

Lesson Module Checklist

- Slides
- WB
- 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
- Materials uploaded
- Backup slides, CCC info, handouts on flash drive
- Spare 9v battery for mic

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

- 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



Instructor: **Rich Simms**

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

Passcode: **761867**



Buzz



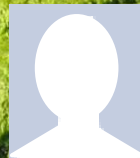
Carlos



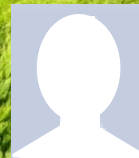
Elijah



Emily



Enrique C.



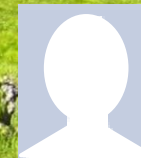
Enrique R.



Jon M.



Jon W.



Jordan



Joseph



JJ



Kiernan



Maria



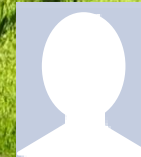
Mathew



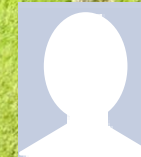
Mike C.



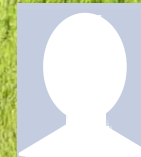
Michael F.



Mike M.



Nick L.



Patrick



Rebecca



Ricardo



Robert



Ruth



Steve



Tess



Tim



Troy

Quiz

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

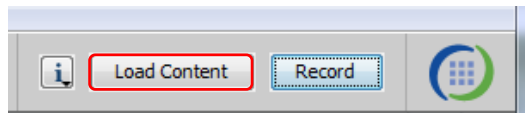
See electronic white board

email answers to: risimms@cabrillo.edu

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

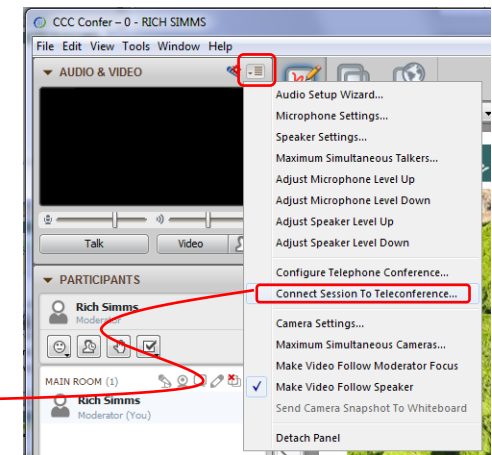
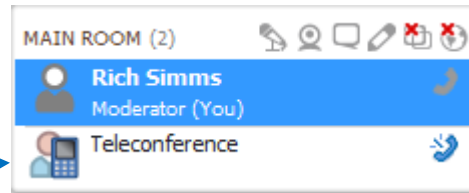


[] Preload White Board with *cis*lesson??*-WB*



[] Connect session to Teleconference

Session now connected to teleconference



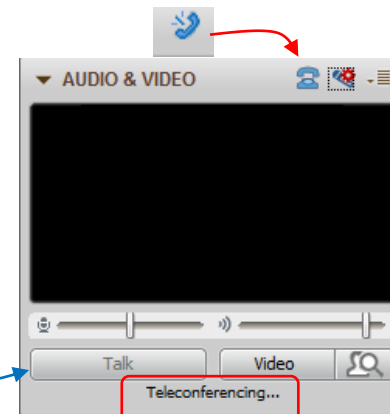
[] Is recording on?



Red dot means recording

[] Use teleconferencing, not mic

Should be greyed out



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



[] layout and share apps

The screenshot displays a Windows desktop environment with several applications open. The taskbar at the bottom shows icons for Internet Explorer, File Explorer, and other standard Windows applications. The desktop background is a light blue gradient.

Applications and their content:

- CCC Confer**: A window titled "CCC Confer - 0 - RIC..." showing a video feed of a person and a list of participants.
- File Explorer**: A window titled "cis90lesson07.pdf - Foxit Reader" showing a directory structure with folders like "boot", "bin", "etc", and "sbin".
- Web Browser**: A window titled "simms-teach.com/docs/cis90/cis-90-TEST-1-Fall-12.pdf" displaying a document with flashcard questions.
- Terminal**: A window titled "simben90@oslab:~" showing a command prompt with the user "simben90" and the directory "/home/cis90/simben".
- vSphere Client**: A window titled "vCenter - vSphere Client" showing a list of virtual machines, including "CIS 192".

