

Lesson Module Checklist

- Slides
- WB converted

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

- Lock turnin directory at midnight
- Opus - hide script tested
- Practice test ready on Blackboard
- P2 Test system online and unlocked

- 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: **888-886-3951**

Passcode: **136690**



Francisco



Leila



Justin



Jesus



Shenghong



Paul



Roberto



Sam



Navin



Jimmy



Luis



Tommy



Adrian



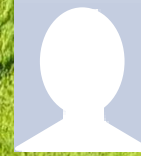
Ann



Cameron



Cody



Alejandrino



Deane



Nadia



Richard Z.



Gabriel



Ryan



Takashi



Jeff



Nick



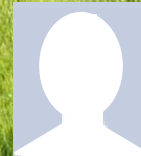
Jonathan



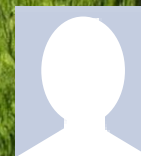
Shea



Dylan



Joshua



Richard I.



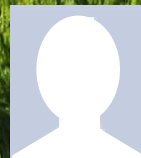
Aaron



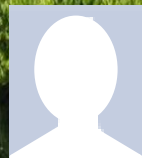
Nicole



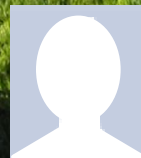
James



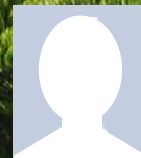
Matthew



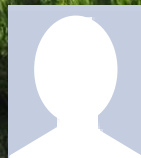
Abraham



Chris



Ronald

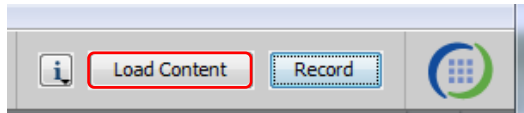


Scott



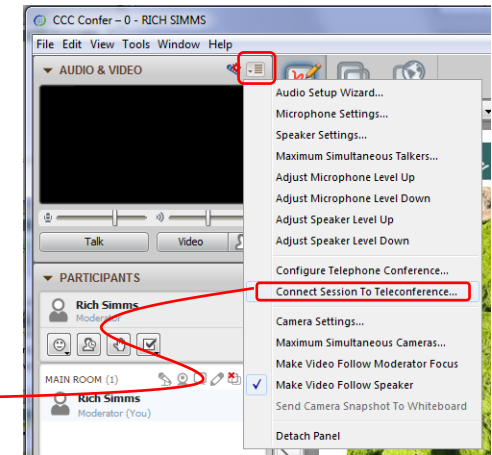
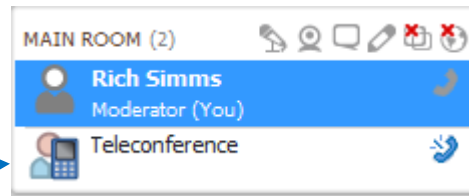
Instructor CCC Confer checklist

[] Preload White Board



[] Connect session to Teleconference

Session now connected to teleconference



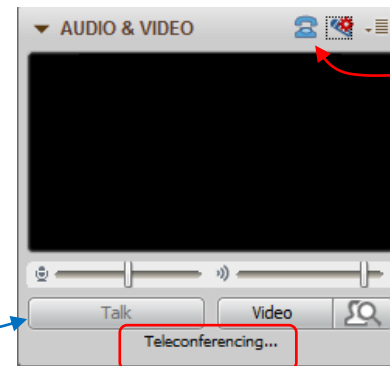
[] Is recording on?



Red dot means recording

[] Use teleconferencing, not mic

Should be greyed out



Should show as this live "off hook" telephone handset icon and the Teleconferencing ... message displayed



Instructor CCC Confer checklist

The screenshot displays a Windows desktop with several applications open:

- CCC Confer**: A video conferencing window on the left showing a participant named Rich Simms.
- foxit for slides**: A Foxit Reader window displaying a PDF document titled 'cis90lesson07.pdf'.
- chrome**: A Google Chrome browser window showing a webpage from 'simms-teach.com' with flashcard questions.
- putty**: A terminal window showing a login session for 'simben90' on 'oslab.cabrillo.edu'.
- vSphere Client**: A vCenter console window showing the vSphere interface.

Red boxes with arrows point to these applications, labeling them as 'foxit for slides', 'chrome', and 'vSphere Client'. The terminal window shows the following output:

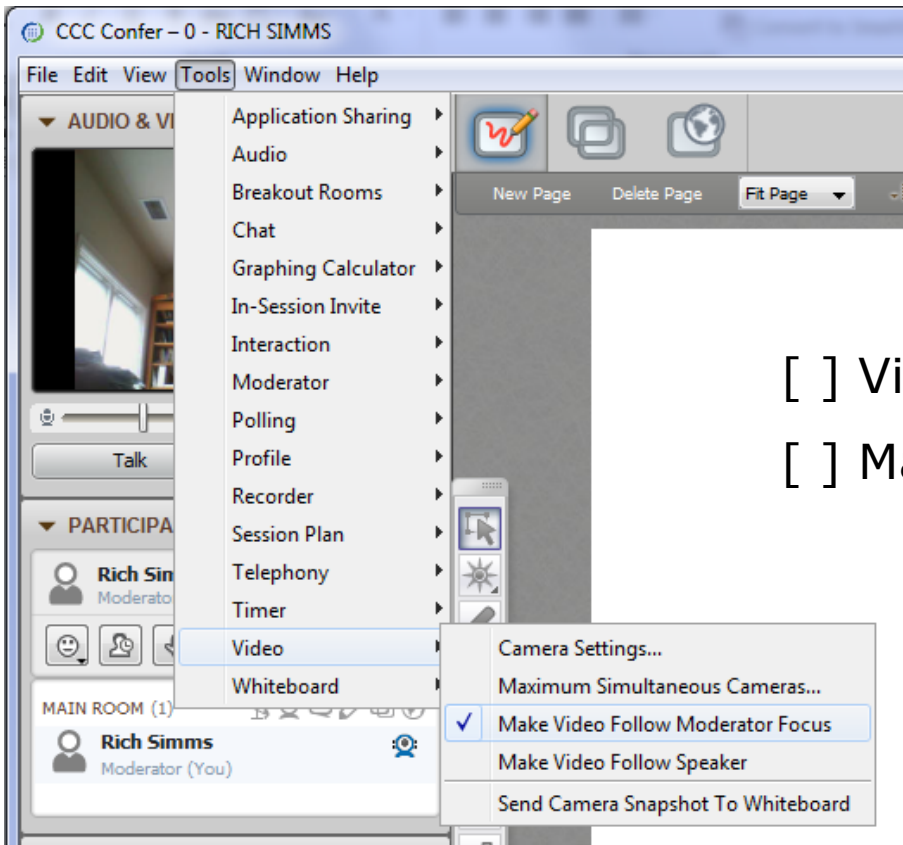
```
simben90@oslab:~  
login as: simben90  
simben90@oslab.cabrillo.edu's password:  
Access denied  
simben90@oslab.cabrillo.edu's password:  
Last login: Mon Oct 8 18:58:43 2012 from  
d.com
```

[] layout and share apps





Instructor CCC Confer checklist

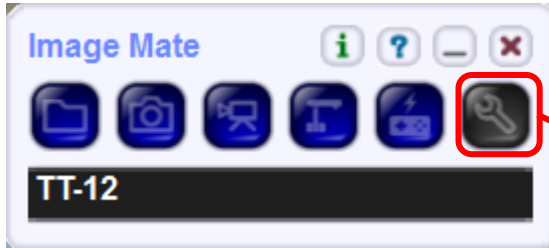


[] Video (webcam)

[] Make Video Follow Moderator Focus



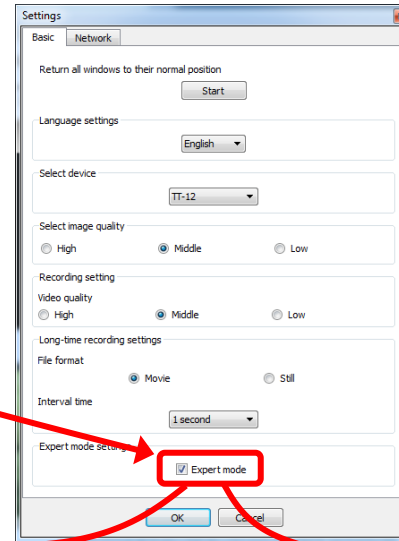
Using Elmo with CCC Confer



Elmo rotated down to view side table



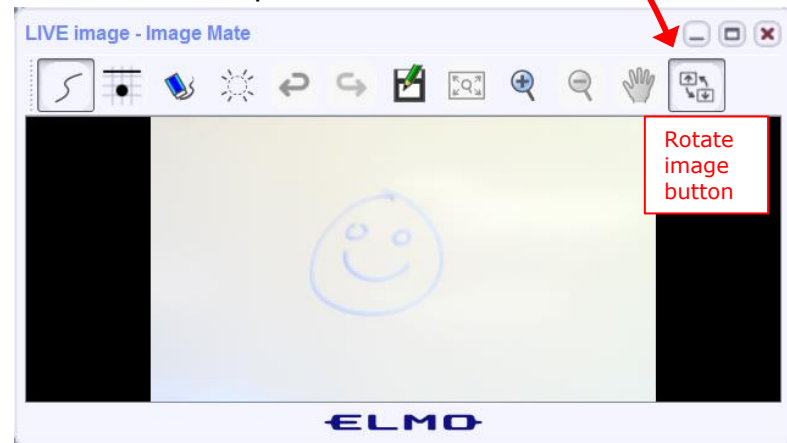
Run and share the Image Mate program just as you would any other app with CCC Confer



The "rotate image" button is necessary if you use both the side table and the white board.

