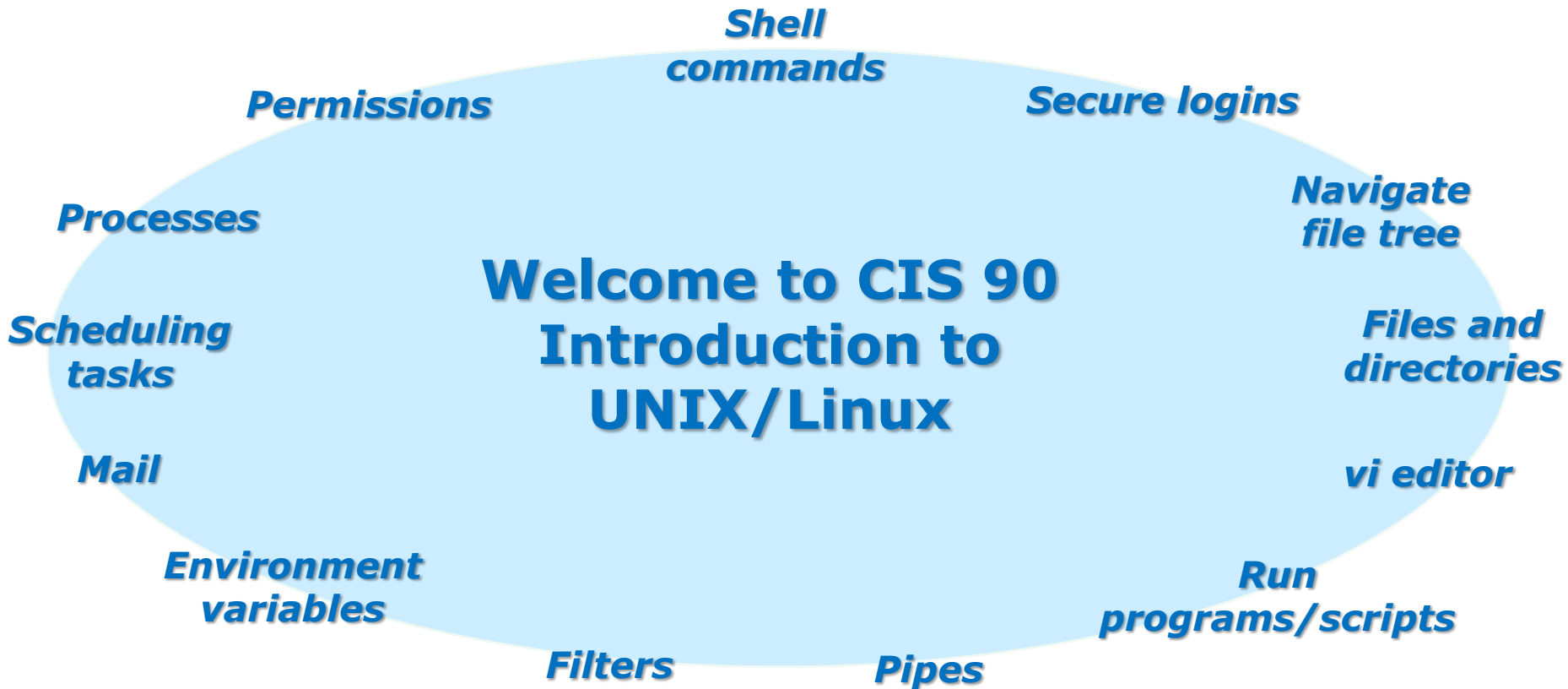


Lesson Module Checklist

- Slides
- WB converted
- Flash cards
- Page numbers
- 1st minute quiz
- Web Calendar summary
- Web book pages
- Commands
- Lab tested and uploaded
- Tech file email for Lab 9 ready
- Lock/unlock turnin directory
- Send lab 9 tech letter at noon day of class
- Apache configured for student websites
- Materials uploaded
- Backup slides, CCC info, handouts on flash drive
- Spare 9v battery for mic



Student Learner Outcomes

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

Introductions and Credits



Jim Griffin

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



Rich Simms

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

And thanks to:

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



Student checklist (How to attend from home or in the classroom)

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

Student checklist (How to layout your screen when attending class)

CCC Confer

Downloaded PDF of Lesson Slides

The screenshot displays a virtual classroom environment with several overlapping windows:

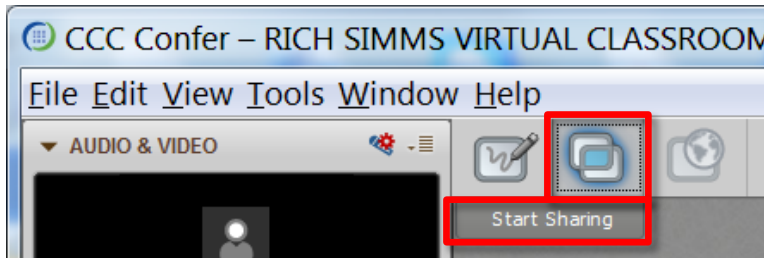
- Blackboard Co. x Google x Rich's Cabrillo x simms-teach.com**: The top browser bar shows multiple tabs.
- simms-teach.com/cis90calendar.php**: A calendar page for CIS 90 (Spring) with a sidebar containing links like Login, Flashcards, Admin, and a table with Lesson and Date columns.
- CCC Confer - RICH SIMMS VIRTUAL CLASSROOM**: A video conferencing window showing a participant named Rich-Simms and a list of participants (Benji Simms, Rich-Simms, Benji Simms (You)).
- CIS 90 - Lesson 1**: A presentation window titled "Class Activity - Where are you now?" featuring a Google map of California.
- cis90lesson01.pdf - Adobe Acrobat Pro**: A PDF window showing the "CIS 90 System Playground" slide.
- Terminal Window**: A window showing login sessions to Opus, including timestamps and IP addresses.

CIS 90 website
Calendar page

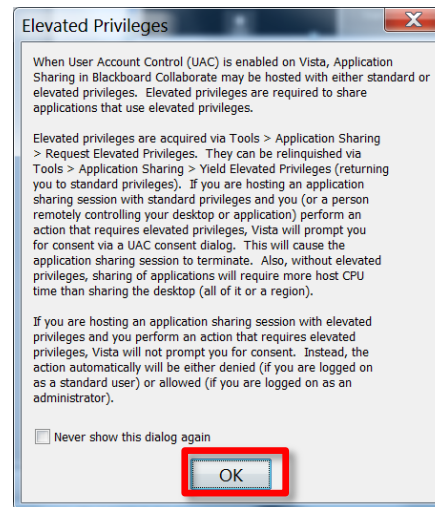
One or more login
sessions to Opus

Student checklist (To share your desktop with the class)

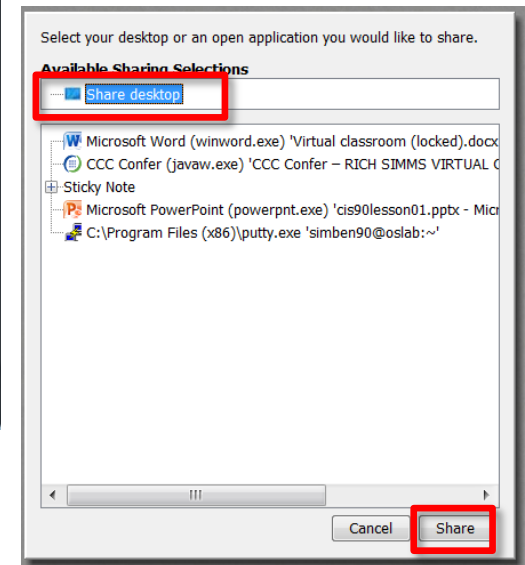
1) Instructor gives you sharing privileges



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



3) Click OK button.



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



Instructor: **Rich Simms**

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

Passcode: **136690**



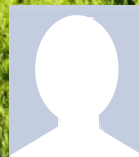
Al



Tim



Emilio



Chris



Eddie



Clara



Ryan



Ethan



Monte



Cameron



Tanner



Tess



Ahmad



Mike



Mario



Roberto



Diego



John



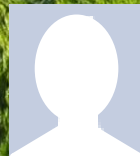
Efrain



Mateo



Django



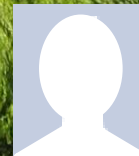
Ian



Abraham C.



Benji



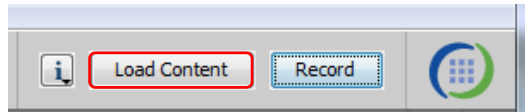
Abraham N.



Rich's CCC Confer checklist - setup

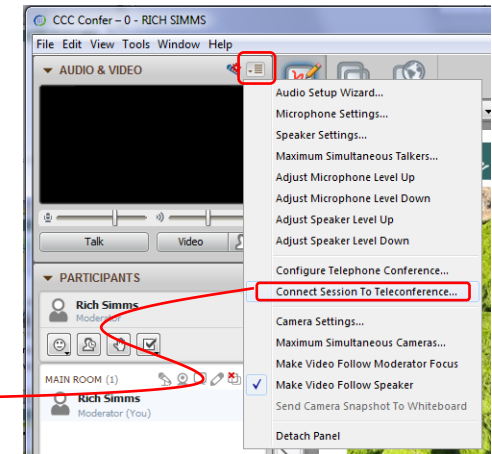
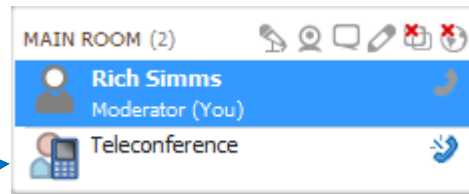


[] Preload White Board

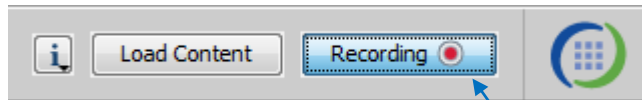


[] Connect session to Teleconference

Session now connected to teleconference



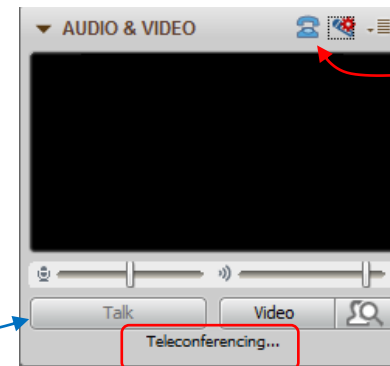
[] Is recording on?



Red dot means recording

[] Use teleconferencing, not mic

Should be greyed out



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

Rich's CCC Confer checklist - app layout



The screenshot displays a Windows desktop environment with the following components and annotations:

- CCC Confer -0 - RIC...:** A window on the left showing a video feed of Rich Simms, a list of participants (Rich Simms, Moderator), and a chat area.
- File Explorer:** A window in the center showing a directory structure with folders like 'boot', 'bin', 'etc', and 'sbin', and files like 'mail' and 'ls'. A red box labeled **foxit for slides** points to the 'mail' file.
- Terminal:** A window in the center showing a login session for 'simben90@oslab:~'. It displays the command 'ls' and its output, followed by a prompt for a command to copy files. A red box labeled **putty** points to the terminal window.
- Chrome:** A window on the right showing a web page from 'simms-teach.com/docs/cis90/cis-90-TEST-1-Fall-12.pdf'. It contains flashcard questions. A red box labeled **chrome** points to the browser window.
- vSphere Client:** A window on the bottom right showing the vCenter interface with a list of virtual machines. A red box labeled **vSphere Client** points to the window.

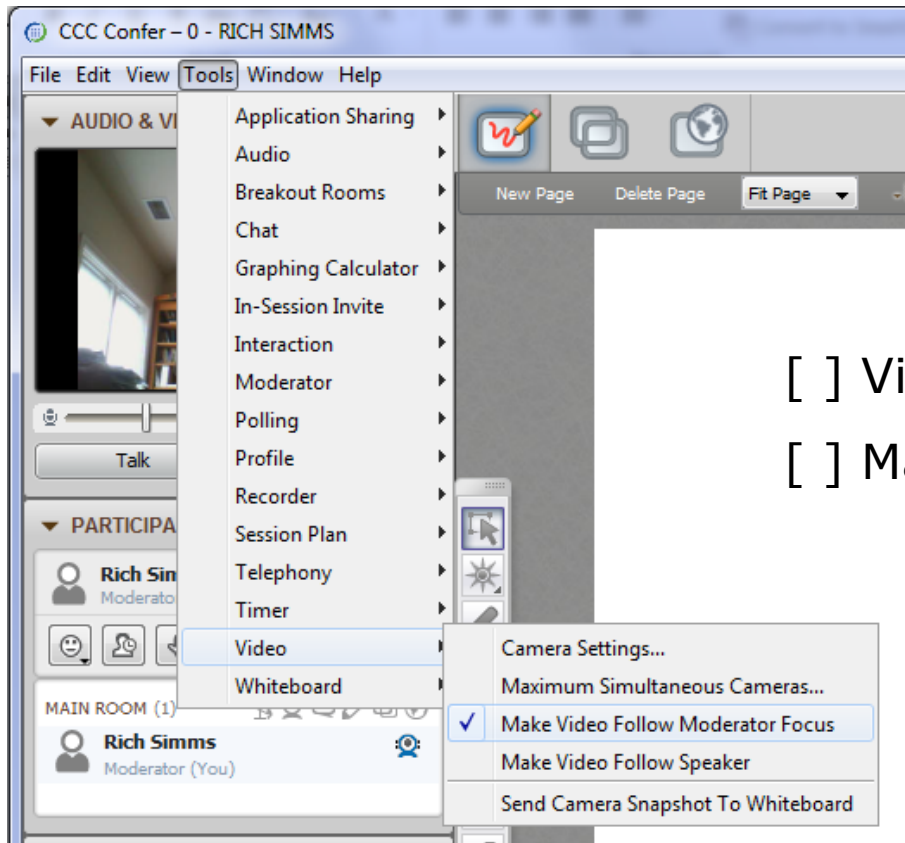
Red lines connect the text labels to their respective windows in the screenshot.

[] layout and share apps





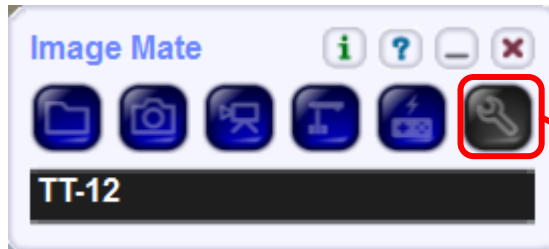
Rich's CCC Confer checklist - video



[] Video (webcam)

[] Make Video Follow Moderator Focus

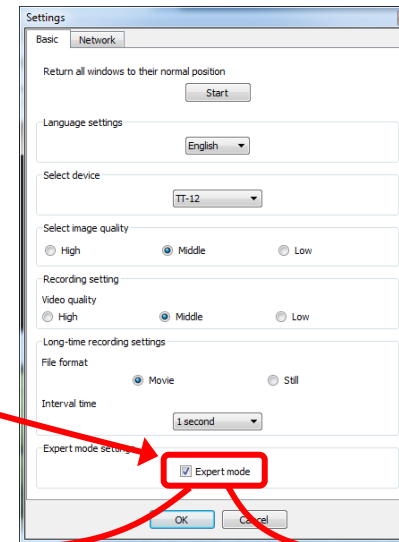
Rich's CCC Confer checklist - Elmo



Elmo rotated down to view side table



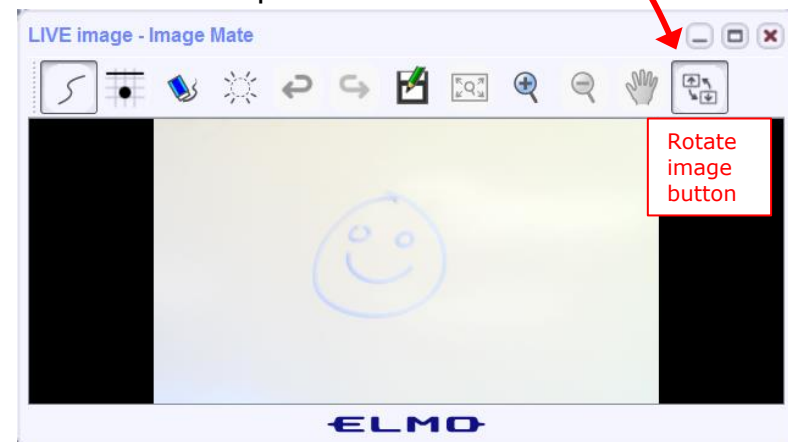
Run and share the Image Mate program just as you would any other app with CCC Confer



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

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

Elmo rotated up to view white board



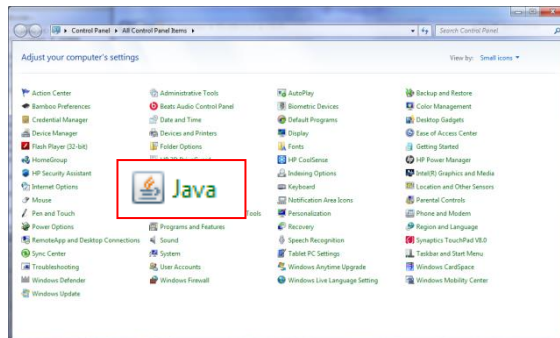
Rich's CCC Confer checklist - universal fix



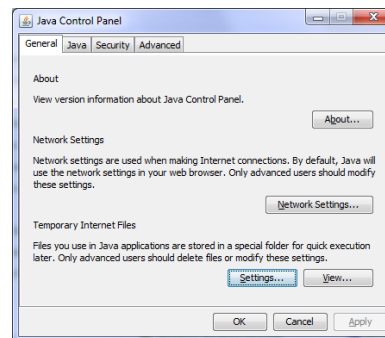
Universal Fix for CCC Confer:

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

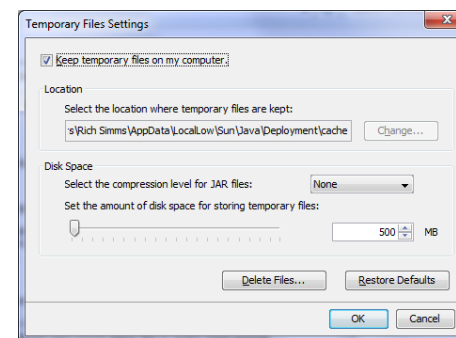
Control Panel (small icons)



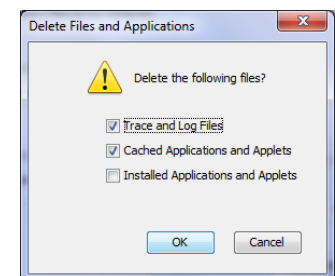
General Tab > Settings...



500MB cache size



Delete these



Google Java download



Quiz

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

See electronic white board

email answers to: risimms@cabrillo.edu

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

vi editor

Objectives	Agenda
<ul style="list-style-type: none">• Create and modify text files	<ul style="list-style-type: none">• Quiz• Questions• Housekeeping• grep workout• Shell six steps (review)• Signals (review)• Using &• Job control (review)• Load balancing & scheduling (review)• Text editors• vi 101• vi modes• Tangent on spell• Wrap up

Sound Check

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

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

Questions



Questions?