Annotations and Red Boxes:

- foxit for slides**: A red box pointing to the Foxit Reader window.
- chrome**: A red box pointing to the web browser window.
- putty**: A red box pointing to the terminal window.
- vSphere Client**: A red box pointing to the vSphere Client window.

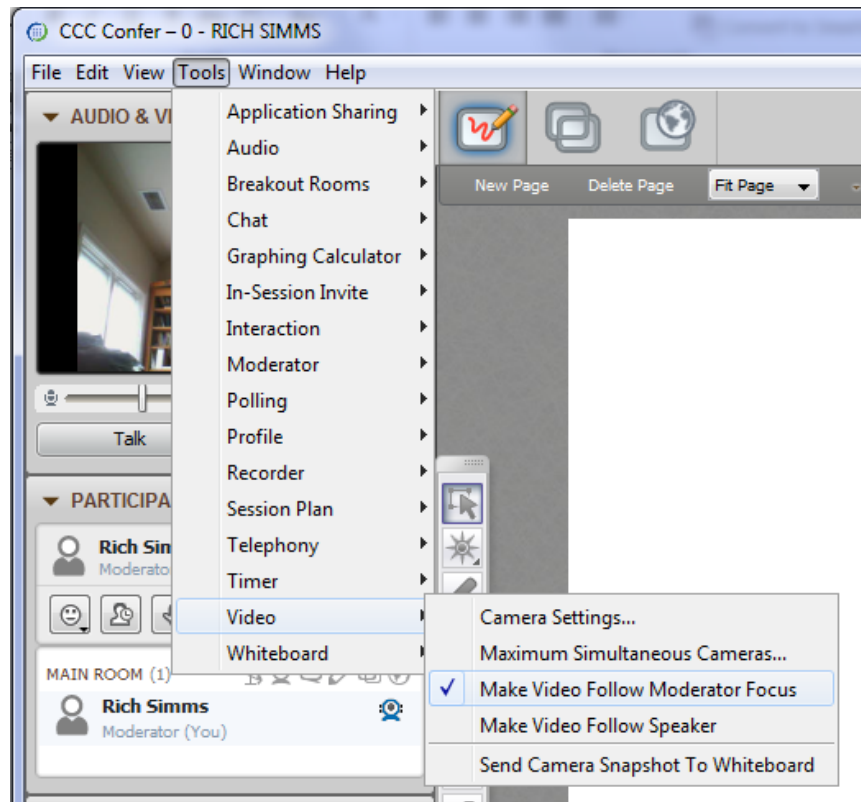
Other visible elements include a "Recycle Bin" icon on the desktop, a "Current directory" section with "source" and "destination" fields, and a "CHAT" window on the left side of the desktop.



[] Video (webcam) optional

[] Follow moderator

[] Double-click on postages stamps



Universal Fix for CCC Confer:

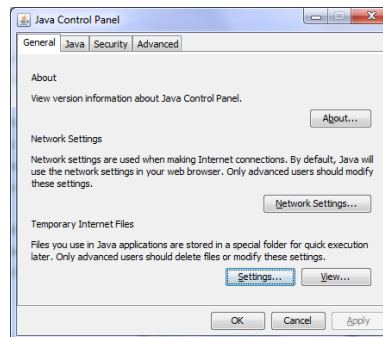
- 1) Shrink (500 MB) and delete Java cache
- 2) Uninstall and reinstall latest Java runtime



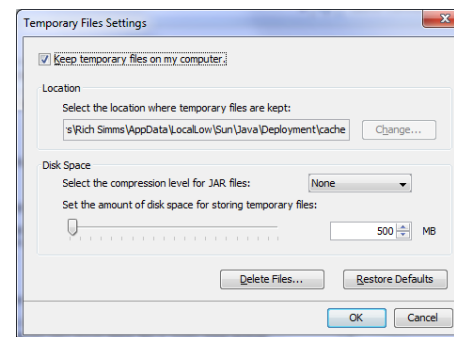
Control Panel (small icons)



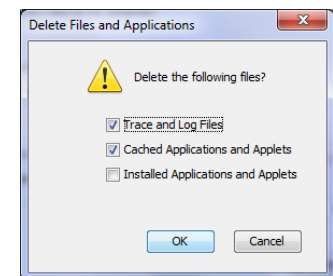
General Tab > Settings...



