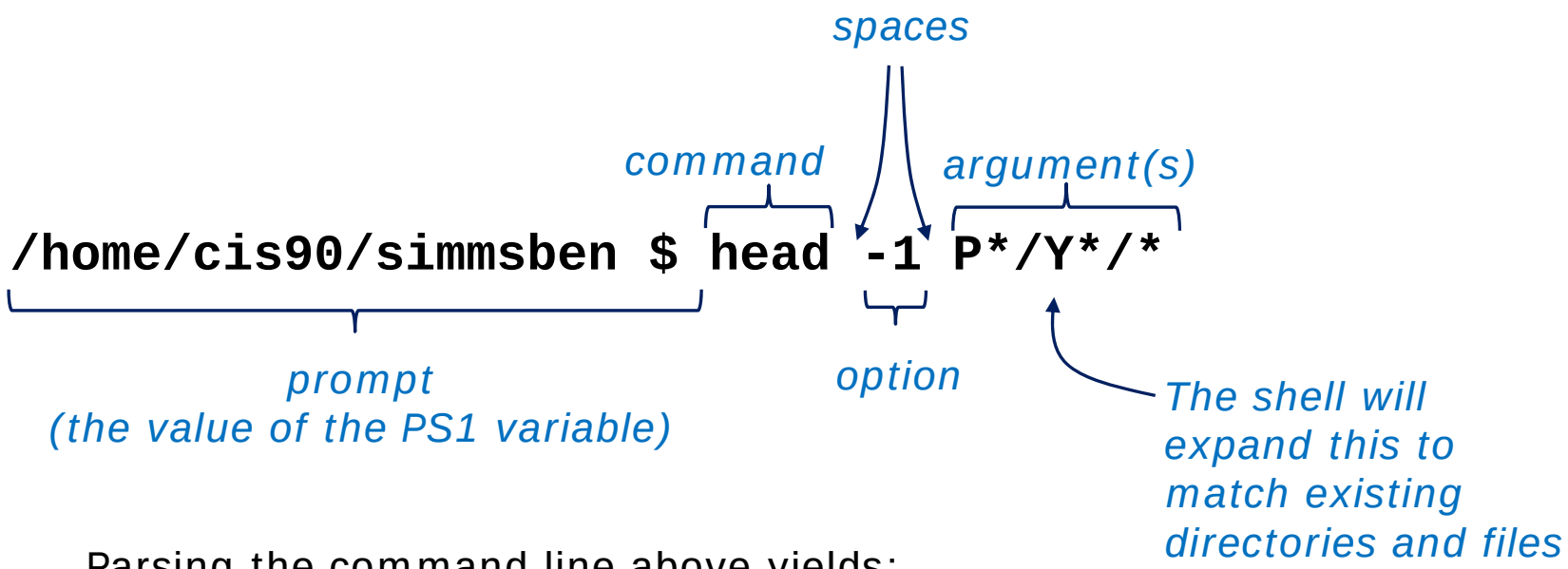
Command Line Syntax Review



Parsing the command line above yields:

One command: **ls**
One options: **i**
Two arguments:
Poems (a relative pathname to a directory)
/boot/group (an absolute pathname to a directory)

Command Line Syntax Review



Parsing the command line above yields:

One command: **head**
One option: **1**
Three arguments:

Poems/Yeats/mooncat (a relative pathname to a file)
Poems/Yeats/old (a relative pathname to a file)
Poems/Yeats/whitebirds (a relative pathname to a file)

Your turn now!

/home/cis90ol/simmsben $ ls -ls /usr/bin/ls*

1) What portion of the line above is the shell prompt?

/home/cis90ol/simmsben $

2) Parse the command the user typed and identify:

The name of the program/script to run: **ls**

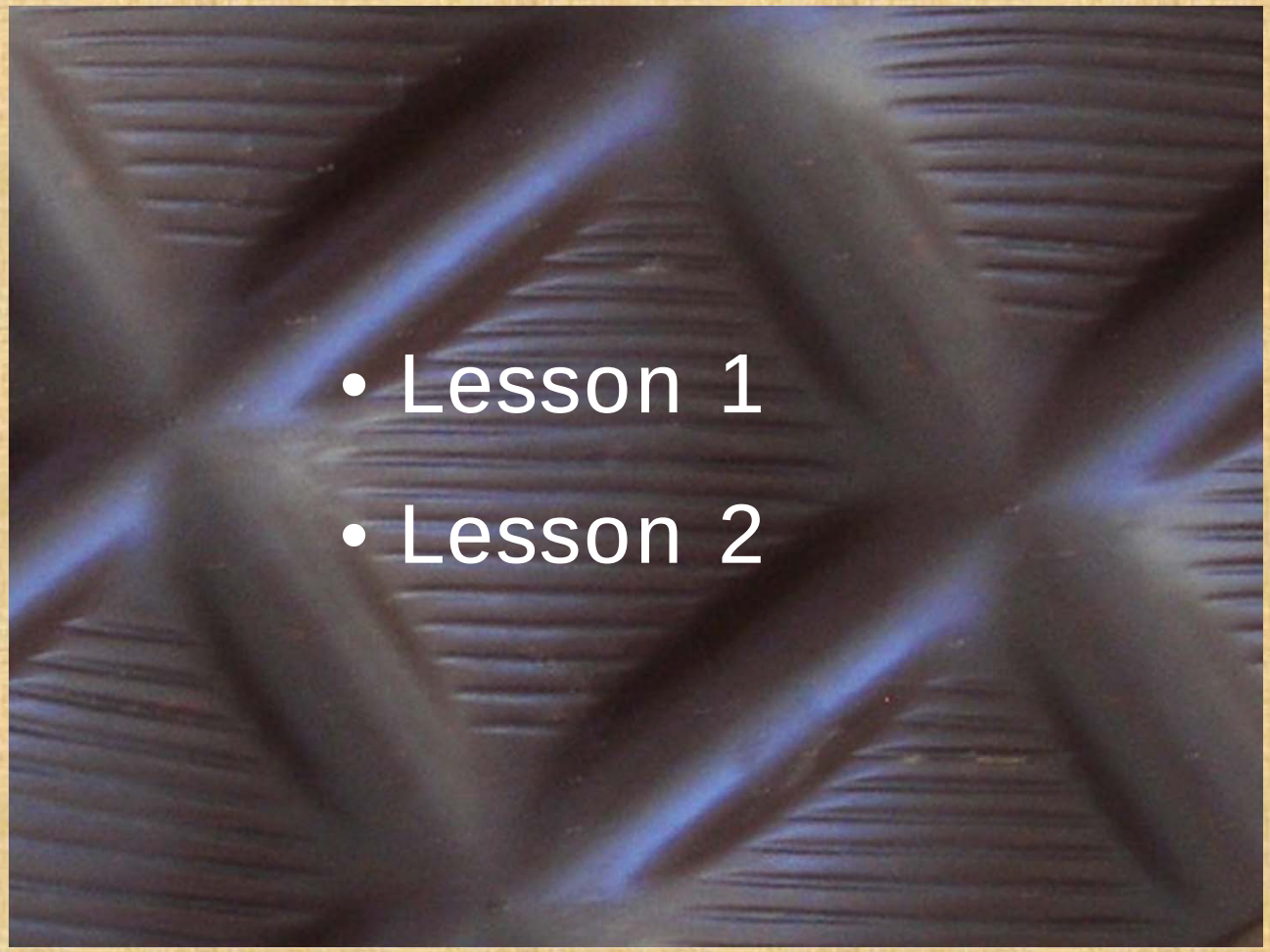
2 options: **l** and **s** (long and size in blocks)

6 arguments:

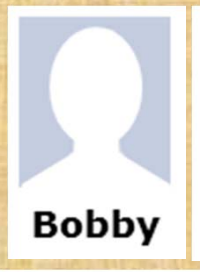
/usr/bin/lsattr
/usr/bin/lsb_release
/usr/bin/lsdiff
/usr/bin/lshal
/usr/bin/lspgpot
/usr/bin/lss16toppm

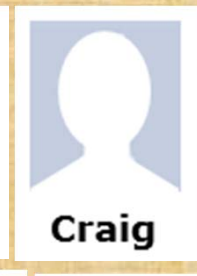
Class Exercise

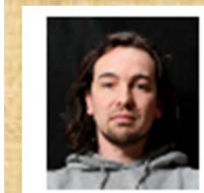
Flashcards

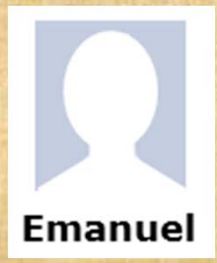


Team debian 10


Bobby


Craig


Daniel W


Emanuel


Gabriel


Jason


Josh


Marisol


Quinton


Tanner


Tajvia

Team Mandriva 10


Chris


Dan M


David


Eric


Geoffrey


Jeff


Jesse


Merrick


Tommy


Terence


Yu-Chen

Meta Characters (review)

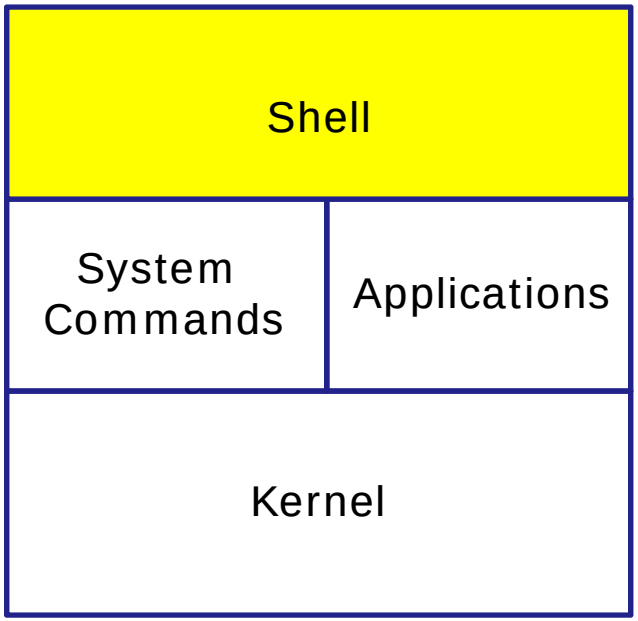
Metacharacters

Have special interpretation by the shell

Char	Description
\	Treat the following metacharacter as a plain character. Also called "escaping" the next character.
$	The following text is a shell (environment) variable and the value should be used.
<cr>	Carriage return marks the end of the command
;	Separates multiple commands on one line
'	used to enclose a string that the shell will not do further interpretation
"	Used to enclose a string that the shell will do further interpretation.
>	Redirects stdout (more in Lesson 8)
2>	Redirects stderr (more in Lesson 8)
*	Matches all non-hidden file names when used alone or zero or more characters when used as prefix, infix or postfix
?	Matches any single character of a file name
[]	Matches any single character contained within the brackets
#	Not an official metacharacter, but any text following the # is ignored by the shell

