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/)

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:

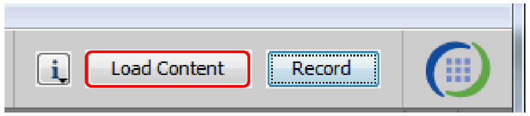
See electronic white board

email answers to: `risimms@cabrillo.edu`

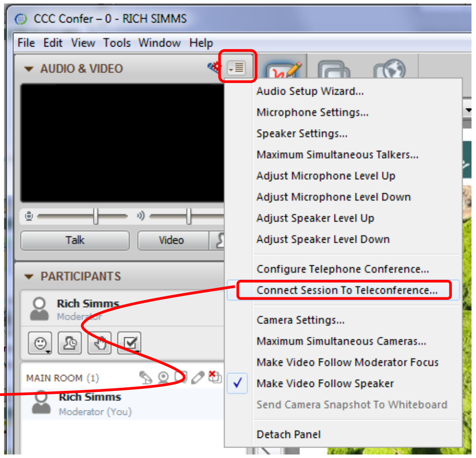
(answers must be emailed within the first few minutes of class for credit)

CIS 90 - Lesson 9

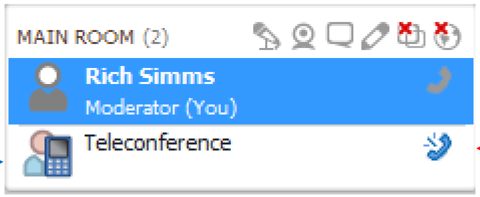
[ ] **Preload White Board with *cis*lesson??*-WB***



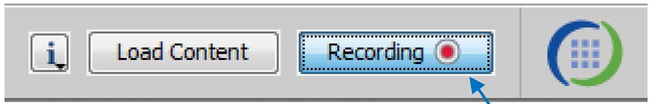
[ ] **Connect session to Teleconference**


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



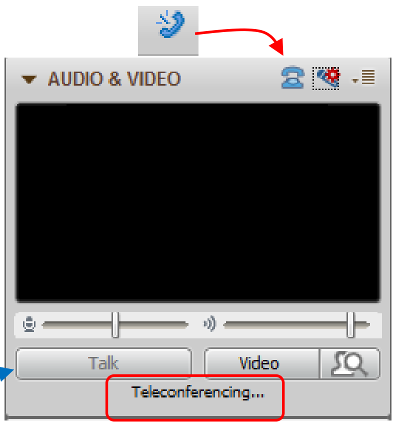
[ ] **Is recording on?**



Red dot means recording

[ ] **Use teleconferencing, not mic**

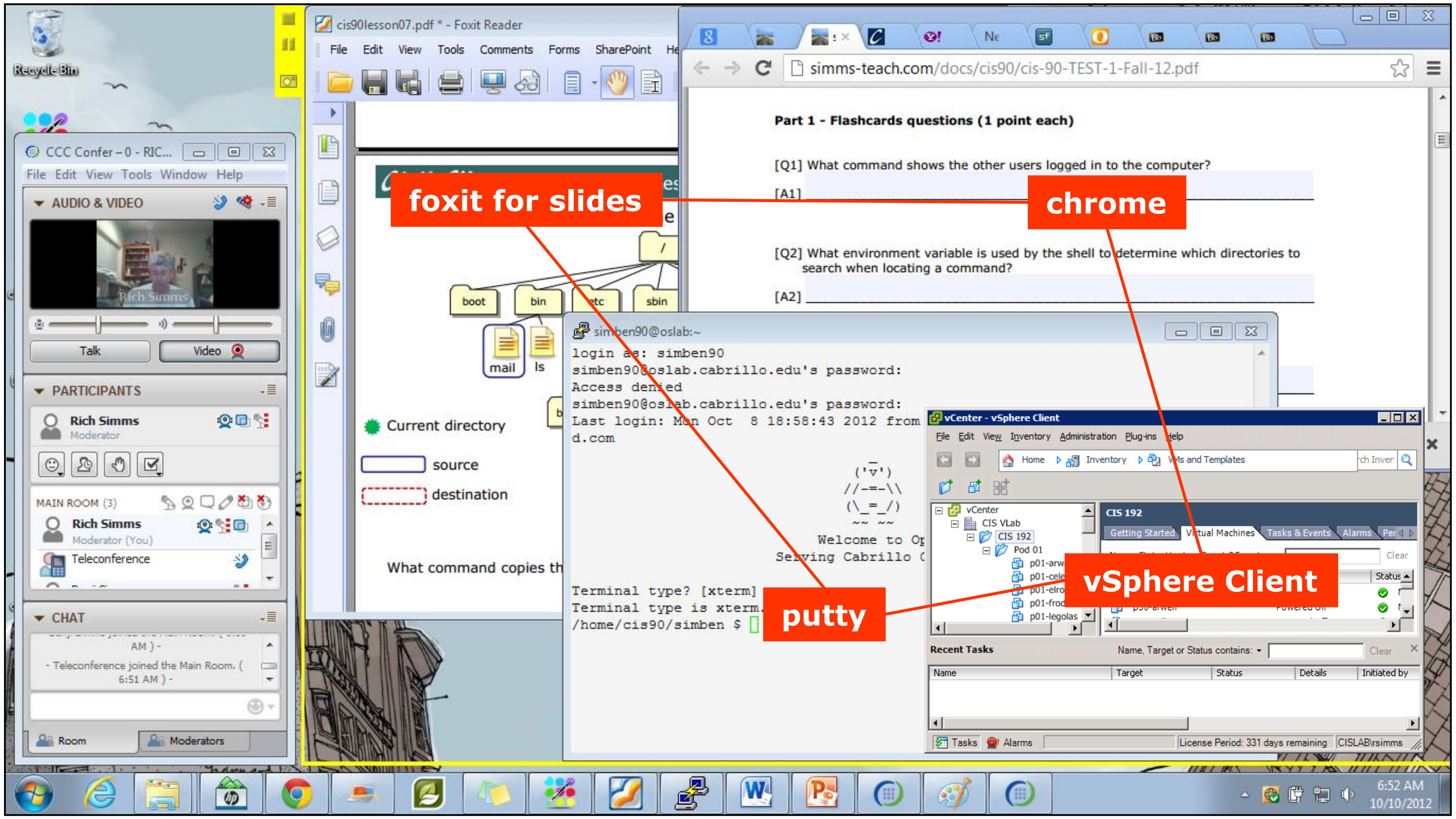
Should be greyed out



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]

