



Rich's 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 at 3PM
- P2 Test system online and unlocked at 3PM

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



Student Learner Outcomes

1. Navigate and manage the UNIX/Linux file system by viewing, copying, moving, renaming, creating, and removing files and directories.
2. Use the UNIX features of file redirection and pipelines to control the flow of data to and from various commands.
3. With the aid of online manual pages, execute UNIX system commands from either a keyboard or a shell script using correct command syntax.

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/>)



Student checklist for laying out screen when attending class

- Browse to the CIS 90 website Calendar page
 1. <http://simms-teach.com>
 2. Click CIS 90 link on left panel
 3. Click Calendar link near top of content area
 4. Locate today's lesson on the Calendar

- Download the presentation slides for today's lesson for easier viewing

- Click Enter virtual classroom to join CCC Confer session

- Connect to Opus using Putty or ssh command



Student checklist for laying out screen when attending class

Google

CCC Confer

Downloaded PDF of Lesson Slides

The screenshot shows a virtual classroom interface with several overlapping windows:

- Blackboard Course Page:** Displays 'Rich's Cabrillo College CIS 90 Calendar' with a sidebar containing navigation links like 'Login', 'Flashcards', 'Admin', and 'CIS 90 (Spring) Course Home'.
- CCC Confer Virtual Classroom:** The main window showing a video feed of 'Rich Simms' and a 'PARTICIPANTS' list with 'Rich Simms' and 'Benji Simms'. A 'CHAT' window shows a conversation about textbooks.
- Google Maps:** A window titled 'Where are you now?' showing a map of the San Francisco Bay Area.
- Adobe Acrobat Pro:** A window titled 'cis90lesson01.pdf' showing a slide titled 'The CIS 90 System Playground' with a diagram of server racks.
- Terminal Window:** A window showing a login prompt for 'Opus' with a password field and a 'Welcome to Opus' message.

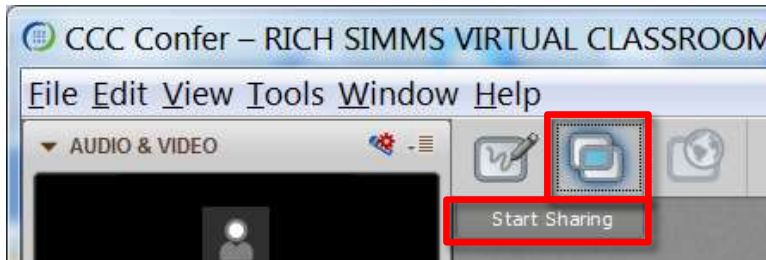
CIS 90 website Calendar page

One or more login sessions to Opus

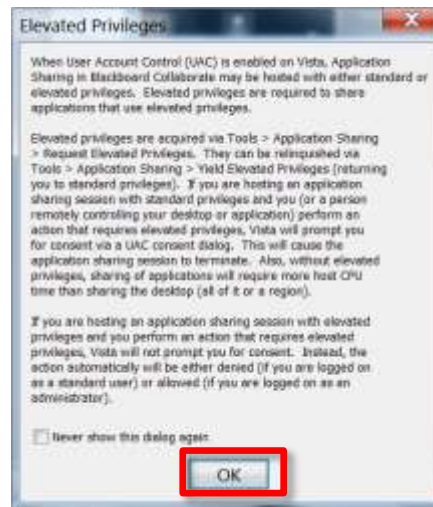


Student checklist for sharing desktop with classmates

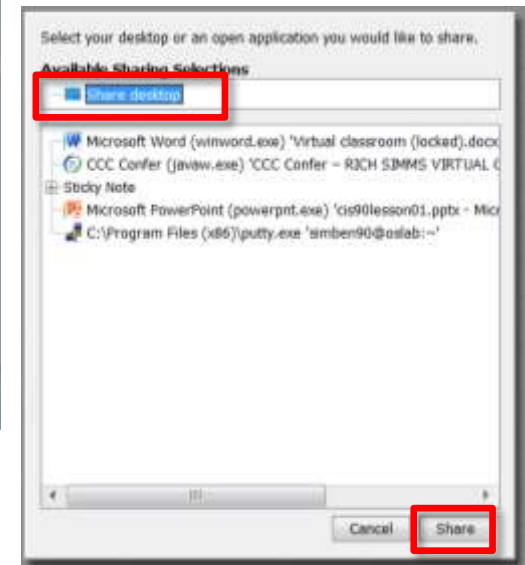
1) Instructor gives you sharing privileges



2) Click overlapping rectangles icon. If white "Start Sharing" text is present then click it as well.



3) Click OK button.



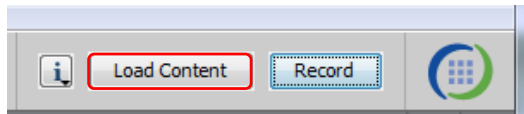
4) Select "Share desktop" and click Share button.



Rich's CCC Confer checklist - setup

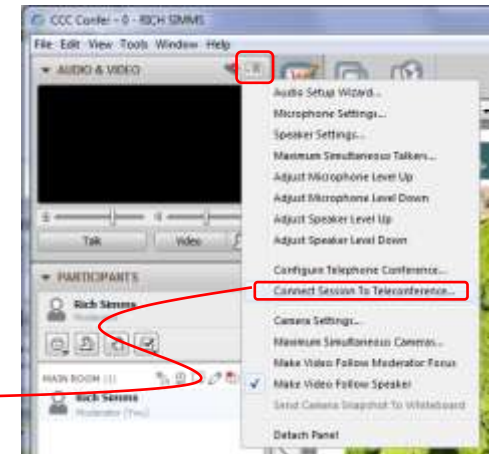
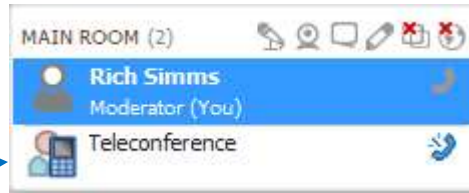


[] Preload White Board

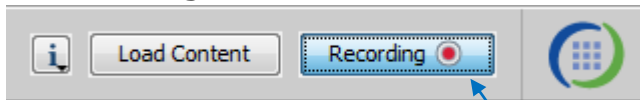


[] Connect session to Teleconference

Session now connected to teleconference



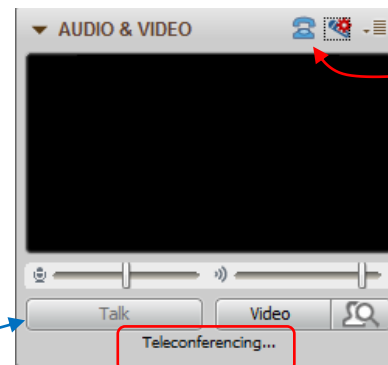
[] Is recording on?



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



Rich's CCC Confer checklist - screen layout and share



foxit for slides

chrome

vSphere Client

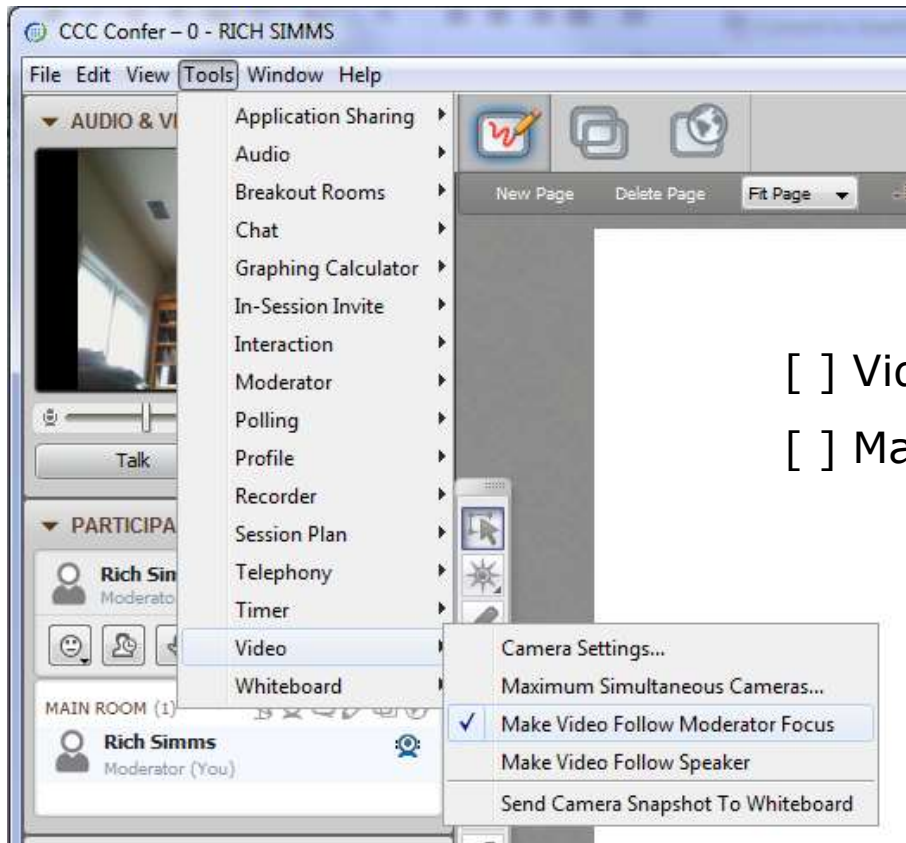
putty

[] layout and share apps





Rich's CCC Confer checklist - webcam setup



- [] Video (webcam)
- [] Make Video Follow Moderator Focus



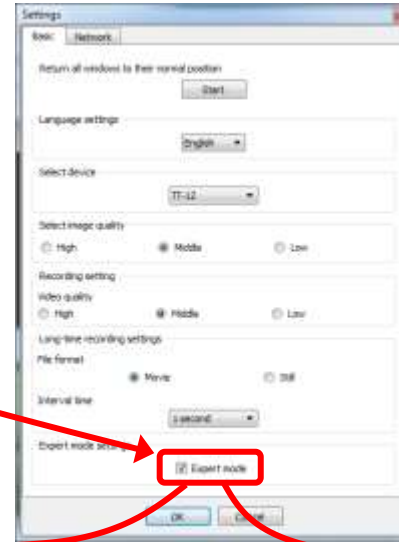
Rich's CCC Confer checklist - Elmo



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





Rich's CCC Confer checklist - universal fix

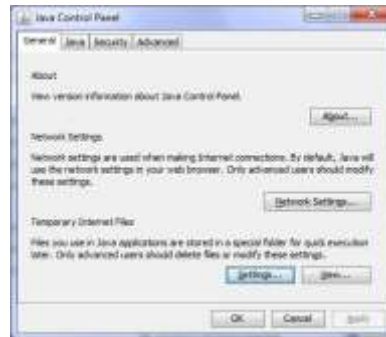
Universal Fix for CCC Confer:

- 1) Shrink (500 MB) and delete Java cache
- 2) Uninstall and reinstall latest Java runtime
- 3) <http://www.cccconfer.org/support/technicalSupport.aspx>

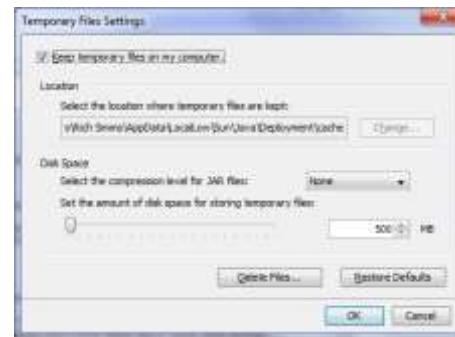
Control Panel (small icons)



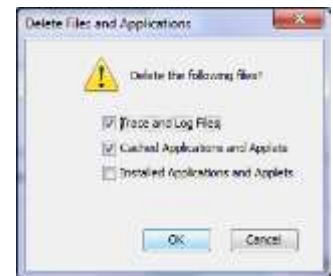
General Tab > Settings...