Life of the Shell

*The shell processes metacharacters during the **Parse** step*



1) **Prompt** for a command
2) **Parse** (interpret metacharacters, expand file names and dissect command line into options and arguments)
3) **Search** for program (along the path)
4) **Execute** program by loading into memory (becomes a process), hookup input and outputs, and pass along command line options and arguments.
5) **Nap** (wait till process is done)
6) **Repeat**

Metacharacters
#

 # has the ability to make everything that follows the # be ignored by the shell. Good for adding comments in scripts

```
/home/cis90/simmsben $ #OK lets escape the carriage return in next example
/home/cis90/simmsben $
```

Note there is no error message because everything after the # is ignored

Metacharacters
$

$ metacharacter has the ability to "show the value of"

```
/home/cis90/simmsben $ EYES=brown
/home/cis90/simmsben $ echo EYES
EYES
/home/cis90/simmsben $ echo $EYES
brown

/home/cis90/simmsben $ echo $LOGNAME
simmsben
/home/cis90/simmsben $
```

echo the string EYES

echo the value of the variable EYES

echo the value of the predefined environment variable LOGNAME

Metacharacters
" and '

Weak "double" quotes allow the shell to process $ metacharacters inside the quoted string

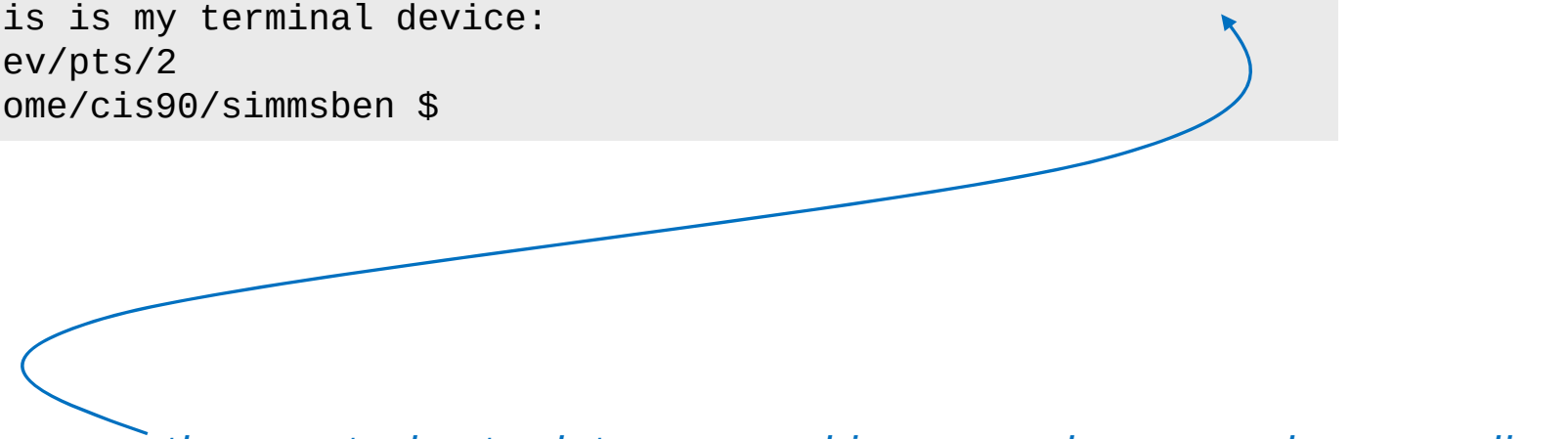
```
/home/cis90/simmsben $ echo "I am in $PWD"
I am in /home/cis90/simmsben


/home/cis90/simmsben $ echo 'I am in $PWD'
I am in $PWD
/home/cis90/simmsben $
```

Strong "single" quotes block the shell from processing $ metacharacters inside the quoted string

Metacharacters

;

```
/home/cis90/simmsben $ #Lets put two commands on one line
/home/cis90/simmsben $ echo "This is my terminal device:"; tty
This is my terminal device:
/dev/pts/2
/home/cis90/simmsben $
```



the ; metachacter lets you combine several commands on one line

Metacharacters

\

```
/home/cis90/simmsben $ #OK lets escape the carriage return in next example
/home/cis90/simmsben $ echo Lets start line 1 here \
> and finish it here
Lets start line 1 here and finish it here
/home/cis90/simmsben $
```

The \ is used to escape the next character typed. Use an escape to disable the special abilities of a metacharacter.

Escaping a carriage return (the Enter key) tells the shell to keeping inputting more characters from the next line for the current command being entered.

Metacharacters

\

Escaping the # means it is no longer treated as comment

```
/home/cis90/simmsben $ \#OK lets put a comment here
-bash: #OK: command not found
/home/cis90/simmsben $
/home/cis90/simmsben $
/home/cis90/simmsben $ echo $PS1
$PWD $
/home/cis90/simmsben $ echo \$PS1
$PS1
/home/cis90/simmsben $
```

and you get an error when the shell processes your comment

Escaping the $ means $ is no longer treated "the value of"

Class Exercise

- Use the # metacharacter
  #this is just a comment

- Use the $ and ; metacharacter
  echo $LOGNAME; echo LOGNAME

- Use the \ metacharacter
  \#This is not a comment

- Use strong an weak quotes metacharacters
  echo "My username is $LOGNAME"
  echo 'Use $LOGNAME to show your username'

File Name Expansion (more)

Filename Expansion Characters

More metacharacters for making file name wildcards

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

? matches any single character in any of your current directory's filenames.

[] matches any single character contained within the brackets.

Metacharacters

*

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

*The * metacharacter can be used to match the filenames in your current working directory*

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

*During the Parse step the shell replaces the * with the names of the files in the current directory.*

*The **echo** command above never sees the *, instead it gets all the matched filenames as arguments .*

Metacharacters

*

echo *

is modified by the shell to be as if the user typed in the following instead:

echo bigfile bin delete empty Hidden Lab2.0 Lab2.1 letter Miscellaneous mission Poems proposal1 proposal2 proposal3 small_town spellk text.err text.fxd timecal what_am_i

(all on one line)

Filename expansion happens during the shell parsing step, before the command is even located or executed.

1) Prompt
2) Parse
3) Search for program (along the path)
4) Execute program
5) Nap (wait till process is done)
6) Repeat

Metacharacters

*

*Note the * metacharacter by itself does not match any hidden files in your current working directory*

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


/home/cis90/simmsben $ ls -a
.               .bashrc   empty     letter          Poems       spellk
.zshrc
..              bigfile   Hidden    Miscellaneous   proposal1   text.err
.bash_history   bin       Lab2.0    mission         proposal2   text.fxd
.bash_logout    delete    Lab2.1    .mozilla        proposal3   timecal
.bash_profile   .emacs    .lesshst  .plan           small_town  what_am_i
```

Metacharacters

*

```
/home/cis90/simmsben $ echo *.*
Lab2.0 Lab2.1 text.err text.fxd
```

*Note, DOS uses *.* to match all files.*

BUT, this does not work the same way in UNIX and instead matches only files containing a period

Metacharacters

File name expansion characters

Char	Description
*	Matches all non-hidden file names when used alone or zero or more characters when used as prefix, infix or postfix
?	Matches any single character of a file name
[]	Matches any single character contained within the brackets

Metacharacters

*

Char	Description
*	Matches all non-hidden file names when used alone or zero or more characters when used as prefix, infix or postfix
?	Matches any single character of a file name
[]	Matches any single character contained within the brackets

```
/home/cis90/simmsben/Poems $ ls -a
.  ..  ant  Blake  nursery  Shakespeare  twister  Yeats
/home/cis90/simmsben/Poems $ echo *
ant Blake nursery Shakespeare twister Yeats
```

All non-hidden files in current directory

Metacharacters

*

Char	Description
*	Matches all non-hidden file names when used alone or zero or more characters when used as prefix, infix or postfix
?	Matches any single character of a file name
[]	Matches any single character contained within the brackets

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


/home/cis90/simmsben/Poems $ echo a*
ant
```

All non-hidden files starting with an "a"

Metacharacters

*

Char	Description
*	Matches all non-hidden file names when used alone or zero or more characters when used as prefix, infix or postfix
?	Matches any single character of a file name
[]	Matches any single character contained within the brackets

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


/home/cis90/simmsben/Poems $ echo ../p*
../proposal1 ../proposal2 ../proposal3
```

All files in parent directory starting with a "p"

Metacharacters

?

Char	Description
*	Matches all non-hidden file names when used alone or zero or more characters when used as prefix, infix or postfix
?	Matches any single character of a file name
[]	Matches any single character contained within the brackets

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

/home/cis90/simmsben/Poems $ echo B???e
Blake
```

All five letter file names starting with "B" and ending with an "e"

Metacharacters
[]

Char	Description
*	Matches all non-hidden file names when used alone or zero or more characters when used as prefix, infix or postfix
?	Matches any single character of a file name
[]	Matches any single character contained within the brackets

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

/home/cis90/simmsben/Poems $ echo [SB]*
Blake Shakespeare
```

All files names starting with "S" or "B"

Metacharacters

Filename expansion metacharacters

Tip: Use the echo command to verify how bash will do an expansion

```
/home/cis90/simmsben/Poems $ echo [SB]*
Blake Shakespeare
```

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

/home/cis90/simmsben/Poems $ echo B???e
Blake
```

Class Exercise

- Change to your home directory
- Use the **file** command on all files starting with prop
  file prop*
- Print the headings of all files starting with l or t
  head [lt]*
- Use **ls** command to list only 3 character filenames in /bin and sort by size
  ls -lS /bin/???
- Make up your own wildcard using *, [], and ? in one command

Environment Variables (review)

Shell (Environment) Variables

common environment variables

Shell Variable	Description
HOME	Users home directory (starts here after logging in and returns with a cd command (with no arguments)
LOGNAME	User's username for logging in with.
PATH	List of directories, separated by :'s, for the Shell to search for commands (which are program files) .
PS1	The prompt string.
PWD	Current working directory
SHELL	Name of the Shell program being used.
TERM	Type of terminal device , e.g. dumb, vt100, xterm, ansi, etc.

