

Lesson Module Checklist

- Slides
- Flash cards
- First minute quiz
- Web calendar summary
- Web book pages
- Commands
- Howtos

- Lab tested
- Opus - submit and turnin directory tested

- Bring Add Codes
- Bring printed roster

- Backup slides, Confer links, handouts on flash drive
- 9V backup battery for microphone

Introductions and Credits



Jim Griffin

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



Rich Simms

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

And thanks to:

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



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



Daniel



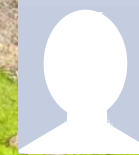
Riley



Solomon



Curtis



Dillon



Pam



Aaron



Liz



Gabe



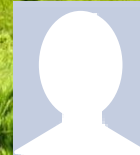
Lucie



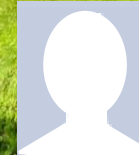
Liam



Michael L.



Ryan



Ben L.



Roger



Ariana



Evan



Alex



Natalia



Efrain



Perky



Anthony



Christopher



Samantha



Michael S.



Paul S.



Justin



Hilario



Todd



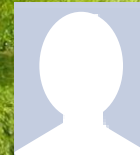
Tyrone



Ben C.



Nicholas



Paul N.



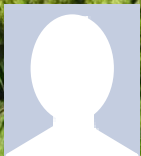
Justin



Andrew



Matthew



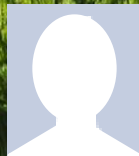
Jordan



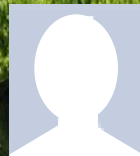
Mark



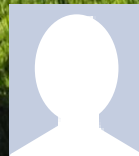
Dimitri



Ryan



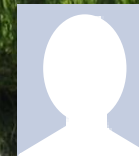
Greg



MJ



Jay



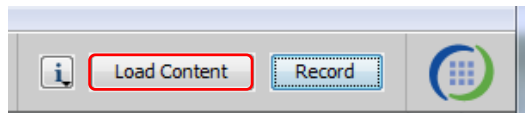
Brian



Rich

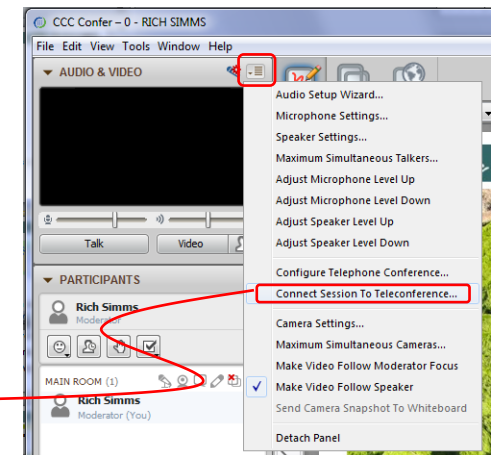
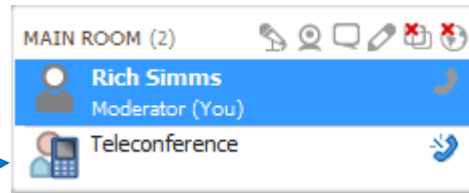


[] Preload White Board with *cis*lesson??*-WB*



[] Connect session to Teleconference

Session now connected to teleconference



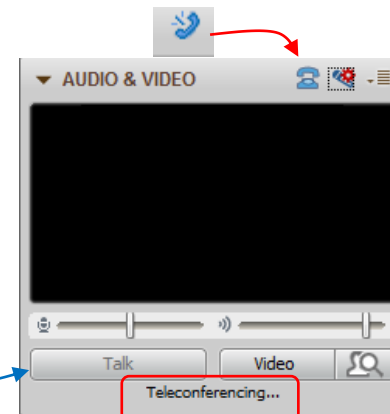
[] Is recording on?



Red dot means recording

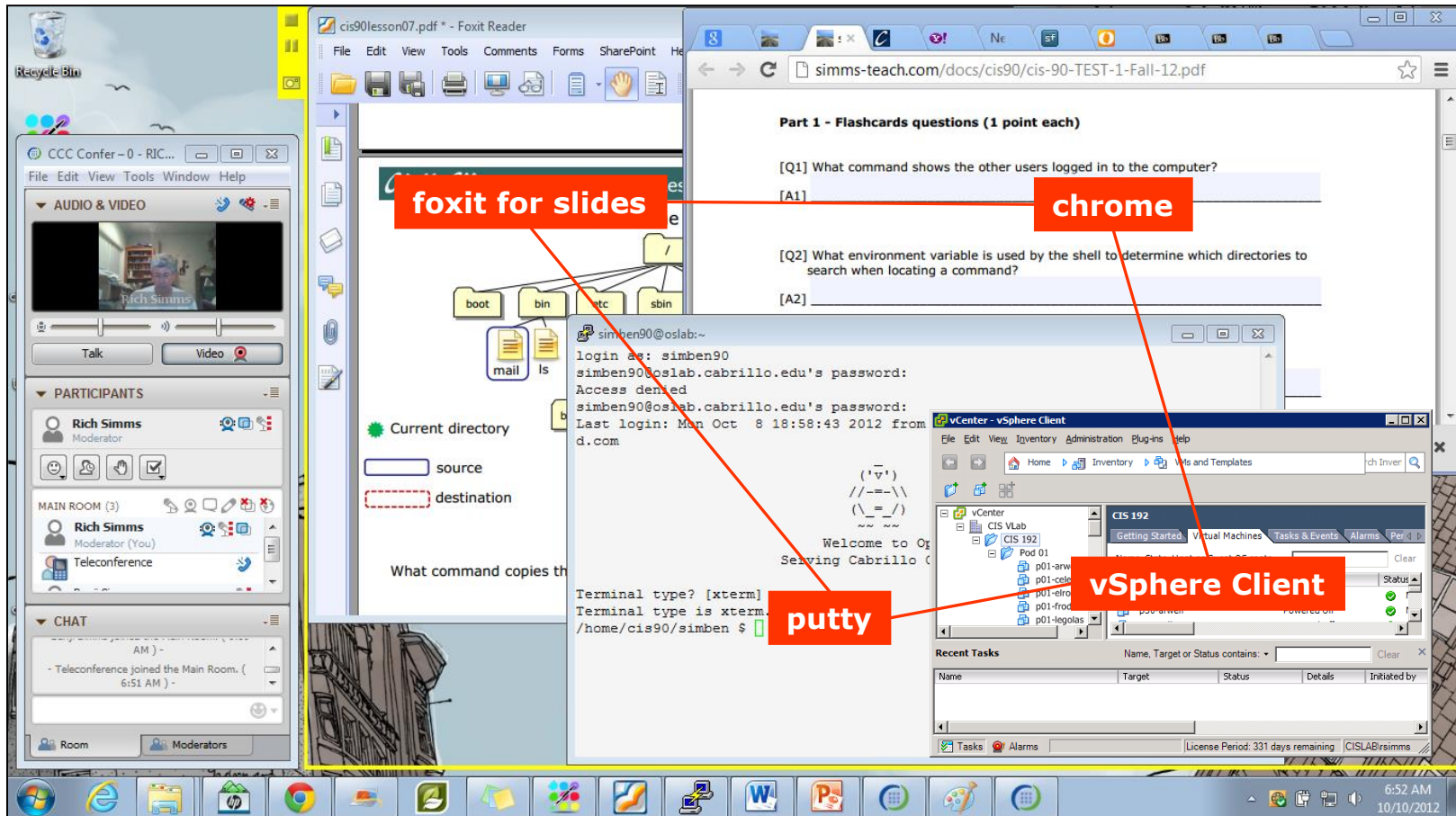
[] Use teleconferencing, not mic

Should be greyed out



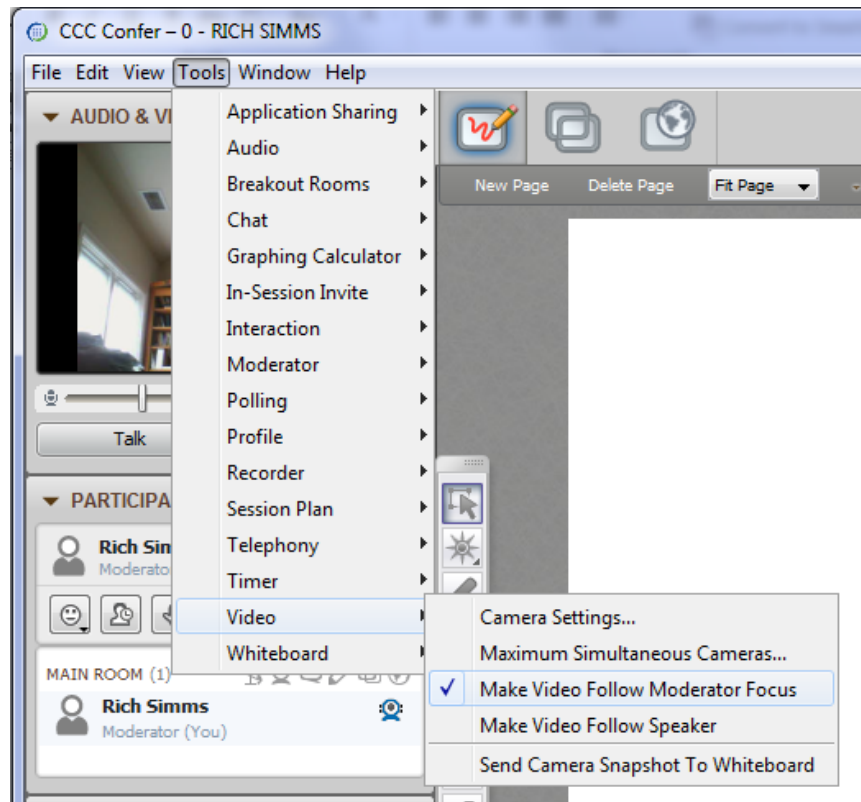


- [] Video (webcam) optional
- [] layout and share apps





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



Universal Fix for CCC Confer:

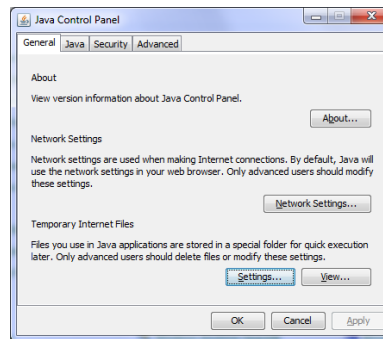
- 1) Shrink (500 MB) and delete Java cache
- 2) Uninstall and reinstall latest Java runtime



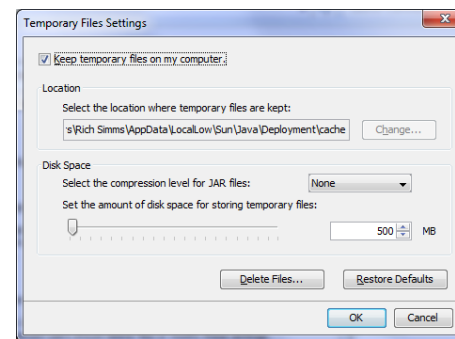
Control Panel (small icons)



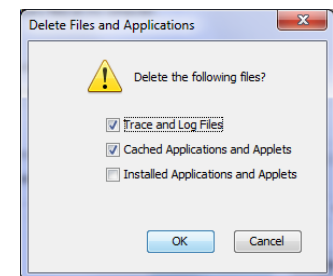
General Tab > Settings...



500MB cache size



Delete these



Google Java download



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)



Commands

Objectives	Agenda
• Understand how the UNIX login operation works. • Meet John the Ripper and learn how vulnerable a poor password is. • Understand basic command syntax and operation. • Understand program files and what happens when they are run. • Understand how the shell works and environment variables. • Understand how to get documentation when online.	• Quiz • Questions and Review • Putty tips • Deep dive on logging in • Passwords • Housekeeping • New commands • Programs/processes • Command line syntax • Environment variables • Metacharacters • Life of the shell • Docs • Wrap up

Questions

Questions

How this course works?

Previous lessons

Previous labs?

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.



Review and clarifications

Forum Top Issues

- 1) How to get into VLab
- 2) How to switch between virtual terminals
- 3) Unintended repeated keystrokes in VLab
- 4) Blank PDF submittals (surveys and lab submittals)

Thanks to everyone who posted these issues on the forum!

And thanks to everyone who posted solutions to these issues on the forum!



The new Lesson 1 tools in your toolbox

cal
clear
date
exit
history

Prints calendars
Clears the screen
Shows the time and date
Exits login session
Shows previous commands

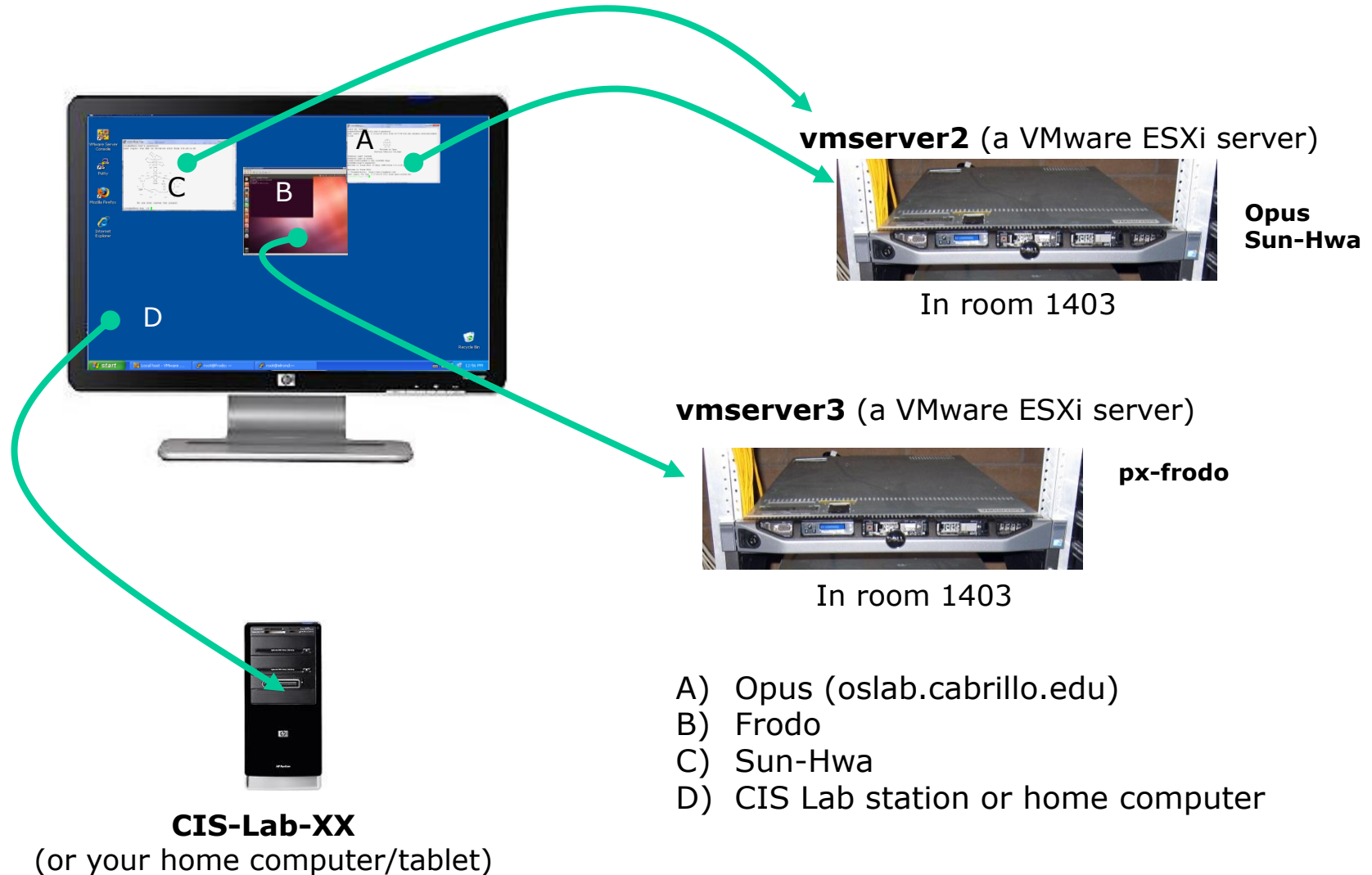
hostname
id
ps
ssh
uname

Shows name of computer being interacted with
Shows UID's, GID's and SELinux information
Shows process information
Initiates connection and login to remote computer
Shows name of operating system kernel

tty
who
who am i

Shows name of terminal device
Shows all users who are logged in and from where
*Like **who**, but only shows your login session*

We used three physical and three virtual computers for Lab 1 !!



Terminals

The image shows two terminal windows. The top window, titled 'rsimms@opus:~/cis90/lab02', displays the output of 'ls' and 'ls -l' commands, showing file permissions, owners, sizes, and dates. The bottom window, titled 'guest90@opus:~/', shows the output of 'cal' and 'cal' commands, displaying a calendar for September 2010.

Terminal emulators like PuTTY or Mac terminal (with scroll bars, colors, customizable backgrounds, fonts and sizes) and runs on another computer

The image shows a graphical terminal window titled 'cis90@eko: ~'. It has a yellow background and a menu bar with 'File Edit View Terminal Help'. The window contains a list of system commands and their outputs, including 'df', 'cat', 'cal', and 'ls'. The window also has a status bar at the bottom showing the current directory and file manager.

Graphical terminals (with scroll bars, colors, customizable backgrounds, fonts and sizes) available on the graphical desktop



tty = teletype

Terminals were used in the old days to interact with computers.

Today we use **terminal emulators** that are software programs.

The image shows a virtual terminal window with a black background and white text. It displays the output of 'hostname', 'tty', 'ps', and 'date' commands. The window has a simple, minimalist design with no scroll bars or other graphical elements.

Virtual terminals (use ctrl-alt-fn) (no scroll bars, also called a console)

Class Activity

Command Review

Login to Opus if you haven't already

*Now follow along as we review the commands
learned last week and new commands for this week*

Terminals types and devices

```
login as: simben90  
simben90@oslab.cabrillo.edu's password:  
Last login: Sat Sep  1 09:26:51 2012 from 50-0-68-  
235.dsl.dynamic.fusionbroadband.com
```

```
      _  
    ('v'  
  //--=\   
  (\_=_/  
   ~ ~ ~ ~
```

```
Welcome to Opus  
Serving Cabrillo College
```

```
Terminal type? [xterm]  
Terminal type is xterm.  
/home/cis90/simben $ tty  
/dev/pts/3
```

*The terminal type is **xterm***

*The terminal device for this session is **/dev/pts/3***

The **terminal type** is not the same as the **terminal device**

How can I print a calendar?

```
/home/cis90/simben $ cal
```

```
September 2012
Su Mo Tu We Th Fr Sa
                1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30
```

*The **cal** command*

```
/home/cis90/simben $ cal 9 2001
```

```
September 2001
Su Mo Tu We Th Fr Sa
                1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30
```

*Month and year **arguments***

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

A command can have arguments

What is the current time and date?