Lesson material?

Labs? Tests?

How this course works?

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

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

- Francis Bacon

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

- Mahatma Gandhi

Chinese
Proverb

他問一個問題，五分鐘是個傻子，他不問一個問題仍然是一個傻瓜永遠。

He who asks a question is a fool for five minutes; he who does not ask a question remains a fool forever.



Test 2

Post Mortem

Test 2 – Results

Missed Q26 = 19
Missed Q4 = 18
Missed Q30 = 17
Missed Q25 = 17
Missed Q29 = 16
Missed Q24 = 16
Missed Q27 = 15
Missed Q22 = 15
Missed Q21 = 15
Missed Q20 = 15
Missed Q23 = 14
Missed Q28 = 13
Missed Q17 = 13
Missed Q19 = 12
Missed Q18 = 12

Missed Q3 = 10
Missed Q2 = 10
Missed Q13 = 9
Missed Q11 = 8
Missed Q15 = 7
Missed Q12 = 7
Missed Q9 = 6
Missed Q6 = 6
Missed Q14 = 6
Missed Q10 = 6
Missed Q7 = 5
Missed Q16 = 5
Missed Q8 = 4
Missed Q5 = 2
Missed Q1 = 1

Extra Credit

Missed Q33 = 19

Missed Q32 = 19

Missed Q31 = 18



*20 tests
submitted*



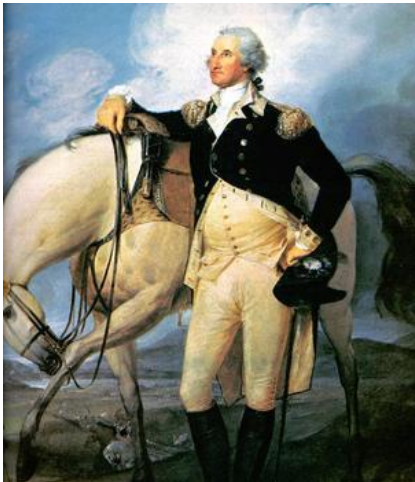
*5 tests not
submitted*





Q16) There is a file in the `/etc` directory named *passwd*. This file has information on all user accounts including usernames, UIDs, first and last name, etc. What is the absolute pathname of this file?

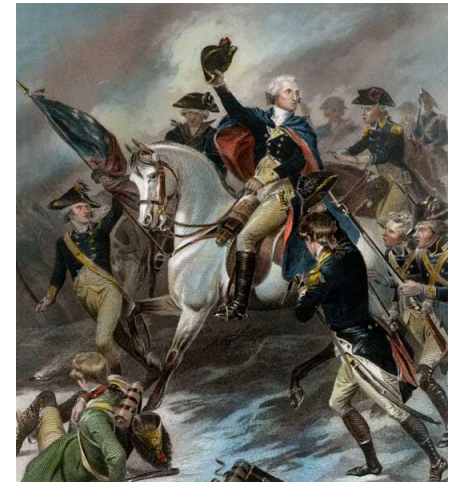
Correct answer:
`/etc/passwd`



<http://www.sodahead.com/united-states/what-color-was-george-washingtons-white-horse/question-636725/>



<http://kids.britannica.com/comptons/art-55428/General-George-Washington-and-his-staff-welcoming-a-provision-train>



<http://www.mountvernon.org/content/revolutionary-war-princeton-white-horse>

Questions today on Test #2?

If not, anyone interested in another Sunday afternoon workshop on April 19th or 26th?

The goal would be to be able to answer each Test #2 question in less than 30 seconds.



Housekeeping

1. Lab 8 due tonight

at 11:59pm

at> cat files.out bigshell > lab08

at> cp lab08 /home/rsimms/turnin/cis90/lab08.\$LOGNAME

at> **<Ctrl-D>**

Don't wait till midnight tonight to see if this worked! Submit with an earlier time.

2. A check8 script is available for Lab 8
3. Note: Lab 9 and five posts due next week
4. You can still send me your photo for our class page if you want 3 points extra credit

Tentative CS/CIS Technology Career Workshop



Breakthroughs happen here.™

Computer Information Systems & Computer Science



Computer Information Systems & Computer Science

Technology Career Meetup

When: Friday May 1st, 2015 @ 12 noon
Where: Cabrillo College Room 828

Learn more about careers in Computer Science (CS) and Computer and Information Systems (CIS) and about classes for Spring 2015. Meet CS and CIS Faculty.

[Cabrillo Home](#) | [Library](#) | [Blackboard](#) | [Learning Resources](#) | [Student Services](#) | [Directories A-Z](#) | [WebAdvisor](#) | [Cabrillo Wiki](#)

[Recent Site Activity](#) | [Report Abuse](#) | [Print Page](#) | [Remove Access](#) | Powered By [Google Sites](#)

Final Exam

Test #3 (final exam) is **MONDAY** May 18 7-9:50AM

Monday	5/18	<p>Test #3 (the final exam)</p> <p>Time</p> <ul style="list-style-type: none"> 7:00AM - 9:50AM in Room 828 <p>Materials</p> <ul style="list-style-type: none"> Test (blackboard) <p>CCC Confer</p> <ul style="list-style-type: none"> Enter virtual classroom Class archives 		<p><u>5 posts</u> <u>Lab X1</u> <u>Lab X2</u></p>
---------------	------	--	--	---

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

- All students will take the test at the same time. The test must be completed by 9:50AM.
- Working and long distance students can take the test online via CCC Confer and BlackBoard.
- Working students will need to plan ahead to take time off from work for the test.

Monitoring your grades

Points that could have been earned:

7 quizzes:	21 points
7 labs:	210 points
2 tests:	60 points
2 forum quarters:	40 points
Total:	331 points

The CIS 90 website

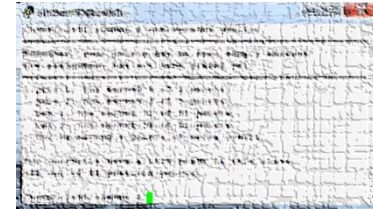


*Send me your
survey to get your
LOR code name.*

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

Or on Opus

checkgrades codename
(where codename is your LOR codename)

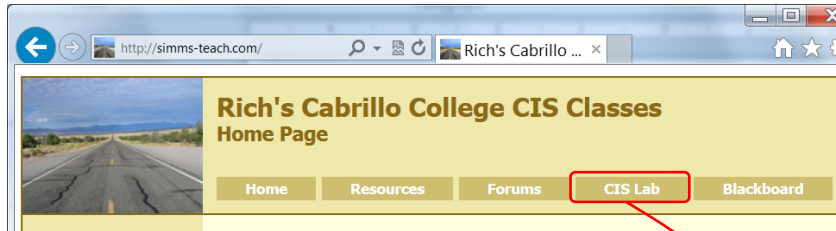


*The checkgrades script was written by
Jessie a past CIS 90 Alumnus*

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

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

Want some help working the labs?



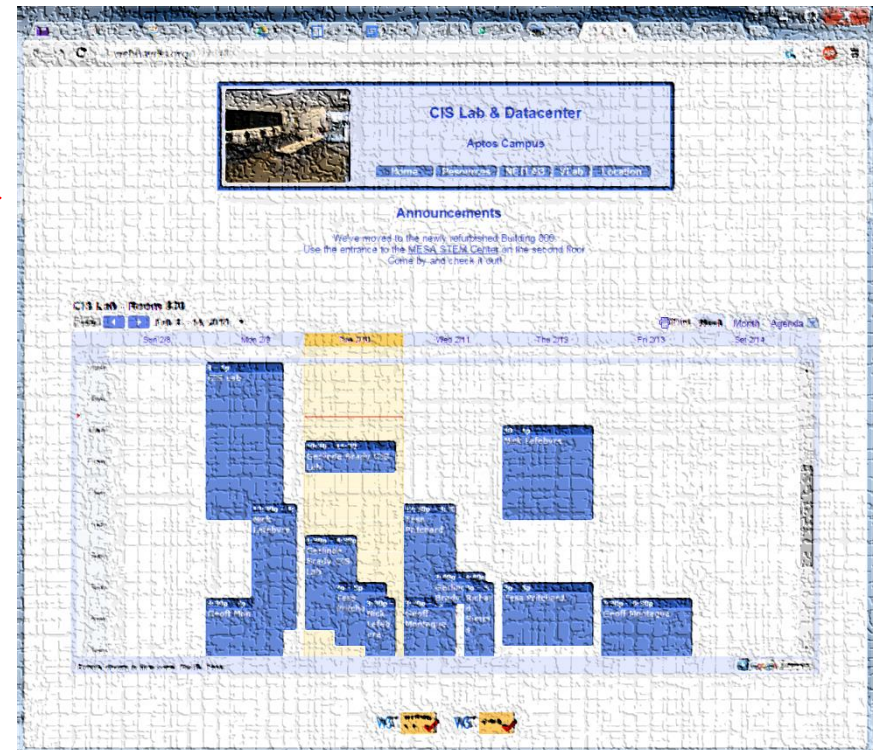
If you would like some additional come over to the CIS Lab. There are student lab assistants and instructors there to help you.

*Nick, Sean, and Geoff are
CIS 90 Alumni.*

*Tess and Mike F. are in our
class!*

*Michael M. is the other
Linux instructor.*

*I'm in there Wednesdays
3:00-5:30pm.*



CIS 90 Tutoring Available

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

Search...

Quick links FAQ Register Login

Board index < Cabrillo College Spring 2015 Courses < CIS 90 - Spring 2015

Introduction

Post Reply Search this topic... 2 posts • Page 1 of 1

Introduction

by **Geoffrey Montague** » Fri Mar 20, 2015 12:01 pm

Hello CIS 90!

I just wanted to introduce myself to those I haven't yet met in the class. I'm a student assistant that works in the STEM Center (Room 830). I'm there Mondays and Wednesdays from 3:30 to 8:30, and Fridays 11:30 to 3. Feel free to stop by anytime with CIS 90 questions or just to say hello. There are also other awesome student assistants and instructors that are there to help you. Here is the schedule for those who work in the CIS Department at the STEM Center: <http://webhawks.org/~cislab/>. Additionally if you would like to schedule a workshop with me you can send me an email (gmontague0205 (AT) gmail.com), and I will try to respond back within 24 hours.

Cheers!

-Geoff

Geoffrey Montague
Posts: 24
Joined: Thu Feb 03, 2011 7:06 pm



Geoffrey Montague

Contact Geoff at:

gmontague0205 (AT) gmail.com

More CIS 90 Tutoring Available

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

Search...

[Quick links](#) [FAQ](#) [Register](#) [Login](#)

[Board index](#) < [Cabrillo College Spring 2015 Courses](#) < [CIS 90 - Spring 2015](#)

Takashi Tamasu

Tutoring Available

[Post Reply](#) [Search this topic...](#) 2 posts • Page 1 of 1

Tutoring Available
by **Takashi Tamasu** » Wed Mar 11, 2015 9:45 am

Hi all,
I am the Alpha Gamma Sigma tutoring coordinator and there are free tutoring available for your class.
As a campus community service, AGS students tutor other Cabrillo Students FREE of charge. In exchange for their time and energy, tutors receive service points towards their AGS membership requirement and awards.

There are paper forms next to the AGS inbox or you can do it online via <http://goo.gl/forms/sHHfZkSc8f>

Takashi Tamasu
Posts: 105
Joined: Wed Jan 29, 2014 2:46 pm

grip workout



Some perfect times to use the **grep** command:

- 1) To search through the output of a command for some text

```
command | grep "text string"
```

- 2) To search inside one or more files for some text

```
grep "text string" file1 file2 ... fileN
```

- 3) To search (recursively) inside all files in a branch of the UNIX file tree for some text

```
grep -R "text string" directory
```

grep usage – search output of a command

Is the CUPS daemon (print service) running right now?

```
/home/cis90/simben $ ps -ef | grep cups  
root      6251      1   0   Jul31   ?           00:00:04 cupsd -C /etc/cups/cupsd.conf  
simben90  27027  26966   0   08:47 pts/3      00:00:00 grep cups
```

Yes it is, with PID=6251

grep practice

- Is the cronjob daemon (crond) running right now?
- Type the crond PID into the chat window

grep usage – search output of a command

Is the Apache web server (httpd) installed?

*This shows all installed
package names*

*This searches for package
names containing "httpd"*

```
/home/cis90/simben $ rpm -qa | grep httpd  
httpd-tools-2.2.15-15.el6.centos.1.i686  
httpd-2.2.15-15.el6.centos.1.i686  
httpd-manual-2.2.15-15.el6.centos.1.noarch
```

Yes, version 2.2.15 has been installed

grep practice

- Has the mysql-server package been installed on Opus?
- If installed on Opus, type the version of mysql in the chat window

grep usage – search output of a command

When were the last 5 times I logged in?

```
/home/cis90/simben $ last | grep $LOGNAME | head -n5
```

simben90	pts/0	50-0-68-235.dsl.	Mon Apr 23 05:39	still	logged in
simben90	pts/6	10.64.25.2	Wed Apr 18 12:48 - 16:51	(04:02)	
simben90	pts/5	10.64.25.2	Wed Apr 18 12:48 - 16:51	(04:02)	
simben90	pts/4	10.64.25.2	Wed Apr 18 12:48 - 16:51	(04:03)	
simben90	pts/1	50-0-68-235.dsl.	Wed Apr 18 09:06 - 10:23	(01:17)	

This scans the latest wtmp log file and lists your most recent five logins to Opus

grep practice

- For the time period covered by the current wtmp log file. What was the date of your earliest login?
- Type your earliest login date into the chat window

grep usage – search output of a command

```
[rsimms@oslab ~]$ ls /bin/*sh
```

```
/bin/bash /bin/csh /bin/dash /bin/ksh /bin/rbash /bin/sh /bin/tcsh
```

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

```
$ dash
```

```
$ sh
```

```
sh-4.1$ csh
```

Look familiar? (lab 8) Shows how to compare shells by size and record the biggest one in a file.

```
[rsimms@oslab ~]$ ps -l
```

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
0	S	201	27553	27552	0	80	0	-	1308	-	pts/0	00:00:00	bash
0	S	201	27651	27553	0	80	0	-	1376	-	pts/0	00:00:00	ksh
0	S	201	27652	27651	0	80	0	-	517	-	pts/0	00:00:00	dash
0	S	201	27653	27652	0	80	0	-	1307	-	pts/0	00:00:00	sh
0	S	201	27654	27653	0	80	0	-	1458	-	pts/0	00:00:00	csh
0	R	201	27663	27654	0	80	0	-	1214	-	pts/0	00:00:00	ps

size

```
[rsimms@oslab ~]$ ps -l | grep csh
```

```
0 S 201 27654 27653 0 80 0 - 1458 - pts/0 00:00:00 csh
```

```
[rsimms@oslab ~]$ ps -l | grep csh > bigshell
```

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

```
0 S 201 27654 27653 0 80 0 - 1458 - pts/0 00:00:00 csh
```




grep practice

Instructor note: add write permission to others on Benji's terminal device

- Run **bash**, **dash**, **ksh**, **sh** and **cs** shells and use **ps -l** to see which is the smallest.
- Redirect the line of **ps -l** output for the smallest shell to Benji Simms's terminal: **/dev/pts/??**
- Sign it with **echo "From *first name*" > /dev/pts/??**
- Then **exit** each shell till your are back to just one bash shell running.

grep usage – search inside files

How many CIS 90 user accounts are there?

```
/home/cis90/simben $ grep cis90 /etc/passwd | wc -l  
29
```

There are 29

grep practice

- How many CIS 172 accounts are there on Opus?
- Type the number of CIS 172 accounts into the chat window

grep usage – search inside files

Example: What is my account information in /etc/passwd?

```
/home/cis90/simben $ grep $LOGNAME /etc/passwd  
simben90:x:1000:90:Benji Simms:/home/cis90/simben:/bin/bash
```

or

```
/home/cis90/simben $ grep simben90 /etc/passwd  
simben90:x:1000:90:Benji Simms:/home/cis90/simben:/bin/bash
```

or

```
/home/cis90/simben $ cat /etc/passwd | grep $LOGNAME  
simben90:x:1000:90:Benji Simms:/home/cis90/simben:/bin/bash
```

Diagram illustrating the fields in the output of the `grep` command, with arrows pointing to the corresponding fields in the output line:

- username* (points to `simben90`)
- password (just a placeholder now)* (points to `x`)
- User ID (UID)* (points to `1000`)
- Group ID (GID)* (points to `90`)
- Comment* (points to `Benji Simms`)
- Home directory* (points to `/home/cis90/simben`)
- Shell* (points to `/bin/bash`)

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

grep practice

- Does your user ID in */etc/passwd* match the uid output by the **id** command?
- Type your answer (yes or no) and your uid from the **id** command into the chat window

grep usage – search inside files in all or part of the file tree

Where does the PS1 "prompt" variable get set?

```
/home/cis90/simben $ grep -R "PS1=" /etc/bash* $HOME 2> /dev/null
/etc/bash_completion.d/git:# PS1='[\u@\h \W$(__git_ps1 "
(%s)"]\ $ '
/etc/bashrc: [ "$PS1" = "\s-\v\\\$ " ] && PS1="[\u@\h \W]\$ "
/etc/bashrc: # PS1="[\u@\h:\l \W]\$ "
/home/cis90/simben/class/labs/lab04.graded:21) PS1='$PWD $ '
/home/cis90/simben/class/exams/test01.graded:(A32) PS1='\d $ '
/home/cis90/simben/.bash_profile:PS1='$PWD $ '
/home/cis90/simben/lab04.graded:21) PS1='$PWD $ '
/home/cis90/simben/test01.graded:(A32) PS1='\d $ '
```

It is set more than once during login. We will learn in a future lesson that the one in .bash_profile is done last and is what you end up using.

grep practice

- Find the file in the */usr/lib* portion of the file tree that contains "hot pototo dance" (yes, potato is misspelled).
- Type the absolute pathname of the file in the chat window.

grep usage – search inside files in all or part of the file tree

```
simben90@oslab:~  
/home/cis90/simben $ grep Benji /etc/passwd  
simben90:x:1047:190:Benji Simms:/home/cis90/simben:/bin/bash  
/home/cis90/simben $  
/home/cis90/simben $  
/home/cis90/simben $ grep --color "Benji" /etc/passwd  
simben90:x:1047:190:Benji Simms:/home/cis90/simben:/bin/bash  
/home/cis90/simben $  
/home/cis90/simben $  
/home/cis90/simben $ grep -R --color "Benji" /etc/p*  
/etc/passwd:simben90:x:1047:190:Benji Simms:/home/cis90/simben:/bin/bash  
/etc/passwd-:simben90:x:1047:190:Benji Simms:/home/cis90/simben:/bin/bash  
/etc/passwd.OLD:simben90:x:1001:190:Benji Simms:/home/cis90/simben:/bin/bash  
grep: /etc/pki/dovecot/private/dovecot.pem: Permission denied  
grep: /etc/pki/dovecot/certs/dovecot.pem: Permission denied  
grep: /etc/pki/CA/private: Permission denied  
grep: /etc/pki/rsyslog: Permission denied  
grep: /etc/pki/tls/private/localhost.key: Permission denied  
grep: /etc/pki/tls/certs/localhost.crt: Permission denied  
grep: /etc/polkit-1/localauthority: Permission denied  
/home/cis90/simben $ █
```