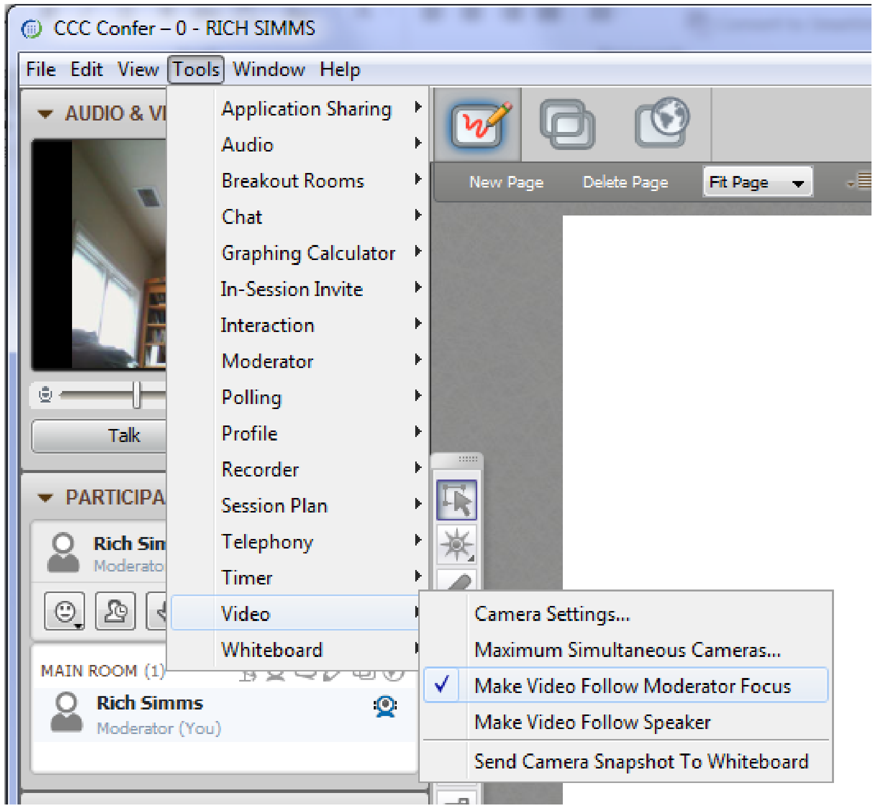
What command copies th

```
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

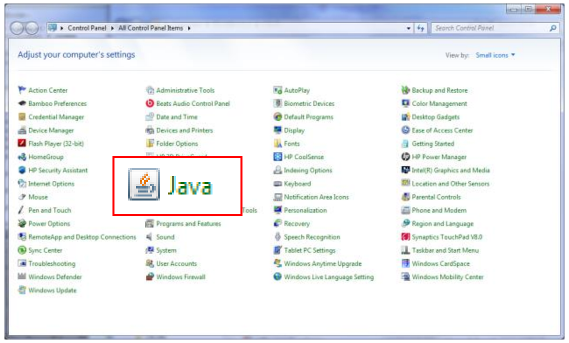


Camera Settings...
Maximum Simultaneous Cameras...
✓ Make Video Follow Moderator Focus
Make Video Follow Speaker
Send Camera Snapshot To Whiteboard

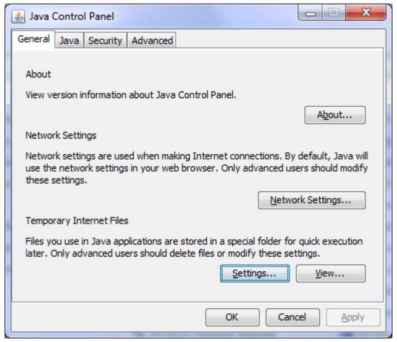
Universal Fix for CCC Confer:

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

Control Panel (small icons)



General Tab > Settings...



About

View version information about Java Control Panel.

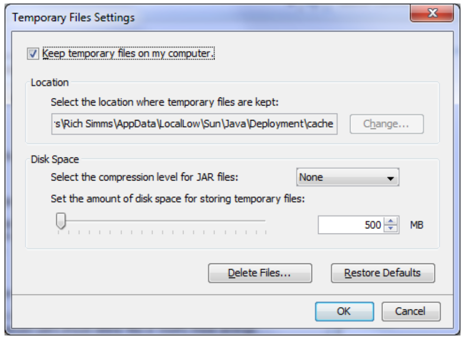
Network Settings

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:

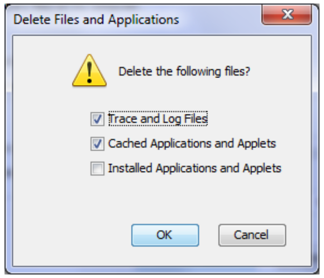
s\Rich Simms\AppData\LocalLow\Sun\Java\Deployment\cache

Disk Space

Select the compression level for JAR files: None

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

Delete these



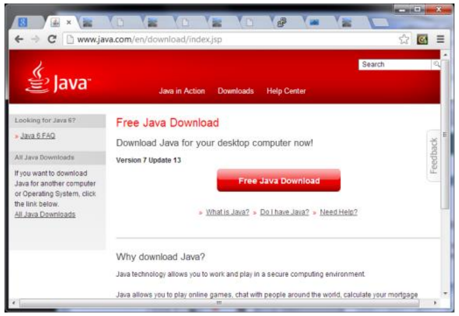
Delete the following files?

[✓] Trace and Log Files

[✓] Cached Applications and Applets

[ ] Installed Applications and Applets

Google Java download



Free Java Download

Download Java for your desktop computer now!

Version 7 Update 13

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

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.*

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*

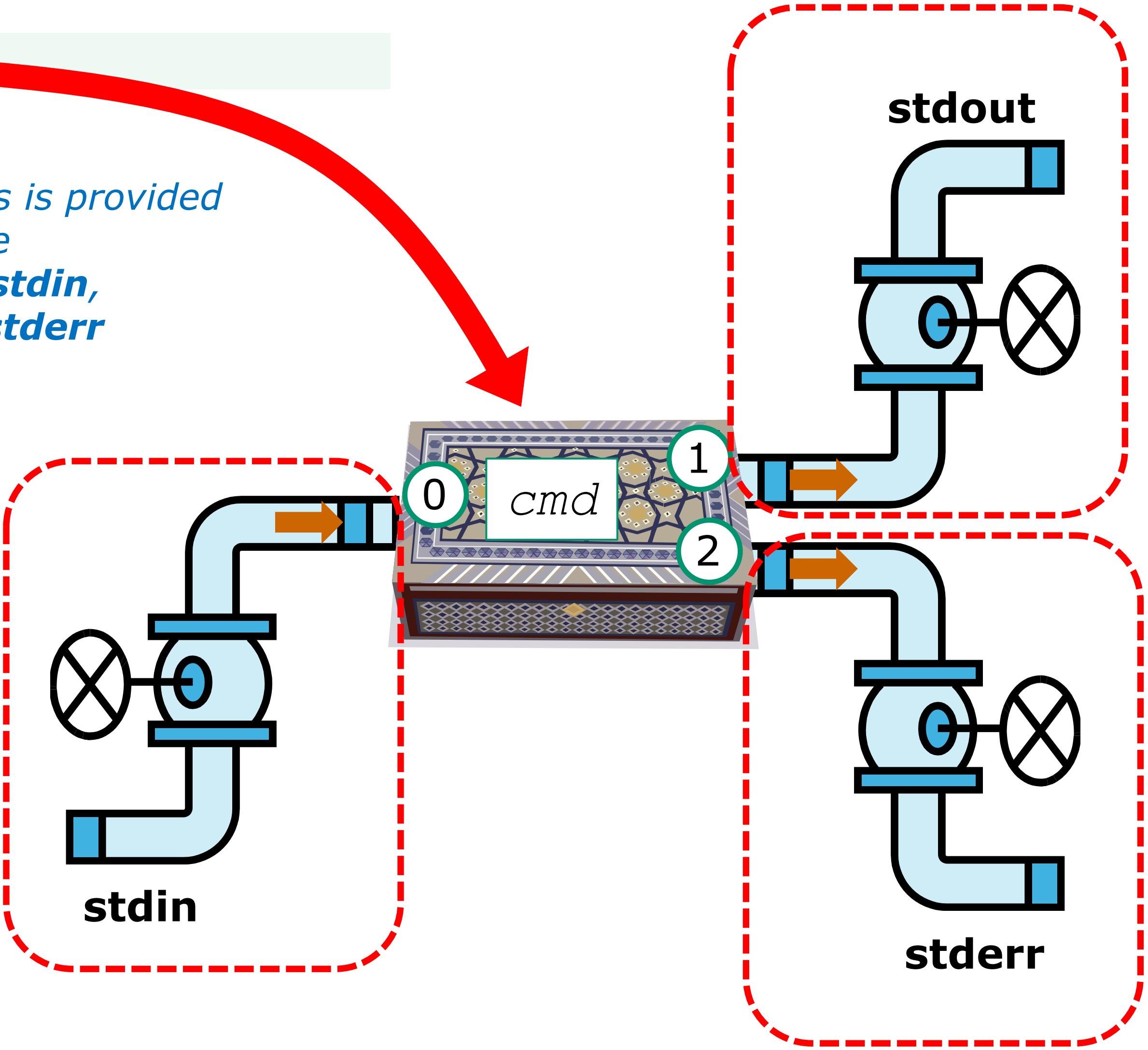
Examples