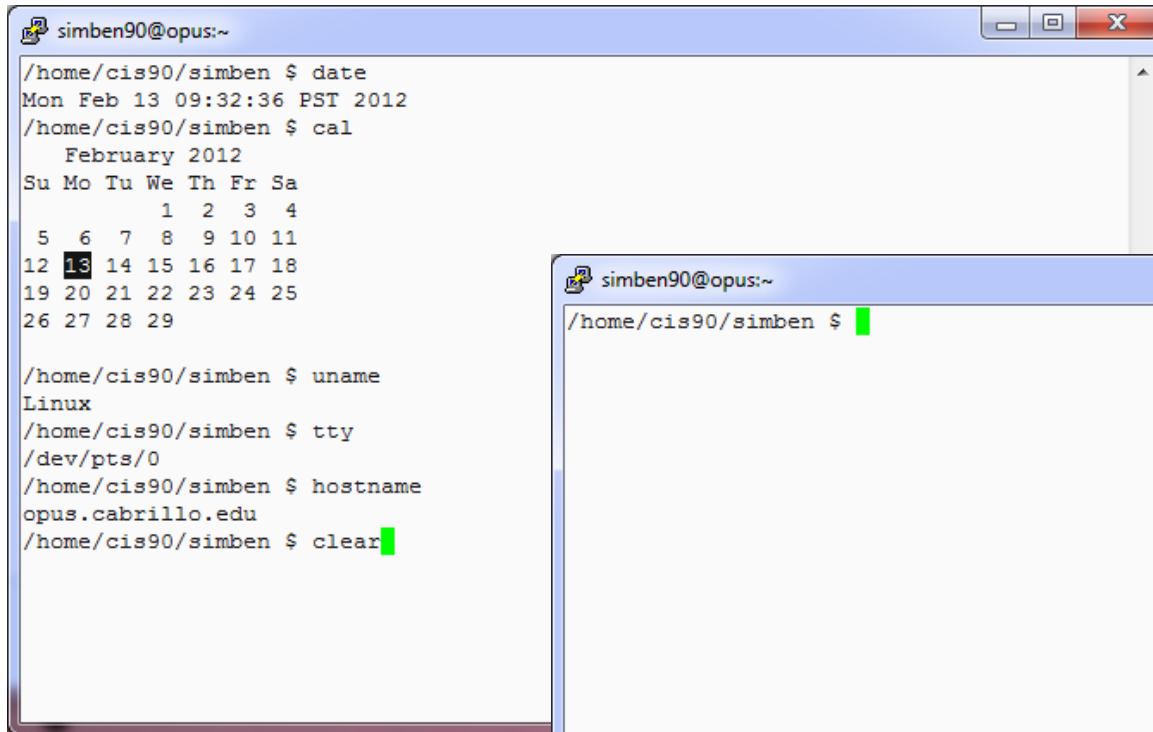
The shell "prompt"

The "command"

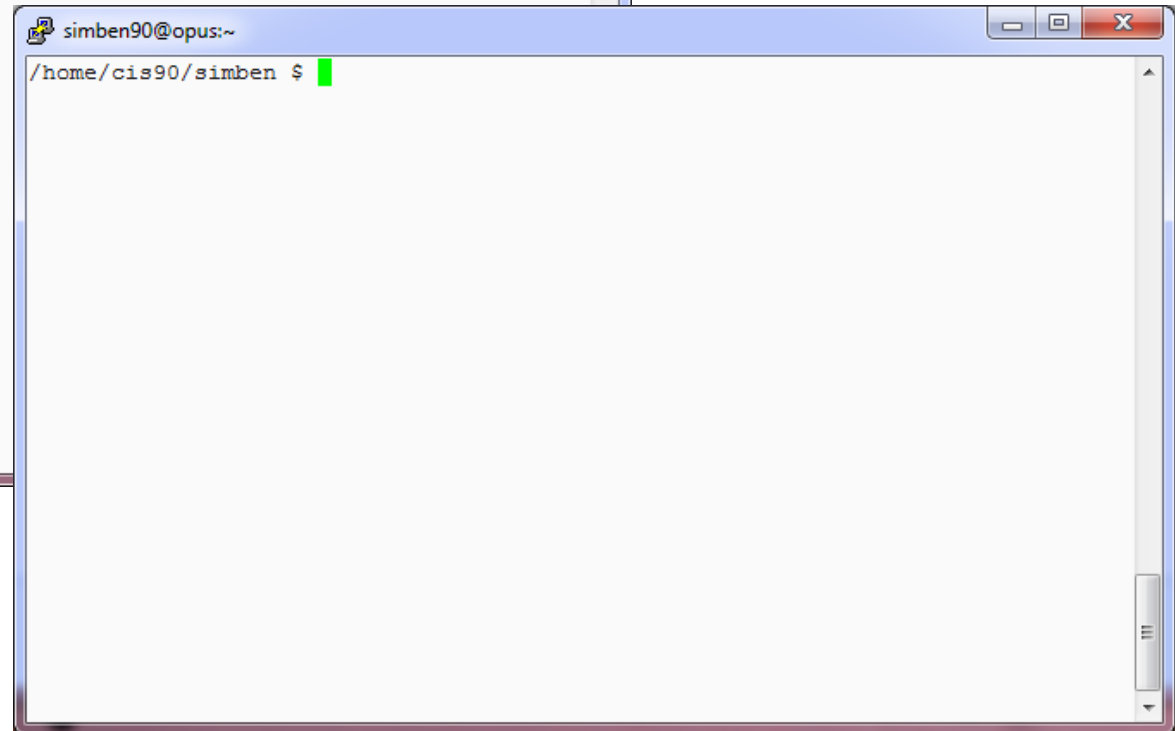
```
/home/cis90/simben $ date  
Sat Sep  1 14:03:33 PDT 2012  
/home/cis90/simben $
```

The prompt is output by the shell, you type the command

How do I clear the screen?



```
simben90@opus:~  
/home/cis90/simben $ date  
Mon Feb 13 09:32:36 PST 2012  
/home/cis90/simben $ cal  
February 2012  
Su Mo Tu We Th Fr Sa  
      1  2  3  4  
5  6  7  8  9 10 11  
12 13 14 15 16 17 18  
19 20 21 22 23 24 25  
26 27 28 29  
  
/home/cis90/simben $ uname  
Linux  
/home/cis90/simben $ tty  
/dev/pts/0  
/home/cis90/simben $ hostname  
opus.cabrillo.edu  
/home/cis90/simben $ clear
```

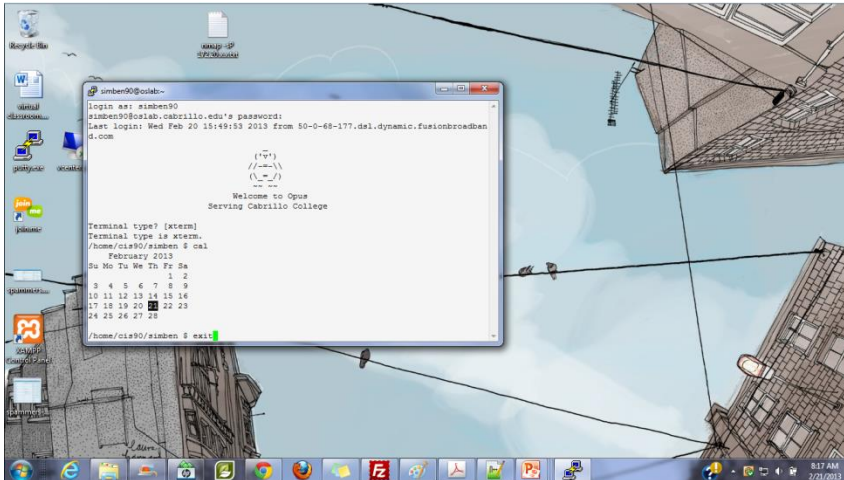


```
simben90@opus:~  
/home/cis90/simben $
```

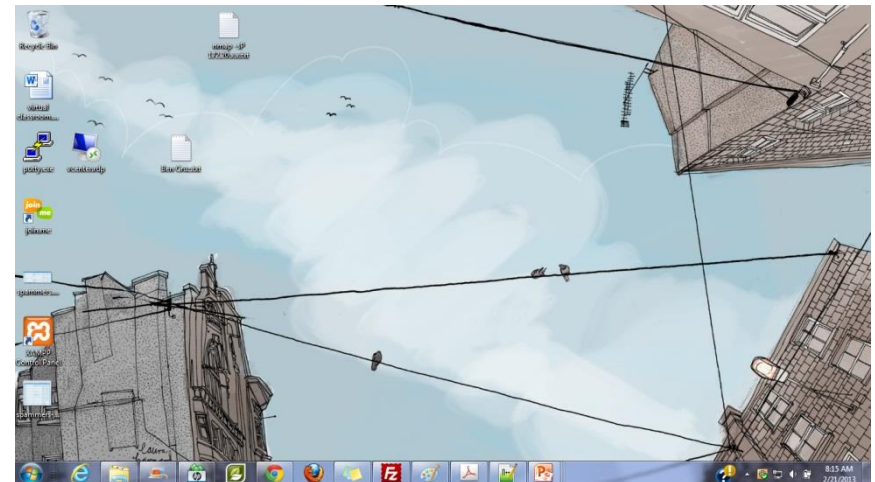
The **clear** command scrolls previous commands out of sight

How do I end this login session?

before **exit**



after **exit**



The **exit** command ends the session and the terminal window disappears ... POOF!

Viewing your command history

```
/home/cis90/simben $ history
```

```
1  hostname
2  exit
3  who
4  who -q
5  ps -e
```

```
< snipped >
```

```
177 cal 9 2001
178 exit
179 who
180 cal
181 tty
182 uname
183 ps
184 id
185 exit
186 history
```

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

*The **history** command outputs the commands used previously ... even from previous login sessions*

Tip: Use the "Up Arrow" key to quickly re-issue a previous command!

What is the name of the computer I'm interacting with?

```
/home/cis90/simben $ hostname  
oslab.cabrillo.edu  
/home/cis90/simben $
```

We still refer to Opus as "Opus" in this class however it's official hostname is really oslab.cabrillo.edu. This may change in the future after some network changes are made.

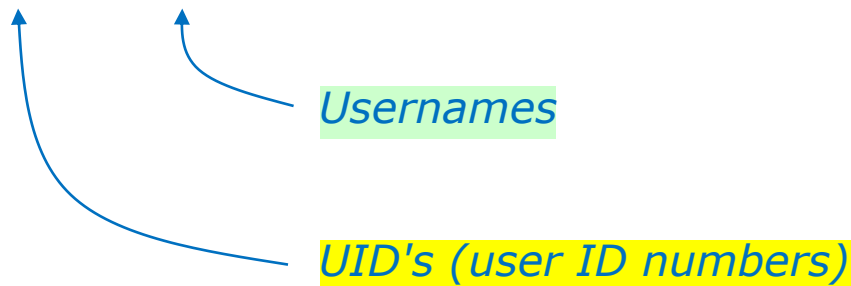
```
  _  
  ('v')  
  //--\\  
  (\\_/_/)  
  ~ ~ ~  
  Welcome to Opus  
  Serving Cabrillo College
```

What is the UID (User ID) for my account or other accounts?

```
/home/cis90/simben $ id
uid=1001(simben90) gid=190(cis90) groups=190(cis90),100(users)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

```
/home/cis90/simben $ id milhom90
uid=1002(milhom90) gid=190(cis90) groups=190(cis90),100(users)
```

```
/home/cis90/simben $ id simben90
uid=1001(simben90) gid=190(cis90) groups=190(cis90),100(users)
```



We are all just numbers to the Linux kernel

What shell am I using?

```

/home/cis90/simben $ ps
  PID TTY          TIME CMD
 28994 pts/0    00:00:00 bash
 29093 pts/0    00:00:00 ps
  
```

Process ID numbers →

Terminal device being used →

*the shell is sleeping and waiting for **ps** command to finish* →

***ps** command is running as it outputs this* →

The **ps** command outputs the current processes you own including the shell program you are using

How do I log into another computer system?

*The **ssh** command is used to log into another computer*

username on remote computer

IP address of remote computer

```
/home/cis90/simben $ ssh cis90@172.20.4.34
cis90@172.20.4.34's password:
Welcome to Ubuntu 12.04.1 LTS (GNU/Linux 3.2.0-29-generic x86_64)
```

*Notice how
the prompt
changes on
the remote
computer*

```
* Documentation:  https://help.ubuntu.com/
361 packages can be updated.
109 updates are security updates.
```

```
Last login: Wed Feb 20 17:26:25 2013 from oslab.cabrillo.edu
cis90@frodo-108:~$
```

*Note: You can also **ssh** into the same computer you are using already for an additional session.*

What kernel am I running on?

```
/home/cis90/simben $ uname  
Linux
```

The **uname** command (with no arguments) outputs the name of the operating system kernel

What terminal device am I using?

```
/home/cis90/simben $ tty  
/dev/pts/5
```

The **terminal type** is not the same as the **terminal device**

Who else is logged in and from where?

```
/home/cis90/simben $ who
simben90 pts/0      2013-02-21 08:17 (50-0-68-28.dsl.dynamic.fusion.com)
simben90 pts/1      2013-02-21 08:45 (50-0-68-28.dsl.dynamic.fusion.com)
milhom90 pts/2      2013-02-21 08:46 (50-0-68-28.dsl.dynamic.fusion.com)
rsimms    pts/4      2013-02-21 08:46 (50-0-68-28.dsl.dynamic.fusion.com)
rodduk90  pts/7      2013-02-21 08:46 (50-0-68-28.dsl.dynamic.fusion.com)
simben90  pts/8      2013-02-21 08:49 (172.20.4.34)
milhom90  pts/9      2013-02-21 08:50 (sun-hwa.cislab.net)
```

username

*terminal device
(pts/5 = /dev/pts/5)*

*where they logged
in from (hostname
or IP address)*

The who command shows who is logged in, their terminal device, when they logged in and from where they logged in

Which is my login session?

```
/home/cis90/simben $ who
```

```
simben90 pts/0      2013-02-21 08:17 (50-0-68-28.dsl.dynamic.fusion.com)
simben90 pts/1      2013-02-21 08:45 (50-0-68-28.dsl.dynamic.fusion.com)
milhom90 pts/2      2013-02-21 08:46 (50-0-68-28.dsl.dynamic.fusion.com)
rsimms    pts/4      2013-02-21 08:46 (50-0-68-28.dsl.dynamic.fusion.com)
rodduk90  pts/7      2013-02-21 08:46 (50-0-68-28.dsl.dynamic.fusion.com)
simben90  pts/8      2013-02-21 08:49 (172.20.4.34)
milhom90  pts/9      2013-02-21 08:50 (sun-hwa.cislab.net)
```

```
/home/cis90/simben $ who am i
```

```
simben90 pts/0      2013-02-21 08:17 (50-0-68-177.dsl.dynamic.fusion.com)
```

```
/home/cis90/simben $ tty
```

```
/dev/pts/0
```

When logged in multiple times use the terminal device to distinguish the sessions

Name Lingo

Example Linux System



username = rsimms

name of terminal device used by rsimms = `/dev/pts/15`

(terminal type = ansi)

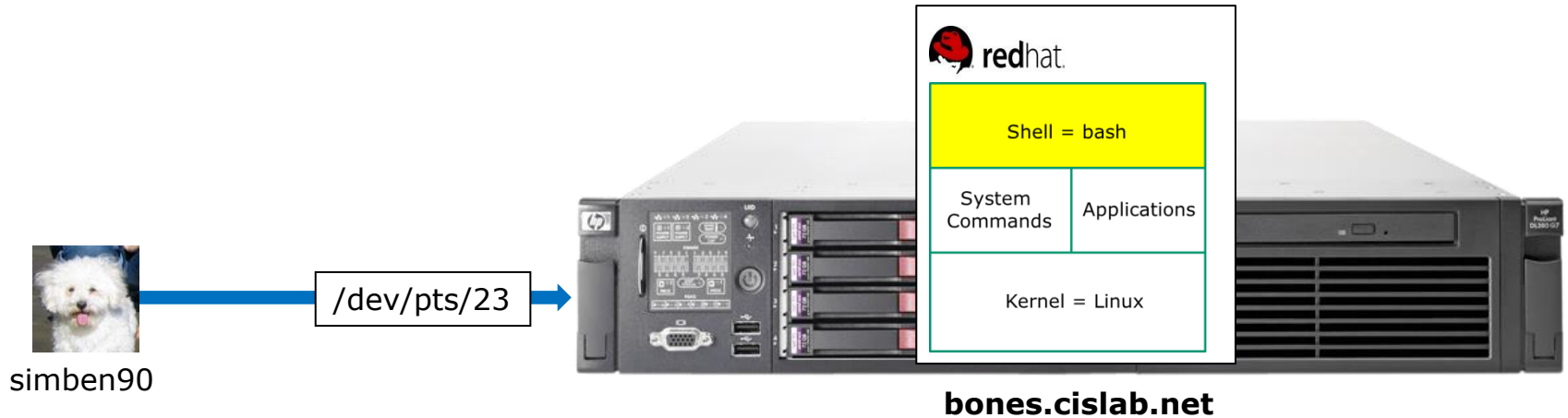
hostname = Criter

Name of distro = SUSE Linux Enterprise

Name of shell = sh

Name of kernel = Linux

Another Example Linux System



user**name** = simben90

name of terminal device used by simben90 = /dev/pts/23

(terminal type = xterm)

host**name** = bones.cislab.net

Name of distro = Red Hat Enterprise Linux

Name of shell = bash

Name of kernel = Linux

Test your knowledge

What's the name of the terminal device I'm using right now?

```
login as: simben90
simben90@oslab.cabrillo.edu's password:
Last login: Sat Sep  1 09:26:51 2012 from 172.30.90.83
```

```
      _
    ('v')
  //--=\
  (\_=_/)
    ~ ~
```

```
      Welcome to Opus
    Serving Cabrillo College
```

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

What's the name of the terminal device I'm using right now?

```
login as: simben90
simben90@oslab.cabrillo.edu's password:
Last login: Sat Sep  1 09:26:51 2012 from 172.30.90.83
```

```
      _
    ('v')
  //--=\
(\ _ _ /)
  ~ ~ ~ ~
```

```
Welcome to Opus
Serving Cabrillo College
```

```
Terminal type? [xterm]
Terminal type is xterm.
/home/cis90/simben $
/home/cis90/simben $ tty
/dev/pts/0
/home/cis90/simben $
```

Answer: /dev/pts/0

*Use the **tty** command
to find out*

What type of terminal am I using right now?

```
login as: simben90
simben90@oslab.cabrillo.edu's password:
Last login: Sat Sep  1 09:26:51 2012 from 172.30.90.83
```

```
      _
    ('v')
  //--=\
  (\_=_/)
    ~ ~
```

```
      Welcome to Opus
    Serving Cabrillo College
```

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

What type of terminal am I using right now?

```
login as: simben90
simben90@oslab.cabrillo.edu's password:
Last login: Sat Sep  1 09:26:51 2012 from 172.30.90.83
```

```
      _
    ('v')
  //-==-\
  (\_=_/)
    ~ ~
```

```
      Welcome to Opus
    Serving Cabrillo College
```

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

Answer: xterm

We have the answer already!

What is the hostname of the computer I'm using?

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

What is the hostname of the computer I'm using?

```
/home/cis90/simben $  
/home/cis90/simben $ hostname  
oslab.cabrillo.edu  
/home/cis90/simben $
```

Answer: oslab.cabrillo.edu

*Use the **hostname**
command to find out*

What is the name of the OS (operating System) kernel?

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

What is the name of the OS (operating System) kernel?

```
/home/cis90/simben $  
/home/cis90/simben $ uname  
Linux  
/home/cis90/simben $
```

*Use the **uname**
command to find out*

Answer: Linux

What is the name of the Linux Distribution being run?

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

What is the name of the Linux Distribution being run?

```
/home/cis90/simben $  
/home/cis90/simben $ cat /etc/release  
cat: /etc/release: No such file or directory  
/home/cis90/simben $ cat /etc/issue  
CentOS release 6.2 (Final)  
Kernel \r on \l
```

```
/home/cis90/simben $ cat /etc/*-release  
CentOS release 6.2 (Final)  
CentOS release 6.2 (Final)  
CentOS release 6.2 (Final)  
/home/cis90/simben $
```

*Use the **cat /etc/issue** or
cat /etc/*-release
commands to find out*

Answer: CentOS

What is my username and uid (user ID number)?

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

What is my username and uid (user ID number)?

```
/home/cis90/simben $  
/home/cis90/simben $ id  
uid=1001(simben90) gid=190(cis90)  
groups=190(cis90),100(users)  
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023  
/home/cis90/simben $
```

Answer: username=simben90 and the uid=1001

*Use the **id** command
to find out*

What is the name of the shell I'm using?

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

What is the name of the shell I'm using?

```
/home/cis90/simben $  
/home/cis90/simben $ ps  
  PID TTY          TIME CMD  
28237 pts/0    00:00:00 bash  
28752 pts/0    00:00:00 ps  
/home/cis90/simben $
```

Answer: bash

*Use the **ps** command to find out.*

We will soon learn another command for doing this.

Putty Tips

(Note: tty = teletype)

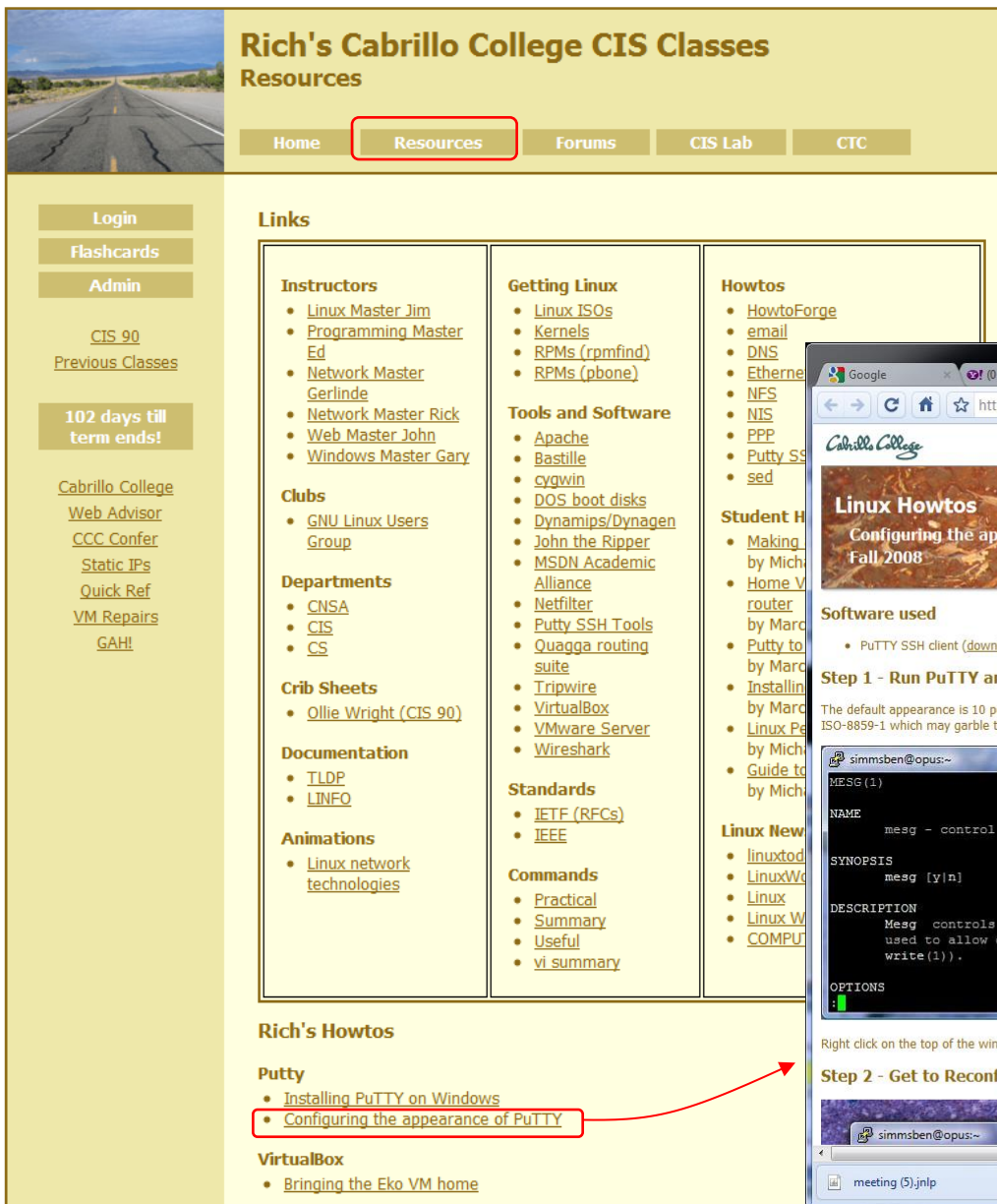
The Putty program