500MB cache size



Delete these



Google Java download



vi editor

Objectives	Agenda
<ul style="list-style-type: none">• Create and modify text files	<ul style="list-style-type: none">• Quiz• Questions from last week• more on grep• Review on processes• The vi editor• Wrap up

Questions



Questions?

Lesson material?

Labs? Tests?

How this course works?

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

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

- Francis Bacon

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

- Mahatma Gandhi

Chinese
Proverb

他問一個問題，五分鐘是個傻子，他不問一個問題仍然是一個傻瓜永遠。

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

Test 2

Post Mortem

Test 2 – Results

Missed Q4 = 15
Missed Q24 = 15
Missed Q30 = 14
Missed Q28 = 13
Missed Q25 = 13
Missed Q29 = 12
Missed Q26 = 12
Missed Q21 = 12
Missed Q2 = 10
Missed Q17 = 10
Missed Q27 = 9
Missed Q23 = 9
Missed Q13 = 8
Missed Q20 = 7
Missed Q18 = 7

Missed Q16 = 7
Missed Q11 = 7
Missed Q3 = 6
Missed Q22 = 6
Missed Q19 = 6
Missed Q12 = 5
Missed Q9 = 4
Missed Q6 = 4
Missed Q15 = 4
Missed Q14 = 4
Missed Q7 = 2
Missed Q5 = 2
Missed Q8 = 1
Missed Q10 = 0
Missed Q1 = 0