Shell (Environment) Variables

Show variable values

Use echo to show the values of one or more variables

```
/home/cis90/simmsben/Poems $ # Print some of the shell variables
/home/cis90/simmsben/Poems $ echo $HOME $LOGNAME $PS1 $PWD $SHELL $TERM
/home/cis90/simmsben simmsben $PWD $ /home/cis90/simmsben/Poems /bin/bash
xterm

/home/cis90/simmsben/Poems $ echo $PATH
/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/cis90/simmsben/../bin:/ho
me/cis90/simmsben/bin:.
```

Shell (Environment) Variables

Set variable values

Use an "=" with no spaces to set values of variables

```
/home/cis90/simmsben/Poems $ # Change the prompt variable
/home/cis90/simmsben/Poems $ PS1='[\u@\h \W]\$'


[simmsben@opus Poems]$ # Change it back again
[simmsben@opus Poems]$ PS1='$PWD $ '
```

Shell (Environment) Variables

env command – show all environment variables

```
/home/cis90/simmsben/Poems $ env
HOSTNAME=opus.cabrillo.edu
SHELL=/bin/bash
TERM=xterm
HISTSIZE=1000
USER=simmsben
LS_COLORS=no=00:fi=00:di=00;34:ln=00;36:pi=40;33:so=00;35:bd=40;33;01:cd=40;33;01:or=01;05;37;41:mi
=01;05;37;41:ex=00;32:*.cmd=00;32:*.exe=00;32:*.com=00;32:*.btm=00;32:*.bat=00;32:*.sh=00;32:*.csh=
00;32:*.tar=00;31:*.tgz=00;31:*.arj=00;31:*.taz=00;31:*.lzh=00;31:*.zip=00;31:*.z=00;31:*.Z=00;31:*
.gz=00;31:*.bz2=00;31:*.bz=00;31:*.tz=00;31:*.rpm=00;31:*.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=
00;35:*.xbm=00;35:*.xpm=00;35:*.png=00;35:*.tif=00;35:
USERNAME=
MAIL=/var/spool/mail/simmsben
PATH=/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/cis90/simmsben/../bin:/home/cis90/simmsbe
n/bin:.
INPUTRC=/etc/inputrc
PWD=/home/cis90/simmsben/Poems
LANG=en_US.UTF-8
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
SHLVL=1
HOME=/home/cis90/simmsben
BASH_ENV=/home/cis90/simmsben/.bashrc
LOGNAME=simmsben
CVS_RSH=ssh
LESSOPEN=|/usr/bin/lesspipe.sh %s
G_BROKEN_FILENAMES=1
_=/bin/env
OLDPWD=/home/cis90/simmsben
/home/cis90/simmsben/Poems $
```

*Use the **env** command to show all environment variables (a subset of the shell variables)*

Shell Variables

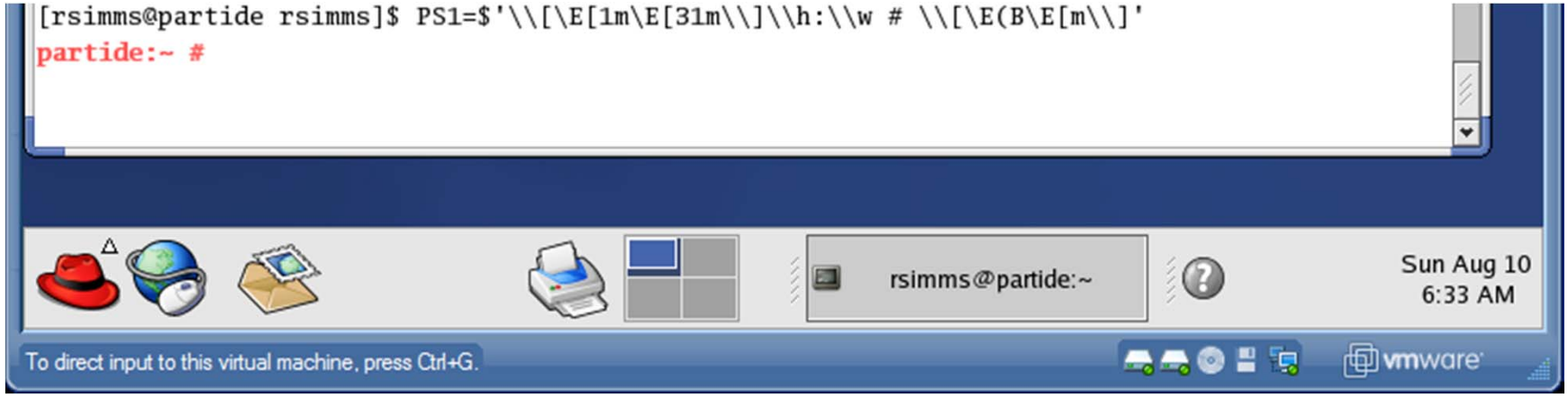
set command – show all shell variables

```
/home/cis90/simmsben/Poems $ set
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
BASH_ENV=/home/cis90/simmsben/.bashrc
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="3" [1]="2" [2]="25" [3]="1"
[4]="release" [5]="i686-redhat-linux-gnu")
BASH_VERSION='3.2.25(1)-release'
COLORS=/etc/DIR_COLORS.xterm
COLUMNS=80
CVS_RSH=ssh
DIRSTACK=()
EUID=1160
GROUPS=()
G_BROKEN_FILENAMES=1
HISTFILE=/home/cis90/simmsben/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/home/cis90/simmsben
HOSTNAME=opus.cabrillo.edu
HOSTTYPE=i686
IFS=$' \t\n'
IGNOREEOF=10
INPUTRC=/etc/inputrc
LANG=en_US.UTF-8
LESSOPEN='|/usr/bin/lesspipe.sh %s'
LINES=24
LOGNAME=simmsben
LS_COLORS='no=00:fi=00:di=00;34:ln=00;36:pi=40;33:so=00;35
:bd=40;33;01:cd=40;33;01:or=01;05;37;41:mi=01;05;37;41:ex=
00;32:*.cmd=00;32:*.exe=00;32:*.com=00;32:*.btm=00;32:*.ba
t=00;32:*.sh=00;32:*.csh=00;32:*.tar=00;31:*.tgz=00;31:*.a
rj=00;31:*.taz=00;31:*.lzh=00;31:*.zip=00;31:*.z=00;31:*.Z
=00;31:*.gz=00;31:*.bz2=00;31:*.bz=00;31:*.tz=00;31:*.rpm=
00;31:*.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=00;35:*.x
bm=00;35:*.xpm=00;35:*.png=00;35:*.tif=00;35:'
MACHTYPE=i686-redhat-linux-gnu
MAIL=/var/spool/mail/simmsben
MAILCHECK=60
OLDPWD=/home/cis90/simmsben
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/
cis90/simmsben/../bin:/home/cis90/simmsben/bin:.
PIPESTATUS=([0]="0")
PPID=26514
PROMPT_COMMAND='echo -ne
"\033]0;${USER}@${HOSTNAME%%.*}:${PWD/#$HOME/~}"; echo -ne
"\007"'
PS1='$PWD $'
PS2='> '
PS4='+ '
PWD=/home/cis90/simmsben/Poems
SHELL=/bin/bash
SHELLOPTS=braceexpand:emacs:hashall:histexpand:ignoreeof:i
nteractive-comments:monitor
SHLVL=1
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
TERM=xterm
UID=1160
USER=simmsben
USERNAME=
_=env
consoletype=pty
```

*Use the **set** command to show all shell variables (which includes the environment variables)*

bash shell tip

"wild" openSUSE root prompt applied on RH9



Some prompt strings (which are based on the PS1 environment variable) get pretty fancy!

Class Exercise

- Change your prompt with:
  PS1='$LOGNAME, command please: '

- Change your prompt with:
  PS1='[\u@\h \W]\$ '

- Change your prompt with:
  PS1="$PWD $ "
  Now change directories using **cd**, what happenned?

- Restore original prompt with:
  PS1='$PWD $ '

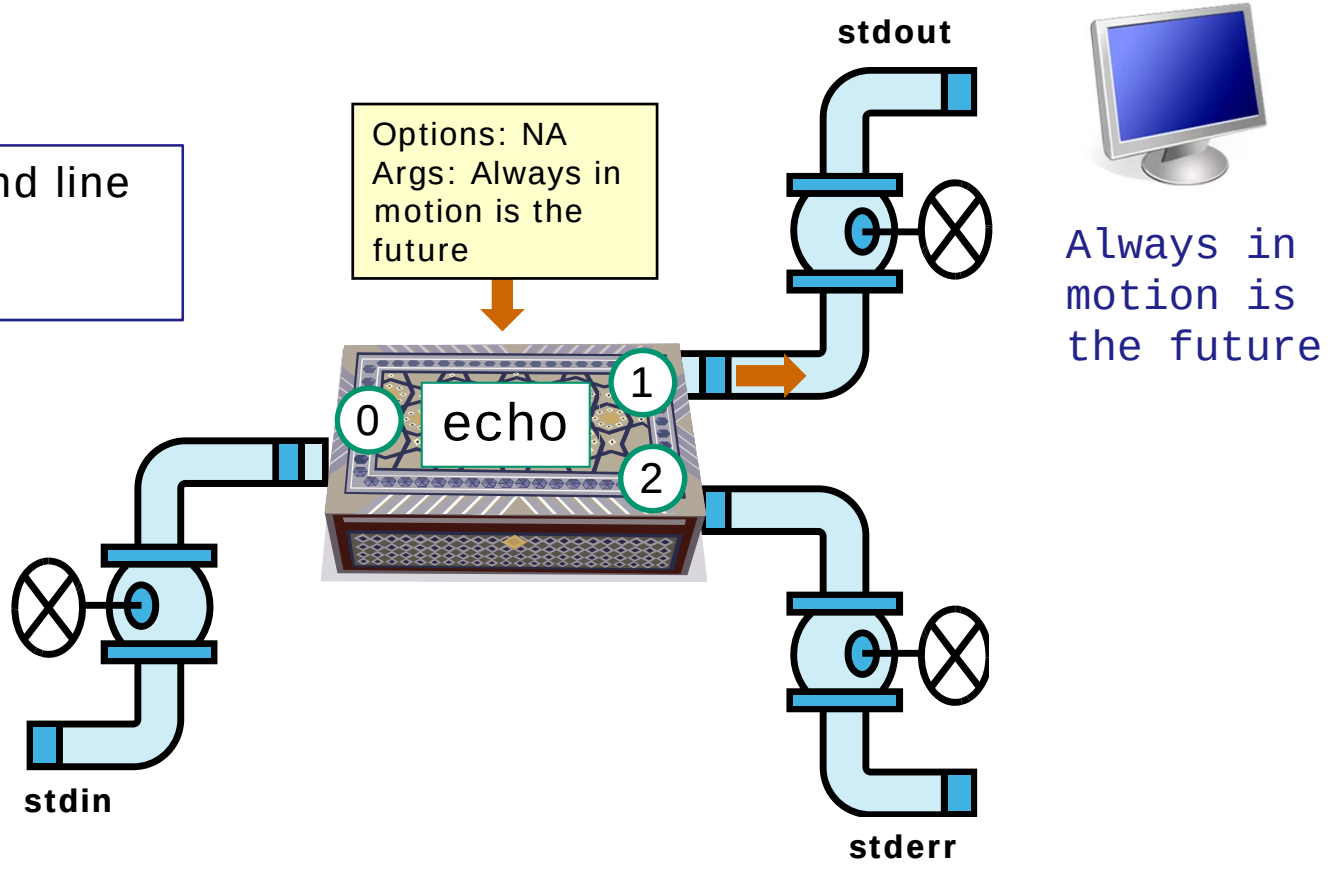
Program to Process (continuing)