```
/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**

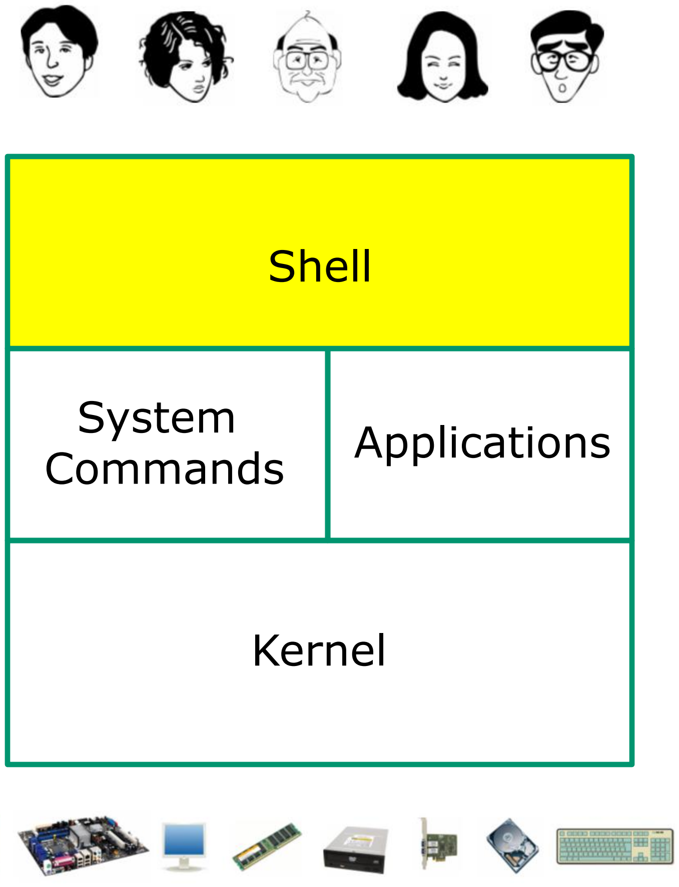
```
$ cmd
```

*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.

```
/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

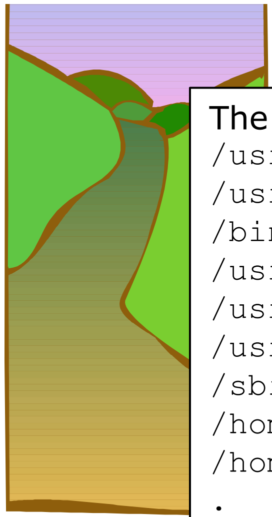
1) Prompt
2) Parse
3) Search
4) Execute
5) Nap
6) Repeat

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 students path.

Example

argument *option* *redirection*

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

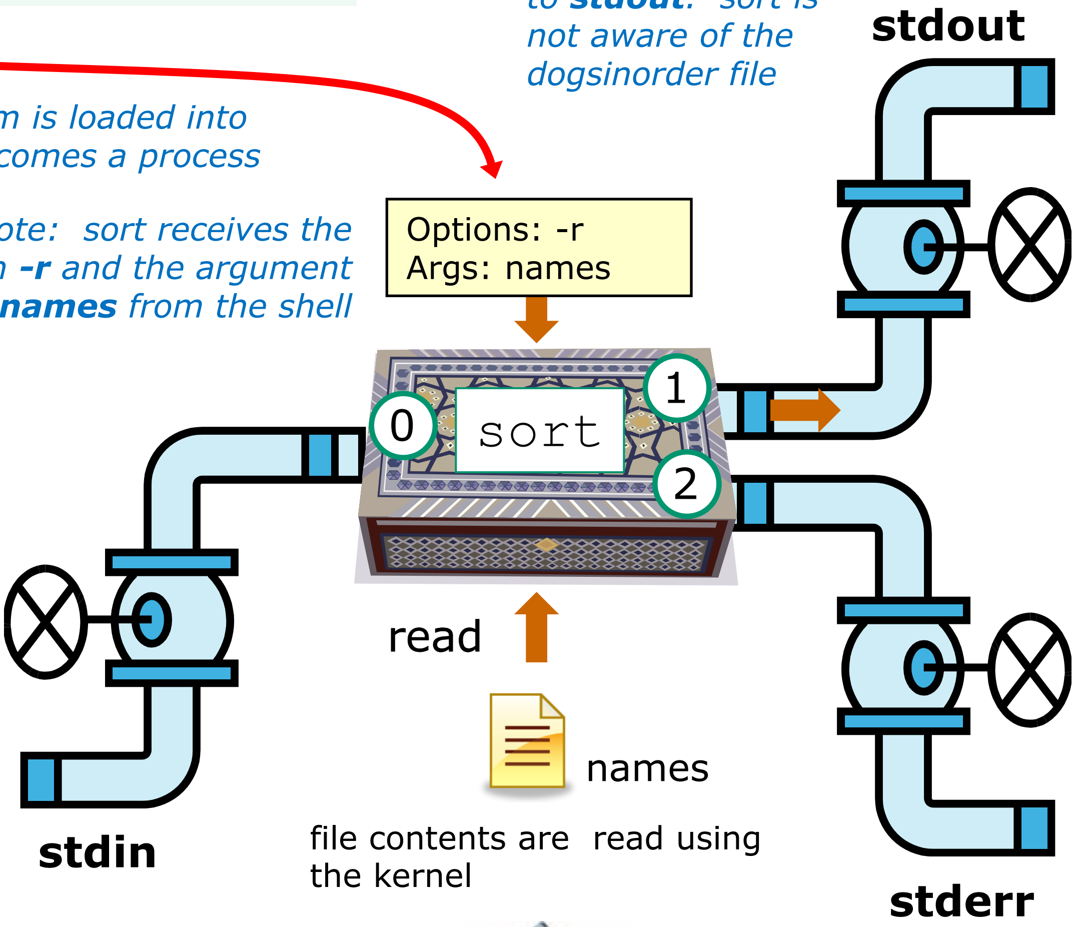
The sort program is loaded into memory and becomes a process

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

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

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

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



