

## Lesson Module Checklist

- Slides
- WB
  
- Flash cards
- Page numbers
- 1<sup>st</sup> minute quiz
- Web Calendar summary
- Web book pages
- Commands
  
- Opus - hide script tested
- Practice test uploaded
- Sun-Hwa - trouble made and test script run
  
- 9V backup battery for microphone
- Backup slides, CCC info, handouts on flash drive



## Student checklist

- 1) Browse to the CIS 90 website Calendar page
  - <http://simms-teach.com>
  - Click CIS 90 link on left panel
  - Click Calendar link near top of content area
  - Locate today's lesson on the Calendar
  
- 2) Download the presentation slides for today's lesson for easier viewing
  
- 3) Click Enter virtual classroom to join CCC Confer session
  
- 4) Connect to Opus using Putty or ssh command

## 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/>)



Instructor: **Rich Simms**

Dial-in: **888-450-4821**

Passcode: **761867**



Buzz



Carlos



Elijah



Emily



Enrique C.



Enrique R.



Jon M.



Jon W.



Jordan



Joseph



JJ



Kiernan



Maria



Mathew



Mike C.



Michael F.



Mike M.



Nick L.



Patrick



Rebecca



Ricardo



Robert



Ruth



Steve



Tess



Tim



Troy

## Quiz

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

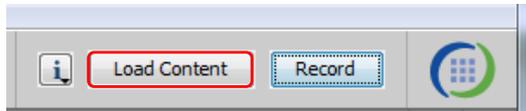
**See electronic white board**

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

**(answers must be emailed within the first few minutes of class for credit)** 5

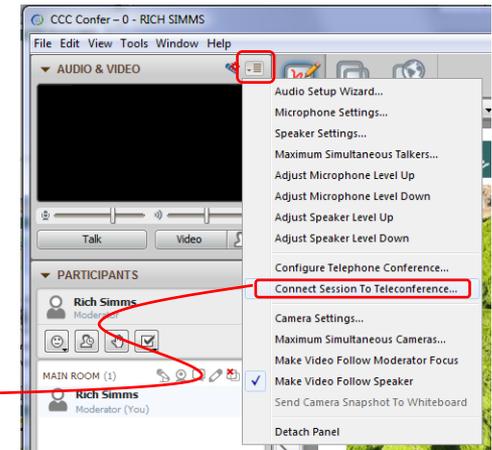
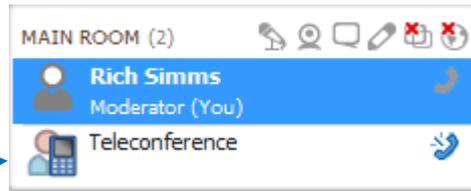


# [ ] Preload White Board with *cis\*lesson??\*-WB*



# [ ] Connect session to Teleconference

*Session now connected to teleconference*



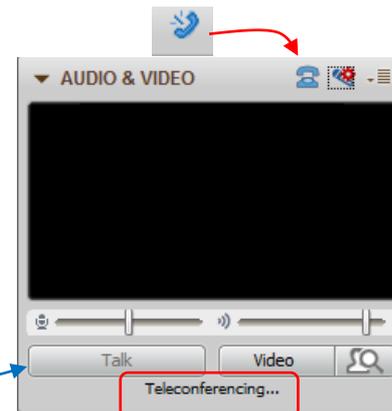
# [ ] Is recording on?



*Red dot means recording*

# [ ] Use teleconferencing, not mic

*Should be greyed out*



*Keep wireless mic transmitter away from cell phone and podium if excess static occurs*



## [ ] layout and share apps

The screenshot displays a Windows desktop with a multi-pane layout. The desktop background is a light blue sky with birds. The taskbar at the bottom shows icons for Recycle Bin, CCC Confer, Chrome, Firefox, Internet Explorer, and several other applications. The system tray in the bottom right corner shows the date and time as 6:52 AM on 10/10/2012.

Key elements in the screenshot include:

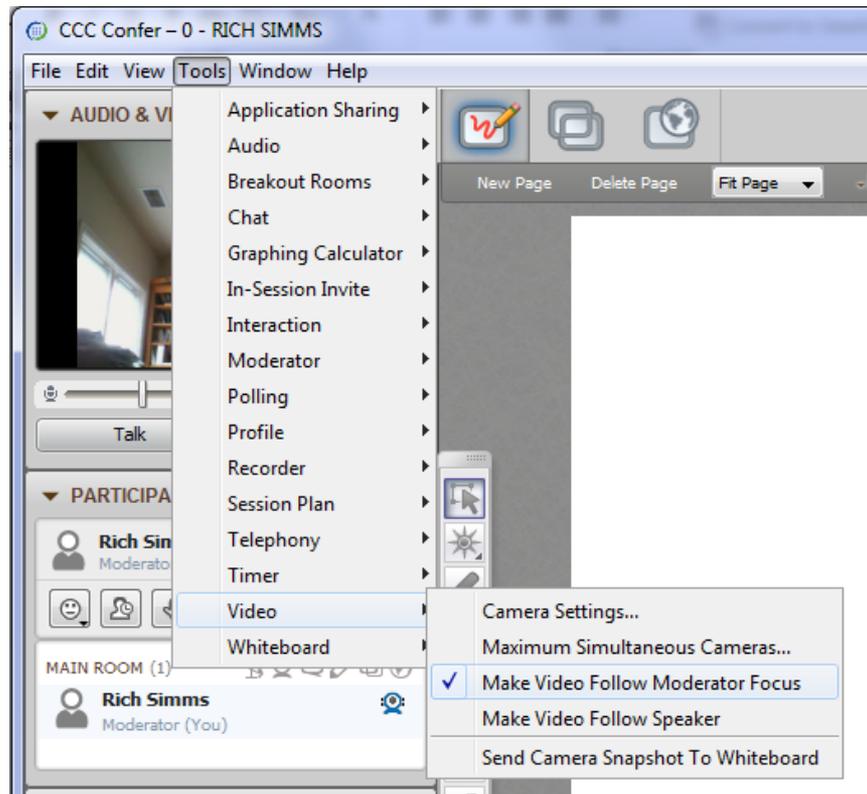
- CCC Confer - 0 - RIC...:** A window on the left side of the screen showing a video feed of Rich Simms and a list of participants.
- cis90lesson07.pdf \* - Foxit Reader:** A window in the center showing a PDF document with a file explorer overlay. Red callout boxes labeled "foxit for slides" and "chrome" point to the PDF viewer and the browser respectively.
- simms-teach.com/docs/cis90/cis-90-TEST-1-Fall-12.pdf:** A browser window on the right showing a test page with questions. A red callout box labeled "chrome" points to the browser.
- Terminal:** A window in the foreground showing a terminal session with the following text:

```
simben90@oslab:~$  
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  
  
simben90@oslab:~$  
Current directory  
source  
destination  
  
What command copies th  
  
Terminal type? [xterm]  
Terminal type is xterm.  
/home/cis90/simben $
```
- vSphere Client:** A window in the bottom right corner showing the vSphere Client interface with a tree view of virtual machines and a table of recent tasks.

Red callout boxes with white text are used to highlight specific applications: "foxit for slides" points to the Foxit Reader window, "chrome" points to the Chrome browser window, and "vSphere Client" points to the vSphere Client window.



- [ ] Video (webcam) optional
- [ ] Follow moderator
- [ ] Double-click on postage stamps



## Universal Fix for CCC Confer:

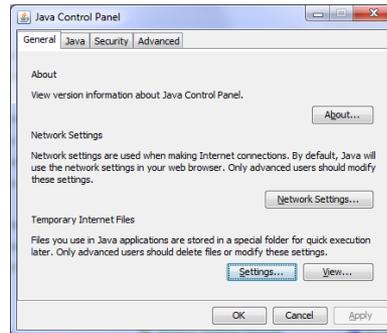
- 1) Shrink (500 MB) and delete Java cache
- 2) Uninstall and reinstall latest Java runtime



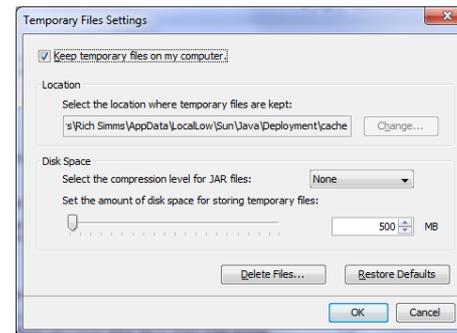
Control Panel (small icons)



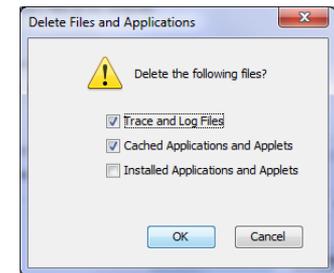
General Tab > Settings...



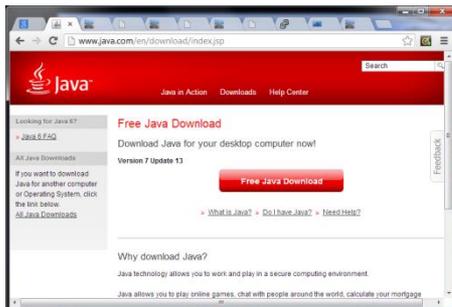
500MB cache size



Delete these



## Google Java download



## Review

### Objectives

- Get ready for the next test
- Practice skills
- Introduction to processes

### Agenda

- 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? Tests?

How this course works?

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

*Who questions much, shall learn much, and retain much.*

- Francis Bacon

*If you don't ask, you don't get.*

- Mahatma Gandhi

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.*



# Lab 6

# Post Mortem



Step 01 -

Step 02 -

Step 03 -

Step 04 -

Step 05 - x

Step 06 -

Step 07 -

Step 08 -

Step 09 - xxxxxxxx Set the permissions of your poems directory and ...

