Lesson Module Checklist

- Slides
- WB

- Flash cards
- Page numbers
- 1st minute quiz
- Web Calendar summary
- Web book pages
- Commands

- Opus - hide script tested -
- Practice test uploaded -
- Sun-Hwa - trouble made and rocks hidden

- 9V backup battery for microphone
- Backup slides, CCC info, handouts on flash drive

Introductions and Credits

Jim Griffin
- Created this Linux course
- Created Opus and the CIS VLab
- Jim’s site: http://cabrillo.edu/~jgriffin/

Rich Simms
- HP Alumnus
- Started teaching this course in 2008 when Jim went on sabbatical
- Rich’s site: http://simms-teach.com

And thanks to:
- John Govsky for many teaching best practices: e.g. the First Minute quizzes, the online forum, and the point grading system (http://teacherjohn.com/)

Cabrillo College est. 1959

CIS 90 - Lesson 9

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

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

Quiz

Please answer these questions **in the order** shown:

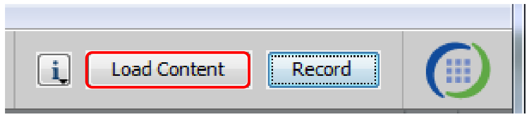
See electronic white board

email answers to: `risimms@cabrillo.edu`

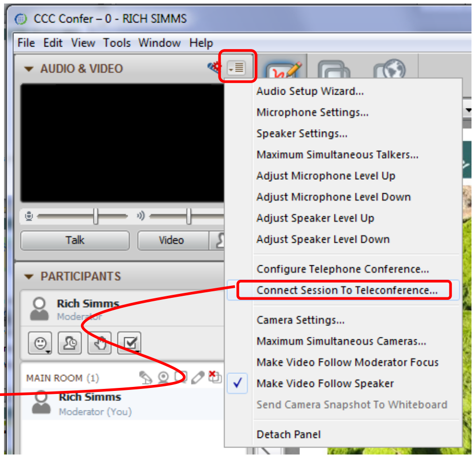
(answers must be emailed within the first few minutes of class for credit)

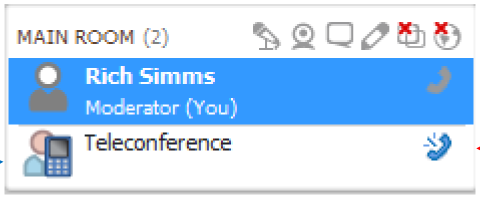
CIS 90 - Lesson 9

[ ] **Preload White Board with *cis*lesson??*-WB***



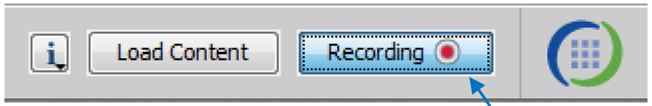
[ ] **Connect session to Teleconference**


CCC Confer – 0 - RICH SIMMS
File Edit View Tools Window Help
AUDIO & VIDEO
Audio Setup Wizard...
Microphone Settings...
Speaker Settings...
Maximum Simultaneous Talkers...
Adjust Microphone Level Up
Adjust Microphone Level Down
Adjust Speaker Level Up
Adjust Speaker Level Down
Configure Telephone Conference...
Connect Session To Teleconference...
Camera Settings...
Maximum Simultaneous Cameras...
Make Video Follow Moderator Focus
Make Video Follow Speaker
Send Camera Snapshot To Whiteboard
Detach Panel
Talk
Video
PARTICIPANTS
Rich Simms
MAIN ROOM (1)
Rich Simms
Moderator (You)

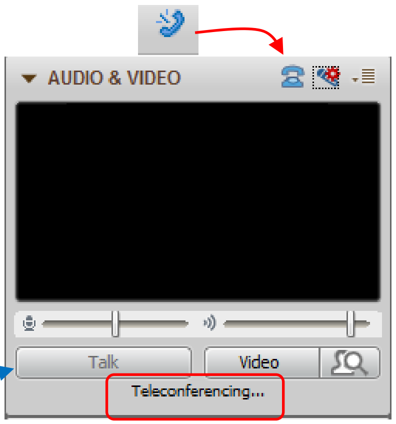


Session now connected to teleconference

[ ] **Is recording on?**



Red dot means recording



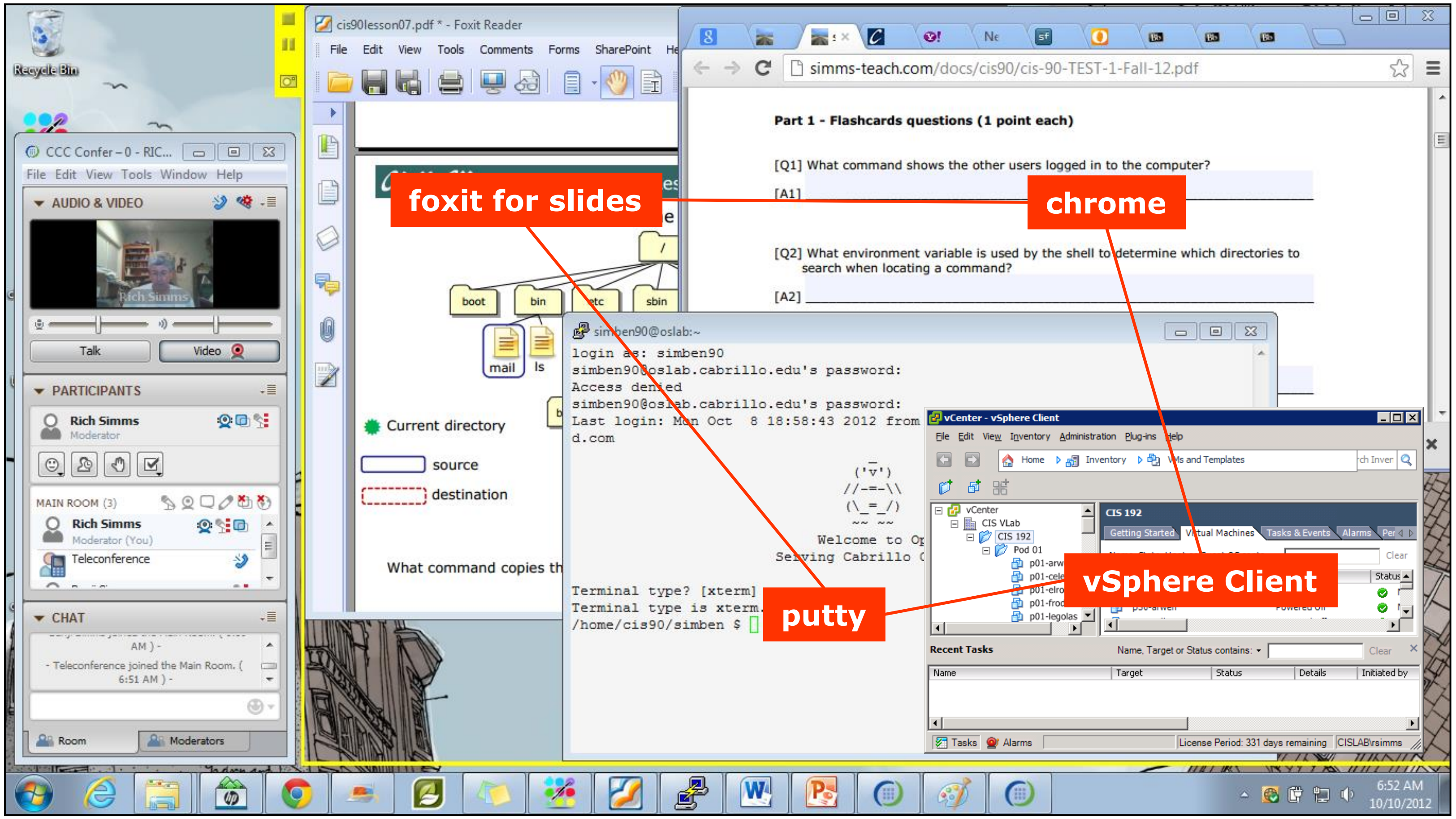
[ ] **Use teleconferencing, not mic**

Should be greyed out

CIS 90 - Lesson 9

[ ] **Video (webcam) optional**

[ ] **layout and share apps**



simms-teach.com/docs/cis90/cis-90-TEST-1-Fall-12.pdf

Part 1 - Flashcards questions (1 point each)

[Q1] What command shows the other users logged in to the computer?

[A1] ______________________________

[Q2] What environment variable is used by the shell to determine which directories to search when locating a command?

[A2] ______________________________

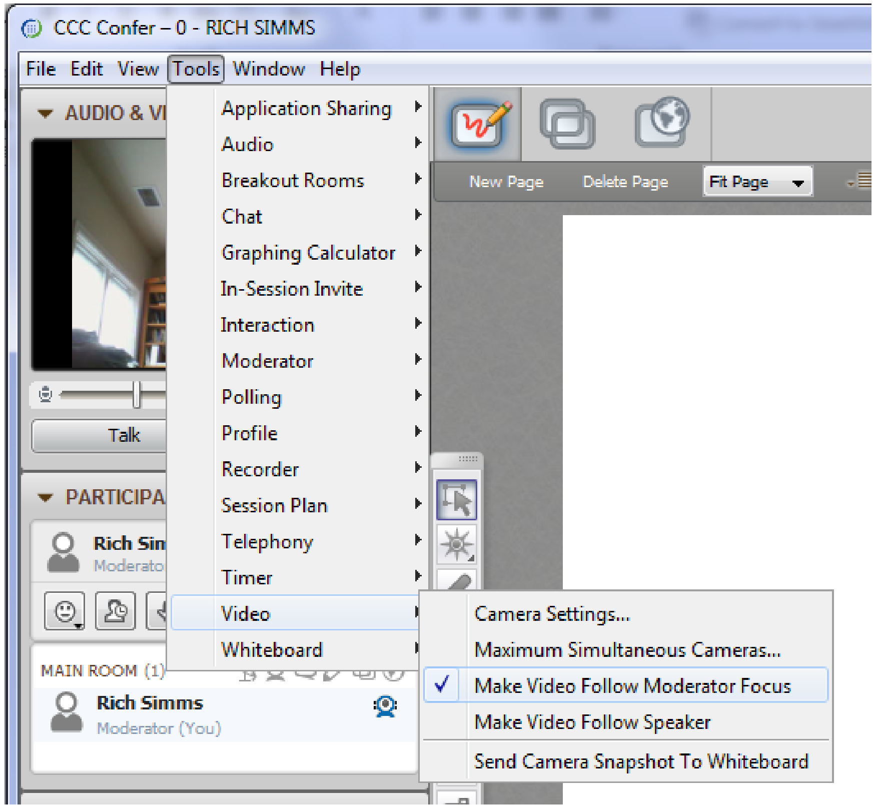
What command copies th

```
login as: simben90
simben90@oslab.cabrillo.edu's password:
Access denied
simben90@oslab.cabrillo.edu's password:
Last login: Mon Oct  8 18:58:43 2012 from
d.com

                                          ('v')
                                         //-=-\\
                                         (\_=_/)
                                          ~~ ~~
                                      Welcome to Op
                                  Serving Cabrillo C


Terminal type? [xterm]
Terminal type is xterm.
/home/cis90/simben $
```

[ ] Video (webcam) optional

[ ] Follow moderator

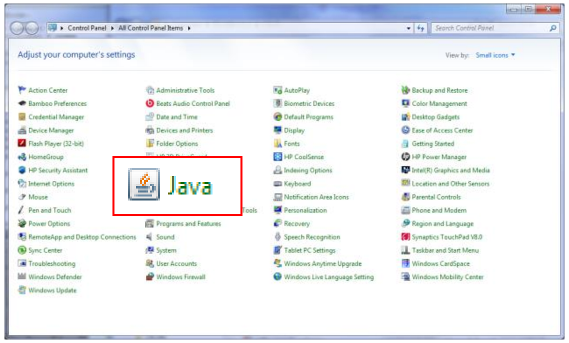
[ ] Double-click on postages stamps



Universal Fix for CCC Confer:

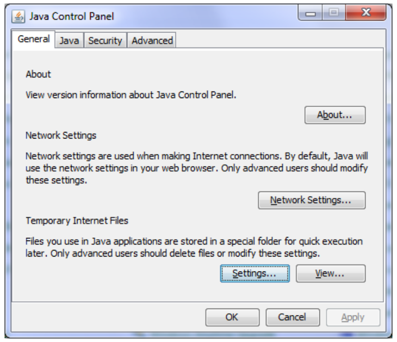
1) Shrink (500 MB) and delete Java cache
2) Uninstall and reinstall latest Java runtime

Control Panel (small icons)



Control Panel ▸ All Control Panel Items ▸
Adjust your computer's settings
View by: Small icons
Java

General Tab > Settings…



About

View version information about Java Control Panel.