Quite interesting that they consider you to be an "expert" in order to use this button!

Elmo rotated up to view white board



Instructor CCC Confer checklist

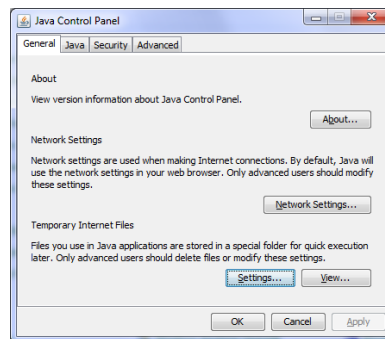
Universal Fix for CCC Confer:

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

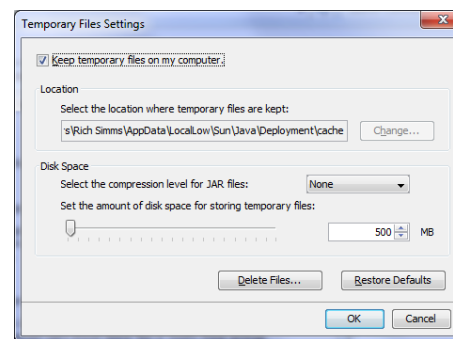
Control Panel (small icons)



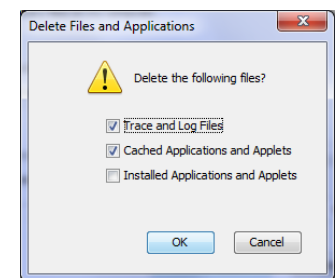
General Tab > Settings...



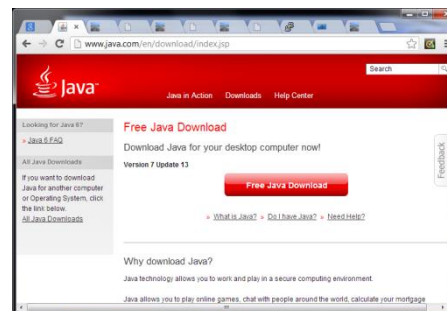
500MB cache size



Delete these



Google Java download



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) 10

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

Lab 6 Results

(steps where points were taken off)

- Step 01 -
- Step 02 -
- Step 03 - xxx
- Step 04 -
- Step 05 -
- Step 06 -
- Step 07 -
- Step 08 -
- Step 09 - xxxxx x
- Step 10 - xxxx
- Step 11 - x
- Step 12 - xxx
- Step 13 - xx
- Step 14 - xxxxx x
- Step 15 - xxxxx xx
- Step 16 - xxxxx
- Step 17 -
- Step 18 -
- Step 19 -
- Step 20 -
- Step 21 -
- Step 22 -
- Step 23 - xxxx

Set the permissions of your poems directory and ...

Set all ordinary files under the poems ...

Change the permissions of your bin ...

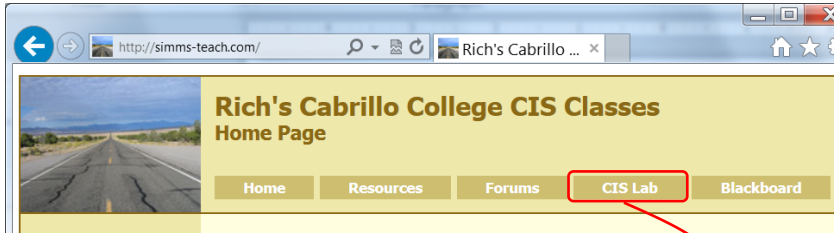
Make all ordinary files under class/labs and ...

Try setting the umask to 777 ...

*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/>

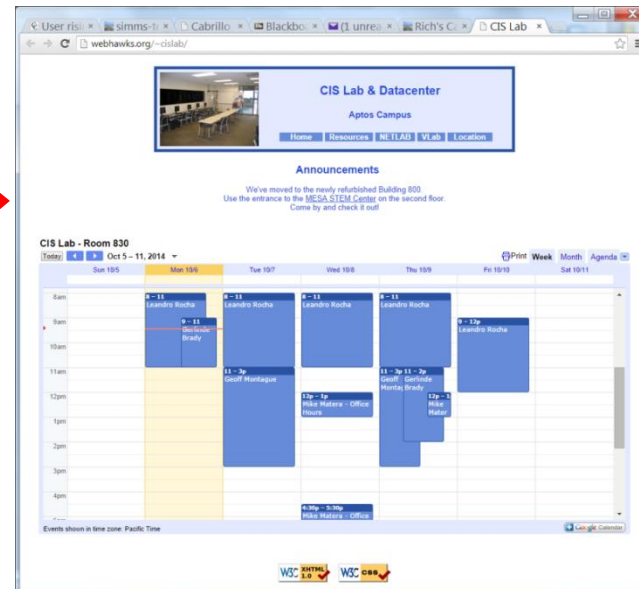


Not submitting tests or lab work?

If you would like some additional help come over to the CIS Lab.

*Leandro and Geoff are both
CIS 90 Alumni.*

*Michael is the other Linux
instructor.*



Or hang around after class. Rich has his office hours right after each class in Room 828.

CIS 90 Tutoring Available

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

The screenshot shows the website for the Tutorials Center at Cabrillo College. The main content area is titled 'TUTORIALS' and includes an 'ANNOUNCEMENTS & DEADLINES' section with a list of subjects: New subjects for Spring 2014, American Sign Language, Computer Applications/Business Technology (CABT), Computer and Information Systems (CIS), and History 17A. Below this is a 'Welcome to the Tutorials Center!' section with a list of services: Tutoring is by appointment, Sessions are weekly, Tutoring sessions are scheduled in small groups, and Come directly to the TC office. A list of classes being tutored for Spring 2014 is provided, with 'Computer and Information Systems (CIS) 81, 90, 172' highlighted in a red box. The 'CONTACT INFORMATION' section for the Tutorials Center includes the location (Room 1080A), phone (831.479.6470), email (tutorialscenter@cabrillo.edu), coordinator (Lori Chavez), and hours (Monday-Thursday: 9am-5pm, Friday: 9am-1pm).



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.

More CIS 90 Tutoring Available

(1 unread) - rich x User risimms log x Cabrillo College x Rich's Cabrillo C x

oslab.cis.cabrillo.edu/forum/viewtopic.php?f=101&t=3324&sid=63dda9cf0a544936a540e216474d4c16

phpBB® creating communities
Cabrillo College: Computer and Information Systems
Forum for students in the Computer Networking and System Administration and/or Computer Support Specialist programs

Search... Search
Advanced search

Board index < Cabrillo College Fall 2014 Courses < CIS 90 - Fall 2014

FAQ Register Login

Do you need tutoring ?

POSTREPLY Search this topic... Search 5 posts • Page 1 of 1

Do you need tutoring ?
by Takashi Tamasu » Thu Oct 16, 2014 9:37 pm

I belong to the AGS (Alpha Gamma Sigma) Honor Society at Cabrillo and one of the functions this club does is offer FREE tutoring. One of the tutors listed CIS 90 as one of the classes that he is willing to tutor. If someone needs tutoring you can either submit a tutor request form at our site or tell me a number and when you can be reached at that number to arrange tutoring. [https://sites.google.com/site/cabrilloa ... edirects=0](https://sites.google.com/site/cabrilloa...edirects=0)

BTW I am the tutor coordinator for AGS

cheers Takashi

Takashi Tamasu
Posts: 59
Joined: Wed Jan 29, 2014 3:46 pm



Housekeeping

Housekeeping

1. Lab 7 due 11:59PM tonight -- **don't forget to submit your latest version!** (read your Opus email for submittal status)
2. A **check7** script is available
3. Test #2 is **next week**
4. Blackboard Practice Test #2 available at 3PM today. Will not be available after real test starts.
5. No lab assigned this week (so you can work on the practice test)
6. Ask your questions regarding the test on the forum **BEFORE** the next class starts!