Step 10 - xxxxxx Set all ordinary files under the poems ...

Step 11 - xxxx Change the permissions of your bin ...

Step 12 - xxx

Step 13 - x

Step 14 - xxx

Step 15 - xxxxx Make all ordinary files under class/labs and ...

Step 16 - xx

Step 17 - x

Step 18 -

Step 19 - x

Step 20 -

Step 21 - x

Step 22 -

Step 23 - xxxxx Try setting the umask to 777 ...

# Lab 6 Results

(steps where points were taken off)

*For more on Steps 9-10 see the Backup Slides in Lesson 8 (module titled Lab 6 Tips)*

## CIS Lab Schedule

<http://webhawks.org/~cislab/>

The screenshot shows a web browser window with the URL [webhawks.org/~cislab/](http://webhawks.org/~cislab/). The page features a header for "CIS Lab & Datacenter" at the "Aptos Campus" with navigation links for Home, Resources, NETLAB, and Location. Below this is an "Announcements" section with a message about moving to Building 800. The main content is a "CIS Lab Spring 2014" calendar for the week of Feb 16 - 22, 2014. The calendar shows various lab sessions with times and names of staff members.

Day	Time	Staff
Mon 2/17	9:00 - 1:30p	Mike Mate
Mon 2/17	9:30 - 9:30	Geoff Jayd, Noni on CS
Mon 2/17	2:30p - 4p	Charles CS
Tue 2/18	9:30 - 10:30	Charles CS
Tue 2/18	11:10	Jeffrey Bergmani
Tue 2/18	12:45p	Mon Leandro, gue RO
Tue 2/18	2:45p - 3:15p	Mike Jay, Mat de n CS
Tue 2/18	5:30p - 9p	Brett CS
Wed 2/19	9:30 - 11	Joyden CS
Wed 2/19	2p - 3p	Leg 2:15p - 3p, Mon Char, gue es CS
Wed 2/19	3:30p - 3:30	Rich Mik, Sim s, Mat ero
Thu 2/20	9:30 - 10:30	Charles CS
Thu 2/20	1p - 5p	Leandro Rocha
Thu 2/20	2:05p - 2:15p	Mike Matera n CS
Thu 2/20	3:30p - 6p	Rich Simms
Fri 2/21	11 - 2p	Charles CS

Not submitting tests or lab work?

Would like some help?

Come to the CIS Lab to work with classmates, lab assistants and instructors on Lab assignments.

*Rich is in the lab Wednesdays and Fridays from 3:30 - 6:00 PM*

# Free CIS 90 Tutoring Available

<http://www.cabrillo.edu/services/tutorials/>

The screenshot shows the Cabrillo College website's Tutorials Center page. The page is titled "TUTORIALS" and includes a navigation menu with options like "ABOUT", "ACADEMICS/CAREERS", "ADMISSIONS", "CLASS SCHEDULES", "REGISTRATION", and "WEBADVISOR". The main content area is divided into several sections:

- ANNOUNCEMENTS & DEADLINES:** Lists new subjects for Spring 2014, including American Sign Language, Computer Applications/Business Technology (CABT), Computer and Information Systems (CIS), and History 17A.
- Welcome to the Tutorials Center!:** States that free peer tutoring is offered to students. It lists several conditions: tutoring is by appointment, sessions are weekly, sessions are in small groups, and students should come directly to the TC office to schedule.
- The following classes are being tutored for Spring 2014:** Lists various classes, with "Computer and Information Systems (CIS) 81, 90, 172" highlighted in a red box.
- CONTACT INFORMATION:** Provides details for the Tutorials Center, including location (Room 1080A), phone number (831.479.6470), email (tutorialscenter@cabrillo.edu), and coordinator (Lori Chavez).



Matt Smithey

All students interested in tutoring in CIS 90, 172, and 81 classes need to come directly to the Tutorials Center to schedule, register and fill out some paperwork. This is just a one-time visit.

The tutoring will take place at the STEM center and they will log in and log out on a computer you have designated (I will figure out exactly what that means).

Matt is available M: 9:00-5:00, T: 9-11 and 2-5, Wed: 9-12 and Th: 9-11 and 3-5.



# Housekeeping



## Housekeeping

1. Lab 7 due 11:59PM tonight (**don't forget to submit it**)
2. A **check7** script is available
3. **Test #2 is in two weeks** (next week is Spring Break)
4. Practice Test #2 available now
5. No lab assigned this week (so you can work on the practice test)
6. Opus will be **unavailable** Sunday evening and all day Monday (3/30-3/31)

## Final Exam

### Test #3 (final exam)

- Must be face-to-face or proctored (not online using CCC Confer).
- Room 828 on campus.

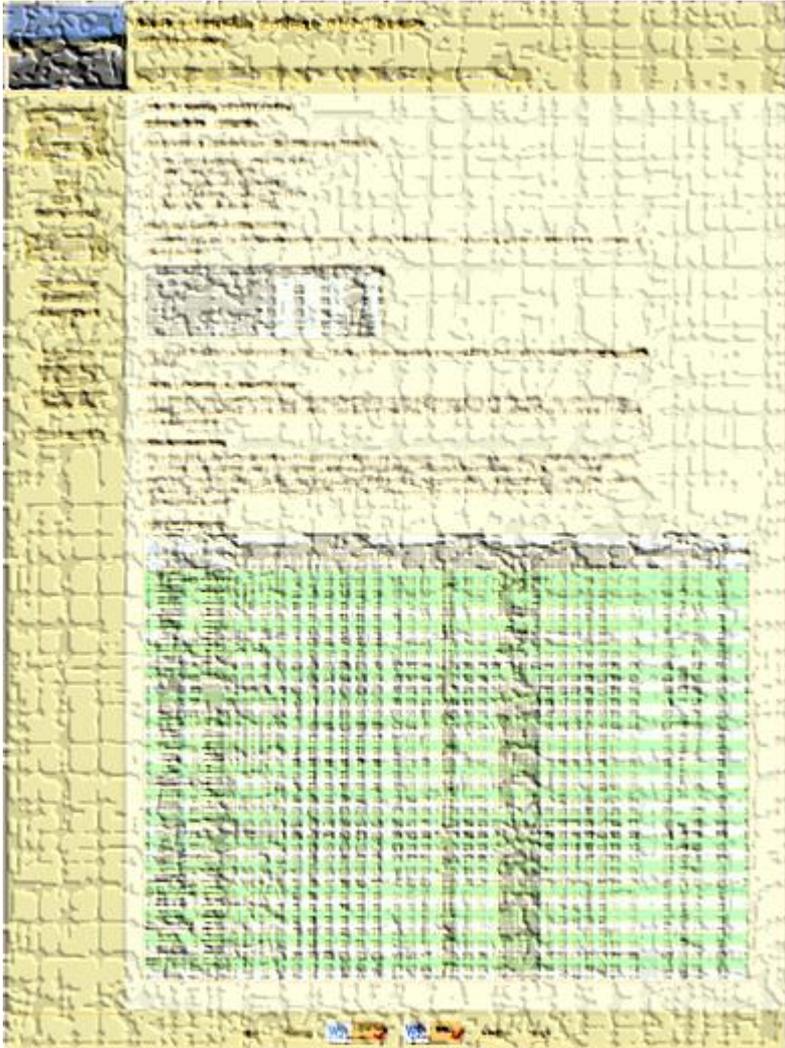
	5/21	<p><b>Test #3 (the final exam)</b></p> <p><b>Time</b></p> <ul style="list-style-type: none"> <li>• 7:00AM - 9:50AM in Room 828</li> </ul> <p><b>Materials</b></p> <ul style="list-style-type: none"> <li>• Test (<a href="#">download</a>)</li> </ul>		<p><a href="#">5 posts</a></p> <p><a href="#">Lab X1</a></p> <p><a href="#">Lab X2</a></p>
--	------	---	--	--

- If you are a long distance student, contact the instructor for options.

<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 3/25/2014

<b>Points that could have been earned:</b>	
6 quizzes:	18 points
6 labs:	180 points
1 test:	30 points
2 forum quarters:	40 points
<b>Total:</b>	<b>268 points</b>

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

alatar: 64% (172 of 268 points)  
 anborn: 78% (210 of 268 points)  
 aragorn: 88% (237 of 268 points)  
 arwen: 99% (266 of 268 points)  
 beregond: 0% (0 of 268 points)  
 bilbo: 54% (145 of 268 points)  
 celebrian: 98% (265 of 268 points)  
 dwalin: 97% (260 of 268 points)  
 eomer: 91% (245 of 268 points)  
 faramir: 94% (253 of 268 points)  
 frodo: 95% (256 of 268 points)  
 gwaihir: 106% (285 of 268 points)  
 ioreth: 96% (259 of 268 points)  
 legolas: 91% (244 of 268 points)

marhari: 65% (176 of 268 points)  
 orome: 83% (225 of 268 points)  
 pallando: 0% (0 of 268 points)  
 pippen: 79% (212 of 268 points)  
 quickbeam: 99% (267 of 268 points)  
 rian: 0% (0 of 268 points)  
 samwise: 85% (230 of 268 points)  
 shadowfax: 0% (0 of 268 points)  
 strider: 86% (232 of 268 points)  
 theoden: 58% (157 of 268 points)  
 treebeard: 107% (287 of 268 points)  
 tulkas: 86% (231 of 268 points)  
 ulmo: 80% (216 of 268 points)

## 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.*



# Linux at School

# Our Opus server on campus

vmserver2



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

VMs on vmserver2

The screenshot shows the vSphere Client interface for a VM named 'opus'. The left sidebar lists various VMs under the host 'vmserver2', including 'ds1', 'ds2', 'Hershey', 'Jeff', 'ns1', 'ns2', 'ns-slinky', 'opus', 'Sun-Hwa', 'Sun-Hwa-II', 'UCSClient', 'UCSPE1', 'UCSPE2', 'UCSPE3', and 'vCenter2y'. The main window displays the configuration for 'opus' with the following details:

General		Resources		
Guest OS:	CentOS 4/5/6 (32-bit)	Consumed Host CPU:	8 MHz	
VM Version:	7	Consumed Host Memory:	792.00 MB	
CPU:	1 vCPU	Active Guest Memory:	10.00 MB	
Memory:	1024 MB	Refresh Storage Usage		
Memory Overhead:	61.21 MB	Provisioned Storage:	25.11 GB	
VMware Tools:	Running (Current)	Not-shared Storage:	21.93 GB	
IP Addresses:	172.30.5.20	Used Storage:	21.93 GB	
DNS Name:	oslab.cishawks.net	Storage	Drive Type	Capacity
State:	Powered On	disk2-1	Non-SSD	456.50 GB
Host:	vmserver2.cslab.net	Network	Type	
Active Tasks:		Server Network	Standard port group	

At the bottom of the interface, there is a 'Recent Tasks' table with columns for Name, Target, Status, Details, and Initiated by. The license period is shown as 63 days remaining.