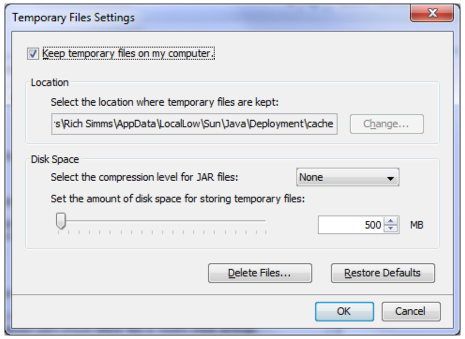
Network Settings

Network settings are used when making Internet connections. By default, Java will use the network settings in your web browser. Only advanced users should modify these settings.

Temporary Internet Files

Files you use in Java applications are stored in a special folder for quick execution later. Only advanced users should delete files or modify these settings.

500MB cache size



[✓] Keep temporary files on my computer.

Location

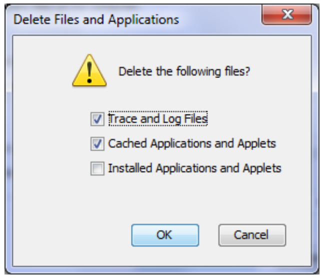
Select the location where temporary files are kept: s\Rich Simms\AppData\LocalLow\Sun\Java\Deployment\cache

Disk Space

Select the compression level for JAR files: None

Set the amount of disk space for storing temporary files: 500 MB

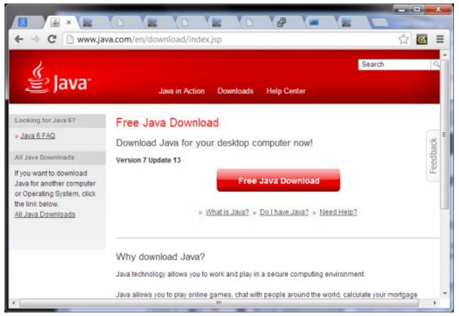
Delete these



Delete the following files?

[✓] Trace and Log Files
[✓] Cached Applications and Applets
[ ] Installed Applications and Applets

Google Java download



www.java.com/en/download/index.jsp

Looking for Java 6?
» Java 6 FAQ

All Java Downloads

If you want to download Java for another computer or Operating System, click the link below.
All Java Downloads

Free Java Download

Download Java for your desktop computer now!

Version 7 Update 13

» What is Java? » Do I have Java? » Need Help?

Why download Java?

Java technology allows you to work and play in a secure computing environment.

Java allows you to play online games, chat with people around the world, calculate your mortgage

Review

Objectives	Agenda
• Get ready for the next test • Practice skills • Introduction to processes	• Quiz • Questions • More on I/O • Shell six steps • Subtle I/O • 2>&1 • C program I/O • More on umask • Pipeline practice • Housekeeping • Wireless Penetration (Ryan) • Test Review • Wrap up • Practice test workshop

Questions

Questions

Lesson material?

Labs?

How this course works?

- Graded work in home directories
- Answers in /home/cis90/answers

Chinese Proverb	他問一個問題，五分鐘是個傻子，他不問一個問題仍然是一個傻瓜永遠。 *He who asks a question is a fool for five minutes; he who does not ask a question remains a fool forever.*

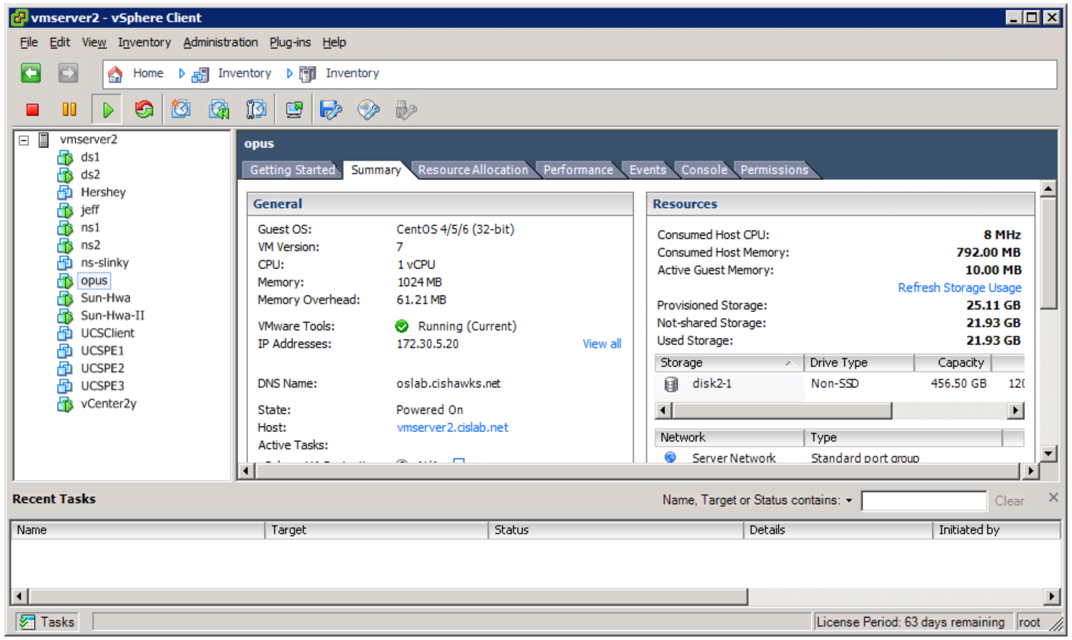
Opus

Opus Centos 6.2 Linux Server

vmserver2

Opus is a VM running on the vmserver2 server in the CIS Lab

VMs on vmserver2



vmserver2 - vSphere Client
File Edit View Inventory Administration Plug-ins Help
Home Inventory Inventory
vmserver2
ds1
ds2
Hershey
jeff
ns1
ns2
ns-slinky
opus
Sun-Hwa
Sun-Hwa-II
UCSClient
UCSPE1
UCSPE2
UCSPE3
vCenter2y
opus
Getting Started Summary Resource Allocation Performance Events Console Permissions

General

Guest OS:	CentOS 4/5/6 (32-bit)
VM Version:	7
CPU:	1 vCPU
Memory:	1024 MB
Memory Overhead:	61.21 MB
VMware Tools:	Running (Current)
IP Addresses:	172.30.5.20 View all
DNS Name:	oslab.cishawks.net
State:	Powered On
Host:	vmserver2.cislab.net
Active Tasks:	

Resources

Consumed Host CPU:	8 MHz
Consumed Host Memory:	792.00 MB
Active Guest Memory:	10.00 MB
	Refresh Storage Usage
Provisioned Storage:	25.11 GB
Not-shared Storage:	21.93 GB
Used Storage:	21.93 GB

Storage	Drive Type	Capacity
disk2-1	Non-SSD	456.50 GB

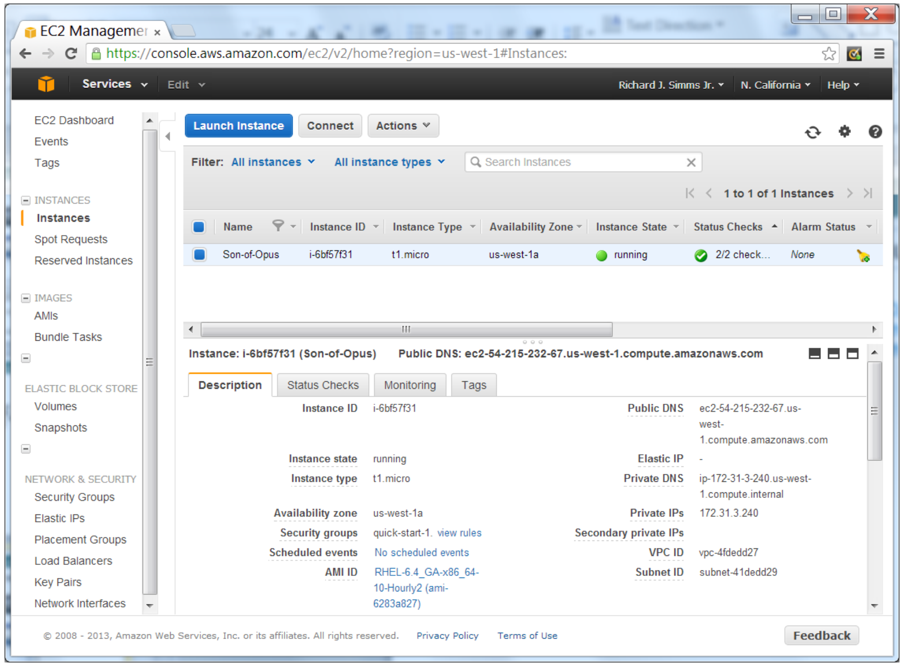
Network	Type
Server Network	Standard port group

Recent Tasks
Name, Target or Status contains:
Clear
Name Target Status Details Initiated by
Tasks
License Period: 63 days remaining
root

<u>SSH access to Opus</u>
hostname: oslab.cishawks.net (port 2220)

Son-of-Opus

Son-of-Opus Red Hat 6.4 Linux Server



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
Son-of-Opus	i-6bf57f31	t1.micro	us-west-1a	● running	✓ 2/2 check...	None

Instance: i-6bf57f31 (Son-of-Opus) Public DNS: ec2-54-215-232-67.us-west-1.compute.amazonaws.com

Instance ID	i-6bf57f31	Public DNS	ec2-54-215-232-67.us-west-1.compute.amazonaws.com
Instance state	running	Elastic IP	-
Instance type	t1.micro	Private DNS	ip-172-31-3-240.us-west-1.compute.internal
Availability zone	us-west-1a	Private IPs	172.31.3.240
Security groups	quick-start-1. view rules	Secondary private IPs	
Scheduled events	No scheduled events	VPC ID	vpc-4fdedd27
AMI ID	RHEL-6.4_GA-x86_64-10-Hourly2 (ami-6283a827)	Subnet ID	subnet-41dedd29

© 2008 - 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Son-of-Opus is a VM running on Amazon Web Services

<u>SSH access to Son-of-Opus</u>
hostname: son-of-opus.simms-teach.com (port 2220)

Baby-Opus

Baby-Opus Debian 7 Linux Server

Raspberry Pi

Baby-Opus is a VM running on my Raspberry Pi

<u>SSH access to Baby-Opus</u>
hostname: <*ip-address*> (port 22)

More on I/O

(input/output)

Input and Output

File Redirection

The 3 standard UNIX file descriptors:

Name	Integer Value
stdin (**st**an**d**ard **in**)	0
stdout (**st**an**d**ard **out**)	1
stderr (**st**an**d**ard **err**or)	2

*Every process is provided with three file descriptors: **stdin**, **stdout** and **stderr***

Input and Output

File Redirection

*The input and output of a program can be **redirected** to and from other files as follows:*

~~0~~< *filename*

Redirects **stdin**, input will now come from *filename* rather than the keyboard.