Spring 2015 CIS Classes

The screenshot shows a web browser window displaying a forum post on the phpBB platform. The browser's address bar shows the URL: `oslab.cis.cabrillo.edu/forum/viewtopic.php?f=101&t=3364&sid=868a592b85461cbec5d24f5`. The forum header includes the phpBB logo with the tagline 'creating communities', the site name 'Cabrillo College: Computer and Information Systems', and a search bar. Below the header, navigation links include 'Board index < Cabrillo College Fall 2014 Courses < CIS 90 - Fall 2014'. The main content area features a post titled 'Spring 2015 CIS Courses' by user 'Rich Simms', dated 'Sat Oct 25, 2014 9:37 am'. The post text reads: 'Hot off the presses! If you like what you are learning and want more, check out the Cabrillo College CIS courses being offered for the Spring 2015 term: <https://sites.google.com/a/cabrillo.edu/test/> If you have completed CIS 81 (Networking Fundamentals) then you will be ready for CIS 192AB (UNIX/Linux Network Administration). In this class you learn how to setup networks, gateways, firewalls; build network services like DHCP, DNS, web, mail and file servers and more. - Rich'. A user profile for 'Rich Simms' is shown on the right, with 1484 posts and a join date of 'Sat Jan 16, 2010 6:47 pm'. The post is the only one on the page, as indicated by '1 post • Page 1 of 1'.



To nas_alladjuncts@cabrillo.edu, nas_allclassified@cabrillo.edu, nas_allfulltime@cabrillo.edu

Hello,

Please take some time in class **this week** to remind your students that they will lose their priority registration in spring 2015 if they do not complete the following by this **Friday, October 31**:

- Declare an active Program/Major
- Begin an education plan using the Student Planning tool

To do both of these, students must log in to WebAdvisor and look under the Academic Planning area on the home page of the Student Menu. They should select "Verify/Change My Program/Major" to declare a major. In order to have an education plan, they should select "Student Planning" and must have **at least one spring 2015 course** planned using this tool. The instructions are attached. I logged on and the easiest way to do this is "Search for Courses" from the Student Planning home page, then choose "Add Course to Plan" and then choose "Spring 2015."

Other items that will influence their registration priority:

- Complete Assessment/Orientation (if needed)
- Be in good Academic Standing
- Have completed fewer than 100 degree-applicable units (except for exempted majors)

Thank you,

Sarah

Final Exam

Test #3 (final exam)

	12/18	<p>Test #3 (the final exam)</p> <p>Time</p> <ul style="list-style-type: none"> • 1:00PM - 3:50PM in Room 828 <p>Materials</p> <ul style="list-style-type: none"> • Test (blackboard) <p>CCC Confer</p> <ul style="list-style-type: none"> • Enter virtual classroom • Class archives 		<p>5 posts</p> <p>Lab X1</p> <p>Lab X2</p>
--	-------	--	--	--

- All students will take the test at the same time.
- Working students will need to plan ahead to take time off from work for the test.

<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 11/27/2014



Points that could have been earned:

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

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

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 - vSphere Client

File Edit View Inventory Administration Plug-ins Help

Home Inventory Inventory

vmserver2

- ds1
- ds2
- Hershey
- Jeff
- ns1
- ns2
- ns-slinky
- opus
- Sun-Hwa
- Sun-Hwa-II
- UCSClient
- UCSPE1
- UCSPE2
- UCSPE3
- vCenter2y

opus

Getting Started Summary Resource Allocation Performance Events Console Permissions

General

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

Resources

Consumed Host CPU: **8 MHz**
 Consumed Host Memory: **792.00 MB**
 Active Guest Memory: **10.00 MB** [Refresh Storage Usage](#)
 Provisioned Storage: **25.11 GB**
 Not-shared Storage: **21.93 GB**
 Used Storage: **21.93 GB**

Storage	Drive Type	Capacity
disk2-1	Non-SSD	456.50 GB 120

Network Type
 Server Network Standard port group

Recent Tasks Name, Target or Status contains: Clear

Name	Target	Status	Details	Initiated by

Tasks License Period: 63 days remaining root

Opus is a VM running on one of the ESXi servers in the CIS Lab

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/>

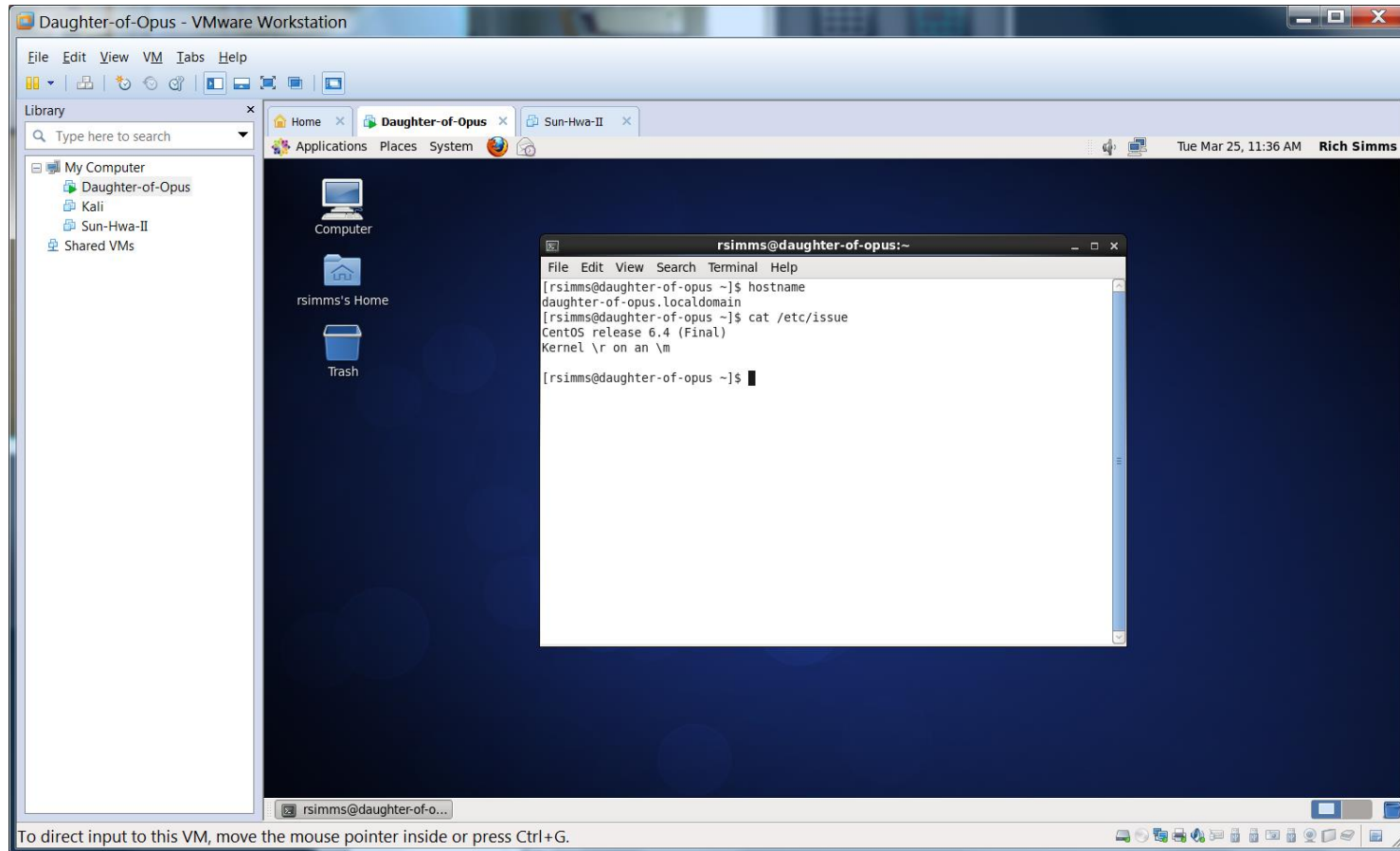
4)