```

rsimms@server0-01:~
[rsimms@server0-01 rsimms]$ ls /bin
arch      cut      fgrep    ls      pwd      sync
ash       date     gawk     mail    r        r
ash.static dd      grep     mkdir   r        r
awk       df       gtar     mknod   r        r
basename dmesg   gunzip   mktemp  r        r
bash      dnsdomainname gzip      more    r        r
bash2     doexec  hostname mount    r        r
bsh       domainname igawk    mt      s        s
cat       dumpkeys ipcalc   mv      s        s
chgrp     echo    kbd_mode netstat s        s
chmod     ed      kill     nice    s        s
chown     egrep   link     nisdomainname s
cp        env     ln      pgawk   s        s
cpio      ex      loadkeys ping     s        s
csh       false   login    ps      s
[rsimms@server0-01 rsimms]$

rsimms@nosmo:~/depot/gcal-3.01/src
[rsimms@nosmo src]$ ls /bin
alsaunmute  dnsdomainname  kbd_mode  nisdomainname  sync
arch         doexec         keyctl    pgawk           tar
ash          domainname     kill      ping            tcsh
ash.static   dumpkeys      ksh       ping6           touch
awk          echo           link      ps              tracepath
basename     ed            ln        pwd             tracepath6
bash         egrep         loadkeys  red             traceroute
bsh          env           login     rm              traceroute6
cat          ex            ls        rmdir           true
chgrp        false        mail      rpm             umount
chmod        fgrep        mailx     rvi             uname
chown        gawk         mkdir     rview          unicode_start
cp           gettext      mknod     sed             unicode_stop
cpio         grep         mktemp    setfont         unlink
csh          gtar         more      setserial       usleep
cut          gunzip       mount     sh              vi
date         gzip         mt        sleep           view
dd           hostname     mv        sort            ypdomainname
df           igawk        netstat   stty            zcat
dmesg        ipcalc       nice      su
[rsimms@nosmo src]$
  
```

*Why does Putty sometimes have a **black background** and sometimes a **white background**?*



Rich's Cabrillo College CIS Classes Resources

Home **Resources** Forums CIS Lab CTC

Login
Flashcards
Admin

CIS 90
Previous Classes

102 days till term ends!

Cabrillo College
Web Advisor
CCC Confer
Static IPs
Quick Ref
VM Repairs
GAH!

Links

Instructors - Linux Master Jim - Programming Master Ed - Network Master Gerlinde - Network Master Rick - Web Master John - Windows Master Gary Clubs - GNU Linux Users Group Departments - CNSA - CIS - CS Crib Sheets - Ollie Wright (CIS 90) Documentation - TLDP - LINFO Animations - Linux network technologies	Getting Linux - Linux ISOs - Kernels - RPMS (rpmfind) - RPMS (pbone) Tools and Software - Apache - Bastille - cygwin - DOS boot disks - Dynamips/Dynagen - John the Ripper - MSDN Academic Alliance - Netfilter - Putty SSH Tools - Quagga routing suite - Tripwire - VirtualBox - VMware Server - Wireshark Standards - IETF (RFCs) - IEEE Commands - Practical - Summary - Useful - vi summary	Howtos - HowtoForge - email - DNS - Ethernet - NFS - NIS - PPP - Putty SSH - sed Student Help - Making a backup by Michael - Home V router by Marco - Putty to by Marco - Installing by Marco - Linux Pe by Michael - Guide to by Michael Linux News - linuxtoday - LinuxWorld - Linux - Linux W - COMPU
---	--	--

Rich's Howtos

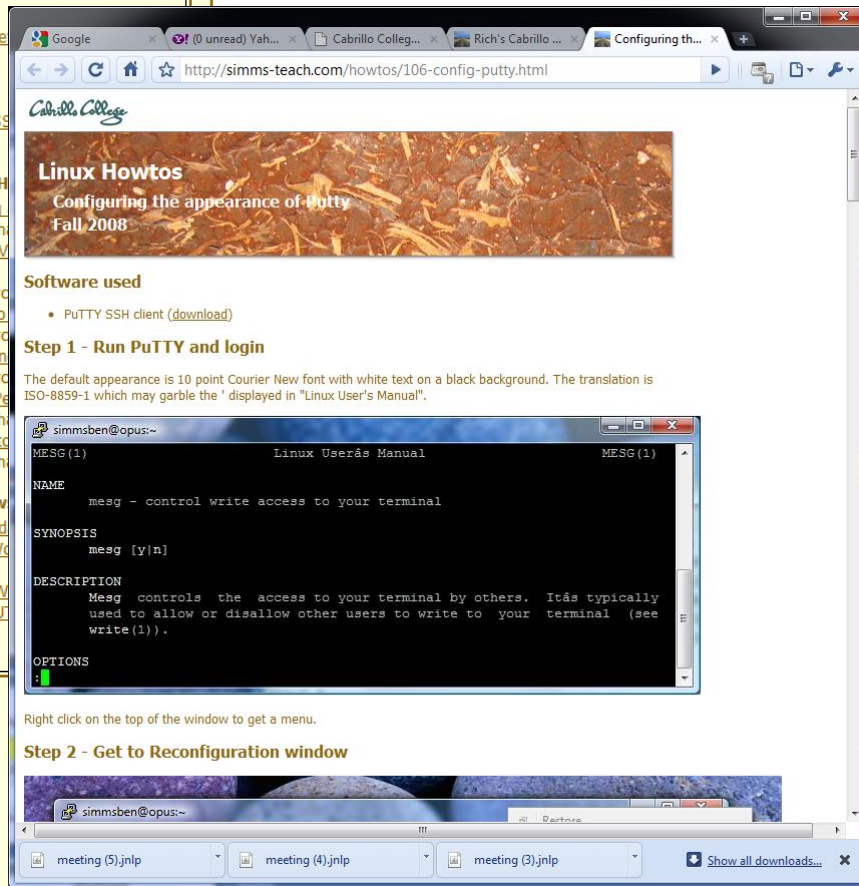
Putty

- Installing PuTTY on Windows
- Configuring the appearance of PuTTY

VirtualBox

- Bringing the Eko VM home

There is a Howto on the Resource page to walk you through customizing Putty



Linux Howtos
Configuring the appearance of PuTTY
Fall, 2008

Software used

- PuTTY SSH client (download)

Step 1 - Run PuTTY and login

The default appearance is 10 point Courier New font with white text on a black background. The translation is ISO-8859-1 which may garble the ' displayed in "Linux User's Manual".

simmsben@opus:~
MSG (1) Linux User's Manual MSG (1)

NAME
msg - control write access to your terminal

SYNOPSIS
msg [y/n]

DESCRIPTION
Msg controls the access to your terminal by others. It's typically used to allow or disallow other users to write to your terminal (see write(1)).

OPTIONS

Right click on the top of the window to get a menu.

Step 2 - Get to Reconfiguration window

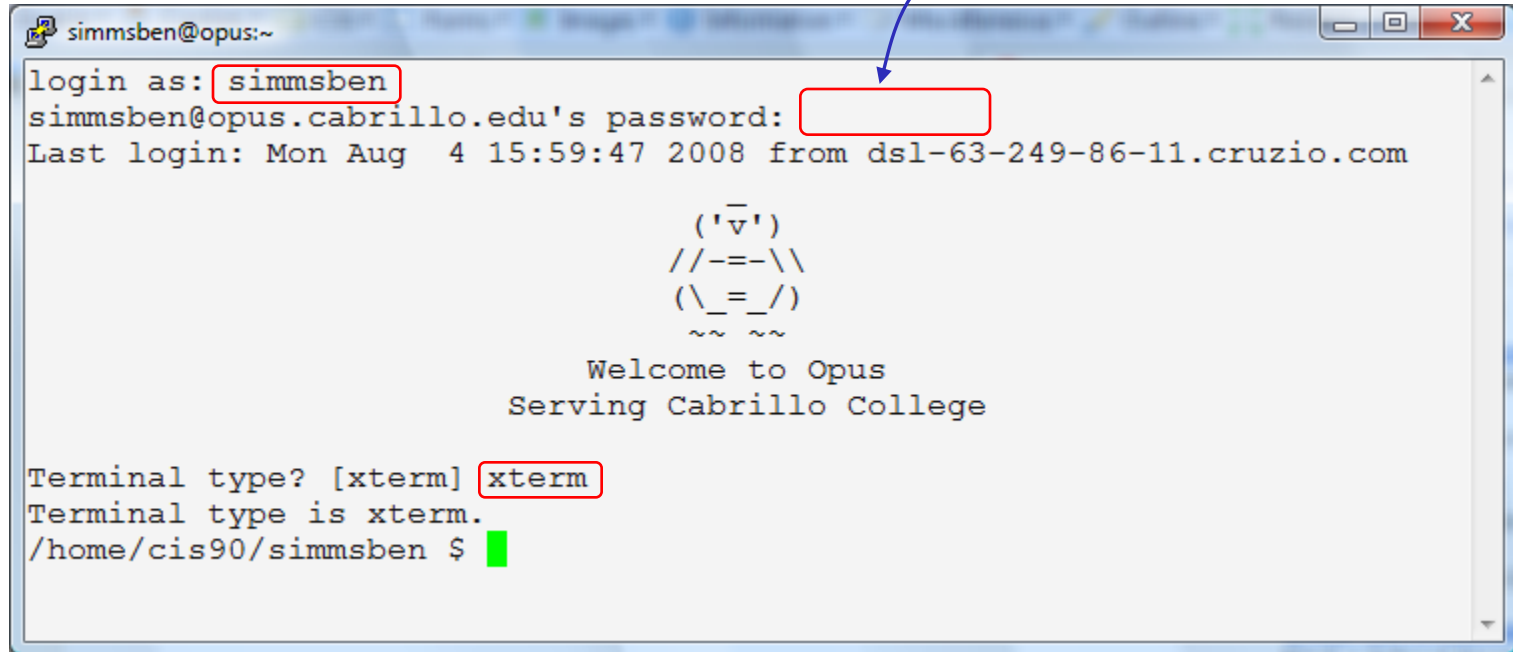
simmsben@opus:~



Logging In (A deep dive)

Logging in

Note: the password is never echoed for security reasons



```
simmsben@opus:~  
login as: simmsben  
simmsben@opus.cabrillo.edu's password:  
Last login: Mon Aug 4 15:59:47 2008 from dsl-63-249-86-11.cruzio.com  
  
      _  
    ('v')  
  //--\ \  
  (\_=_/ )  
   ~ ~ ~  
  
  Welcome to Opus  
  Serving Cabrillo College  
  
Terminal type? [xterm] xterm  
Terminal type is xterm.  
/home/cis90/simmsben $
```

always requires:

username + password + terminal type

Note: Terminal Type ≠ Terminal Device

/etc/passwd

```
simben90@oslab:~$ cat /etc/passwd
cis90@P01-Hugo ~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
```

The SUPER user

snipped

```
speech-dispatcher:x:112:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/sh
hplip:x:113:7:HPLIP system user,,,:/var/run/hplip:/bin/false
saned:x:114:123:./home/saned:/bin/false
haldaemon:x:115:125:Hardware abstraction layer,,,:/var/run/hald:/bin/false
mdm:x:116:128:MDM Display Manager:/var/lib/mdm:/bin/false
rsimms:x:1000:1000:Rich Simms,,,:/home/rsimms:/bin/bash
sshd:x:104:65534:./var/run/sshd:/usr/sbin/nologin
cis90:x:1001:1001:CIS 90 Student,,,:/home/cis90:/bin/bash
hamlet:x:1002:1002:Hamlet,,,:/home/hamlet:/bin/bash
juliet:x:1003:1003:Juliet,,,:/home/juliet:/bin/bash
romeo:x:1004:1004:Romeo,,,:/home/romeo:/bin/bash
ophelia:x:1005:1005:Ophelia,,,:/home/ophelia:/bin/bash
cis90@P01-Hugo ~$
```

Regular users

All user accounts are kept in the /etc/passwd file

Passwords are no longer kept here though!

Passwords are now kept (encrypted) in the /etc/shadow file

Login and Passwords

- 1) **init** starts up the **mingetty** program for each terminal which then prompts for the username, gets it, then starts login.

```
CentOS release 4.6 (Final)
Kernel 2.6.9-67.ELsmp on an i686
nosmo login: _
```

```
[root@nosmo ~]# ps t tty1
  PID TTY          STAT       TIME COMMAND
 3545 tty1      Ss+        0:00 /sbin/mingetty tty1
```

- 2) **login** collects the password and checks it with **/etc/passwd** and **/etc/shadow**

```
CentOS release 4.6 (Final)
Kernel 2.6.9-67.ELsmp on an i686
nosmo login: rsimms
Password: _
```

```
[root@nosmo ~]# ps t tty1
  PID TTY          STAT       TIME COMMAND
 3545 tty1      Ss+        0:00 /bin/login -
```

- 3) If a match then the shell specified in the **/etc/passwd** file is started

```
CentOS release 4.6 (Final)
Kernel 2.6.9-67.ELsmp on an i686
nosmo login: rsimms
Password:
Last login: Mon Jul  7 14:25:17 on tty1
[rsimms@nosmo ~]$ _
```

```
[root@nosmo ~]# ps t tty1
  PID TTY          STAT       TIME COMMAND
 4917 tty1      Ss+        0:00 -bash
```

/etc/passwd

*This command, which we will learn how to do later, outputs **just one line** of the /etc/passwd file on Opus*

```
/home/cis90/simben $ cat /etc/passwd | grep simben
simben90:x:1001:190:Benji Simms:/home/cis90/simben:/bin/bash
```

username

Comment

Home directory

Shell

Group ID (GID)

User ID (UID)

password (just a placeholder now)

Note the field separator used in /etc/passwd is a ":"

```
/home/cis90/simben $ id
uid=1001(simben90) gid=190(cis90) groups=190(cis90),100(users)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Now you know the source of some of the information displayed by the id command

/etc/shadow

```
cis90@P01-Kate: ~
cis90@P01-Hugo ~ $ cat /etc/shadow
cat: /etc/shadow: Permission denied
cis90@P01-Hugo ~ $ su - ← Change to root user
Password:
P01-Hugo ~ # cat /etc/shadow
root:$6$ukABmQnw$9hYrvIw6C02NfeFpipLhHO3RPJ6Ce6PaimpVCxYyGCIYW0f7PP1EEUaJZmTybAV
Bf9lzQEOM8rv.q35UONgSn0:15534:0:99999:7:::
daemon:!:15455:0:99999:7:::
bin:!:15455:0:99999:7:::
sys:!:15455:0:99999:7:::
```

All passwords are encrypted and kept in the /etc/shadow file now.

Only the root user can view this file!

snipped

```
haldaemon:!:15463:0:99999:7:::
mdm:!:15469:0:99999:7:::
rsimms:$6$Lr34V/iY$4h9JiAqOAeqY3/ovoieAgzUM8FeuVJRaPBODryjJBm6LyBOQIib0DvEEVN0Ns
eXp07votHzgAqWa93I52zmbx/:15534:0:99999:7:::
sshd:!:15536:0:99999:7:::
cis90:$6$gkVktZ1c$Ak53/yfpfALvLW06TrqaKGIVVgilKQSbd4dfvZCxdvBq5cG/YgKxbgEm2xRw1N
KkuZp600bcNOS1/u2f5S9MD/:15545:0:99999:7:::
hamlet:$6$REkRWsGt$1SIEQ2k1IgfKk0PNTSe54UMx4625operWLysAYnzFmtHX.Og3EPQjQRUT5OeP
k3GzN8fVutWWQ0TMnehvWC/11:15554:0:99999:7:::
juliet:$6$3Np10Yj1$YQM18ZzgUXDd9GghYpQ5iNzMDlhy0gBBQ050PunH1WELd7kzVZviejtsRa6w5
P5yuKLUzOuUzhPznoEJ9nudR.:15554:0:99999:7:::
romeo:$6$dJIpMMT3$9LlztGMzgm77WvH1.atsvn3RqFKGGgpdF/En5eXhclS9YkKp2ALJcUgEK8QnFK
VdOpa2dNKcrmGAa6uANMEU./:15554:0:99999:7:::
ophelia:$6$4wiI89bw$5kVgeK/.a2GDCQJBTJuqCBPUT7z.136R6yN3SbBpcPJ83QsvBNm9HcDvUxMu
/wiHKRLmBOaaoQD.Tu4SfysKx/:15554:0:99999:7:::
P01-Hugo ~ #
```

Class Activity

```
/home/cis90/simben $ cat /etc/passwd | grep simben  
simben90:x:1001:190:Benji Simms:/home/cis90/simben:/bin/bash
```

username → *User ID (UID)* → *Group ID (gid)* → *Comment* → *Home directory* → *Shell*

password (just a placeholder now)

Note the field separator used in /etc/passwd is a ":"

1. cat /etc/passwd

- Find your own username
- Compare your /etc/passwd home directory with your prompt
- Compare your /etc/passwd shell with output from the ps command
- Compare your /etc/passwd uid and gid with output from the **id** command

2. cat /etc/shadow

What happens when you try to look at /etc/shadow?

Your Opus Password

Your Opus password

- Strong passwords are critical!
- **Botnets** and **ne-er-do-wells** are constantly attempting to break into computers attached to the Internet!
(Even my little Frodo VM at home)

They never stop trying

*The ne'er-do-wells trying to break in ...
this is why you need strong passwords*

----- SSHD Begin -----

```
SSHD Killed: 1 Time(s)
SSHD Started: 1 Time(s)
Disconnecting after too many authentication failures for user:
guest90 : 1 Time(s)
```

Failed logins from:

```
76.254.22.196 (adsl-76-254-22-196.dsl.pltn13.sbcglobal.net): 2 times
201.7.115.194 (201-7-115-194.spopa302.ipd.brasiltelecom.net.br): 2135 times
210.240.12.14: 20 times
```

Illegal users from:

```
201.7.115.194 (201-7-115-194.spopa302.ipd.brasiltelecom.net.br): 564 times
210.240.12.14: 42 times
```

```
Users logging in through sshd:
guest:
  76.254.22.196 (adsl-76-254-22-196.dsl.pltn13.sbcglobal.net): 2 times
jimg:
  70.132.20.25 (adsl-70-132-20-25.dsl.snfc21.sbcglobal.net): 7 times
ordazedw:
  76.254.22.196 (adsl-76-254-22-196.dsl.pltn13.sbcglobal.net): 1 time
root:
  63.249.86.11 (dsl-63-249-86-11.cruzio.com): 3 times
  70.132.20.25 (adsl-70-132-20-25.dsl.snfc21.sbcglobal.net): 1 time
rsimms:
  63.249.86.11 (dsl-63-249-86-11.cruzio.com): 2 times
```

From a logwatch report showing malicious attempts to break into Opus

They never stop trying

The firewall on Opus slows down but does not end the attacks

Failed logins from:

```
122.249.183.95 (x183095.ppp.asahi-net.or.jp): 3 times
218.64.5.131 (131.5.64.218.broad.nc.jx.dynamic.163data.com.cn): 3
times
```

Illegal users from:

```
78.46.83.76 (static.76.83.46.78.clients.your-server.de): 3 times
218.4.157.178: 3 times
```

```
pam_succeed_if(sshd:auth): error retrieving information about user
teamspeak : 1 time(s)
reverse mapping checking getaddrinfo for
131.5.64.218.broad.nc.jx.dynamic.163data.com.cn failed - POSSIBLE
BREAK-IN ATTEMPT! : 3 time(s)
pam_succeed_if(sshd:auth): error retrieving information about user ts
: 2 time(s)
pam_succeed_if(sshd:auth): error retrieving information about user
plcmspip : 2 time(s)
pam_succeed_if(sshd:auth): error retrieving information about user
PlcmSpIp : 1 time(s)
```

We used to get up thousands of attempts every day until we made some changes to the firewall on Opus. Attacks always would come from different computers around the world.

/var/log/wtmp and var/log/btmp

```
[root@opus log]# lastb | sort | cut -f1 -d' ' | grep -v ^$ | uniq -c > bad
[root@opus log]# sort -g bad > bad.sort
[root@opus log]# cat bad.sort | tail -50
```

471	ftp	
472	public	
490	test	
490	tomcat	610 test
498	user	656 noc
506	service	686 www
508	mike	690 postfix
508	username	723 john
524	cyrus	734 testing
530	pgsql	738 adam
532	test1	746 alex
544	master	754 info
554	linux	798 tester
554	toor	832 library
576	paul	935 guest
584	support	990 admin
590	testuser	1002 office
604	irc	1022 temp
		1070 ftpuser
		1138 webadmin
		1298 nagios
		1332 web
		1374 a
		1384 student
		1416 postgres
		1690 user
		1858 oracle
		1944 mysql
		2086 webmaste
		5324 test
		10803 root
		10824 admin
		18679 root
		24064 root

```
[root@opus log]#
```

Top 50 usernames used by the ne'er-do-wells

How to make a strong password

- The longer the better (8 or more characters)
- Not in any dictionary
- Use upper case, lowercase, punctuation, digits
- Something you can remember
- Keep it secret
- Change when compromised

Wh0le#!!

(Whole sh'bang)

KuKu4(co)2

(Cuckoo for Cocoa Puffs)

#0p&s@ve

(shop and save)

Idl02\$da

(I do laundry on Tuesday)

How to change your password on Opus

```
/home/cis90/simmsben $ passwd
Changing password for user simben90.
Changing password for simben90
(current) UNIX password: 
New UNIX password: 
Retype new UNIX password: 
passwd: all authentication tokens updated successfully.
/home/cis90/simmsben $
```

Note, the passwords are not echoed as you type them.