```
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.
```

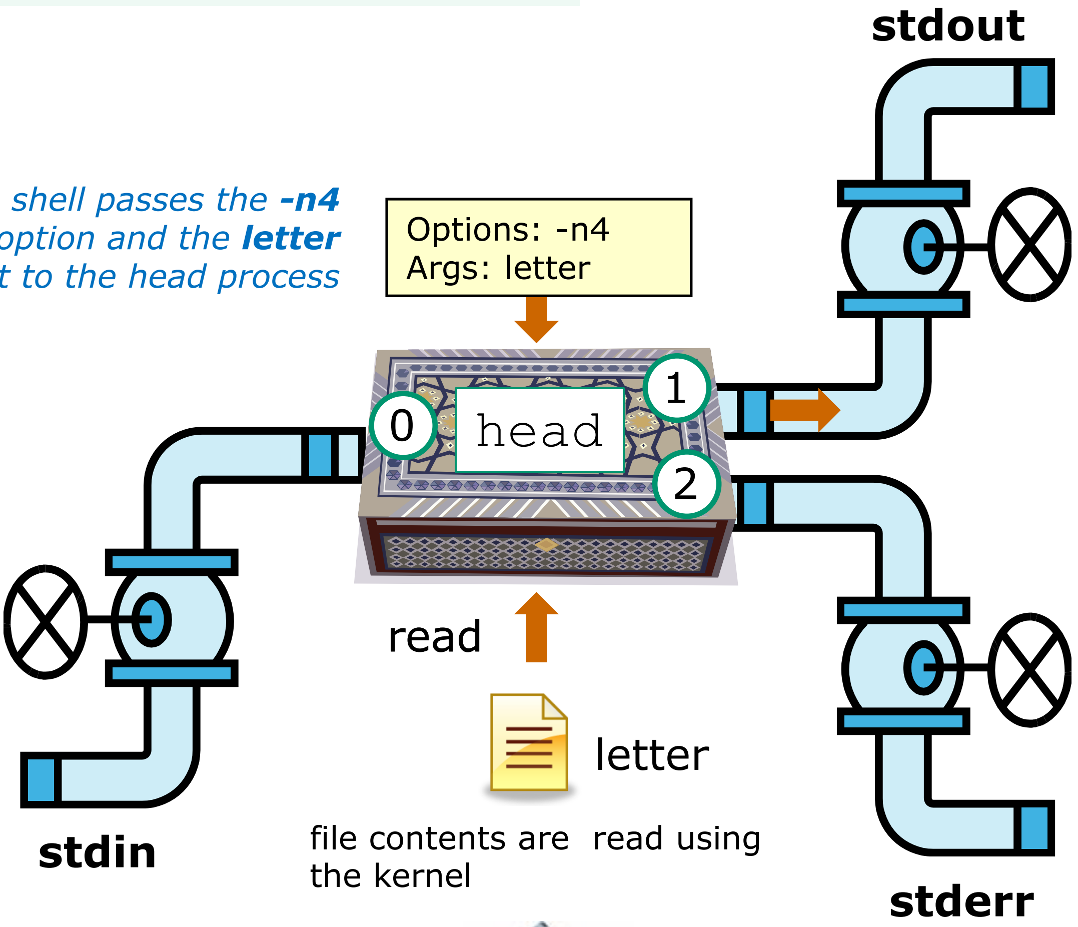
```
/home/cis90/simben $  head -n4 < letter
Hello Mother!  Hello Father!

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

option

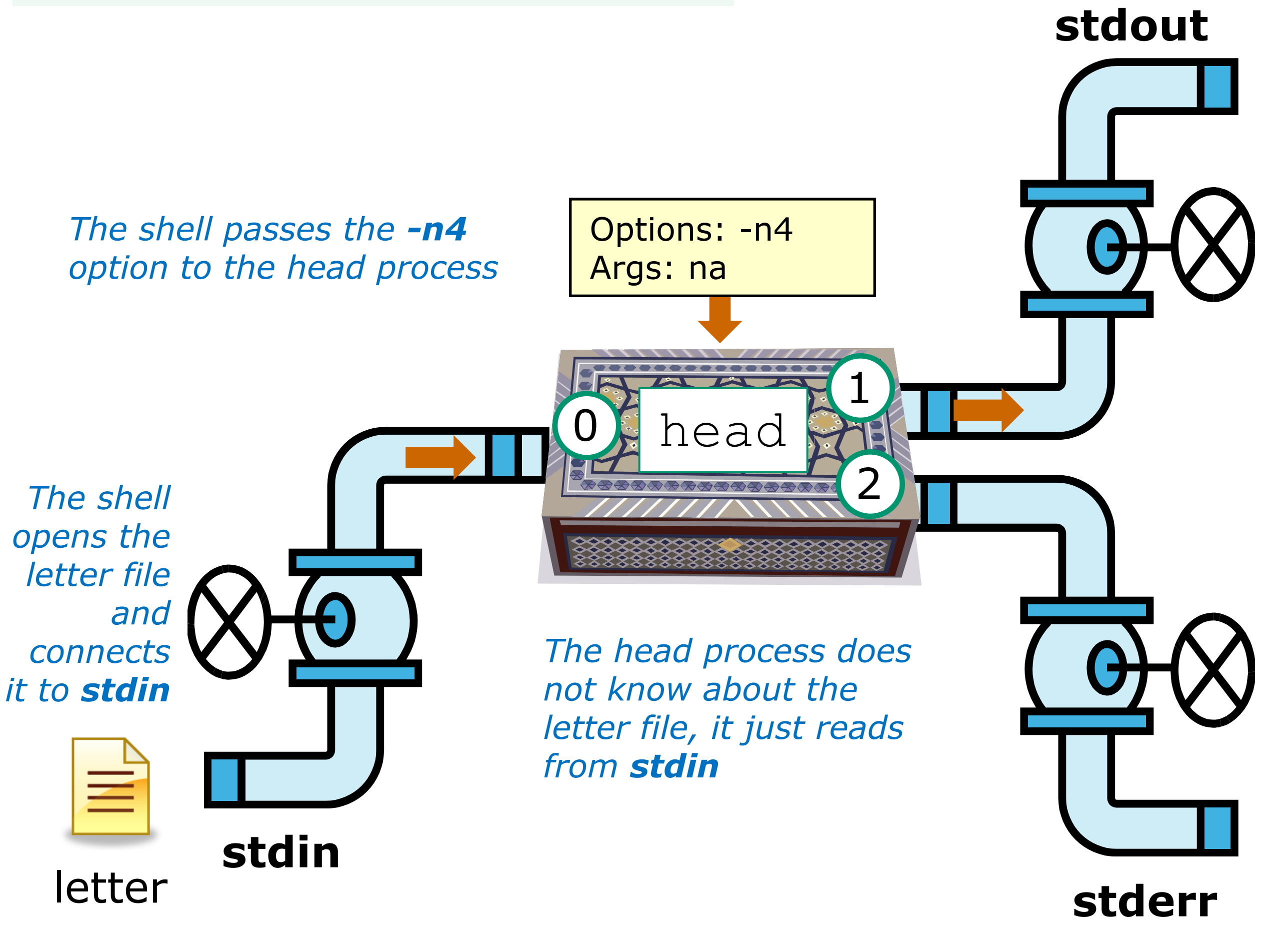
redirection

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

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

*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***



```
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.
```