Extra Credit

Missed Q31 = 18
Missed Q32 = 17
Missed Q33 = 14

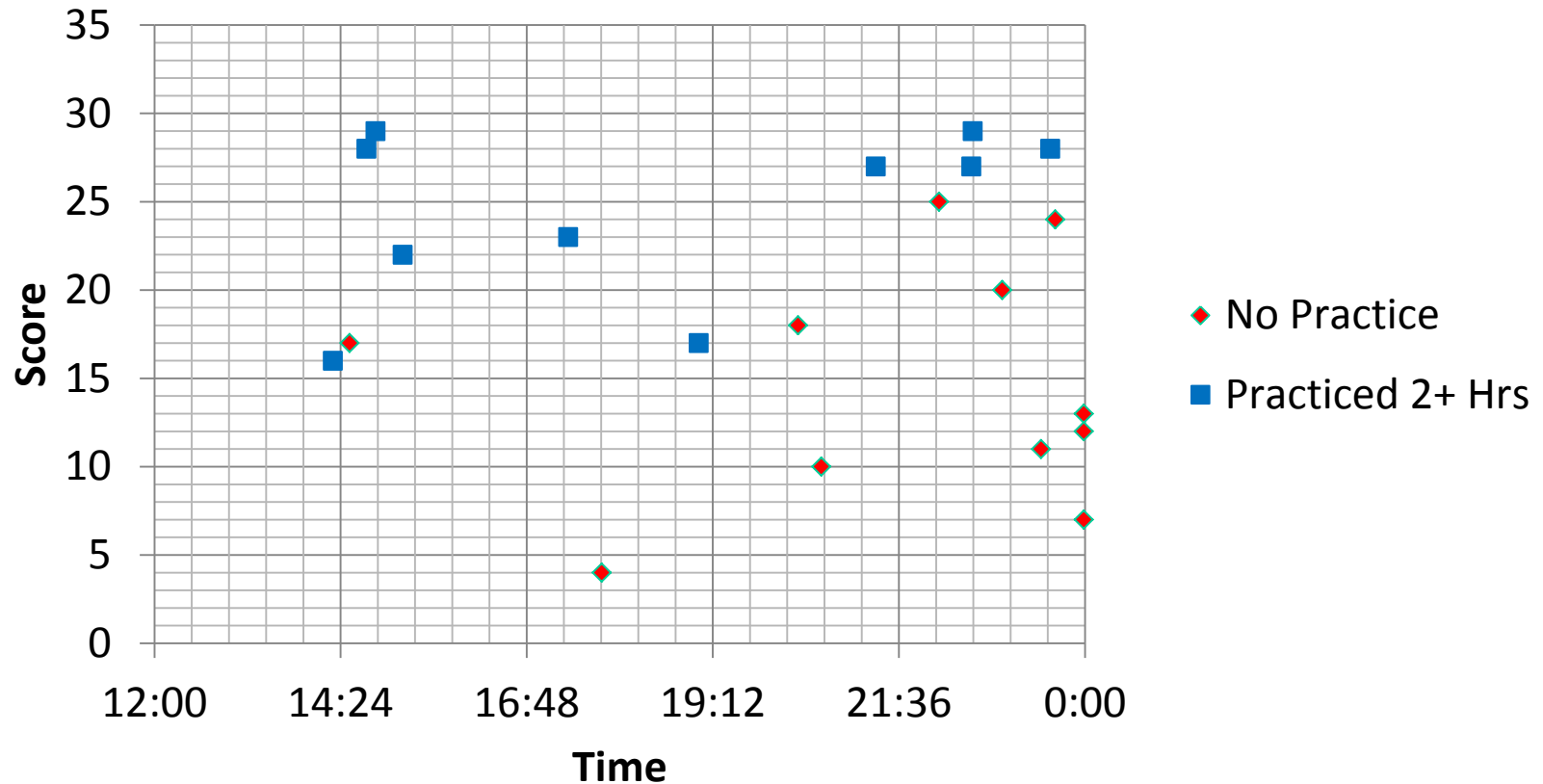
*21 tests
submitted*



*6 tests not
submitted*



CIS 90 Test 2



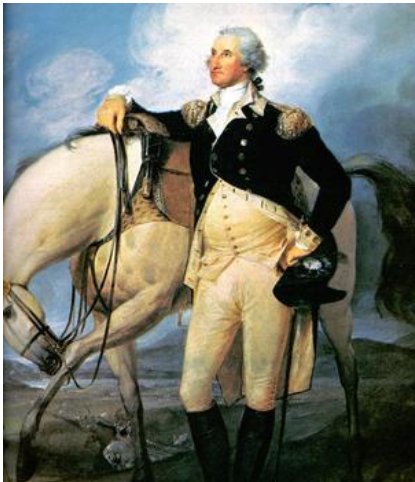
Working the practice test tended to result in higher scores and shorter completion times

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

To check your answer (lesson 4)

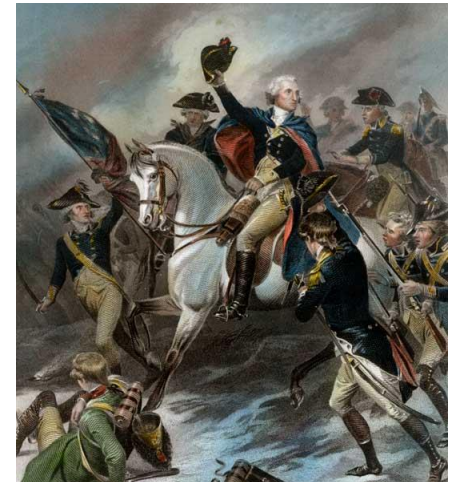
```
[simben90@sun-hwa-ii ~]$ ls /etc/passwd
/etc/passwd
```



<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>

Re: Questions about the Final Exam

by **Rich Simms** » Sun Apr 13, 2014 11:05 am

I just reconfigured son-of-opus as a Test #2 system. You can use it for review of what we've learned so far or to better understand any questions missed on the last test.

To login use: `ssh -p 2220 username@son-of-opus.simms-teach.com`

Have fun!

- Rich



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! Test 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

Careers for CS and CIS students

by **Rich Simms** » Sat Apr 12, 2014 7:33 am

Find out about the following Cabrillo programs and classes next week:

Computer Support Specialists
Network & System Administrators
Software Developers

The CS/CIS instructors will be presenting and answering questions at this meetup next week.

Technology Career Meetup

When: April 18, 2014

Time: 1:00-2:30 PM

Where: Cabrillo College, Room 828

Contact: Rick Graziani 831.477.3573

Contact: Gerlinde Brady 831.477.5672

Contact: Steve Hodges 831.479.6328

<http://www.cabrillo.edu/academics/cis/>

<http://www.cabrillo.edu/academics/computerscience/>

Computer Support Specialists are in High Demand!