~~1~~> *filename*

Redirects **stdout**, output will now go to *filename* instead of the terminal.

2> *filename*

Redirects **stderr**, error messages will now go to *filename* instead of the terminal.

>> *filename*

Redirects **stdout**, output will now be appended to *filename*.

The redirection is specified on the command line

Shell prints this to prompt user to enter a command

Shell parses this command line

Prompt | Command | Options | Arguments | **Redirection**

***Redirection** connects **stdin**, **stdout** and **stderr** to non-default devices*

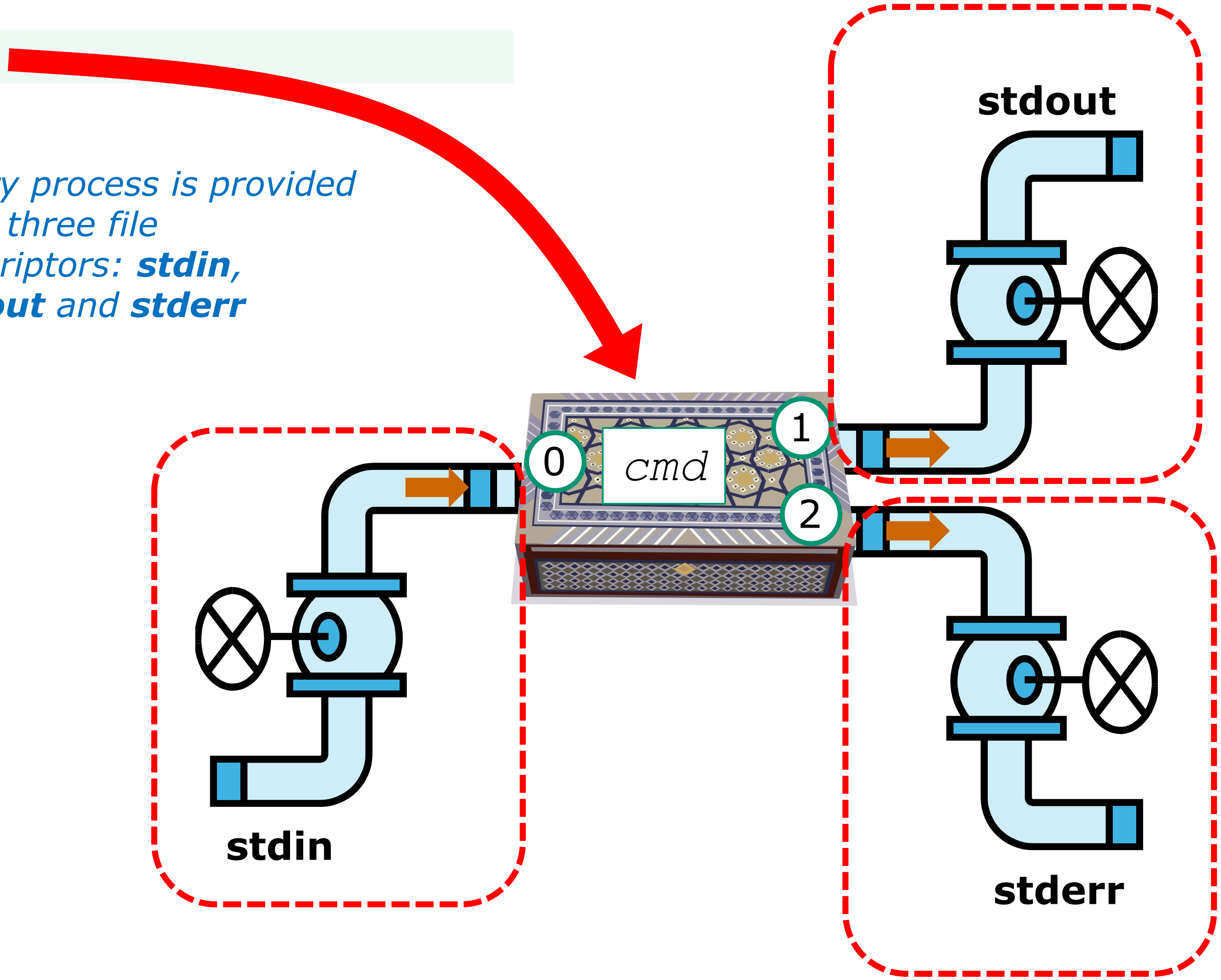
Examples

```
/home/cis90/simben $ cat
/home/cis90/simben $ cat -A letter
/home/cis90/simben $ cat    < letter
/home/cis90/simben $ cat -b < letter > out
/home/cis90/simben $ cat    bogus 2> /dev/null
/home/cis90/simben $ cat -e < bogus 2> /dev/null
/home/cis90/simben $ cat -e < letter > out 2> /dev/null
```

A program loaded into memory becomes a **process**

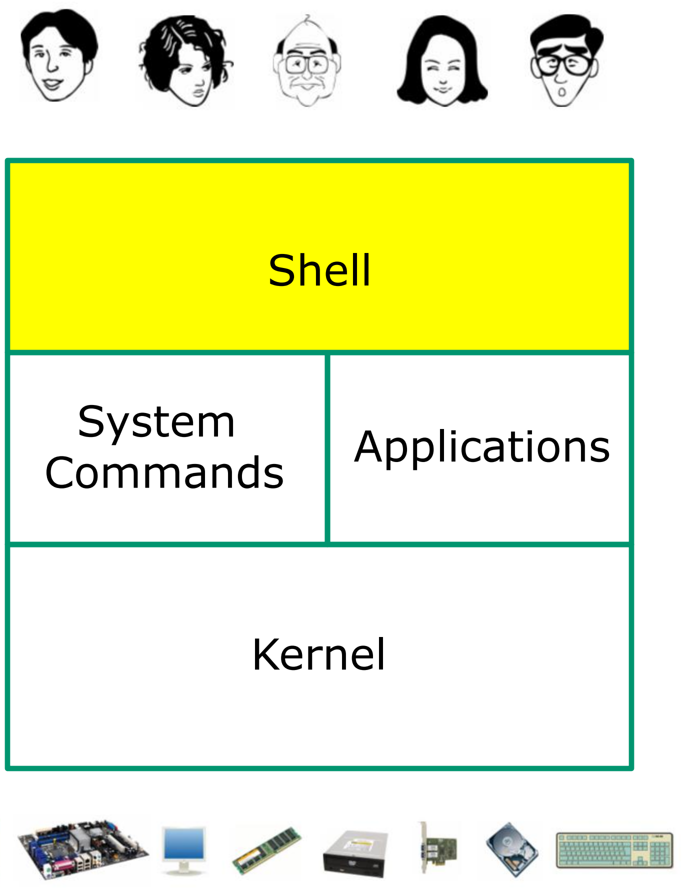
```
$ cmd
```

Every process is provided with three file descriptors: ***stdin****,* ***stdout*** *and* ***stderr***



All Together Now Example

Life of the Shell



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

Example

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

The shell begins by echoing a **prompt** string to your terminal device:

- Your specific terminal device can be identified by using the **tty** command.
- The format of the prompt is defined by the contents of the PS1 variable.

$$\underbrace{\texttt{/home/cis90/simben \$}}$$

In this case the PS1 variable is set to '$PWD $ ' which results in a prompt that shows the current location in the file tree followed by a blank, a $, and another blank.

Example

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

Following the prompt, the user then enters a command followed by the Enter key:

- The Enter key generates a <newline> which is a shell metacharacter. All metacharacters have special meanings to the shell.
- The <newline> characters instructs the shell that the command line is ready to be processed.

```
/home/cis90/simben $ sort -r names > dogsinorder
```

The user types in a command line followed by the Enter key

Example

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

The shell **parses** the command line entered by the user:

- The command line is carefully scanned to identify the command, options, arguments and any redirection information.
- Variables and filename expansion characters (wildcards) get processed.

```
/home/cis90/simben $ sort -r names > dogsinorder
```

Parsing results: `sort` `-r` `names` `> dogsinorder`

*The command is: **sort***
*There is one option: **-r***
*There is one argument: **names***
*Redirection is: redirect **stdout** to a file named **dogsinorder***

Example

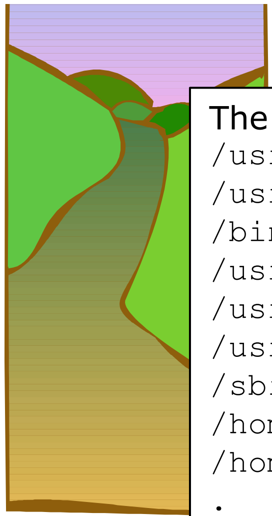
1) Prompt
2) Parse
3) Search
4) Execute
5) Nap
6) Repeat

The shell now **searches** for the command on the path:

- The path, which is an ordered list of directories, is defined by the contents of the PATH variable. Use **echo $PATH** to view.
- The shell will search in order each directory on the path to locate the command.
- If a command, such as xxxx, is not found, the shell will print:

  -bash: xxxx: command not found

- FYI, you can search for commands on the path too, like the shell does, by using the **type** command.