SSH access to Opus

hostname: oslab.cishawks.net (port 2220)



# Your own Linux Systems

# USB "Live" Linux Boot USB Drive



*Allows you to use or try out Linux on an existing computer without installing it*

## USB "Live" Linux Boot USB Drive

*Allows you to use or try out Linux on an existing computer without installing it*

1)



Get the Linux distro of your choice  
See: <http://iso.linuxquestions.org/>

2)



Get a USB flash drive

Google "boot live linux from usb" for instructions  
*or see*

3)

<http://www.howtogeek.com/howto/14912/create-a-persistent-bootable-ubuntu-usb-flash-drive/>

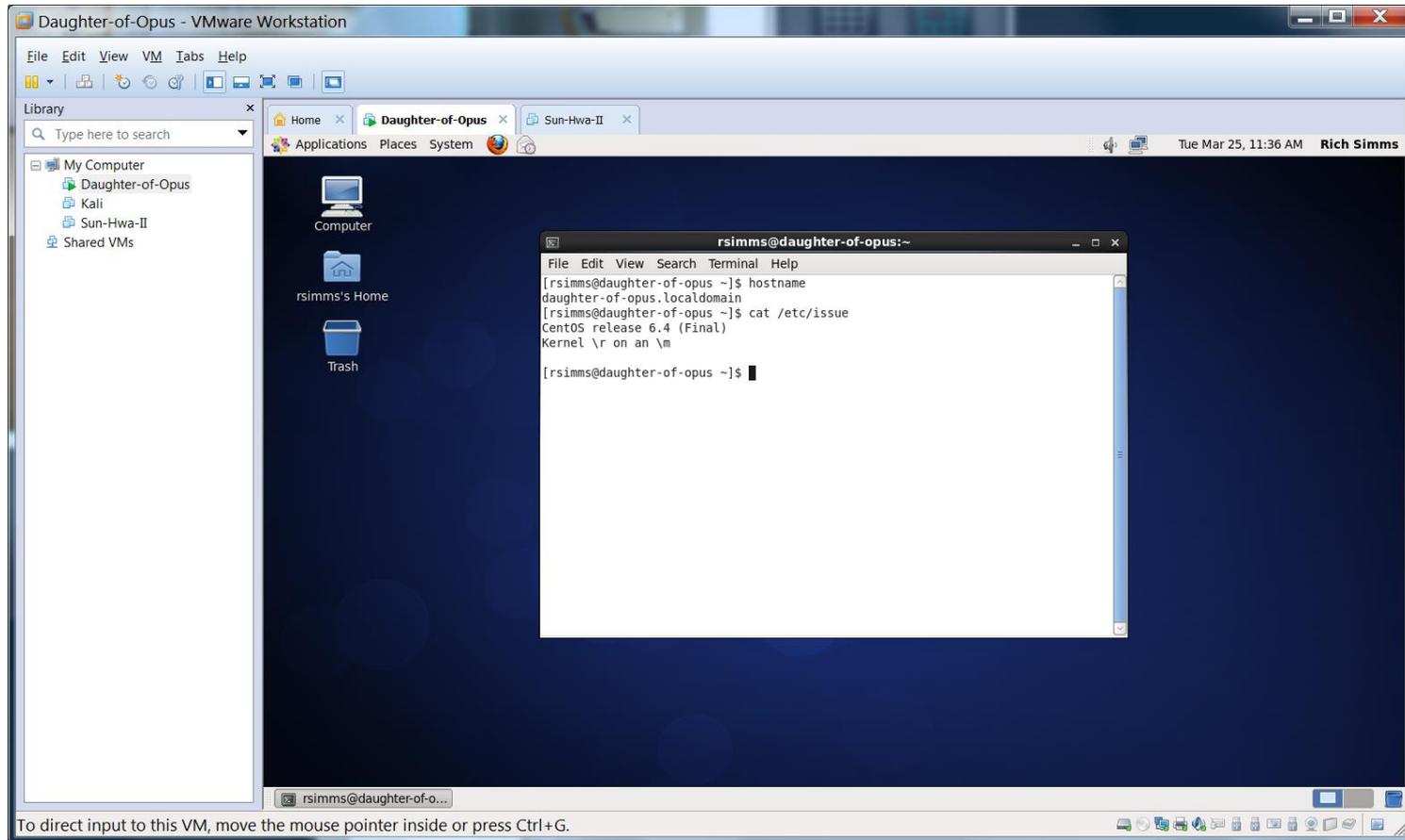
4)



Running native Kali Linux on my Windows laptop  
(BIOS configured to boot from USB if present)

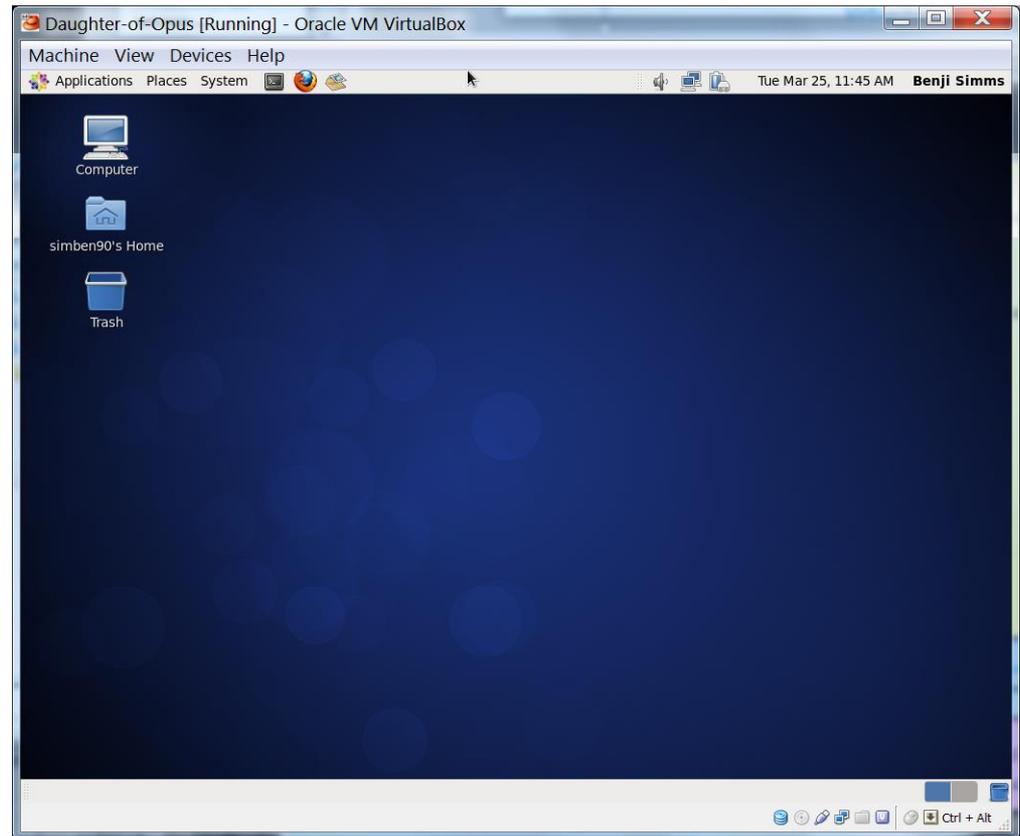
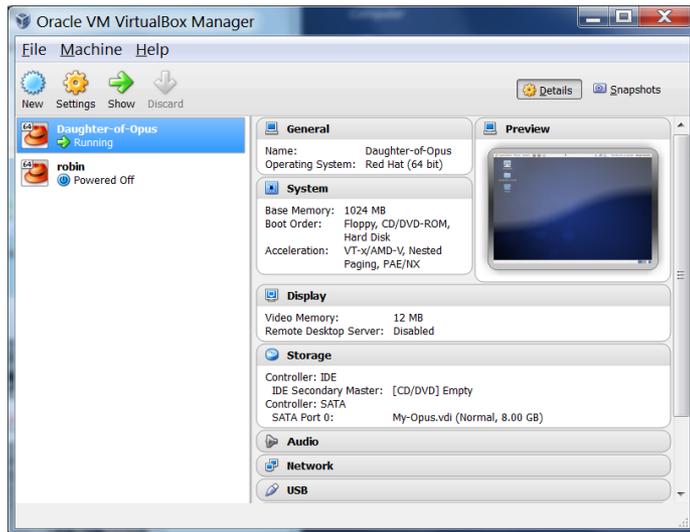
- 1) Power On with USB stick ==> Kali Linux
- 2) Power On without USB stick ==> Windows