Example program to process: echo command

```
[rsimms@opus ~]$ echo Always in motion is the future
Always in motion is the future
[rsimms@opus ~]$
```

Inputs: Command line

Outputs: stdout

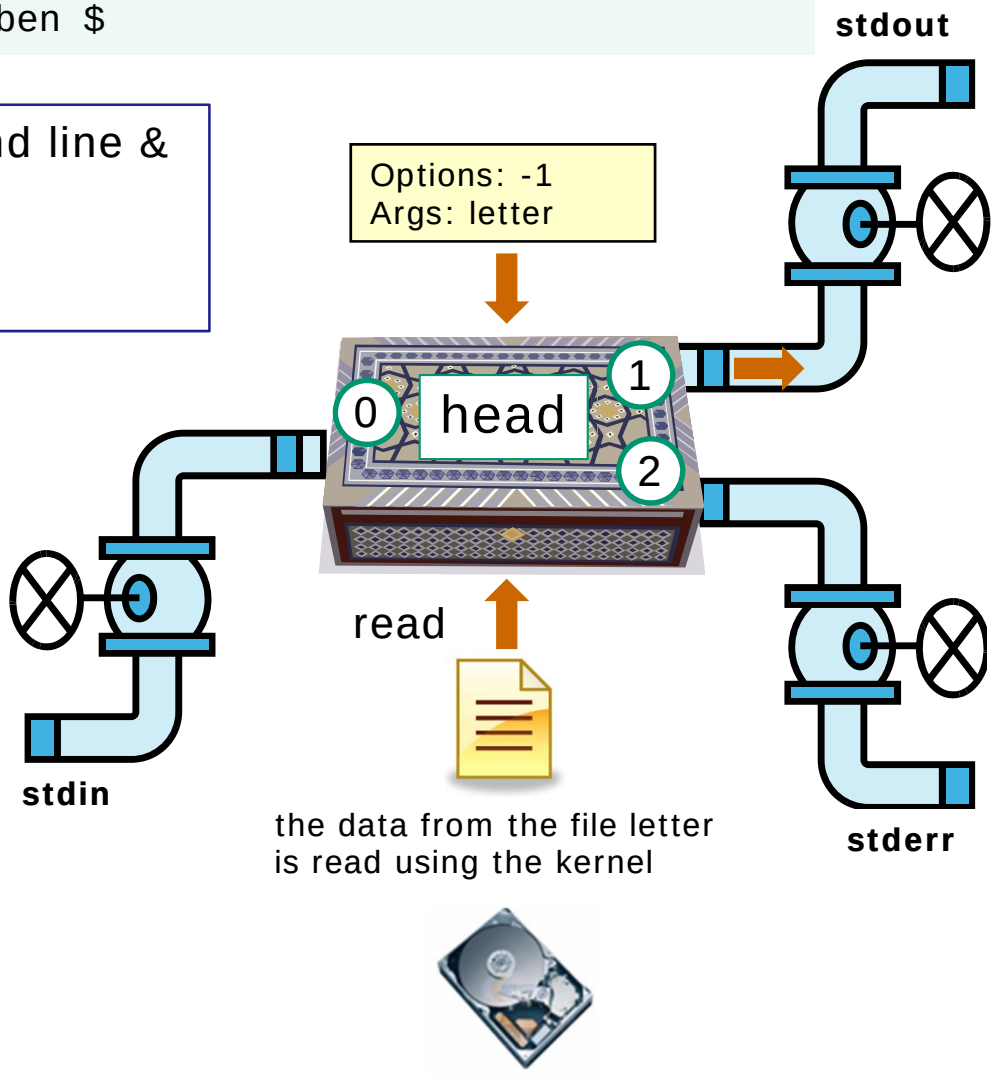


Example program to process: head command

```
/home/cis90/simmsben $ head -1 letter
Hello Mother!  Hello Father!
/home/cis90/simmsben $
```

Inputs: Command line & Operating System

Outputs: stdout

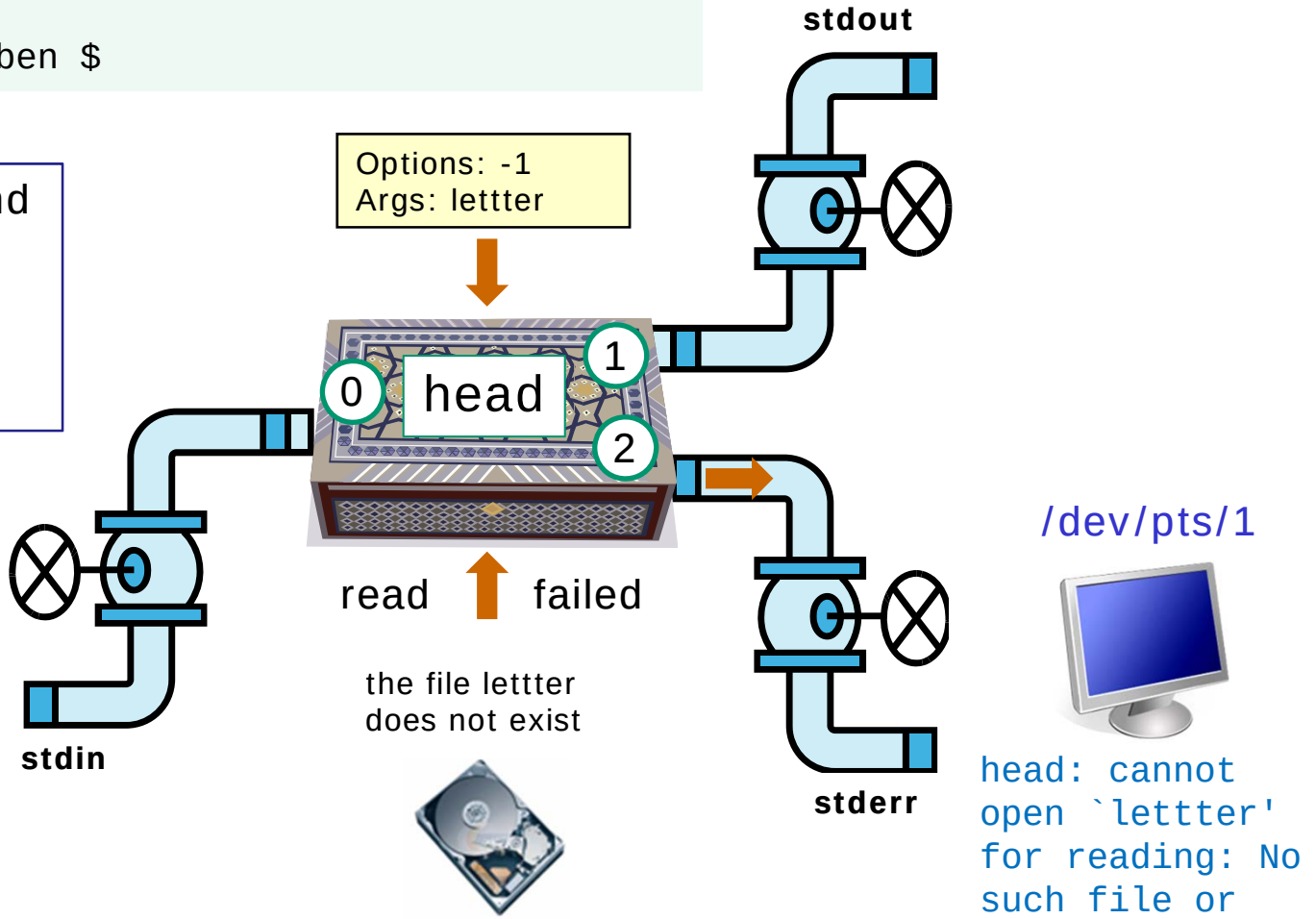


Example program to process: head command

```
/home/cis90/simmsben $ head -1 lettter
head: cannot open `lettter' for reading: No such
file or directory
/home/cis90/simmsben $
```