The ***Path*** (**echo $PATH** to show)

```
/usr/lib/qt-3.3/bin:
/usr/local/bin:
/bin:
/usr/bin:
/usr/local/sbin:
/usr/sbin:
/sbin:
/home/cis90/simben/../bin:
/home/cis90/simben/bin:
.
```

`sort` *The shell locates the sort command in the /bin directory which is the third directory of a CIS 90 student's path.*

Example

```
$ sort -r names > dogsinorder
```

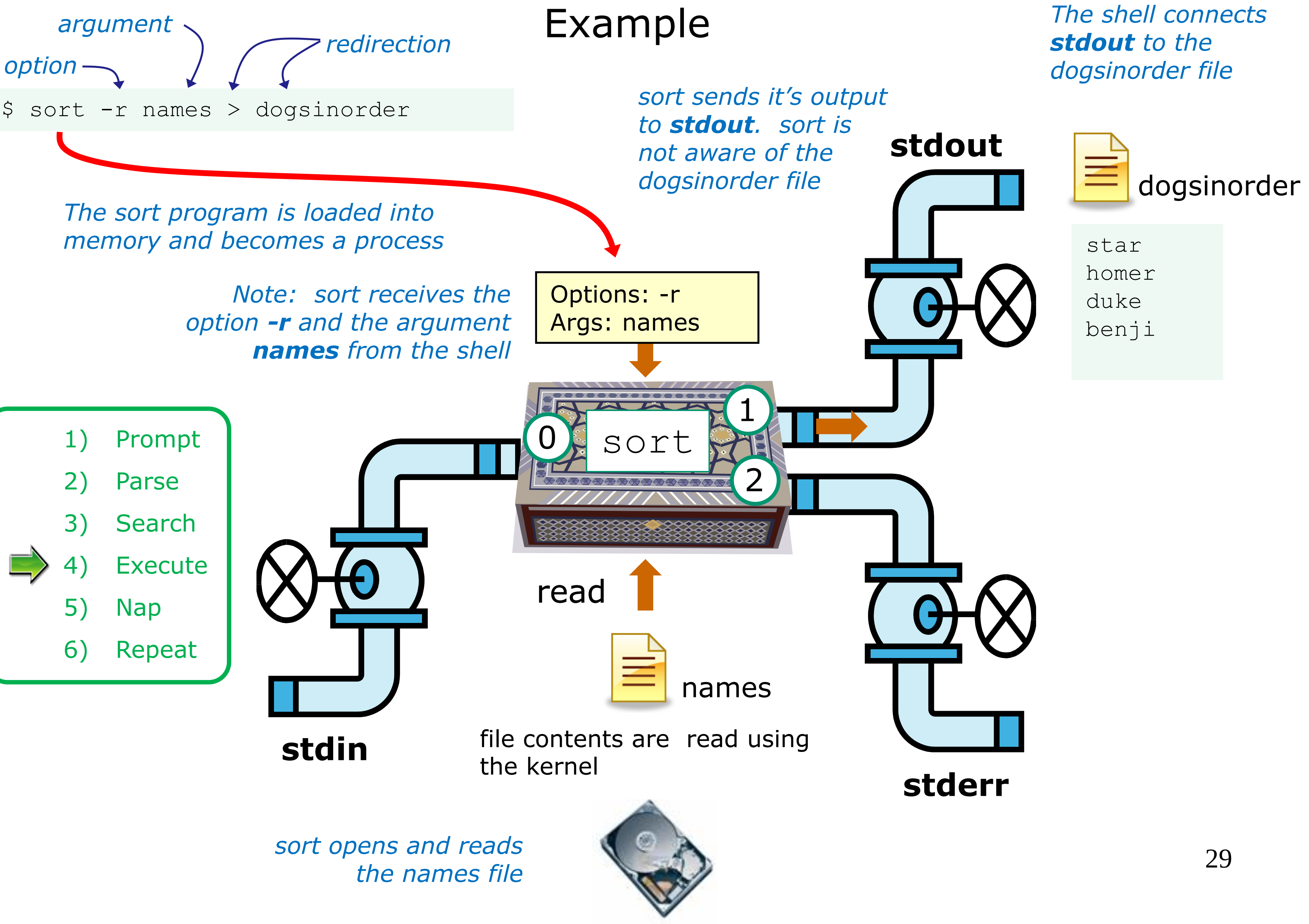
The sort program is loaded into memory and becomes a process

*Note: sort receives the option **-r** and the argument **names** from the shell*

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

*sort sends it's output to **stdout**. sort is not aware of the dogsinorder file*

*The shell connects **stdout** to the dogsinorder file*



```
star
homer
duke
benji
```

file contents are read using the kernel

sort opens and reads the names file

Example

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

While the sort process executes, the shell sleeps

Example

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

When the sort process finishes the shell wakes up and starts all over again to process the next command from the user!

Subtle Differences

What is the difference between:

head -n4 letter

and

head -n4 < letter

```
/home/cis90/simben $ head -n4 letter
Hello Mother!  Hello Father!

Here I am at Camp Granada.  Things are very entertaining,
and they say we'll have some fun when it stops raining.
```

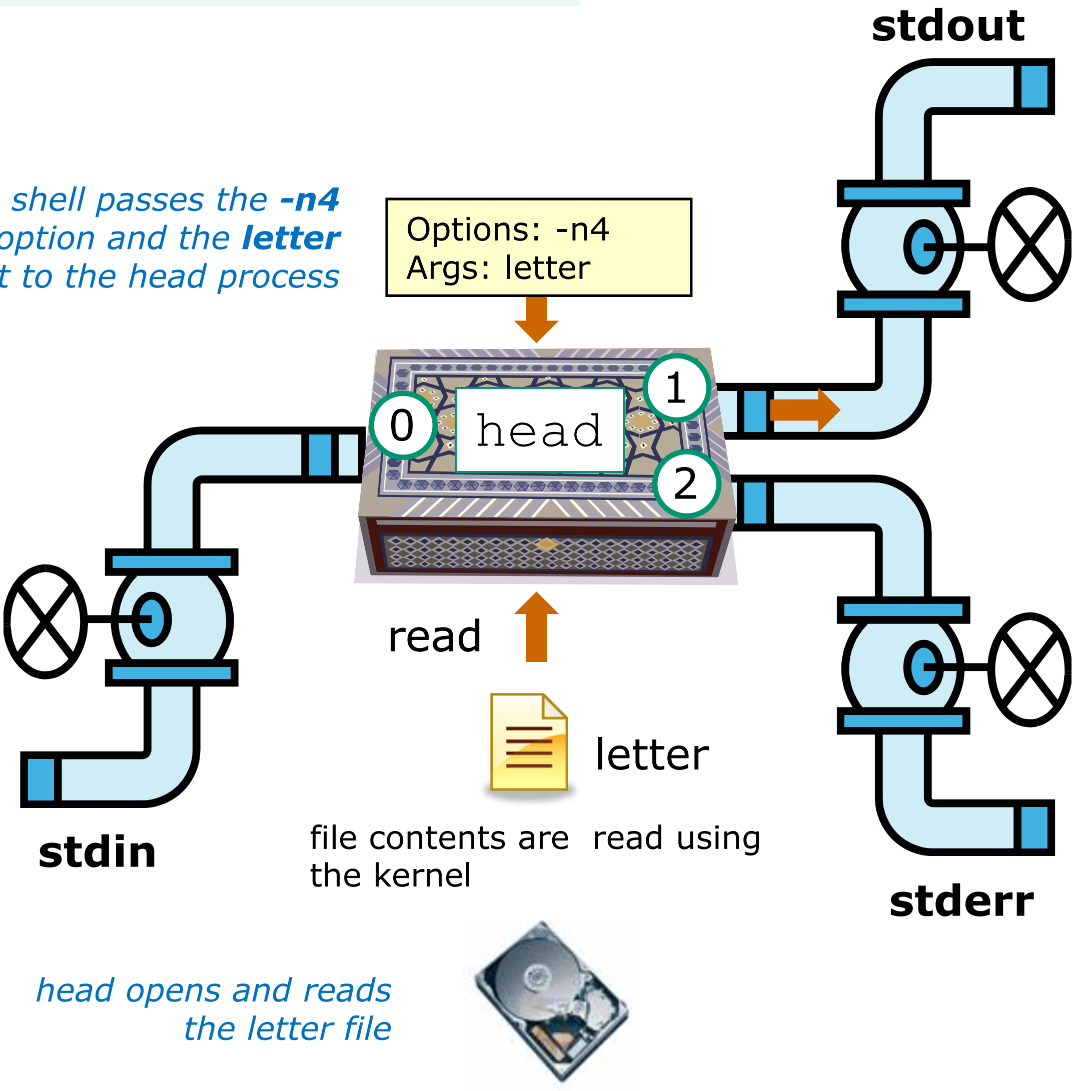
```
/home/cis90/simben $  head -n4 < letter
Hello Mother!  Hello Father!

Here I am at Camp Granada.  Things are very entertaining,
and they say we'll have some fun when it stops raining.
```

head -n4 letter

option *argument*

```
$ head -n4 letter
```

*The shell passes the **-n4** option and the **letter** argument to the head process*

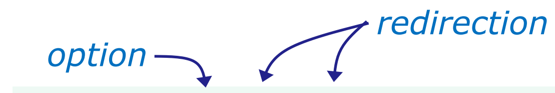


```
Hello Mother!  Hello Father!

Here I am at Camp Granada.  Things are very entertaining,
and they say we'll have some fun when it stops raining.
```

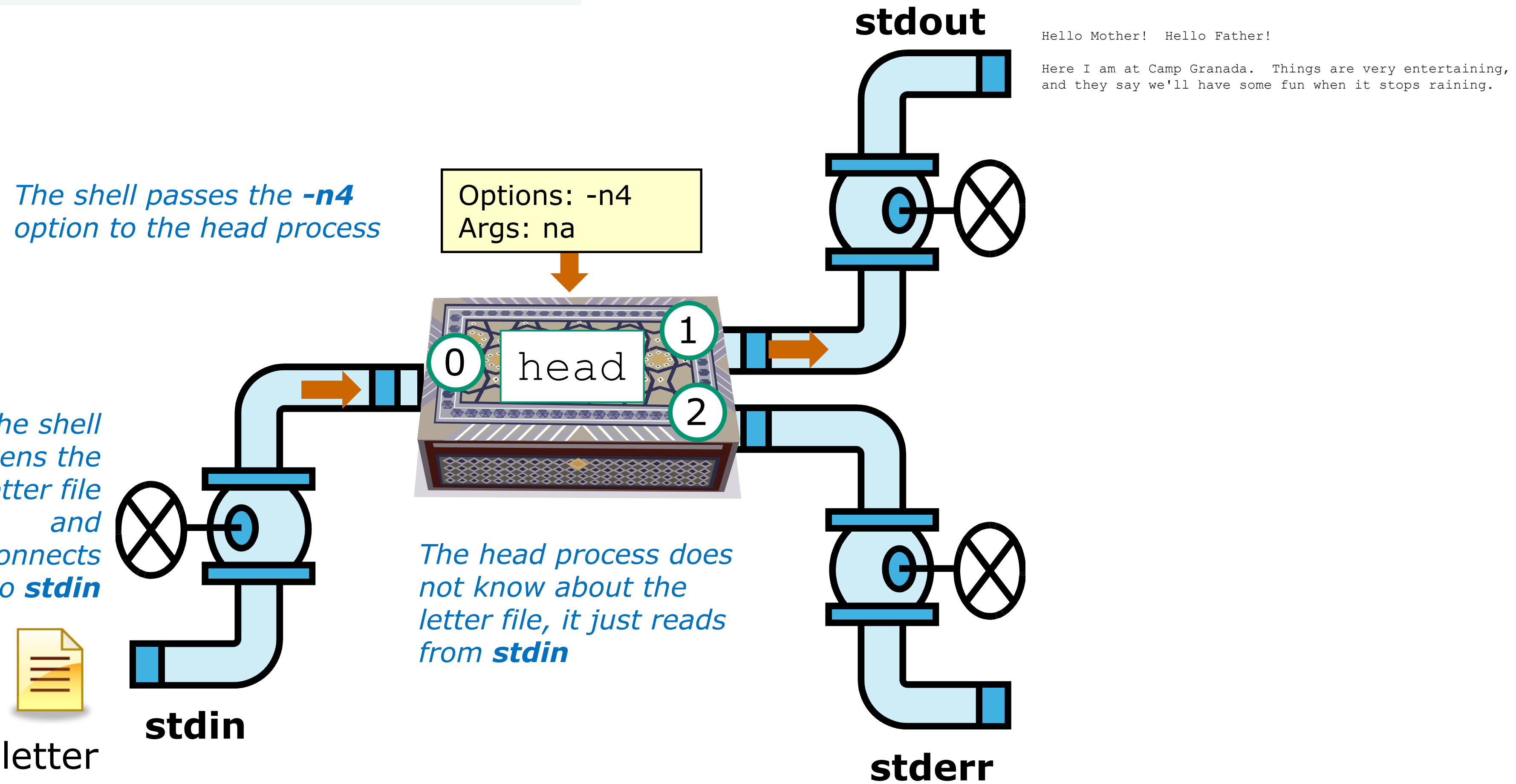
head opens and reads the letter file

head -n4 < letter

```
$ head -n4 < letter
```



*The shell passes the **-n4** option to the head process*



```
Hello Mother!  Hello Father!

Here I am at Camp Granada.  Things are very entertaining,
and they say we'll have some fun when it stops raining.
```