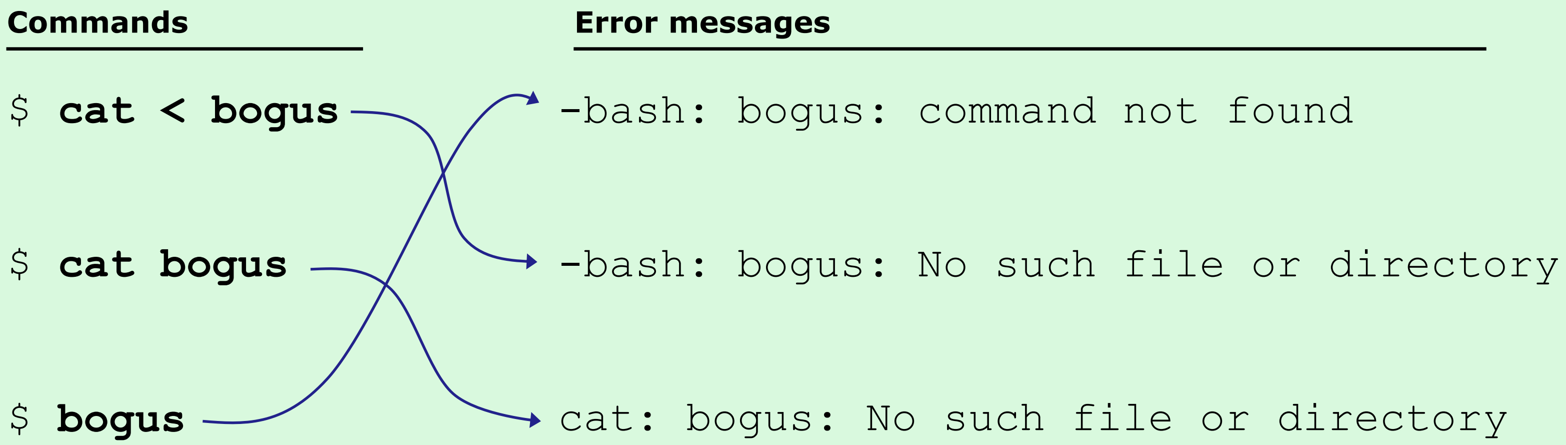
Test your understanding of how the shell and command work as a team

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



Commands

```
$ cat < bogus

$ cat bogus

$ bogus
```

Error messages

```
-bash: bogus: command not found

-bash: bogus: No such file or directory

cat: bogus: No such file or directory
```

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
```

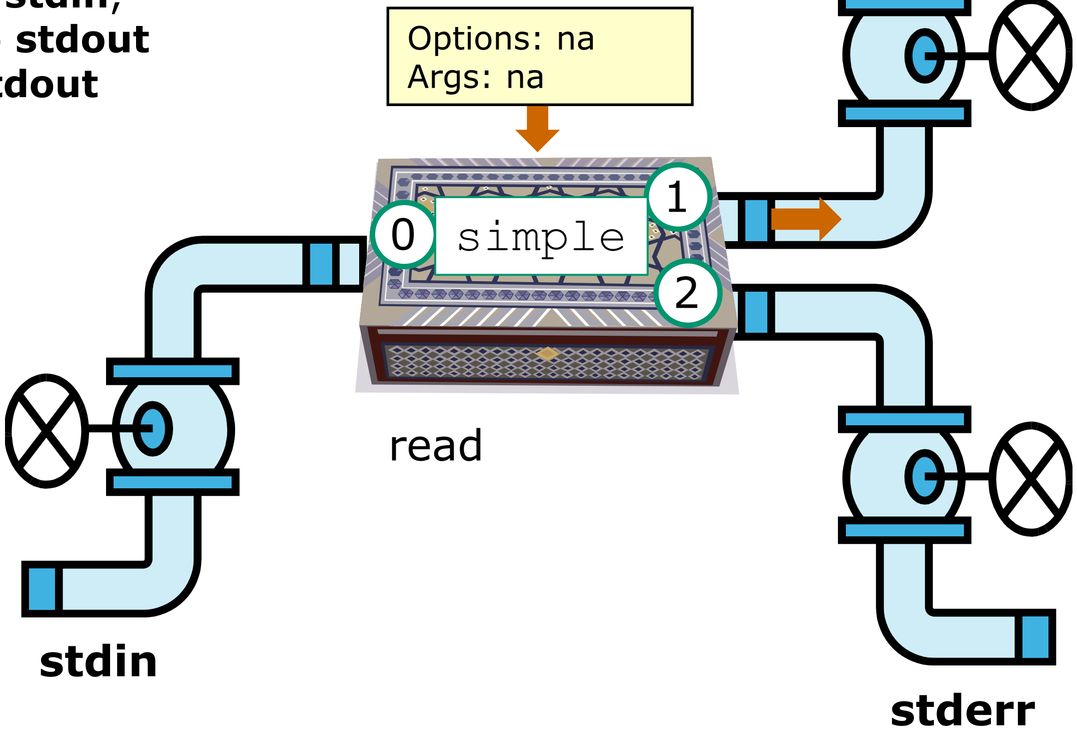
Running the simple program.

*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**



3

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

4

2