# One Daughter-of-Opus VMware Workstation



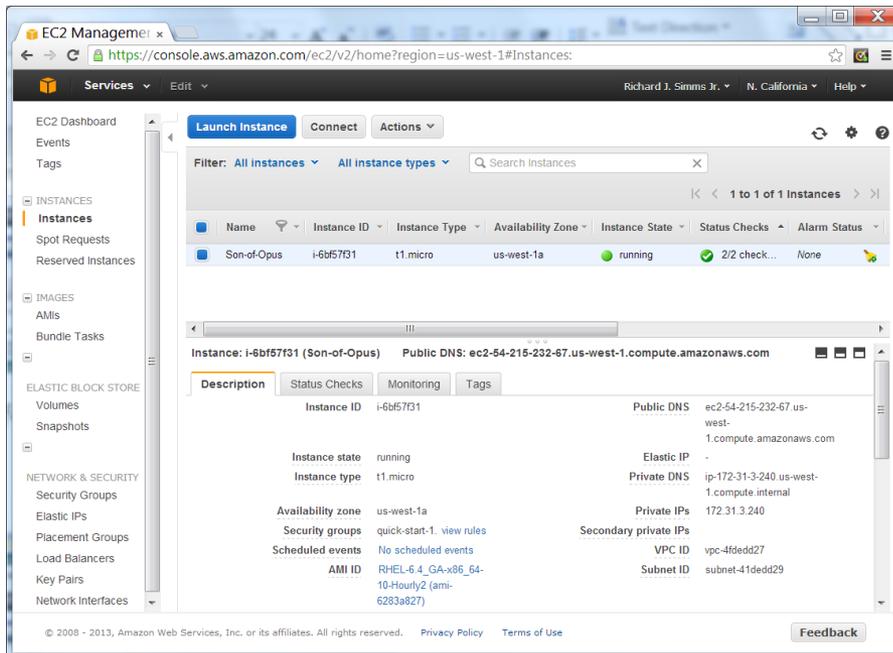
*One Daughter-of-Opus is a VM running on my laptop using  
VMware Workstation (expires in one year)*

# Another Daughter-of-Opus Oracle VirtualBox



*This Daughter-of-Opus is a VM running on my laptop using Oracle VirtualBox (never expires)*

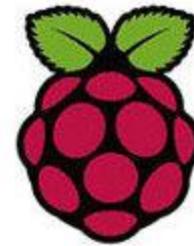
# Son-of-Opus Amazon Web Services



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

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

# Baby-Opus Debian 7 Linux Server



**Raspberry Pi**

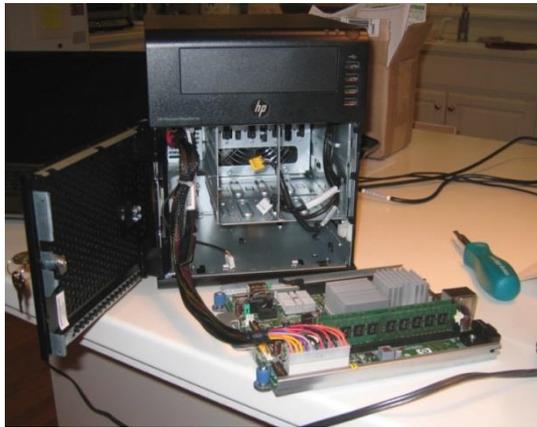
*Baby-Opus is a VM running  
on my Raspberry Pi*

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

```
NoPar#show ip dhcp binding  
MAC b8:27:eb:b7:b3:99  
Reservation for 172.30.1.31
```

## My Home VLab

HP Microserver



VMware ESXi for virtualization



Inexpensive "bare bones" servers are available that come without hard drives or an operating system

<http://simms-teach.com/resources.php>



Microsoft  
Software  
(Academic)

VMware  
Software  
(Academic)



## Rich's Cabrillo College CIS Classes Resources

Home
Resources
Forums
CIS Lab
Blackboard

**Login**

**Flashcards**

**Admin**

CIS 90  
Previous Classes

**60 days - term ends!**

Cabrillo College  
[Web Advisor](#)  
Commands and Files

[VLab RDP file](#)

[CIS 90 VLab VM Assignments](#)

[RIP Dennis Ritchie](#)

Opus Status: UP

### Links

<p><b>Instructors</b></p> <ul style="list-style-type: none"> <li>Linux Master Jim</li> <li>Programming Master Ed</li> <li>Network Master Gerinde</li> <li>Network Master Rick</li> <li>Web Master John</li> <li>Systems Master Michael</li> </ul> <p><b>Clubs</b></p> <ul style="list-style-type: none"> <li>Computer Club</li> <li>Robotics Club</li> </ul> <p><b>Departments</b></p> <ul style="list-style-type: none"> <li>CNSA</li> <li>CIS</li> <li>CS</li> </ul> <p><b>Crib Sheets</b></p> <ul style="list-style-type: none"> <li>Ole Wright (CIS 90)</li> </ul> <p><b>Documentation</b></p> <ul style="list-style-type: none"> <li>TLDP</li> <li>LINFO</li> <li>UNIX Rosetta Stone</li> </ul> <p><b>Animations</b></p> <ul style="list-style-type: none"> <li>Linux network technologies</li> </ul>	<p><b>Getting Linux/UNIX</b></p> <ul style="list-style-type: none"> <li>Linux ISOs</li> <li>Kernels</li> <li>RPMS (rpmfind)</li> <li>RPMS (pbone)</li> <li>OpenSolaris</li> </ul> <p><b>Tools and Software</b></p> <ul style="list-style-type: none"> <li>Apache</li> <li>Bastille</li> <li>CoRD</li> <li>cygwin</li> <li>DOS boot disks</li> <li>Dynamics/Dynagen</li> <li>John the Ripper</li> <li>Netfilter</li> <li>Putty SSH Tools</li> <li>Quagga routing suite</li> <li>Tripwire</li> <li>Wireshark</li> </ul> <p><b>Academic Software for CIS Students</b></p> <ul style="list-style-type: none"> <li>Microsoft Webstore</li> <li>VMware Webstore</li> </ul> <p><b>Virtualization</b></p> <ul style="list-style-type: none"> <li>VirtualBox</li> <li>VMware ESXi and vSphere client</li> </ul> <p><b>Standards</b></p> <ul style="list-style-type: none"> <li>IEEE</li> <li>IETF (RFCs)</li> </ul> <p><b>Training and Tutorials</b></p> <ul style="list-style-type: none"> <li>Free Linux Tutorials</li> <li>The Linux Foundation</li> <li>Linux Survival</li> <li>Learn about Linux</li> <li>The Linux Tutorial</li> </ul>	<p><b>Commands</b></p> <ul style="list-style-type: none"> <li>Practical</li> <li>Command Directory</li> <li>Useful</li> <li>vi summary</li> <li>vi cheat sheet</li> </ul> <p><b>Howtos</b></p> <ul style="list-style-type: none"> <li>HowtoForge</li> <li>email</li> <li>DNS</li> <li>Ethernet (NIC drivers)</li> <li>NES</li> <li>NIS</li> <li>PPP</li> <li>Putty SSH Keys</li> <li>Using sed</li> </ul> <p><b>Student Howtos</b></p> <ul style="list-style-type: none"> <li>Monitor Script by Sean Calahan</li> <li>WiFi Penetration by Ryan Scher</li> <li>Logging into Opus from a Mac by Laura Sreckovic</li> <li>LDAP Implementation by Tim Childers</li> <li>Install and DualBoot into Microsoft Windows 7 and Linux Ubuntu by Richie Fou</li> <li>Making an ethernet cable by Michael George</li> <li>Home VM access via Linksys router by Marc Romansky</li> <li>Putty to VMs by Marc Romansky</li> <li>Installing VirtualBox by Marcos Valdebenito</li> <li>Linux Permissions by Michael Wicherski</li> <li>Guide to /bin/mail by Michael Wicherski</li> </ul>
--	---	--

**Linux News**

Linux Distros (ISOs)



VirtualBox

# More on I/O

(input/output)

# Input and Output

## File Redirection

The 3 standard UNIX file descriptors:

Name	Integer Value
<b>stdin</b> ( <b>st</b> andard <b>in</b> )	0
<b>stdout</b> ( <b>st</b> andard <b>out</b> )	1
<b>stderr</b> ( <b>st</b> andard <b>error</b> )	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