*The shell opens the letter file and connects it to **stdin***

*The head process does not know about the letter file, it just reads from **stdin***

Test your understanding of how the shell and command work as a team

Given: There is no file named *bogus*, associate each command on the left with an error message on the right

Commands	**Error messages**
`$ cat < bogus`	`-bash: bogus: command not found`
`$ cat bogus`	`-bash: bogus: No such file or directory`
`$ bogus`	`cat: bogus: No such file or directory`

Test your knowledge

Given: There is no file named bogus, associate each command on the left with an error message on the right

$$\begin{tikzcd}
\textbf{Commands} & \textbf{Error messages} \\
\texttt{\$ cat < bogus} \arrow[rd] & \texttt{-bash: bogus: command not found} \\
\texttt{\$ cat bogus} \arrow[rd] & \texttt{-bash: bogus: No such file or directory} \\
\texttt{\$ bogus} \arrow[ruu] & \texttt{cat: bogus: No such file or directory}
\end{tikzcd}$$

2>&1

FYI

(more on this in CIS 98)

FYI
only

FYI only

It’s descriptor clobbering time!

```
/home/cis90/simben $ bc > calculations 2> calculations
2+2
7/0
3+3
quit

/home/cis90/simben $ cat calculations
Ru6
ime error (func=(main), adr=5): Divide by zero
```

*Oops! Its not a good idea to redirect **stdout** and **sderr** to the same file because they clobber each other*

FYI only

It’s descriptor collaboration time!

```
/home/cis90/simben $ bc > calculations 2>&1
2+2
7/0
3+3
quit

/home/cis90/simben $ cat calculations
4
Runtime error (func=(main), adr=5): Divide by zero
6
```

*This is the correct way to redirect **stdout** and **sderr** to the same file*

More on I/O

(input/output)

C program example

FYI

only

FYI only

C Program I/O example

```
[rsimms@opus misc]$ cat simple.c
char question[] = "What is your name stranger? ";
char greeting[] = "Well I'm very pleased to meet you, ";
char buffer[80];
main()
{
        int len;

        write(2, question, sizeof(question));
        len = read(0, buffer, 80);
        write(1, greeting, sizeof(greeting));
        write(1, buffer, len);
}
```

This program is available in the depot directory

FYI only

C Program I/O example

```
[rsimms@opus misc]$ cat simple.c
char question[] = "What is your name stranger? ";
char greeting[] = "Well I'm very pleased to meet you, ";
char buffer[80];
main()
{
          int len;

          write(2, question, sizeof(question));
          len = read(0, buffer, 80);
          write(1, greeting, sizeof(greeting));
          write(1, buffer, len);
}
```

*Write question to **stderr***

This simple program asks for a name, then responds with a greeting using the name

FYI only

C Program I/O example

```
[rsimms@opus misc]$ cat simple.c
char question[] = "What is your name stranger? ";
char greeting[] = "Well I'm very pleased to meet you, ";
char buffer[80];
main()
{
        int len;

        write(2, question, sizeof(question));
        len = read(0, buffer, 80);
        write(1, greeting, sizeof(greeting));
        write(1, buffer, len);
}
```

*Read users name from **stdin***

This simple program asks for a name, then responds with a greeting using the name

FYI only

C Program I/O example

```
[rsimms@opus misc]$ cat simple.c
char question[] = "What is your name stranger? ";
char greeting[] = "Well I'm very pleased to meet you, ";
char buffer[80];
main()
{
        int len;

        write(2, question, sizeof(question));
        len = read(0, buffer, 80);
        write(1, greeting, sizeof(greeting));
        write(1, buffer, len);
}
```

*Write greeting to **stdout***

This simple program asks for a name, then responds with a greeting using the name

FYI only

C Program I/O example

```
[rsimms@opus misc]$ cat simple.c
char question[] = "What is your name stranger? ";
char greeting[] = "Well I'm very pleased to meet you, ";
char buffer[80];
main()
{
        int len;

        write(2, question, sizeof(question));
        len = read(0, buffer, 80);
        write(1, greeting, sizeof(greeting));
        write(1, buffer, len);
}
```

*Write users name to **stdout***

This simple program asks for a name, then responds with a greeting using the name

FYI only

C Program I/O example

The make command is used to compile a C source text file into a binary executable

```
[rsimms@opus misc]$ make simple
cc      simple.c   -o simple
```

Unlike a bash script, the C program source code must be compiled into a binary executable before it can be run

FYI only

C Program I/O example

```
[rsimms@opus misc]$ ./simple
What is your name stranger? Rich
Well I'm very pleased to meet you, Rich
```

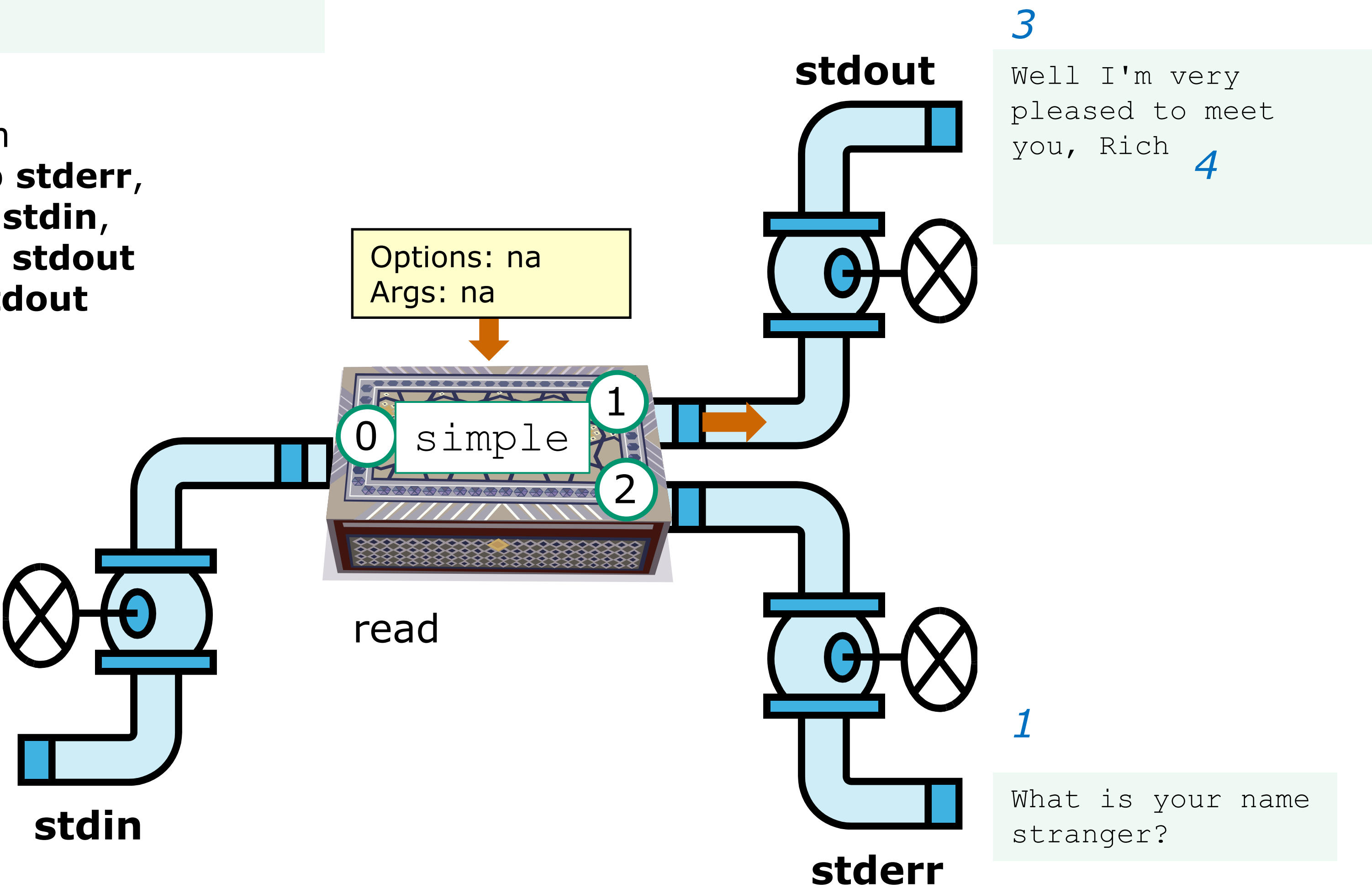
Running the simple program.