500MB cache size



Delete these



Google Java download





Start

Sound Check

*Students that dial-in should mute their line using *6 to prevent unintended noises distracting the web conference.*

*Instructor can use *96 to mute all student lines.*



Instructor: **Rich Simms**

Dial-in: **888-886-3951**

Passcode: **136690**



Chris



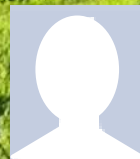
Jeremy



Jennifer



Cameron



Joseph



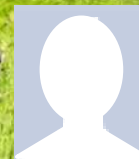
Lisa



May



Sundance



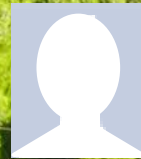
Charlie



Sean



Brenda



Anthony



Will H.



Josh



Michael



Danny



Vic



William D.



Taylor



Thomas



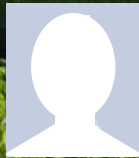
Stewart



Miguel



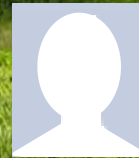
Akasha



Jairo



Tony



Joaquin

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

First Minute Quiz

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

Use CCC Confer White Board

email answers to: risimms@cabrillo.edu

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

Review

Objectives	Agenda
<ul style="list-style-type: none">• Get ready for the next test• Practice skills• Introduction to processes	<ul style="list-style-type: none">• Quiz• Questions• Housekeeping• Linux at school• Linux at home• More on I/O• All together now• Subtle differences• Errors• 2>&1• More on I/O - programming• umask• More pipeline practice• Pipeline and redirection practice• More on pipelines• Eggs, treats and tricks• Review• Make teams• Flashcard practice• Assignment• Wrap up



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.

Would you like some help learning Linux?



If you would like some additional come over to the CIS Lab. There are student lab assistants and instructors there to help you.

*Tess, Michael, and Paul are
CIS 90 Alumni.*

*Mike Matera is the other
Linux instructor.*

I'm in there Mondays.



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. Fine Print:
Test #2 is scheduled for our next class!

Test #2 is next week

Practice test available at 3PM today



**Test #2 is next
week**

**Practice test
available at 3PM
today**



Test #2 is next week

**Practice test available at
3PM today**

Test #2

1. Test #2 is **scheduled for our next class!** Same format as before. The test will start during the last hour of class. If you work you can take it later in the day as long as it is completed by 11:59PM.
2. Blackboard Practice Test #2 is available at 3PM today. It will not be available when the real test starts.
3. Practice Test system is available at 3PM today. It will not be available when the real test starts.
4. No lab assigned this week (so you can work on the practice test)
5. Ask your questions regarding the test on the forum **BEFORE** the next class starts!

Use the forum to discuss practice test questions

- Post if you get stuck on a question
- Post to share tips with others
- Post if you would like a clarification on a test question
- Post your answer to a question to discuss and get feedback from others

Use the forum to arrange study groups

Practice Test 2 Study Group

Locked Search this topic...

3 posts • Page 1 of 1

Practice Test 2 Study Group

by Tess Pritchard • Wed Apr 01, 2015 11:59 am

I know it's late notice, but Mario and I are going to start working through the practice test tomorrow.
Thursdays 1pm in the CIS Lab.
We'd love it if you could join!

Thanks,
Tess

Tess Pritchard

Posts: 30
Joined: Wed Sep 10, 2014
2:15 pm

Example Fall 2014 post to meet in the STEM center to study for a CIS 90 test

online study group

by Benjamin Correia • Mon Mar 02, 2015 4:22 pm

I wasn't able to make the study group on campus today due to work so I thought I'd see if anyone would be willing to meet up online through Skype, hangouts or some other online collaborative work space like Docs...

let me know if your interested in spending an hour or two studying tomorrow morning from 9am to 11, I have work from 12pm until around 10:30 so if people would rather work a night I could devote an hour or so after that time.

if you have any ideas for a online study group pleas feel free to add your thoughts, I am having a bit of trouble really understanding a few things we covered in class so I thought I'd ask my peers for some advice if possible.

-Benji

Benjamin Correia

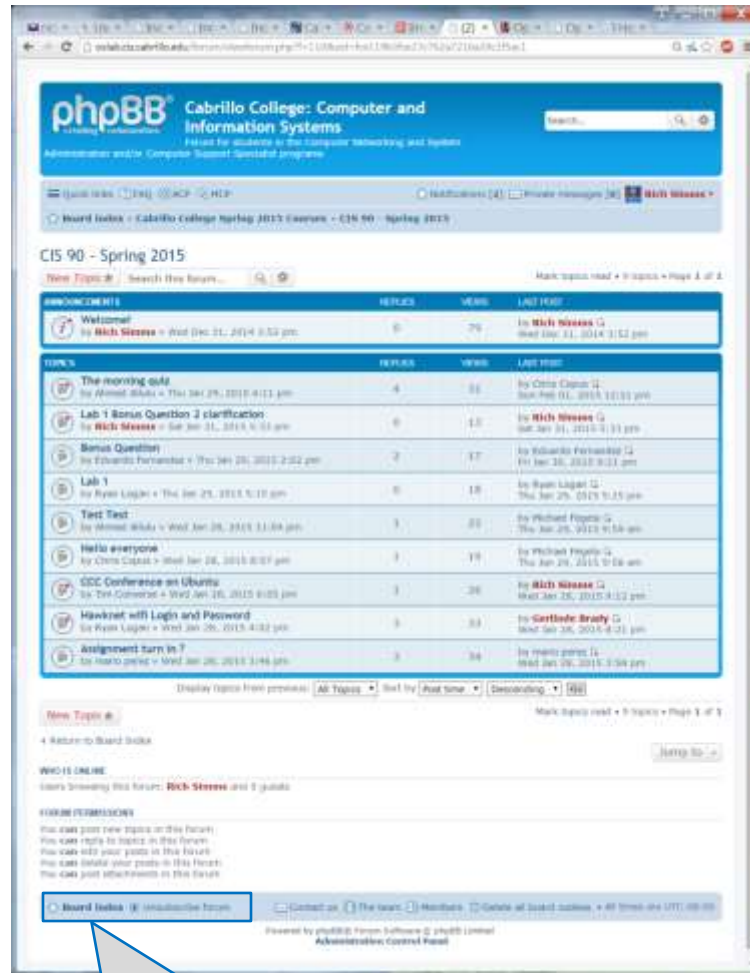
Posts: 28
Joined: Tue Feb 03, 2015
11:11 am

Example Fall 2014 post to meet online to study for a CIS 90 test

Don't miss replies to your forum posts

2) Go to the CIS 90 forum

1) Login to the forum



3) Click the "Subscribe" link at the bottom so that it changes to "Unsubscribe".



Heads up on Final Exam

Test #3 (final exam) is **MONDAY** Dec 14 1-3:50PM

Monday	12/14	Test #3 (the final exam)	5 posts Lab X1 Lab X2
		<p>Time</p> <ul style="list-style-type: none"> MONDAY 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 	

*Extra credit
labs and
final posts
due by
11:59PM*

- All students will take the test at the same time. The test must be completed by 3:50PM.
- Working and long distance students can take the test online via CCC Confer and BlackBoard.
- Working students will need to plan ahead to take time off from work for the test.

FINAL EXAMINATIONS SCHEDULE: FALL 2015: December 14-19

DAYTIME CLASSES: All times in bold refer to the beginning times of classes. **MW/Daily** means Monday alone, Wednesday alone, Monday and Wednesday or any 3 or more days in any combination. **TTh** means Tuesday alone, Thursday alone, or Tuesday and Thursday. **Classes meeting other combinations of days and/or hours not listed must have a final schedule approved by their Division Dean.**

Classes starting between:

6:30 am and 8:55 am, MW/Daily	7:00 am-9:50 am	Monday, Dec. 14
9:00 am and 10:15 am, MW/Daily	7:00 am-9:50 am	Wednesday, Dec. 16
10:20 am and 11:35 am, MW/Daily	10:00 am-12:50 pm	Monday, Dec. 14
11:40 am and 12:55 pm, MW/Daily	10:00 am-12:50 pm	Wednesday, Dec. 16
1:00 pm and 2:15 pm, MW/Daily	1:00 pm-3:50 pm	Monday, Dec. 14
2:20 pm and 3:35 pm, MW/Daily	1:00 pm-3:50 pm	Wednesday, Dec. 16
3:40 pm and 5:30 pm, MW/Daily	4:00 pm-6:50 pm	Wednesday, Dec. 16
6:30 am and 8:55 am, TTh	7:00 am-9:50 am	Tuesday, Dec. 15
9:00 am and 10:15 am, TTh	7:00 am-9:50 am	Thursday, Dec. 17
10:20 am and 11:35 am, TTh	10:00 am-12:50 pm	Tuesday, Dec. 15
11:40 am and 12:55 pm, TTh	10:00 am-12:50 pm	Thursday, Dec. 17
1:00 pm and 2:15 pm, TTh	1:00 pm-3:50 pm	Tuesday, Dec. 15
2:20 pm and 3:35 pm, TTh	1:00 pm-3:50 pm	Thursday, Dec. 17
3:40 pm and 5:30 pm, TTh	4:00 pm-6:50 pm	Thursday, Dec. 17
Friday am	9:00 am-11:50 am	Friday, Dec. 18
Friday pm	1:00 pm-3:50 pm	Friday, Dec. 18
Saturday am	9:00 am-11:50 am	Saturday, Dec. 19
Saturday pm	1:00 pm-3:50 pm	Saturday, Dec. 19

Evening Classes: For the final exam schedule, **Evening Classes** are those that begin at 5:35 pm or later. Also, **"M & W"** means the class meets on **BOTH** Monday and Wednesday. **"T & TH"** means the class meets on **BOTH** Tuesday and Thursday. The following schedule applies to all Evening Classes.

EVENING FINAL SCHEDULE:

Classes beginning at 5:35 pm or later on:

Monday Only OR "M & W"	6:00 pm-8:50 pm	Monday, Dec. 14
Tuesday Only OR "T & TH"	6:00 pm-8:50 pm	Tuesday, Dec. 15
Wednesday Only	7:00 pm-9:50 pm	Wednesday, Dec. 16
Thursday Only	7:00 pm-9:50 pm	Thursday, Dec. 17
Friday Only	6:00 pm-8:50 pm	Friday, Dec. 18

CIS 90 Introduction to UNIX/Linux

Provides a technical overview of the UNIX/Linux operating system, including hands-on experience with commands, files, and tools. Prerequisite: CIS 72. Transfer Credit: CSU.

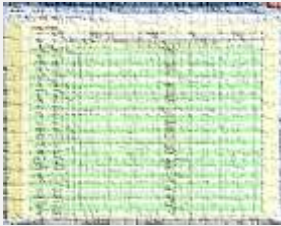
Section	Days	Times	Units	Instructor	Room
89005	W	01:00PM-04:05PM	3.00	R. Simms	OL
	& Arr.	Arr.		R. Simms	OL
Section 89005 is an ONLINE course. Meets weekly throughout the semester online during the scheduled times by remote technology with an additional 50 min online lab per week. For details, see instructor's web page at go.cabrillo.edu/online .					
89006	W	01:00PM-04:05PM	3.00	R. Simms	828
	& Arr.	Arr.		R. Simms	OL
Section 89006 is a Hybrid ONLINE course. Meets weekly throughout the semester at the scheduled times with an additional 50 min online lab per week. For details, see instructor's web page at go.cabrillo.edu/online .					

Where to find your grades

Send me your survey to get your LOR code name.

The CIS 90 website Grades page

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



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

At the end of the term I'll add up all your points and assign you a grade using this table

Or check on Opus

checkgrades *codename*

(where *codename* is your LOR codename)



Written by Jesse Warren a past CIS 90 Alumnus

grades *codename*

(where *codename* is your LOR codename)



Written by Sam Tindell a past CIS 90 Alumnus.
Try his tips, schedule and forums scripts as well!