\$28,000 Average Starting Salaries, Two-Year Program

Many companies hire qualified Computer Support Specialists.



Register Now!

The Computer Support Specialist program at Cabrillo prepares students for a career in the field of computer technical support. Courses provide hands-on experience with computer hardware and software, basic networking and help desk concepts. The program prepares students to pass industry accepted certifications like CompTIA.

Network & System Administrators are in Demand!

\$46,000 Average Starting Salaries

Most companies need qualified Network and System Administrators.



Register Now!

Cabrillo College's Computer Networking and System Administration program prepares students to work in the Information Technology (IT) industry in general, and more specifically in computer networking and system administration. After completing the program you will have the skills needed to transfer to a four-year college or university and complete your degree. Cabrillo transfer students excel at four-year colleges and universities!

Software Developers are in High Demand!

\$61,741 Average Starting Salaries

Companies such as Facebook always need qualified software developers.



Register Now!

To become a qualified software developer, you need a bachelor's degree in computer science. Register now for Cabrillo College's program in Computer Science. After completing the program you will have the development and programming skills needed to transfer to a four-year college or university and complete your degree. Cabrillo transfer students excel at four-year colleges and universities!

Got Questions?

If you are interested in learning more about the Computer Science program, contact the Computer Science department at Cabrillo College today. Or attend our Tech meetup shown below.

Technology Career Meetup

When: April 18, 2014 1:00-2:30 PM

Where: Cabrillo College, Room 828

Final Exam

Test #3 (final exam)

- Must be face-to-face or proctored (not online using CCC Confer).
- Room 828 on campus.
- Timed test (no 11:59PM grace period)

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

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

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

GRADES

- Check your progress on the Grades page
- If you haven't already, send me a student survey to get your LOR secret code name
- Graded labs & tests are placed in your home directories on Opus
- Answers to labs, tests and quizzes are in the `/home/cis90/answers` directory on Opus

Current Point Tally

As of 4/15/2014

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

alatar: 61% (202 of 331 points)
 anborn: 81% (270 of 331 points)
 aragorn: 85% (282 of 331 points)
 arwen: 99% (330 of 331 points)
 beregond: 0% (0 of 331 points)
 bilbo: 54% (181 of 331 points)
 celebrian: 98% (327 of 331 points)
 dwalin: 95% (317 of 331 points)
 eomer: 89% (296 of 331 points)
 faramir: 93% (310 of 331 points)
 frodo: 90% (300 of 331 points)
 gwaihir: 106% (351 of 331 points)
 ioreth: 94% (312 of 331 points)

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

legolas: 87% (291 of 331 points)
 marhari: 66% (220 of 331 points)
 orome: 82% (273 of 331 points)
 pallando: 0% (0 of 331 points)
 pippen: 67% (225 of 331 points)
 quickbeam: 91% (304 of 331 points)
 rian: 0% (0 of 331 points)
 samwise: 80% (266 of 331 points)
 shadowfax: 0% (0 of 331 points)
 strider: 89% (296 of 331 points)
 theoden: 56% (187 of 331 points)
 treebeard: 104% (347 of 331 points)
 tulkas: 85% (282 of 331 points)
 ulmo: 80% (268 of 331 points)

Jesse's checkgrades python script

<http://oslab.cabrillo.edu/forum/viewtopic.php?f=31&t=773&p=2966>

```
/home/cis90/simben $ checkgrades smeagol
```

Remember, your points may be zero simply because the assignment has not been graded yet.

Quiz 1: You earned 3 points out of a possible 3.

Quiz 2: You earned 3 points out of a possible 3.

Quiz 3: You earned 3 points out of a possible 3.

Quiz 4: You earned 3 points out of a possible 3.

Forum Post 1: You earned 20 points out of a possible 20.

Lab 1: You earned 30 points out of a possible 30.

Lab 2: You earned 30 points out of a possible 30.

Lab 3: You earned 30 points out of a possible 30.

Lab 4: You earned 29 points out of a possible 30.

You've earned 15 points of extra credit.