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

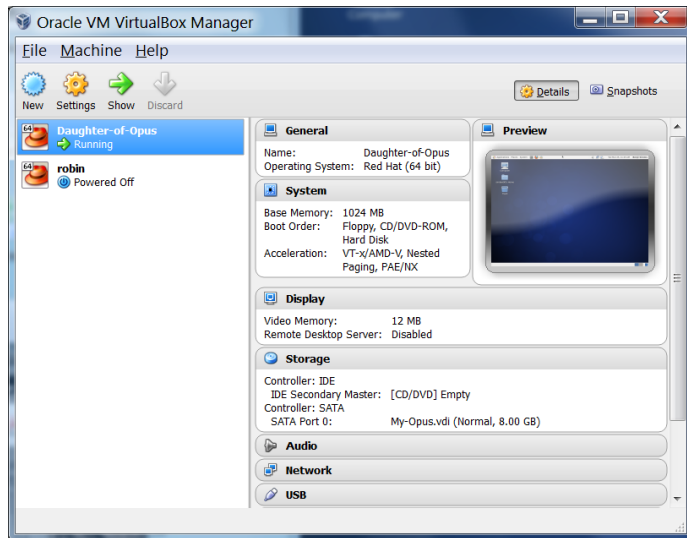
- 1) Power On with USB stick ==> Kali Linux
- 2) Power On without USB stick ==> Windows

One Daughter-of-Opus VMware Workstation

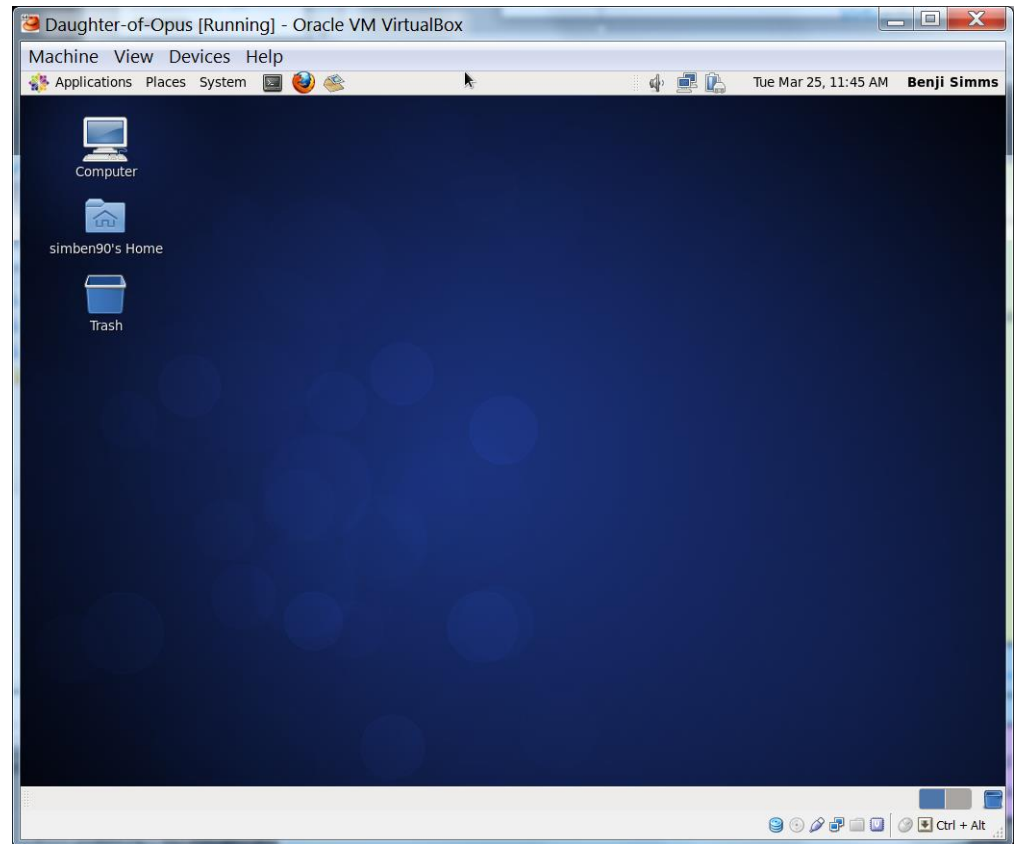


*One Daughter-of-Opus is a VM running on my laptop using
VMware Workstation (expires in one year)*

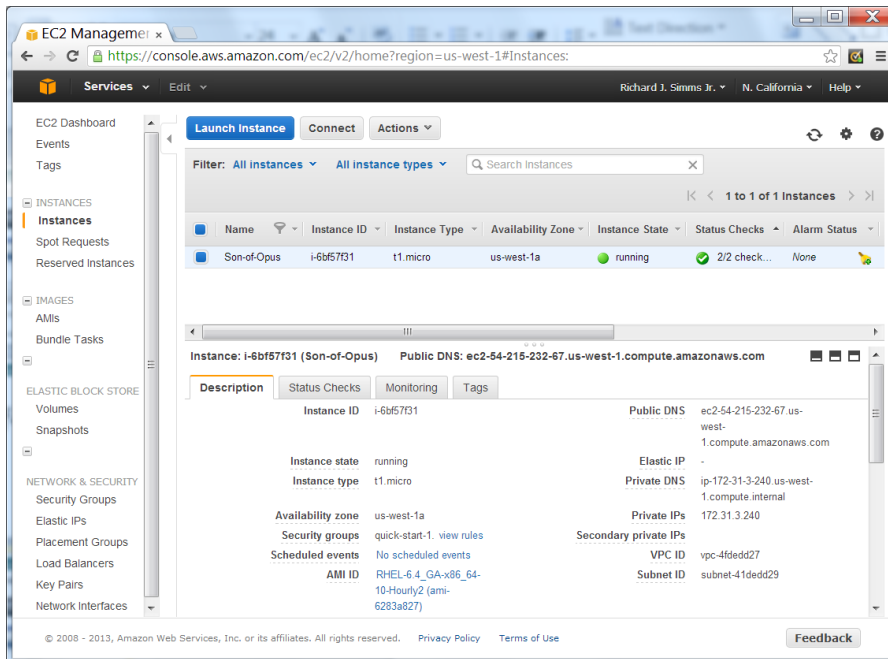
Another Daughter-of-Opus Oracle VirtualBox



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



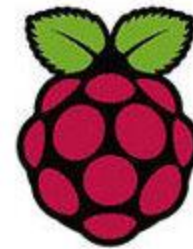
Son-of-Opus Amazon Web Services



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 (Raspian) 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

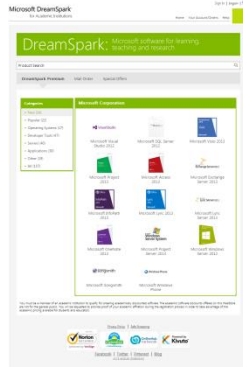


VMware ESXi for virtualization



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

Linux Distros (ISOs)



VirtualBox

More on I/O

(input/output)

Input and Output

File Redirection

The 3 standard UNIX file descriptors:

Name	Integer Value
stdin (st andard in)	0
stdout (st andard out)	1
stderr (st andard error)	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

```

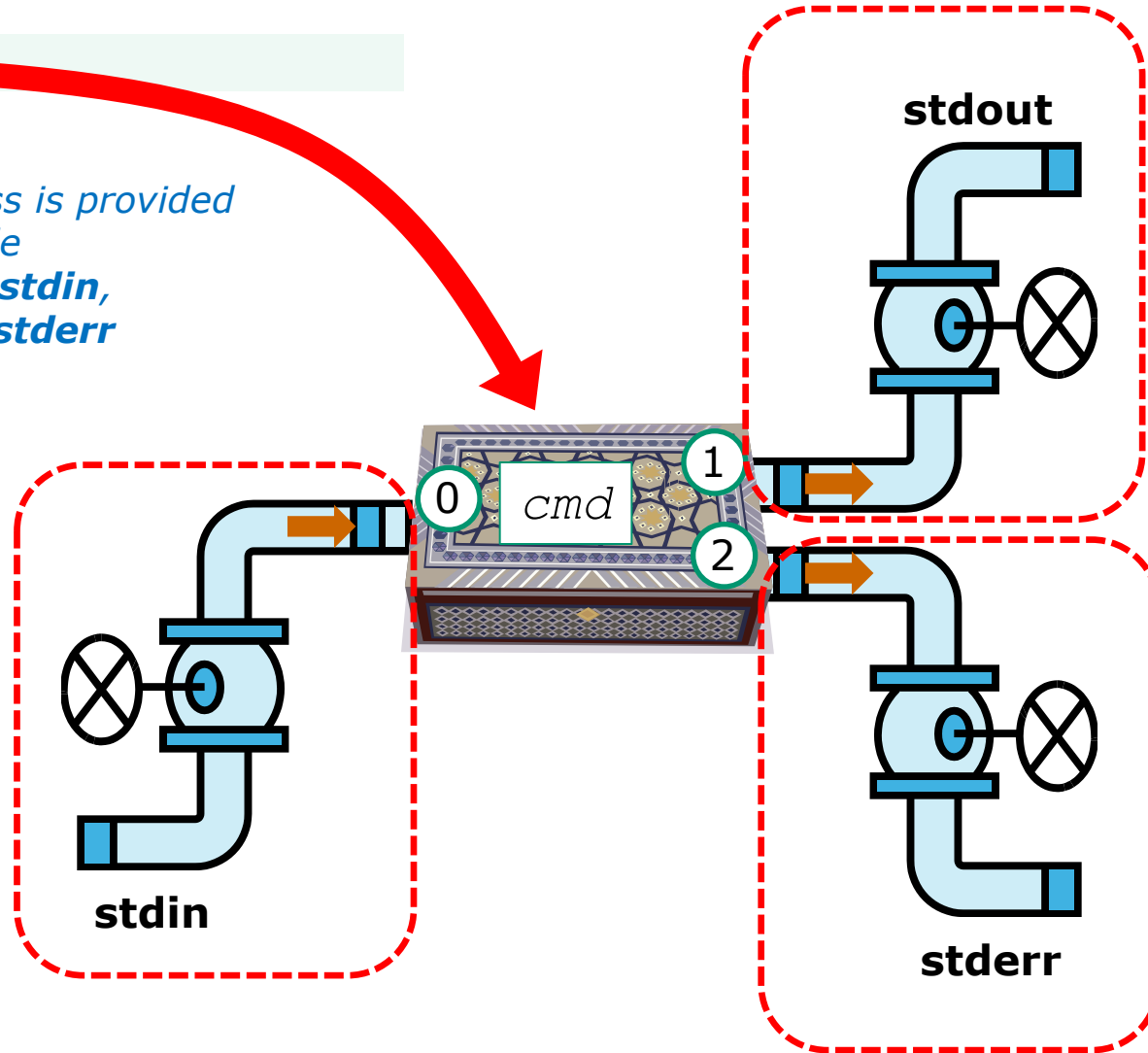
/home/cis90/simben $ cat
/home/cis90/simben $ cat -A letter
/home/cis90/simben $ cat < letter
/home/cis90/simben $ cat -b < letter > out
/home/cis90/simben $ cat bogus 2> /dev/null
/home/cis90/simben $ cat -e < bogus 2> /dev/null
/home/cis90/simben $ cat -e < letter > out 2> /dev/null

```

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

```
$ cmd
```

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

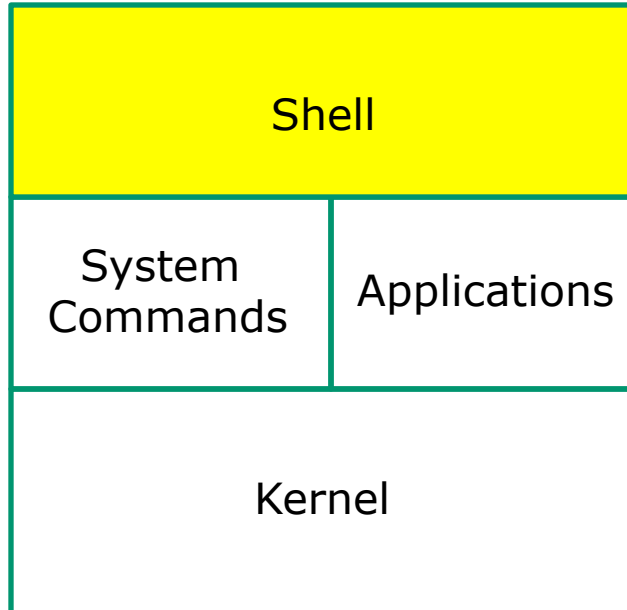




All Together Now Example



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.

Activity

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

The prompt is defined by your PS1 variable

1. Look at the contents of your PS1 variable: **echo \$PS1**
2. Look at the contents of your PWD variable: **echo \$PWD**
3. Send me and yourself the contents of your prompt variable:
echo \$PS1 | mail -s "my prompt" rsimms \$LOGNAME
4. Paste the value of your PWD variable into the chat window when finished

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*

Activity

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

The newline character is an invisible metacharacter that triggers the shell to process the command

1. Put five characters in a file named *five*: **echo 12345 > five**
2. Show the size of your *five* file: **ls -l five**
3. Indicate the size of your *five* file using CCC Confer poll: A=5, B=6, C=other
4. Do a hex dump of your *five* file: **xxd five**
5. Paste the hex value of the newline character in the chat window.
6. Optional: Use **man ascii** to check your answer

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

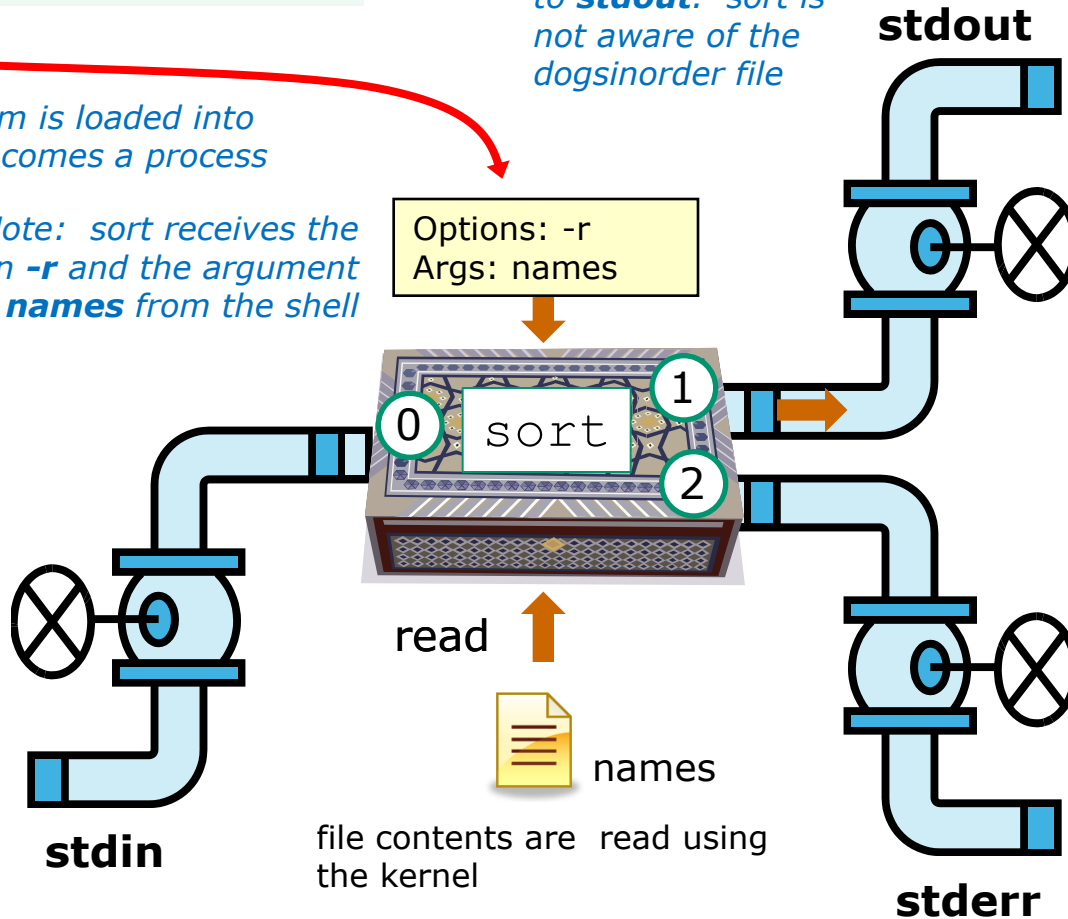
sort sends its output to **stdout**. sort is not aware of the **dogsinorder** file

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



sort opens and reads the names file

Activity

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

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

What two text strings parsed by the shell were passed to the sort command to process?