LPI Linux Essentials Certificate

Linux Essentials Certificate of Achievement				
Objective	# of Questions	Cabrillo	Urban Penguin	NDG Linux Essentials
Topic 1: The Linux Community and a Career in Open Source				
1.1 Linux Evolution and Popular Operating Systems	2	CIS90 Lesson 1	1.1	Module 1
1.2 Major Open Source Applications	2	CIS90 Lesson 1	1.2	Module 2
1.3 Understanding Open Source Software and Licensing	1	CIS90 Lesson 1	1.3	Module 2
1.4 ICT Skills and Working in Linux	2	not covered	1.4	Module 3
Topic 2: Finding Your Way on a Linux System				
2.1 Command Line Basics	2	CIS90 Lesson 2	2.1	Module 4
2.2 Using the Command Line to Get Help	2	CIS90 Lesson 2	2.2	Module 5
2.3 Using Directories and Listing Files	2	CIS 90 Lesson 4	2.3	Module 6
2.4 Creating, Moving and Deleting Files	2	CIS90 Lesson 5	2.4	Module 6
Topic 3: The Power of the Command Line				
3.1 Archiving Files on the Command Line	2	CIS 90 Lesson 14	3.1	Module 7
3.2 Searching and Extracting Data from Files	4	CIS 90 Lesson 8	3.2	Module 8
3.3 Turning Commands into a Script	4	CIS 90 Lesson 13 & 14	3.3	Module 9
Topic 4: The Linux Operating System				
4.1 Choosing an Operating System	1	not covered	4.1	Module 1
4.2 Understanding Computer Hardware	2	CIS 90 Lesson 1	4.2	Module 10
4.3 Where Data is Stored	3	CIS 90 Lesson 1	4.3	Module 11
4.4 Your Computer on the Network	2	CIS 192	4.4	Module 12
Topic 5: Security and File Permissions				
5.1 Basic Security and Identifying User Types	2	CIS 191	5.1	Module 13
5.2 Creating Users and Groups	2	CIS 191	5.2	Module 14
5.3 Managing File Permissions and Ownership	2	CIS 90 Lesson 7	5.3	Module 15
5.4 Special Directories and Files	1	CIS 90 Lesson 4	5.4	Module 16

The Urban Penguin

LINUX ESSENTIALS

Home LPI

Welcome to this self study video series of tutorials. These videos can be used in preparing you for the LPI Linux Professional Institute, Linux Essentials, Certification. These materials are meant as a stand alone learning solution in readiness for your exam and are targeted towards anyone who is aiming for the certification or just wants to know more about what Linux is and what it can offer. The Urban Penguin is an **Approved LPI Training Partner** and we provide both free training via these videos and, if you prefer to assist direct with the penguin, then we can offer **video training** at a reasonable cost.

Objective	Description	Click to Access
000	What is LPI Linux Essentials	Click to Access
1.1	Linux evolution and popular operating systems	Click to Access
1.2	Major Open Source applications	Click to Access
1.3	Understanding Open Source Software and licensing	Click to Access
1.4	EC2, VMs and working with Linux	Click to Access
2.1	Command line basics	Click to Access
2.2	Using the command line to get help	Click to Access
2.3	Using directories and listing files	Click to Access
2.4	Creating, moving and deleting	Click to Access
3.1	Archiving files from the command line	Click to Access
3.2	Backup and restoring data from files	Click to Access
3.3	Turning commands into a script	Click to Access
4.1	Choosing an operating system	Click to Access
4.2	Understanding computer hardware	Click to Access
4.3	Where data is stored	Click to Access
4.4	Your computer on the network	Click to Access
5.1	Basic security and user types	Click to Access
5.2	Creating users and groups	Click to Access
5.3	Managing file permissions and ownership	Click to Access
5.4	Special directories and files	Click to Access

Inductive and live video based Linux Training

<http://www.theurbanpenguin.com/lpi/le.html>

No registration, no logging in, just click and watch the videos

NDG Linux Essentials via Cisco Networking Academy

NDG Linux Essentials

2.3 Major Open Source Applications

The Linux kernel is not a wide variety of software services (many hardware platforms, a computer can act as a server, which means it primarily handles data on other's behalf, or can act as a desktop), which means a user will be interacting with it directly. The machine can run software or it can be used as a development machine in the process of creating software. You can even use multiple roles as there is no restriction to Linux about the use of the machine, it's mainly a matter of configuring which applications run.

One advantage of this is that you can simulate almost all aspects of a production environment from development to testing, to deployment on actual servers hardware, which saves costs and time. An awesome training Linux, you can run the same server applications on your desktop or responsive virtual server that you run on a large external Service Provider. Of course, you will not be able to handle the volume a large provider results, as they will have much more expensive hardware, but you can simulate almost any configuration without needing powerful hardware or server licensing.

Linux software generally falls into one of three categories:

- **Server software** - software that has no direct interaction with the monitor and keyboard of the machine it runs on. Its purpose is to serve information to other computers, called **clients**. Sometimes server software may not talk to other computers but will get all these and "store" it data.
- **Desktop software** - a web browser, text editor, music player, or other software that you interact with. In many cases, such as a web browser, the software is talking to a server on the other end and downloading the data for you. Hence, the desktop software is the client.
- **Tools** - a broad category of software that exists to assist a user to manage your system. You might have a tool that helps you configure your display, or something that provides a Linux shell, or even more sophisticated tools that convert entire code to something that the computer can read etc.

Additional, we will create virtual environments, similar to the kernel of the LPI.

Test, Export, Zip

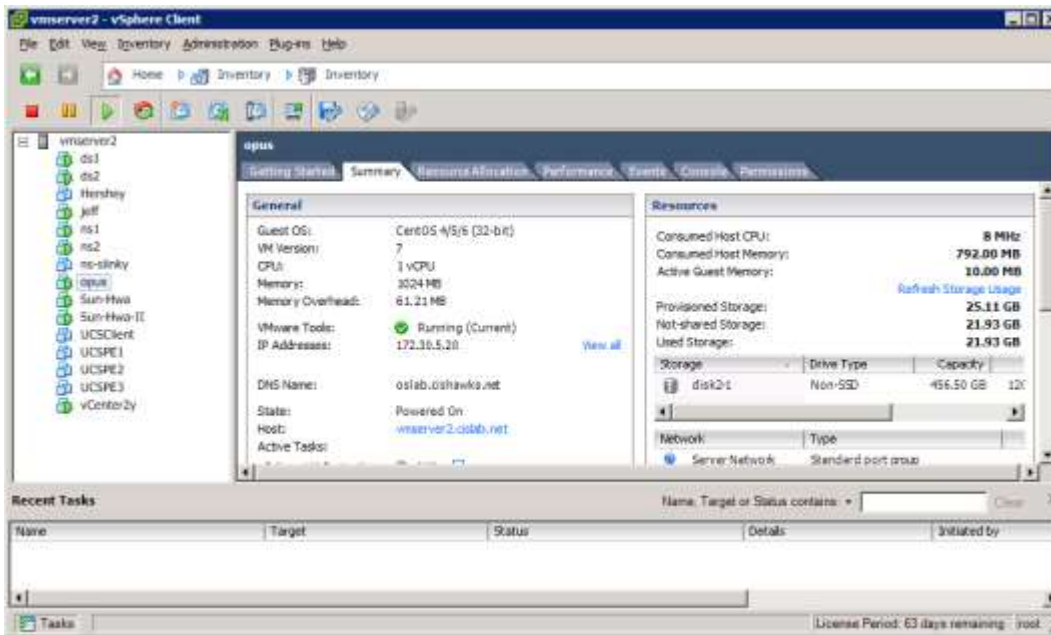
<https://www.netacad.com/>

Complete course with reading, live VM and tests. Contact me if you would like a student account for the NDG Linux Essentials course.



Linux at School

Our Opus server on campus



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)



Linux at Home

USB "Live" Linux Multi-Boot USB Flash Drive

Windows



CentOS



Kali



YUMI formatted Flash Drive
(www.pendrivelinux.com)

Linux Mint



Ubuntu



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 distros of your choice
See: <http://iso.linuxquestions.org/>

2)



Get a USB flash drive

Google "boot live linux from usb" for instructions

3)