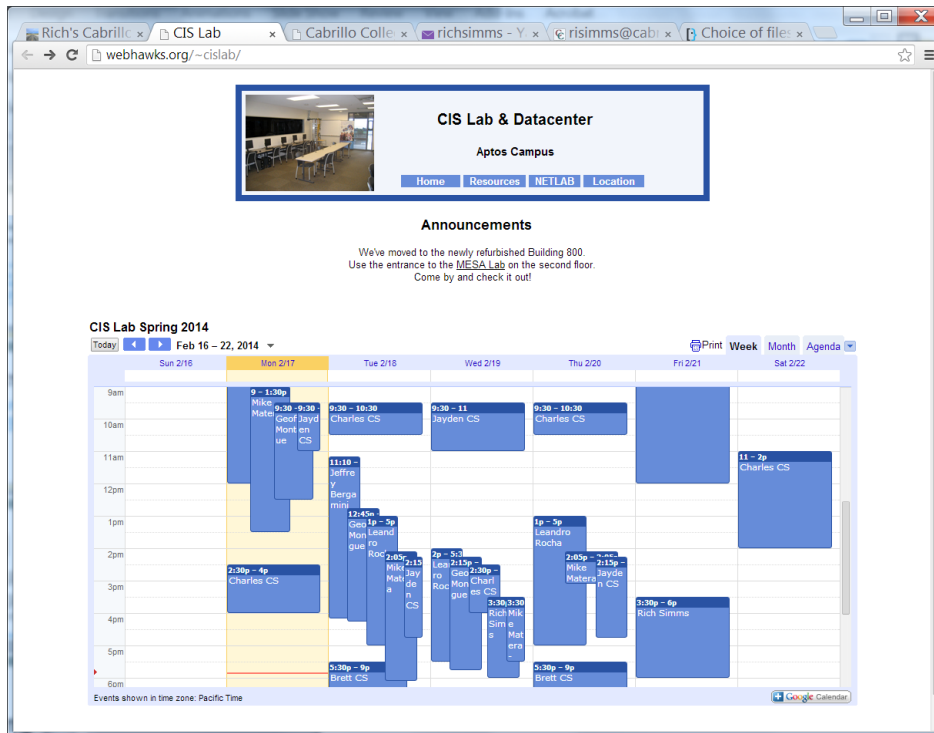
You currently have a 109% grade in this class. (166 out of 152 possible points.)

*Use your LOR
code name as
an argument on
the checkgrades
command*

Jesse is a CIS 90 Alumnus. He wrote this python script when taking the course. It mines data from the website to check how many of the available points have been earned so far.

CIS Lab Schedule

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



Not submitting tests or lab work?

Would like some help?

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

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

Free CIS 90 Tutoring Available

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

TUTORIALS

ANNOUNCEMENTS & DEADLINES

- New subjects for Spring 2014:
- American Sign Language
- Computer Applications/Business Technology (CABT)
- Computer and Information Systems (CIS)
- History 17A

Welcome to the Tutorials Center!

We offer FREE peer tutoring to Cabrillo students who are enrolled in the course/s for which they need help.

- Tutoring is by appointment. The days and times of tutoring sessions are established by the office.
- Sessions are weekly and for the duration of the semester.
- Tutoring sessions are scheduled in small groups. Sessions last 1-2 hours depending on the class. Occasionally, sessions may be one to one but that is not guaranteed.
- Come directly to the TC office to schedule (second floor of library).

The following classes are being tutored for Spring 2014:

- Accounting 1A, 1B, 6, 54A, 151A, 159, 163
- American Sign Language (ASL) 1, 2
- Biology 4, 5, 6
- Computer Applications/Business Technology (CABT) 31, 38, 41, 101, 157, 160
- Computer and Information Systems (CIS) 81, 90, 172**
- Chemistry 1A, 1B, 2, 30A, 30B, 32

CONTACT INFORMATION

Tutorials Center

Location: Room 1080A - Learning Resource Center

Phone: 831.479.6470

Email: tutorialscenter@cabrillo.edu

Coordinator: Lori Chavez

Phone: 831.479.6126

Email: lochavez@cabrillo.edu

Hours: Monday - Thursday: 9am - 5pm
Friday: 9am - 1pm

[MAP, DIRECTIONS, & PARKING](#)

[DEPARTMENT STAFF & FACULTY DIRECTORY](#)