Inputs: Command line & Operating System

Outputs: stderr

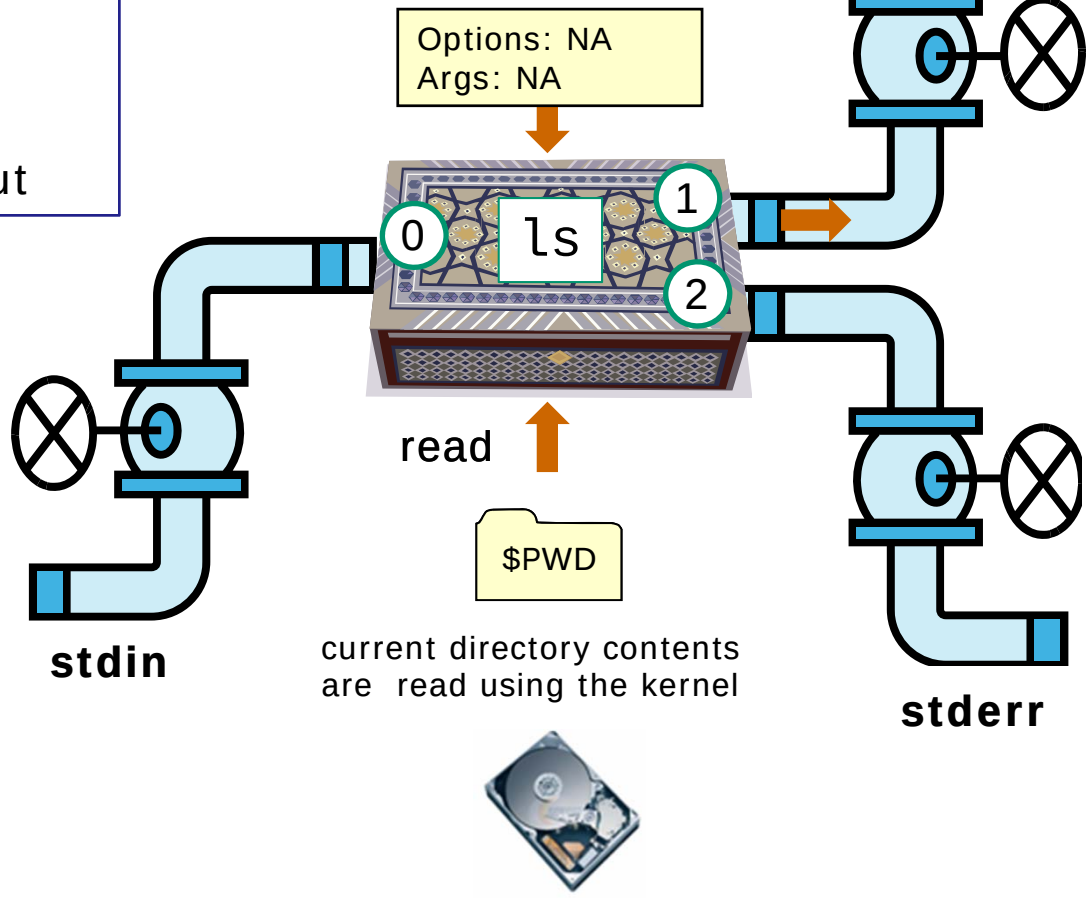


Example program to process: ls command

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

Inputs: Command line & Operating System

Outputs: stdout

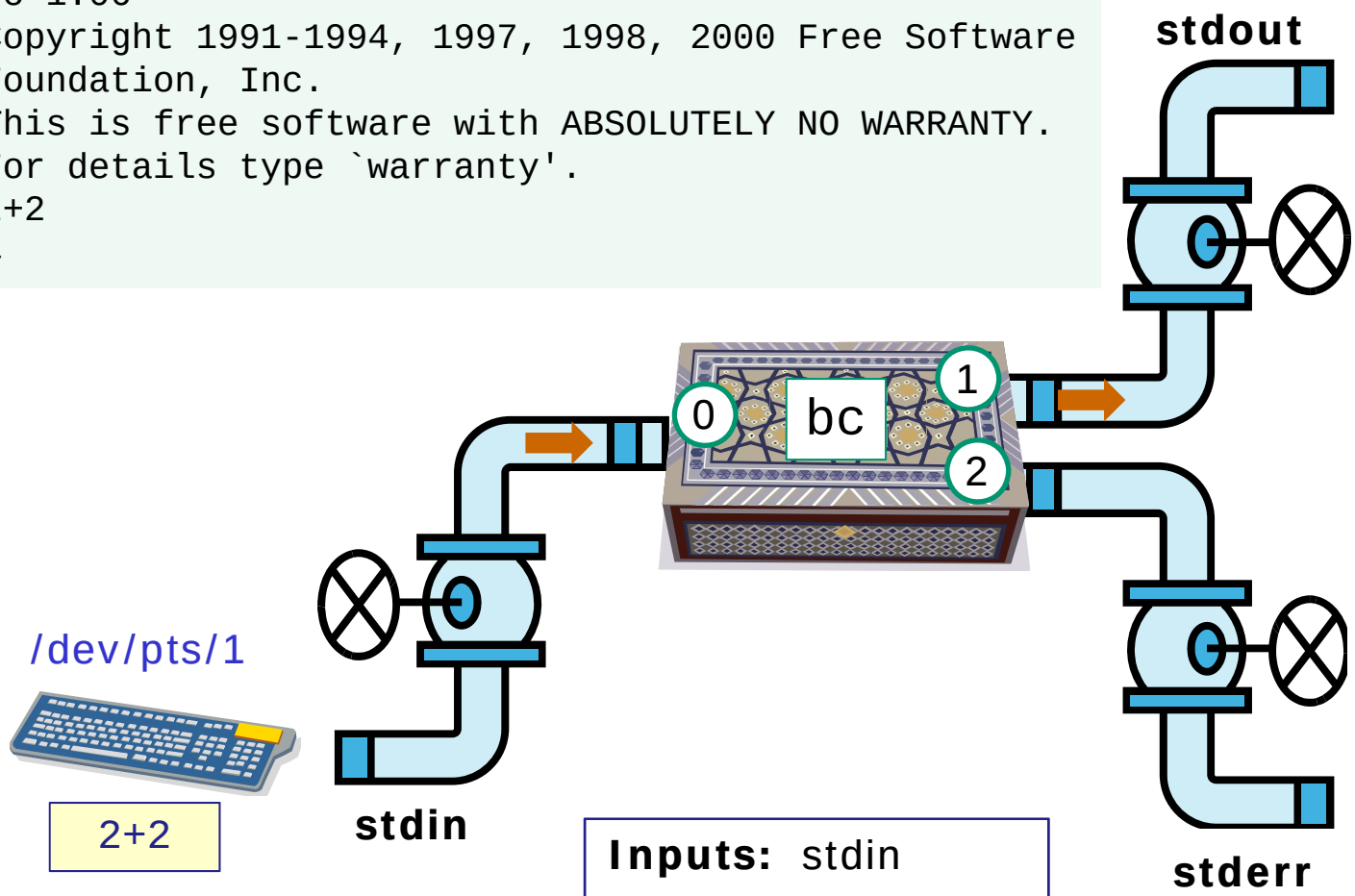


/dev/pts/1

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

Example program to process: bc command

```
/home/cis90/simmsben $ bc
bc 1.06
Copyright 1991-1994, 1997, 1998, 2000 Free Software
Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
2+2
4
```



Inputs: stdin

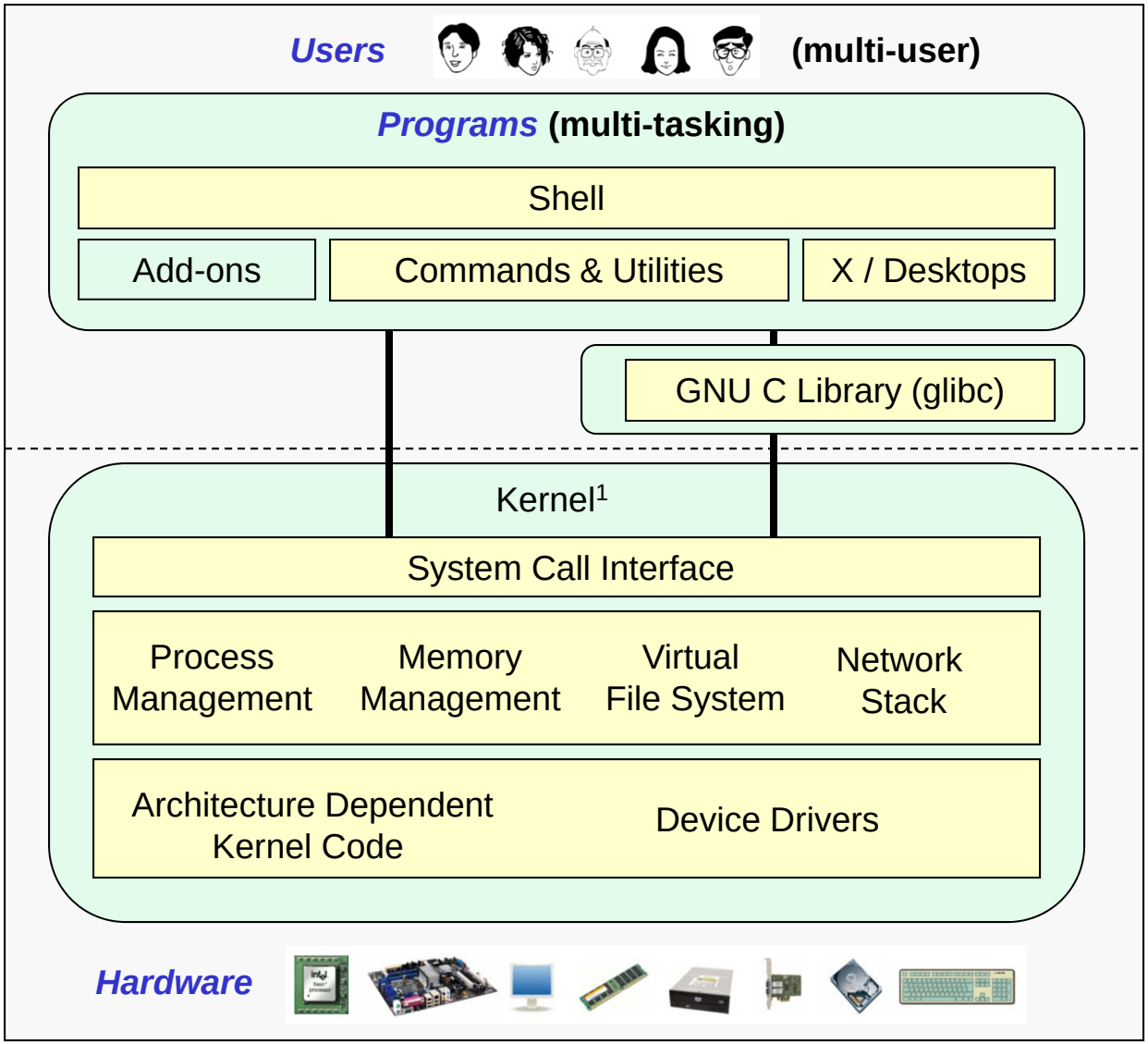
Outputs: stdout

Architecture (review)