*Use color with the **--color** option*

Shell six steps (REVIEW)

Example Command

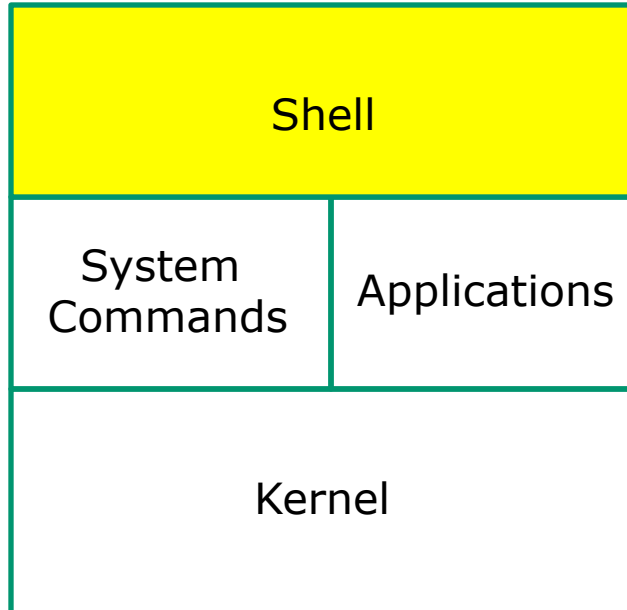
```
/home/cis90/simben $ find / -name *egg 2> /dev/null
```

```
/home/cis90/vasjor/1968.egg  
/home/cis90/cis/1968.egg  
/home/cis90/hictre/1968.egg  
/home/cis90/rodduk/basket/1968.egg  
/home/cis90/lemrob/1968.egg  
/home/cis90/fahmic/basket/1968.egg  
/home/cis90/casric/1968.egg  
/home/cis90/calmic/1968.egg  
/home/cis90/mosmic/1968.egg  
/home/cis90/mccpat/1968.egg  
/home/cis90/matjon/basket/1968.egg  
/home/cis90/tilbuz/basket/1968.egg  
/home/cis90/daweli/1968.egg  
< snipped >  
/home/cis90/rudtro/1968.egg  
/home/cis90/keljos/basket/1968.egg  
/home/cis90/thinic/1968.egg  
/home/cis90/casenr/1968.egg  
/home/cis90/fitcon/1968.egg  
/home/cis90/simben/1968.egg  
/home/cis90/vivrut/1968.egg  
/home/cis90/specod/1968.egg  
/home/cis90/ramjua/1968.egg  
/home/cis90/lefnic/basket/1968.egg  
/home/cis90/hormat/1968.egg  
/home/cis90/simben $
```

*On the next slides we
will walk through each
of the six steps the shell
performs for this
command*



Prompt Step



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



Prompt Step

(uses **PS1** variable)

/home/cis90/simben \$



The shell prompt is output from the bash shell program directed to your terminal device

- Benji is using the bash shell. There are many other shells such as sh, ksh and csh. In */etc/passwd* the last field in the line for his account determines the shell that is run when logging in.
- The bash program resides in the */bin* directory
- The command prompt appearance is defined by the PS1 variable. You can output a prompt yourself using **echo \$PS1**

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

/home/cis90/simben $ ls -l /bin/bash
-rwxr-xr-x. 1 root root 874248 May 10 2012 /bin/bash
```




Prompt Step

*Note there is an invisible
<newline> metacharacter at
the end of the command*

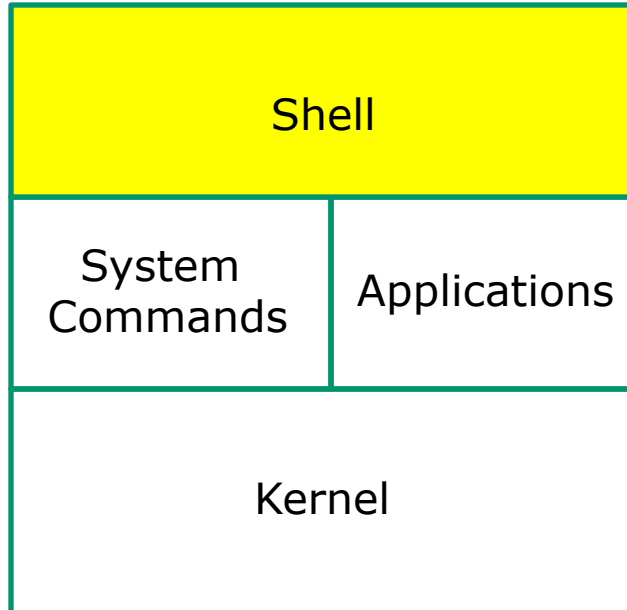
```
/home/cis90/simben $ find / -name *egg 2> /dev/null
```

*Benji types this find command in
response to the shell prompt*

The prompt step is not complete until the user type the Enter/Return key



Parse Step



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



Parse Step

The shell uses spaces to separate options, arguments and redirection

find **/** **-name** ***egg** **2>** **/dev/null**

The shell must expand filename expansion characters and variables during the parse step. Note there is an invisible <newline> metacharacter at the end of the command

Parsing RESULTS:

Command: **find**

Options and arguments:

/

-name

1968.egg

This will be passed to the command (if the command can be located on the path)

Redirection:

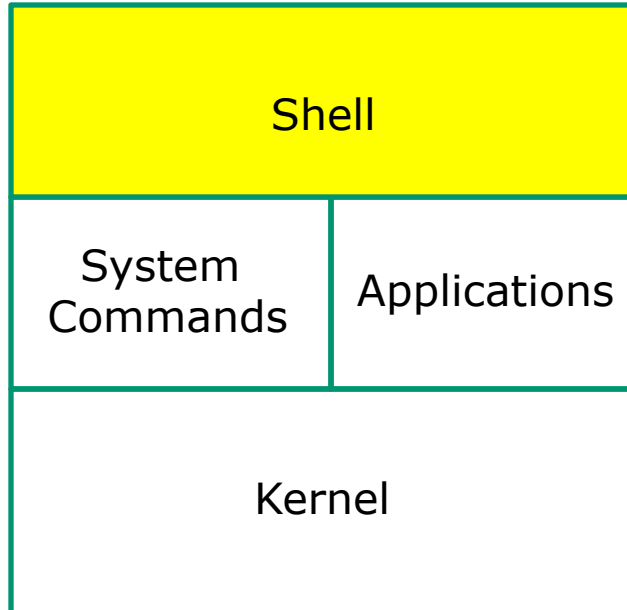
Connect **stderr** to **/dev/null** (the "bit bucket")

This will be handled by the shell. The command, if loaded, will not see this

*Note: Because Benji had a 1968.egg file in his home directory, the shell expands *egg to 1968.egg*



Search Step



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



Search Step

(uses **PATH** variable)

Command: **find**

*The shell now must search, in order, every directory on Benji's path to locate the first occurrence of the **find** command.*

Benji's path is defined by the value of his PATH variable

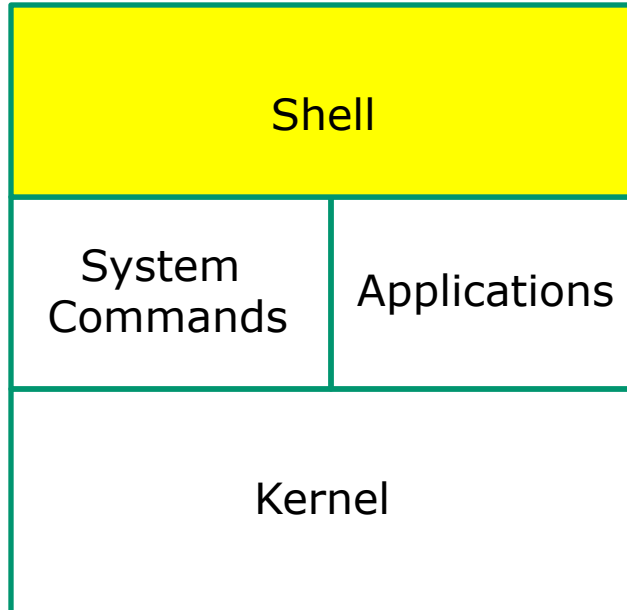
- 1st directory searched: /usr/lib/qt-3.3/bin
- 2nd directory searched: /usr/local/bin
- 3rd directory searched: **/bin**
- 4th directory searched: /usr/bin
- 5th directory searched: /usr/local/sbin
- 6th directory searched: /usr/sbin
- 7th directory searched: /sbin
- 8th directory searched: /home/cis90/simben/../../bin
- 9th directory searched: /home/cis90/simben/bin
- 10th directory searched: .

*The shell locates the
find command in the
/bin directory*

```
/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:.
/home/cis90/simben $ type find
find is /bin/find
```



Execute Step

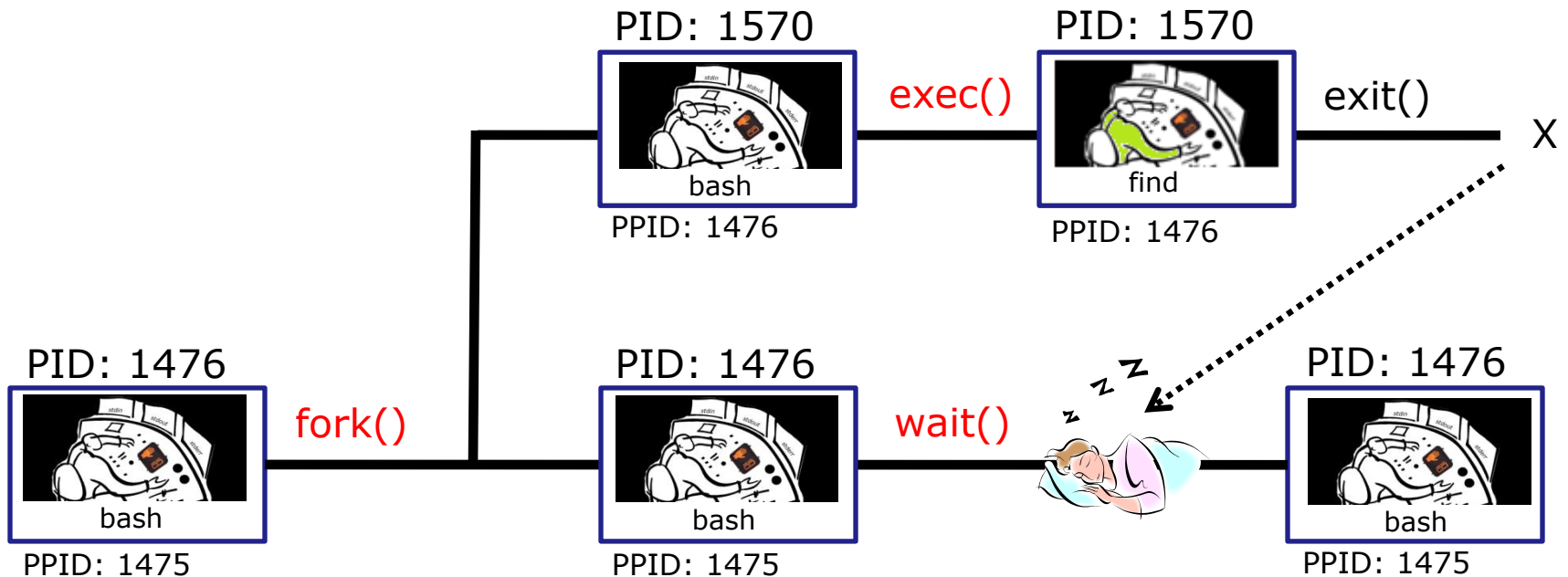


- 1) Prompt
- 2) Parse
- 3) Search
- 4) Execute**
- 5) Nap
- 6) Repeat





Execute Step



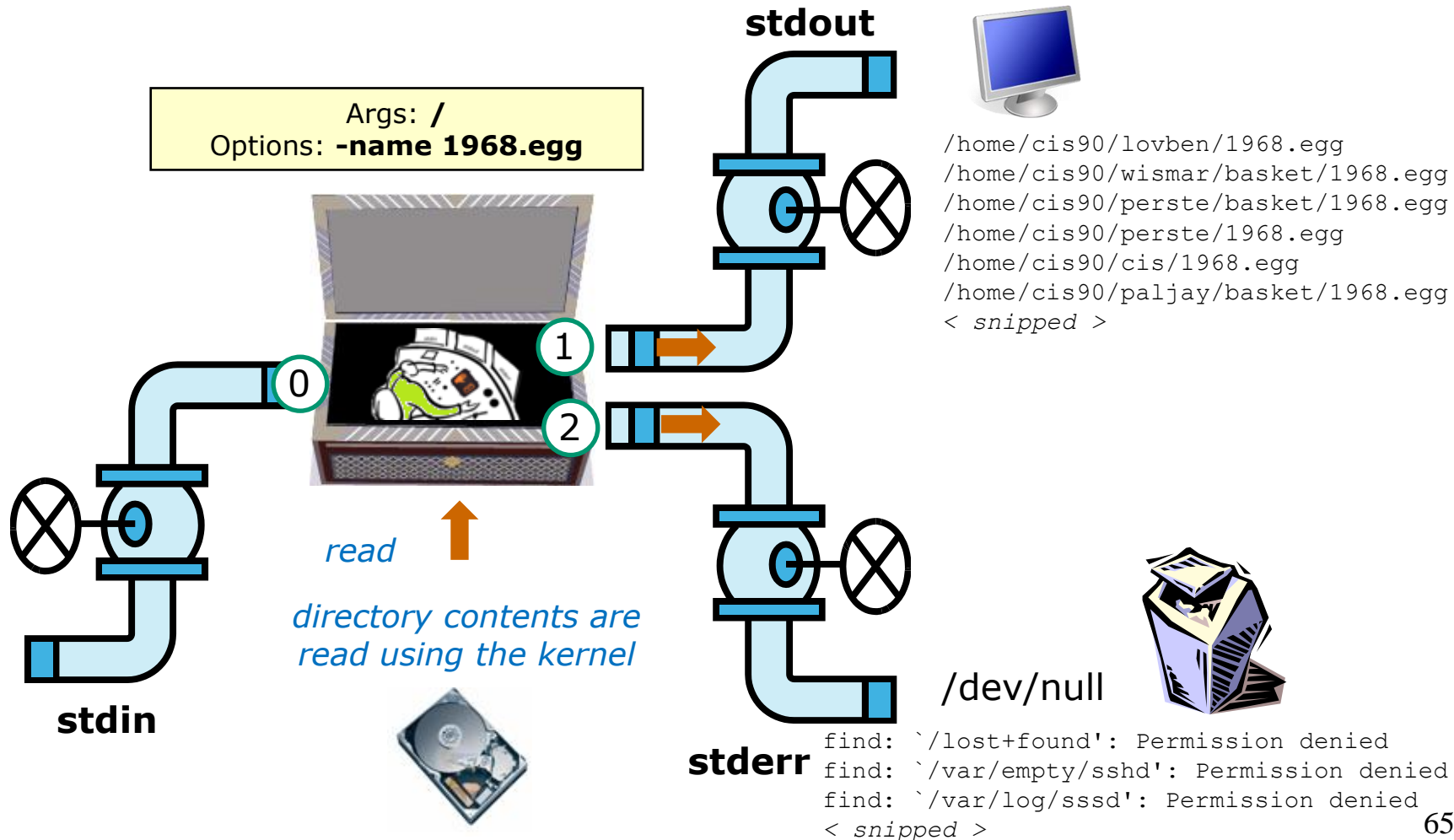
bash executes the ***find*** command by:

- 1) Cloning itself with a ***fork()*** system call to create a new child process.
- 2) With an ***exec()*** system call, the new child process is overlaid with the *find* code instructions.
- 3) *bash* sleeps by making a ***wait()*** system call while the *find* child process runs.
- 4) The child process makes an ***exit()*** system call when it has finished.
- 5) After that, the parent *bash* process wakes up and the child process is killed.



Execute Step

```
/home/cis90/simben $ find / -name *egg 2> /dev/null
```



This is what the find process might look like



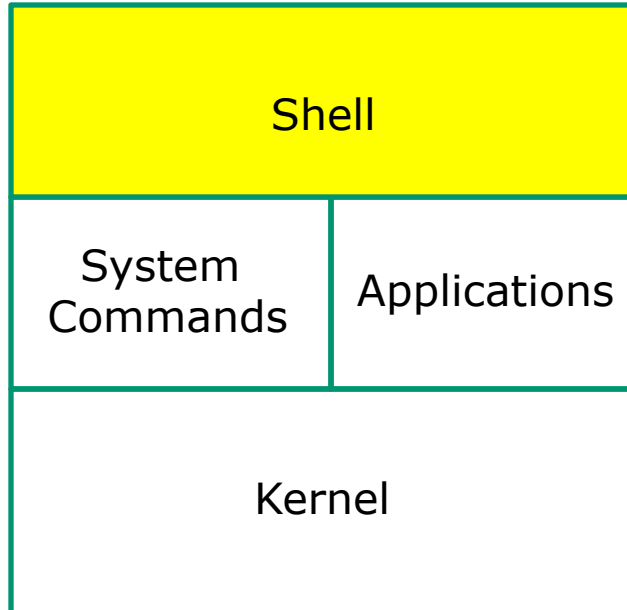
A process:

- Is provided with parsed & expanded options and arguments from the shell
- may read from **stdin**
- may write to **stdout**
- may write error messages to **stderr**
- and may get interrupted from time to time by a **signal**

The **find** process is running



Nap Step

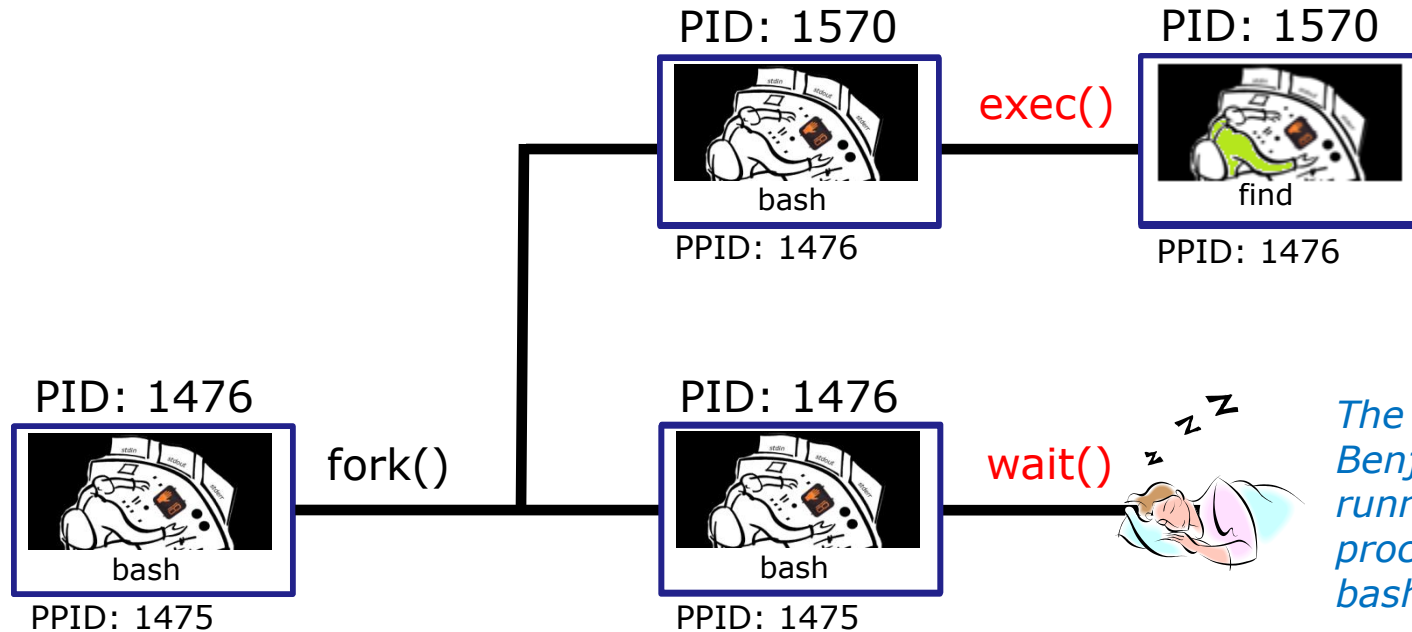


- 1) Prompt
- 2) Parse
- 3) Search
- 4) Execute
- 5) Nap**
- 6) Repeat





Nap Step



The `PS` command shows Benji's **find** command is running as a child process while the parent bash shell sleeps

Sleeping