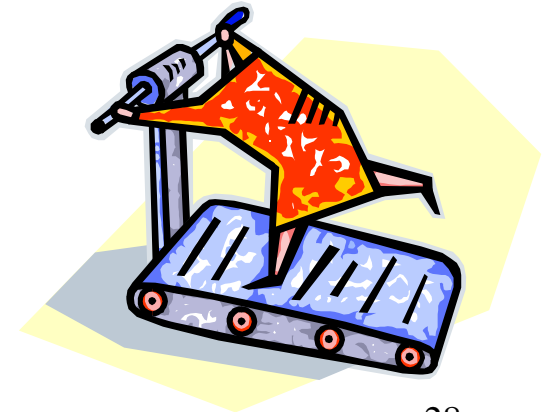
Matt Smithey

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

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

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

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 portion (or all) 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

- For the bash, dash, ksh, sh and csh shells, which shell process uses the least memory?
- What command that would redirect the line of output for the command using the least amount of memory to the file *smallshell*
- Type the command you used and its output into the chat window

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?

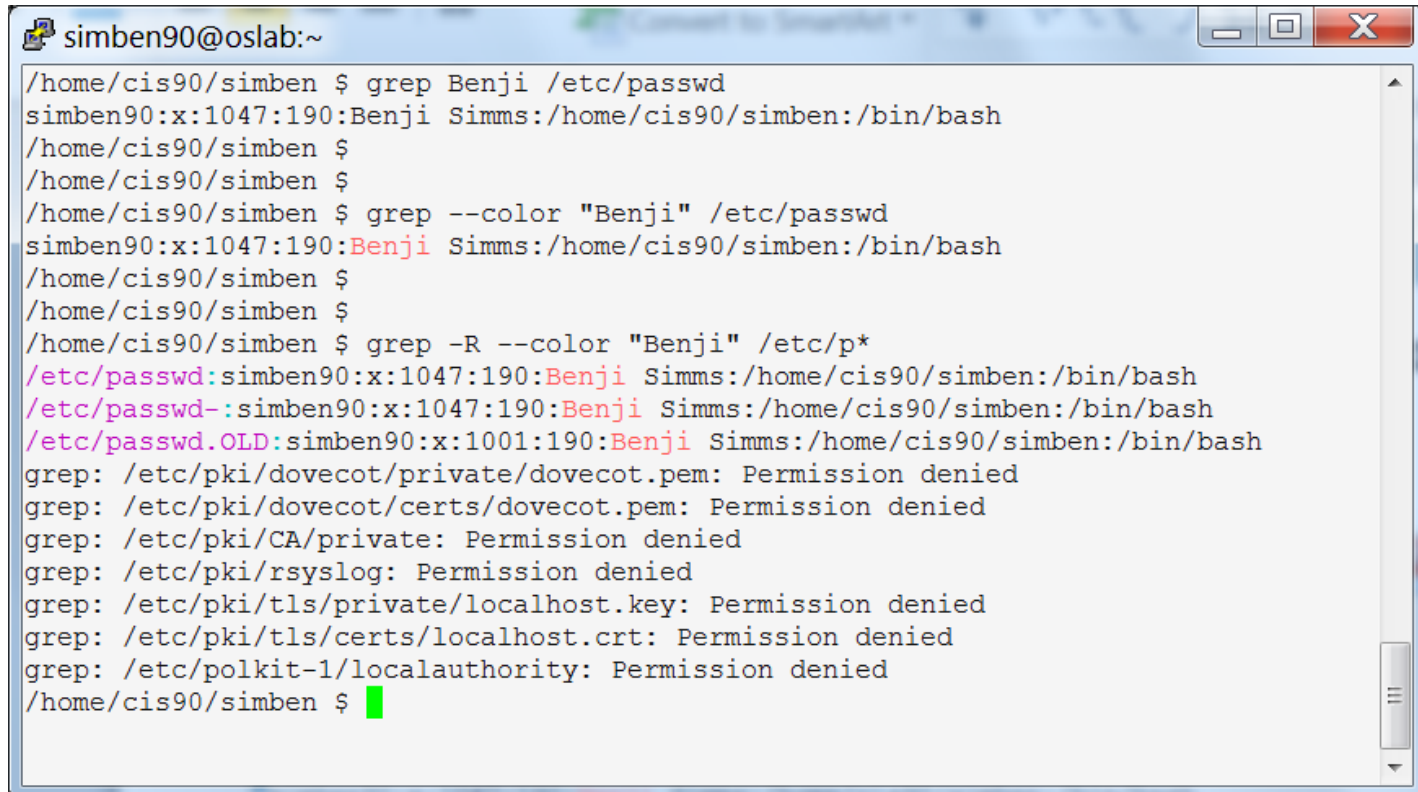
```
/home/cis90/simben $ grep -R "PS1=" /etc/bash* $HOME 2> /dev/null
/etc/bash_completion.d/git:# PS1='[\u@\h \W$(__git_ps1 "
(%s)"]\ $ '
/etc/bashrc: [ "$PS1" = "\s-\v\\\$ " ] && PS1="[\u@\h \W]\\\$ "
/etc/bashrc: # PS1="[\u@\h:\l \W]\\\$ "
/home/cis90/simben/class/labs/lab04.graded:21) PS1='$PWD $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

A terminal window titled 'simben90@oslab:~' showing a series of commands and their outputs. The commands demonstrate the use of grep to search for the name 'Benji' in the /etc/passwd file and recursively in the /etc directory. The output shows that grep found 'Benji' in /etc/passwd and /etc/passwd.OLD, but it denied permission to search in several other files and directories. The text 'Benji' is highlighted in red in the output lines.