This changes your password on Opus only (not on Vlab or the forum)

John the Ripper

An open source cracker that tries common passwords first followed by a brute force dictionary attack



john-1.7.9/run/password.lst has most popular passwords to try first



Housekeeping

Housekeeping

1. Send me your student survey
2. Lab 1 submittal due by 11:59PM tonight
3. Last day to add is Saturday 2/23

Turn OFF the recording

Roll Call

Turn recording back ON

CIS 90 – Code Names

Lord of the Rings Characters

Current Progress

Code Name	Grading Choice	Q1	Q2	Q3	Q4
Max Points		3	3	3	3
aragorn	Grade				
arwen	Grade				
balrog	Grade				
boromir	Grade				
denethor	Grade				
dwalin	Grade				
elrond	Grade				
eomer	Grade				
eowyn	Grade				
faramir	Grade				
frodo	Grade				
galadriel	Grade				
gimli	Grade				
glorfindel	Grade				
ioeth	Grade				
legolas	Grade				
lobelia	Grade				
nazgul	Grade				
pippin	Grade				
saruman	Grade				
sauron	Grade				
theoden	Grade				
treebeard	Grade				

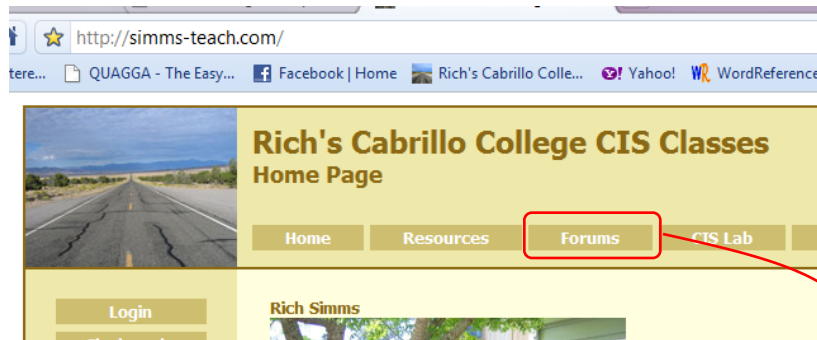
Everyone who is enrolled for this course will be assigned a LOR code name.

I will use your grading choice on the survey you send me (you can change your mind later)

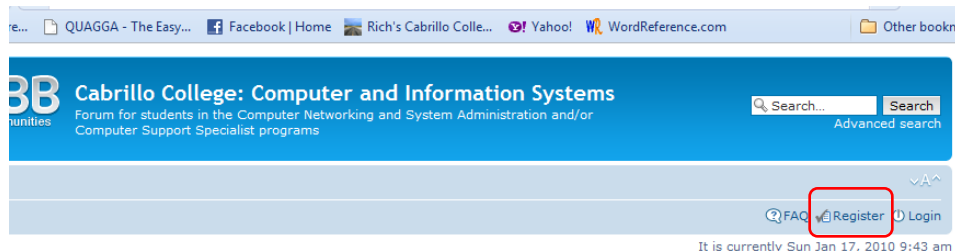
*I'll start sending out code names tomorrow for **everyone who sends or has sent me their survey.***

Forum spambot counter-measures

simms-teach.com



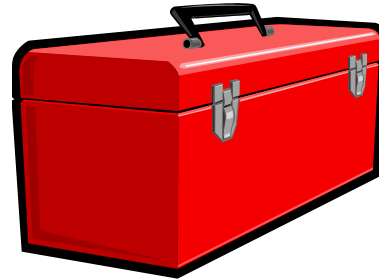
oslab.cabrillo.edu/forum



To Register:

1. Browse to the forum
2. Click on  Register
3. Review and agree to terms
4. Your **username** must be your **first and last name separated by a space**
 - e.g. Rich Simms
Not rsimms71 or richsimms

- *New security question in place*
- *Each registration must be manually approved by your instructor*



More commands for your toolbox



New commands for this lesson

cat *filename*

*print file(s) ("cat" comes from **concatenate**)*

cd *[pathname]*

Change to a new directory

ls *[pathname]*

List files in a directory

echo *string*

Print string (on screen)

file *pathname*

Show additional file information

type *command*

Shows where command resides on the path

man *command*

Show manual page for a command

bc

Binary calculator

banner *text*

Make a banner

passwd

Change password

apropos *command*

Looks up references in the whatis database

cat command

Concatenate files and print on the standard output

```
/home/cis90/simben $ cat 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.
```

< snipped >

```
Wait a minute!  It's stopped hailing!  Guys are swimming!  
Guys are sailing!  Playing baseball, gee that's better!  
Mother, Father, kindly disregard this letter.
```

Alan Sherman

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

cd and ls commands

Change directory and list directory contents

/home/cis90/simben \$ **cd** *Using **cd** by itself with no argument will return you to your home directory*

/home/cis90/simben \$ **ls** *List files in current directory*

bigfile	lab01-submitted	letter	Poems	small_town	timecal
bin	lab01-submitted.bak	log	proposal1	spellk	what_am_i
empty	Lab2.0	Miscellaneous	proposal2	text.err	
Hidden	Lab2.1	mission	proposal3	text.fxd	

/home/cis90/simben \$ **cd Poems/** *Change to the Poems directory*

/home/cis90/simben/Poems \$ **ls**

ant Blake nursery Shakespeare twister Yeats

/home/cis90/simben/Poems \$

Notice how your prompt changes when changing into the Poems directory

ls command

List directory contents

```
/home/cis90/simben $ ls
```

```
bigfile  Lab2.0          mission    proposal3  text.fxd
bin      Lab2.1              Poems     small_town  timecal
empty    letter              proposal1 spellk      what_am_i
Hidden   Miscellaneous        proposal2 text.err
```

If no argument is specified, the current directory is listed

```
/home/cis90/simben $ ls Poems/
```

```
ant  Blake  nursery  Shakespeare  twister  Yeats
```

If one or more directories are specified as arguments then they will be listed

```
/home/cis90/simben $ ls /bin/uname  
/bin/uname
```

If one or more filenames are specified as arguments then those filenames will be listed

Regular files show as black, directories show as blue and executable programs/scripts show as green

echo command

Echo (output) the arguments on the command line

```
/home/cis90/simben $ echo hello rich  
hello rich
```

```
/home/cis90/simben $ echo 123  
123
```

```
/home/cis90/simben $ echo 1 2 3  
1 2 3
```

file command

Show extended file information

```
/home/cis90/simben $ file letter
```

```
letter: ASCII English text
```

```
/home/cis90/simben $ file Miscellaneous/
```

```
Miscellaneous/: directory
```

```
/home/cis90/simben $ file timecal
```

```
timecal: shell archive or script for antique kernel text
```

type command

Locate where a command resides on your path

```
[rsimms@opus run]$ type cal  
cal is /usr/bin/cal
```

*The **cal** command is on the user's path and is located in the /usr/bin directory*

```
/home/cis90/simben $ type bogus  
-bash: type: bogus: not found
```

*The **bogus** command is not on the user's path*

```
[rsimms@opus run]$ type uname cal  
uname is /bin/uname  
cal is /usr/bin/cal
```

*Both **uname** and **cal** commands are on the user's path. **uname** is in the /bin directory and **cal** is in the /usr/bin directory*

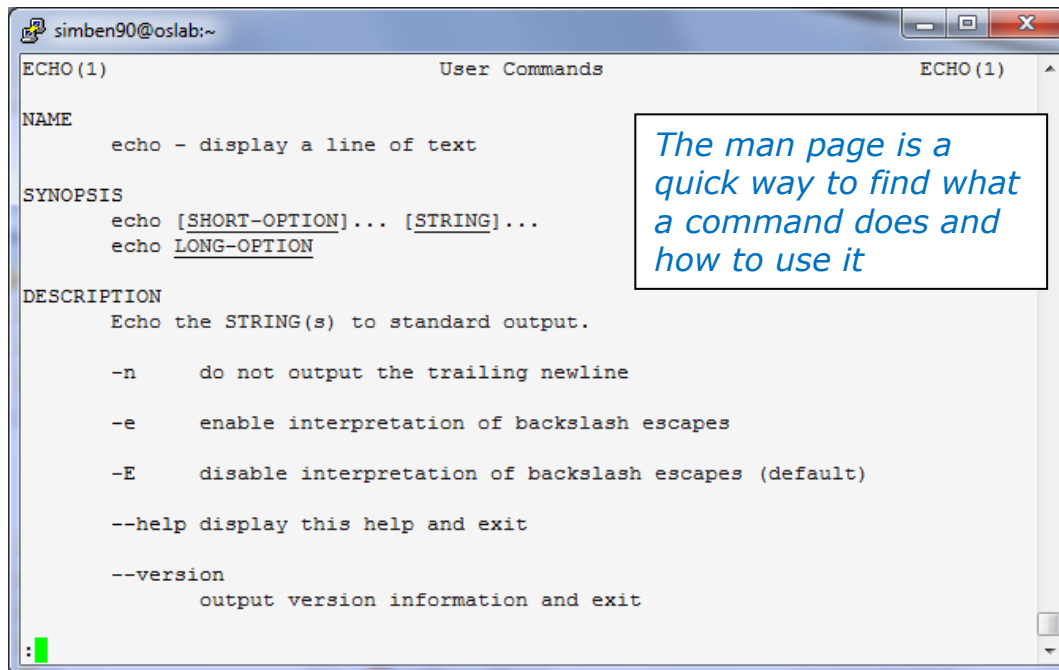
name of the file (command/program)

name of the directory where file is found

man command

Show the manual page (documentation) for a command

```
/home/cis90/simben $ man echo
```



The man page is a quick way to find what a command does and how to use it



Use these keys to scroll

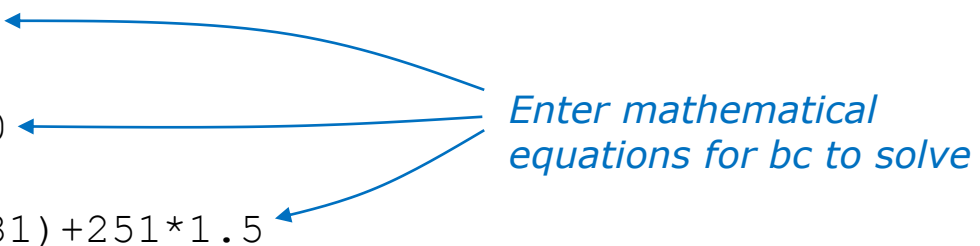


Use q key to quit

bc command

A binary calculator

```
/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
3*30
90
(3*31)+251*1.5
469.5
quit
/home/cis90/simben $
```



Enter mathematical equations for bc to solve

*Use quit to
end program*

banner command

Make a banner

```
/home/cis90/simben $ banner I Love Linux
```

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

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

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

*Similar to echo
command but
outputs banner sized
letters instead*

apropos command

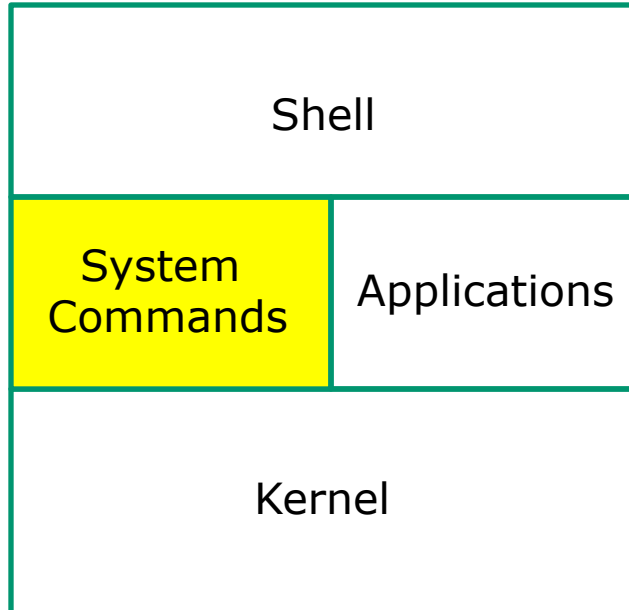
apropos - search the whatis database for strings

```
/home/cis90/simben $ apropos echo
echo                (1)  - display a line of text
echo                (1p) - write arguments to standard output
echo [builtins]    (1)  - bash built-in commands, see bash(1)
lessecho            (1)  - expand metacharacters
pam_echo            (8)  - PAM module for printing text messages
ping                (8)  - send ICMP ECHO_REQUEST to network hosts
ping6 [ping]        (8)  - send ICMP ECHO_REQUEST to network hosts
/home/cis90/simben $
```

Where are the commands?

UNIX/Linux Architecture

System Commands



- 100's of system commands and utilities .
- Commands like **ls** (list directories), **cat** (print a file), **rm** (remove a file), ... etc.
- Utilities like **vi** (text editor), **sort** (sorts file contents), **find** (searches), ... etc.
- Larger utilities like **sendmail** (email), **tar** (backup), **tcpdump** (sniffer), ... etc.
- Administrative utilities like **useradd**, **groupadd**, **passwd** (change password), ... etc.



Executable binary code (programs) or scripts

There are lots and LOTS of commands & utilities in the four "bin" (binary) directories

```
rsimms@server0-01~  
[rsimms@server0-01 rsimms]$ ls /usr/sbin  
accept          ntpd  
adduser         ntpdate  
adsl-connect    ntpdc  
adsl-setup      ntp-genkeys  
adsl-start      ntpq  
adsl-status     ntptime  
adsl-stop       ntptimeset  
alternatives    ntptrace  
anacron         ntp-wait  
apmd            ntsysv  
arping          packer  
atd             pcbitctl  
atrun           ping6  
authconfig      pmap_dump  
autocount       pmap_set  
avmcpictctl    pppd  
bonobo-activation-sysconf pppdump  
build-locales-archive pppoe  
camel-index-control pppoe-relay  
camel-lock-helper pppoe-server  
capipinit       pppoe-sniff  
chat            pppstats  
Chkfontpath     praliases
```

/bin

```
/usr/bin
```

```
/sbin
```

```
/usr/sbin
```

The /bin directory

Use **ls /bin** to view

```
simben90@oslab:~  
/home/cis90/simben $ ls /bin  
alsaunmute      dbus-monitor    hostname        netstat         sort  
arch             dbus-send       ipcalc          nice            stty  
awk             dbus-uuidgen    iptables-xml    nisdomainname   su  
basename        dd              kbd_mode        ping            sync  
bash            df              keyctl          ping6           tar  
cat             dmesg           kill            plymouth        taskset  
cgclassify       dnsdomainname  link            ps              tcsh  
cgcreate        domainname     ln              pwd             touch  
cgdelete        dumpkeys       loadkeys        raw            tracepath  
cgexec          echo           login           rbash           tracepath6  
cgget           ed             ls              readlink        traceroute  
cgset           egrep          lsblk           red             traceroute6  
cgsnapshot      env            lscgroup        redhat_lsb_init true  
chgrp           ex             lssubsys        rm              umount  
chmod           false          mail            rmdir           uname  
chown           fgrep          mailx           rnano           unicode_start  
cp             find           mkdir           rpm            unicode_stop  
cpio           findmnt        mknod           rvi            unlink  
csh            gawk           mktemp          rview          usleep  
cut            gettext        more            sed            vi  
dash           grep           mount           setfont        view  
date           gtar           mountpoint      setserial      ypdomainname  
dbus-clean-up-sockets gunzip         mv             sh             zcat  
dbus-daemon     gzip          nano            sleep
```

/bin has essential commands used by everyone.

*Can you find the Lesson 1 **date**, **hostname**, **ps** and **uname** commands?*

*Can you find the **bash** shell?*

Commands are either program or script files that can be executed

The /usr/bin directory

Use **ls /usr/bin** to view

```
simben90@oslab:~  
/home/cis90/simben $ ls /usr/bin  
[  
a2p          gst-feedback-0.10    powertop  
ab           gst-inspect          ppdc  
abrt-action-analyze-backtrace  gst-inspect-0.10    ppdhtml  
abrt-action-analyze-c          gst-launch           ppdi  
abrt-action-analyze-core       gst-launch-0.10     ppdmerge  
abrt-action-analyze-oops       gst-typefind        pppdo  
abrt-action-analyze-python     gst-typefind-0.10   ppl-config  
abrt-action-generate-backtrace  gst-xmlinspect      ppm2tiff  
abrt-action-install-debuginfo   gst-xmlinspect-0.10 pr  
abrt-action-list-dsos           gst-xmllaunch       precat  
abrt-action-save-package-data   gst-xmllaunch-0.10 pre-grohtml  
abrt-action-trim-files          gtbl                preunzip  
abrt-cli                       gtk-query-immodules-2.0-32 prezip  
abrt-dump-oops                 gtk-update-icon-cache prezip-bin  
                                gtroff              printafm
```

There are a "ton" of additional commands (programs) in this directory.

You will need to scroll through a lot of pages to see them all!

snipped

```
grotty       png2theora     zforce  
groups       pnm2ppa        zgrep  
gs           pod2html       zip  
gsbj         pod2latex      zipcloak  
gsdj         pod2man        zipgrep  
gsdj500     pod2text      zipinfo  
gsj          pod2usage     zipnote  
gslp        podchecker    zipsplit  
gsnd        podselect     zless  
gsoclim     POST          zmore  
gstack      post-grohtml  znew  
gst-feedback poweroff       zsoelim  
/home/cis90/simben $
```

*Can you find the Lesson 1 **cal**, **clear**, **id**, **ssh**, **tty**, and **who** commands we used in Lab 1?*

The /sbin directory

Use **ls /sbin** to view this directory

```

simben90@oslab:~
/home/cis90/simben $ ls /sbin
accton          fsck.cramfs      kpartx          nameif           scsi_id
addpart         fsck.ext2        ldconfig        netreport        securetty
agetty          fsck.ext3        load_policy     new-kernel-pkg  service
alsactl         fsck.ext4        logsav          nologin          setfiles
arp             fsck.ext4dev     losetup         pam_console_apply setpci
arping          fsck.msdos       lsinitrd        pam_tally2       setregdomain
audispd         fsck.vfat        lsmod           pam_timestamp_check setsysfont
auditctl        fsfreeze         lspci           parted           sfdisk
auditd          fstab-decode     lspcmcia        partprobe        sgpio
aureport        fstrim           lvchange        partx            shutdown
ausearch        fuser            lvconvert       pccardctl        slattach
autrace         genhostid        lvcreate        pidof            sln
badblocks       getkey           lvdisplay       pivot_root       start
blkid           grub             lvextend        plipconfig       start_udev
blockdev        grubbb          lum             plmouthd         status

snipped

dumpe2fs        iptables-restore mkfs.ext4        restorecon       vgimport
e2fsck          iptables-save    mkfs.ext4dev     rfkill           vgimportclone
e2image         iptunnel         mkfs.msdos       rmmod            vgmerge
e2label         iw               mkfs.vfat        rmt              vgmknodes
e2undo          iwconfig         mkhomedir_helper rngd              vgreduce
ether-wake       ivervent         mkinitrd         route            vgrename
ethtool         iwgetid          mkswap           rpcbind          vgs
faillock        iwlist           modinfo          rpc.statd         vgsan
fdisk           iwpriv           modprobe         rrestore         vgsplit
findfs          iwsby           mount.cifs        rsyslogd         weak-modules
fixfiles        kdump           mount.nfs         rtmon            wipefs
fsadm           kexec           mount.nfs4        runlevel
fsck            killall5         mount.tmpfs       runuser
/home/cis90/simben $

```

These are essential commands and utilities used by system administrators.

*This is where the **chkconfig**, **ifconfig** and **iptables** commands are found.*

You will learn how to use these commands in CIS 191 and CIS 192.

Use **ls /usr/sbin** to view this directory

snipped

You will learn how to use these commands in CIS 191 and CIS 192.

Programs

Binary code vs text
scripts

UNIX commands & utilities are executable programs

A program can be binary code:

- Binary machine code is unprintable. A programmer must use hex dumps to examine binary code.
- Binary machine code executes very quickly and is targeted for a specific CPU instruction set.
- The binaries are produced by compiling source code written in a higher level language such as C, or C++.

A program can be a text-based script:

- A script can be directly viewed and printed.
- A script does not need to be compiled. It is interpreted on the fly and because of that doesn't run as fast as binary code.
- Common scripting languages include bash, perl and python.

Two example programs: apropos and cal

Lets take a deep dive on two random commands:

apropos - searches the whatis database for a string of text

cal - prints a calendar

*I'll be using this graphic to indicate
a program that has been loaded
into memory to be executed*





apropos

What do they do?



cal

The **apropos** command looks up the argument it gets in the *whatis* database.

```
/home/cis90/simben $ apropos uname
oldolduname [obsolete] (2) - obsolete system calls
olduname [obsolete] (2) - obsolete system calls
uname (1) - print system information
uname (1p) - return system name
uname (2) - get name and information about current kernel
uname (3p) - get the name of the current system
```

The **cal** prints a calendar

```
/home/cis90/simben $ cal
February 2012
Su Mo Tu We Th Fr Sa
      1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29
```

Where are the programs located?



apropos



cal

Use the **type** command see where other commands are located:

```
/home/cis90/simben $ type apropos cal  
apropos is hashed (/usr/bin/apropos)  
cal is /usr/bin/cal
```

The **apropos** and **cal** commands are used as arguments on the **type** command

They are both in the /usr/bin directory.

Note: Sometimes you will see "Hashed" which means the command has been run previously and its location on the path has been temporarily "remembered" to speed up subsequent path searches for the same command.

Listing the program files



apropos



cal

- 1) Change into the /usr/bin directory:

```
/home/cis90/simben $ cd /usr/bin
```

*The /usr/bin pathname is used as and argument on the **cd** command*

- 2) List the two files in that directory:

```
/usr/bin $ ls apropos cal  
apropos  cal
```

*The **apropos** and **cal** commands are used as arguments on the **ls** command*

- 3) Use the **-l** option on the **ls** command to show additional information:

```
/usr/bin $ ls -l apropos cal  
-rwxr-xr-x 1 root root 1786 Jul 12 2006 apropos  
-rwxr-xr-x 1 root root 18764 Jul 3 2009 cal
```

Note the execute permissions set (more on this later)

Getting more information on the program files



apropos



cal

```
/usr/bin $ file apropos
```

```
apropos: Bourne shell script text executable
```

```
/usr/bin $
```

```
/usr/bin $ file cal
```

```
cal: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV),  
for GNU/Linux 2.6.9, dynamically linked (uses shared libs),  
for GNU/Linux 2.6.9, stripped
```

```
/usr/bin $
```

*The **file** command shows that **apropos** is a shell script and **cal** is binary code (has been compiled from higher level source code)*



cal
(binary code)

The **cat** command can print the apropos file because it is a readable (and editable) **ASCII** script

The **cat** command "chokes" trying to print the **binary** cal file.