```
[rsimms@oslab ~]$ ps -l -u simben90
```

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
5	S	1001	1475	1470	0	80	0	-	3392	?	?	00:00:00	sshd
0	S	1001	1476	1475	0	80	0	-	1308	?	pts/1	00:00:00	bash
0	R	1001	1570	1476	40	80	0	-	1179	?	pts/1	00:00:00	find

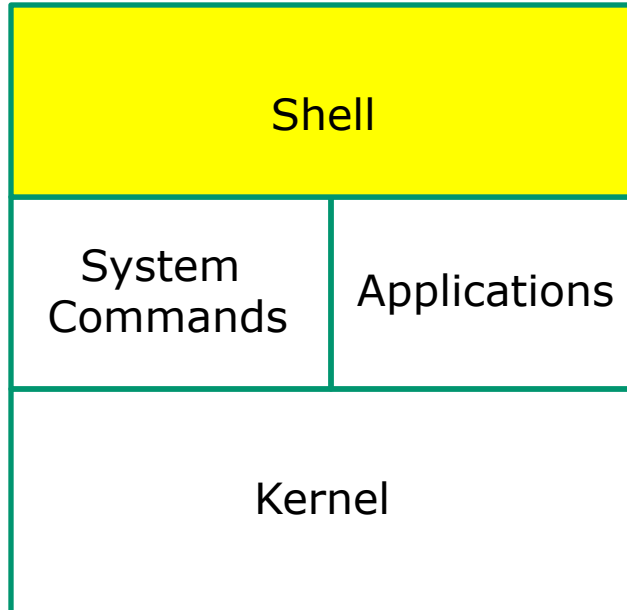
Parent

Child

Running



Repeat Step

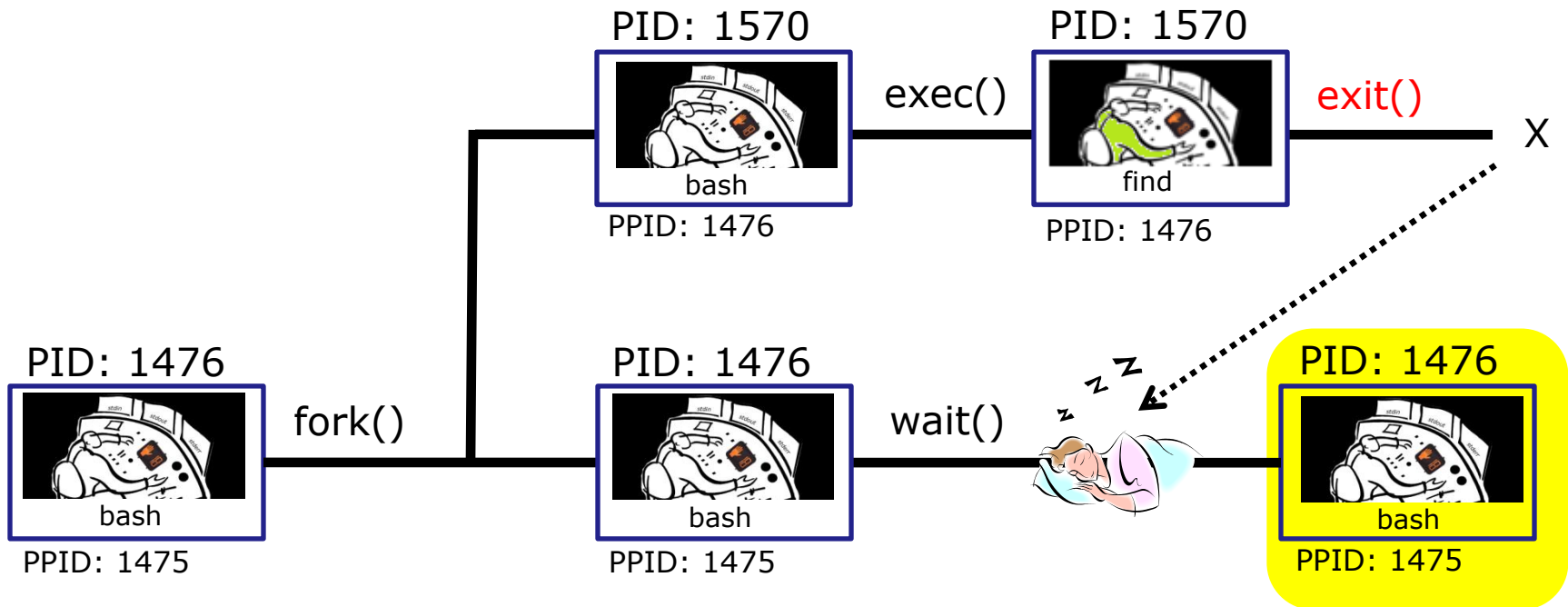


- 1) Prompt
- 2) Parse
- 3) Search
- 4) Execute
- 5) Nap
- 6) Repeat**





Repeat Step



*The child process makes an **exit()** system call when it has finished. The parent bash process wakes up, the child process is killed and we are ready to start the process all over again with the next command.*

Process activity

- See if you can do a **ps** command that illustrates what happens when a user runs a long **grep** command.
- The **ps** output should show "parent" bash S=Sleeping while the "child" **grep** command is either R=Running or in D=Uninterruptible sleep (IO)
- Start a second login session to observe your processes
- Write your grep PID and status into the chat window when done

/home/cis90/simben \$ **grep -r "pototo" /usr**

```
simben90@oslab:~
/home/cis90/simben $ grep -r "pototo" /usr/lib /usr/src
grep: /usr/lib/audit: Permission denied
/usr/lib/perl5/Net/DNS/Resolver/Recurse.pm:# Purpose: Do that "hot pototo dance"
on args.
grep: /usr/lib/cups/backend/serial: Permission denied
grep: /usr/lib/cups/backend/ipp: Permission denied
grep: /usr/lib/cups/backend/http: Permission denied
grep: /usr/lib/cups/backend/dnssd: Permission denied
grep: /usr/lib/cups/backend/lpd: Permission denied
grep: /usr/lib/cups/backend/mdns: Permission denied
grep: /usr/lib/cups/backend/https: Permission denied
/home/cis90/simben $
```

/home/cis90/guest \$ **ps -lu simben90**

```

simben90  2899  8841  8820  0  80  0 - 2899  ?    pts/0    00:00:00  sshd
0  S  1001  8842  8841  0  80  0 - 1308  ?    pts/0    00:00:00  bash
0  D  1001  9032  8842  21  80  0 - 1369  ?    pts/0    00:00:02  grep
/home/cis90/guest $ ps -lu simben90
F S  UID  PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4  S  1001  6283  6270  0  80  0 - 1308  ?    pts/1    00:00:00  bash
5  S  1001  8841  8820  0  80  0 - 2899  ?    ?         00:00:00  sshd
0  S  1001  8842  8841  0  80  0 - 1308  ?    pts/0    00:00:00  bash
0  D  1001  9032  8842  21  80  0 - 1369  ?    pts/0    00:00:02  grep
/home/cis90/guest $ ps -lu simben90
F S  UID  PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4  S  1001  6283  6270  0  80  0 - 1308  ?    pts/1    00:00:00  bash
5  S  1001  8841  8820  0  80  0 - 2899  ?    ?         00:00:00  sshd
0  S  1001  8842  8841  0  80  0 - 1308  ?    pts/0    00:00:00  bash
0  R  1001  9032  8842  23  80  0 - 1369  ?    pts/0    00:00:03  grep
/home/cis90/guest $
```



Review of Signals

Signals

PLATE 4

COMMERCIAL CODE SIGNALS

EXAMPLES OF THE SEVERAL HOISTS

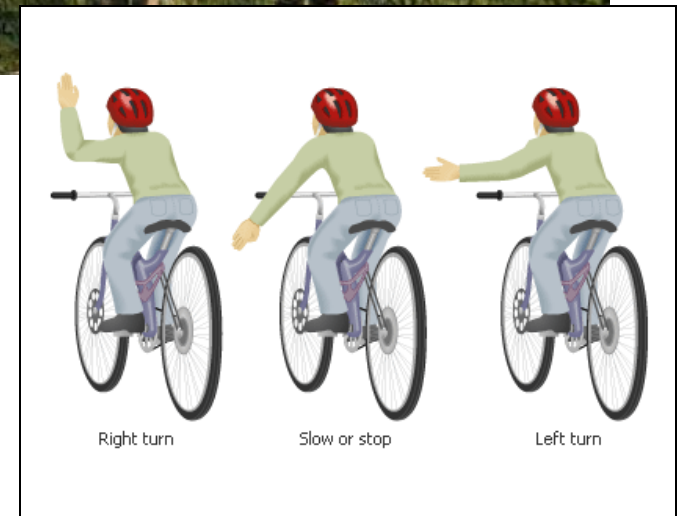
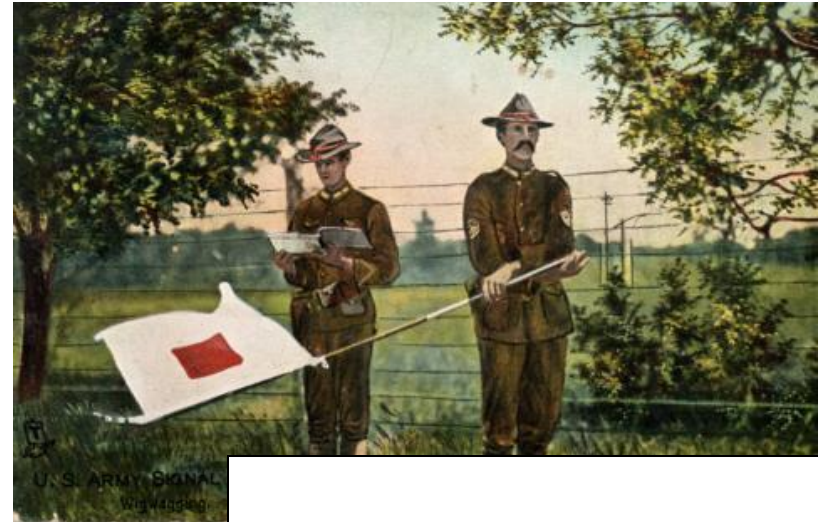
WHICH CAN BE MADE HAVING TWO, THREE, OR FOUR FLAGS.

When a word contains two letters of the same name, the second time of its occurrence it must begin or be in the 2nd Hoist; and on its 3rd occurrence, it must begin or be in the 3rd Hoist.

URGENT & IMPORTANT SIGNALS		COMPASS SIGNALS		3 FLAGS			
CODE FLAG OVER 1 FLAG OR 2 FLAG SIGNALS							
CODE FLAG P	A	A	Q	A	K		
"I Am about to Sail"	"Do Not"	N 1/2 E	E	X			
	"abandon the Vessel"				S 3/4 W		
LATITUDE & LONGITUDE SIGNALS			CODE FLAG OVER 2 FLAGS				
CODE FLAG	Q	GENERAL SIGNAL	CODE FLAG	Q	GENERAL SIGNAL		
A	OR H		E	OR Y			
O	X		H	Z			
12° Latitude	North Latitude		23° Longitude	East Longitude			
NUMERAL TABLE		GENERAL VOCABULARY		GEOGRAPHICAL SIGNALS ALPHABETICAL ORDER.			
CODE FLAG UNDER 2 FLAGS		3 FLAG SIGNAL		4 FLAG SIGNAL			
Y	I	X	A	E			
S	X	K	E				
CODE FLAG			Z				
10,000	Tons of Coal	Glasgow, Scotland.					
ALPHABETICAL SPELLING TABLE			NAMES OF VESSELS FROM CODE LIST.				
SPELLING SIGNAL			4 FLAG SIGNALS				
J	G	C	H				
O	B	S	C				
H	D	F	L				
N	N	P	B				
John	Abb	off	Glasgow of Glasgow				
			1058 Tons No 32696				

1, 2, 3 OR 4 FLAG SIGNALS

JAMES BROWN & SON GLASGOW.



This is what a process might look like



A process:

- Is provided with parsed/expanded options and arguments from the shell
- may read from **stdin**
- may write to **stdout**
- may write error messages to **stderr**
- and may get interrupted from time to time by a **signal**

*A **process** is a **program** that has been loaded into memory and is either running (executing instructions) or waiting to run*

Signals

The result of sending a signal to a process:

- be ignored
- default action (die)
- execute some predefined function



Signals

SIGHUP	1	Hangup (POSIX)	
SIGINT	2	Terminal interrupt (ANSI)	Ctrl-C
SIGQUIT	3	Terminal quit (POSIX)	Ctrl-\
SIGILL	4	Illegal instruction (ANSI)	
SIGTRAP	5	Trace trap (POSIX)	
SIGIOT	6	IOT Trap (4.2 BSD)	
SIGBUS	7	BUS error (4.2 BSD)	
SIGFPE	8	Floating point exception (ANSI)	
SIGKILL	9	Kill (can't be caught or ignored) (POSIX)	
SIGUSR1	10	User defined signal 1 (POSIX)	
SIGSEGV	11	Invalid memory segment access (ANSI)	
SIGUSR2	12	User defined signal 2 (POSIX)	
SIGPIPE	13	Write on a pipe with no reader, Broken pipe (POSIX)	
SIGALRM	14	Alarm clock (POSIX)	
SIGTERM	15	Termination (ANSI)	

Use kill -l to see all signals

Signals

SIGSTKFLT	16	Stack fault
SIGCHLD	17	Child process has stopped or exited, changed (POSIX)
SIGCONT	18	Continue executing, if stopped (POSIX)
SIGSTOP	19	Stop executing(can't be caught or ignored) (POSIX)
SIGTSTP	20	Terminal stop signal (POSIX) Ctrl-Z or Ctrl-F
SIGTTIN	21	Background process trying to read, from TTY (POSIX)
SIGTTOU	22	Background process trying to write, to TTY (POSIX)
SIGURG	23	Urgent condition on socket (4.2 BSD)
SIGXCPU	24	CPU limit exceeded (4.2 BSD)
SIGXFSZ	25	File size limit exceeded (4.2 BSD)
SIGVTALRM	26	Virtual alarm clock (4.2 BSD)
SIGPROF	27	Profiling alarm clock (4.2 BSD)
SIGWINCH	28	Window size change (4.3 BSD, Sun)
SIGIO	29	I/O now possible (4.2 BSD)
SIGPWR	30	Power failure restart (System V)

Use kill -l to see all signals

Signals



Signals are asynchronous messages sent to processes

They can result in one of three courses of action:

1. be ignored,
2. default action (die)
3. execute some predefined function.

Signals are sent:



Using the kill command: \$ **kill -# PID**

- Where # is the signal number and PID is the process id.
- if no number is specified, SIGTERM (-15) is sent.

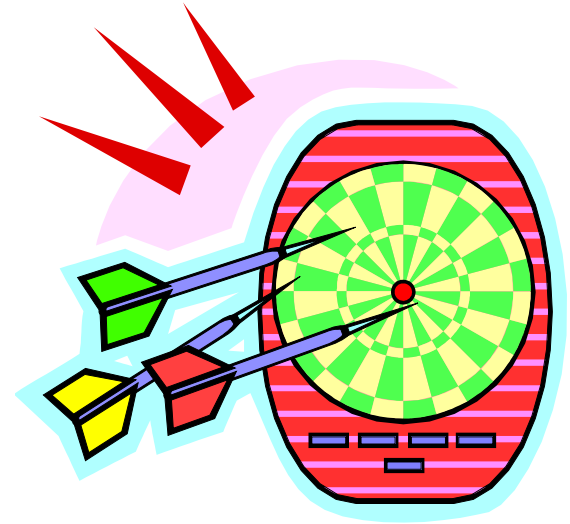


Using special keystrokes

- limited to just a few signals
- limited to when you have control of the keyboard

Use kill -l to see all signals

Target Practice



Activity

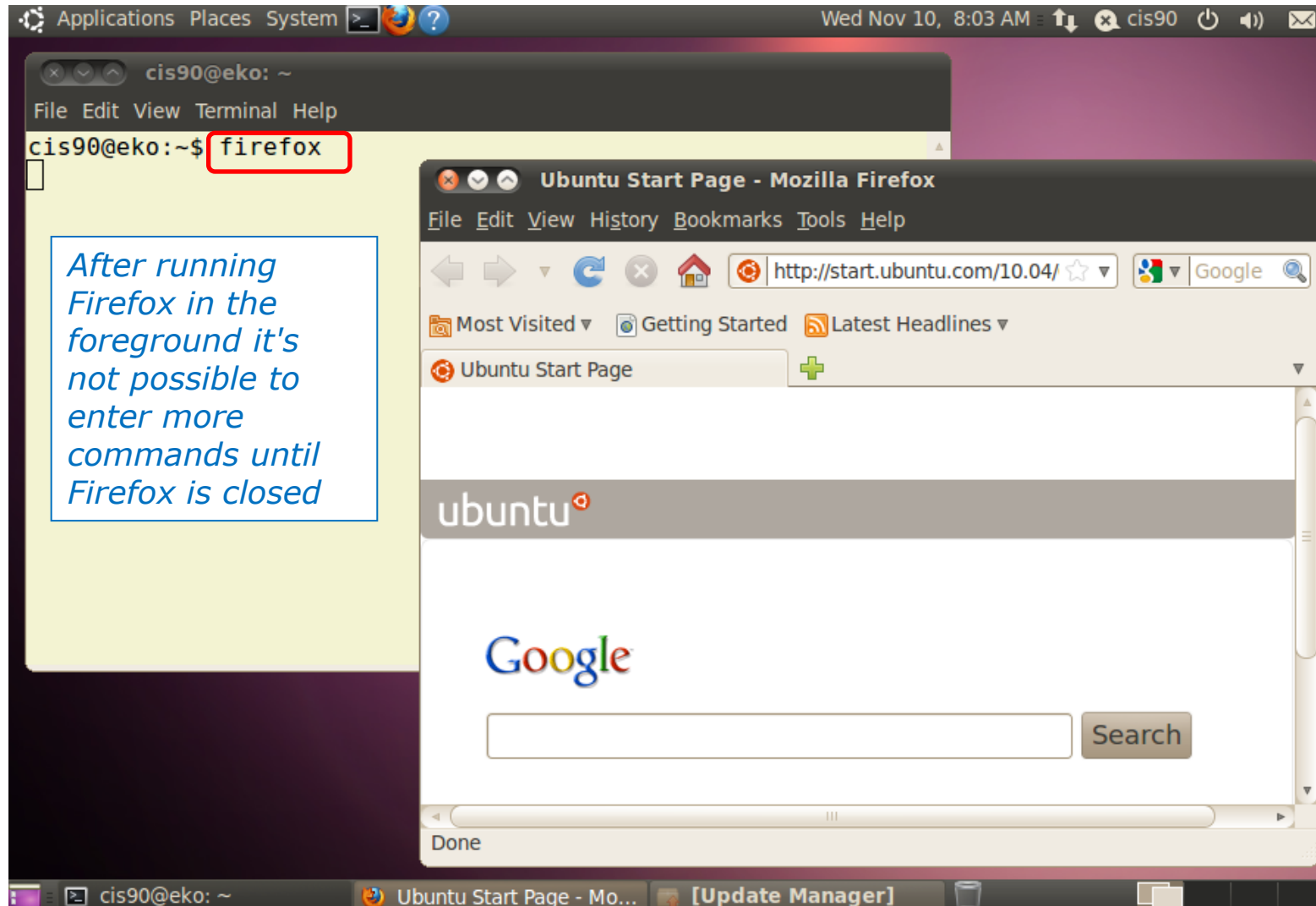
- 1) Run the **annoy** program
- 2) Try sending it a SIGINT with **Ctrl-C**
- 3) Try sending it a SIGQUIT with **Ctrl-**
- 4) Bring up another terminal and try signals 1 through 64
 - Use **ps -u \$LOGNAME** to find the **annoy** *PID*
 - Try **kill -1 *PID***
 - Try **kill -2 *PID***
 - Try **kill -3 *PID***
 - *and so forth ...*
 - OR
 - Try **killall -1 annoy**
 - Try **killall -2 annoy**
 - Try **killall -3 annoy**
 - *and so forth ...*
- 5) Write the signals that kill **annoy** into the chat window

Using &

to run a command
in the background

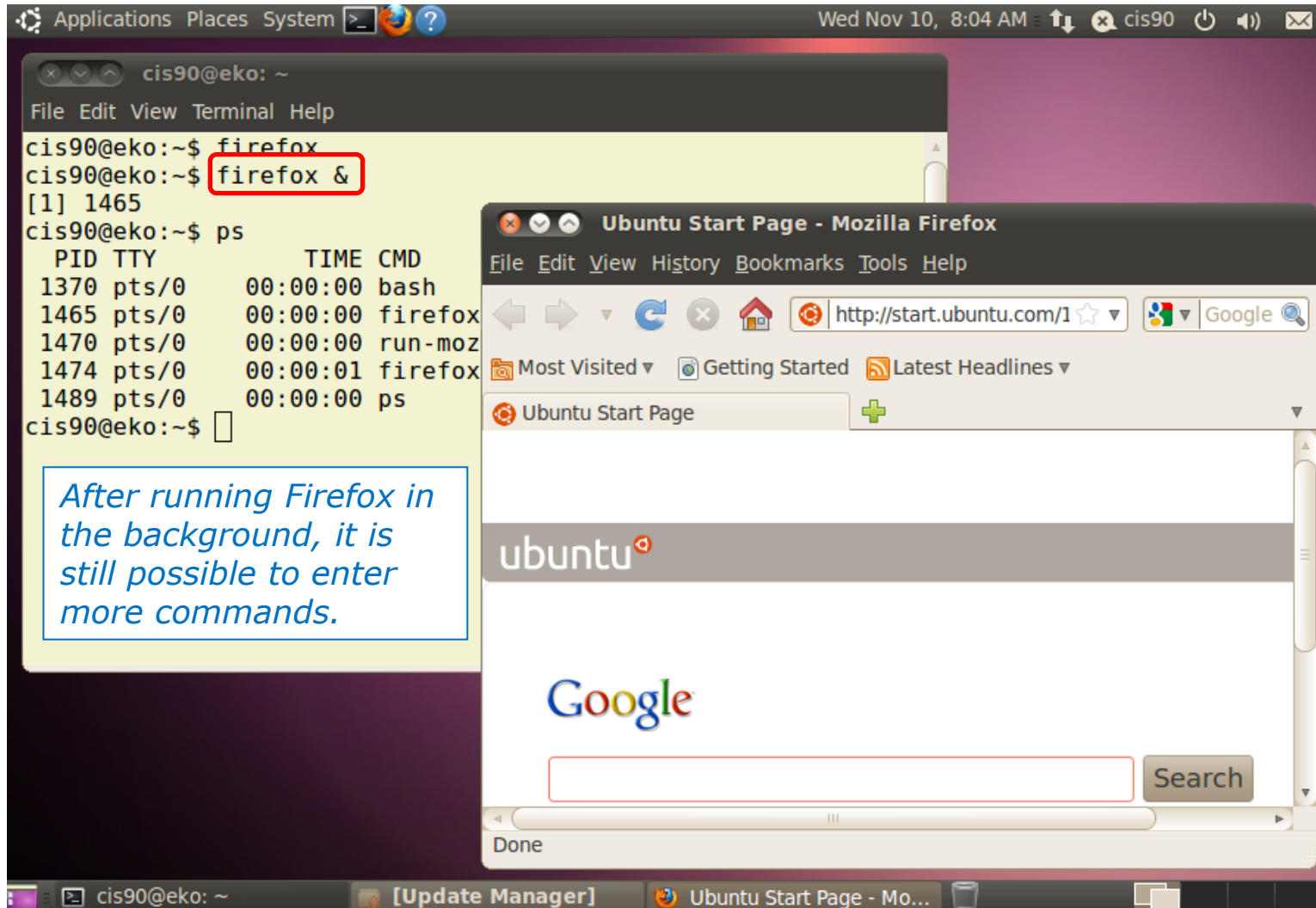
Job Control

Using **&** to run a command in the background



Job Control

Using **&** to run a command in the background



The screenshot shows a Linux terminal window with the following commands and output:

```
cis90@eko: ~
File Edit View Terminal Help
cis90@eko:~$ firefox
cis90@eko:~$ firefox &
[1] 1465
cis90@eko:~$ ps
  PID TTY          TIME CMD
 1370 pts/0    00:00:00 bash
 1465 pts/0    00:00:00 firefox
 1470 pts/0    00:00:00 run-moz
 1474 pts/0    00:00:01 firefox
 1489 pts/0    00:00:00 ps
cis90@eko:~$
```

After running Firefox in the background, it is still possible to enter more commands.

The Firefox window shows the Ubuntu Start Page with the URL <http://start.ubuntu.com/1> and a Google search bar.

& append to a command to run it in the background

Example 1

```
/home/cis90/simben $ grep -r potato /usr /opt 2> /dev/null
```

 **No prompt**

For long running commands or scripts you must wait for the command to finish before you type more commands

Example 2

```
/home/cis90/simben $ grep -r potato /usr /opt 2> /dev/null &
```

```
[1] 21175
```

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

```
Tue Apr 15 14:43:09 PDT 2014
```

Hit enter to get the prompt and continue working while the find command runs in the background

Job Control (Review)

Job Control

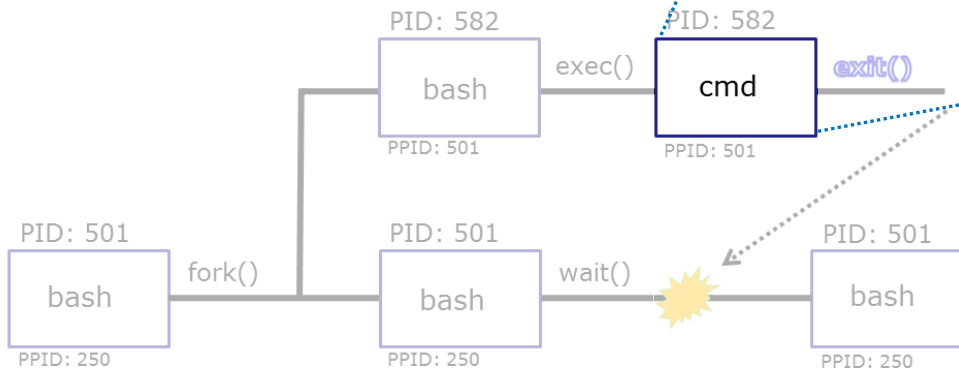
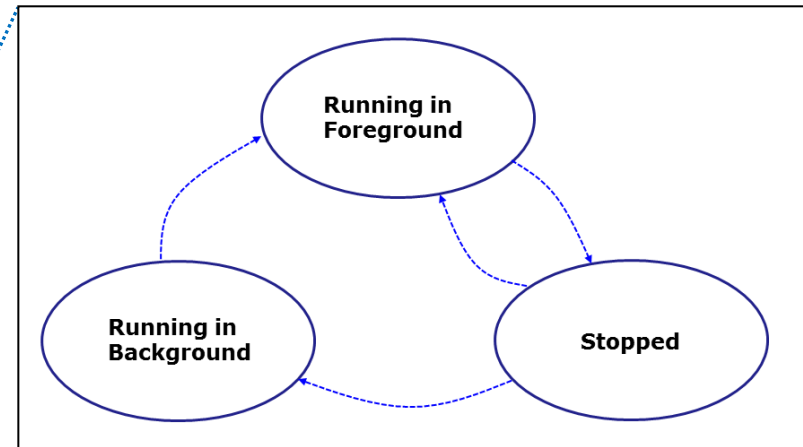
A feature of the bash shell

&	Append to a command to run it in the background
bg	Resumes a suspended job in the background
fg	Brings the most recent background process to the foreground
jobs	Lists all background jobs

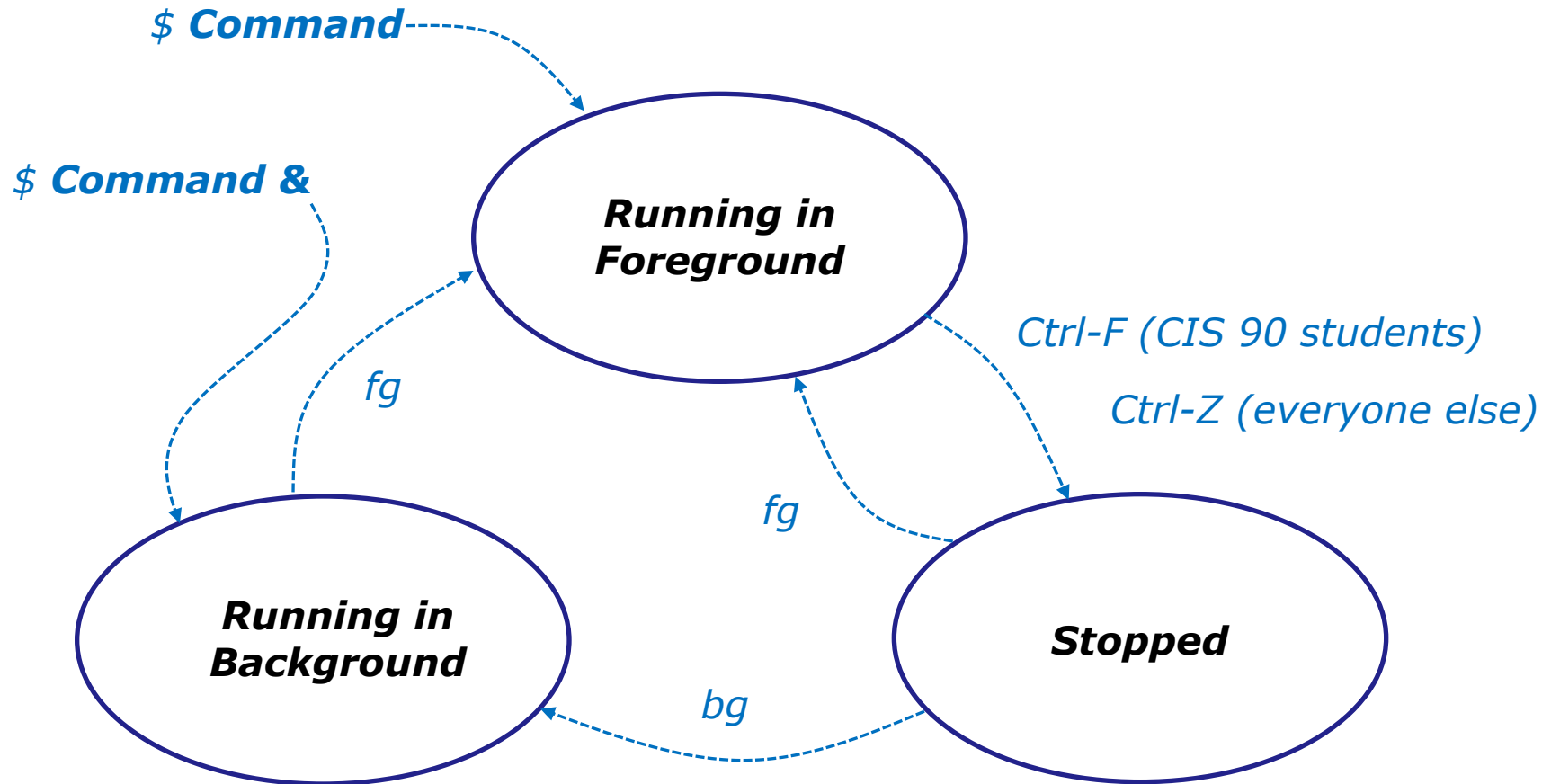
*Use **jobs**, **bg**, **fg** to list and resume jobs in the foreground or background*

Job Control A feature of the bash shell

When a process is **running** (status=R) the user can **stop** it (status=T) and choose whether it runs in the **background** or **foreground**



Job Control A feature of the bash shell



Use the **jobs** command to view
stopped and background jobs

Job Control

Find out with keystroke combination is configured to suspend a process

```
/home/cis90ol/simmsben $ stty -a
speed 38400 baud; rows 24; columns 80; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>;
eol2 = <undef>; swch = <undef>; start = ^Q; stop = ^S; susp = ^F; rprnt = ^R;
werase = ^W; lnext = ^V; flush = ^O; min = 1; time = 0;
-parenb -parodd cs8 -hupcl -cstopb cread -clocal -crtscts -cdtrdsr
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon -ixoff
-iuclc -ixany -imaxbel -iutf8
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt
echoctl echoke
/home/cis90ol/simmsben $
```

In this case it is Ctrl-F that will be used to suspend a process

How is yours configured?

Job Control

Managing jobs

```
/home/cis90ol/simmsben $ sleep 120
Ctrl-Z or Ctrl-F (to suspend process)
[1]+  Stopped                  sleep 120
```

```
/home/cis90ol/simmsben $ sleep 110
Ctrl-Z or Ctrl-F (to suspend process)
[2]+  Stopped                  sleep 110
```

```
/home/cis90ol/simmsben $ sleep 100
Ctrl-Z or Ctrl-F (to suspend process)
[3]+  Stopped                  sleep 100
```

```
/home/cis90ol/simmsben $ jobs
[1]  Stopped                  sleep 120
[2]-  Stopped                  sleep 110
[3]+  Stopped                  sleep 100
```

Lets start up 3 sleep commands and suspend each of them.

Note: The sleep command is a simple way to run a command that will take awhile to finish.

***sleep 120** will last 120 seconds before it is finished.*

Job Control

Managing jobs

```
/home/cis90ol/simmsben $ jobs
```

```
[1]      Stopped                sleep 120
[2]-     Stopped                sleep 110
[3]+     Stopped                sleep 100
```

```
/home/cis90ol/simmsben $ ps -l
```

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
0	S	1082	5364	5363	0	75	0	-	1168	wait	pts/2	00:00:00	bash
0	T	1082	5452	5364	0	75	0	-	929	finish	pts/2	00:00:00	sleep
0	T	1082	5453	5364	0	75	0	-	929	finish	pts/2	00:00:00	sleep
0	T	1082	5454	5364	0	75	0	-	929	finish	pts/2	00:00:00	sleep
0	R	1082	5459	5364	0	77	0	-	1054	-	pts/2	00:00:00	ps

Note, all three processes are sTopped

Job Control

Managing jobs

```
/home/cis90ol/simmsben $ bg 2 Let's resume job 2 in the background
```

```
[2]- sleep 110 &
```

```
/home/cis90ol/simmsben $ jobs
```

```
[1]- Stopped sleep 120
```

```
[2] Running sleep 110 &
```

```
[3]+ Stopped sleep 100
```

```
/home/cis90ol/simmsben $ bg 1 Let's resume job 1 in the background
```

```
[1]- sleep 120 &
```

```
/home/cis90ol/simmsben $ jobs
```

```
[1] Running sleep 120 &
```

```
[2]- Running sleep 110 &
```

```
[3]+ Stopped sleep 100
```

```
/home/cis90ol/simmsben $ fg 3 Let's resume job 1 in the foreground
```

```
sleep 100
```

*At this point we lose control of the keyboard again
until sleep 100 is finished*

Job Control

Managing jobs

```
/home/cis90ol/simmsben $ jobs  
[1]-  Done  
sleep 120  
[2]+  Done  
sleep 110
```

*Background jobs are
all done!*

Review of Load Balancing & Scheduling

Load Balancing

The **at** command:

- reads from stdin for a list of commands to run
- runs those commands at the specified time
- Any output from those commands will be emailed
- Use **atq** and **atrm** to manage scheduled commands

*Use **at** to schedule commands to run in the future*

Load Balancing

Managing queued jobs

`at now + 5 minutes`

`at now + 1 hour`

`at 7:58AM`

`at 7:47PM 11/25/2014`

`at teatime`

Ways to specify future times

Load Balancing

Managing queued jobs

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

```
25      2011-11-12 14:09 a simben90
28      2011-12-12 03:00 a simben90
27      2011-11-19 12:10 a simben90
26      2011-11-12 16:00 a simben90
24      2011-11-12 12:14 a simben90
```

*The **atq** command lists jobs queued to run in the future*

```
/home/cis90/simben $ atrm 24
```

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