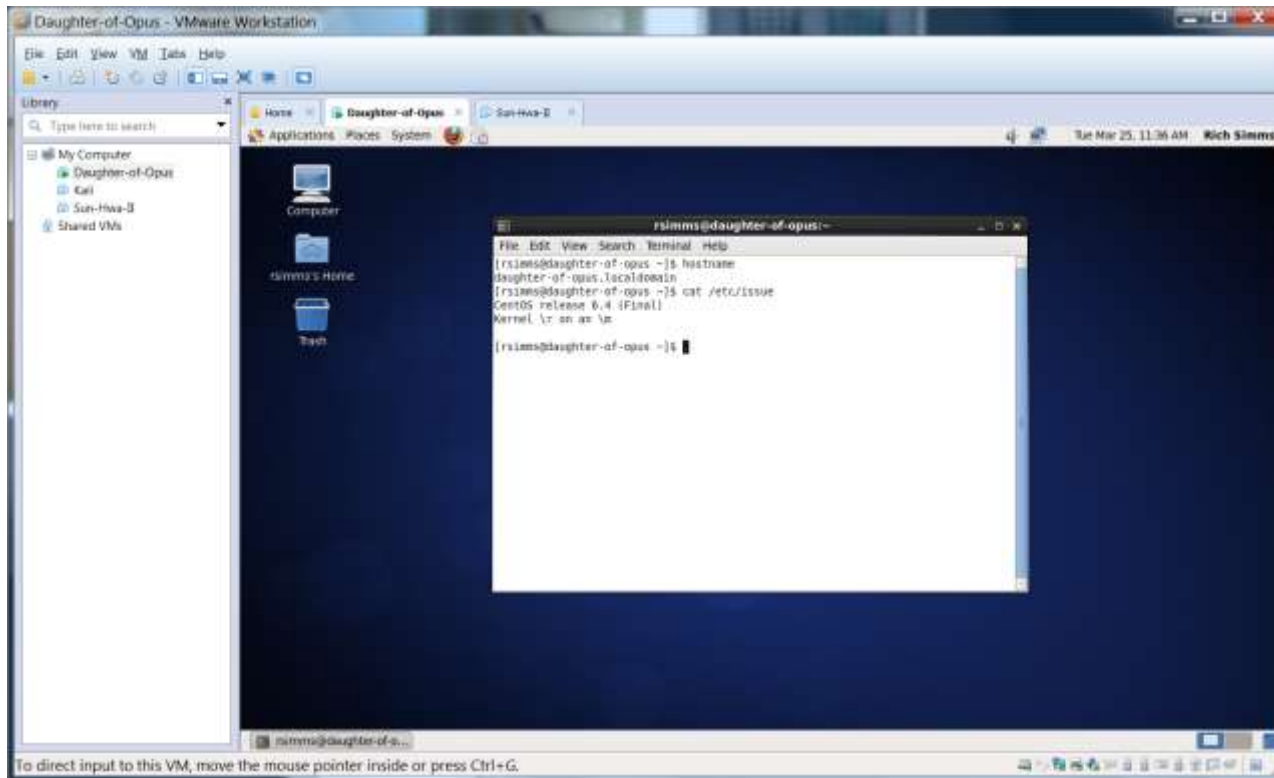
or see

<http://www.pendrivelinux.com/yumi-multiboot-usb-creator/>

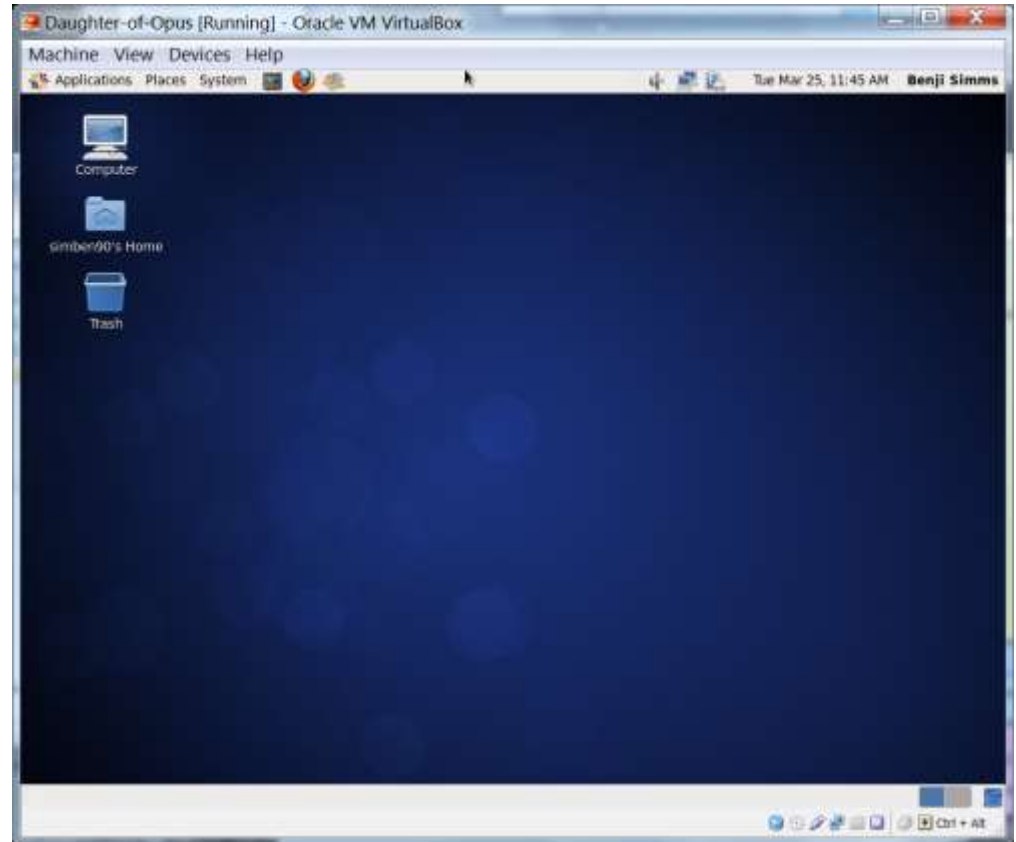
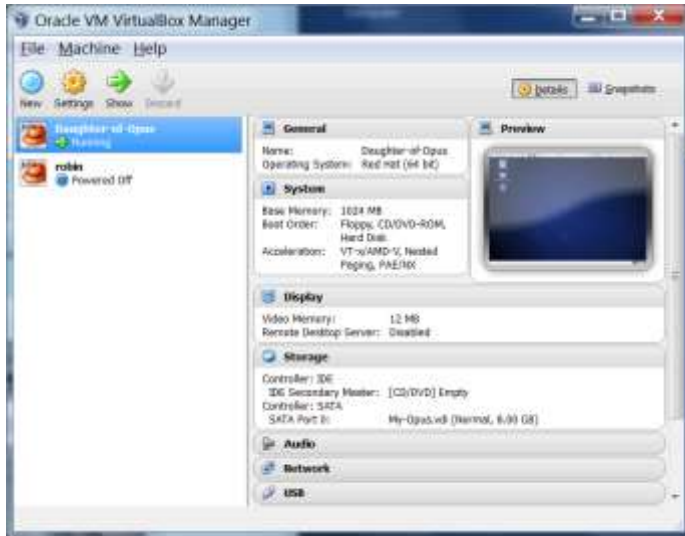
4)



Configure your BIOS to boot from USB then select the Operating System as your computer boots up

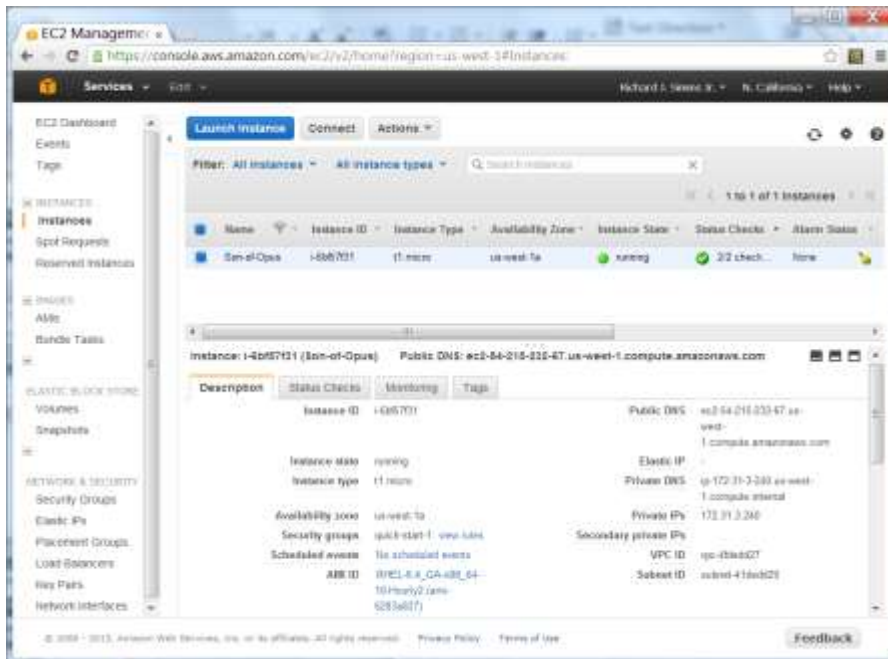


One Daughter-of-Opus is a VM running on my laptop using VMware Workstation (expires in one year)



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 -p 2220 <username>@son-of-opus.simms-teach.com`

Small Form Factor Servers

HP Microserver



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

uLab

A very tiny home made datacenter

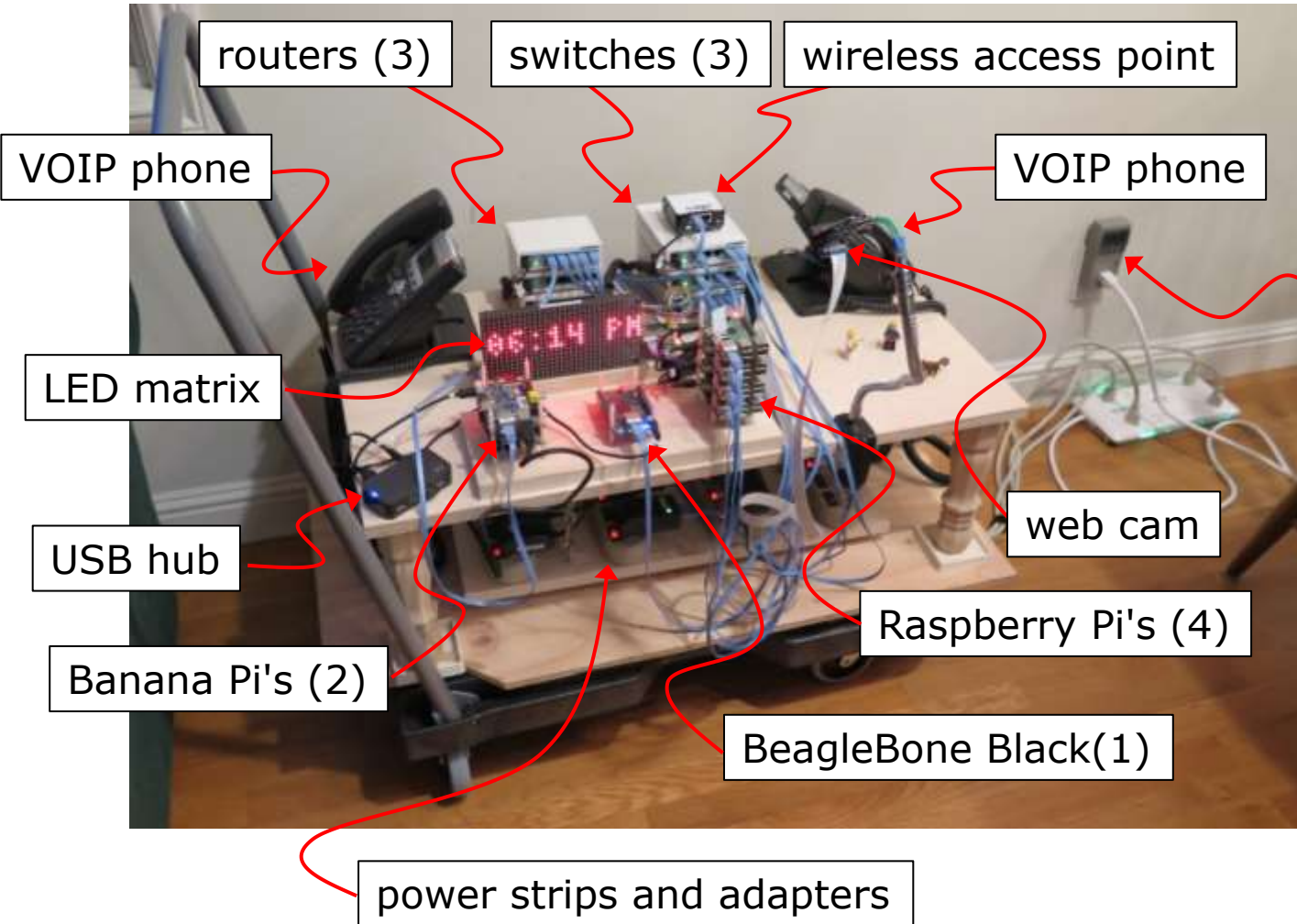


ssh <username>@baby-opus

```
root@baby-opus:~# ./led-who-start  
root@baby-opus:~# ./led-who-stop
```

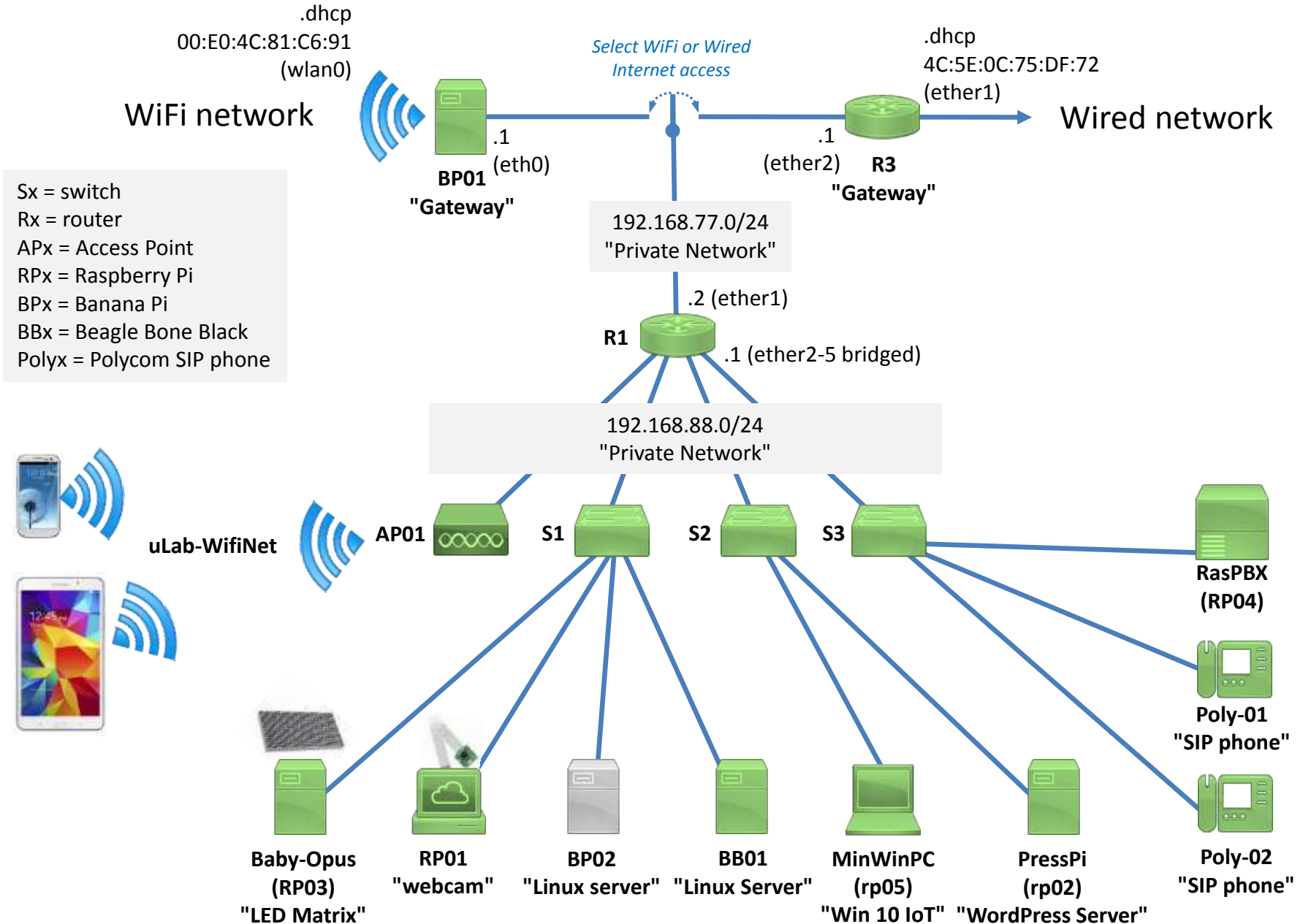
```
ip dhcp pool r3-uLab-828  
host 172.30.1.33 255.255.255.0  
client-identifier 014c.5e0c.75df.72  
  
ip route 192.168.77.0 255.255.255.0 172.30.1.33  
ip route 192.168.88.0 255.255.255.0 172.30.1.33
```