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

## Examples

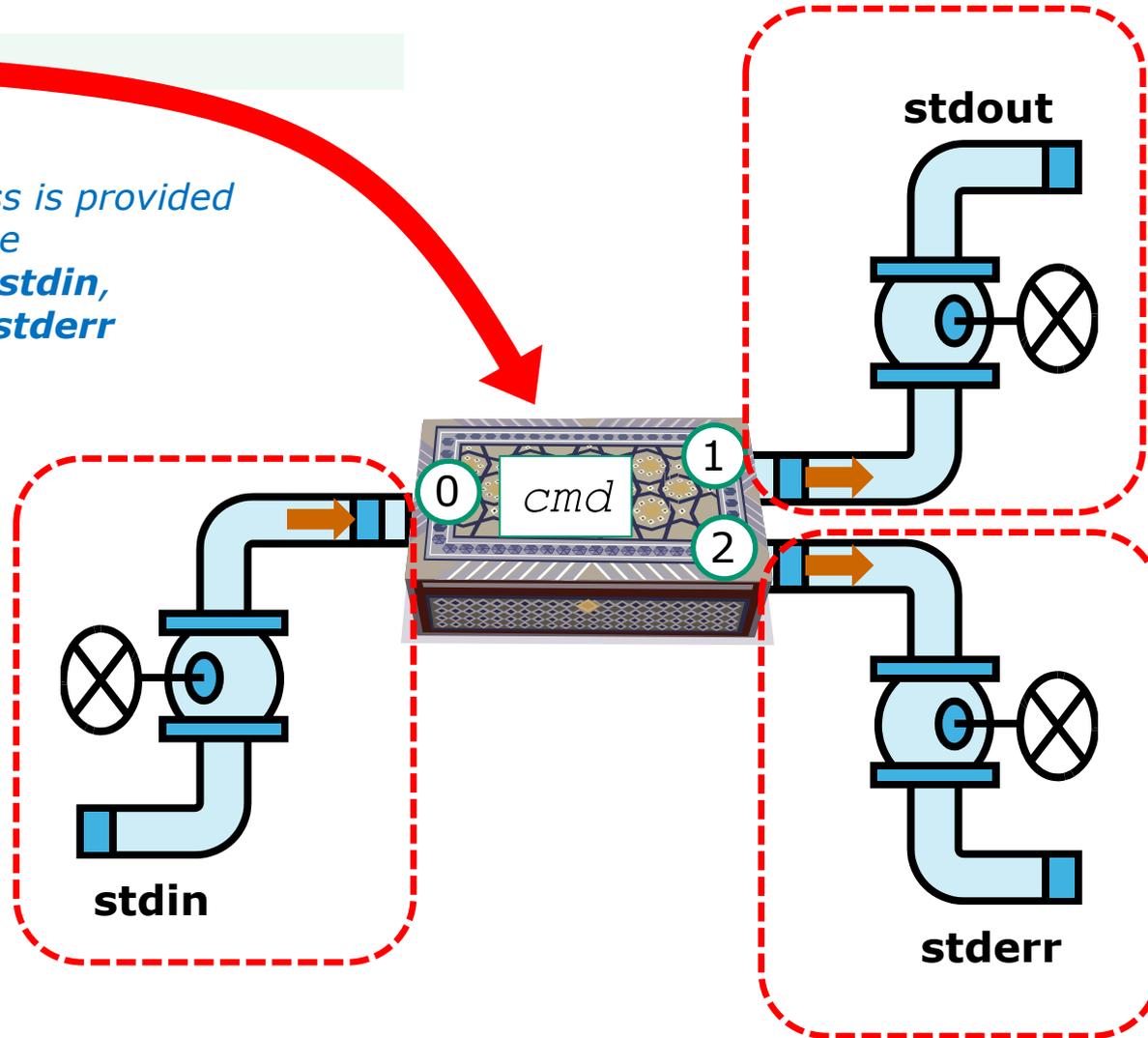
```

/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**

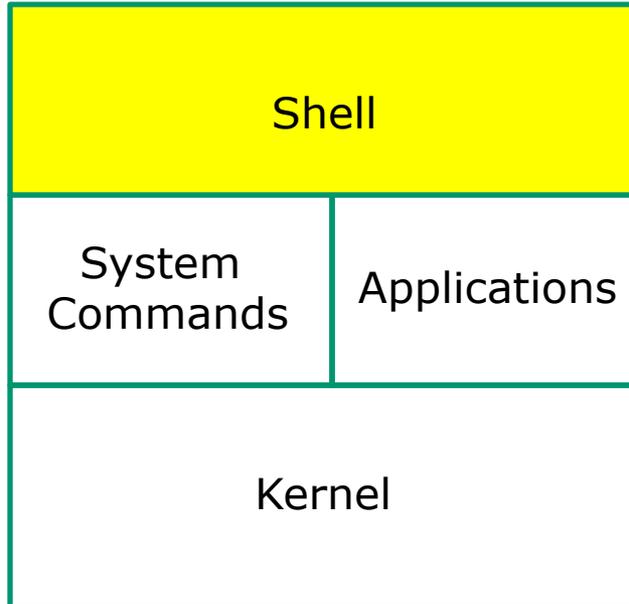




# All Together Now Example



# 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> character 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

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

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

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

The sort program is loaded into memory and becomes a process

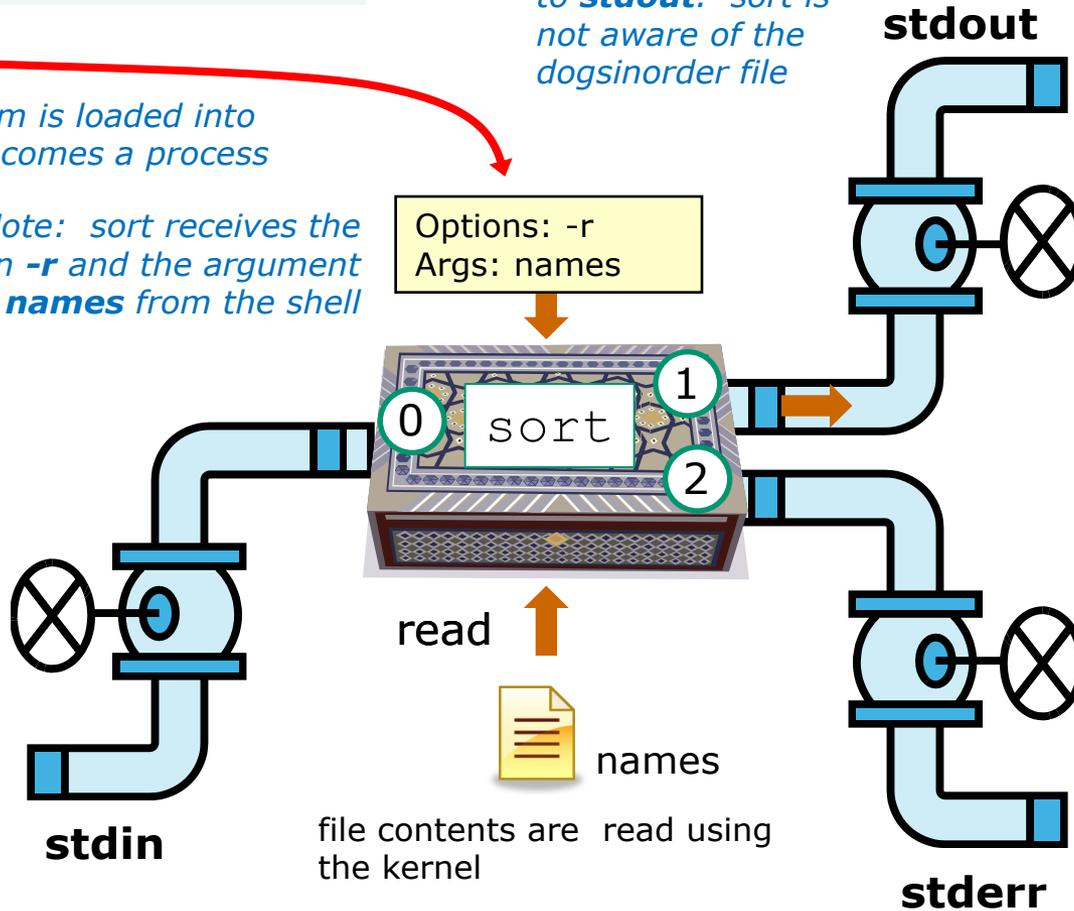
sort sends its output to **stdout**. sort is not aware of the dogsinorder file

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



```
star
homer
duke
benji
```

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



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!
```

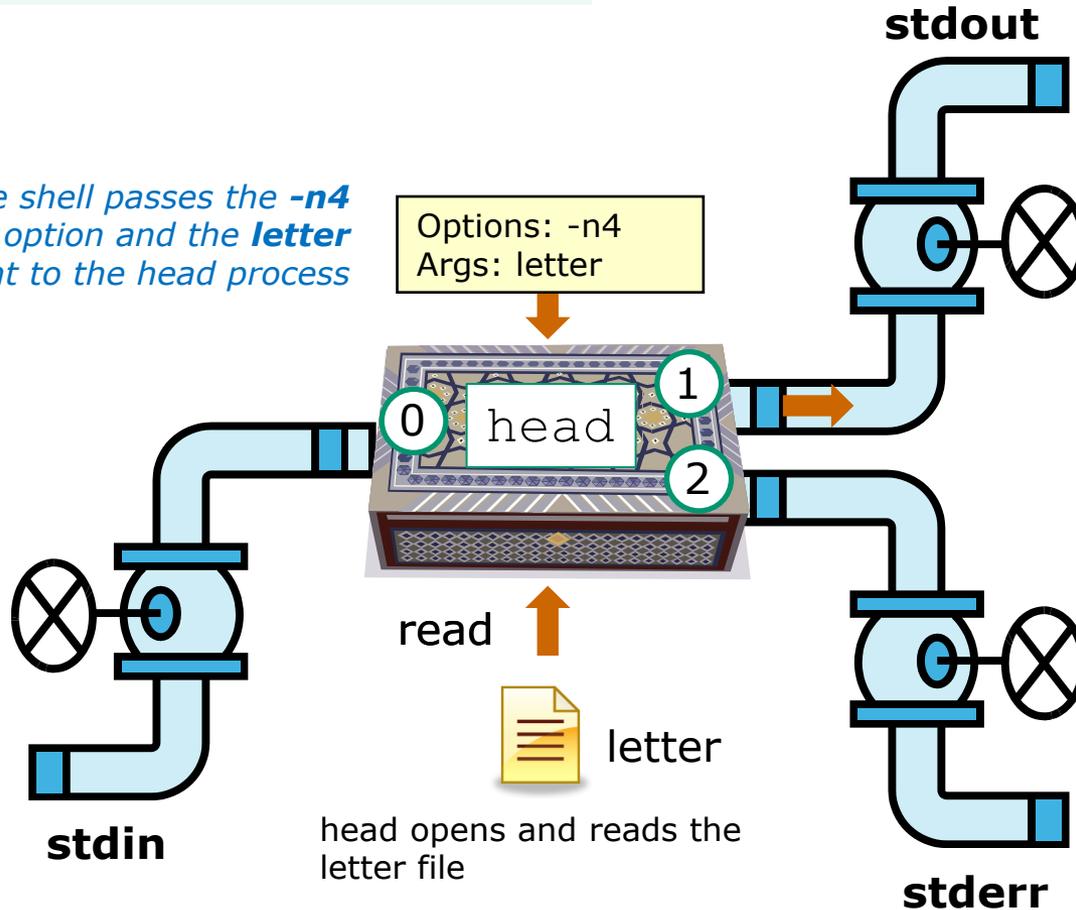
```
Here I am at Camp Granada. Things are very entertaining,  
and they say we'll have some fun when it stops raining.
```