```
25      2011-11-12 14:09 a simben90
28      2011-12-12 03:00 a simben90
27      2011-11-19 12:10 a simben90
26      2011-11-12 16:00 a simben90
```

*The **atrm** command is used to remove jobs from the queue*

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

*Note: The **jobs** command lists processes running or suspended in the background and is NOT used for **at** commands.*

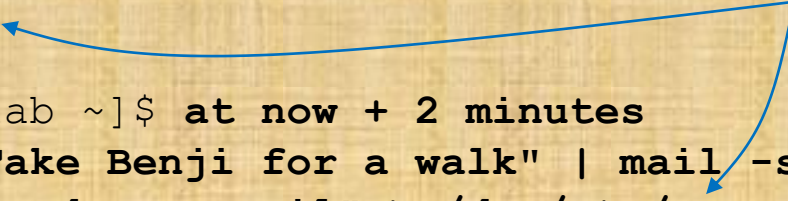
Load Balancing

Try it yourself with your own terminal device and username:

```
[rsimms@oslab ~]$ tty
/dev/pts/x
```

These should match

```
[rsimms@oslab ~]$ at now + 2 minutes
at> echo "Take Benji for a walk" | mail -s "walk the dog" $LOGNAME
at> echo "Read your mail" > /dev/pts/x
at> <EOT>
job 11 at 2012-11-05 11:02
[rsimms@oslab ~]$ atq
11      2012-11-05 11:02 a rsimms
[rsimms@oslab ~]$
```



Type what happens in the chat window:

text editors



There are lots of text editors ...

Windows

notepad
notepad++
textpad

Text editors and word processors are different!

Mac

TextWrangler

- *Word processors are used by many different people to create documents containing text and graphics.*

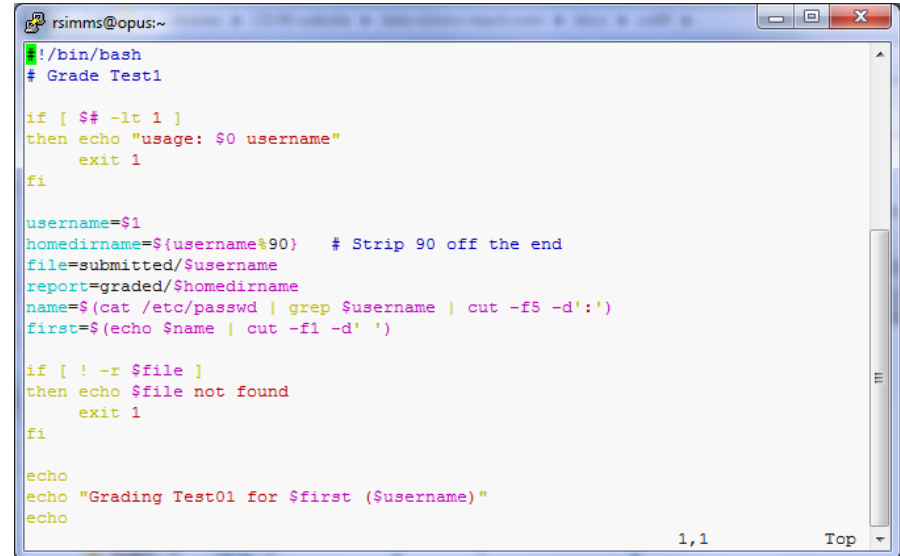
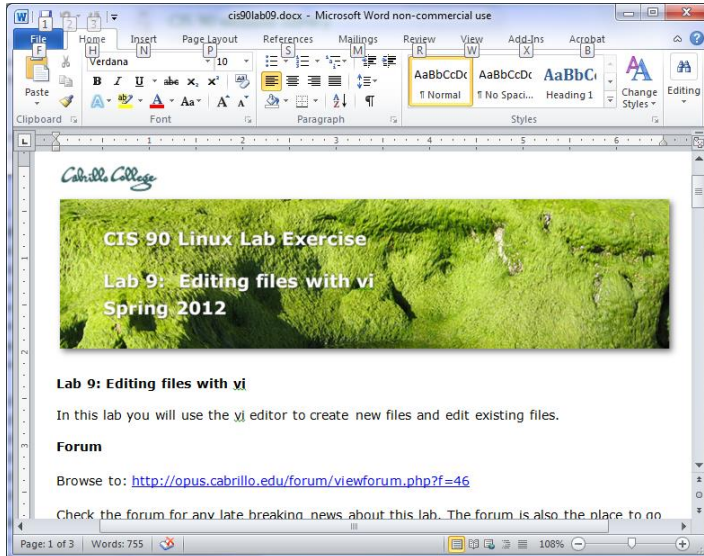
Linux

gedit
emacs
nano
vi
jove

- *Text editors are used by programmers to develop software and web designers to create web sites.*



Thanks Maria!



Word processors allow a rich set of formatting (fonts, sizes, styles, color) and graphics to be added to documents.

Text editors use color to show the language syntax

vi 101

On Opus we are actually running VIM

```
/home/cis90/simben $ type -a vi  
vi is aliased to `vim'  
vi is /bin/vi  
/home/cis90/simben $ type vim  
vim is hashed (/usr/bin/vim)
```

History:

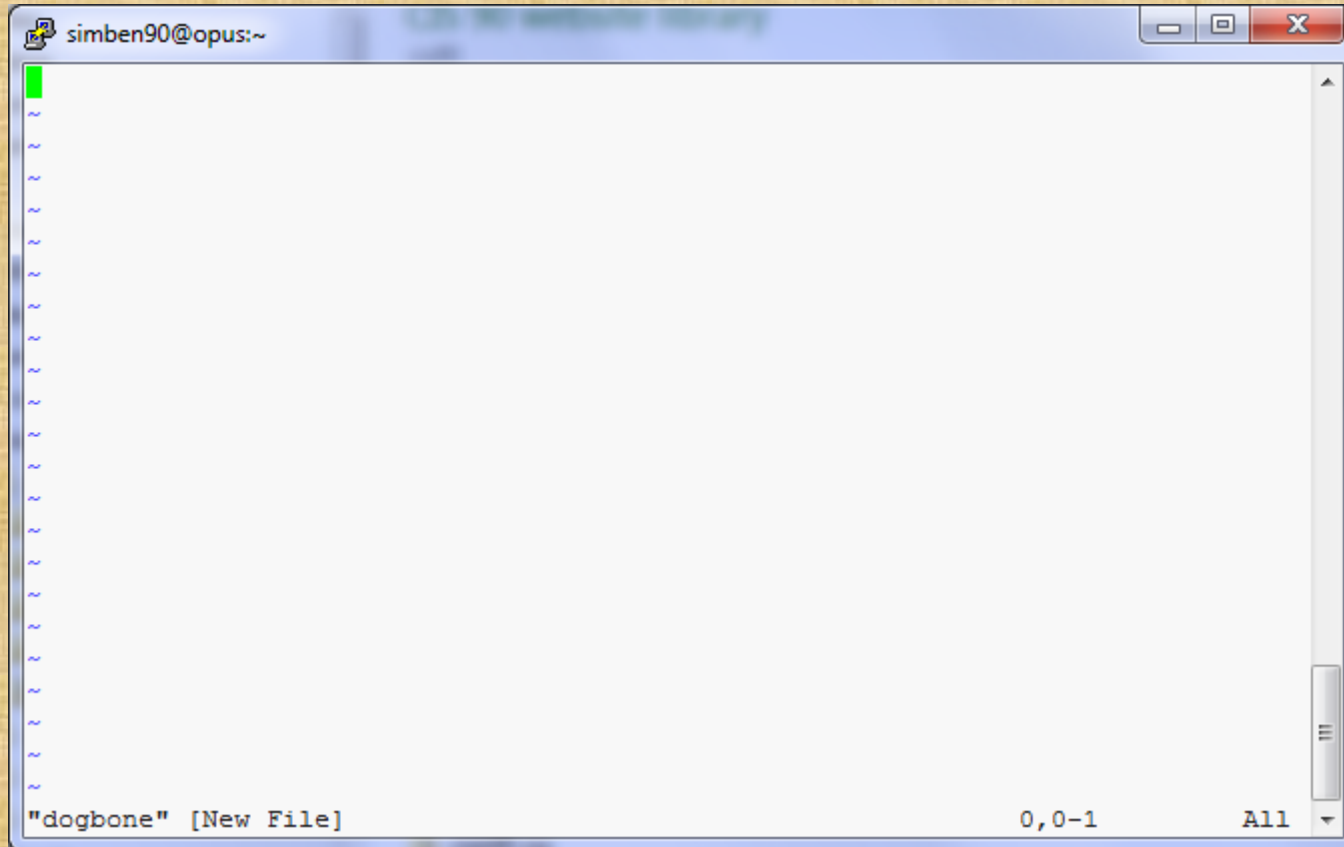
- The original vi code was written by Bill Joy for BSD Unix
- Bill Joy co-founded Sun Microsystems in 1982
- vi (for "visual")
- vim is an enhanced version of vi

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

```
/home/cis90/simben $ vi dogbone
```

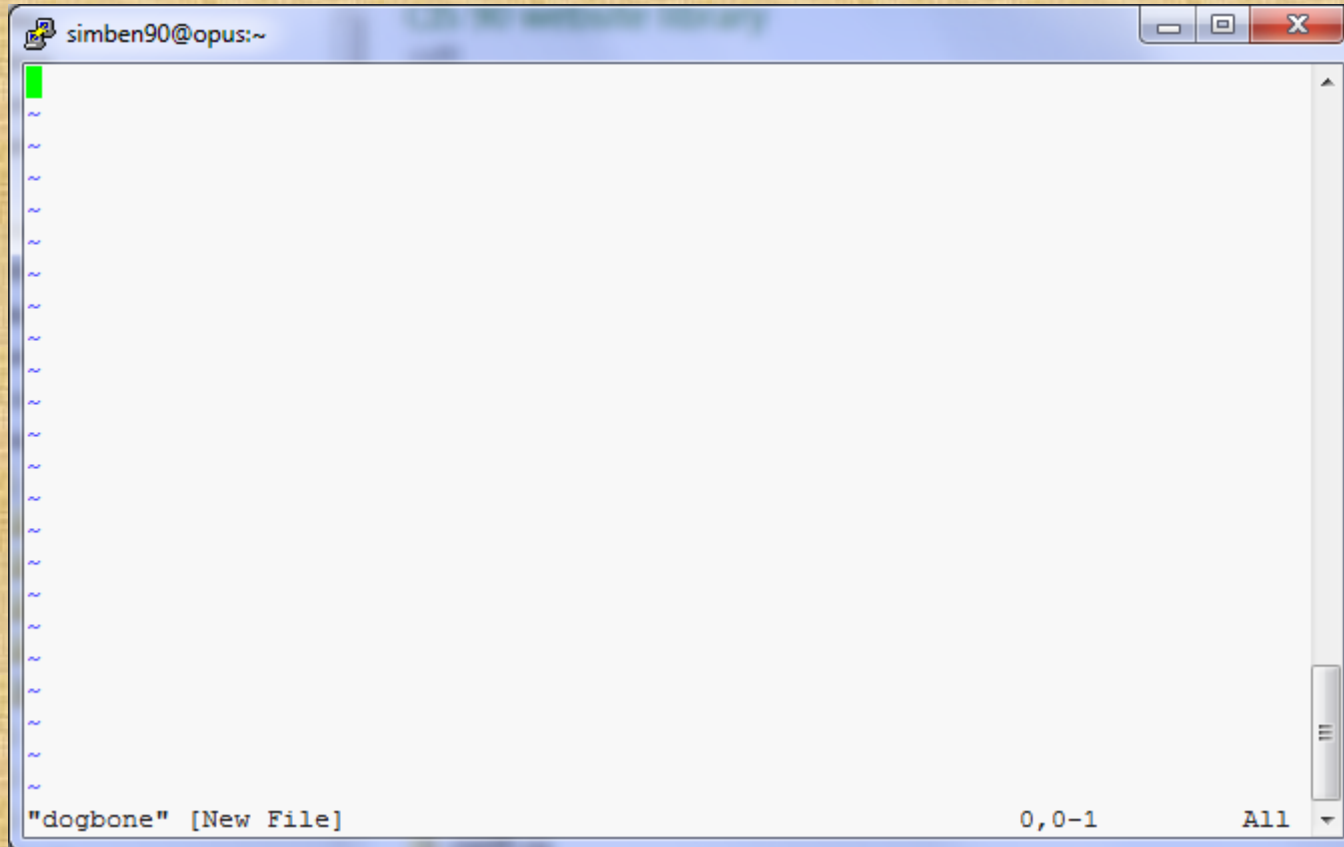
Type this

See this ...



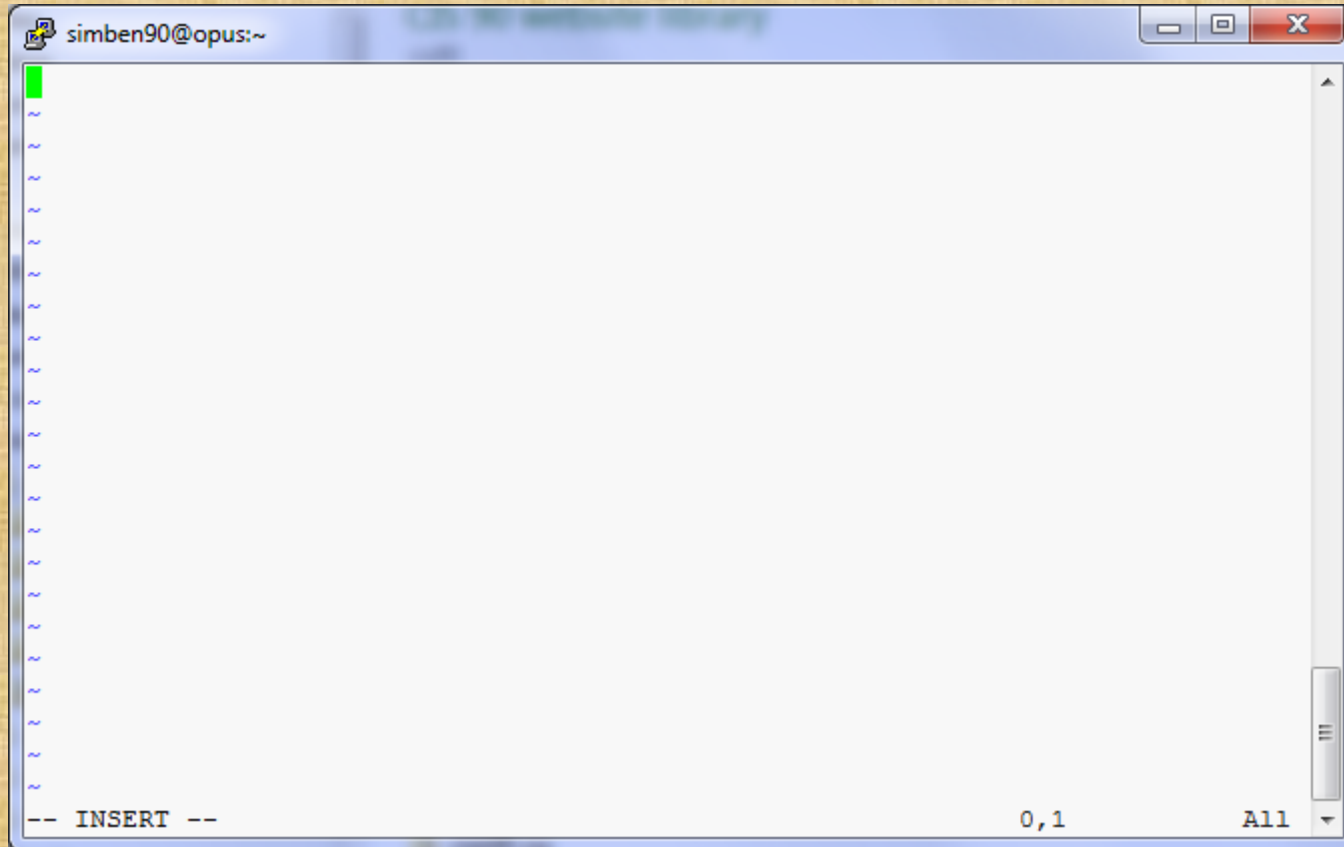
Take your hands OFF THE MOUSE – don't use it in vi!

*Tap the letter **i** key (for insert)*



Keep your hands OFF THE MOUSE – don't use it in vi!

See this ...



Keep your hands OFF THE MOUSE – don't use it in vi!

A screenshot of a Linux terminal window titled "simben90@opus:~". The terminal displays several commands being executed:

```
echo -n "What is your name? "  
read NAME  
echo -n "What is your favorite bone? "  
read BONE  
echo "Hi $NAME, your favorite bone is $BONE"
```

A green cursor is visible at the end of the last command. Below the main code block, there are several tilde (~) symbols representing additional lines of code or output. At the bottom left, it says "-- INSERT --". At the bottom right, there is a status bar showing "6,1" and "All".

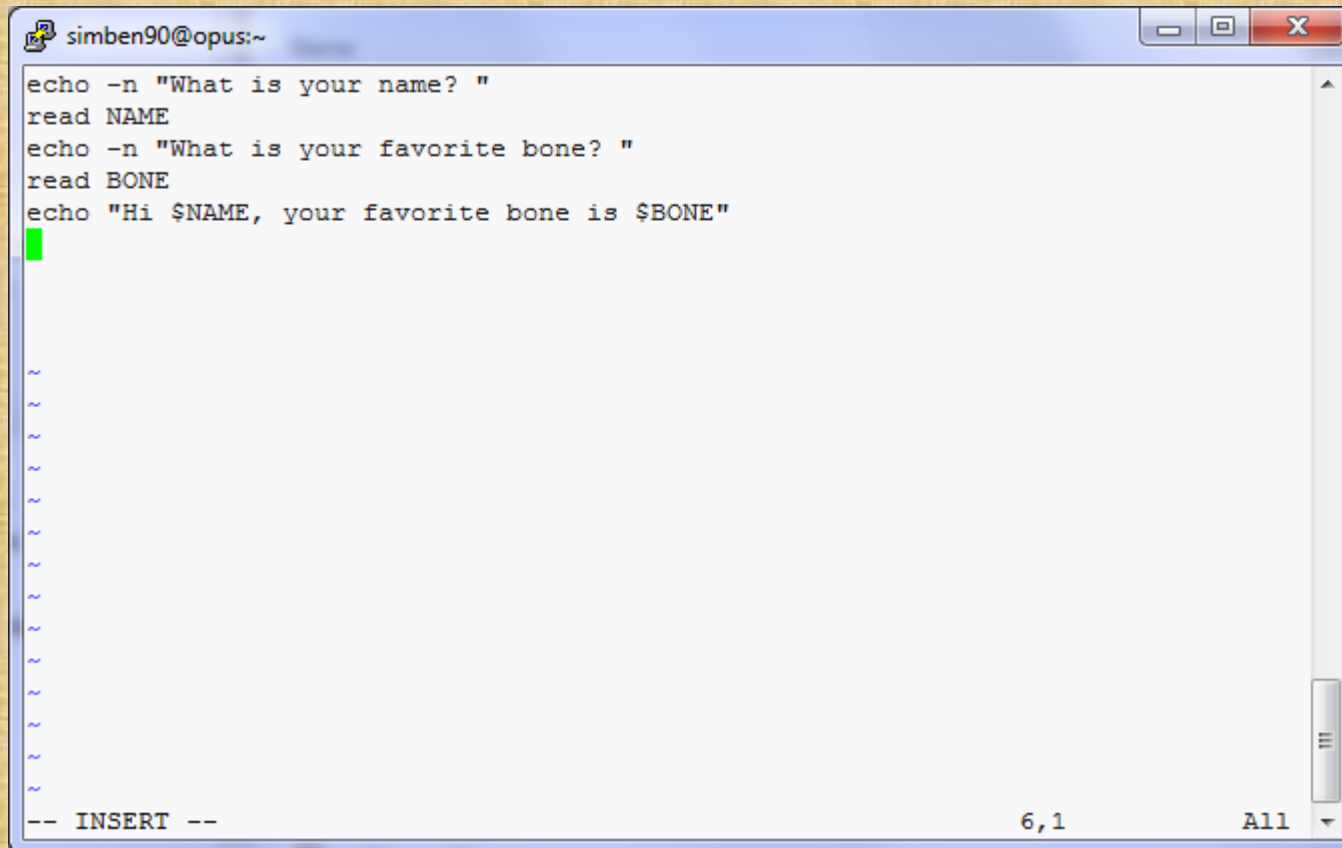
`simben90@opus:~`