uLab



39 watts total

uLab Network Map



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

Links

<p>Instructors</p> <ul style="list-style-type: none"> Linux Master Jim Programming Master Ed Network Master Gerardo Network Master Rich 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"> CIS CS CS <p>Crib Sheets</p> <ul style="list-style-type: none"> CIS 90 (CIS 90) <p>Documentation</p> <ul style="list-style-type: none"> TLDP LNBEI LNBI Rosetta Stone <p>Animations</p> <ul style="list-style-type: none"> Linux network technologies 	<p>Getting Linux</p> <ul style="list-style-type: none"> Linux ISOs Kernel RPMs (preinst) RPMs (phone) Openstack <p>Tools and Software</p> <ul style="list-style-type: none"> Apache httplib ColD cygwin DCS boot disks DynamicSystem John the Ripper netfilter Putty SSH Tools Quake making suite Ipqstat WinStack <p>Academic Software for CIS Students</p> <ul style="list-style-type: none"> Microsoft Webtoce VMware Webtoce <p>Virtualization</p> <ul style="list-style-type: none"> VirtualBox VMware ESX and vSphere client <p>Standards</p> <ul style="list-style-type: none"> IEEE IEEE (IEEE) <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 Shell vi summary vi cheat sheet <p>Howtos</p> <ul style="list-style-type: none"> HostForge email DNS Ethernet (NIC drivers) NFS NO PDF Putty SSH Keys Using sed <p>Student Howtos</p> <ul style="list-style-type: none"> Horizon Script by Sean Cahalan WiFi Penetration by Ryan Schae Looking into Caps from a Mac by Laura Sirokovic LDAP Implementation by Tim Chidlers Install and DualBoot into Microsoft Windows 7 and Linux Ubuntu by Richie Foo Making an ethernet cable by Michael George Home VM access via Linksys router by Marc Romansky Putty to Vm by Marc Romansky Installing VirtualBox by Marcos Valdebenito Linux Permissions by Michael Wicherzki Guide to /devnull by Michael Wicherzki <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.

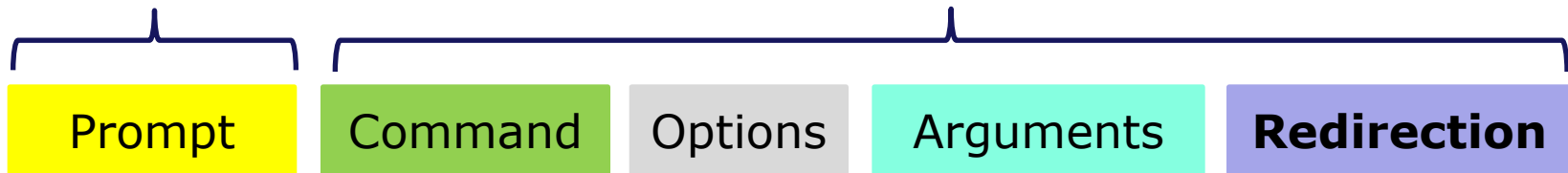
>> *filename*

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

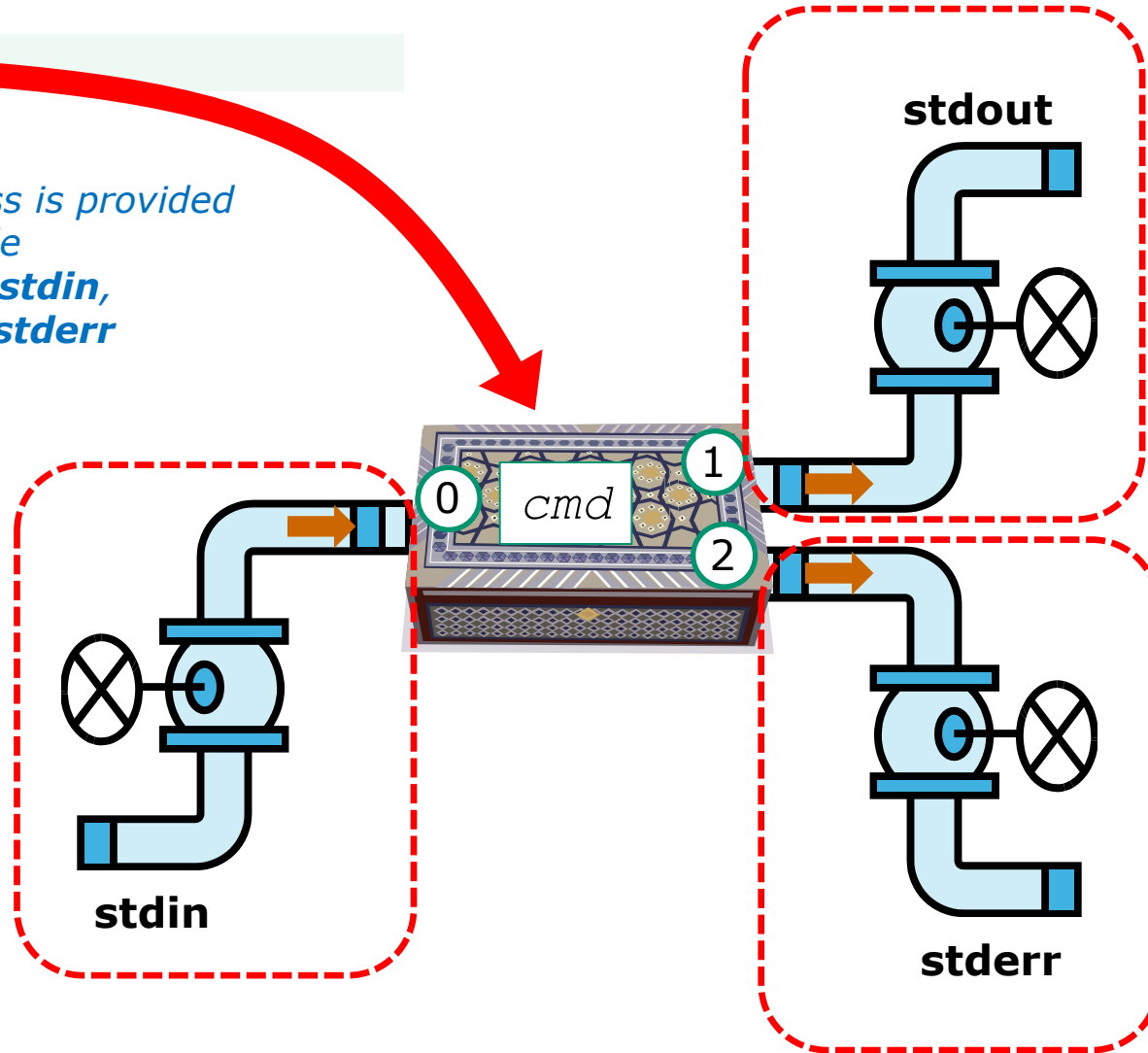
```

/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**

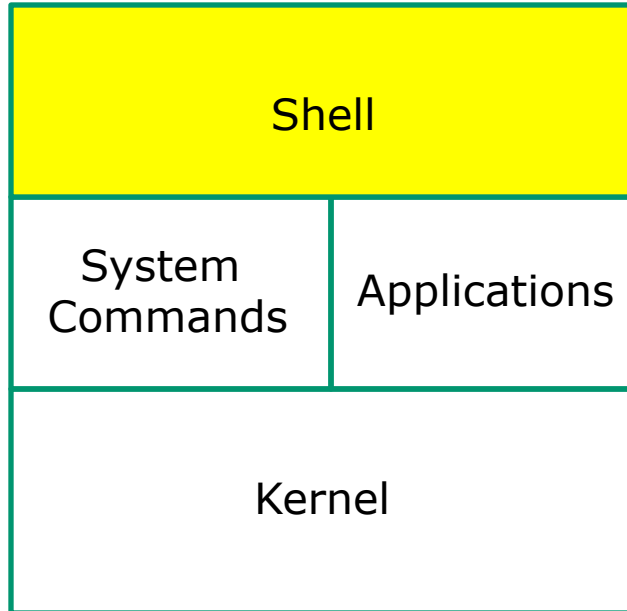




All Together Now Example



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 (show with **echo \$PS1**).

```
/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. Do a hex dump of your *five* file: **xxd five**
4. Put the size of your *five* file and the hex value of the newline character in the chat window.
5. 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.

Activity

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

Prove to yourself that the shell will find the **sort** command is in the */bin* directory.

1. Use **echo \$PATH** to view your path.
2. Check the first three directories on your path to see if one of them contains the sort command:
 - Use **ls -li /usr/lib/qt-3.3/bin | grep sort**
 - Use **ls -li /usr/local/bin | grep sort**
 - Use **ls -li /bin | grep sort**
3. Write the inode number of the sort program file in the chat window.

Example

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

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



```
bella
benji
daisy
duke
homer
ivy
lucy
oscar
```

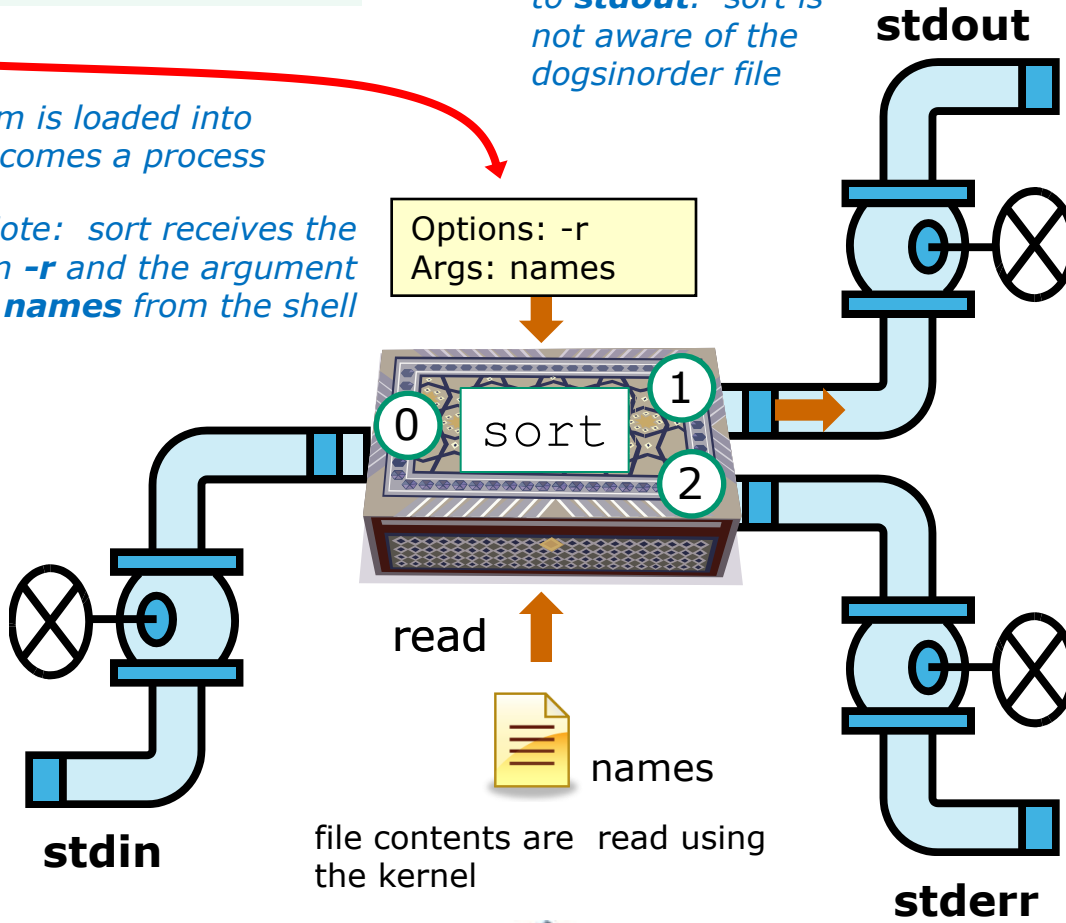
The sort program is loaded into memory and becomes a process