```
Rich
```

1

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

FYI only

C Program I/O example

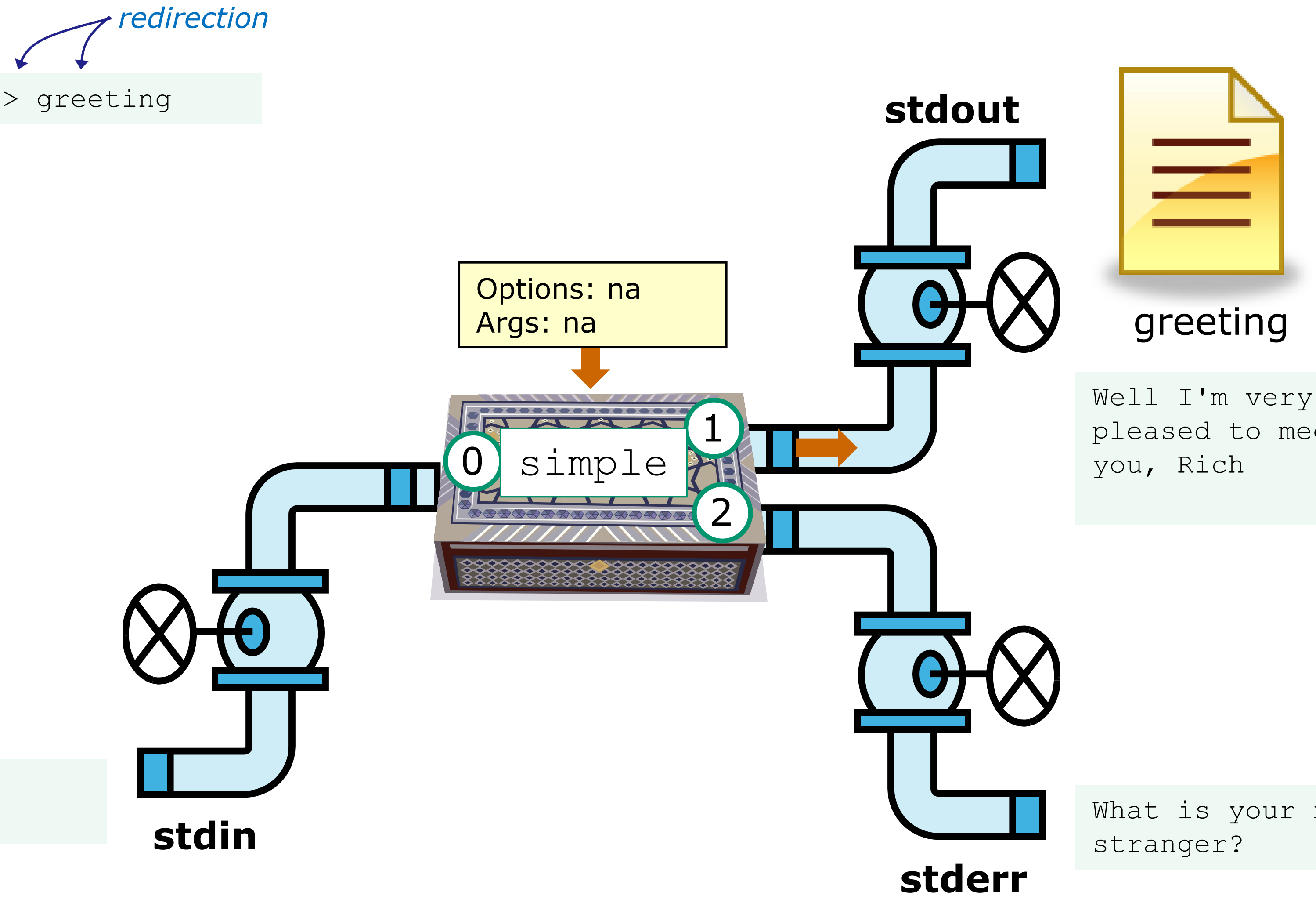
```
[rsimms@opus misc]$ ./simple  >  myfile
What is your name stranger? Rich

[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

```
$ ./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 NOT 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

"Subtraction method"

Current umask setting

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

New file - start with 666

```
 666
-002
 664
```

```
/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

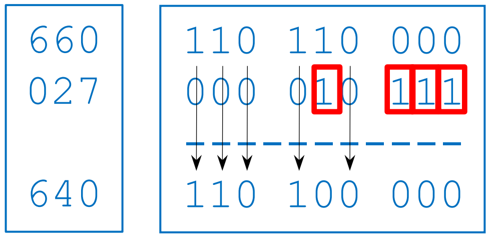
```
 777
-002
 775
```

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

*Shortcut: For new files, when each digit in the **mask** is less than the corresponding digit of the **default permissions** then doing a simple arithmetic subtraction works to determine the new permissions.*

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
```



```
660    110 110 000
027    000 010 111
       -----------
640    110 100 000
```

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.

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 ":"
```