Put your answer in the chat window

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!
```

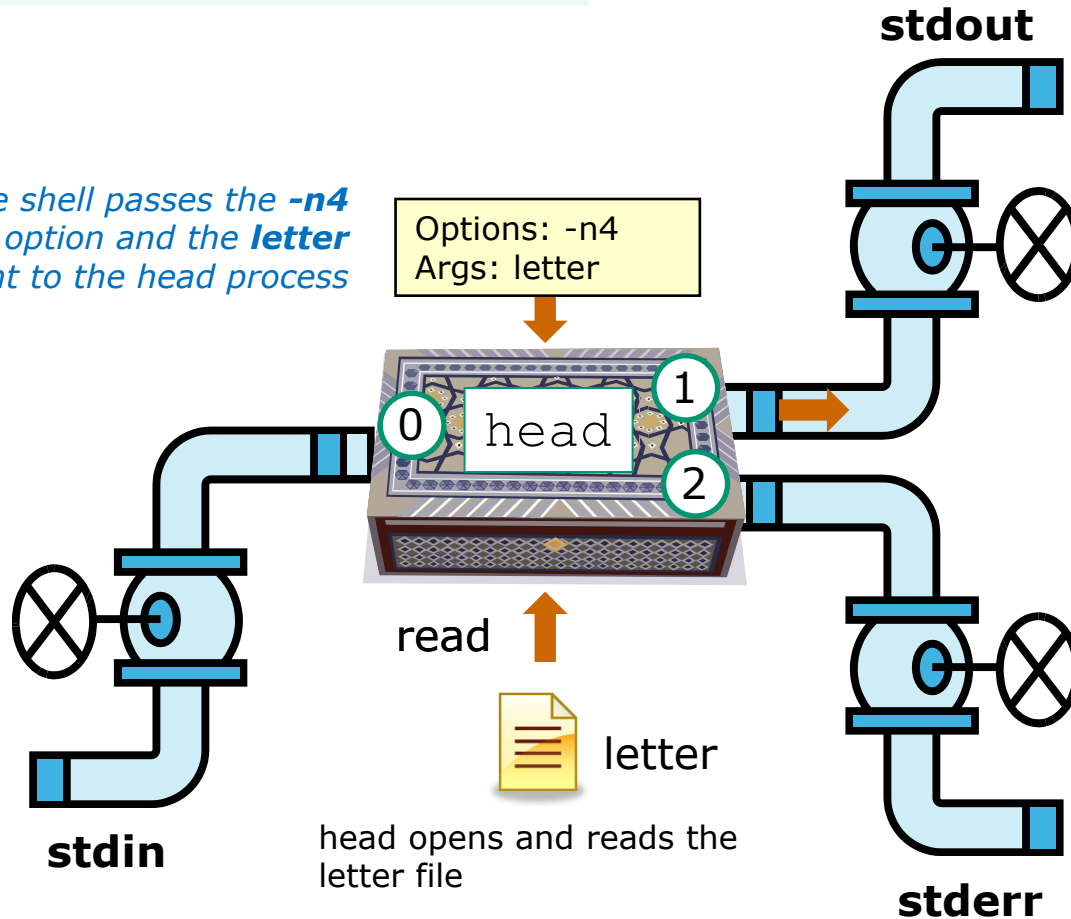
```
Here I am at Camp Granada. Things are very entertaining,  
and they say we'll have some fun when it stops raining.
```


head -n4 letter

option → ← *argument*

```
$ head -n4 letter
```

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



head opens and reads the letter file

head opens and reads the letter file



head -n4 < letter

option → *redirection*

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

stdout

Hello Mother! Hello Father!

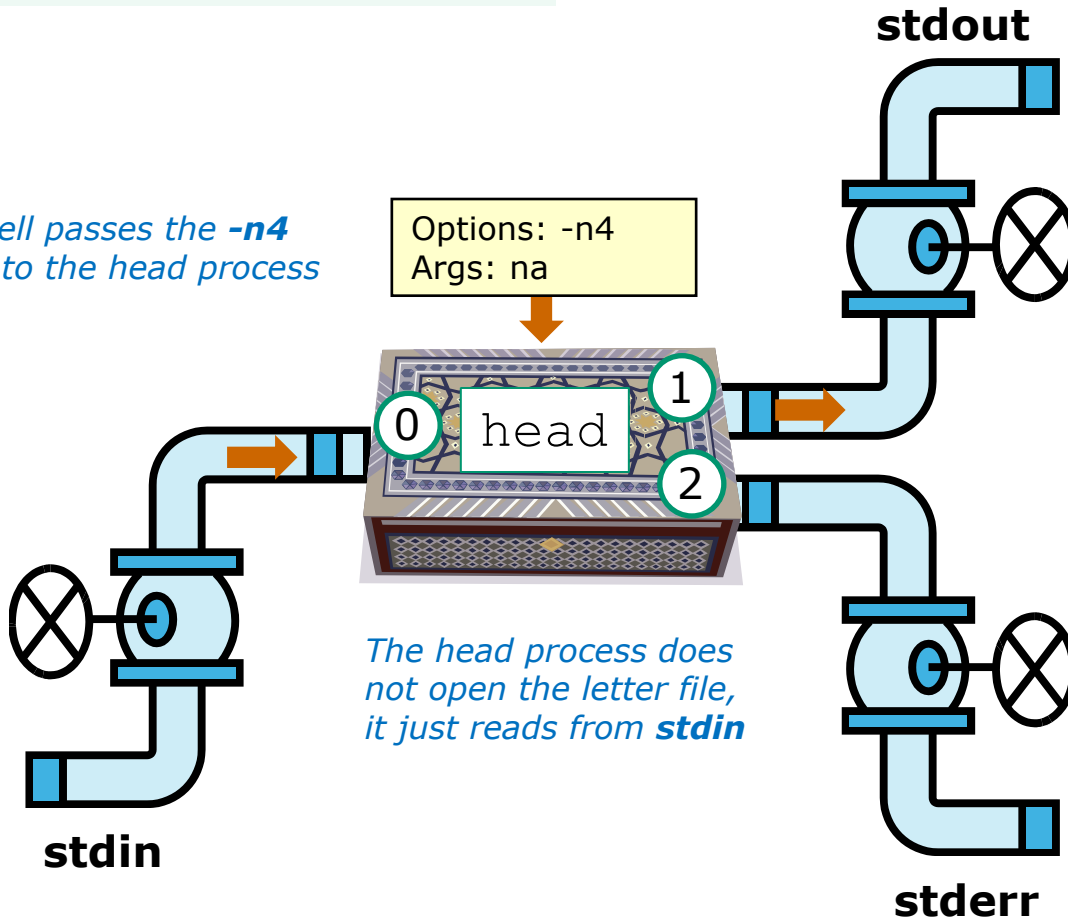
Here I am at Camp Granada. Things are very entertaining,
and they say we'll have some fun when it stops raining.

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**





Errors

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

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

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

```
[rsimms@oslab ~]$ cat bogus  
cat: bogus: No such file or directory
```

```
[rsimms@oslab ~]$ bogus  
-bash: bogus: command not found
```

```
[rsimms@oslab ~]$ cat < bogus  
-bash: bogus: No such file or directory
```

```
[rsimms@oslab ~]$ bogus < bogus  
-bash: bogus: No such file or directory
```

1) Prompt

2) Parse

3) Search

4) Execute

5) Nap

6) Repeat

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

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

```
[rsimms@oslab ~]$ cat bogus
```

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

```
[rsimms@oslab ~]$ bogus
```

```
-bash: bogus: command not found
```

```
[rsimms@oslab ~]$ cat < bogus
```

```
-bash: bogus: No such file or directory
```

```
[rsimms@oslab ~]$ bogus < bogus
```

```
-bash: bogus: No such file or directory
```

1) Prompt

2) Parse

3) Search

4) Execute

5) Nap

6) Repeat

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
```



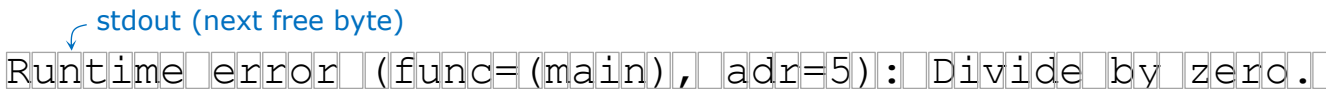
stderr (next free byte)



2+2

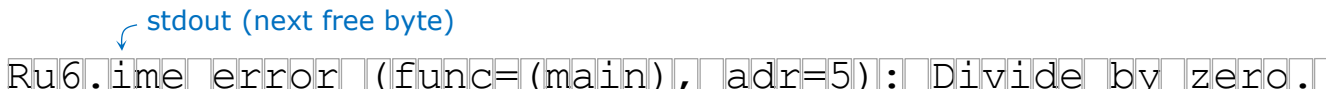
stderr (next free byte)

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);
    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



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

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

Running the simple program.