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

Options: -r
Args: names

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

- 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!
```

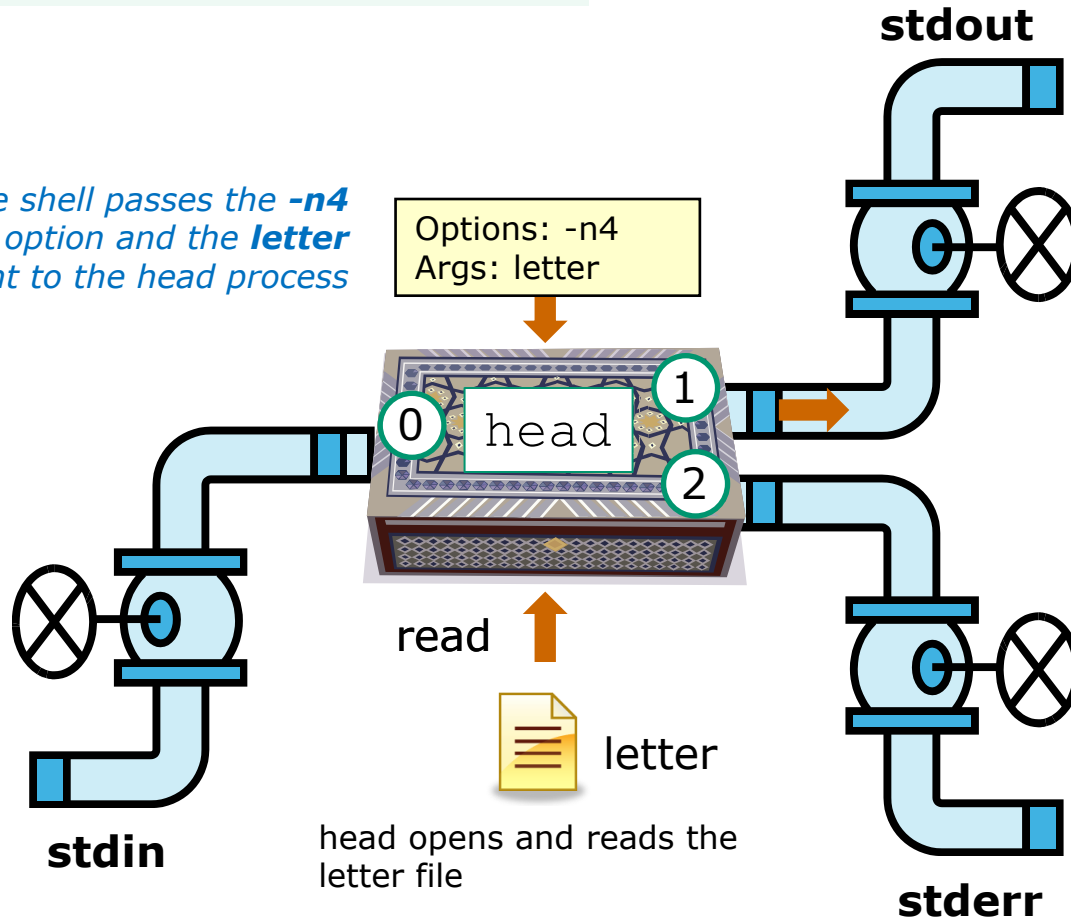
```
Here I am at Camp Granada. Things are very entertaining,  
and they say we'll have some fun when it stops raining.
```

head -n4 letter

option → ← *argument*

```
$ head -n4 letter
```

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



```
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

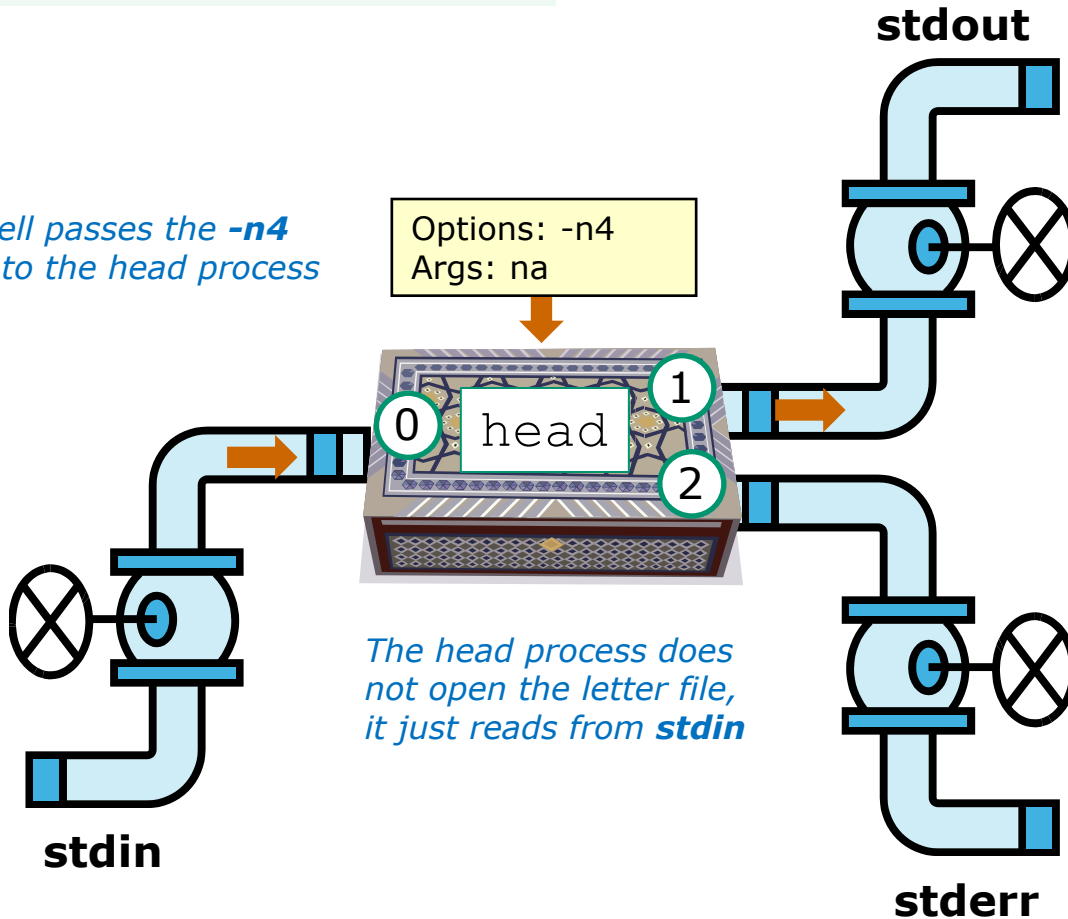
Options: -n4
Args: na

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**



letter

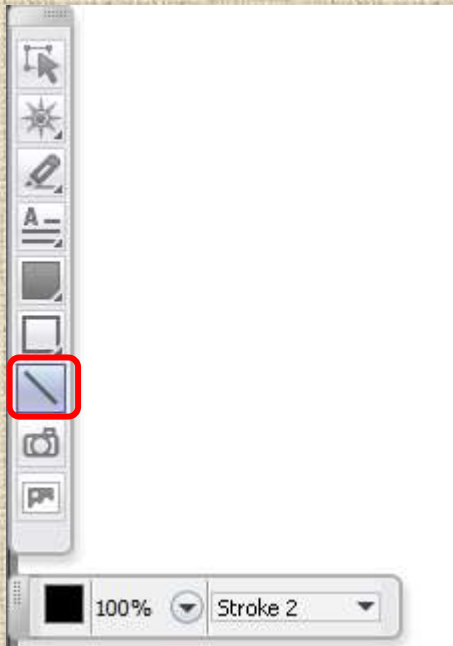




Errors

Instructor: Switch to CCC Confer Whiteboard

CCC Confer Whiteboard Activity



Select the straight line drawing tool and connect the like images

CCC Confer Whiteboard Activity

Connect with a straight line the command with the error message

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

CCC Confer Whiteboard Activity

Connect with a straight line the command with the error message

Commands

\$ **cat < bogus**

\$ **cat bogus**

\$ **bogus**

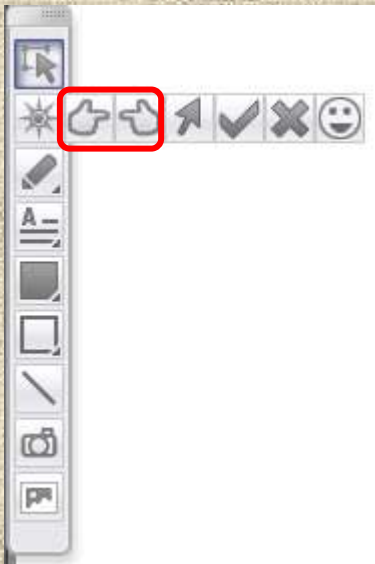
Error messages

-bash: bogus: command not found

-bash: bogus: No such file or directory

cat: bogus: No such file or directory

CCC Confer Whiteboard Activity



1

2

3

4

Select one of the pointing finger markers and point at the number called out by the instructor

CCC Confer Whiteboard Activity

Given: There is no file named *bogus*

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

*Point your electronic finger at the shell step
where the error message was generated*

Shell Steps

1) Prompt

2) Parse

3) Search

4) Execute

5) Nap

6) Repeat

CCC Confer Whiteboard Activity

Given: There is no file named *bogus*

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

*Point your electronic finger at the shell step
where the error message was generated*

Shell Steps

1) Prompt

2) Parse

3) Search

4) Execute

5) Nap

6) Repeat

CCC Confer Whiteboard Activity

Given: There is no file named *bogus*

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

*Point your electronic finger at the shell step
where the error message was generated*

Shell Steps

1) Prompt

2) Parse

3) Search

4) Execute

5) Nap

6) Repeat

CCC Confer Whiteboard Activity

Given: There is no file named *bogus*

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

*Point your electronic finger at the shell step
where the error message was generated*

Shell Steps

1) Prompt

2) Parse

3) Search

4) Execute

5) Nap

6) Repeat

CCC Confer Whiteboard Activity

Given: There is no file named *bogus*

```
[rsimms@oslab ~]$ cat bogus  
cat: bogus: No such file or directory 1) Execute
```

```
[rsimms@oslab ~]$ bogus  
-bash: bogus: command not found 3) Search
```

```
[rsimms@oslab ~]$ cat < bogus  
-bash: bogus: No such file or directory 2) Parse
```

```
[rsimms@oslab ~]$ bogus < bogus  
-bash: bogus: No such file or directory 2) Parse
```