GNU/Linux Operating System Architecture

User Space

Kernel Space



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

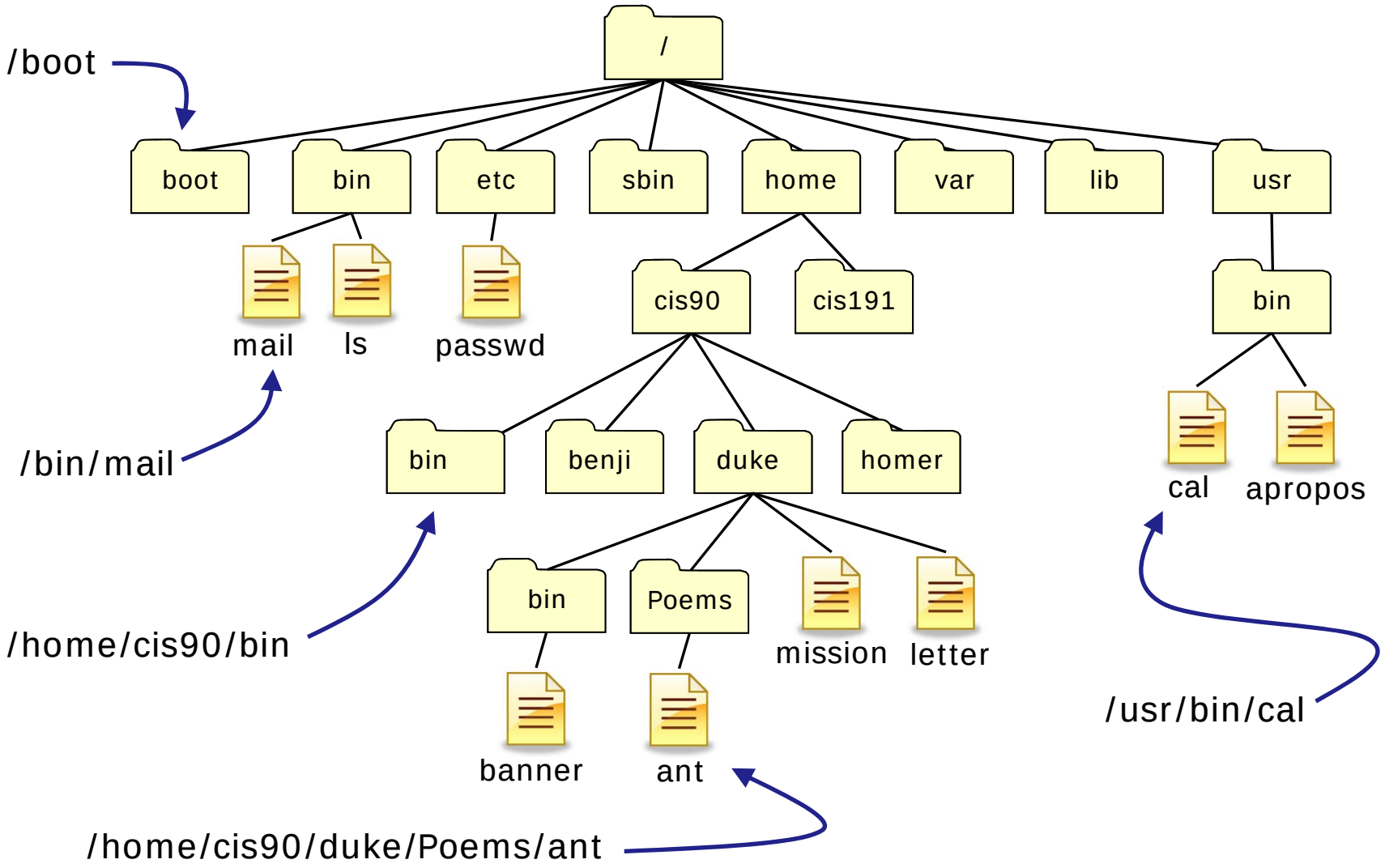
Linus Torvalds, as a student, initially conceived and assembled the Linux kernel in 1991. The kernel was later re-licensed under the GNU General Public License in 1992.

[1]See "Anatomy of the Linux kernel" by M. Tim Jones at http://www-128.ibm.com/developerworks/linux/library/l-linux-kernel/

File System (review)

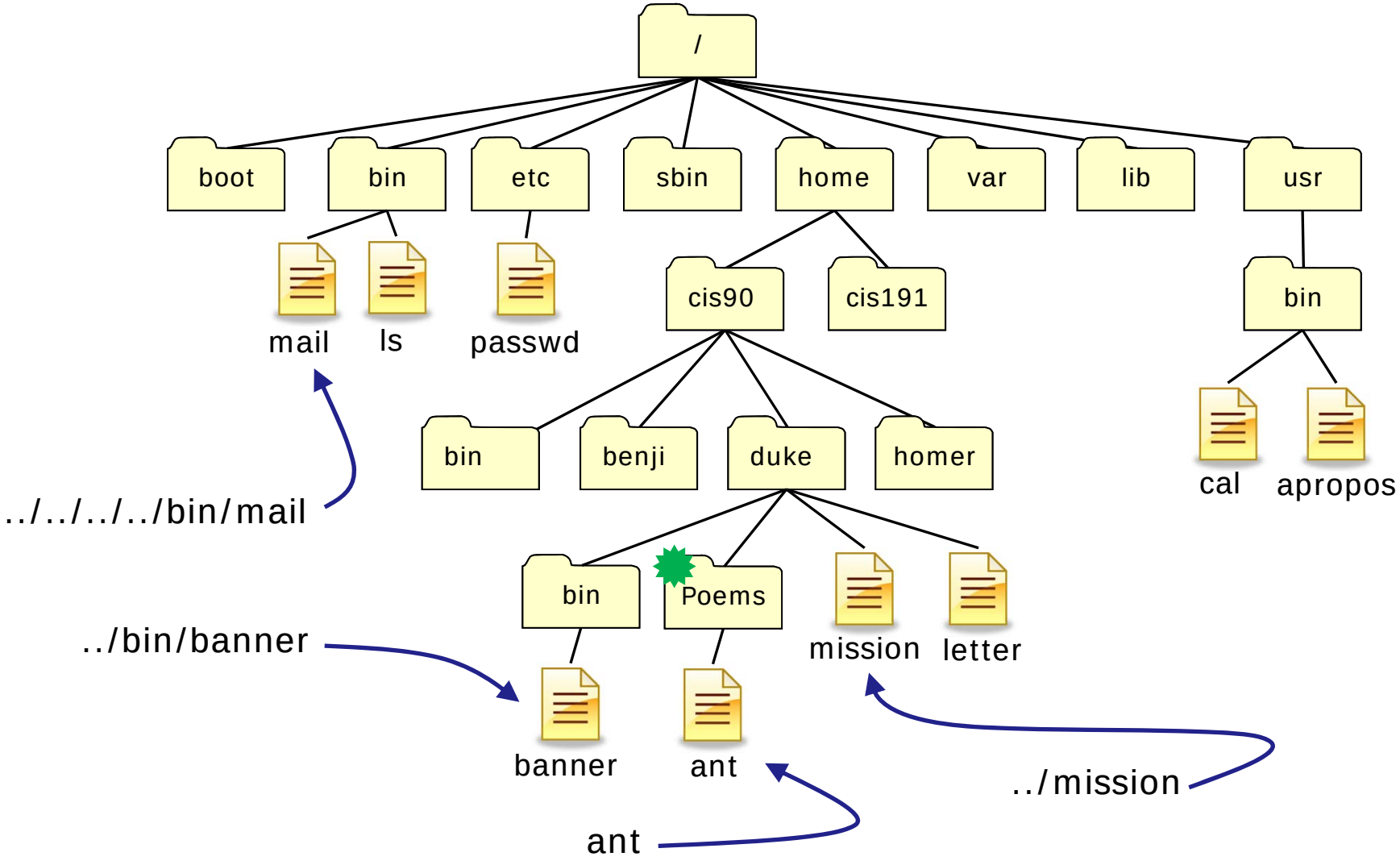
Absolute Pathnames

Fully specified names starting with /



Relative Pathnames

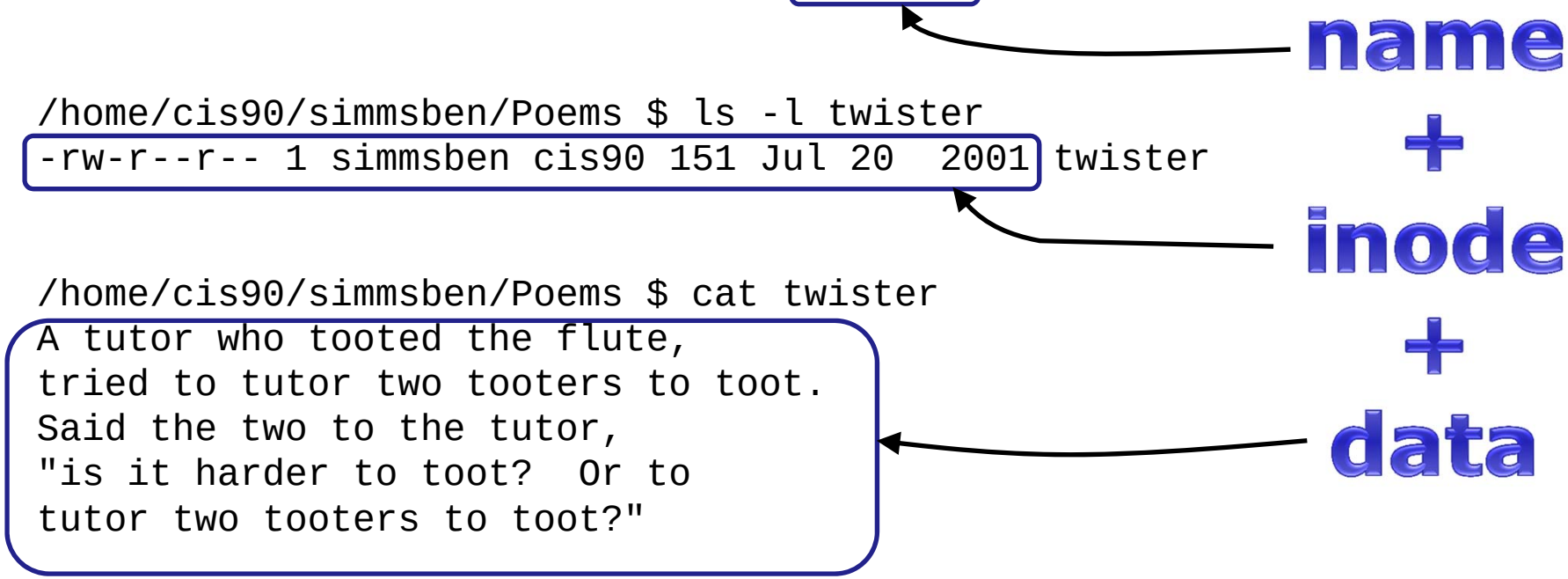
Names that start relative to the current working directory (✹)



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

UNIX Files
The three elements of a file

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

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

/home/cis90/simmsben/Poems $ cat 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?"
```



File Types and Commands

Long listing code (ls –l)	Type	How to make one
d	directory	mkdir
-	regular • Programs • Text • Data (binary)	touch
l	symbolic link	ln -s
c	character device files	mknod
b	block device files	mknod

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

Various Types of files