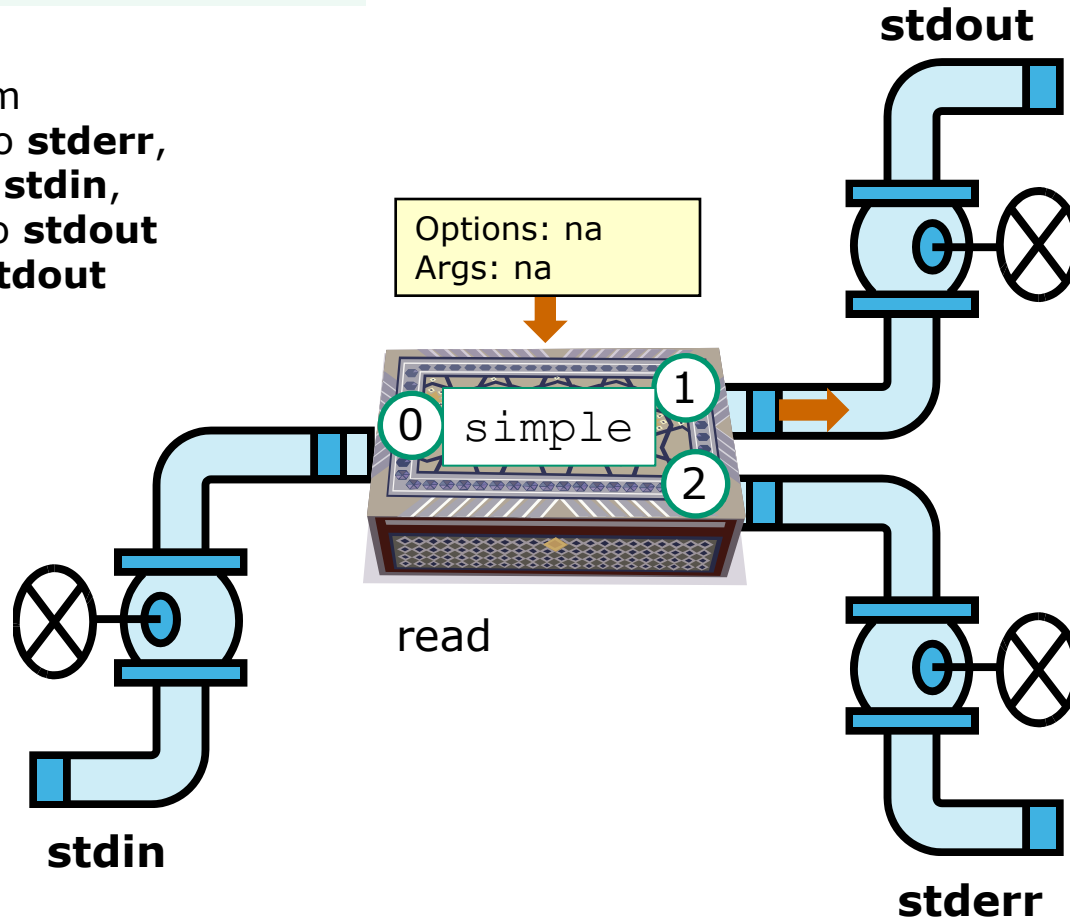
*Note I need to preface **simple** with a "./" to run it as this directory is not on my path. This is not necessary for CIS 90 students as they already have the . directory in their path.*