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
```



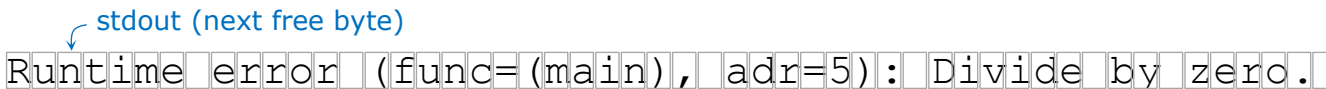
stderr (next free byte)



2+2

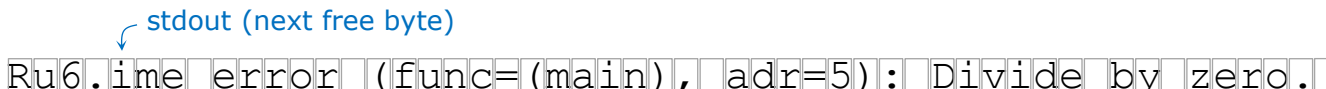
stderr (next free byte)

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)

programming examples





C Program I/O example View the program

```

/home/cis90/simben/bin $ cat simple.c
char question[] = "What is your name stranger? ";
char greeting[] = "Well I'm very pleased to meet you, ";
char buffer[80];
main()
{
    int len;

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

```

Write question to **stderr**

Read name from **stdin**

Write greeting to **stdout**

Write name to **stdout**

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



C Program I/O example

Compile the program

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

```
/home/cis90/simben/bin $ 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 Execute the program

```
/home/cis90/simben/bin $ simple  
What is your name stranger? Rich  
Well I'm very pleased to meet you, Rich
```

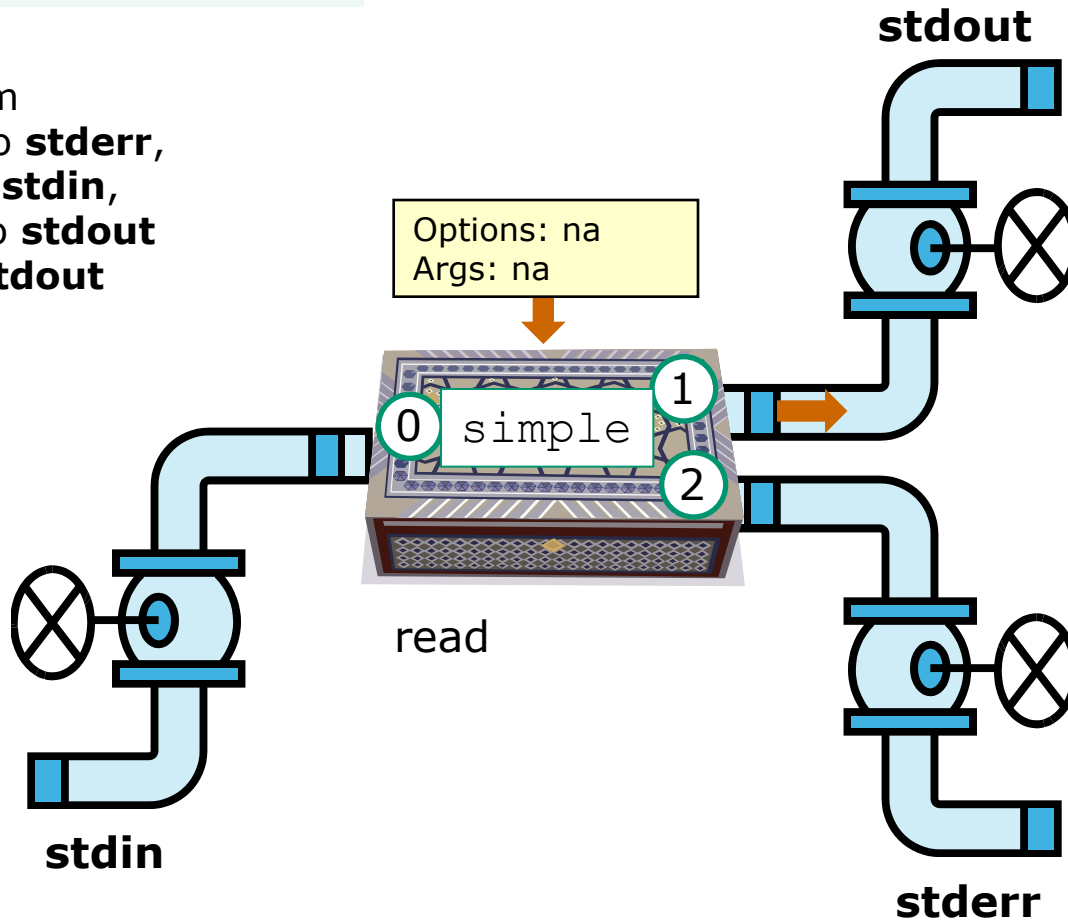
Running the simple program.

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

```
/home/cis90/simben/bin $ simple > myfile  
What is your name stranger? Rich
```

```
/home/cis90/simben/bin $ cat myfile  
Well I'm very pleased to meet you, Rich
```

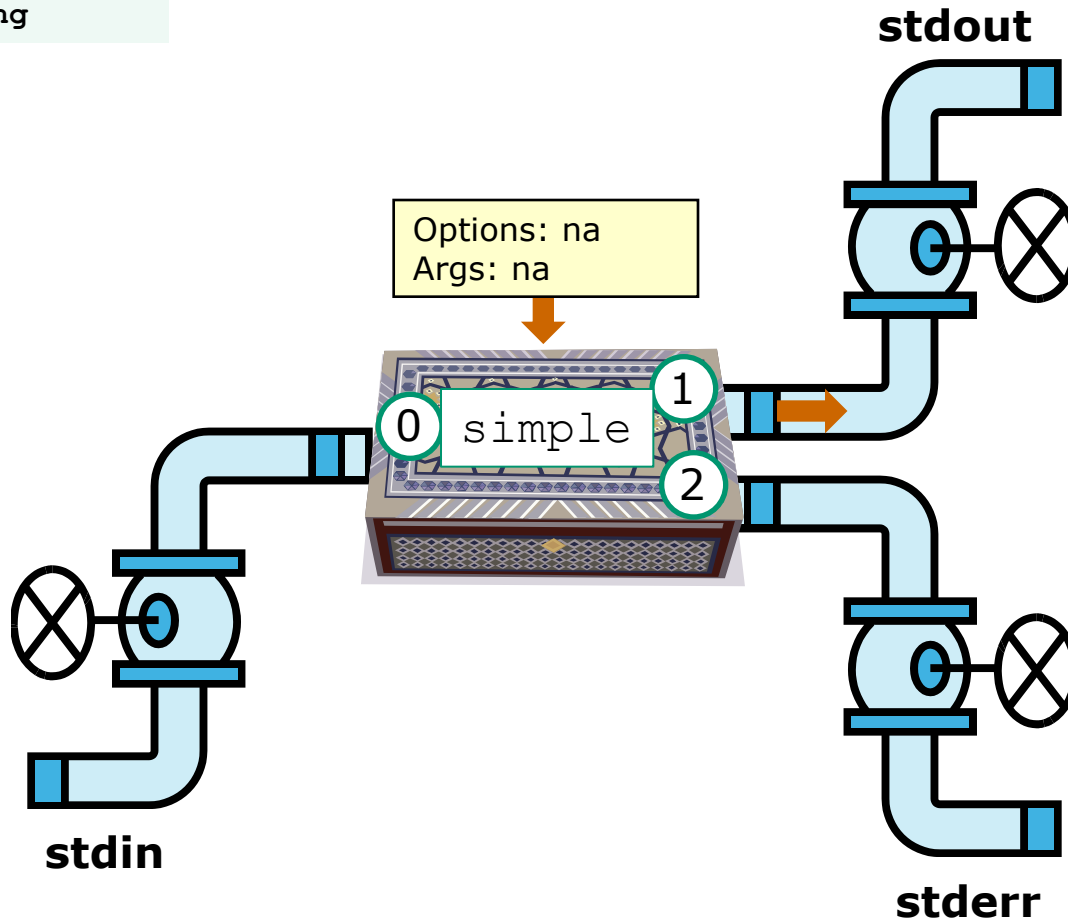
*In this example,
output has been
redirected to a file
named myfile.*

*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 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



C++ Program I/O example View the program

```
/home/cis90/simben/bin $ cat simpleplus.cpp
```

```
#include <iostream>
using namespace std;
```

```
int main() {
    string question = "What is your name stranger? ";
    cerr << question;
```

*Write question to **stderr***

```
    string buffer;
```

```
    cin >> buffer;
```

*Read name from **stdin***

```
    string greeting = "Well I'm very pleased to meet you, ";
    cout << greeting << buffer << endl;
```

```
    return 0;
```

```
}
```

*Write greeting and name to **stdout***

This program is available in the depot directory



C++ Program I/O example

Compile the program

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

```
/home/cis90/simben/bin $ make simpleplus  
g++      simpleplus.cpp  -o simpleplus
```

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 Execute the program

```
/home/cis90/simben/bin $ simpleplus  
What is your name stranger? Rich  
Well I'm very pleased to meet you, Rich
```

Running the simpleplus program



Activity

1. Change to your bin directory
cd ~/bin
2. Copy the C++ source code from the depot directory
cp ../../depot/simpleplus.cpp .
3. Look at your program
cat simpleplus.cpp
4. Compile the program
make simpleplus
5. Run the program
simpleplus



Python Script I/O example

View the program

```
/home/cis90/simben $ cat simple.py
import sys
sys.stderr.write("What is your name stranger? ") Output question to stderr
name = sys.stdin.readline() Input name from stdin
sys.stdout.write("Well I'm very pleased to meet you, " + name)
```



*Output greeting and name to **stdout***

This program is available in the depot directory



Python Script I/O example View the program

```
/home/cis90/simben $ python simple.py  
What is your name stranger? Rich  
Well I'm very pleased to meet you, Rich  
/home/cis90/simben $
```

Running the python simple.py script



Activity

1. Change to your bin directory
cd ~/bin
2. Copy the python script from the depot directory
cp ../../depot/simple.py .
3. Look at your program
cat simple.py
4. Run the script
python simple.py



umask

(review)

Review - applying umask bits

Example umask setting

```
/home/cis90/simben/lesson9 $ umask  
0002 this mask indicates which permissions should NOT  
be set on new files or directories
```

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

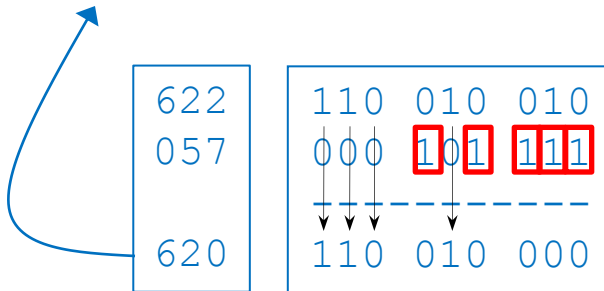
```
/home/cis90/simben/lesson9 $ mkdir newdir  
/home/cis90/simben/lesson9 $ ls -ld newdir  
drwxrwxr-x 2 simben cis90 4096 Oct 27 07:23 newdir
```