```
simmsben@opus:~
/home/cis90/simmsben $ls -la
total 320
drwx------ 9 simmsben cis90  4096 Aug  8 11:51 .
drwxr-x--- 9 rsimms   cis90  4096 Jun 30 14:57 ..
-rw------- 1 simmsben cis90 11409 Aug  7 19:20 .bash_history
-rw------- 1 simmsben cis90    24 Jul 20  2001 .bash_logout
-rw------- 1 simmsben cis90   354 Sep 17  2003 .bash_profile
-rw------- 1 simmsben cis90   146 Jan 18  2004 .bashrc
-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-rw-r-- 1 simmsben cis90  1044 Aug  8 11:52 deleteme
-rw-r--r-- 1 simmsben cis90   515 Jun 30 14:57 .emacs
-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------- 1 simmsben cis90    35 Aug  8 13:58 .lesshst
-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 4 simmsben cis90  4096 Jun 30 14:57 .mozilla
-rw-r--r-- 1 simmsben cis90    40 Jul 20  2001 .plan
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------- 1 simmsben cis90   661 Jul 24 13:59 .viminfo
-rw-r--r-- 1 simmsben cis90   352 Jul 20  2001 what_am_i
-rw------- 1 simmsben cis90   126 Aug  7 14:23 .Xauthority
-rw-r--r-- 1 simmsben cis90   658 Jun 30 14:57 .zshrc
/home/cis90/simmsben $
```

Hidden file or directory, any name starting with a .

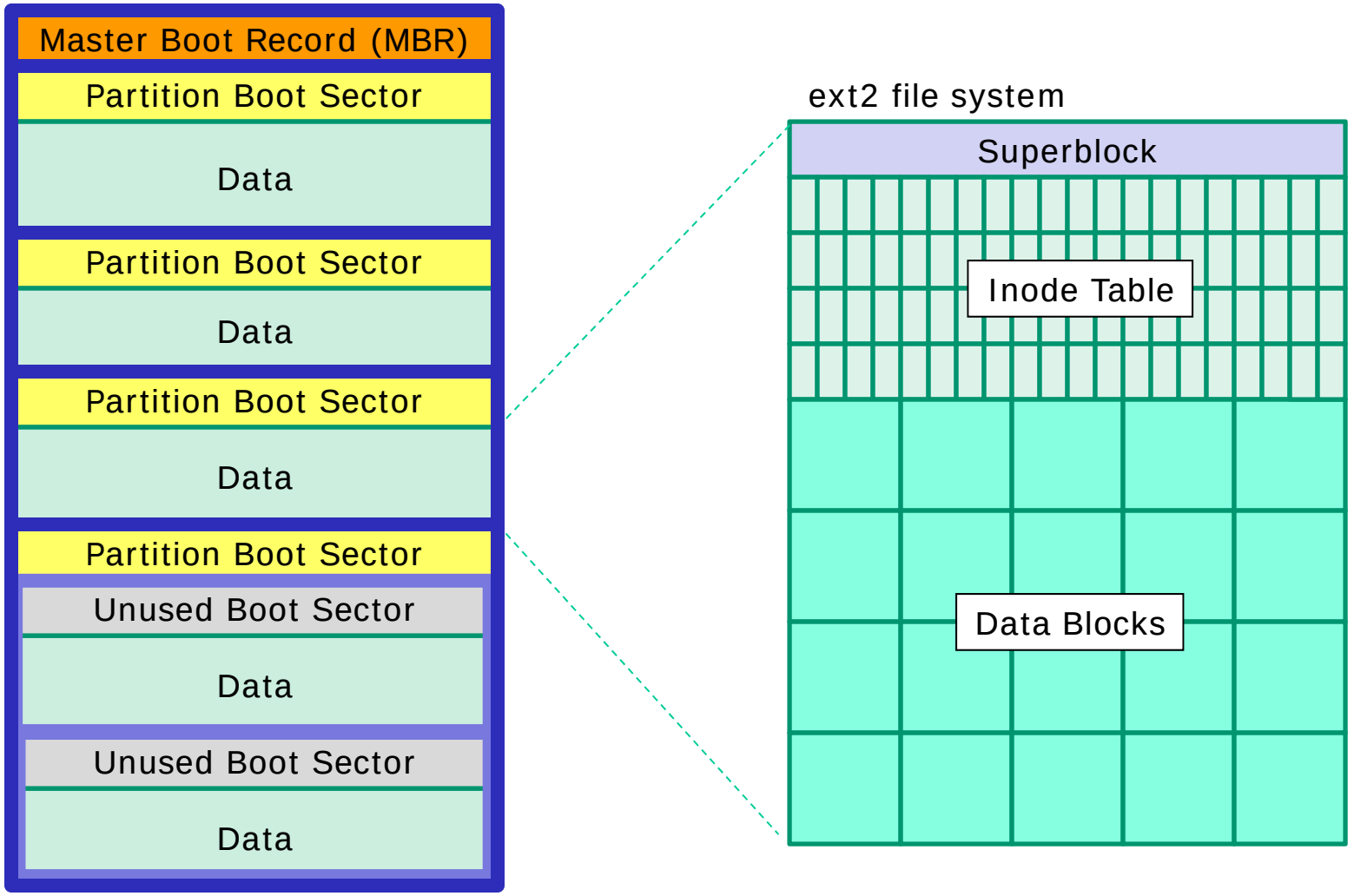
Directory (blue), d in column 1

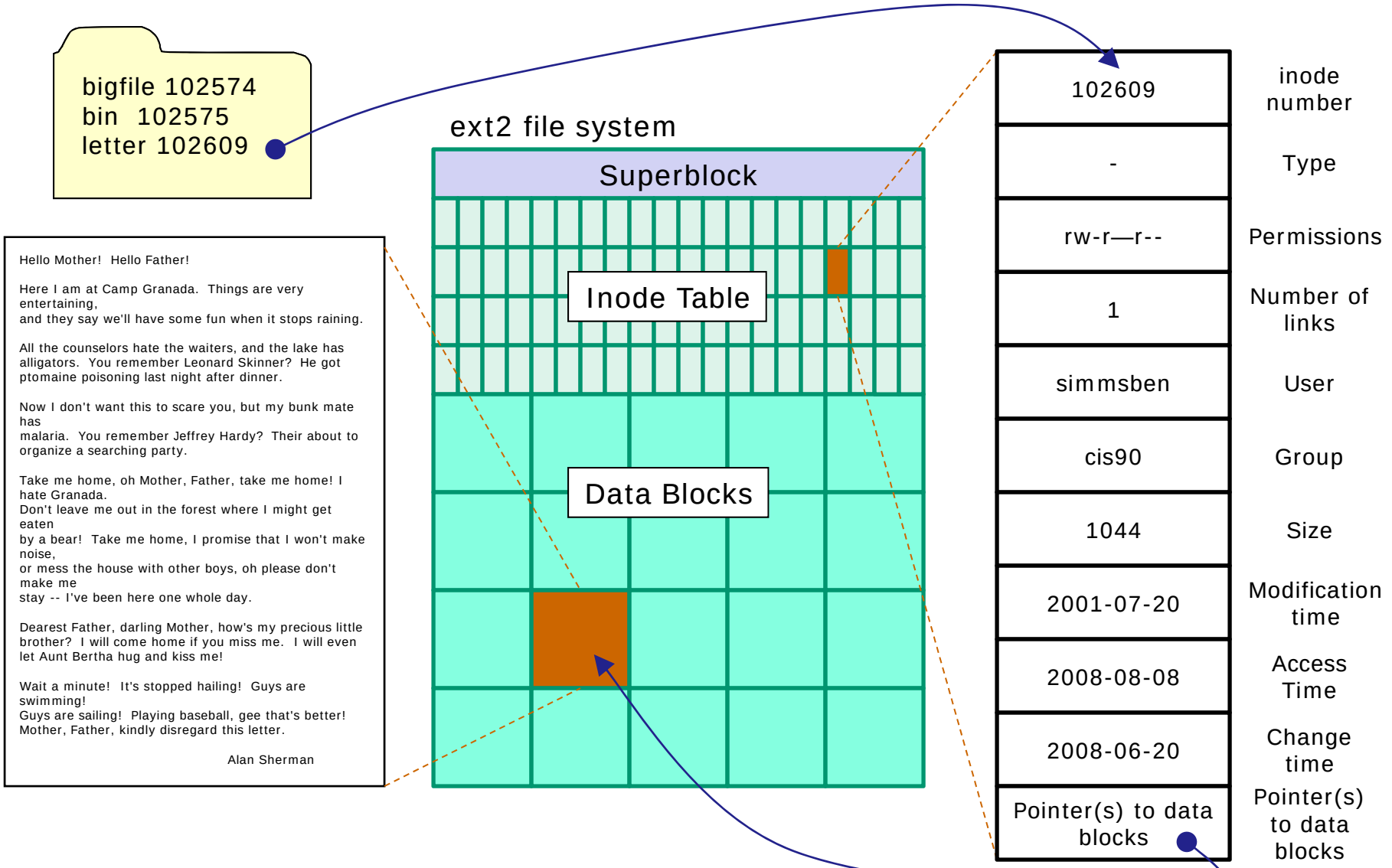
Executable file (green) with execute bits set

Regular file, - in column 1

File Systems

Linux

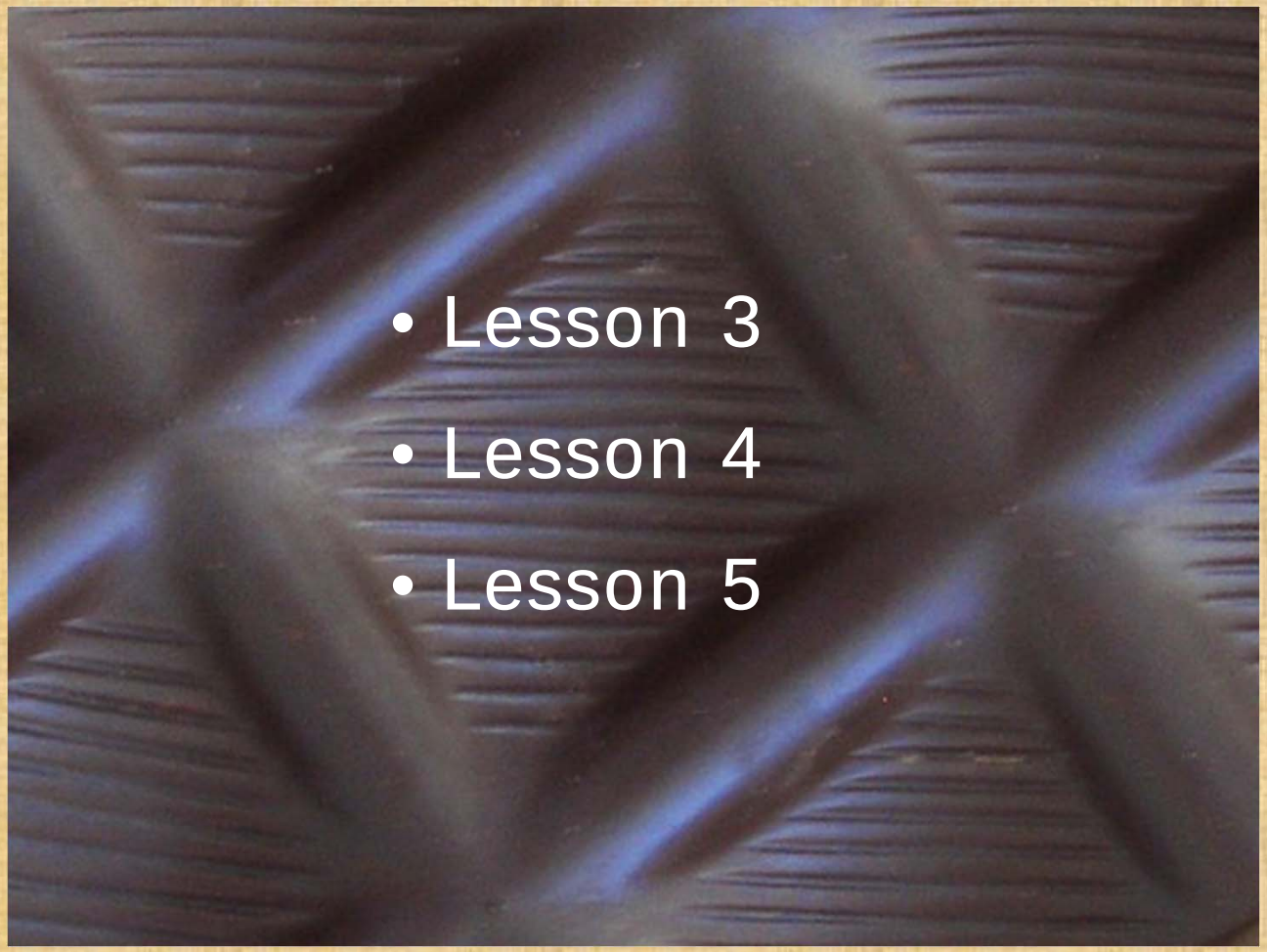




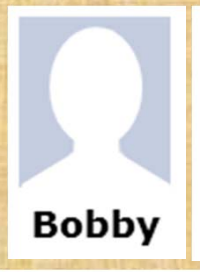