That's because binary files contain unprintable characters.

How binary programs are created



cal

From: gcal-3.01.tar.gz

```
rsimms@nosmo:~/depot/gcal-3.01/src
[rsimms@nosmo src]$ head -50 gcal.c
/*
 * gcal.c: Main part which controls the extended calendar program.
 *
 * Copyright (c) 1994, 95, 96, 1997, 2000 Thomas Esken
 *
 * This software doesn't claim completeness, correctness or usability.
 * On principle I will not be liable for ANY damages or losses (implicit
 * or explicit), which result from using or handling my software.
 * If you use this software, you agree without any exception to this
 * agreement, which binds you LEGALLY !!
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the 'GNU General Public License' as published by
 * the 'Free Software Foundation'; either version 2, or (at your option)
 * any later version.
 *
 * You should have received a copy of the 'GNU General Public License'
 * along with this program; if not, write to the:
 *
 * Free Software Foundation, Inc.
 * 59 Temple Place - Suite 330
 * Boston, MA 02111-1307, USA
 */

static char rcsid[]="$Id: gcal.c
```

*Note: The **cal** binary code resulted from compiling the original gcal.c source code.*

```
rsimms@nosmo:~/depot/gcal-3.01/src
[rsimms@nosmo src]$ file /usr/bin/cal
/usr/bin/cal: ELF 32-bit LSB executable, Intel 80386, version 1
(SYSV), for GNU/Linux 2.2.5, dynamically linked (uses shared lib
s), stripped
[rsimms@nosmo src]$
```

Because GNU Linux software is licensed under the GPL you can make your own custom version of the commands or the kernel!

FYI

See this forum post from a previous class for an example of obtaining the source code for a Linux command and modifying it:

<http://oslab.cabrillo.edu/forum/viewtopic.php?f=31&t=683&p=2774>

Lab #2...even though 'info uname' output states...

by Dan McNamara » Fri Feb 18, 2011 12:53 pm

Hi Folks,

Does anyone happen to know if there are ways to manipulate output from uname such that it is listed in the order that I want it to be? Under 'Commands' in Lab #2, question 11, we are asked what options would we use to display just the operating system, its kernel release numbers and the machine's network node hostname. I got that okay. However, what if I wanted the output to display following the constructs of the question, i.e.:

```
opus.cabrillo.edu 2.6.18-164.el5 GNU/Linux (the default)
```

```
GNU/Linux 2.6.18-164.el5 opus.cabrillo.edu (what I'd like it to be)
```

Doing a 'man uname' doesn't cover this but 'info uname' states:

If multiple options or '-a' are given, the selected information is printed in this order:

```
KERNEL-NAME NODENAME KERNEL-RELEASE KERNEL-VERSION  
MACHINE PROCESSOR HARDWARE-PLATFORM OPERATING-SYSTEM
```

I can live with the default output as it does answer the question...it just kind of bugs me that it's not in the order that I would prefer. Mixing the order of the options has no effect on the default output.

Just wondering....



Dan McNamara

Posts: 38
Joined: Fri Feb 04, 2011 5:21 pm

*It all started when Dan did Lab 2 and wanted to change the way **uname** ordered its output!*



Inputs to programs (commands and scripts)

You will get these questions when you submit Lab 2

Name a UNIX command that gets its input only from the command line?

Name an interactive command that reads its input from the keyboard?

Name a UNIX command that gets its input from the Operating System?

Name a UNIX command that gets its input only from the command line?

```
/home/cis90/simmen $ echo hello world  
hello world
```

```
/home/cis90/simben $ banner hello world  
#       # ##### #       #       #####  
#       # #       #       #       #  
#       # #       #       #       #  
##### ##### #       #       #  
#       # #       #       #       #  
#       # #       #       #       #  
#       # ##### ##### #####  
  
#       # ##### ##### #       #####  
# # # # # # # # # # # # # #  
# # # # # # # # # # # # # #  
# # # # # ##### # # # # #  
# # # # # # # # # # # # # #  
# # # # # # # # # # # # # #  
## ## ##### # # #####
```

*The **echo** and **banner** commands are examples of commands that get their input from the command line*

Name an interactive command that reads its input from the keyboard?

```
/home/cis90/simmsben $ bc
bc 1.06
Copyright 1991-1994, 1997, 1998, 2000 Free
Software Foundation, Inc.
This is free software with ABSOLUTELY NO
WARRANTY.
For details type `warranty'.
2+2
4
500-200+3
303
sqrt(64)
8
quit
```

```
/home/cis90/simmsben $ passwd
Changing password for user simmsben.
Changing password for simmsben
(current) UNIX password:
New UNIX password:
BAD PASSWORD: is too similar to the old
one
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated
successfully.
```

*The **bc** (binary calculator) and **passwd** commands are examples of interactive commands that read their input from the keyboard*

Name a UNIX command that gets its input from the Operating System?

```
/home/cis90/simben $ who
dycktim pts/1      2010-09-07 17:07 (nosmo-nat.cabrillo.edu)
root :0           2009-12-18 17:30
velasoli pts/2      2010-09-07 17:08 (adsl-35-201-114-102.dsl.net)
guest90 pts/3       2010-09-07 16:56 (nosmo-nat.cabrillo.edu)
rsimms pts/4       2010-09-07 15:54 (dsl-45-78-13-81.dhcp.com)
guest90 pts/5       2010-09-07 16:59 (nosmo-nat.cabrillo.edu)
watsohar pts/6      2010-09-07 17:03 (nosmo-nat.cabrillo.edu)
swansgre pts/7      2010-09-07 17:10 (nosmo-nat.cabrillo.edu)
guest90 pts/8       2010-09-07 17:10 (nosmo-nat.cabrillo.edu)
abbenste pts/9      2010-09-07 17:11 (nosmo-nat.cabrillo.edu)
```

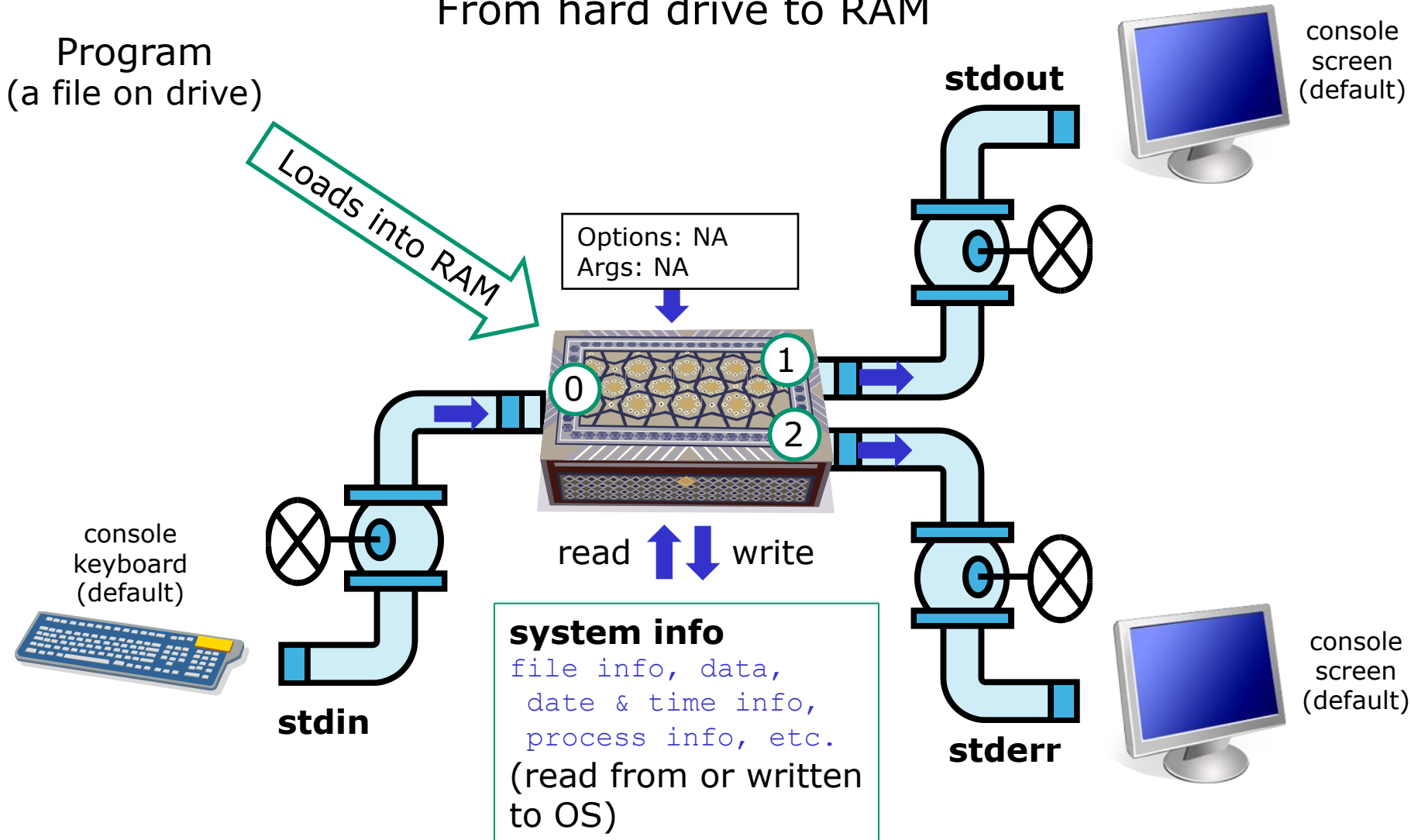
```
/home/cis90/simben $ uname
Linux
```

*The **who** and **uname** commands are examples of commands that get their input from the Operating System*

Drill Down on running programs

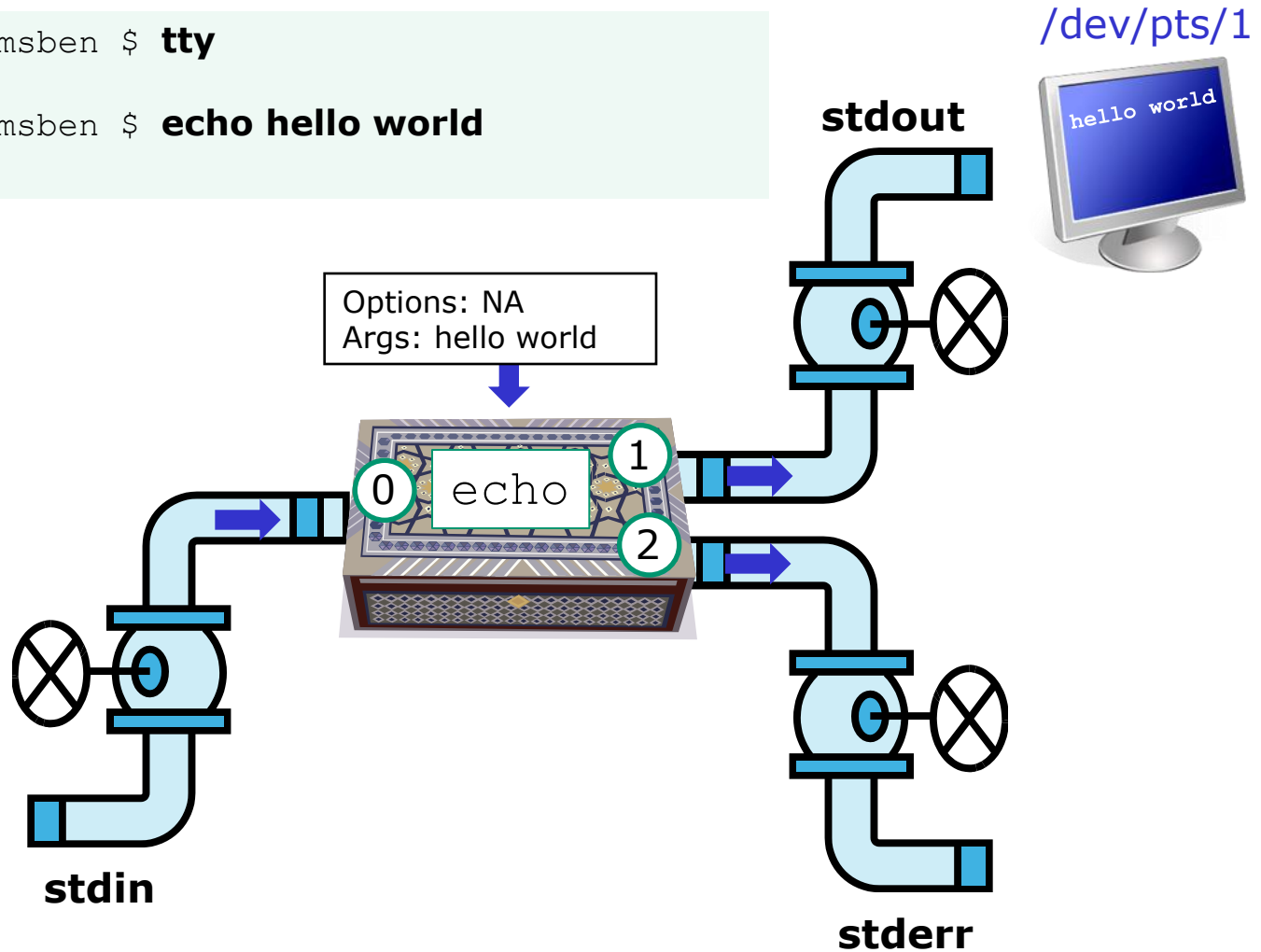
The next slides are a preview of future lessons on processes ... for now just you don't need to understand all the ins and outs of how this works.

Program to Process From hard drive to RAM



echo command

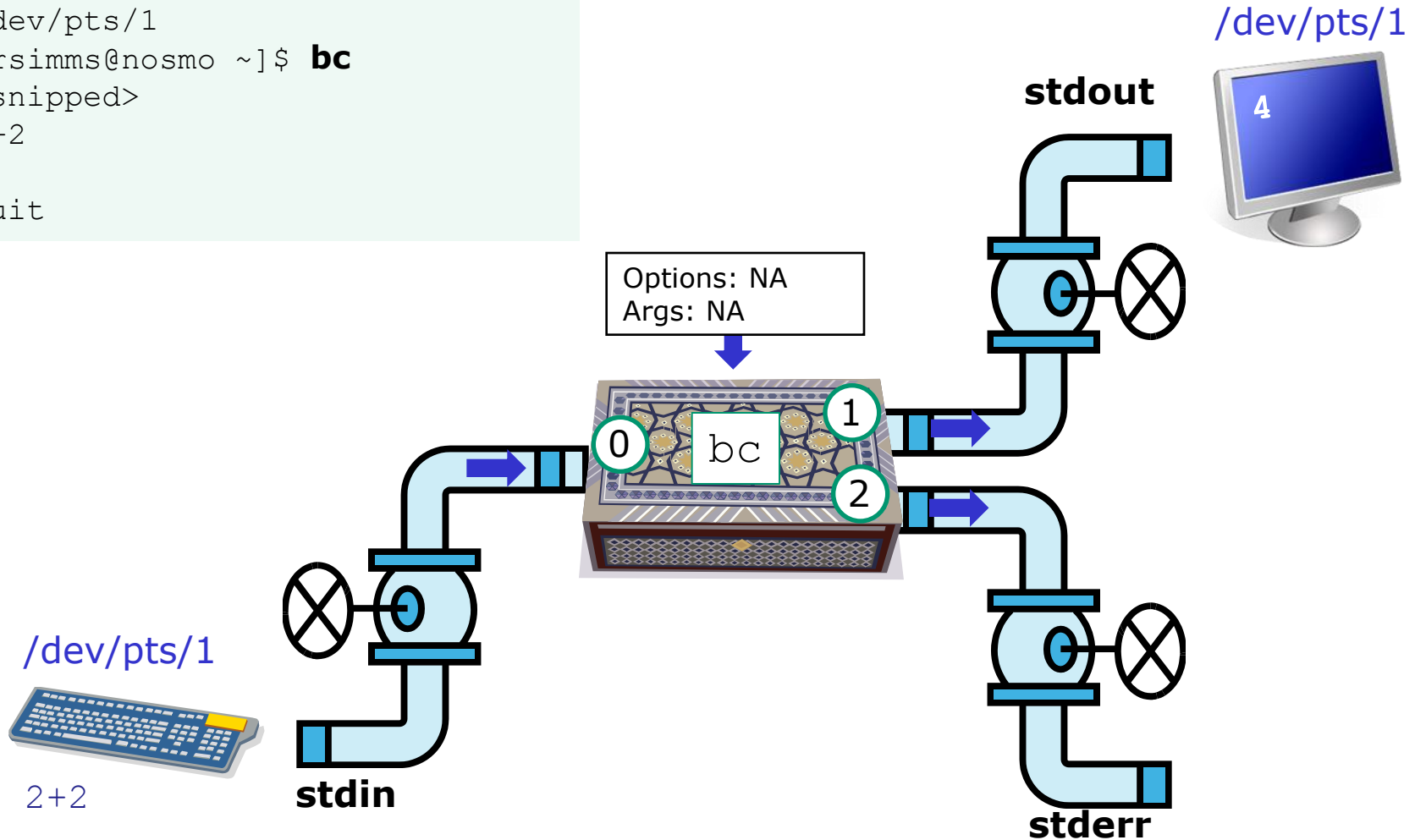
```
/home/cis90/simmsben $ tty
/dev/pts/1
/home/cis90/simmsben $ echo hello world
hello world
```



*The **echo** command is an example of a command that gets its input from the command line*

bc command

```
[rsimms@nosmo ~]$ tty
/dev/pts/1
[rsimms@nosmo ~]$ bc
<snipped>
2+2
4
quit
```



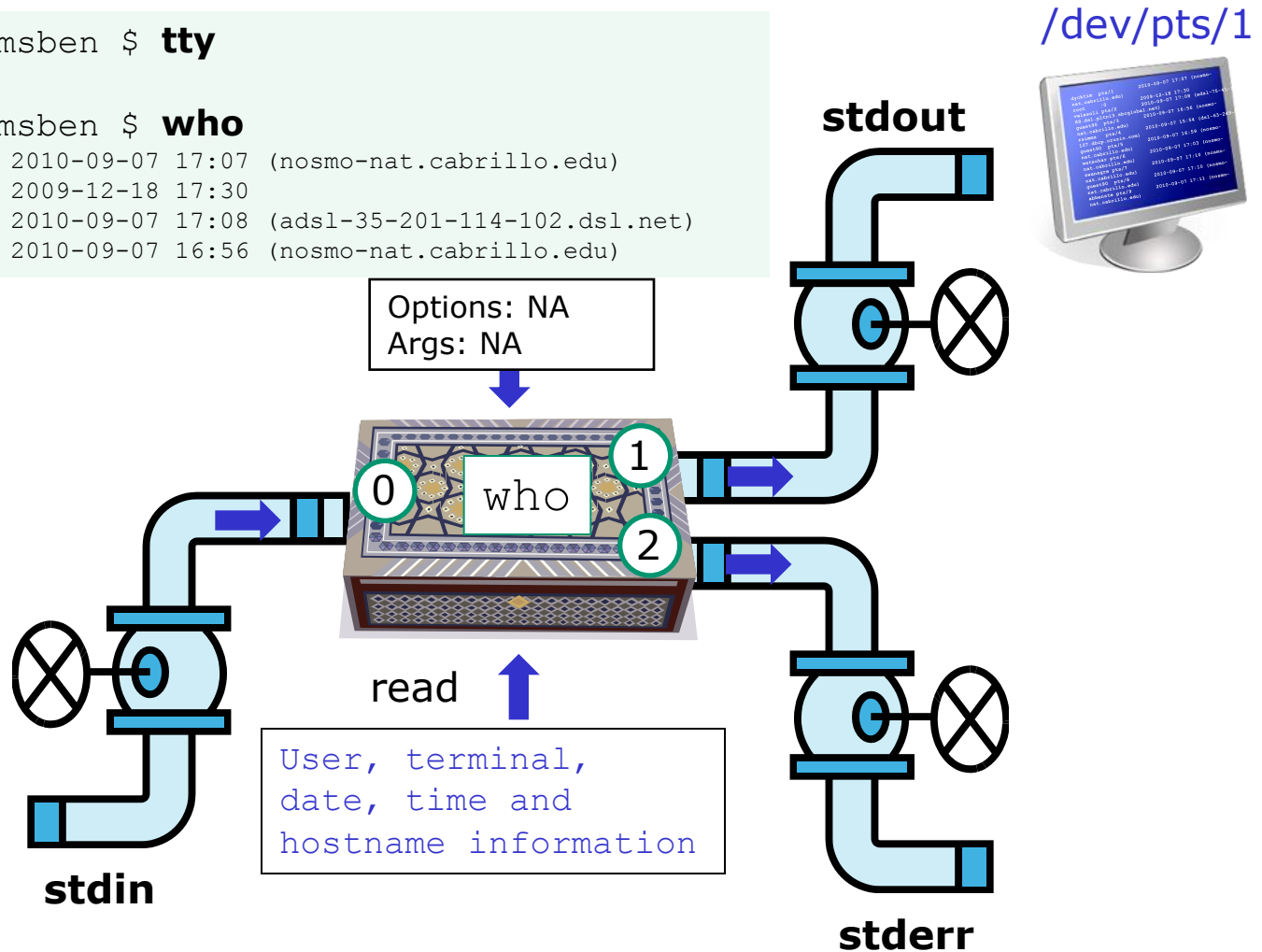
*The **bc** (binary calculator) command is an example of an interactive command that reads its input from the keyboard*

who command

```
/home/cis90/simmsben $ tty
/dev/pts/1
```

```
/home/cis90/simmsben $ who
```

```
dycktim pts/1      2010-09-07 17:07 (nosmo-nat.cabrillo.edu)
root    :0          2009-12-18 17:30
velasoli pts/2      2010-09-07 17:08 (adsl-35-201-114-102.dsl.net)
guest90 pts/3       2010-09-07 16:56 (nosmo-nat.cabrillo.edu)
```



The **who** command is an example of a command that gets its input from the Operating System

Class Exercise

Running Programs

1. Use **echo Hello World** and **banner Hello World** commands
(these commands get their input from the command line)
2. Use **bc** to add 2+2, use quit to end
(this command reads its input from the keyboard)
3. Run the **who**, **tty**, and **uname** commands
(these commands get their input from the operating system)



Command Syntax

(grammar lesson)

Command Syntax

Command**Options****Arguments****Redirection**

Command – is the name of an executable program file.