# 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
```

**stdout**

Hello Mother! Hello Father!

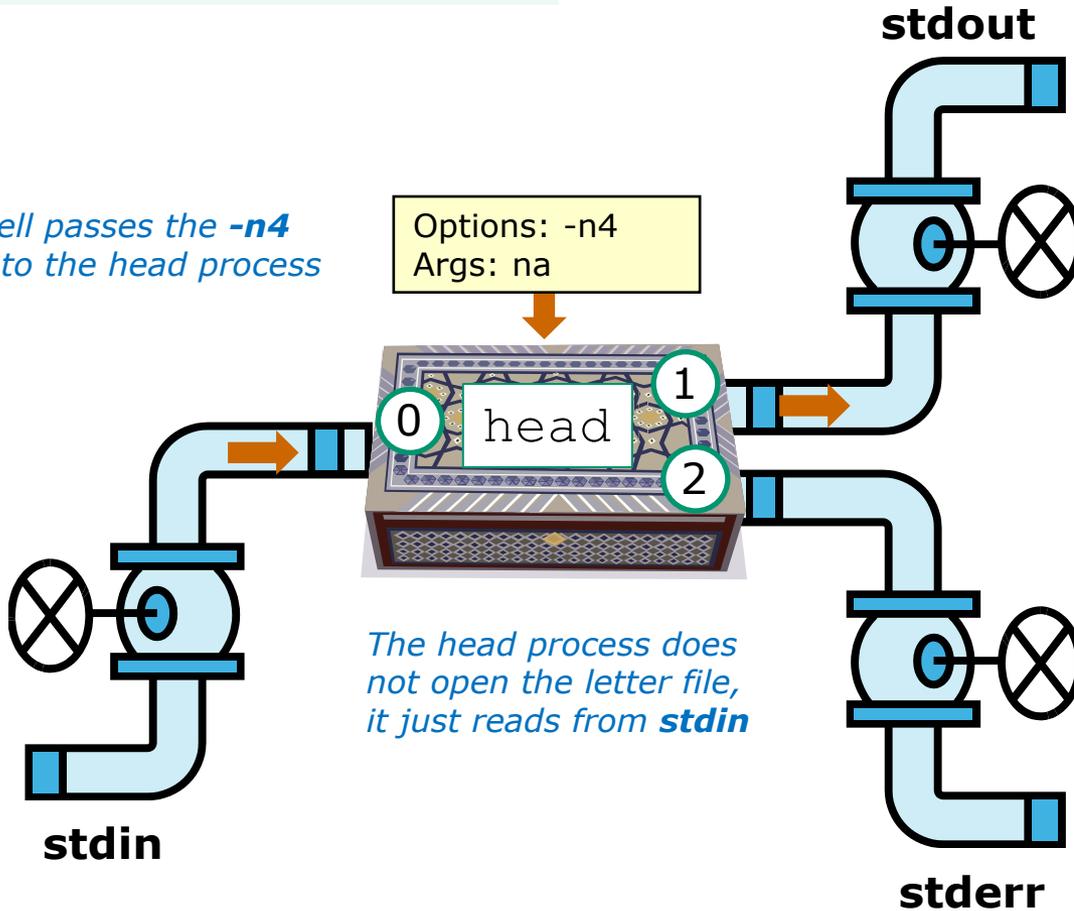
Here I am at Camp Granada. Things are very entertaining,  
and they say we'll have some fun when it stops raining.

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

Options: -n4  
Args: na

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

The head process does not open 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

\$ **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



## Test your knowledge

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

2 > & 1

FYI

(more on this in CIS 98)





## 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*



# It's descriptor clobbering time!

```
/home/cis90/simben $ bc > calculations 2> calculations
```



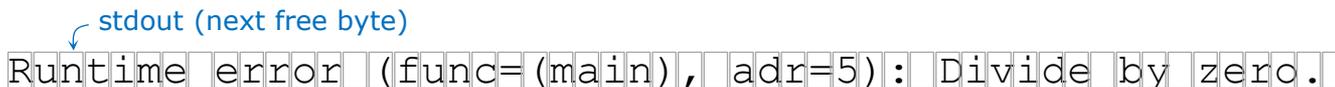
stderr (next free byte)



2+2

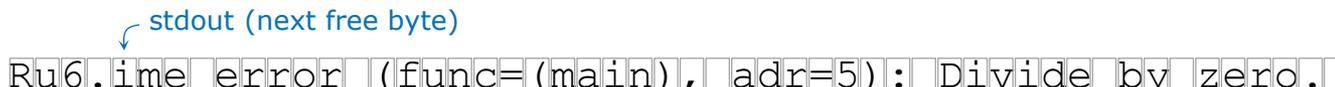
stderr (next free byte)

*The <newline> character is represented by a "."*



7/0

stderr (next free byte)



3+3

stderr (next free byte)

```
/home/cis90/simben $ cat calculations
```

```
Ru6
```

```
ime error (func=(main), adr=5): Divide by zero
```

*Each file descriptor keeps its own separate index into the calculations file for where to write the next line.*



## 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 **stderr** to the same file*

# More on I/O

(input/output)

C program  
example





## 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*



## 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)); Write question to stderr
```

```
    len = read(0, buffer, 80);
```

```
    write(1, greeting, sizeof(greeting));
```

```
    write(1, buffer, len);
```

```
}
```

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



## 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); Read users name from stdin
    write(1, greeting, sizeof(greeting));
    write(1, buffer, len);
}
```

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



## 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 greeting to stdout
    write(1, buffer, len);
}
```

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



## 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*



## 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*

## C Program I/O example

**FYI**  
only

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

*Running the simple program.*

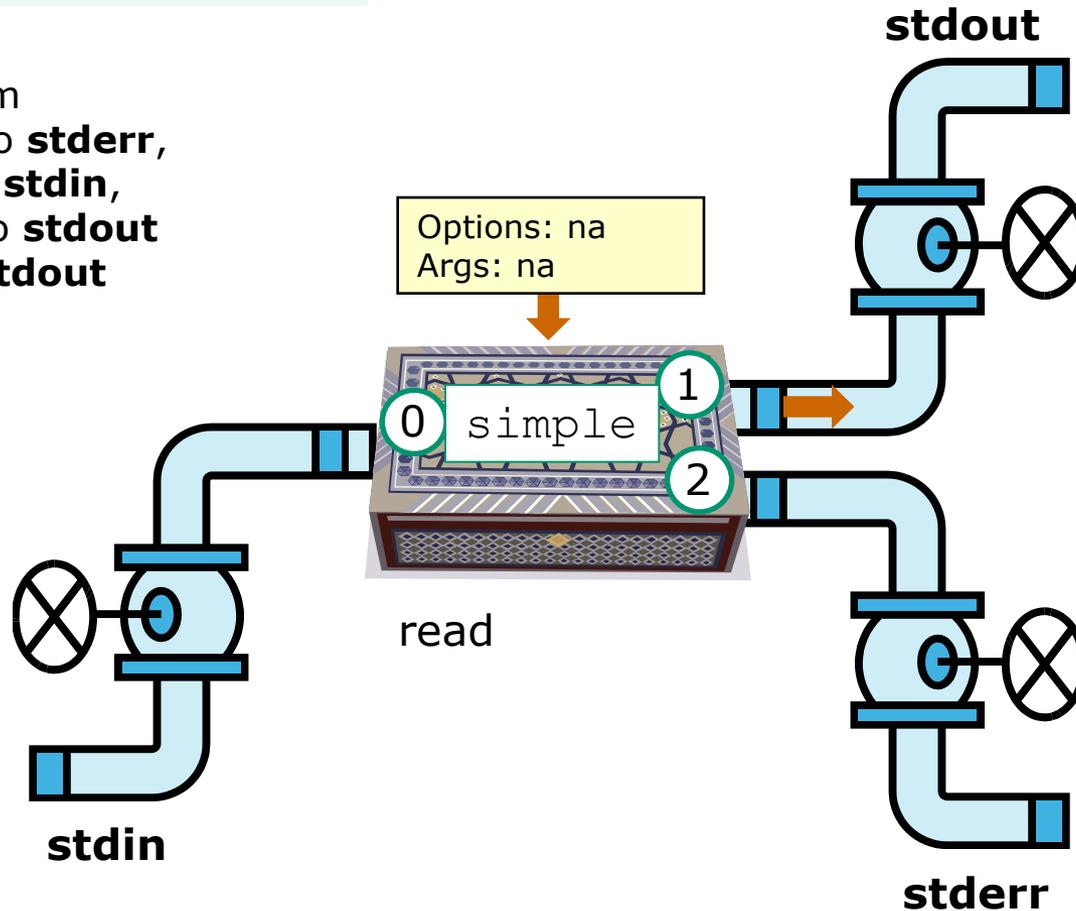
*Note I need to preface **simple** with a "./" to run it as this directory is not on my path. This is not necessary for CIS 90 students as they already have the . directory in their path.*

## C Program I/O example

```
$ ./simple
```

The **simple** program

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



2

Rich

3

Well I'm very  
pleased to meet  
you, Rich

4

1

What is your name  
stranger?