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 Example umask setting
/home/cis90/simben $ umask
0057
```

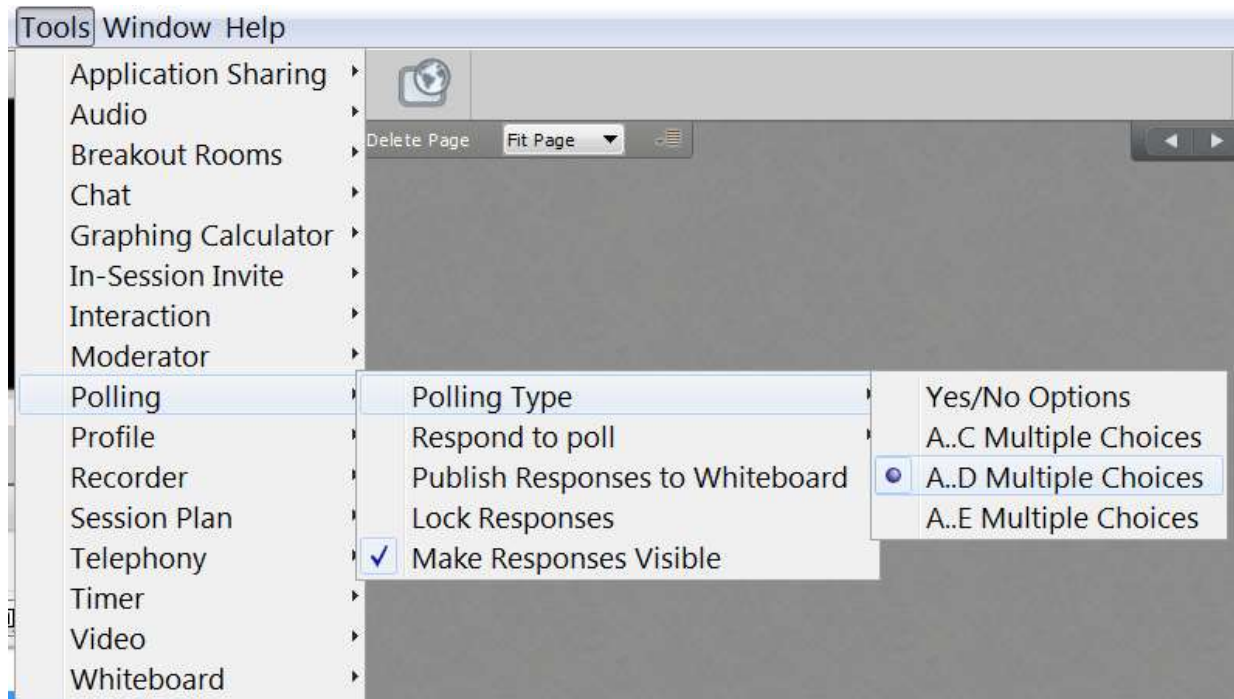
```
/home/cis90/simben $ chmod 622 myfile
/home/cis90/simben $ cp myfile myfile.bak
/home/cis90/simben $ ls -l myfile*
-rw--w--w-. 1 simben90 cis90 0 Mar 24 17:50 myfile
-rw--w----. 1 simben90 cis90 0 Mar 24 17:51 myfile.bak
```



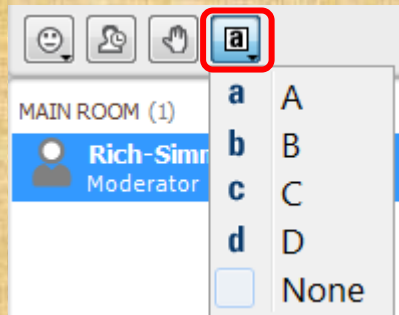
Copied file - start with original file's permissions and apply the mask

*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.*

Rich's CCC Confer poll setup



Activity



Which pizza is the best?

- A. Round Table
- B. Pizza My Heart
- C. Tony & Alba's
- D. Upper Crust

Respond to the poll above

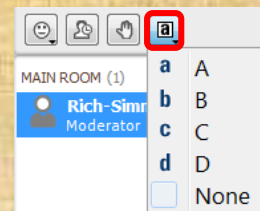
Activity

I want to change the permissions on an existing file

Which command does this?

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

Respond to the poll above



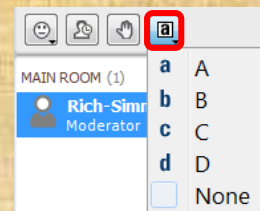
Activity

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

Which command does this?

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

Respond to the poll above



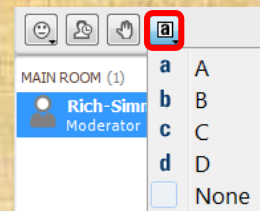
Activity

I want to show the owner of a file and its permissions in mnemonic format e.g. rwxr-xr-x

Which command does this?

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

Respond to the poll above



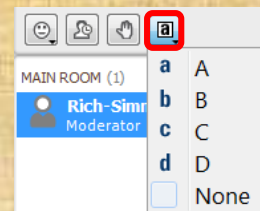
Activity

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

Which command does this?

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

Respond to the poll above





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.

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.

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.



Pipeline and Redirection Practice

bc command with no redirection or piping

```
/home/cis90/simben $ bc
bc 1.06.95
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006 Free
Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
2+2
4
4/0
Runtime error (func=(main), adr=5): Divide by zero
quit
/home/cis90/simben $
```


Piping output to bc command

```
/home/cis90/simben $ echo 2+2 | bc
```

```
4
```

```
/home/cis90/simben $ echo 4/0 | bc
```

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

Redirecting stdin of bc command

Setup:

```
/home/cis90/simben $ echo 2+2 > datafile  
/home/cis90/simben $ echo 4/0 >> datafile  
/home/cis90/simben $ cat datafile  
2+2  
4/0
```

Example:

```
/home/cis90/simben $ bc < datafile  
4  
Runtime error (func=(main), adr=5): Divide by zero
```

Piping output to bc command

Setup:

```
/home/cis90/simben $ echo 2+2 > datafile  
/home/cis90/simben $ echo 4/0 >> datafile  
/home/cis90/simben $ cat datafile  
2+2  
4/0
```

Example:

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

Redirecting stdin, stdout and stderr of bc command

Setup:

```
/home/cis90/simben $ echo 2+2 > datafile  
/home/cis90/simben $ echo 4/0 >> datafile  
/home/cis90/simben $ cat datafile  
2+2  
4/0
```

Example:

```
/home/cis90/simben $ bc < datafile > results 2> errors  
/home/cis90/simben $ cat results  
4  
/home/cis90/simben $ cat errors  
Runtime error (func=(main), adr=5): Divide by zero
```



Piping stdout and redirecting stdin, stderr of bc command

Setup:

```
/home/cis90/simben $ echo 2+2 > datafile
/home/cis90/simben $ echo 4/0 >> datafile
/home/cis90/simben $ cat datafile
2+2
4/0
```

Example:

```
/home/cis90/simben $ bc < datafile 2> errors | mail -s "Example" simben90
/home/cis90/simben $ cat errors
Runtime error (func=(main), adr=5): Divide by zero
```

Activity

Setup:

```
/home/cis90/simben $ echo 2+2 > datafile  
/home/cis90/simben $ echo 4/0 >> datafile  
/home/cis90/simben $ cat datafile  
2+2  
4/0
```

Example:

```
/home/cis90/simben $ bc < datafile 2> errors | mail -s "Example" $LOGNAME  
/home/cis90/simben $ cat errors  
Runtime error (func=(main), adr=5): Divide by zero
```

Past the email you receive into 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: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/nfs2_fs.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/include/linux/cdrom.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_j.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-220.23.1.el6.i686/scripts/get_maintainer.pl:patch@pernan.chief:Linus Torvalds:torvalds@linux-foundation.org);
/usr/src/kernels/2.6.32-220.23.1.el6.i686/scripts/package/Makefile: # 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 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: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/nfs2_fs.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/include/linux/cdrom.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_j.h: * Copyright (C) 1991, 1992 Linus Torvalds
/usr/src/kernels/2.6.32-71.el6.i686/scripts/get_maintainer.pl:patch@pernan.chief:Linus Torvalds:torvalds@linux-foundation.org);
/usr/src/kernels/2.6.32-71.el6.i686/scripts/package/Makefile: # 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

Finding Things

Task

Find all files in the `/home/cis90/bin` that are regular files and belong to the staff group.

```
find /home/cis90/bin -group staff -type f
```

```
/home/cis90/bin/enlightenment
/home/cis90/bin/allscripts
/home/cis90/bin/list
/home/cis90/bin/submit.sp15.v1
/home/cis90/bin/tinsam90/schedule.pyc
/home/cis90/bin/tinsam90/schedule.py
/home/cis90/bin/tinsam90/forums.py
/home/cis90/bin/tinsam90/tips.py
/home/cis90/bin/tinsam90/grade.py
/home/cis90/bin/submitx
/home/cis90/bin/old/submit.fa14.v5
/home/cis90/bin/old/checkgrades.py.fa14
/home/cis90/bin/old/allscripts.sp14
/home/cis90/bin/old/check10.v2
/home/cis90/bin/old/submit.fa14.v1
/home/cis90/bin/old/check10.v1
/home/cis90/bin/old/submit.fa14.v4
/home/cis90/bin/old/checkgrades.py.sp14
/home/cis90/bin/old/submit.fa14.v2
/home/cis90/bin/old/submit.fa14.v3
/home/cis90/bin/old/submit.fa14.v6
/home/cis90/simben $
```

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

Finding Things

Task

Count all the directories in the `/home/cis90` branch of the file tree that belong to the cis90 group. Discard any permission errors.

```
???? /home/????? -type ? -group ????? ?? /dev/null | ?? -?
```

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



Room 1



Room 2



Room 3



Room 4

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 4 rooms)



Flashcard Practice

Flashcards



debian

Room 1

Points:



Room 2

Points:



Room 3

Points:



Room 4

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  
(Use countdown alias)
```


Assignment

Practice Test

Practice Test



Practice test available

- Available on Blackboard
- Work alone or together in study groups
- 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 2015)

Instructions **HONOR CODE:**

This is a practice test and you may work with others on it. You are encouraged to compare and discuss answers with your classmates using the forum, study groups or both. 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. Because this system is on a private network log into Opus first then ssh into sun-hwa-p2. The sun-hwa-p2 system will **not be available** after the real test starts.

Grading will be based on your answers AND that you correctly implemented the "DO THIS FIRST" portion of the question.

On the practice test you can ask your classmates for help on the forum if you get stuck. On the real test you can ask the instructor for the answer and forfeit the points.

Please KEEP YOUR ANSWERS TO A SINGLE LINE ONLY !!

This test must be completed in one sitting. The submittal will be made automatically when the time is up. If you submit early by accident you will not be able to re-enter and continue. If this happens on the real test don't panic! Just email the instructor any remaining answers before the time is up.

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

A sunset over a beach with a cliff on the right. The sky is filled with vibrant colors of orange, pink, and purple. The text 'Wrap up' is overlaid in the center.

Next Class

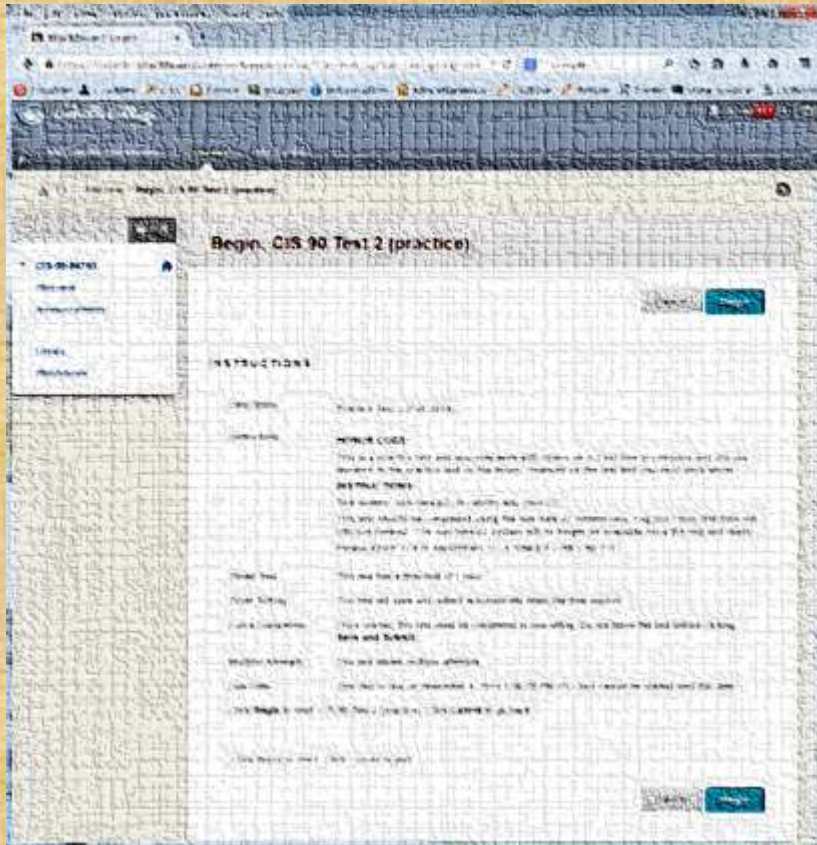
No Quiz

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!**
 - **Make a personal reference "crib sheet" document**
 - **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