Options – a special type of argument that is used to control how the program operate operates.

Arguments – the objects the command is directed to work upon. Multiple arguments are separated by spaces.

Redirection – The default input stream (stdin) is from the console keyboard, the default output (stdout) and error (stderr) streams go to the console screen. Redirection can modify these streams to other files or devices.

Command Syntax

Command**Options****Arguments****Redirection**

Command – usually at the beginning of the line

Options – follow the command, usually starts with a dash, may be combined after a single “-” or separated by spaces (`-iad = -i -a -d`)

Arguments – follow the options. Multiple arguments must be separated by spaces.

Redirection – Will be a `<`, `>`, `>>`, `2>` or `|` followed by where the redirection is going or coming from.

Spaces are required between commands, options, arguments and any redirection

Multiple spaces are treated as a single space (unless inside quotes)

One of the things the shell does is to parse commands issued by the user

from Dictionary.com

parse [pahrs, pahrz] **verb, parsed, pars-ing.**
verb (used with object)

1. to analyze (a sentence) in terms of grammatical constituents, identifying the parts of speech, syntactic relations, etc.
2. to describe (a word in a sentence) grammatically, identifying the part of speech, inflectional form, syntactic function, etc.
3. Computers . to analyze (a string of characters) in order to associate groups of characters with the syntactic units of the underlying grammar.

Command Syntax

Command

Options

Arguments

Redirection

The command syntax is the underlying grammar used to parse the command line

```
/home/cis90/simben $ hostname  
opus.cabrillo.edu
```

```
/home/cis90/simben $ uname -o  
GNU/Linux
```

```
/home/cis90/simben $ ls -ld Poems/  
drwxr-xr-x 5 simben90 cis90 4096 Jan 18 2004 Poems/
```

```
/home/cis90/simben $ ls -li letter > /dev/null
```

More on redirection in later lessons

Command Syntax

Command	Options	Arguments	Redirection
clear			
who			
who	-Hu		
is			
id		root	
ls			
ls	-l		
ls	-l -i	Poems/	
ls	-li	letter log	
ls	-ld	Miscellaneous	> myfile
echo		red blue	
echo		"red blue"	
echo		Hello	>> myfile

More on redirection in later lessons

Parsing Practice

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ echo I love Linux  
I love Linux
```

Please parse the command line above

Command:

Options:

How many:

What are they:

Arguments:

How many:

What are they:

Redirection:

How many:

What is redirected:

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ echo I love Linux  
I love Linux
```

Please parse the command line above

Command: echo

Options:

How many: NA

What are they: NA

Arguments:

How many: 3

What are they: I, Love, Linux

Redirection:

How many: NA

What is redirected: NA

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ ls -ld /bin /usr/bin  
drwxr-xr-x 2 root root 4096 Nov 23 13:49 /bin  
drwxr-xr-x 2 root root 61440 Nov 23 13:49 /usr/bin
```

Please parse the command line above

Command:

Options:

How many:

What are they:

Arguments:

How many:

What are they:

Redirection:

How many:

What is redirected:

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ ls -ld /bin /usr/bin  
drwxr-xr-x 2 root root 4096 Nov 23 13:49 /bin  
drwxr-xr-x 2 root root 61440 Nov 23 13:49 /usr/bin
```

Please parse the command line above

Command: ls

Options:

How many: 2
What are they: l, d

Arguments:

How many: 2
What are they: /bin, /usr/bin

Redirection:

How many: NA
What is redirected: NA

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ ls-ld/bin/usr/bin  
-bash: ls-ld/bin/usr/bin: No such file or directory
```

Please parse the command line above

Command:

Options:

How many:

What are they:

Arguments:

How many:

What are they:

Redirection:

How many:

What is redirected:

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ ls-lb/bin/usr/bin  
-bash: ls-lb/bin/usr/bin: No such file or directory
```

Please parse the command line above

Command: ls-lb/bin/usr/bin

Options:

How many:	NA
What are they:	NA

Arguments:

How many:	NA
What are they:	NA

Redirection:

How many:	NA
What is redirected:	NA

*Spaces are required between
commands, options,
arguments and any
redirection*

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ file proposall timecal  
proposall: ASCII English text  
timecal:  shell archive or script for antique kernel text
```

Please parse the command line above

Command:

Options:

How many:

What are they:

Arguments:

How many:

What are they:

Redirection:

How many:

What is redirected:

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ file proposal1 timecal  
proposal1: ASCII English text  
timecal:  shell archive or script for antique kernel text
```

Please parse the command line above

Command: file

Options:

How many: NA
What are they: NA

Arguments:

How many: 2
What are they: proposal1, timecal

Redirection:

How many: NA
What is redirected: NA

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ ls -l -i -a /bin Poems/ letter small_town > /dev/null  
/home/cis90/simben $
```

Please parse the command line above

Command:

Options:

How many:

What are they:

Arguments:

How many:

What are they:

Redirection:

How many:

What is redirected:

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ ls -l -i -a /bin Poems/ letter small_town > /dev/null  
/home/cis90/simben $
```

Please parse the command line above

Command: ls

Options:

How many: 3
What are they: l, i, a

Arguments:

How many: 4
What are they: /bin, Poems/, letter, small_town

Redirection:

How many: 1
What is redirected: stdout redirected to /dev/null

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ echo "1 2 3 4 5"  
1 2 3 4 5
```

Please parse the command line above

Command:

Options:

How many:

What are they:

Arguments:

How many:

What are they:

Redirection:

How many:

What is redirected:

Command Syntax

Command

Options

Arguments

Redirection

```
/home/cis90/simben $ echo "1 2 3 4 5"  
1 2 3 4 5
```

Please parse the command line above

Command: echo

Options:

How many: NA

What are they: NA

Arguments:

How many: 1

What are they: "1 2 3 4 5"

Redirection:

How many: NA

What is redirected: NA

Variables

Variables

Just like any programming language, the shell has variables:

- A shell variable gives a name to a location in memory where data can be kept during the session.
- Shell variables are lost when a session ends.
- The shell variables used to customize the users environment are called *Environment* variables.
- To look at the value of a variable use the **echo** command and precede the variable name with a \$

echo \$PS1 *shows the current value of the PS1 variable*

- To change the value of a variable, use an = sign with no surrounding blanks and no \$

PS1="Enter next command: " *sets the PS1 prompt variable*

Variables

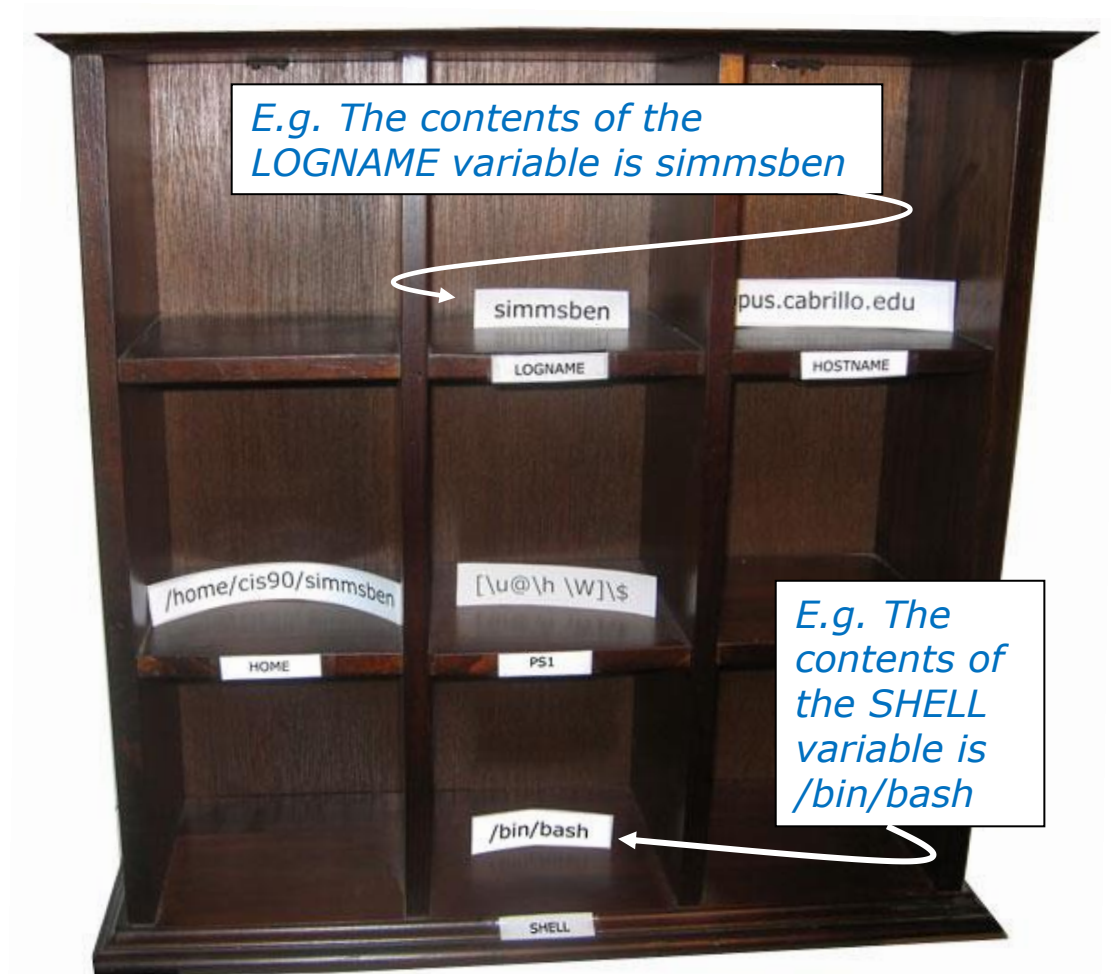
Think of variables as named boxes containing data

```
$ echo $LOGNAME  
simmsben
```

```
$ echo $HOSTNAME  
opus.cabrillo.edu
```

```
$ echo $HOME  
/home/cis90/simmsben
```

```
$ echo $SHELL  
/bin/bash
```



Showing Variable Values

To show the value of a variable use the echo command and precede the variable name with a \$

```
/home/cis90/simben $ echo $SHELL      Shows the name of your shell  
/bin/bash
```

```
/home/cis90/simben $ echo $LOGNAME    Shows your username  
simben90
```

```
/home/cis90/simben $ echo I am $LOGNAME and I use the $SHELL shell  
I am simben90 and I use the /bin/bash shell
```

If the \$ is not used, echo prints the name of the variable instead

```
/home/cis90/simben $ echo PS1  
PS1  
/home/cis90/simben $ echo LOGNAME  
LOGNAME  
/home/cis90/simben $ echo I am LOGNAME and I use the SHELL shell  
I am LOGNAME and I use the SHELL shell
```

Showing Variable Values

```
/home/cis90/simben $ echo $TERM    Shows your terminal type
xterm
```

```
/home/cis90/simben $ echo $PWD    Shows your current working directory
/home/cis90/simben
```

```
/home/cis90/simben $ echo $PS1    Shows your level 1 prompt string
$PWD $
```

```
/home/cis90/simben $ echo $HOME    Shows your home directory
/home/cis90/simben
```

```
/home/cis90/simben $ echo $PATH    Shows the directories making up your path
/usr/lib/qt-
3.3/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/s
bin:/home/cis90/simben/../../bin:/home/cis90/simben/bin:.
```

Shell (Environment) Variables

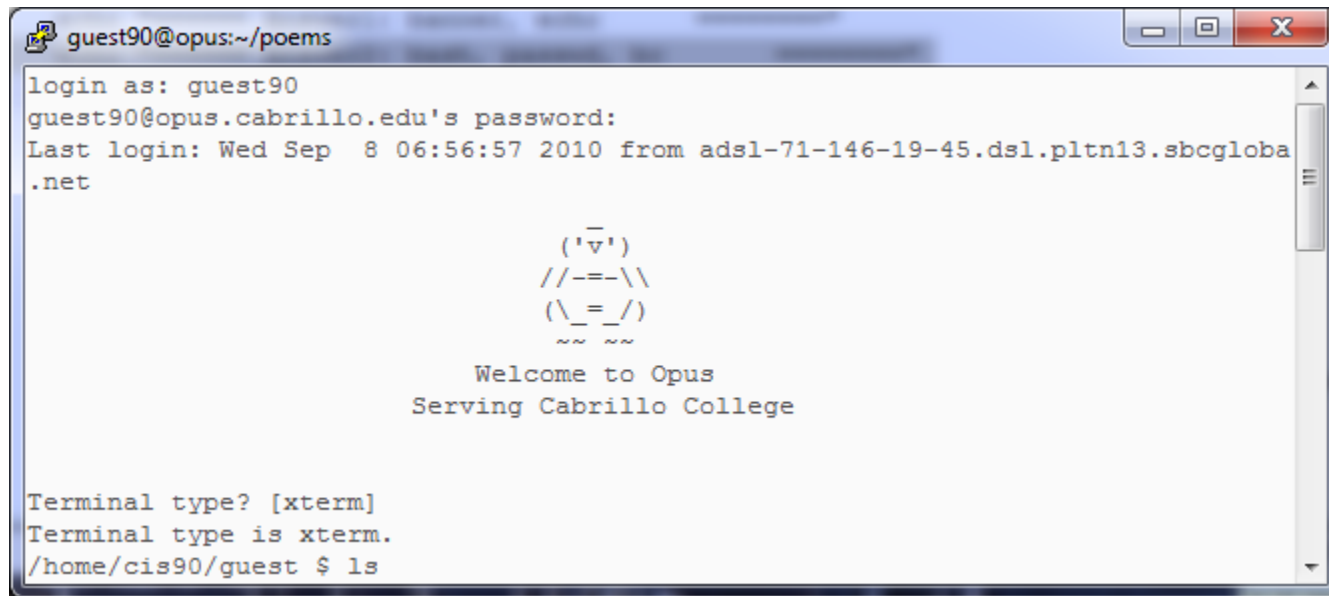
common environment variables

Shell Variable	Description
HOME	Users home directory (starts here after logging in and returns with a <code>cd</code> command (with no arguments)
LOGNAME	User's username for logging in with.
PATH	List of directories, separated by <code>:</code> 's, for the Shell to search for commands (which are program files) .
PS1	The prompt string.
PWD	Current working directory
SHELL	Name of the Shell program being used.
TERM	Type of terminal device , e.g. dumb, vt100, xterm, ansi, linux, etc.

Shell (Environment) Variables

common environment variables

Shell Variable	Description
TERM	Type of terminal, e.g. dumb, vt100, xterm, ansi, linux, color, etc.



```

guest90@opus:~/poems
login as: guest90
guest90@opus.cabrillo.edu's password:
Last login: Wed Sep  8 06:56:57 2010 from adsl-71-146-19-45.dsl.pltn13.sbcglobe.net

      _
    ('v')
  //--\\
  (\_=/)
   ~~~~

  Welcome to Opus
  Serving Cabrillo College

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

Note the TERM variable gets set every time we log into Opus

Setting Variable Values

To change the value of a variable, use an = sign with no surrounding blanks and no \$

```
/home/cis90/simben $ echo $TERM  
xterm
```

Show the current terminal type

```
/home/cis90/simben $ TERM=dumb  
/home/cis90/simben $ echo $TERM  
dumb
```

Change the terminal type and display the new value

```
/home/cis90/simben $ TERM=xterm  
/home/cis90/simben $ echo $TERM  
xterm
```

Change the terminal type back to the original value

In Lab 2 you will see what happens when the terminal type is changed



Changing the prompt (PS1 variable)

Changing the prompt

```
/home/cis90/simben $ echo $PS1
$PWD $
/home/cis90/simben $ cd Poems/
/home/cis90/simben/Poems $ cd /bin
/bin $ cd
/home/cis90/simben $
```

View the current prompt variable which contains another variable \$PWD followed by a \$.

The PWD variable always contains the name of the current directory. Notice how the prompt changes when you change directories.

```
/home/cis90/simben $ PS1="By your command > "
By your command > date
Mon Sep  3 17:25:32 PDT 2012
By your command >
```

Set the prompt to a new value

```
By your command > PS1='What can I do for you $LOGNAME? '
What can I do for you simben90? date
Mon Sep  3 17:26:10 PDT 2012
What can I do for you simben90?
```

Set the prompt to a new value

```
What can I do for you simben90? PS1='$PWD $ '
/home/cis90/simben $
/home/cis90/simben $
```

Restore the original CIS 90 prompt. This prompt is automatically set every time you login

Changing the prompt

Special Codes	Meaning
\!	history command number
\#	session command number
\d	date
\h	hostname
\n	new line
\s	shell name
\t	time
\u	user name
\w	entire path of working directory
\W	only working directory
\\$	\$ or # (for root user)

The PS1 variable (defines the prompt) can be set to any combination of text, variables and these special codes.

Changing the prompt

There are some special \codes you can use when setting the prompt

\h gets replaced by the hostname

\W gets replaced by the base working directory

\u gets replaced by the username

```
/home/cis90/simben $ PS1="[\u@\h \W]\$ "
```

```
[simben90@oslab ~]$ date
```

```
Mon Sep 3 17:38:54 PDT 2012
```

```
[simben90@oslab ~]$
```

\\$ gets replaced by a \$ for regular users or # if the root user

user name

hostname

indicates regular user

*working directory
(~ is shorthand for the home directory)*

Environment variables

Changing the shell prompt

Prompt string	Result
PS1='\$PWD \$ '	/home/cis90/simmsben/Poems \$
PS1="\w \$ "	~/Poems \$
PS1="\W \$ "	Poems \$
PS1="\u@\h \$ "	simmsben@opus \$
PS1='\u@\h \$PWD \$ '	simmsben@opus /home/cis90/simmsben/Poems \$
PS1='\u@\\$HOSTNAME \$PWD \$ '	simmsben@opus.cabrillo.edu /home/cis90/simmsben/Poems \$
PS1='\u \! \$PWD \$ '	simmsben 825 /home/cis90/simmsben/Poems \$
PS1="[\u@\h \W] \$ "	[simmsben@opus Poems] \$

Important: Use single quotes around variables that change. For example if you use \$PWD with double quotes, the prompt will not changes as you change directories! More on this later ...



*Need a fresh start -- just log out
and back in again and your prompt
will be back to normal!*

Listing all the variables

Shell Variables set command

```
/home/cis90/simben $ set
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:expand_aliases:extquote:force_ignores
complete:interactive_comments:login_shell:progcomp:promptvars:sourcepath
BASH_ALIASES=()
BASH_ARGC=()
BASH_ARGV=()
BASH_CMDS=()
BASH_ENV=/home/cis90/simben/.bashrc
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="4" [1]="1" [2]="2" [3]="1" [4]="release" [5]="i386-
redhat-linux-gnu")
BASH_VERSION="4.1.2(1)-release"
COLORS=/etc/DIR_COLORS
COLUMNS=123
CVS_RSH=ssh
DIRSTACK=()
EUID=1001
GROUPS=()
G_BROKEN_FILENAMES=1
HISTCONTROL=ignoredups
HISTFILE=/home/cis90/simben/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/home/cis90/simben
HOSTNAME=oslab.cabrillo.edu
HOSTTYPE=i386
ID=1001
IFS=$' \t\n'
IGNOREEOF=10
LANG=en_US.UTF-8
LESSOPEN='|/usr/bin/lesspipe.sh %s'
LINES=38
LOGNAME=simben90
```

*The **set** command shows all shell variables including the special environment variables.*

```
LS_COLORS='rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;3
3;01:cd=40;33;01:or=40;31;01:mi=01;05;37;41:su=37;41:sg=30;43:ca=30;41:tw=
30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz
=01;31:*.lzh=01;31:*.lзма=01;31:*.tlz=01;31:*.txz=01;31:*.zip=01;31:*.z=01
;31:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lz=01;31:*.xz=01;31:*.bz2=01;31:*.tb
z=01;31:*.tbz2=01;31:*.bz=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=0
1;31:*.rar=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;3
1:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35
:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:
*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.
mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.ogm=01;35:*.mp4=01;35:*.
m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.as
f=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=
01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;3
5:*.emf=01;35:*.axv=01;35:*.anx=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=01;36:
*.au=01;36:*.flac=01;36:*.mid=01;36:*.midi=01;36:*.mka=01;36:*.mp3=01;36:*.
mpc=01;36:*.ogg=01;36:*.ra=01;36:*.wav=01;36:*.axa=01;36:*.oga=01;36:*.sp
x=01;36:*.xspf=01;36:'
MACHTYPE=i386-redhat-linux-gnu
MAIL=/var/spool/mail/simben90
MAILCHECK=60
OLDPWD=/bin
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/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:.
PIPESTATUS=([0]="127")
PPID=17309
PROMPT_COMMAND='printf "\033]0;%s@%s:%s\007" "${USER}" "${HOSTNAME%%.*}"
"${PWD}/${HOME}/~"'
PS1='$PWD $ '
PS2='> '
PS4='+ '
PWD=/home/cis90/simben
QTDIR=/usr/lib/qt-3.3
QTINC=/usr/lib/qt-3.3/include
QTLIB=/usr/lib/qt-3.3/lib
SELINUX_LEVEL_REQUESTED=
SELINUX_ROLE_REQUESTED=
SELINUX_USE_CURRENT_RANGE=
SHELL=/bin/bash
SHELLOPTS=braceexpand:emacs:hashall:histexpand:history:ignoreeof:interacti
ve-comments:monitor
SHLVL=1
SSH_CLIENT='50.0.68.235 51849 2220'
SSH_CONNECTION='50.0.68.235 51849 172.30.5.20 2220'
SSH_TTY=/dev/pts/2
TERM=xterm
UID=1001
USER=simben90
USERNAME=
_=ser
colors=/etc/DIR_COLORS
/home/cis90/simben $
```

Shell (Environment) Variables

env command