## 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
```

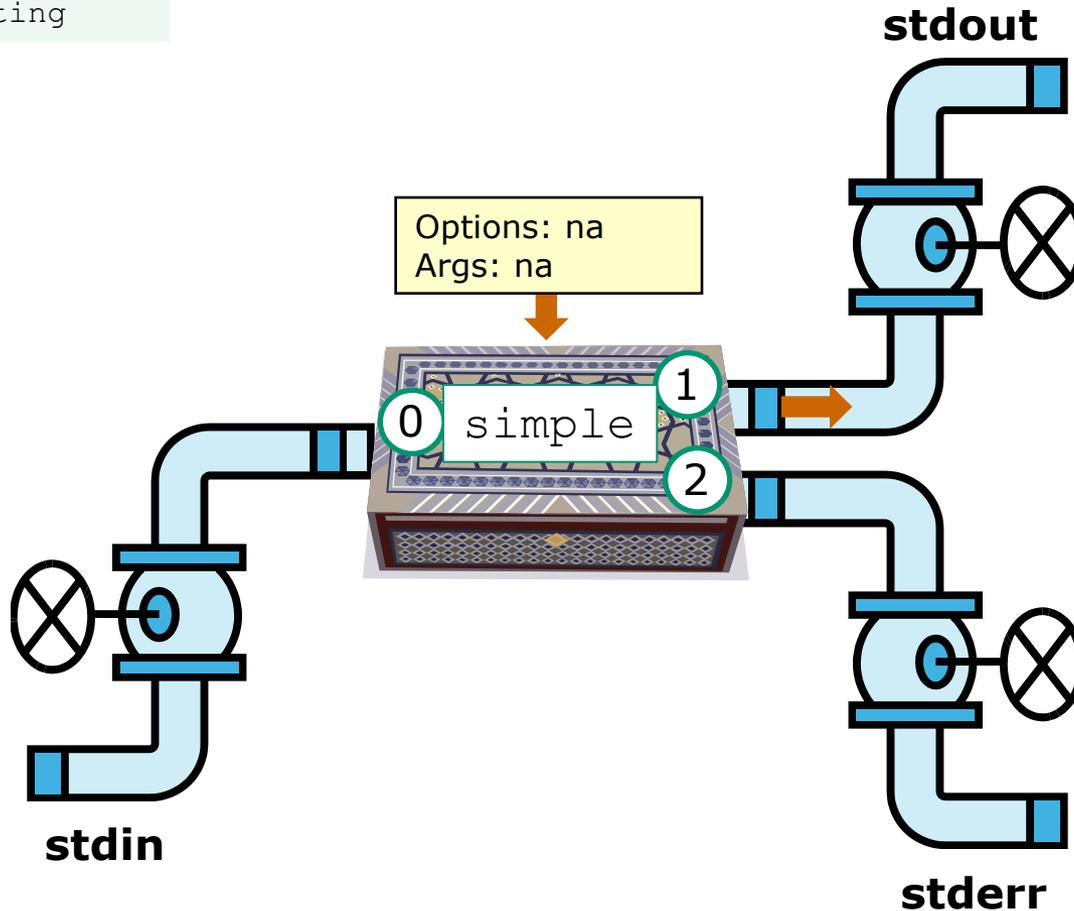
*In this example,  
output has been  
redirected to a file  
named myfile.*

*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
```

*redirection*



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. Look at your program  
**cat simple.c**
4. Compile the program  
**make simple**
5. Run the program  
**simple**

# More on umask (review)

## Review - applying umask bits

Current umask setting

```
/home/cis90/simben/lesson9 $ umask  
002
```

*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 removes the corresponding permission bit for future new files and directories*

## Review - Copying files

```
/home/cis90/simben $ umask 057
```

```
/home/cis90/simben $ umask
```

```
0057
```

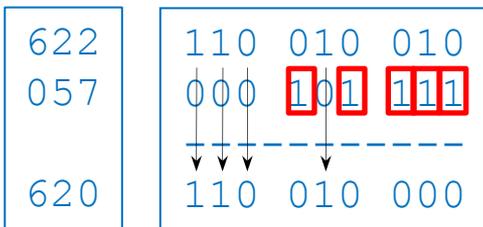
```
/home/cis90/simben $ chmod 622 myfile
```

```
/home/cis90/simben $ cp myfile myfile.bak
```

```
/home/cis90/simben $ ls -l myfile*
```

```
-rw--w--w-. 1 simben90 cis90 0 Mar 24 17:50 myfile
```

```
-rw--w----. 1 simben90 cis90 0 Mar 24 17:51 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.*



# More Pipeline Practice

# Pipelines

## Task

Record the last times Homer Miller logged in on a Monday to a file named *mylog* AND count them

**grep Homer /etc/passwd**

**milhom90**:x:1202:190:Homer Miller:/home/cis90/milhom:/bin/bash

**last**

**last | grep milhom90**

**last | grep milhom90 | grep "Mon"**

**last | grep milhom90 | grep "Mon" | tee mylog**

**cat mylog**

**last | grep milhom90 | grep "Mon" | tee mylog | wc -l**

**cat mylog**

## Class Exercise Pipeline Tasks

### Task

Count the last times Rich Simms was logged in on a Tuesday and record them in a file named mylog

```
grep "?????" /etc/passwd
```

```
last | grep ??????
```

```
last | grep ?????? | grep "Tue"
```

```
last | grep ?????? | grep "Tue" | ??? mylog
```

```
cat mylog
```

```
last | grep ?????? | grep "Tue" | ??? mylog | wc -?
```

```
cat mylog
```

*Put your answer in the chat window.*



# More Pipeline Practice

## Pipelines

### Task

Print your last name as shown in /etc/passwd:

```
cat /etc/passwd
```

```
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" "
```

## Class Exercise Pipeline Tasks

### Task

What is the first name of the user milhom90?

```
cat /etc/passwd
```

```
cat /etc/passwd | grep ????????
```

```
cat /etc/passwd | grep ????????
```

```
cat /etc/passwd | grep ????????
```

*Put your answer in the chat window.*



# More Pipeline Practice

## Pipelines

### Task

Print a sorted list of the first names for CIS 172 students

```
cat /etc/passwd
```

```
cat /etc/passwd | grep cis172
```

```
cat /etc/passwd | grep cis172 | cut -f 5 -d ":"
```

```
cat /etc/passwd | grep cis172 | cut -f 5 -d ":" | cut -f1 -d" "
```

```
cat /etc/passwd | grep cis172 | cut -f 5 -d ":" | cut -f1 -d" " | sort
```

## Class Exercise Pipeline Tasks

### Task

Print a sorted list of the first names for CIS 90 students

```
cat /etc/??????
```

```
cat /etc/?????? | grep ??????
```

```
cat /etc/?????? | grep ?????? | cut -f ? -d "?"
```

```
cat /etc/?????? | grep ?????? | cut -f ? -d "?" | cut -f? -d"?" | ?????
```

*Put your list 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
Jack and Jill went up the hill
to fetch a pail of water.
Oops .... this doesn't work!
/home/cis90/simben $
```

# xargs command

*xargs to the rescue!*



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

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

## 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!*



# 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)

*xargs to the rescue again!*

```

/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** 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 $
```

*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.*



# Things that Hide

## Finding Things

### Task

Find all files in the `/usr/src` branch of the file tree that contain "Torvalds"

`grep -r "Torvalds" /usr/src`

```

/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/powerpc/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/s390/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/parisc/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/alpha/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/x86/boot/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/x86/include/asm/387.h: * Copyright (C) 1994 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/x86/include/asm/rq.h: * (C) 1992, 1993 Linus Torvalds, (C) 1997 Ingo Molnar
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/x86/include/asm/thread_info.h: * - Incorporating suggestions made by Linus Torvalds and Dave Miller
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/x86/include/asm/stop.h: * Copyright 1992, Linus Torvalds.
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/x86/include/asm/sacktrace.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/x86/include/asm/hw_irq.h: * (C) 1992, 1993 Linus Torvalds, (C) 1997 Ingo Molnar
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/x86/include/asm/delay.h: * Copyright (C) 1993 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/x86/include/asm/sync_bitops.h: * Copyright 1992, Linus Torvalds.
/usr/src/kernels/2.6.32-220.23.1.el6.i686/arch/x86/boot/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/roport.h: * Authors: Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/pagemap.h: * Copyright 1995 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/thread_info.h: * - Incorporating suggestions made by Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/ext2_fs.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/ide.h: * with heavy changes by Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/completion.h: * (C) Copyright 2001 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/ext3_fs.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/ide.h: * Copyright (C) 1994-2002 Linus Torvalds & authors
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/delay.h: * Copyright (C) 1993 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/ext3_fs_sb.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/ext3_fs_jb.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/scripts/get_maintainer.pl:patch@pergonan.chief:Linus Torvalds:torvalds@linux-foundation.org);
/usr/src/kernels/2.6.32-220.23.1.el6.i686/scripts/package/builddeb:Copyright: 1991 - 2009 Linus Torvalds and others.
/usr/src/kernels/2.6.32-220.23.1.el6.i686/scripts/checkstack.pl: # Inspired by Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/arch/powerpc/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/arch/s390/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/arch/alpha/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/arch/alpha/boot/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/arch/x86/include/asm/387.h: * Copyright (C) 1994 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/arch/x86/include/asm/rq.h: * (C) 1992, 1993 Linus Torvalds, (C) 1997 Ingo Molnar
/usr/src/kernels/2.6.32-71.el6.i686/arch/x86/include/asm/thread_info.h: * - Incorporating suggestions made by Linus Torvalds and Dave Miller
/usr/src/kernels/2.6.32-71.el6.i686/arch/x86/include/asm/stop.h: * Copyright 1992, Linus Torvalds.
/usr/src/kernels/2.6.32-71.el6.i686/arch/x86/include/asm/sync_bitops.h: * Copyright (C) 1992 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/arch/x86/include/asm/delay.h: * Copyright (C) 1993 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/arch/x86/boot/Makefile: # Copyright (C) 1994 by Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/roport.h: * Authors: Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/pagemap.h: * Copyright 1995 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/thread_info.h: * - Incorporating suggestions made by Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/ext2_fs.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/ide.h: * with heavy changes by Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/completion.h: * (C) Copyright 2001 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/ext3_fs.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/ide.h: * Copyright (C) 1994-2002 Linus Torvalds & authors
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/delay.h: * Copyright (C) 1993 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/ext3_fs_sb.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/ext3_fs_jb.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/scripts/get_maintainer.pl:patch@pergonan.chief:Linus Torvalds:torvalds@linux-foundation.org);
/usr/src/kernels/2.6.32-71.el6.i686/scripts/package/builddeb:Copyright: 1991 - 2009 Linus Torvalds and others.
/usr/src/kernels/2.6.32-71.el6.i686/scripts/checkstack.pl: # Inspired by Linus Torvalds
[rsimms@oslab ~]$