*Note I need to preface **simple** with a "**./**" to run it as this directory is not on my path. This is not necessary for CIS 90 students as they already have the . directory in their path.*

C Program I/O example

```
$ ./simple
```

The **simple** program
1. writes question to **stderr**,
2. reads name from **stdin**,
3. writes greeting to **stdout**
4. writes name to **stdout**



```
Well I'm very
pleased to meet
you, Rich
```

```
Rich
```

```
What is your name
stranger?
```

FYI only

C Program I/O example

```
[rsimms@opus misc]$ ./simple  >  myfile
What is your name stranger? Rich

[rsimms@opus misc]$ cat myfile
Well I'm very pleased to meet you, Rich
```

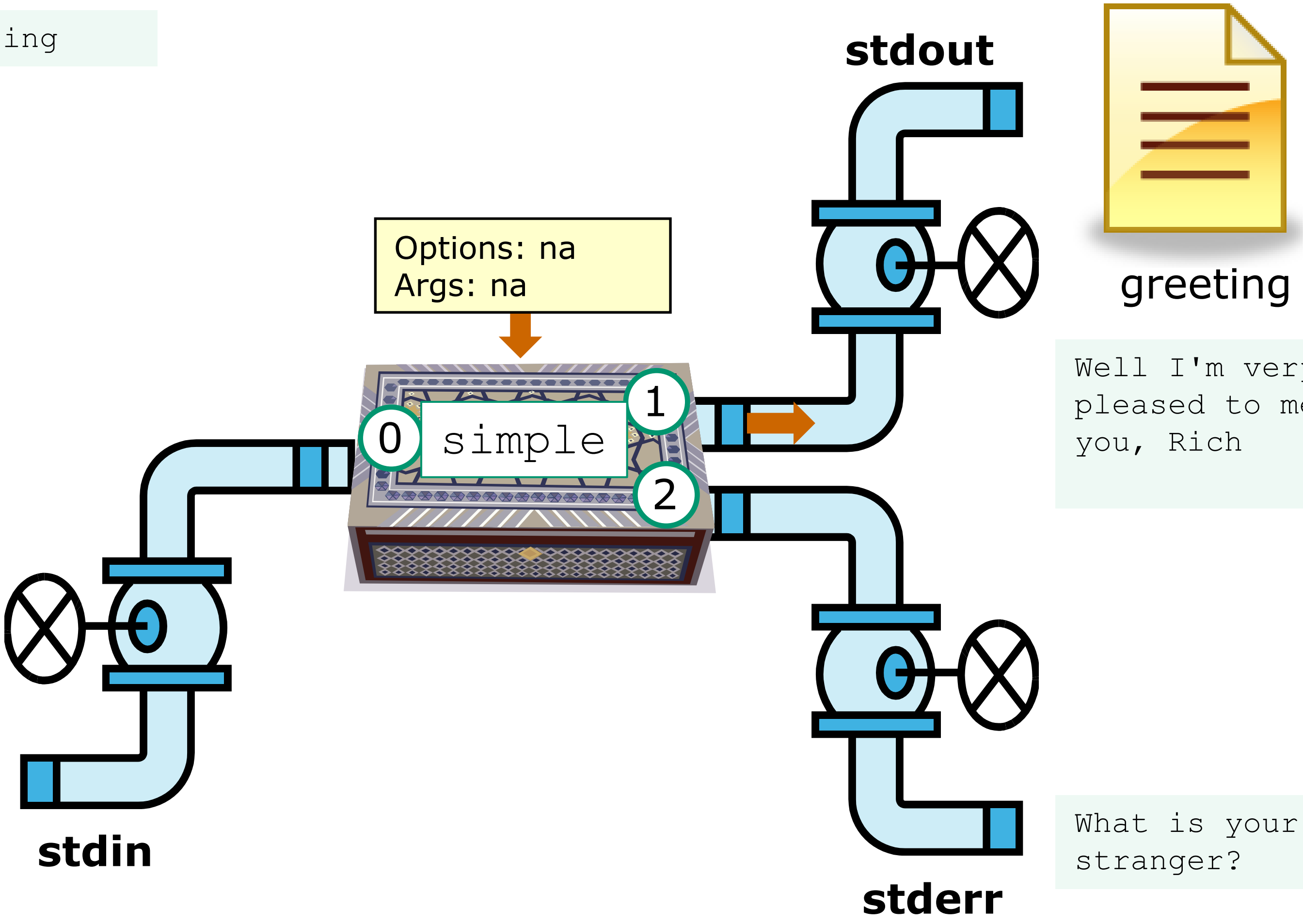
In the second example, output has been redirected to a file named myfile.

*The simple program has no special knowledge (coding instructions) for a file named myfile. It just writes to **stdout** and that output will go to wherever **stdout** had been directed.*

C Program I/O example

redirection

```
$ ./simple > greeting
```



```
Well I'm very
pleased to meet
you, Rich
```

```
Rich
```

```
What is your name
stranger?
```

Activity

1. Change to your bin directory
   ```
   cd bin
   ```

2. Copy the simple.c source code from the depot directory
   ```
   cp  ~/../depot/simple.c  .
   ```

3. Compile the program
   ```
   make simple
   ```

4. Run the program
   ```
   simple
   ```

More on umask

(shortcut)

Review - applying umask bits

Current umask setting

```
/home/cis90/simben/lesson9 $ umask
0002
```

this mask indicates which permissions should <u>NOT</u> be set on the new file or directory

New file - start with 666 and apply mask

```
666    110 110 110
002    000 000 010
       -----------
664    110 110 100
```

```
/home/cis90/simben/lesson9 $ touch newfile
/home/cis90/simben/lesson9 $ ls -l newfile
-rw-rw-r-- 1 simben cis90 0 Oct 27 07:22 newfile
```

New directory - start with 777 and apply mask

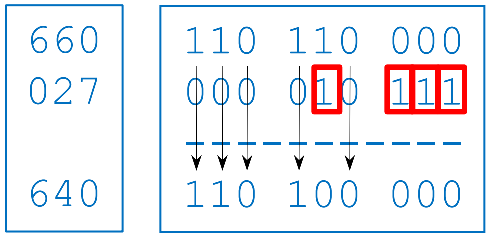
```
777    111 111 111
002    000 000 010
       -----------
775    111 111 101
```

```
/home/cis90/simben/lesson9 $ mkdir newdir
/home/cis90/simben/lesson9 $ ls -ld newdir
drwxrwxr-x 2 simben cis90 4096 Oct 27 07:23 newdir
```

Any umask bits set to 1 remove the corresponding permission bit for the new file or directory

Review - Copying files

```
/home/cis90/simben/lesson9 $ umask 027
/home/cis90/simben/lesson9 $ umask
0027

/home/cis90/simben/lesson9 $ chmod 660 myfile
/home/cis90/simben/lesson9 $ cp myfile myfile.bak
/home/cis90/simben/lesson9 $ ls -l myfile*
-rw-rw---- 1 simben cis90 0 Oct 27 08:02 myfile
-rw-r----- 1 simben cis90 0 Oct 27 08:04 myfile.bak
```



Start with original file's permissions and apply the mask

Remember, for new files resulting from copying, instead of using the ***default permissions*** *(666 for file and 777 for directory), use the* ***original file permissions*** *as the starting point for the mask to be applied to.*

Pipeline Practice

(from last lesson)

Class Exercise
Pipeline Tasks

Background
The **last** command searches through /var/log/wtmp and prints out a list of users logged in since that file was created.

Task
Can you see the last times you were logged in on a Tuesday and then count them?

```
cat /var/log/wtmp* > logins
last -f logins | grep $LOGNAME
last -f logins | grep $LOGNAME | grep "Tue"
last -f logins | grep $LOGNAME | grep "Tue" | wc -l
```

How many times have you logged in on a Tuesday?
Put your answer in the chat window.

More Pipeline Practice

(from last lesson)

Class Exercise
Pipeline Tasks

Background
The **cut** command can cut a field out of a line of text where each field is delimitated by some character.

The */etc/passwd* file uses the ":" as the delimiter between fields. The 5th field is a comment field for the user account.

Task
Build up a pipeline, one pipe at a time:

```
cat /etc/passwd
cat /etc/passwd | grep cis90
cat /etc/passwd | grep $LOGNAME
cat /etc/passwd | grep $LOGNAME | cut -f 5 -d ":"
cat /etc/passwd | grep $LOGNAME | cut -f 5 -d ":" | cut -f2 -d" "
```

What gets printed with the last pipeline?
Put your answer in the chat window.

More on pipelines

Not all commands are filters
(filters read from stdin and write to stdout)

*The **wc** command is a filter.*

```
/home/cis90/simben $ head -n2 poems/Anon/nursery
Jack and Jill went up the hill
to fetch a pail of water.
/home/cis90/simben $ head -n2 poems/Anon/nursery | wc -l
2
/home/cis90/simben $
```

*But the **echo** command isn't (doesn't read from **stdin**)*

```
/home/cis90/simben $ head -n2 poems/Anon/nursery | echo

/home/cis90/simben $
```

xargs command

***xargs** to the rescue!*

*The **xargs** command will read **stdin** and call another command using the input as the arguments.*

```
/home/cis90/simben $ head -n2 poems/Anon/nursery | xargs echo
Jack and Jill went up the hill to fetch a pail of water.
```

Another example

Why can't Benji make a banner using the output of the date command?

```
/home/cis90/simben $ date | banner
Enter a string of up to 10 characters.
/home/cis90/simben $
```

huh? Oh, this is what banner prints when it receives no arguments on the command line

Because banner is not a filter and does not read from stdin!

Another example

```
/home/cis90/simben $ date | xargs banner
#     # ####### #     #
##   ## #     # ##    #
# # # # #     # # #   #
#  #  # #     # #  #  #
#     # #     # #   # #
#     # #     # #    ##
#     # ####### #     #

#######  #####  #######
#     # #     #    #
#     # #          #
#     # #          #
#     # #          #
#     # #     #    #
#######  #####     #

 #####   #####
#     # #     #
      #       #
 #####   #####
#       #
#       #
####### #######

   #       #             #####   #####          #######  #####
  ##      ##      ###   #     # #     #   ###   #       #     #
 # #     # #      ###         # #         ###   #       #     #
   #       #             #####  ######          ######   #####
   #       #      ###   #       #     #   ###         # #     #
   #       #      ###   #       #     #   ###   #     # #     #
 #####   #####          #######  #####           #####   #####

######  ######  #######
#     # #     #    #
#     # #     #    #
######  #     #    #
#       #     #    #
#       #     #    #
#       ######     #

 #####    ###      #     #####
#     #  #   #    ##    #     #
      # #     #  # #          #
 #####  #     #    #     #####
#       #     #    #    #
#        #   #     #    #
#######   ###    #####  #######
```

xargs *to the rescue again!*

Not all commands are filters
(filters read from stdin and write to stdout)

*The **ls** command does not read from **stdin** either*

```
/home/cis90/simben $ find poems -type d
poems
poems/Shakespeare
poems/Yeats
poems/Anon
poems/Blake


/home/cis90/simben $ find poems -type d | ls -ld
drwxr-xr-x. 18 simben90 cis90 4096 Oct 22 09:49 .
/home/cis90/simben $
```

Benji was hoping that he could get a long listing of his poems directory and all its sub-directories. Instead he gets a long listing of his home directory!