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 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 | 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 $
```

***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

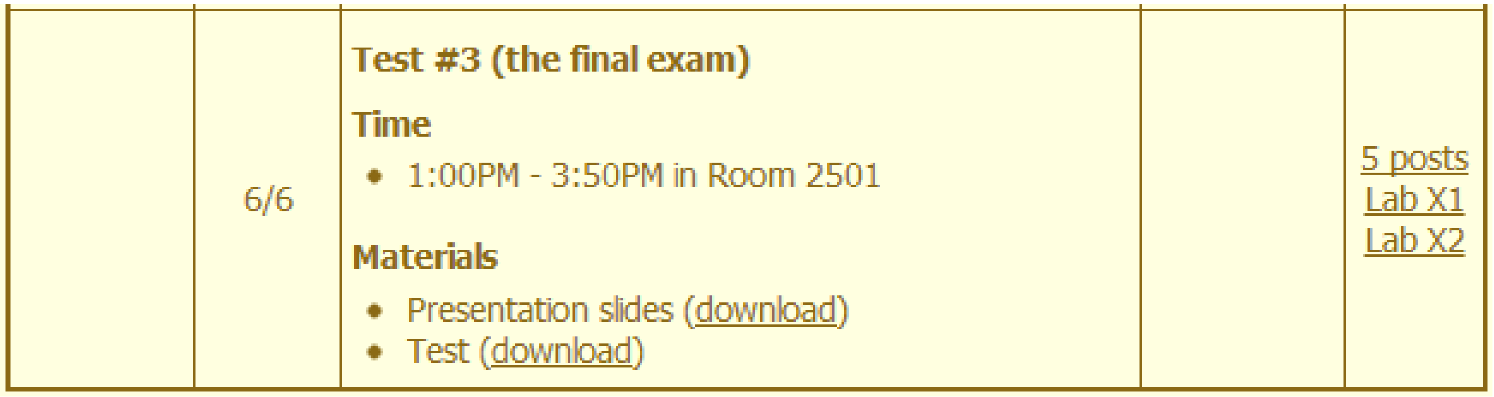
Housekeeping

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 2501 on campus.



	6/6	**Test #3 (the final exam)** **Time** • 1:00PM - 3:50PM in Room 2501 **Materials** • Presentation slides (download) • Test (download)		5 posts Lab X1 Lab X2

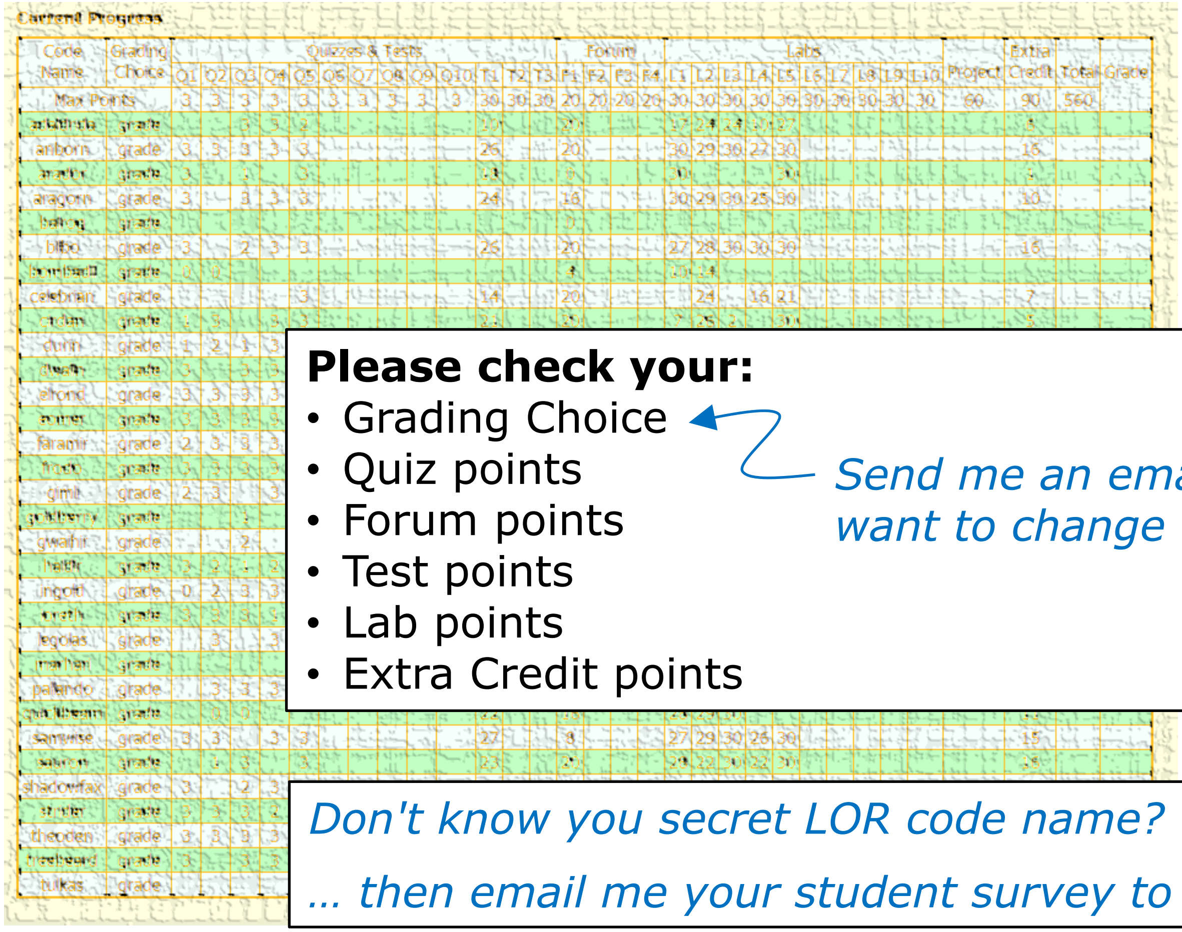
- Note you can earn up to 90 points of extra credit (labs, typos, HowTos, etc.)

- Extra credit labs:
  - Lab X1 (30 points) - see Calendar
  - Lab X2 (30 points) - available now

- HowTos
  - Up to 20 points extra credit for a publishable HowTo document (will be published on the class website)
  - 10 points additional if you do a class presentation
  - Topics must be pre-approved with instructor

Grades Web Page

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



Current Progress

Code Name	Grading Choice	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	T1	T2	T3	F1	F2	F3	F4	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	Project	Extra Credit	Total	Grade
		Quizzes & Tests													Forum				Labs													
Max Points		3	3	3	3	3	3	3	3	3	3	30	30	30	20	20	20	20	30	30	30	30	30	30	30	30	30	30	60	90	560	
adaldrida	grade			3	3	2						10			20				17	24	24	10	27							6		
anborn	grade	3	3	3	3	3						26			20				30	29	30	27	30							16		
arador	grade	3		1		3						18			0				30				30							1		
aragorn	grade	3		3	3	3						24			16				30	29	30	25	30							10		
balrog	grade														0																	
bilbo	grade	3		2	3	3						26			20				27	28	30	30	30							16		
bombadil	grade	0	0												4				10	14												
celebrian	grade					3						14			20					24		16	21							7		
cirdan	grade	1	3		3	3						21			20				7	25	2		30							5		
durin	grade	1	2	1	3																											
dwalin	grade	3		3	3																											
elrond	grade	3	3	3	3																											
eomer	grade	3	3	3	3																											
faramir	grade	2	3	3	3																											
frodo	grade	3	3	3	3																											
gimli	grade	2	3		3																											
goldberry	grade			1																												
gwaihir	grade			2																												
haldir	grade	3	2	1	2																											
ingold	grade	0	2	3	3																											
ioreth	grade	3	3	3	1																											
legolas	grade		3		3																											
marhari	grade																															
pallando	grade		3	3	3																											
quickbeam	grade		0	0																												
samwise	grade	3	3		3	3						27			8				27	29	30	26	30							15		
sauron	grade		1	3		3						23			20				28	22	30	22	30							18		
shadowfax	grade	3		2	3																											
strider	grade	3	3	3	2																											
theoden	grade	3	3	3	3																											
treebeard	grade	3		3	3																											
tulkas	grade																															

Please check your:

- Grading Choice
- Quiz points
- Forum points
- Test points
- Lab points
- Extra Credit points

Send me an email if you want to change this

Don't know you secret LOR code name?

… then email me your student survey to get it!

CIS 90 - Lesson 9

```
[rsimms@oslab bin]$ date
Wed Apr 17 16:19:46 PDT 2013
```

Use Jesse's Python script on Opus:

checkgrades *<code name>*

If you feel you are not where you want to be then contact me to help you make a development plan.