```

*Do a recursive grep to search the **contents** of files in an entire branch of the file tree.*

## Finding Things

### Task

Count the number of files in the `/usr/src` branch of the file tree that contain "Stallman"

```
grep -? "Stallman" /???/??? | wc -?
```

*Write your answer in the chat window*

## Finding Things

### Task

Find all files in the `/usr/share/doc` branch of the file tree that are named "BUGS"

```
find /usr/share/doc -name "BUGS"
```

```
/usr/share/doc/pp1-0.10.2/BUGS  
/usr/share/doc/ltrace-0.5/BUGS  
/usr/share/doc/perl-IO-Socket-SSL-1.31/BUGS  
/usr/share/doc/glibc-2.12/BUGS  
/usr/share/doc/parted-2.1/BUGS  
/usr/share/doc/cvs-1.11.23/BUGS  
/usr/share/doc/patchutils-0.3.1/BUGS  
/usr/share/doc/procps-3.2.8/BUGS  
/usr/share/doc/gettext-0.17/BUGS  
/usr/share/doc/curl-7.19.7/BUGS  
/usr/share/doc/sed-4.2.1/BUGS  
/usr/share/doc/SDL-1.2.14/BUGS  
/usr/share/doc/cairo-1.8.8/BUGS  
/usr/share/doc/emacs-common-23.1/BUGS  
/usr/share/doc/tcsh-6.17/BUGS  
/usr/share/doc/unzip-6.0/BUGS  
/usr/share/doc/vsftpd-2.2.2/BUGS  
/usr/share/doc/dejavu-fonts-common-2.30/BUGS  
/usr/share/doc/nano-2.0.9/BUGS  
[rsimms@oslab ~]$
```

*Use find to search for files by name, type, user, group, etc.*

## Finding Things

### Task

Count all the files in the `/home` branch of the file tree that are owned by `rsimms`. Discard any permission errors.

```
find /???? -user ?????? 2> /dev/??? | ?? -l
```

*Write your answer in the chat window*



# Eggs



## Egg Hunt

Instructor: `sudo /home/rsimms/cis90/basket/hide-the-eggs`

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.  
`ls -l /home/rsimms`
2. Make a new directory named *basket* in your home directory and see how many egg files you can move into it.  
`mkdir -p /home/rsimms/basket`
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.

# 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



**Room 1**



**Room 2**



**Room 3**



**Room 4**



**Room 5**

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

**Points:**



Room 2

**Points:**



Room 3

**Points:**



Room 4

**Points:**



Room 5

**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

# Practice Test

```

C:\Users\Rich Simms\Dropbox\data-my-cis90\cis-90-TEST-2-Sp14-practice.txt - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
cis-90-TEST-2-Sp14-practice.txt
1 CIS 90 - Spring 2014 - PRACTICE TEST 2 - 30 points
2
3 HONOR CODE:
4 This is a practice test and you may work with others on it. However on the real test you
  must work alone. Submit the practice test even though it's not graded to verify you know
  how to do it correctly. Feel free to compare and discuss answers to the practice test on
  the forum.
5
6 NAME: --> please replace this text with your name <--
7
8 INSTRUCTIONS:
9 Download or copy and paste this page from your web browser into a text file on your
  computer. Don't use a word processor like MS Word! Instead use a text editor like Notepad
  (Windows) or TextWrangler (Mac) to add your answers to the questions below.
10
11 This test should be completed using the Sun-Hwa system only. Log into Opus first then ssh
  into Sun-Hwa.
12
13 For questions with a *** you will be expected to do the requested operation in addition to
  answering the question. When grading, the instructor will check your answers and verify the
  operations were successfully completed. For questions not marked with a *** it may still be
  helpful to use Sun-Hwa to check your answers.
14
15 [ ]'s are used to indicate the directory you should be in to do an operation. This will be
  your starting point for any relative pathnames.
16
17 Please KEEP YOUR ANSWERS TO A SINGLE LINE ONLY and preserve the tags, e.g. "A1)", "A2)",
  etc. used to label the answers.
18
19 Note to instructor:
20 - Create student and extra accounts (Sun-Hwa)
21 - Run setup-test2 P2 (Sun-Hwa)
22 - Remove /etc/nologin (Sun-Hwa)
23
24 [ ----- ]
25 [ Troubleshooting (1 point each) ]
26 [ ----- ]
27
28 Instructions: Login to Sun-Hwa from Opus using just: ssh sun-hwa
29
30 Q1) *** Before doing anything else run the flag command. What is the output from the flag
  command?
31 A1)
Normal text file length: 11734 lines: 31 Ln: 12 Col: 1 Sel: 0 DosWindow ANSI INS
  
```

## Practice test available

- Available on the website
- Work alone or together
- Use the forum to compare answers and approaches to questions
- Test #2 will be graded by looking at both your answers to the questions and the work you did on the testing server.

8 INSTRUCTIONS:

9 Download or copy and paste this page from your web browser into a text file on your  
computer. Don't use a word processor like MS Word! Instead use a text editor like Notepad  
(Windows) or TextWrangler (Mac) to add your answers to the questions below.

10  
11 This test should be completed using the Sun-Hwa system only. Log into Opus first then ssh  
into Sun-Hwa.

12  
13 For questions with a \*\*\* you will be expected to do the requested operation in addition to  
answering the question. When grading, the instructor will check your answers and verify the  
operations were successfully completed. For questions not marked with a \*\*\* it may still be  
helpful to use Sun-Hwa to check your answers.

14  
15 [ ]'s are used to indicate the directory you should be in to do an operation. This will be  
your starting point for any relative pathnames.

16  
17 Please KEEP YOUR ANSWERS TO A SINGLE LINE ONLY and preserve the tags, e.g. "A1)", "A2)",  
etc. used to label the answers.

18

```
217 [ ----- ]
218 [ Submit this test ]
219 [ ----- ]
220
221 When finished, leave your home directory intact. Copy and paste this completed test into a
text-only email with NO attachments to:
222
223     rsimms@oslab.cabrillo.edu
224     <your-opus-username>@oslab.cabrillo.edu
225
226 CONFIRM on Opus that your email was successfully sent and that your work is READABLE using
the mail command.
227
228
```

# Wrap up

## Next Class

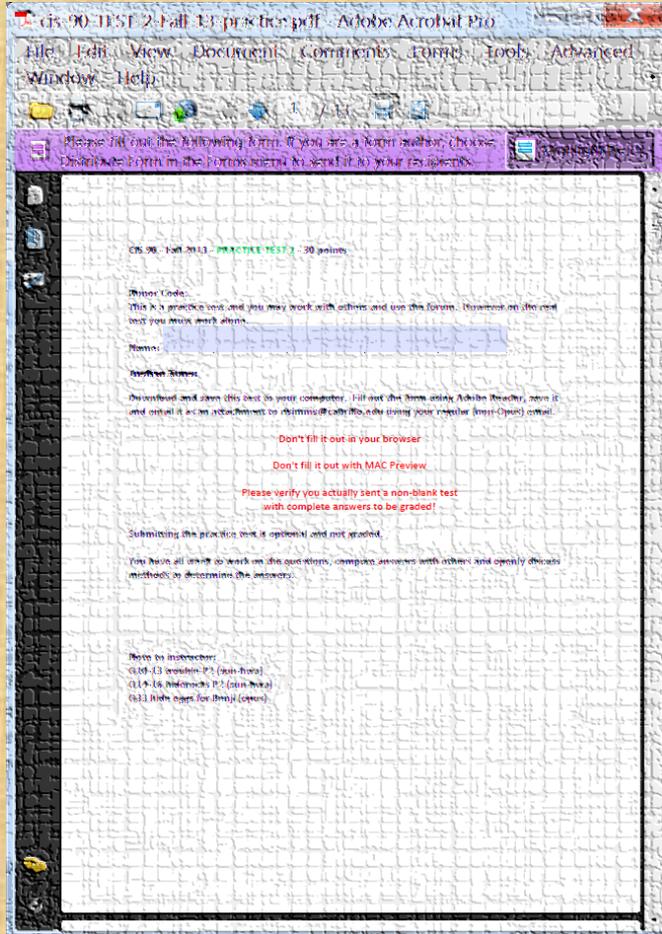
No Quiz

**Test 2!**

Cumulative Test (30 points) with focus on Lessons 6-9:

- 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 6-9 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



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