Not all commands are filters
(filters read from stdin and write to stdout)

```
/home/cis90/simben $ find poems -type d | xargs ls -ld
drwxr-xr-x. 6 simben90 cis90 4096 Oct 20 15:06 poems
drwxr-xr-x. 2 simben90 cis90 4096 Oct  5 10:26 poems/Anon
drwxr-xr-x. 2 simben90 cis90 4096 Oct 20 15:06 poems/Blake
drwxr-xr-x. 2 simben90 cis90 4096 Oct 20 15:06 poems/Shakespeare
drwxr-xr-x. 2 simben90 cis90 4096 Oct 20 15:06 poems/Yeats
/home/cis90/simben $
```

*The **ls** command is not a filter so it does not read from **stdin***

***xargs** to the rescue!*

***xargs** reads the names of the files found by the **find** command and uses them as arguments on the **ls -ld** command*

Not all commands are filters
(filters read from stdin and write to stdout)

```
/home/cis90/simben $ find poems -type d -exec ls -ld {} \;
drwxr-xr-x. 6 simben90 cis90 4096 Oct 20 15:06 poems
drwxr-xr-x. 2 simben90 cis90 4096 Oct 20 15:06 poems/Shakespeare
drwxr-xr-x. 2 simben90 cis90 4096 Oct 20 15:06 poems/Yeats
drwxr-xr-x. 2 simben90 cis90 4096 Oct  5 10:26 poems/Anon
drwxr-xr-x. 2 simben90 cis90 4096 Oct 20 15:06 poems/Blake
/home/cis90/simben $
```

By the way, the find command also has a ***-exec*** *option that will run a command on what is found. The* ***{}*** *represent the arguments which are names of files found by the* ***find*** *command.*

Housekeeping

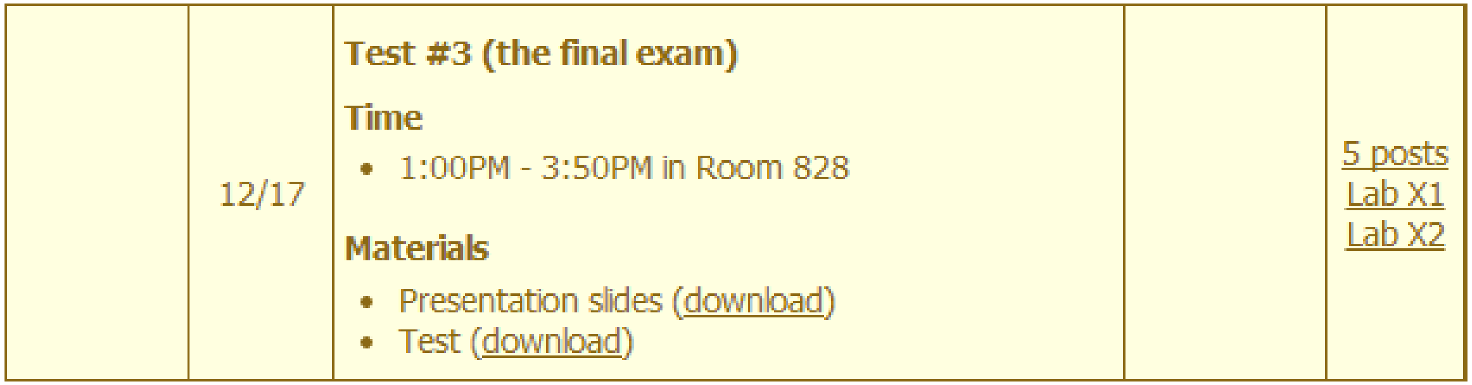
Housekeeping

1. Lab 7 due 11:59PM tonight
2. A **check7** script is available
3. Test #2 is <u>next week</u>
4. Practice Test #2 available now
5. No lab assigned this week (so you can work on the practice test)

Final Exam

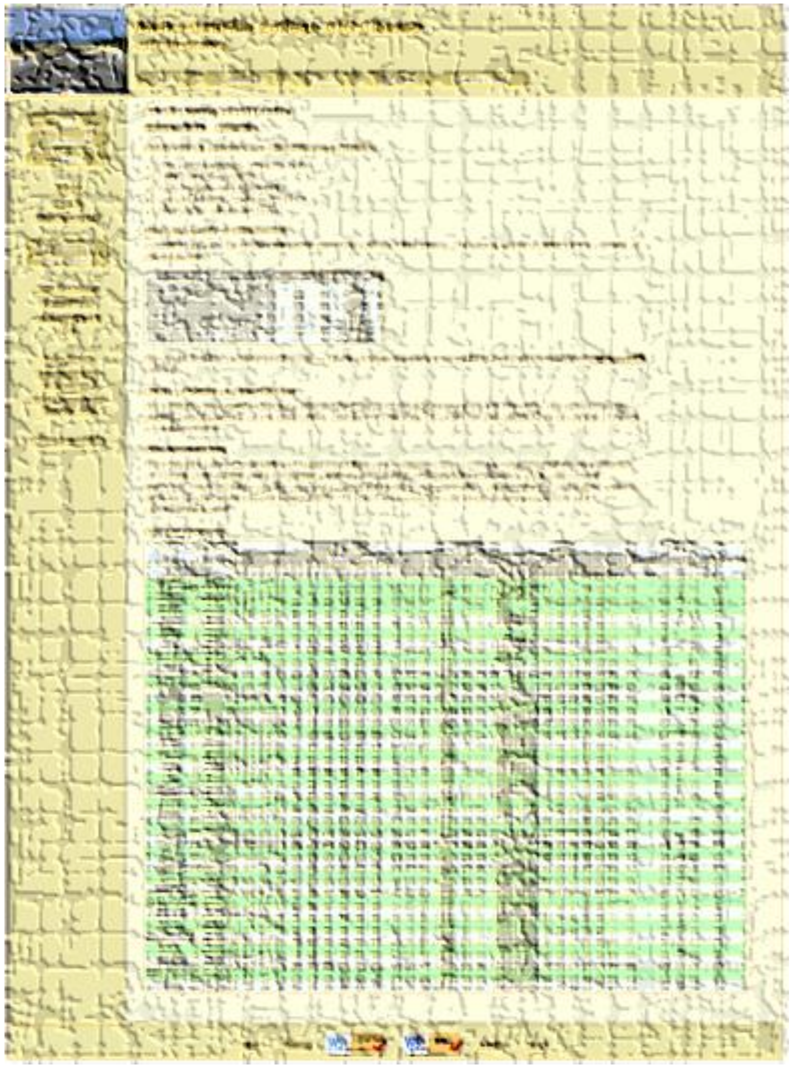
Test #3 (final exam)

- Must be face-to-face or proctored (not online using CCC Confer).
- We will be in room 828 on campus.



	12/17	**Test #3 (the final exam)** **Time** • 1:00PM - 3:50PM in Room 828 **Materials** • Presentation slides (download) • Test (download)		5 posts Lab X1 Lab X2

http://simms-teach.com/cis90grades.php



GRADES

- Check your progress on the Grades page
- If you haven't already, send me a student survey to get your LOR secret code name
- Graded labs & tests are placed in your home directories on Opus
- Answers to labs, tests and quizzes are in the */home/cis90/answers* directory on Opus

Current Point Tally

As of 10/28/2013

Points that could have been earned:

6 quizzes:	18 points
6 labs:	180 points
1 test:	30 points
2 forum quarters:	40 points
Total:	**268 points**

Percentage	Total Points	Letter Grade	Pass/No Pass
90% or higher	504 or higher	A	Pass
80% to 89.9%	448 to 503	B	Pass
70% to 79.9%	392 to 447	C	Pass
60% to 69.9%	336 to 391	D	No pass
0% to 59.9%	0 to 335	F	No pass

adaldrida: 98% (265 of 268 points)
anborn: 0% (0 of 268 points)
aragorn: 98% (263 of 268 points)
arwen: 84% (226 of 268 points)
balrog: 55% (150 of 268 points)
barliman: 1% (4 of 268 points)
beregond: 71% (191 of 268 points)
boromir: 2% (8 of 268 points)
celebrian: 82% (220 of 268 points)
dori: 54% (146 of 268 points)
dwalin: 86% (231 of 268 points)
elrond: 96% (258 of 268 points)
eomer: 82% (220 of 268 points)
faramir: 100% (269 of 268 points)
frodo: 96% (258 of 268 points)
gimli: 95% (257 of 268 points)
goldberry: 105% (284 of 268 points)

huan: 45% (122 of 268 points)
ingold: 100% (269 of 268 points)
ioreth: 70% (188 of 268 points)
legolas: 73% (198 of 268 points)
marhari: 101% (271 of 268 points)
pallando: 103% (278 of 268 points)
pippen: 94% (253 of 268 points)
quickbeam: 39% (105 of 268 points)
samwise: 81% (219 of 268 points)
sauron: 101% (273 of 268 points)
shadowfax: 69% (187 of 268 points)
strider: 86% (232 of 268 points)
theoden: 101% (272 of 268 points)
treebeard: 89% (241 of 268 points)
tulkas: 99% (266 of 268 points)
ulmo: 61% (166 of 268 points)

If you are not happy with your current standing contact the instructor ASAP

Jesse’s checkgrades python script

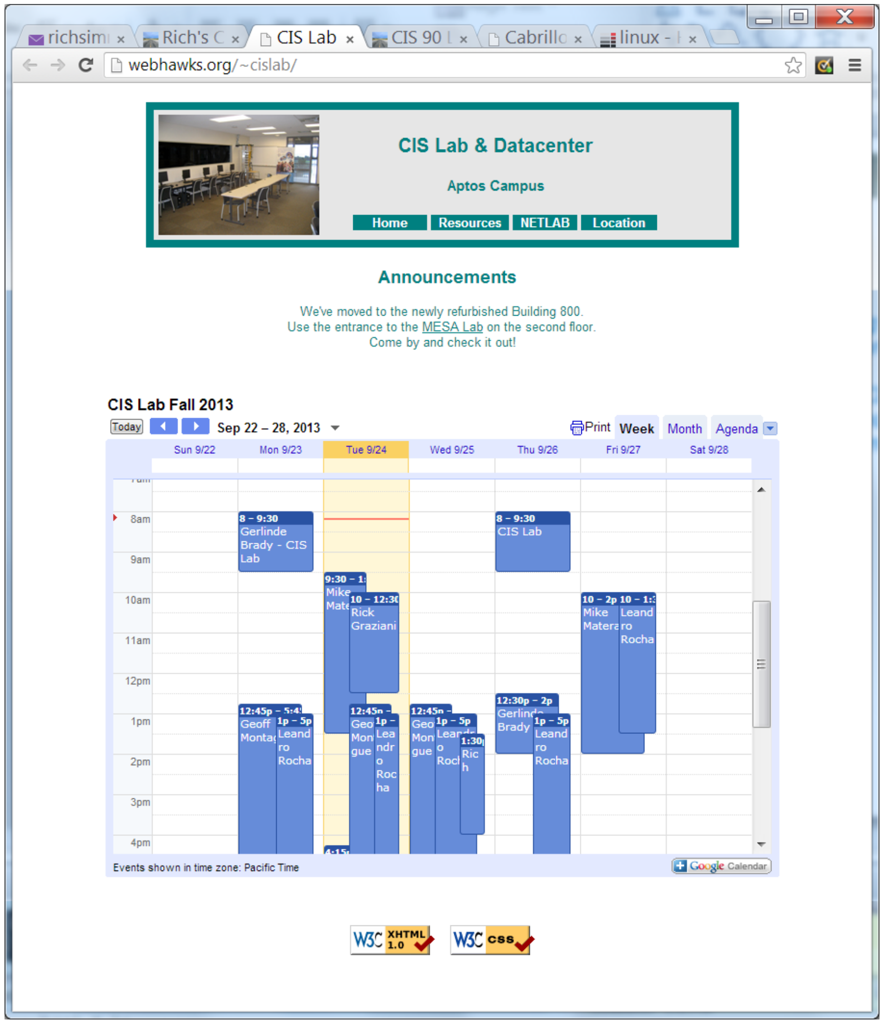
http://oslab.cabrillo.edu/forum/viewtopic.php?f=31&t=773&p=2966

```
/home/cis90/simben $ checkgrades smeagol

Remember, your points may be zero simply because the
assignment has not been graded yet.

Quiz 1: You earned 3 points out of a possible 3.
Quiz 2: You earned 3 points out of a possible 3.
Quiz 3: You earned 3 points out of a possible 3.
Quiz 4: You earned 3 points out of a possible 3.


Forum Post 1: You earned 20 points out of a possible 20.

Lab 1: You earned 30 points out of a possible 30.
Lab 2: You earned 30 points out of a possible 30.
Lab 3: You earned 30 points out of a possible 30.
Lab 4: You earned 29 points out of a possible 30.


You've earned 15 points of extra credit.

You currently have a 109% grade in this class. (166 out of
152 possible points.)
```

Use your LOR code name as an argument on the checkgrades command

Jesse is a CIS 90 Alumnus. He wrote this python script when taking the course. It mines data from the website to check how many of the available points have been earned so far.