```
echo -n "What is your name? "  
read NAME  
echo -n "What is your favorite bone? "  
read BONE  
echo "Hi $NAME, your favorite bone is $BONE"  
  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
-- INSERT --
```

6,1 All

Keep your hands OFF THE MOUSE – don't use it in vi!

Have your neighbor check that your five lines are PERFECT



The screenshot shows a terminal window titled 'simben90@opus:~'. Inside the terminal, a shell script is being edited using the vi editor. The script contains the following lines:

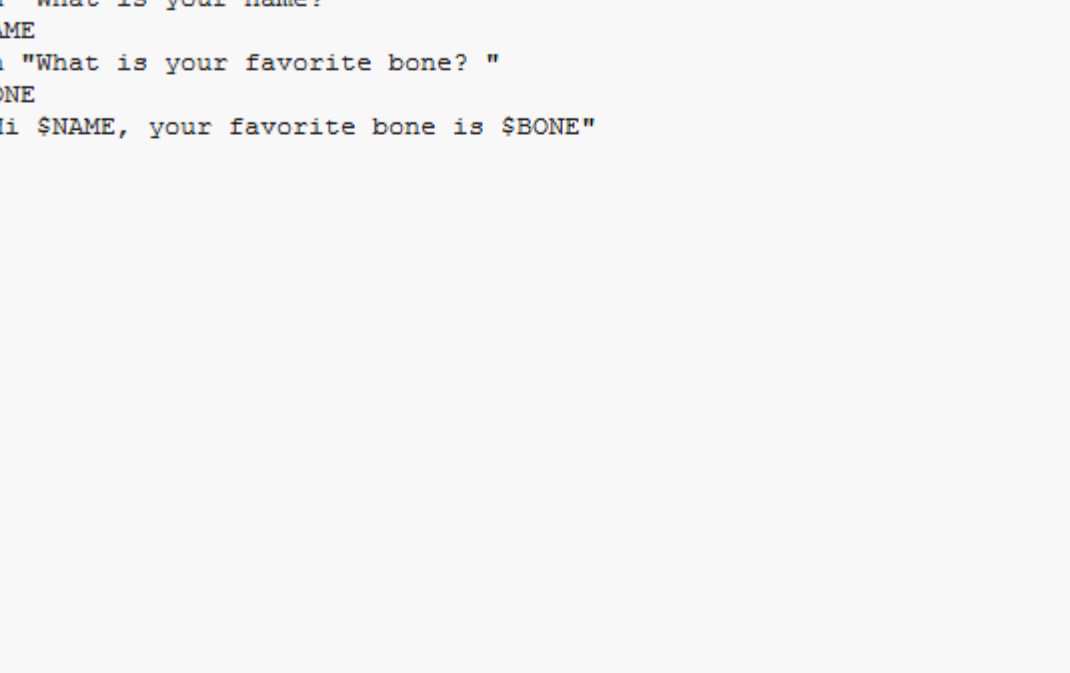
```
echo -n "What is your name? "  
read NAME  
echo -n "What is your favorite bone? "  
read BONE  
echo "Hi $NAME, your favorite bone is $BONE"
```

The cursor is positioned at the end of the fifth line. Below the script, there are several tilde (~) characters, likely representing a scroll buffer or a list of lines. At the bottom of the terminal window, the status bar shows '-- INSERT --' on the left, '6,1' in the center, and 'All' on the right.

Keep your hands OFF THE MOUSE – don't use it in vi!

[illegible]

111



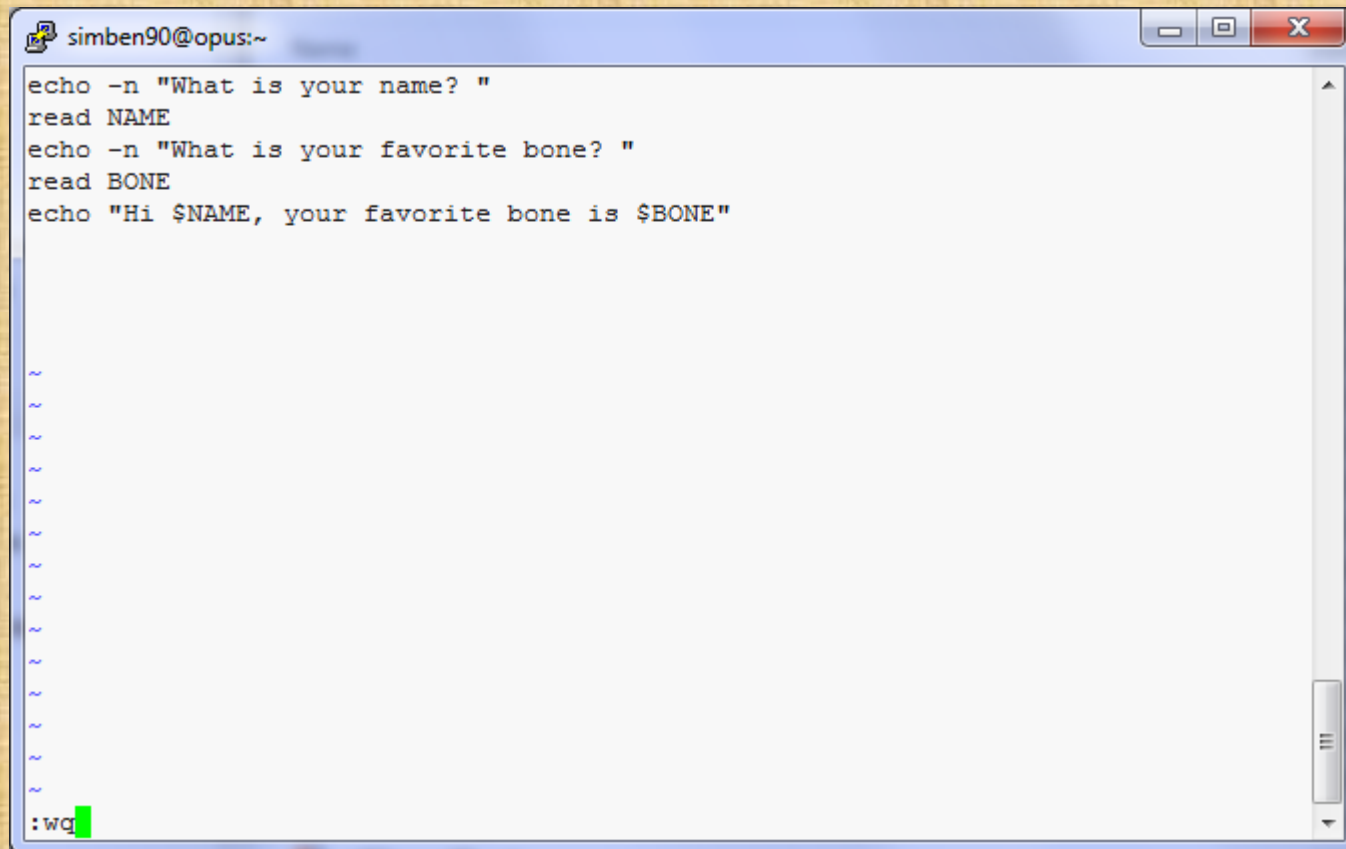
The screenshot shows a terminal window with a blue title bar. The title bar text is "simben90@opus:~". The terminal content is as follows:

```
echo -n "What is your name? "  
read NAME  
echo -n "What is your favorite bone? "  
read BONE  
echo "Hi $NAME, your favorite bone is $BONE"
```

Below the script, there are 15 tilde (~) characters, one on each line. At the bottom of the terminal, there is a prompt character ":" followed by a green cursor block.

Keep your hands OFF THE MOUSE – don't use it in vi!

Type **wq**



A terminal window titled 'simben90@opus:~' showing a script being edited in the vi editor. The script contains the following lines:

```
echo -n "What is your name? "  
read NAME  
echo -n "What is your favorite bone? "  
read BONE  
echo "Hi $NAME, your favorite bone is $BONE"
```

The terminal shows several tilde (~) characters representing input, and a green cursor is visible at the end of the line ':wq'.

Keep your hands OFF THE MOUSE – don't use it in vi!



Tap the enter key

```
/home/cis90/simben $ vi dogbone  
/home/cis90/simben $
```



Add execute permissions and try your new script

```
/home/cis90/simben $ chmod +x dogbone  
  
/home/cis90/simben $ dogbone  
What is your name? Benji  
What is your favorite bone? chicken  
Hi Benji, your favorite bone is chicken  
/home/cis90/simben $
```

vi

COMMAND mode
INSERT mode
command LINE mode


```
/home/cis90/simben $ cp letter myletter
/home/cis90/simben $ vi myletter
```

COMMAND mode

```
simben90@opus:~
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.

All the counselors hate the waiters, and the lake has
alligators. You remember Leonard Skinner? He got
ptomaine poisoning last night after dinner.

Now I don't want this to scare you, but my bunk mate has
malaria. You remember Jeffrey Hardy? Their about to
organize a searching party.

Take me home, oh Mother, Father, take me home! I hate Granada.
Don't leave me out in the forest where I might get eaten
by a bear! Take me home, I promise that I won't make noise,
or mess the house with other boys, oh please don't make me
stay -- I've been here one whole day.

Dearest Father, darling Mother, how's my precious little
brother? I will come home if you miss me. I will even
let Aunt Bertha hug and kiss me!

"myletter" 29L, 1059C      1,1      Top
```

INSERT mode

```
simben90@opus:~
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.

All the counselors hate the waiters, and the lake has
alligators. You remember Leonard Skinner? He got
ptomaine poisoning last night after dinner.

Now I don't want this to scare you, but my bunk mate has
malaria. You remember Jeffrey Hardy? Their about to
organize a searching party.

Take me home, oh Mother, Father, take me home! I hate Granada.
Don't leave me out in the forest where I might get eaten
by a bear! Take me home, I promise that I won't make noise,
or mess the house with other boys, oh please don't make me
stay -- I've been here one whole day.

Dearest Father, darling Mother, how's my precious little
brother? I will come home if you miss me. I will even
let Aunt Bertha hug and kiss me!

-- INSERT --      1,1      Top
```

Command LINE mode

```
simben90@opus:~
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.

All the counselors hate the waiters, and the lake has
alligators. You remember Leonard Skinner? He got
ptomaine poisoning last night after dinner.

Now I don't want this to scare you, but my bunk mate has
malaria. You remember Jeffrey Hardy? Their about to
organize a searching party.

Take me home, oh Mother, Father, take me home! I hate Granada.
Don't leave me out in the forest where I might get eaten
by a bear! Take me home, I promise that I won't make noise,
or mess the house with other boys, oh please don't make me
stay -- I've been here one whole day.

Dearest Father, darling Mother, how's my precious little
brother? I will come home if you miss me. I will even
let Aunt Bertha hug and kiss me!

:      1,1      Top
```


vi

Moving around in a file

Use in COMMAND mode

h moves the cursor one character to the left
j moves the cursor down one line
k moves the cursor up one line
l moves the cursor one character to the right

^d scrolls down 10 lines
^u scrolls up 10 lines
^f page forward one page
^b page back one page

*Try typing a
number in front of
these commands
and notice what
happens*

*With vim (not vi) you can use arrow and
page keys instead of these letter commands*

vi

Moving around in a file

Use in COMMAND mode

w moves the cursor one "word" forward

b moves the cursor one "word" back

*Try typing a number in front
of these commands and
notice what happens*

0 (zero) moves the cursor to the beginning of the line

\$ moves the cursor to the end of the line

G moves the cursor to the last line in the file

1G moves the cursor to the first line in the file

105G moves the cursor to line 105

vi

Saving and Quitting

Use in command LINE mode

:w writes any changes to the file you are editing (like Save)

:q quits vi if you have saved your changes

:q! quits vi even if you haven't saved changes

:wq writes and quits

:wq! writes and quits vi even if you haven't saved changes

vi

Reading in and Writing out files

Use in command LINE mode

:w filename saves your file to a new name (like Save As)

:w! filename saves your file to a new name overwriting any previous data

:r filename reads in the contents of *filename* starting from the cursor position

:e filename replaces the current content with the content from *filename*

:%s /string1/string2/g replaces all string1 with string2 in the file

vi

Entering INSERT mode

From COMMAND mode.

- i** Ready to insert characters immediately before the current cursor position
- I** Ready to insert characters at the start of the current line

- a** Ready to append characters immediately after the current cursor position
- A** Ready to append characters at the end of the current line

- o** Ready to input characters in a new line that opens up below the cursor
- O** Ready to input characters in a new line that opens up above the cursor

vi

Cut, Copy, Pasting Commands

Use in COMMAND mode

x Deletes the current character

r Replace the current character with the character you type next

dw Deletes the current word

dd Deletes the current line

D Deletes to the end of the line

yy Copies a line to the clipboard buffer

p Pastes whatever is in the clipboard buffer below the current cursor

P Pastes whatever is in the clipboard buffer above the current cursor

vi

Miscellaneous Useful Commands

Use in COMMAND mode.

^g Tells you the filename you are editing and what line your cursor is on

u Undoes the last command you executed

^r Undo the undo (redo)

. Repeats the last command you executed

/string Searches for the string of characters in the file

n Finds the next occurrence of the current search string looking down the file

N Finds the next occurrence of the current search string looking up the file

~ Changes the case of the current character

Use vi to edit your *edits/text.err* file

```
This is line number1.  
This is line number 1.  
Thi sis line line number 2.  
his is line number3.line number3.  
This is This is line #4.  
this number5 is line .  
Here is line number      6.  
This is lamw number      7.  
Thi is line nunber9.  
This is line  
number10.
```



```
This is line number 1.  
This is line number 2.  
This is line number 3.  
This is line number 4.  
This is line number 5.  
This is line number 6.  
This is line number 7.  
This is line number 8.  
This is line number 9.  
This is line number 10.
```

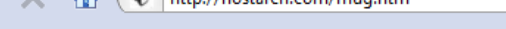
Copy your corrected file into the chat window when finished

http://vim.wikia.com/wiki/Main_Page



Tips and tricks for VIM users

The Mug of vi



File Edit View History Bookmarks Tools Help

http://nostarch.com/mug.htm


Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options

Cabrillo College Home Page (0 unread) Yahoo! Mail, richsimms The Mug of Vi Sams Publishing - Contact Us

NO STARCH PRESS "the finest in geek entertainment"™

Home | Catalog | Where to buy | About | Jobs | Media | Blog | Cart

Google Custom Search Search



The Mug of Vi
12 ounce heavy-duty ceramic

\$12.95

Order now

Hydration harmony

Copyright © 2000 No Starch Press

[See mug text](#)

Click on the image to return to **Mug of Vi** main page.

FILE COMMANDS	DELETING / INSERTING TEXT	CUT / COPY / PASTE
vi filename(s) edit a file or files	dw, dd, x delete word, line, character	0 go to beginning of line (zero)
vi -r filename retrieve saved file after crash	n dd, nx delete <i>n</i> lines, <i>n</i> characters), (move to next, previous sentence
ZZ, :wq, :X save and exit	x, X delete character forward, backward	}, { move to next, previous paragraph
:q, :q! quit; quit without saving	D, ds delete to end of line	w, b move forward, back one word
:w, :w fu save file, save file as <i>fu</i>	dmotion delete from cursor to <i>motion</i> (\$, 0, etc.)	e go to end of current or next word
:e filename edit <i>filename</i>	>, < indent, outdent line	nyy, nY copy <i>n</i> lines
:x filename insert <i>filename</i>	S replace text with blank line	yW, yY copy word, line
:sh drop to shell	o, O insert new line below, above current line	p, P paste text after, before cursor
:!cmd run command <i>cmd</i>	u undo last change	a, i insert text after, before cursor
:r !cmd execute <i>cmd</i> and insert output	. repeat last change	A, I insert text end, beginning of line
/txt, ?txt find <i>txt</i> forward or backward	MOVING AROUND	WICKED COOL STUFF
!^txt find next line that starts with <i>txt</i>	h, l, k, j left, right, up, down one character	~ change case
n, N repeat last search backward, forward	vb, vb left or right <i>n</i> words	xp transpose characters
R replace text from current character	CTRL-B, F back, forward one screen	J combine current line with next
	CTRL-U, D up, down one screen	mp create a mark called <i>p</i>
	\$, G go to end of line, end of file	p return to <i>p</i>
		d x, y' x delete, copy text from mark to cursor
		:> n indent <i>n</i> lines

http://nostarch.com/mug.htm

/bin/mail and vi

```
/home/cis90/simben $ mail milhom90
```

```
Subject: Good Bones
```

```
Hey Homer,
```

```
I really appreciate thatbone you sent me last week.
```

```
Let me knwo if you want to go mark some fench posts  
this weekend.
```

```
Later,
```

```
Ben
```

*You are composing a message and you spot some typos ...
CRUD ... what can you do?*

/bin/mail and vi