```
/home/cis90/simben $ env
```

```
HOSTNAME=oslabb.cabrillo.edu
```

```
SELINUX_ROLE_REQUESTED=
```

```
TERM=xterm
```

```
SHELL=/bin/bash
```

```
HISTSIZE=1000
```

```
SSH_CLIENT=50.0.68.235 51849 2220
```

```
SELINUX_USE_CURRENT_RANGE=
```

```
QTDIR=/usr/lib/qt-3.3
```

```
QTINC=/usr/lib/qt-3.3/include
```

```
SSH_TTY=/dev/pts/2
```

```
USER=simben90
```

```
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=01;05;37;41:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz=01;31:*.lzh=01;31:*.lzm=01;31:*.tlz=01;31:*.txz=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lz=01;31:*.xz=01;31:*.bz2=01;31:*.tbz=01;31:*.tbz2=01;31:*.bz=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.rar=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.axv=01;35:*.anx=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=01;36:*.au=01;36:*.flac=01;36:*.mid=01;36:*.midi=01;36:*.mka=01;36:*.mp3=01;36:*.mpc=01;36:*.ogg=01;36:*.ra=01;36:*.wav=01;36:*.axa=01;36:*.oga=01;36:*.spx=01;36:*.xspf=01;36:
```

```
USERNAME=
```

```
MAIL=/var/spool/mail/simben90
```

```
PATH=/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:.
```

```
PWD=/home/cis90/simben
```

```
LANG=en_US.UTF-8
```

```
SELINUX_LEVEL_REQUESTED=
```

```
HISTCONTROL=ignoredups
```

```
SHLVL=1
```

```
HOME=/home/cis90/simben
```

```
BASH_ENV=/home/cis90/simben/.bashrc
```

```
LOGNAME=simben90
```

```
QTLIB=/usr/lib/qt-3.3/lib
```

```
CVS_RSH=ssh
```

```
SSH_CONNECTION=50.0.68.235 51849 172.30.5.20 2220
```

```
LESSOPEN=|/usr/bin/lesspipe.sh %s
```

```
G_BROKEN_FILENAMES=1
```

```
_=/bin/env
```

```
OLDPWD=/bin
```

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

*The **env** command shows just the environment variables*

Class Exercise

Environment Variables

1. Change your prompt to "What is your command master? "
2. Use **echo** to show your logname (\$LOGNAME)

Meta- characters

Metacharacters

The shell gives special meaning to metacharacters

" - use double quotes to preserve blanks and allow variable expansion

' - use single quotes to preserve blanks and block variable expansion

\$ - use to show the value rather than the name of a variable

;- allows multiple commands on one line

<enter key> - The invisible newline control character marking the end of a command

= - use to set variables to new values

\ - removes (escapes) the special powers of a metacharacter

Other metacharacters we will learn about later include:

*?, *, <, >, >>, !, |, [], {}, &, && and ||*

Metacharacters - quotes

" - use double quotes preserve blanks and allows variable expansion

' - use single quotes preserve blanks and block variable expansion

```
/home/cis90/simben $ echo I am $LOGNAME (3 arguments)
I am simben90 Extra blanks ignored, variable expanded
```

```
/home/cis90/simben $ echo "I am $LOGNAME" (1 argument)
I am simben90 Extra blanks preserved, variable expanded to show value
```

```
/home/cis90/simben $ echo 'I am $LOGNAME' (1 argument)
I am $LOGNAME Extra blanks preserved, variable expansion blocked
```

Sometimes you will hear single quotes called strong quotes as they block variable expansion. Likewise you may hear double quotes called weak quotes because they allow variable expansion.

Metacharacters - quotes

- " - use double quotes preserve blanks and allows variable expansion
- ' - use single quotes preserve blanks and block variable expansion

```
/home/cis90/simben $ echo '"double quotes"'  
"double quotes"
```

```
/home/cis90/simben $ echo "'single quotes'"  
'single quotes'
```

Tip: single quotes can be used to output double quotes and vice-versa

Metacharacters

<enter key> newline control character

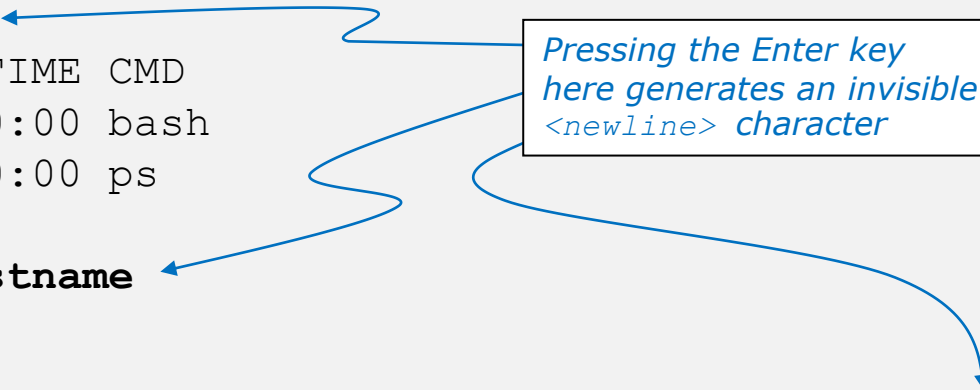
<enter key> - The invisible *newline* control character marking the end of a command

```
[rsimms@opus ~]$ ps
  PID TTY          TIME CMD
 19015 pts/0    00:00:00 bash
 19378 pts/0    00:00:00 ps

[rsimms@opus ~]$ hostname
opus.cabrillo.edu

[rsimms@opus ~]$ echo "Use <enter key> to end the command"
Use <enter key> to end the command
```

Pressing the Enter key here generates an invisible <newline> character



Metacharacters - \ (backslash)

\ - removes (escapes) the special powers of a metacharacter

```
[rsimms@oslab ~]$ echo a b c d e f
a b c d e f
```

```
[rsimms@opus ~]$ echo a b c \
> d e f
a b c d e f
```

*Escape the invisible newline <enter key>
which marks the end of a command*

```
[rsimms@opus ~]$ echo $PS1
[\u@\h \W]\$
```

```
[rsimms@opus ~]$ echo \$PS1
$PS1
```

*Escape the \$ (which shows
the value of the variable)*

```
[rsimms@opus ~]$ echo "Hello World"
Hello World
```

```
[rsimms@opus ~]$ echo \"Hello World\"
\"Hello World\"
```

*Escape the double quote
marks*

Metacharacters - ; (command separator)

; - allows multiple commands on one line

```
[simmsben@opus Poems]$ hostname; uname; echo $LOGNAME; ls  
opus.cabrillo.edu  
Linux  
simmsben  
ant  Blake  nursery  Shakespeare  twister  Yeats
```

*Four commands on
one line*



Shortcuts

More on the Command Line

Handy Shortcuts

- Use up and down arrows to “retype” previous commands
- Left and right arrow for editing current command
- Use <tab> to complete filenames automatically

```
[simmsben@opus Poems]$ hostname; name; echo $LOGNAME; ls Blake/
opus.cabrillo.edu
bash: name: command not found
simmsben
jerusalem tiger
```

Press <tab> after the B and the shell fills in the remaining "lake/"

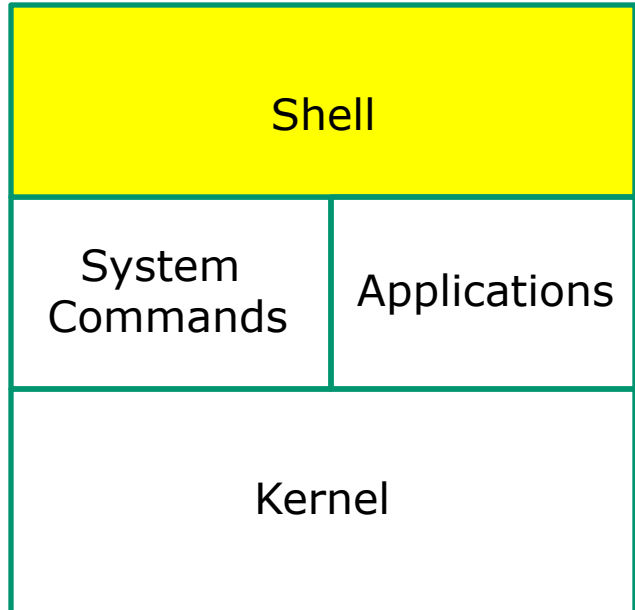
```
[simmsben@opus Poems]$ hostname; uname; echo $LOGNAME; ls Blake/
opus.cabrillo.edu
Linux
simmsben
jerusalem tiger
```

Press up arrow and the shell retypes the previous command

Use the left arrow to backup and fix the typo (uname instead of name)

Shell

The Shell

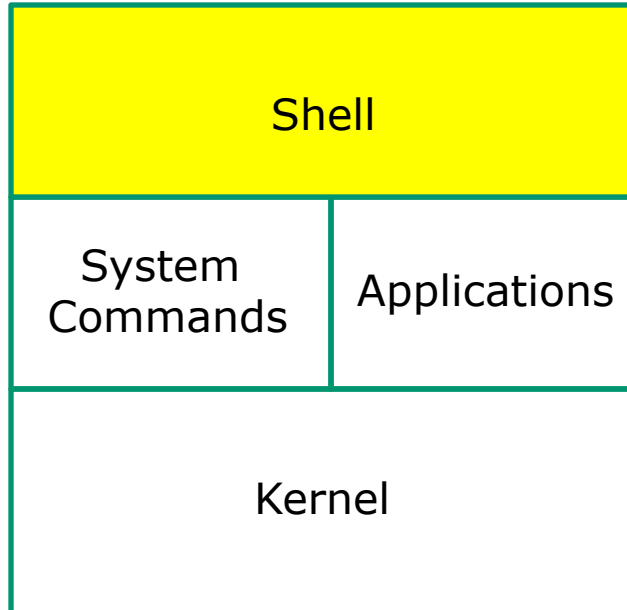


- Allows users to interact with the computer via a “**command line**”.
- **Prompts** for a command, parses the command, finds the right program and gets that program executed.
- Is called a “**shell**” because it hides the underlying operating system.
- Multiple shell programs are available: **sh** (Bourne shell), **bash** (born again shell), **csh** (C shell), **ksh** (Korn shell).
- The shell is a **user interface** and a **programming language** (scripts).
- GNOME and KDE desktops could be called **graphical shells**





Life of the Shell



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





Life of the Shell

Example:

```
/home/cis90/simben $ ls -lt proposal1 proposal2
-rw-r--r--. 1 simben90 cis90 1074 Aug 26 2003 proposal1
-rw-r--r--. 1 simben90 cis90 2175 Jul 20 2001 proposal2
/home/cis90/simben $
```

Shell Steps

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

Lets take a deep dive into how a command gets executed.

Note it is always a team effort by both the shell and the command.



Life of the Shell

Shell Steps

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

1) Prompt user for a command

Example:

*The shell begins by outputting the prompt
(which is based on the PS1 variable)*

```
/home/cis90/simben $ ls -lt proposal1 proposal2
```

Then you type the command

FYI, you can mimic outputting the prompt yourself with these commands:

```
/home/cis90/simben $ echo $PS1 to show value of PS1 variable
```

```
$PWD $
```

```
/home/cis90/simben $ echo $PWD $ echo the output of the  
previous command
```

```
/home/cis90/simben $ was output by the echo command above
```

```
/home/cis90/simben $ was output by the shell (the same output)
```



Life of the Shell

2) Parse command user typed

Shell Steps

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

Example:

```
ls -lt proposal1 proposal2
```

During the parse step the shell identifies all options & arguments, handles any metacharacters and redirection

- Command = ls
- 2 Options = l, t
- 2 Arguments = proposal1, proposal2
- 1 Redirection = NA



Life of the Shell

3) Search path for the program to run

Shell Steps

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

ls -lt proposal1 proposal2

Use this command to see the path directories (separated by ':'s) on your path

```
/home/cis90/simben $ echo $PATH
/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:.
```

*The shell will search each directory in order for an **ls** command*

```
/usr/lib/qt-3.3/bin no
/usr/local/bin no
/bin YES! – it was found in the /bin directory
/usr/bin
/usr/local/sbin
/usr/sbin
/sbin
/home/cis90/simben/../../bin
/home/cis90/simben/bin
.
```

Try mimicking what the shell does to search for ls:

```
/home/cis90/simben $ ls /usr/lib/qt-3.3/bin/ls
ls: cannot access /usr/lib/qt-3.3/bin/ls: No
such file or directory
```

```
/home/cis90/simben $ ls /usr/local/bin/ls
ls: cannot access /usr/local/bin/ls: No such
file or directory
```

```
/home/cis90/simben $ ls /bin/ls
/bin/ls
```



Life of the Shell

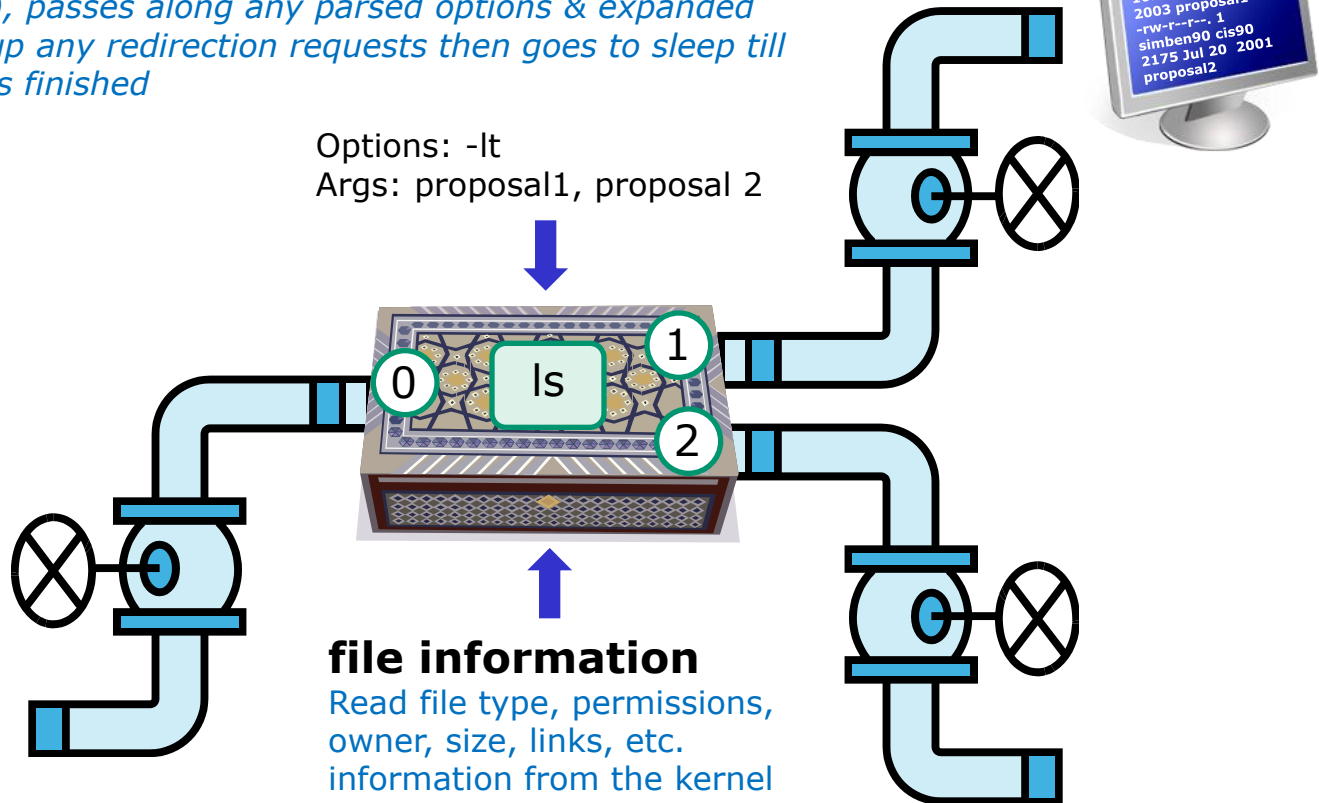
Shell Steps

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

4) Execute the command

```
ls -lt proposal1 proposal2
```

Invokes the kernel to load the program into memory (which becomes a process), passes along any parsed options & expanded arguments, hooks up any redirection requests then goes to sleep till the new process has finished





Life of the Shell

5) Nap while the command (process) runs to completion

(The shell, itself a loaded process, goes into the sleep state and waits till the command process is finished)

Shell Steps

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

```
/home/cis90/simben $ ls -lt proposal1 proposal2
-rw-r--r--. 1 simben90 cis90 1074 Aug 26 2003 proposal1
-rw-r--r--. 1 simben90 cis90 2175 Jul 20 2001 proposal2
```



Life of the Shell

6) And do it all over again
... go to step 1

Shell Steps

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



Life of the Shell

A /home/cis90/simben \$ **Ls -lt proposal1 proposal2**

-bash: Ls: command not found

What's wrong?

Who output the error?

B /home/cis90/simben \$ **ls -lt proposal1 proposal5**

ls: cannot access proposal5: No such file or directory

-rw-r--r--. 1 simben90 cis90 1074 Aug 26 2003 proposal1

What's wrong?

Who output the error?

C /home/cis90/simben \$ **ls -lw proposal1 proposal2**

ls: invalid line width: proposal1

What's wrong?

Who output the error?

D /home/cis90/simben \$ **ls -lt proposal1proposal2**

ls: cannot access proposal1proposal2: No such file or directory

What's wrong?

Who output the error?

E /home/cis90/simben \$ **ls-lt proposal1 proposal2**

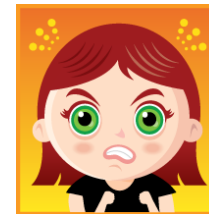
-bash: ls-lt: command not found

What's wrong?

Who output the error?

Life without a path

-bash: xxxx: command not found



Don't get mad, just fix your path!

The Path

The shell uses your path to locate commands to execute

- A path is a ordered set of directories along which the shell will search to locate commands to execute
- The path is defined by the PATH variable
- Show your path with **echo \$PATH**
- If you specify a command xxxx that the shell cannot find on the path it will print the following error message:

-bash: xxxx: command not found
- To run a command that is not on your path the complete absolute pathname must be specified. e.g. /usr/bin/uname

The Path

Use this command to see the directories (separated by :’s) on your path

```
/home/cis90/simben $ echo $PATH
```

```
/usr/lib/qt-
```

```
3.3/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/home/c  
is90/simben/../../bin:/home/cis90/simben/bin:.
```

The shell will search for the ls command along the path in this order:

```
/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
```

```
•
```



*yes, . is a directory too and it is whatever
directory you have currently changed into*

Experiment – Breaking the Path

The **echo**
command is
built into bash

```
/home/cis90/simben $ type echo ps tty
echo is a shell builtin
ps is /bin/ps
tty is /usr/bin/tty
```

the **ps**
command is in
the **/bin**
directory

The **tty** command
is in the **/usr/bin**
directory



/usr/bin



/bin

Experiment – Breaking the Path

*Default
path*

```
/home/cis90/simben $ echo I love Linux
I love Linux
/home/cis90/simben $ date
Mon Sep  3 15:17:52 PDT 2012
/home/cis90/simben $ tty
/dev/pts/2
/home/cis90/simben $
```

TROUBLE!