CIS 90 - Lesson 9



webhawks.org/~cislab/

CIS Lab & Datacenter

Aptos Campus

Announcements

We've moved to the newly refurbished Building 800.
Use the entrance to the MESA Lab on the second floor.
Come by and check it out!

CIS Lab Fall 2013

Sep 22 – 28, 2013

	Sun 9/22	Mon 9/23	Tue 9/24	Wed 9/25	Thu 9/26	Fri 9/27	Sat 9/28
8am		8 – 9:30 Gerlinde Brady - CIS Lab			8 – 9:30 CIS Lab		
9am			9:30 – 1 Mike Mate				
10am			10 – 12:30 Rick Graziani			10 – 2p Mike Matera; 10 – 1 Leandro Rocha	
12pm		12:45p – 5:4 Geoff Monta; 1p – 5p Leandro Rocha	12:45p Geo Mon gue; 1p Lea ndr o Roc ha	12:45p Geo Mon gue; 1p – 5p Lean o Roch; 1:30 Ric h	12:30p – 2p Gerlinde Brady; 1p – 5p Leandro Rocha		

Events shown in time zone: Pacific Time

Google Calendar

W3C XHTML 1.0 W3C CSS

CIS Lab Schedule
http://webhawks.org/~cislab/

Work on assignments together with other classmates

Get help from instructors and student lab assistants

MESA grants requires logging help sessions with MESA funded student assistants

Things that Hide

trick or treat

A number of *trick* and *treat* files have been distributed within your home directory and sub-directories!

1. Can you find them? There should be an obvious one in your home directory. The rest are scattered in the various subdirectories you own.

2. Make a new directory named *bag* in your home directory and see how many *trick* or *treat* files you can move into it.

3. Put a Green Check in CCC Confer next to your name when you have collected 3 treats, electronically "clap" if you collect all six treats and six tricks.

Instructor: sudo /home/rsimms/cis90/halloween/hidetreats

Review

Jim's Summary Pages

Jim has some really good summary information on Lessons 6-8 on his web site:

Lesson 6 - Managing Files
http://cabrillo.edu/~jgriffin/CIS90/files/lecture5.html

Lesson 7 - File Permissions
http://cabrillo.edu/~jgriffin/CIS90/files/lecture6.html

Lesson 8 - Input/Output Processing
http://cabrillo.edu/~jgriffin/CIS90/files/lecture7.html

Make Teams

Breakout Rooms

debian	redhat	suse	CentOS	ubuntu	
Room 1	**Room 2**	**Room 3**	**Room 4**	**Room 5**	**Room 6**

Once you are in your rooms:
1) Write your team's distro name at the top of your room's white board
2) Everyone write their first names under the distro's team name
3) If you want to be fancy add your distro logo to the top of your room's white board!

Make Teams:
CCC Confer: Tools > Breakout Rooms > Create Breakout Rooms … (make 6 rooms)

Flashcard Practice

Flashcards

Room 1	Room 2	Room 3	Room 4	Room 5	Room 6
Points:	**Points:**	**Points:**	**Points:**	**Points:**	**Points:**

Flashcards
L6=20
L7=15
L8=16

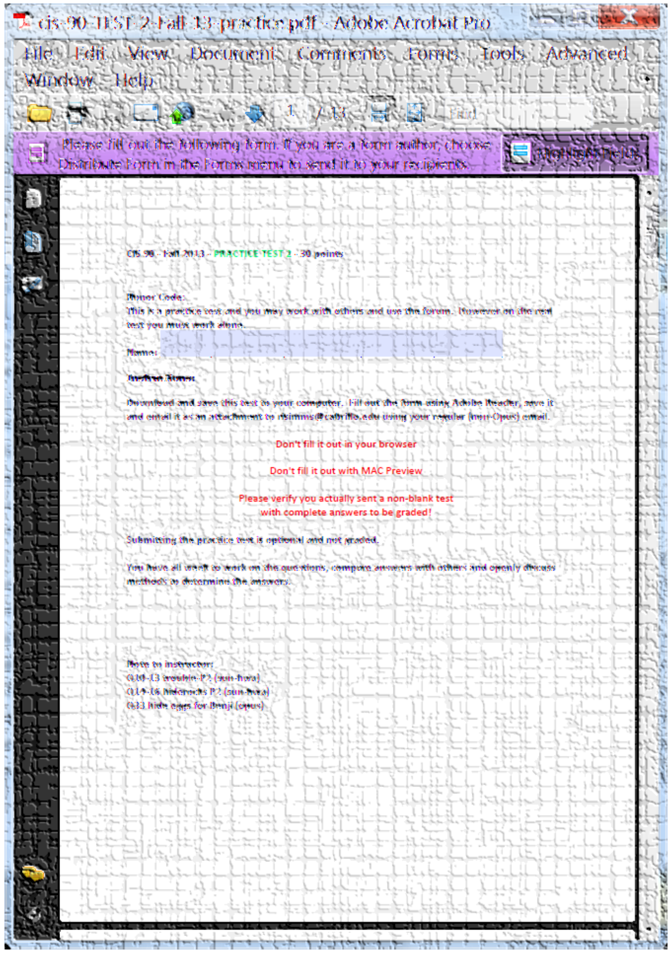
Rules

- Chat window belongs to team that is up
- Team gets the point if anyone on the team writes a correct answer in the chat window in 15 seconds

Instructor timer:

```
i=15; while [ $i -gt 0 ]; do clear; banner $i; let i=i-1; sleep 1;  done; clear; banner done
```

Practice Test


cis-90-TEST-2-Fall-13-practice.pdf - Adobe Acrobat Pro
File Edit View Document Comments Forms Tools Advanced Window Help
1 / 13
Find
Please fill out the following form. If you are a form author, choose Distribute Form in the Forms menu to send it to your recipients.
Highlight Fields

CIS 90 - Fall 2013 - PRACTICE TEST 2 - 30 points

Honor Code:
This is a practice test and you may work with others and use the forum. However on the real test you must work alone.

Name:

Instructions:

Download and save this test to your computer. Fill out the form using Adobe Reader, save it and email it as an attachment to rsimms@cabrillo.edu using your regular (non-Opus) email.

Don't fill it out in your browser

Don't fill it out with MAC Preview

Please verify you actually sent a non-blank test
with complete answers to be graded!

Submitting the practice test is optional and not graded.

You have all week to work on the questions, compare answers with others and openly discuss methods to determine the answers.

Note to instructor:
Q10-13 trouble-P2 (sun-hwa)
Q14-16 hiderocks P2 (sun-hwa)
Q33 hide eggs for Benji (opus)

Practice test available

- Work alone or together
- Use the forum to compare answers and approaches to questions

Note to instructor:
Remove /etc/nologin (sun-Hwa)
Q10-13 trouble-P2 (sun-hwa)
Q14-16 hiderocks P2 (sun-hwa)
Q33 hide treats for Homer (opus)

Breakout Rooms

Room 1	Room 2	Room 3	Room 4	Room 5	Room 6

Return to your rooms:
1) Work together on your practice test question
   - Rooms 1 & 2 work on Q28
   - Rooms 3 & 4 work on Q18
   - Rooms 5 & 6 work on Q10
2) Write how you solved it on your white board
3) Write your answer on your white board

Make Teams:
CCC Confer: Tools > Breakout Rooms > Create Breakout Rooms … (make 6 rooms)

Wrap up

Next Class

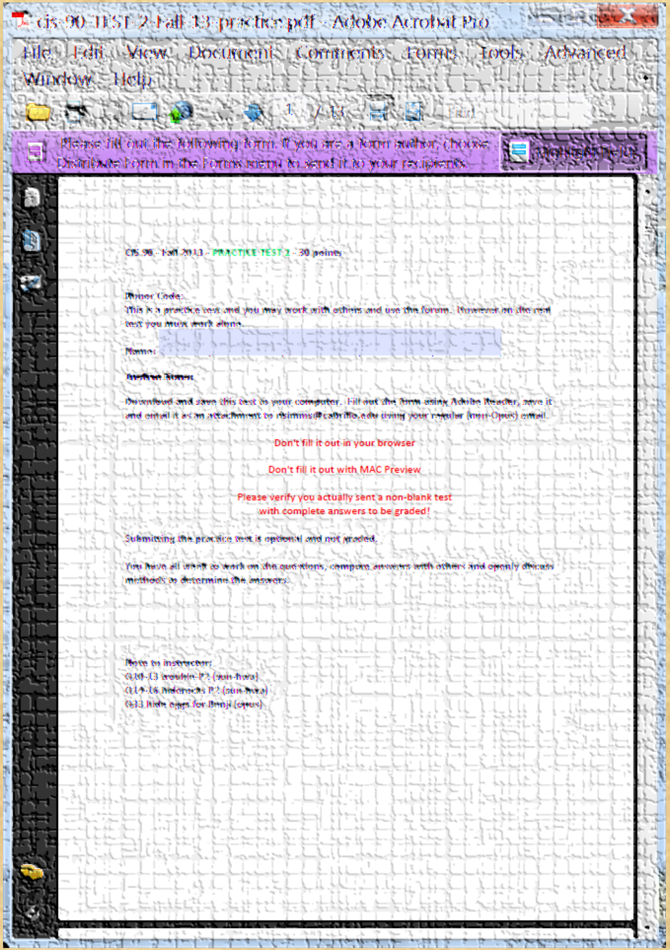
No Quiz

Test 2

Cumulative Test (30 points) with focus on Lessons 6-8:

- Recommended preparation:
  - **Work the practice test!**
  - **Work the practice test!**
  - **Work the practice test!**
  - **Collaborate with others on the forum to compare answers**
  - Review Lessons 6-8 slides and Labs 5-7
  - Try doing some or all of Lab X2 (pathnames)
  - Practice with flash cards
  - Scan previous Lessons so you know where to find things if needed

Optional Workshop Today


cis-90-TEST-2-Fall-13-practice.pdf - Adobe Acrobat Pro
File Edit View Document Comments Forms Tools Advanced Window Help
1 / 13
Find
Highlight Fields

Please fill out the following form. If you are a form author, choose Distribute Form in the Forms menu to send it to your recipients.

CIS 90 - Fall 2013 - PRACTICE TEST 2 - 30 points

Honor Code:
This is a practice test and you may work with others and use the forum. However on the real test you must work alone.

Name:

Instructions:

Download and save this test to your computer. Fill out the form using Adobe Reader, save it and email it as an attachment to rsimms@cabrillo.edu using your regular (non-Opus) email.

Don't fill it out in your browser

Don't fill it out with MAC Preview

Please verify you actually sent a non-blank test
with complete answers to be graded!

Submitting the practice test is optional and not graded.

You have all week to work on the questions, compare answers with others and openly discuss methods to determine the answers.

Note to instructor:
Q10-13 trouble-P2 (sun-hwa)
Q14-16 hiderocks P2 (sun-hwa)
Q33 hide eggs for Benji (opus)

Work the practice test till the end of class today

- Collaborate!
- Ask questions!
- You may leave class once you know how to approach and hopefully answer each question

Backup