```
/home/cis90/simben $ mail milhom90
```

```
Subject: Good Bones
```

```
Hey Homer,
```

```
I really appreciate thatbone you sent me last week.
```

```
Let me knwo if you want to go mark some fench posts  
this weekend.
```

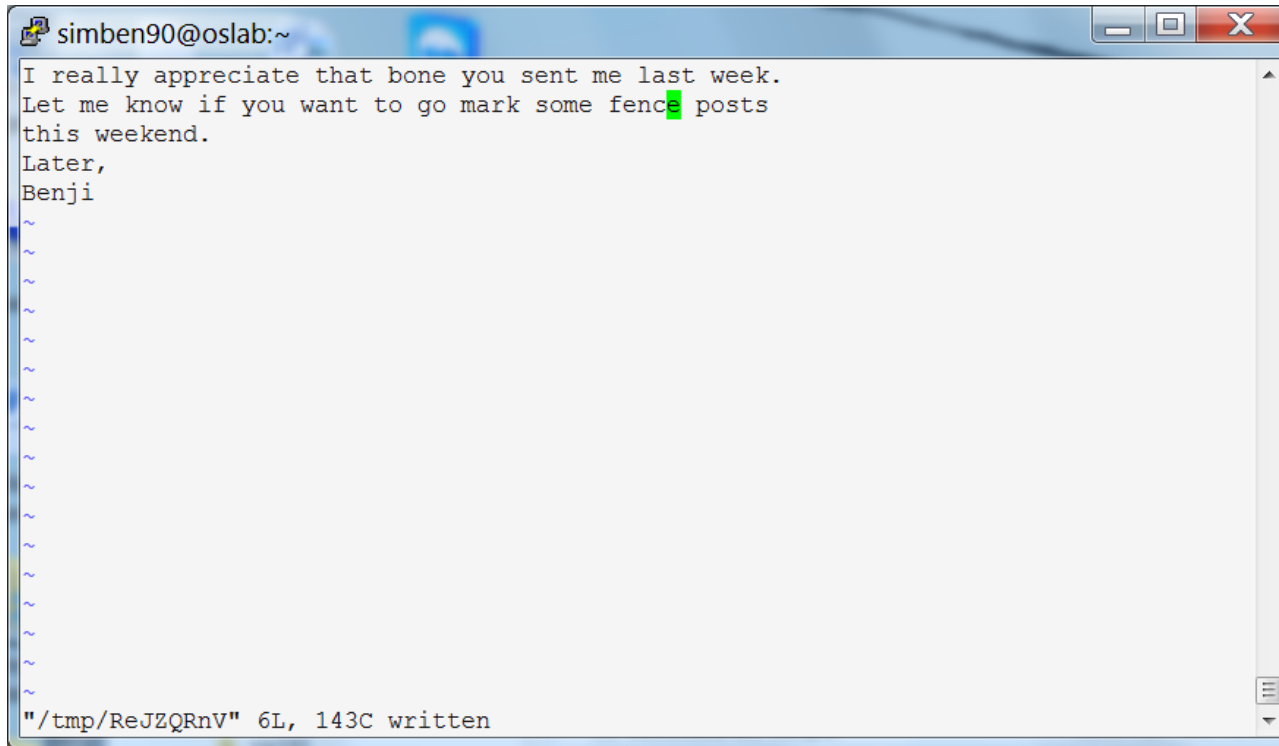
```
Later,
```

```
Ben
```

```
~v
```

Well ... you could try the ~v command

/bin/mail and vi



The screenshot shows a terminal window titled "simben90@oslab:~". Inside the terminal, an email message is being edited using the vi editor. The message content is as follows:

```
I really appreciate that bone you sent me last week.  
Let me know if you want to go mark some fence posts  
this weekend.  
Later,  
Benji
```

Below the message, there are several tilde (~) characters representing the body of the email. At the bottom of the terminal window, a status line indicates: `"/tmp/ReJZQRnV" 6L, 143C written`.

The message is loaded into vi where changes or additions can be made. :wq is used to save and quit vi

/bin/mail and vi

```
/home/cis90/simben $ mail milhom90
Subject: Good Bones
Hey Homer,
I really appreciate thatbone you sent me last week.
Let me knwo if you want to go mark some fench posts
this weekend.
Later,
Ben
~v
(continue)
.
EOT
/home/cis90/simben $
```

The earlier text with typos is still showing, however the corrected version is what is actually sent.

/bin/mail and vi

```
/home/cis90/milhom $ mail
Heirloom Mail version 12.4 7/29/08.  Type ? for help.
"/var/spool/mail/milhom90": 157 messages 5 new 155 unread
>N157 Benji Simms          Mon Nov 10 14:05  25/952  "Good Bones"
& 157
Message 157:
From simben90@oslab.cis.cabrillo.edu  Mon Nov 10 14:05:20 2014
Return-Path: <simben90@oslab.cis.cabrillo.edu>
From: Benji Simms <simben90@oslab.cis.cabrillo.edu>
Date: Mon, 10 Nov 2014 14:05:20 -0800
To: milhom90@oslab.cis.cabrillo.edu
Subject: Good Bones
User-Agent: Heirloom mailx 12.4 7/29/08
Content-Type: text/plain; charset=us-ascii
Status: R
```

Hey Homer,
I really appreciate that bone you sent me last week.
Let me know if you want to go mark some fence posts
this weekend.
Later,
Benji

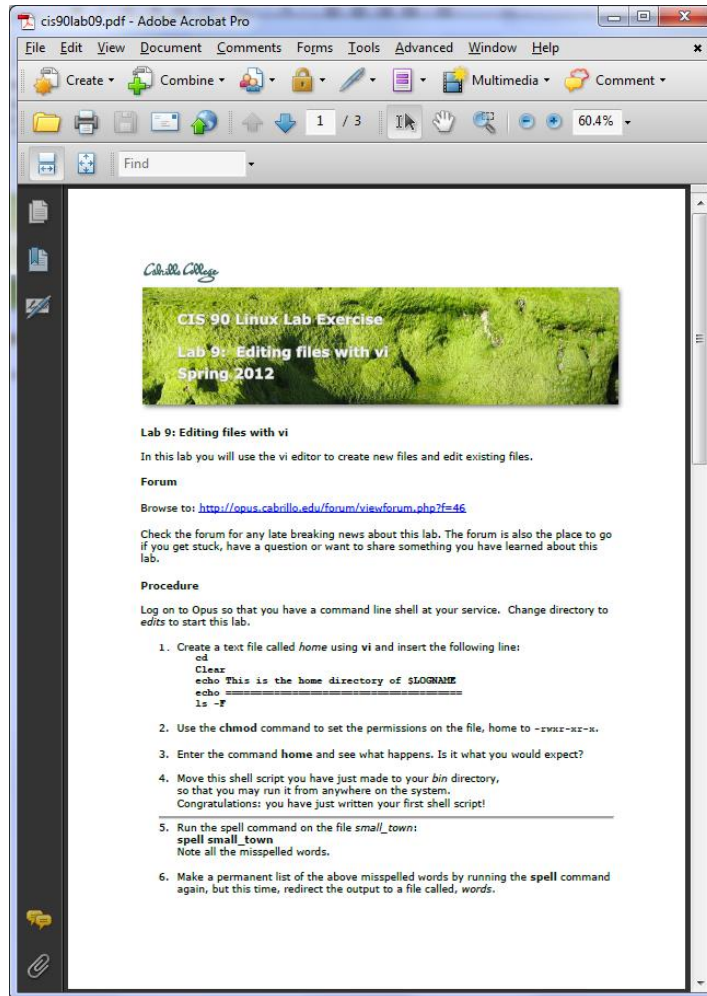
*The message Homer reads has all the
typos fixed.*

Fix an email message before sending

```
/home/cis90/simben/edits $ mail rsimms
Subject: test of vi
sdkfjas;dfllkjas;lkdfj
~v
(continue)
.
EOT
/home/cis90/simben/edits $
```

In vi:

- Use i to enter insert mode
- make changes
- save with <Esc>:wq



Lab 9 will help
you start building
your vi skills!

*Instructor: remember to mail
students the tech file!*

~/cis90/lab09/mail-tech-all

A Tangent on Spell

spell command

```
/home/cis90/roddyduk/edits $ cat text  
Welcome to the CIS 90 class !!
```

```
/home/cis90/roddyduk/edits $ spell text  
CIS
```

***spell** command flags CIS as misspelled word.*

How can we add CIS to the dictionary?

spell command

```
/home/cis90/roddyduk/edits $ cat text
Welcome to the CIS 90 class !!
/home/cis90/roddyduk/edits $ spell text
CIS
```

*How can we add CIS
to the dictionary?*

```
/home/cis90/roddyduk/edits $ man spell
No manual entry for spell
/home/cis90/roddyduk/edits $ type spell
spell is hashed (/usr/bin/spell)
/home/cis90/roddyduk/edits $ file usr/bin/spell
/usr/bin/spell: Bourne shell script text executable
/home/cis90/roddyduk/edits $ cat /usr/bin/spell
#!/bin/sh
```

*Hmmm. No man page
for spell ????????????*

```
# aspell list mimicks the standard unix spell program, roughly.
```

```
cat "$@" | aspell list --mode=none | sort -u
```

*OK, the actual
command is **aspell***

```
/home/cis90/roddyduk/edits $
```

spell command

ASPELL(1)

Aspell Abbreviated User's Manual

ASPELL(1)

NAME

aspell - interactive spell checker

SYNOPSIS

aspell [options] <command>

DESCRIPTION

aspell is a utility that can function as an ispell -a replacement, as an independent spell checker, as a test utility to test out Aspell features, and as a utility for managing dictionaries.

COMMANDS

<command> is one of:

-?,help

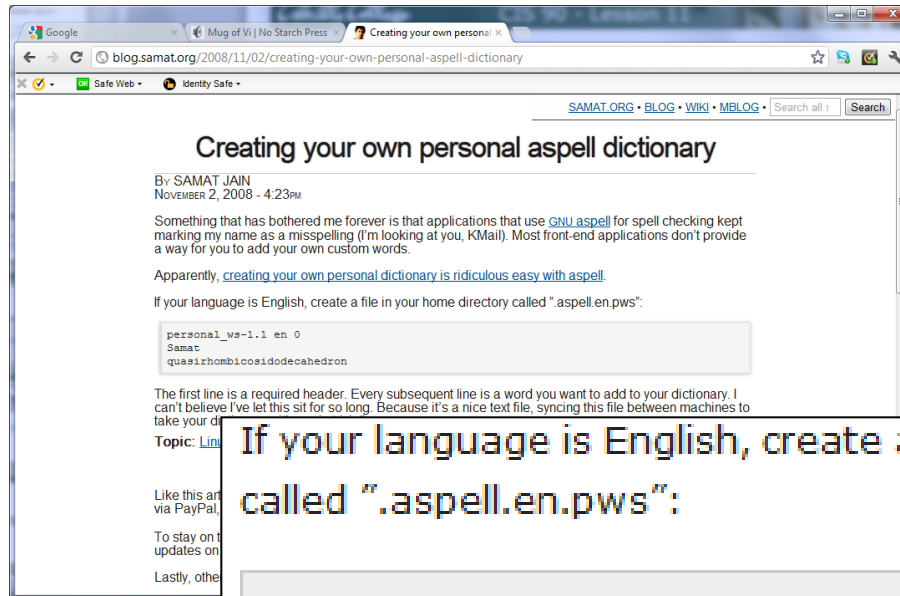
display the help message

-c,check file

to spell-check a file

There must be a way to add CIS but ... lets try google

spell command



*How to add words
to your dictionary*

If your language is English, create a file in your home directory called ".aspell.en.pws":

```
personal_ws-1.1 en 0
Samat
quasirhombicosidodecahedron
```

Googling "linux aspell personal dictionary" yields this page

Bingo! Thank you Samat Jain

spell command

```
/home/cis90/roddyduk/edits $ cd  
/home/cis90/roddyduk $ echo "personal_ws-1.1 en 0" > .aspell.en.pws  
/home/cis90/roddyduk $ echo "CIS" >> .aspell.en.pws  
/home/cis90/roddyduk $ cd edits/  
/home/cis90/roddyduk/edits $ spell text
```

This is how you would add your own custom dictionary to be used with spell checks

```
/home/cis90/simben $ cat edits/spellk  
Spell Check
```

```
Eye halve a spelling chequer  
It came with my pea sea  
It plainly marques four my revue  
Miss steaks eye kin knot sea.  
Eye strike a key and type a word  
And weight four it two say  
Weather eye am wrong oar write  
It shows me strait a weigh.  
As soon as a mist ache is maid  
It nose bee fore two long  
And eye can put the error rite  
Its rare lea ever wrong.  
Eye have run this poem threw it  
I am shore your pleased two no  
Its letter perfect awl the weigh  
My chequer tolled me sew.
```

```
/home/cis90/simben $ spell edits/spellk  
chequer
```

How would you add "chequer"
(the British spelling) to your
personal dictionary?

*Copy the commands used into
the chat window when finished*

Wrap up

New commands:

vi

Run vi editor

New Files and Directories:

na

na

Next Class

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

Lab 9
Five Posts

Quiz questions for next class:

- How do you send a SIGKILL to one of your own processes?
- What vi command is used to exit vi without saving any of the changes you made?
- What vi commands are used for copy and paste?

Backup

The mystery of Ctrl-Z vs Ctrl-F

Signals

Special keystrokes

```
/home/cis90/roddyduk $ stty -a
speed 38400 baud; rows 26; columns 78; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>;
eol2 = <undef>; swtch = <undef>; start = ^Q; stop = ^S; susp = ^F; rprnt = ^R;
werase = ^W; lnext = ^V; flush = ^O; min = 1; time = 0;
```

```
[rsimms@opus ~]$ stty -a
speed 38400 baud; rows 39; columns 84; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>; eol2 = <undef>;
swtch = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W;
lnext = ^V; flush = ^O; min = 1; time = 0;
```

Why does the keystroke to send a Suspend (SIGTSTP or 20) signal differ between roddyduk (^F or Ctrl-F) and rsimms (^Z or Ctrl-Z)?

Job Control

A feature of the bash shell



Ctrl-Z or Ctrl-F (sends SIGTSTP 20 signal)

- Stops (suspends) a foreground process

```
[rsimms@opus ~]$ sleep 5
```

```
[1]+  Stopped                  sleep 5
```

*Ctrl-Z is tapped which
stops the sleep
command*

*PID 7728
is stopped*

```
[rsimms@opus ~]$ ps -l -u rsimms
```

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
5	S	201	5368	5365	0	75	0	-	2460	-	?	00:00:00	sshd
0	S	201	5369	5368	0	76	0	-	1165	wait	pts/0	00:00:00	bash
5	S	201	6203	6200	0	75	0	-	2491	-	?	00:00:00	sshd
0	S	201	6204	6203	0	75	0	-	1165	-	pts/6	00:00:00	bash
0	T	201	7728	6204	0	75	0	-	926	finish	pts/6	00:00:00	sleep
0	R	201	7730	5369	0	78	0	-	1062	-	pts/0	00:00:00	ps

```
[rsimms@opus ~]$
```

Job Control

A feature of the bash shell

bg command

- Resumes a suspended job in the background

```
[rsimms@opus ~]$ sleep 5
```

```
[1]+  Stopped                  sleep 5
```

```
[rsimms@opus ~]$ bg
```

```
[1]+ sleep 5 &
```

```
[rsimms@opus ~]$
```

bg resumes the sleep command

*PID 7728
is gone*

```
[rsimms@opus ~]$ ps -l -u rsimms
```

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
5	S	201	5368	5365	0	75	0	-	2460	-	?	00:00:00	sshd
0	S	201	5369	5368	0	76	0	-	1165	wait	pts/0	00:00:00	bash
5	S	201	6203	6200	0	75	0	-	2491	-	?	00:00:00	sshd
0	S	201	6204	6203	0	75	0	-	1165	-	pts/6	00:00:00	bash
0	R	201	7742	5369	0	78	0	-	1061	-	pts/0	00:00:00	ps

```
[rsimms@opus ~]$
```

Signals

Jim's app script

```
rsimms@opus:/home/cis90/depot
#!/bin/sh
#
# app - script to demonstrate use of signals
#
# Usage:  run app with no options or parameters
#
# Send signals to it with keystrokes or kill command
#
# Notes:
# stty -echo stop the display of characters typed
# stty echo makes typed characters visible again
# stty susp ^Z sets suspend keystroke to Ctrl-Z (to stop foreground processes)
# stty susp @ sets suspend character to @ (to stop foreground processes)
#
trap '' 2 #Ignore SIGINT
trap 'echo -n quit it!' 3 #Handle SIGQUIT
trap 'stty echo susp ^Z;echo ee; echo cleanup;exit' 15 #Handle SIGTERM
clear
banner testing
stty -echo susp @
sleep 1
echo one
sleep 1
echo two
sleep 1
echo -n thr
while :
do sleep 1
done
~
```

This is why Ctrl-F (suspend) stopped working and we had to use Ctrl-Z

13,1 All

Tangent on bg and SIGCONT

Signals

*What is
signal
18?*



Signals

SIGSTKFLT	16	Stack fault
SIGCHLD	17	Child process has stopped or exited, changed (POSIX)
SIGCONT	18	Continue executing, if stopped (POSIX)
SIGSTOP	19	Stop executing(can't be caught or ignored) (POSIX)
SIGTSTP	20	Terminal stop signal (POSIX) Ctrl-Z or Ctrl-F
SIGTTIN	21	Background process trying to read, from TTY (POSIX)
SIGTTOU	22	Background process trying to write, to TTY (POSIX)
SIGURG	23	Urgent condition on socket (4.2 BSD)
SIGXCPU	24	CPU limit exceeded (4.2 BSD)
SIGXFSZ	25	File size limit exceeded (4.2 BSD)
SIGVTALRM	26	Virtual alarm clock (4.2 BSD)
SIGPROF	27	Profiling alarm clock (4.2 BSD)
SIGWINCH	28	Window size change (4.3 BSD, Sun)
SIGIO	29	I/O now possible (4.2 BSD)
SIGPWR	30	Power failure restart (System V)

Signal 18 continues a stopped process ... isn't that what bg does?

The `bg` command is used to resume a stopped process

```
/home/cis90/roddyduk $ sleep 60  
Ctrl-F (or Ctrl-Z) typed here  
[1]+  Stopped                  sleep 60  
/home/cis90/roddyduk $ bg  
[1]+  sleep 60 &  
/home/cis90/roddyduk $ jobs  
[1]+  Running                  sleep 60 &  
/home/cis90/roddyduk $ jobs  
[1]+  Running                  sleep 60 &  
/home/cis90/roddyduk $ jobs  
[1]+  Done                     sleep 60  
/home/cis90/roddyduk $
```

bg resumed the stopped process which runs till it is finished

*Instead of using **bg** to resume a stopped process in the background, lets try a **SIGCONT** (signal 18) instead*

```
/home/cis90/roddyduk $ sleep 60
Ctrl-F (or Ctrl-Z) typed here
[1]+  Stopped                  sleep 60
/home/cis90/roddyduk $ ps -l
F S   UID     PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S   1000  10705  10704  0  76   0  -   1165 wait  pts/0        00:00:00 bash
0 T   1000  10743  10705  0  75   0  -    926 finish pts/0        00:00:00 sleep
0 R   1000  10744  10705  0  78   0  -   1051 -     pts/0        00:00:00 ps
/home/cis90/roddyduk $ jobs
[1]+  Stopped                  sleep 60
/home/cis90/roddyduk $ kill -18 10743
/home/cis90/roddyduk $ jobs
[1]+  Running                  sleep 60 &
/home/cis90/roddyduk $ ps -l
F S   UID     PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S   1000  10705  10704  0  75   0  -   1165 wait  pts/0        00:00:00 bash
0 S   1000  10743  10705  0  85   0  -    926 322800 pts/0        00:00:00 sleep
0 R   1000  10746  10705  0  77   0  -   1050 -     pts/0        00:00:00 ps
/home/cis90/roddyduk $ jobs
[1]+  Running                  sleep 60 &
/home/cis90/roddyduk $ jobs
[1]+  Running                  sleep 60 &
/home/cis90/roddyduk $ jobs
[1]+  Done                     sleep 60
```

Note sending a 18 signal or using the bg command will resume a stopped process