```
[simmsben@opus ~]$ls -il letter
102609 -rw-r--r-- 1 simmsben cis90 1044 Jul 20  2001 letter
```

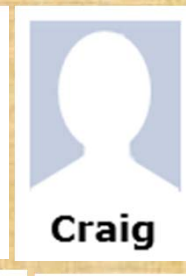
Class Exercise

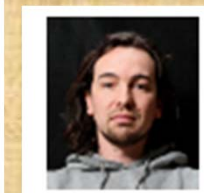
Flashcards

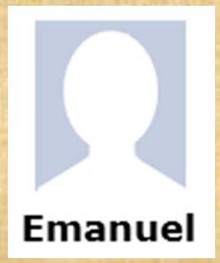


Team debian 11


Bobby


Craig


Daniel W


Emanuel


Gabriel


Jason


Josh


Marisol


Quinton


Tanner


Tajvia

Team Mandriva 11


Chris


Dan M


David


Eric


Geoffrey


Jeff


Jesse


Merrick


Tommy


Terence


Yu-Chen

Flashcards

Flash Cards

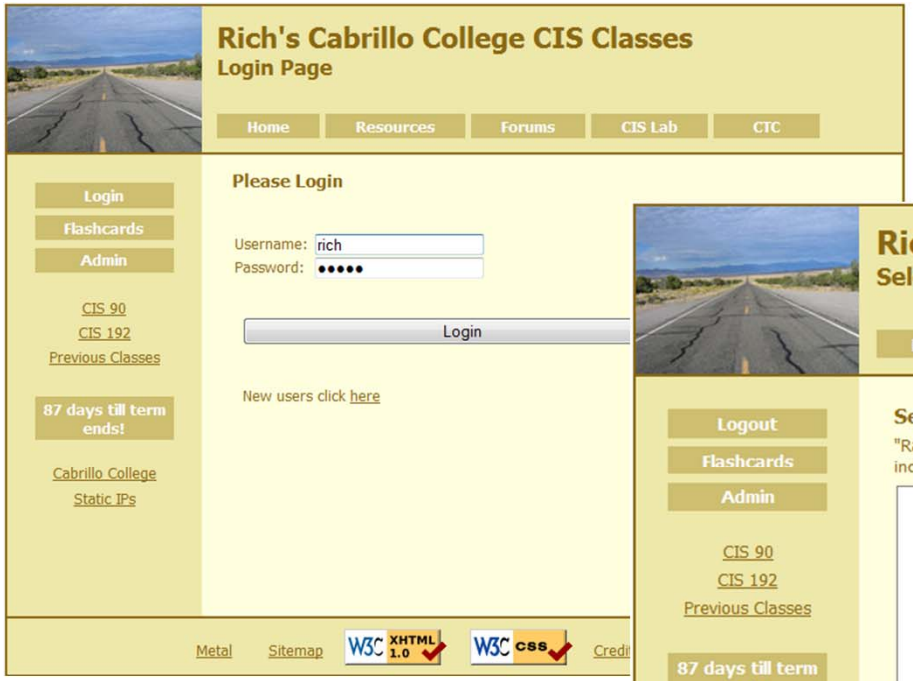
Click on Flashcards in left panel

Register if this is the first time using Flashcards

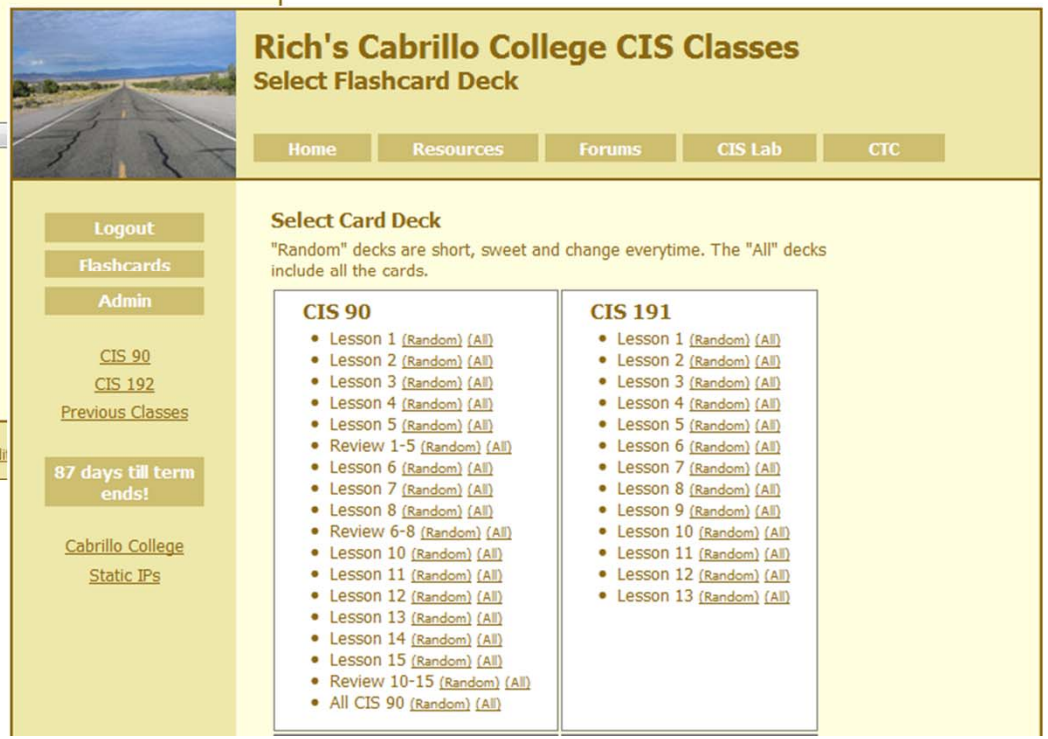
Register and choose a username and password of your choice

Logging in and using Flashcards

Login with your username and password



Select deck of cards



Class Exercise
Flashcards

- Browse to simms-teach.com
- Register with a username and password of your choice
- Verify you can login and use the flash cards.

Test Prep

Test next week

- 30 points, plus some extra credit
- 5 flashcard questions
  - Take directly from the flashcards on the web site
- 10 operational questions
  - You can verify your answers using Opus
- Open book, open notes, open computer
- To be taken during the last half of class
- Should take about 60-90 minutes, however if you need extra time, you can turn it in no later than midnight.
- PDF form format. Fill out the form, save it and email to instructor when finished.

First test – some tips on preparation

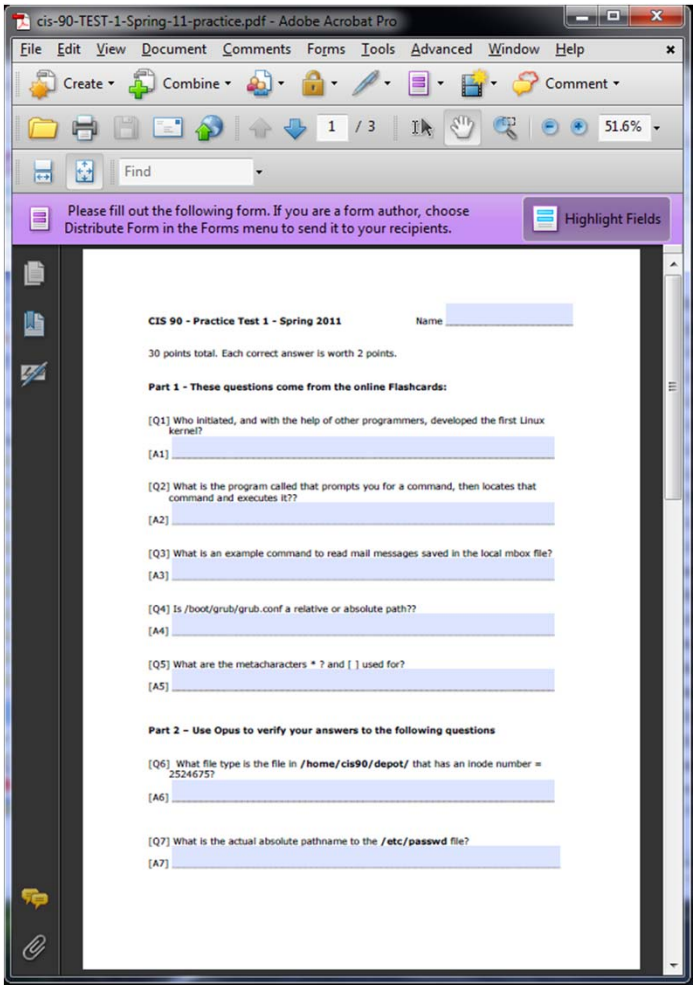
1. Take the practice test, collaborate by comparing answers with each other on the forum.
2. Review Lesson 1-5 PowerPoint slides and learn how to do searches.
3. Review and/or do labs 1-4 over.
4. Read the man pages or google the commands we have learned so far.
5. Use Lesson 1-5 flash cards.
6. Use the forum to ask and answer any questions.

First Test

1. Example flash card question:
   What is the program called that prompts you for a command, then locates that command and executes it?

2. Example operational question:
   From your home directory change to the Poems/Yeats/ directory. What one-liner (one ore more commands followed by Enter) would clear the screen and print the last line of all three Yeats poems without having to type the names of each individual poem file name?

Practice Test



A practise test is available on the web site Calendar page

You may need to download the latest version of Adobe Reader if you have problems filling it out.

Wrap up

New commands:

NA	NA

New Files and Directories:

NA	NA

Next Class

Assignment: Check Calendar Page on web site to see what is coming up.

No Quiz
No Lab due
Test !

Backup