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
```

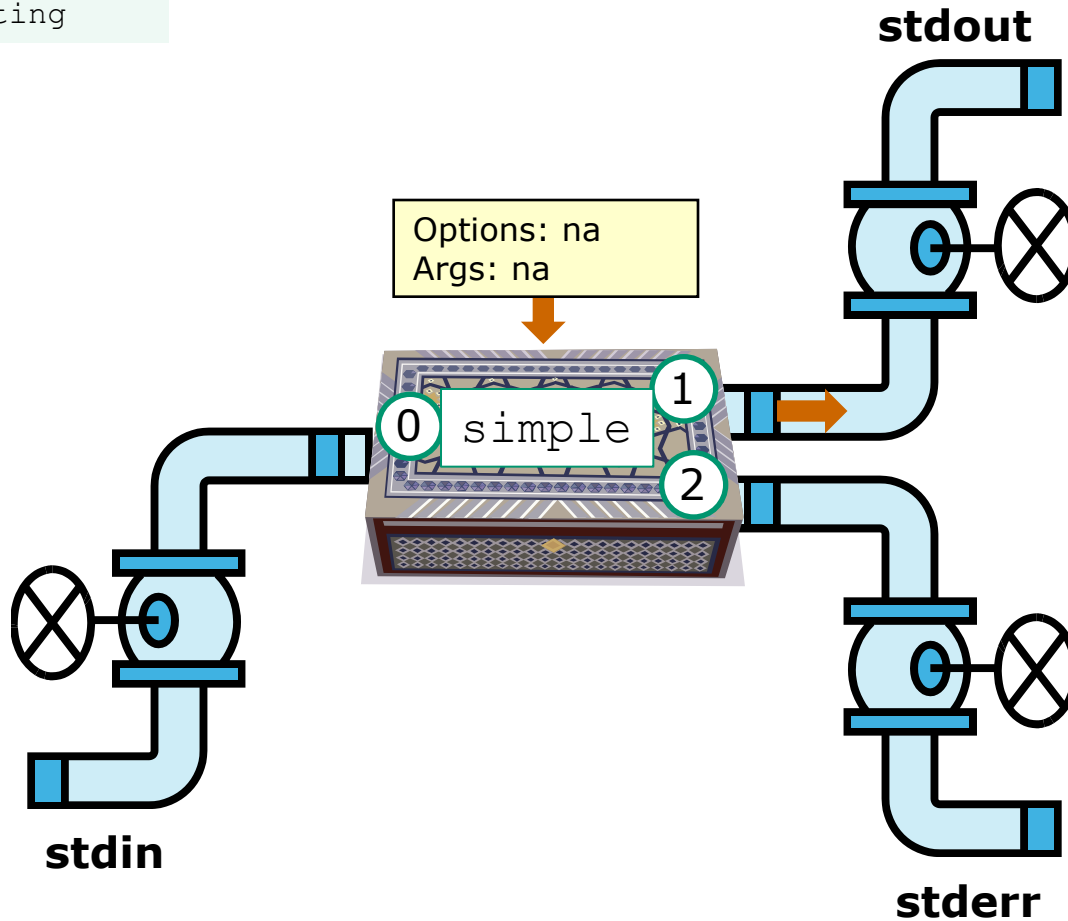
In this example, output has been redirected to a file named myfile.

*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

umask

(review)

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	<pre>/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</pre>
002	000	000	010	
	↓	↓	↓	
664	110	110	100	

New directory - start with 777 and apply mask

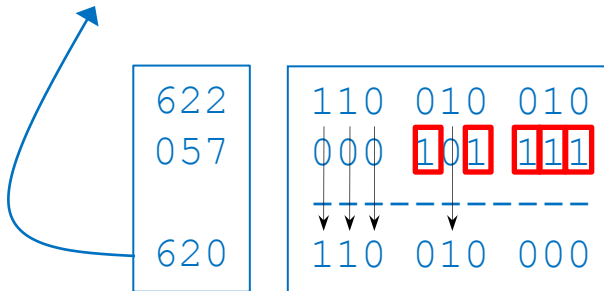
777	111	111	111	<pre>/home/cis90/simben/lesson9 \$ mkdir newdir /home/cis90/simben/lesson9 \$ ls -ld newdir drwxrwxr-x 2 simben cis90 4096 Oct 27 07:23 newdir</pre>
002	000	000	010	
	↓	↓	↓	
775	111	111	101	

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.*

Activity

- 1) I want to change the permissions on an existing file
- 2) I want to restrict specific permissions on files that have not been created yet
- 3) I want to show the permissions on a file in mnemonic format e.g. `rwxr-x---`
- 4) I want to show the permissions on a file in numeric format e.g. `750`

CCC Confer Poll

- A) `stat`
- B) `ls -l`
- C) `chmod`
- D) `umask`

Enter the appropriate command using the CCC Confer Poll

Activity

- 1) I want to change the permissions on an existing file
- 2) I want to restrict specific permissions on files that have not been created yet
- 3) I want to show the permissions on a file in mnemonic format e.g. `rwxr-x---`
- 4) I want to show the permissions on a file in numeric format e.g. `750`

CCC Confer Poll

- A) `stat`
- B) `ls -l`
- C) `chmod`
- D) `umask`

Enter the appropriate command using the CCC Confer Poll

Activity

- 1) I want to change the permissions on an existing file
- 2) I want to restrict specific permissions on files that have not been created yet
- 3) I want to show the permissions on a file in mnemonic format
e.g. `rwxr-x---`
- 4) I want to show the permissions on a file in numeric format
e.g. `750`

CCC Confer Poll

- A) `stat`
- B) `ls -l`
- C) `chmod`
- D) `umask`

Enter the appropriate command using the CCC Confer Poll

Activity

- 1) I want to change the permissions on an existing file
- 2) I want to restrict specific permissions on files that have not been created yet
- 3) I want to show the permissions on a file in mnemonic format e.g. `rwxr-x---`
- 4) I want to show the permissions on a file in numeric format e.g. `750`

CCC Confer Poll

- A) `stat`
- B) `ls -l`
- C) `chmod`
- D) `umask`

Enter the appropriate command using the CCC Confer Poll



Activity

1) I want to change the permissions on an existing file

2) I want to restrict specific permissions on files that have not been created yet

3) I want to show the permissions on a file in mnemonic format e.g. `rwxr-x---`

4) I want to show the permissions on a file in numeric format e.g. `750`

CCC Confer Poll

A) `stat`

B) `ls -l`

C) `chmod`

D) `umask`

Enter the appropriate command using the CCC Confer Poll



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
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!

Another example

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

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

xargs to the rescue again!

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

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

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

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

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

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

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) 1993 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/arch/x86/include/asm/delay.h: * Copyright (C) 1991, 1992 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, Treats and Tricks



trick or treat

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

1. Can you find them? There should be an obvious one in your home directory. The rest are scattered in the various subdirectories you own.
2. Make a new directory named *bag* in your home directory and see how many *trick* or *treat* files you can move into it.
3. Put a Green Check in CCC Confer next to your name when you have collected 3 treats, electronically "clap" if you collect all six treats and six tricks.

Review

Jim's Summary Pages

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

Lesson 6 - Managing Files

<http://cabrillo.edu/~jgriffin/CIS90/files/lecture5.html>

Lesson 7 - File Permissions

<http://cabrillo.edu/~jgriffin/CIS90/files/lecture6.html>

Lesson 8 - Input/Output Processing

<http://cabrillo.edu/~jgriffin/CIS90/files/lecture7.html>



Make Teams

Breakout Rooms



debian

Room 1



redhat

Room 2



suse

Room 3



CentOS

Room 4



ubuntu

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

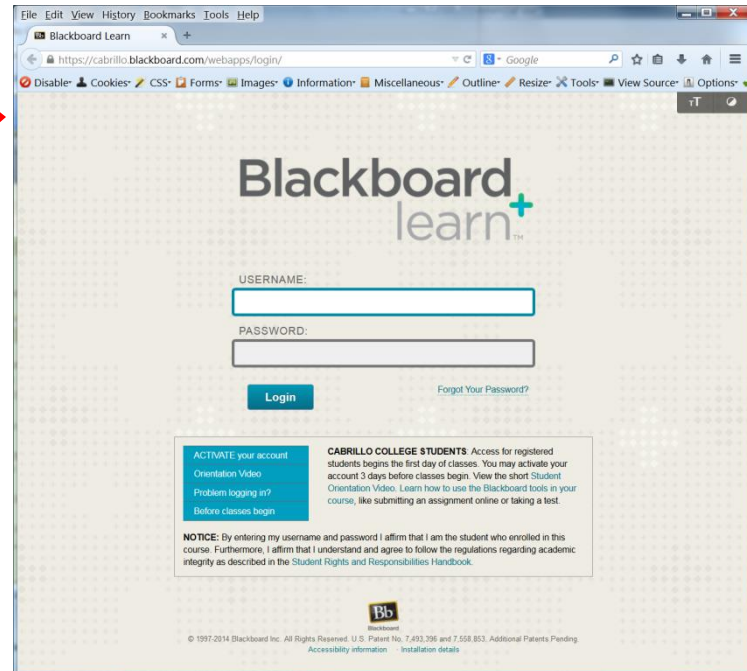
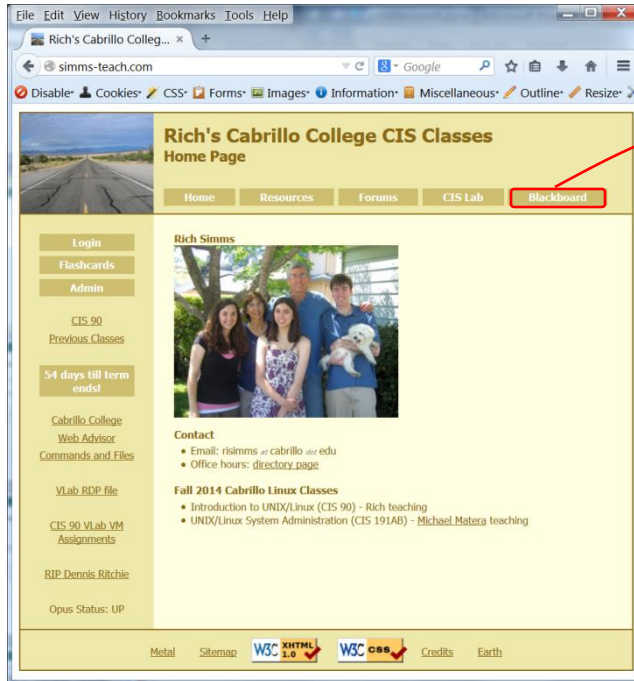
- Chat window belongs to team that is up
- Team gets the point if anyone on the team writes a correct answer in the chat window in 15 seconds

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



Practice test available

- Available on Blackboard
- 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.

Practice Test Honor Code and Instructions

Description Practice Test 2 (Fall 2014)

Instructions **HONOR CODE:**

This is a practice test and you may work with others on it. Feel free to compare and discuss answers to the practice test on the forum. However on the real test you must work alone.

INSTRUCTIONS:

Test system: sun-hwa-p2.cis.cabrillo.edu (port 22)

This test should be completed using the sun-hwa-p2 system only. Log into Opus first then ssh into sun-hwa-p2. The sun-hwa-p2 system will no longer be available once the real test starts.

Please KEEP YOUR ANSWERS TO A SINGLE LINE ONLY !!

Total
Questions 33

Total
Points 30

Make sure you can log into the testing server. Login consulting is free on the practice test.

On the real test though you can "purchase" login consulting from the instructor using some of your points!

Wrap up

Next Class

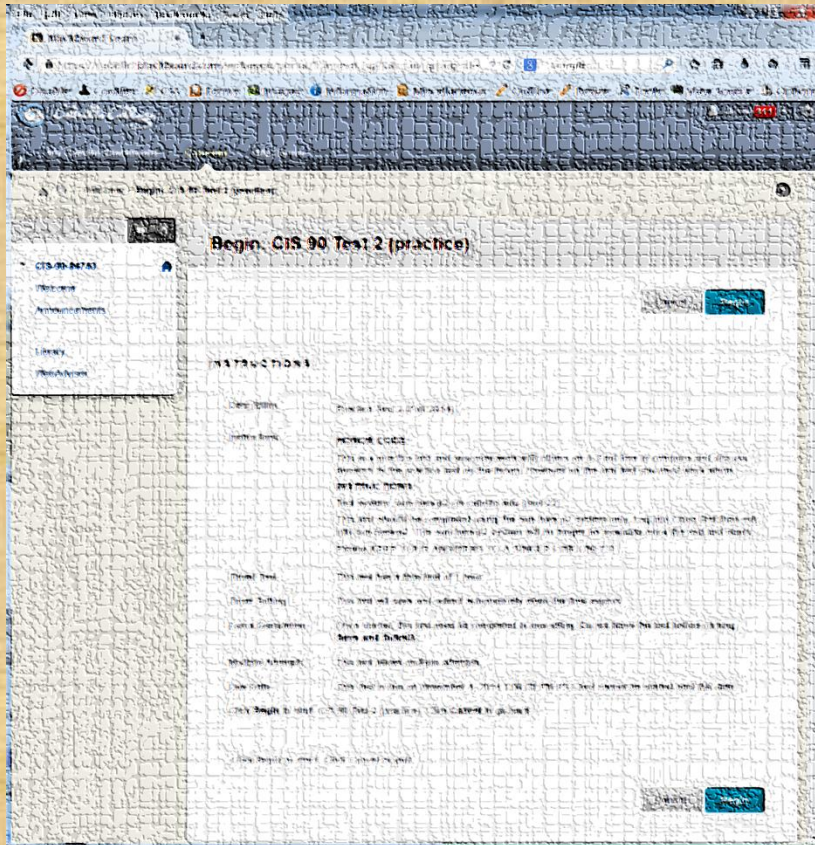
No Quiz

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 and into office hours after class.

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

Backup