```
/home/cis90/simben $ PATH=""
/home/cis90/simben $ echo $PATH

/home/cis90/simben $
```

*Break the path by
setting it to null*

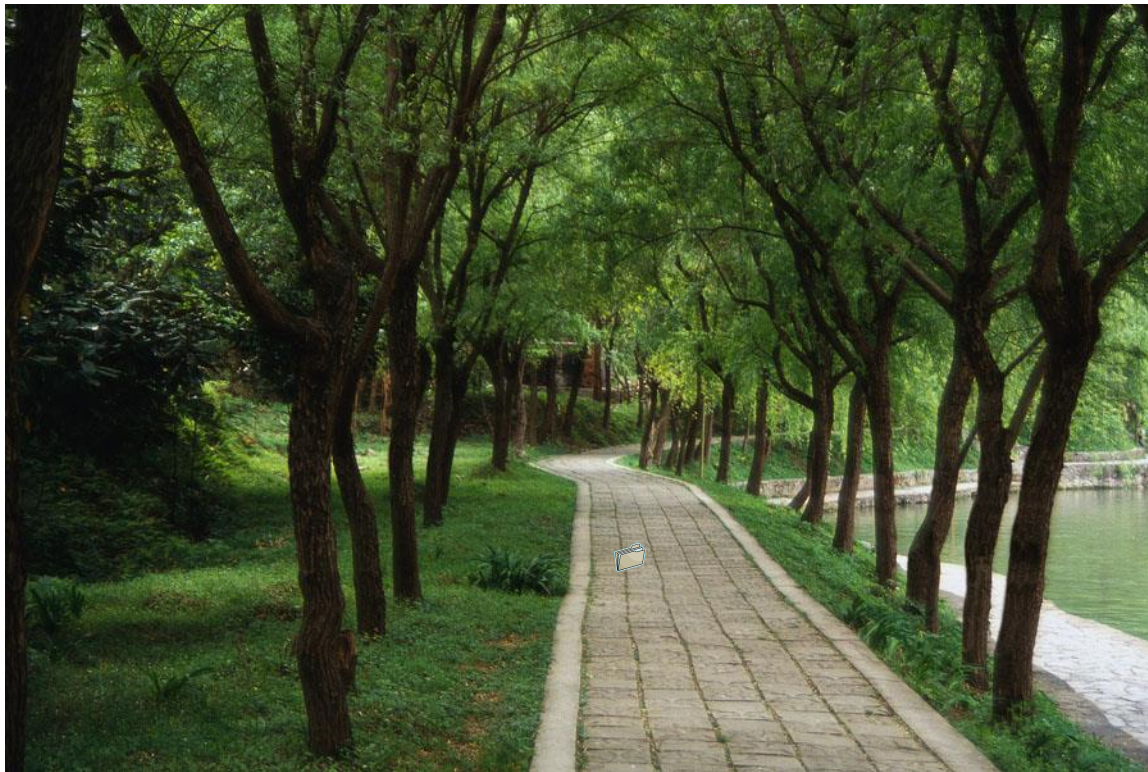
No path

```
/home/cis90/simben $ echo I love Linux
I love Linux
/home/cis90/simben $ date
-bash: date: No such file or directory
/home/cis90/simben $ tty
-bash: tty: No such file or directory
```

*Only **echo**
works because
it is built into
the shell!*

```
/home/cis90/simben $ echo $PATH
```

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



There is nothing on the path!

Experiment – Restoring the Path

```
/home/cis90/simben $ PATH=/bin
/home/cis90/simben $ echo $PATH
/bin
/home/cis90/simben $
```

*Add the /bin
directory to the path*

*date works
because it
resides in the
/bin directory
which is now
on the path*

```
/home/cis90/simben $ echo I love Linux
I love Linux
/home/cis90/simben $ date
Mon Sep  3 15:24:19 PDT 2012
/home/cis90/simben $ tty
-bash: tty: No such file or directory
```

*echo works
because it is built
into the shell*

*tty does not work because it is
in the /usr/bin directory which is
not on the path*

```
/home/cis90/simben $ echo $PATH  
/bin  
/home/cis90/simben $
```



Experiment – Restoring the Path

```
/home/cis90/simben $ PATH=$PATH:/usr/bin  
/home/cis90/simben $ echo $PATH  
/bin:/usr/bin  
/home/cis90/simben $
```

*Append the
/usr/bin directory
to the path*

```
/home/cis90/simben $ echo I love Linux  
I love Linux  
/home/cis90/simben $ date  
Mon Sep 3 15:24:19 PDT 2012  
/home/cis90/simben $ tty  
/dev/pts/2
```

All three commands work because /bin and /usr/bin are on the path.

The shell will only run commands found in the directories that make up the path

```
/home/cis90/simben $ echo $PATH  
/bin:/usr/bin  
/home/cis90/simben $
```





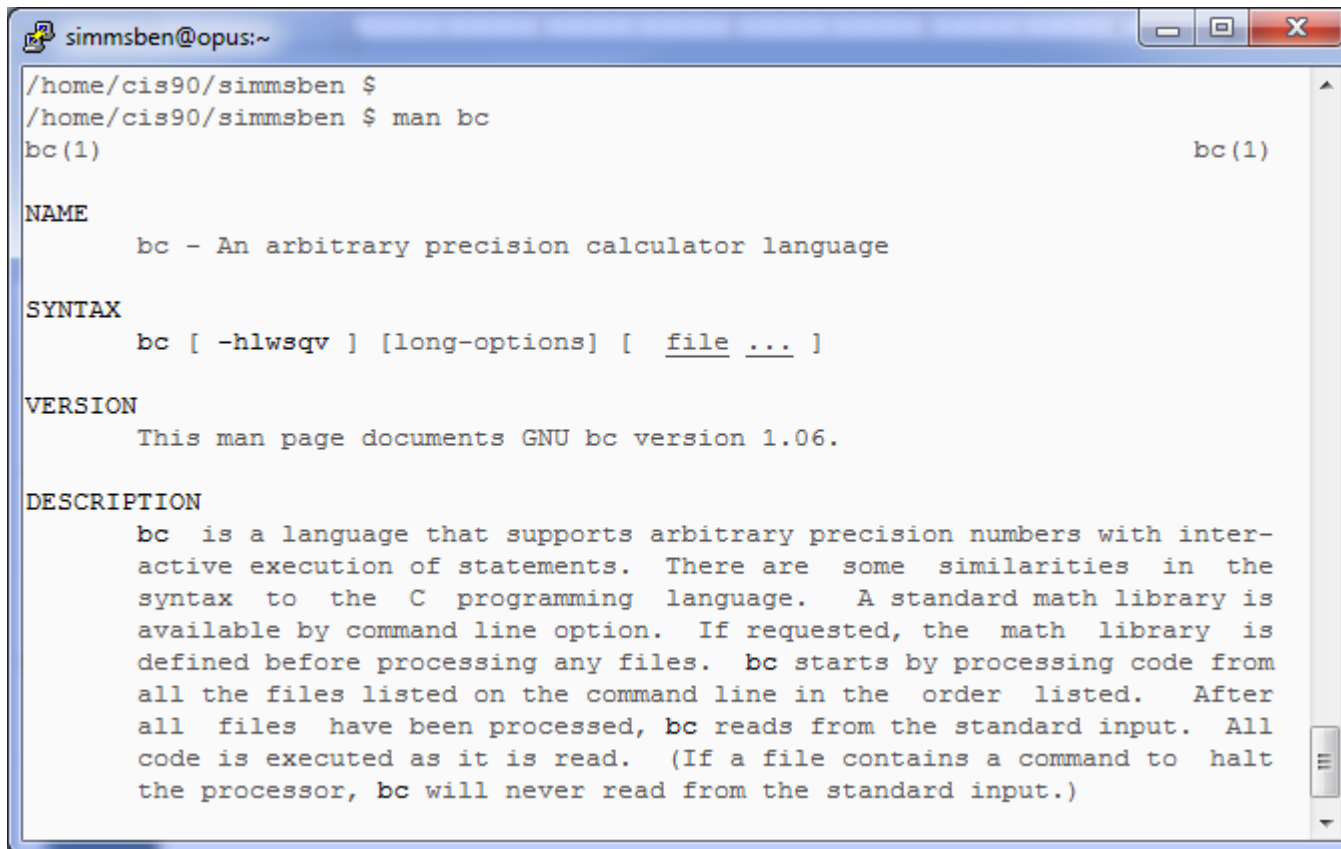
*Need a fresh start -- just log out
and back in again and your path
will be back to normal!*

Docs

Using man (manual) pages

Type the **man** command followed by the name of the command you want documentation on.

Example: **man bc**



```

simmsben@opus:~
/home/cis90/simmsben $
/home/cis90/simmsben $ man bc
bc(1)                                     bc(1)

NAME
    bc - An arbitrary precision calculator language

SYNTAX
    bc [ -hlwsqv ] [long-options] [ file ... ]

VERSION
    This man page documents GNU bc version 1.06.

DESCRIPTION
    bc is a language that supports arbitrary precision numbers with inter-
    active execution of statements. There are some similarities in the
    syntax to the C programming language. A standard math library is
    available by command line option. If requested, the math library is
    defined before processing any files. bc starts by processing code from
    all the files listed on the command line in the order listed. After
    all files have been processed, bc reads from the standard input. All
    code is executed as it is read. (If a file contains a command to halt
    the processor, bc will never read from the standard input.)
  
```



Use these
keys to scroll



Use q key to
quit

Using Google

Do a Google search on "linux xxx command" where xxx is the command you want documentation for.

Example: google linux bc command

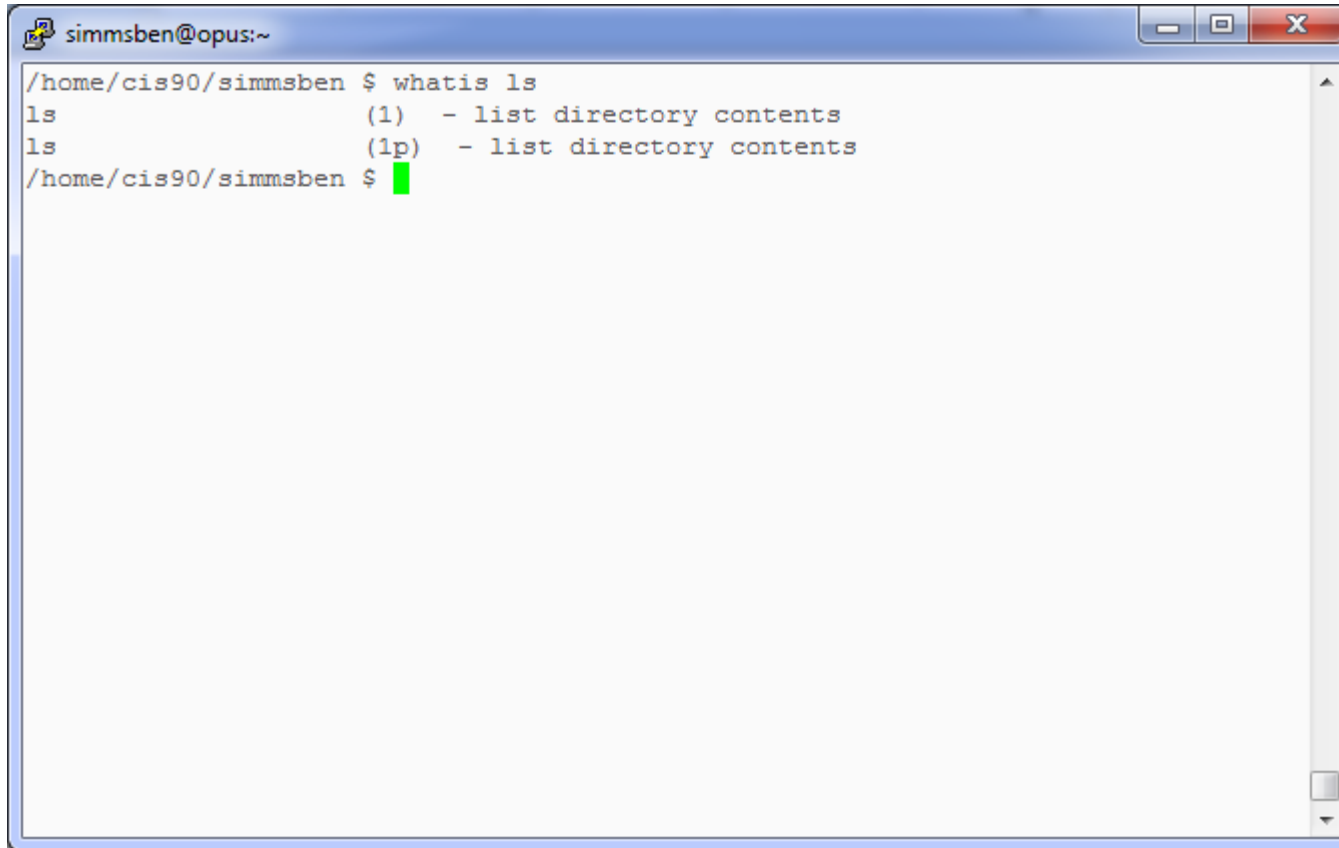
The image shows two overlapping browser windows. The left window displays a Google search for "linux bc command", showing approximately 1,180,000 results. The right window shows the "About.com: Linux" page for the "bc" command. The page includes a "DID YOU KNOW..." section, a "SYNTAX" section, and a "DESCRIPTION" section. The "bc" command is described as an arbitrary precision calculator language. The "SYNTAX" section shows the command format: `bc [ -hlwsvq ] [long-options] [ file ... ]`. The "DESCRIPTION" section explains that "bc" is a language that supports arbitrary precision numbers with interactive execution of statements, and it has some similarities to the C programming language. A standard math library is available by command line option. If requested, the math library is defined before processing any files. `bc` starts by processing code from all the files listed.

Other Documentation

- **whatis** *command* *same as the **man -f** command*
- **apropos** *command* *same as the **man -k** command*
- **info** *command*

Documentation examples

Example: **whatis ls**

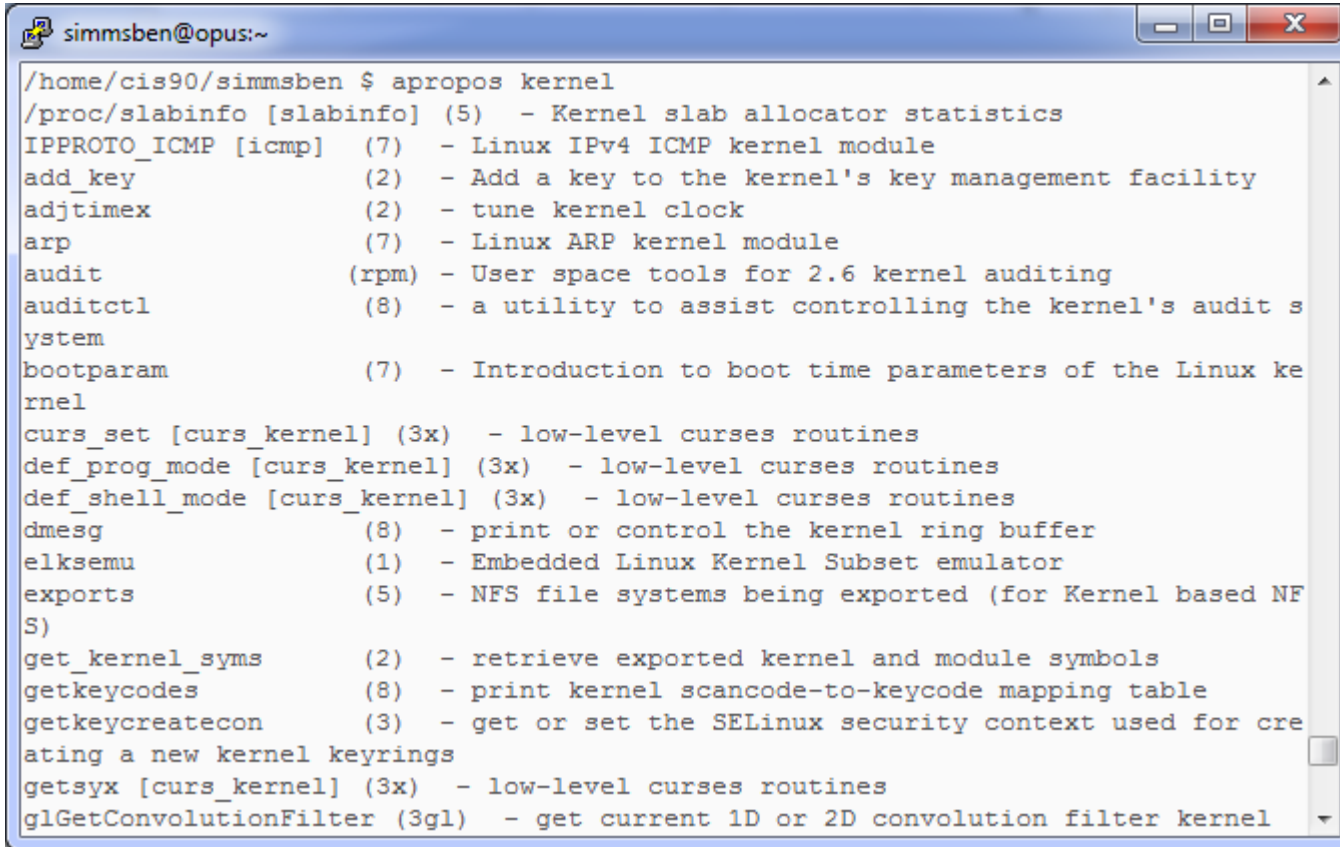


```
simmsben@opus:~  
/home/cis90/simmsben $ whatis ls  
ls                (1)  - list directory contents  
ls                (lp) - list directory contents  
/home/cis90/simmsben $
```

whatis searches the *whatis* database for a complete word. Same as the **man -f** command .

Documentation examples

Example: **apropos kernel**



```
simmsben@opus:~$ /home/cis90/simmsben $ apropos kernel
/proc/slabinfo [slabinfo] (5) - Kernel slab allocator statistics
IPPROTO_ICMP [icmp] (7) - Linux IPv4 ICMP kernel module
add_key (2) - Add a key to the kernel's key management facility
adjtimex (2) - tune kernel clock
arp (7) - Linux ARP kernel module
audit (rpm) - User space tools for 2.6 kernel auditing
auditctl (8) - a utility to assist controlling the kernel's audit system
bootparam (7) - Introduction to boot time parameters of the Linux kernel
curs_set [curs_kernel] (3x) - low-level curses routines
def_prog_mode [curs_kernel] (3x) - low-level curses routines
def_shell_mode [curs_kernel] (3x) - low-level curses routines
dmesg (8) - print or control the kernel ring buffer
elksemu (1) - Embedded Linux Kernel Subset emulator
exports (5) - NFS file systems being exported (for Kernel based NFS)
get_kernel_syms (2) - retrieve exported kernel and module symbols
getkeycodes (8) - print kernel scancode-to-keycode mapping table
getkeycreatecon (3) - get or set the SELinux security context used for creating a new kernel keyrings
getsyx [curs_kernel] (3x) - low-level curses routines
glGetConvolutionFilter (3gl) - get current 1D or 2D convolution filter kernel
```

***apropos** searches the **whatis** database for a string of text. Same as the **man -k** command .*

Documentation examples

Example: **info ls**

```
simmsben@opus:~
file: coreutils.info, Node: ls invocation, Next: dir invocation, Up: Directory listing

10.1 `ls': List directory contents
=====

The `ls' program lists information about files (of any type, including
directories). Options and file arguments can be intermixed
arbitrarily, as usual.

For non-option command-line arguments that are directories, the
default `ls' lists the contents of directories, not recursively,
omitting files with names beginning with `.'. For other file
arguments, by default `ls' lists just the file name. If a file
argument is specified, `ls' operates on the current directory
as if it had been invoked with a single argument of `.'

By default, the output is sorted alphabetically, according to the
locale settings in effect. (1) If standard output is a terminal,
the output is in columns (sorted vertically) and control characters
are output as question marks; otherwise, the output is list-oriented
and control characters are output as-is.

--zz-Info: (coreutils.info.gz)ls invocation, 54 lines --Top-----
Welcome to Info version 4.8. Type ? for help, m for menu item.
```

Navigating info pages:

- Enter to follow links (*'s)
- n or <space> for next page
- p for previous page
- u for up tree
- l for last page
- q to quit

Documentation

Two of my favorite documentation links

Rich's Cabrillo College CIS Classes Resources

Home **Resources** Forums CIS Lab CTC

Login
Flashcards
Admin

CIS 90
Previous Classes

103 days till term ends!

Cabrillo College
Web Advisor
CCC Confer
Static IPs
Quick Ref
VM Repairs
GAH!

Links

Instructors

- Linux Master Jim
- Programming Master Ed
- Network Master Gerlinde
- Network Master Rick
- Web Master John
- Windows Master Gary

Clubs

- GNU Linux Users Group

Departments

- CNSA
- CIS
- CS

Crib Sheets

- Ollie Wright (CIS 90)

Documentation

- TLDP
- LINFO

Animations

- Linux network technologies

Getting Linux

- Linux ISOs
- Kernels
- RPMs (rpmfind)
- RPMs (pbone)

Tools and Software

- Apache
- Bastille
- cygwin
- DOS boot disks
- Dyn
- John
- MS
- All
- Qu
- su
- Tri
- Vir
- VM
- Wi

Howtos

- HowtoForge
- email
- DNS
- Ethernet (NIC drivers)
- NFS
- NIS
- PPP
- Putty SSH Keys
- sed

Google | 0 un... | Yahoo... | Rich's... | The Li... | Cabril... | linux I... |

http://tldp.org/

The Linux Documentation Project

2010-09-06

LDP Worldwide

- Mirrors
- Non-English info
- Translation effort
- Translated Guides
- Translated HOWTOs
- Printed books
- Main site

LDP Information

- FAQ
- Manifesto / license
- History
- Volunteers/Staff
- Job Descriptions
- Mailing lists
- LDP Weekly News
- Archives / RSS feed
- IRC
- Feedback
- Apparel

Workshop

LDP Wiki: The LDP Wiki is the entry point for any work in progress
Members | Authors | Visitors

Documents

HOWTOs: subject-specific help
latest updates | main index | browse by category

Guides: longer, in-depth books
latest updates / main index

FAQs: Frequently Asked Questions
latest updates / main index

man pages: help on individual commands (20060810)

Search / Resources

Links
OMF search

Google | 0 un... | Yahoo... | Rich's... | The Li... | Cabril... | linux I... |

http://www.linfo.org/index.html

The Linux Information Project

Welcome to The Linux Information Project (LINFO). This project is dedicated to providing high quality, comprehensive and easily accessible information about **Linux** and other **free software**. (New to Linux? Start [here](#).)

New on This Site:

- October 27: [root Definition](#) page updated.
- October 19: [Hard Link Definition](#) page added.
- October 12: [Characters: A Brief Introduction](#) page updated.
- October 03: [Byte Definition](#) page updated.
- September 27: [PDP-7 Definition](#) page updated.
- September 24: [The mount Command](#) page added.
- September 20: [The head Command](#) page updated.

Site Contents:

The Linux Documentation and Information Projects

Class Exercise Documentation

Use the man command on itself:

- **man man**

Research the **ls** command using:

- The **whatis** command
- The **man** command
- The **info** command
- Google

Wrap up

New commands:

apropos	- search for string in whatis database
bc	- binary calculator
cat	- print file(s)
cd	- change directory
echo	- print text
env	- show shell environment variables
info	- online documentation with hot links
file	- show file information
ls	- show directory contents
passwd	- change password
set	- show (or set) shell variables
type	- show command location in path
man	- manual page for a command
whatis	- command summary

New Files and Directories:

/etc/passwd	- user accounts
/etc/shadow	- encrypted passwords
/bin	- directory of commands
/sbin	- directory of superuser commands
/usr/bin	- directory of commands, tools and utilities
/usr/sbin	- directory of superuser commands, tools and utilities

Next Class

Assignment: Check Calendar Page on web site to see what is due next week.

Lab #2

Quiz questions for next class:

- Name four directories where one can find commands?
- How do you show your path?
- What is the command to print the manual page for a command?

Backup