```
adaldrida: 73% (197 of 268 points)
anborn: 102% (276 of 268 points)
arador: 50% (136 of 268 points)
aragorn: 79% (213 of 268 points)
balrog: 0% (0 of 268 points)
bilbo: 100% (268 of 268 points)
bombadil: 10% (28 of 268 points)
celebrian: 57% (154 of 268 points)
cirdan: 52% (141 of 268 points)
durin: 86% (232 of 268 points)
dwalin: 81% (219 of 268 points)
elrond: 105% (284 of 268 points)
eomer: 104% (279 of 268 points)
faramir: 101% (271 of 268 points)
frodo: 100% (270 of 268 points)
gimli: 58% (157 of 268 points)
goldberry: 83% (223 of 268 points)
gwaihir: 69% (185 of 268 points)
haldir: 60% (162 of 268 points)
ingold: 91% (244 of 268 points)
ioreth: 95% (257 of 268 points)
legolas: 103% (278 of 268 points)
marhari: 91% (244 of 268 points)
pallando: 97% (260 of 268 points)
quickbeam: 63% (171 of 268 points)
samwise: 95% (257 of 268 points)
sauron: 94% (253 of 268 points)
shadowfax: 83% (225 of 268 points)
strider: 104% (280 of 268 points)
theoden: 101% (272 of 268 points)
treebeard: 94% (252 of 268 points)
tulkas: 90% (242 of 268 points)
```

Help with labs

Like some help with labs?

I'm in the CIS Lab Monday afternoons

or see me during office hours

or ask the Lab Assistants working in the CIS Lab

- See schedule at http://webhawks.org/~cislab/

or contact me to arrange another time online

Student
Presentation

Wireless Penetration

-Ryan Schell

Things that Hide

Egg Hunt

A number of colored eggs 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. Who is the owner and group for this egg file? The rest are scattered in the various subdirectories you own.

2. Make a new directory named *basket* in your home directory and see how many egg files you can move into it.

3. Put a Green Check in CCC Confer next to your name when you have collected 3 eggs, electronically "clap" if you collect all 17.

Instructor: sudo /home/rsimms/cis90/basket/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

Flashcards

redhat	suse	CentOS	ubuntu
Points:	**Points:**	**Points:**	**Points:**
mckeva90	melale90	shepau90	braril90
davmic90	lovben90	berric90	bunsol90
lemrya90	deddil90	valjus90	diapam90
vashil90	perste90	lejmic90	wiltyr90
paljay90	wismar90	cruben90	halluc90
dusaar90	mennat90	rutsam90	gilgab90
blodan90	schrya90	fareli90	mazari90
marand90	goljor90		marrog90
joylia90	bengre90		

Flashcards
L6=20
L7=15
L8=16

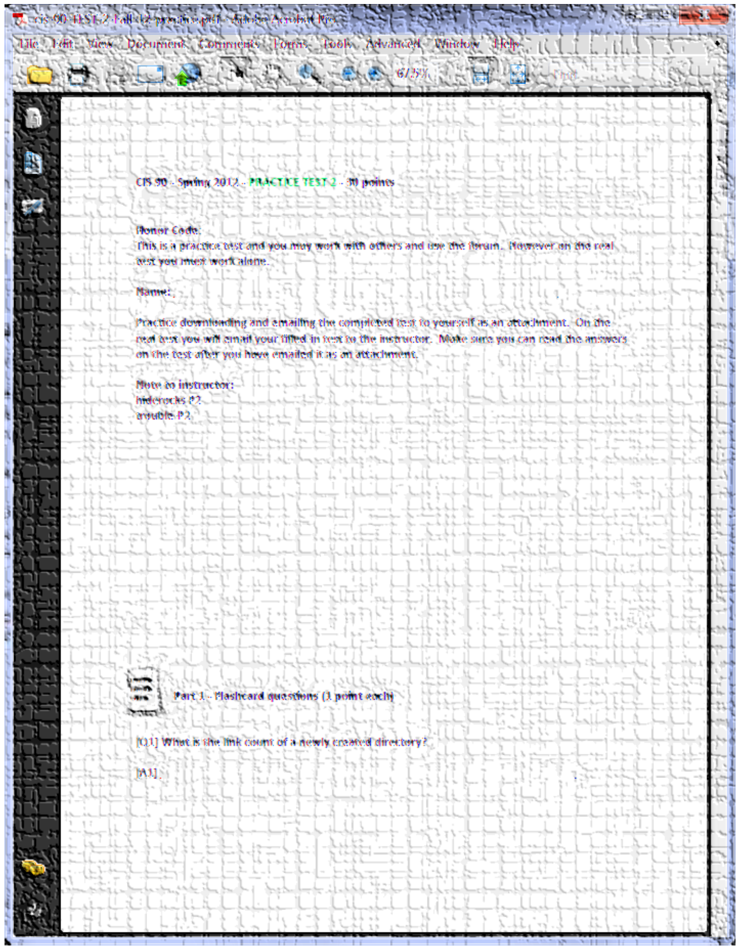
Rules
- Chat window belongs to team that is up
- Team gets the point if anyone on the team writes a correct answer in the chat window in 20 seconds

Make Teams:
```
who | cut -f1 -d" " | uniq | sort -R | tee players | wc -l
```

Instructor timer:
```
i=10; while [ $i -gt 0 ]; do clear; banner $i; let i=i-1; sleep 1;  done; clear; banner done
```

Practice Test

CIS 90 - Lesson 9


cis 90 TEST 2 Fall 12 practice.pdf - Adobe Acrobat Pro
File Edit View Document Comments Forms Tools Advanced Window Help
67.5%

CIS 90 - Spring 2012 - 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:

Practice downloading and emailing the completed test to yourself as an attachment. On the real test you will email your filled in test to the instructor. Make sure you can read the answers on the test after you have emailed it as an attachment.

Note to instructor:
hiderocks P2
trouble P2

Part 1 - Flashcard questions (1 point each)

[Q1] What is the link count of a newly created directory?

[A1]

Practice test available

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

Wrap up

Next Class

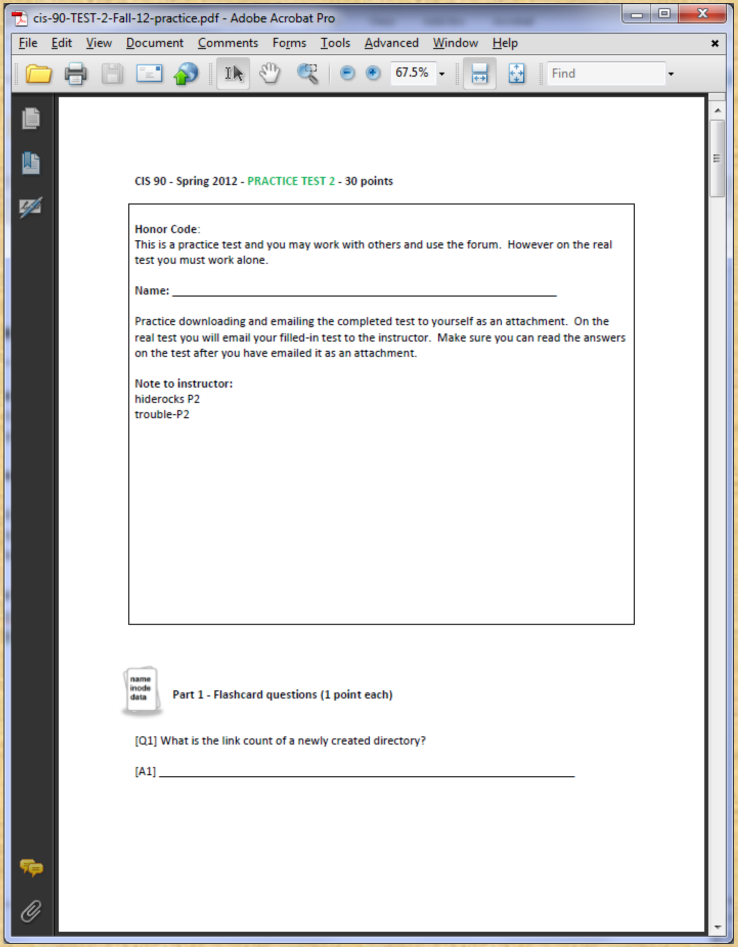
No Quiz

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

CIS 90 - Lesson 9


cis-90-TEST-2-Fall-12-practice.pdf - Adobe Acrobat Pro
File Edit View Document Comments Forms Tools Advanced Window Help
67.5%
Find

CIS 90 - Spring 2012 - 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: ______________________________

Practice downloading and emailing the completed test to yourself as an attachment. On the real test you will email your filled-in test to the instructor. Make sure you can read the answers on the test after you have emailed it as an attachment.

Note to instructor:
hiderocks P2
trouble-P2

Part 1 - Flashcard questions (1 point each)

[Q1] What is the link count of a newly created directory?

[A1] ______________________________

Work the practice test

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

Backup