```
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

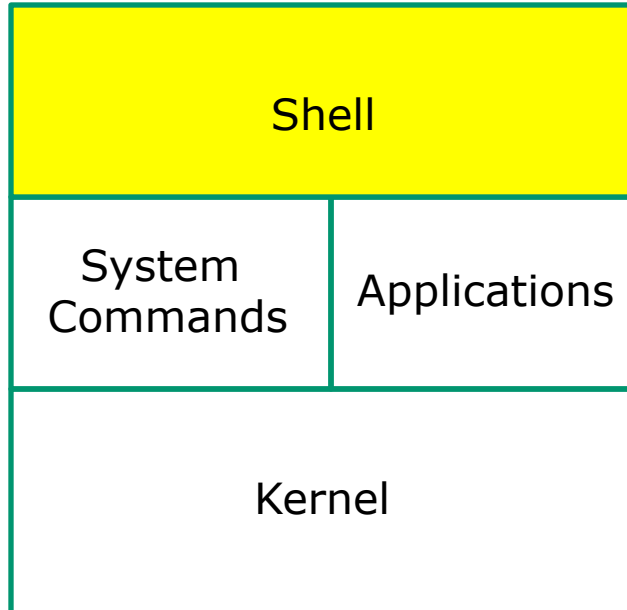
```
/home/cis90/simben $ find / -name *egg 2> /dev/null
```

```
/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. The last field in the line for his account in */etc/passwd* determines the shell that is run when he logs 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

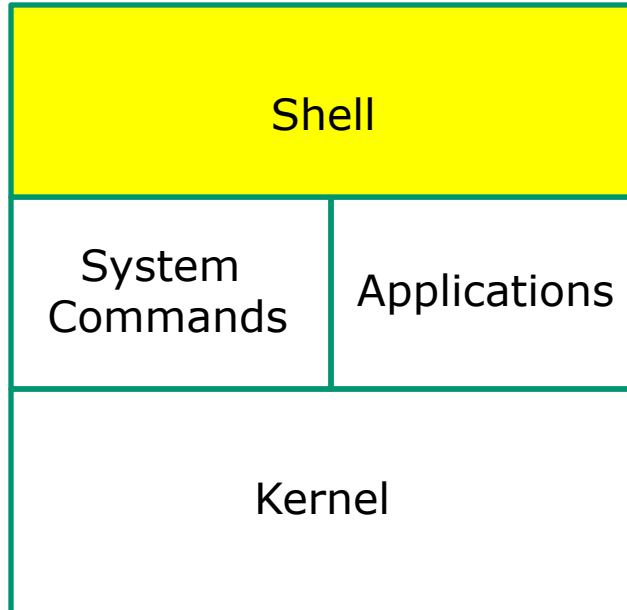
```
/home/cis90/simben $ find / -name *egg 2> /dev/null
```



Benji types this find command in response to the shell prompt



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:

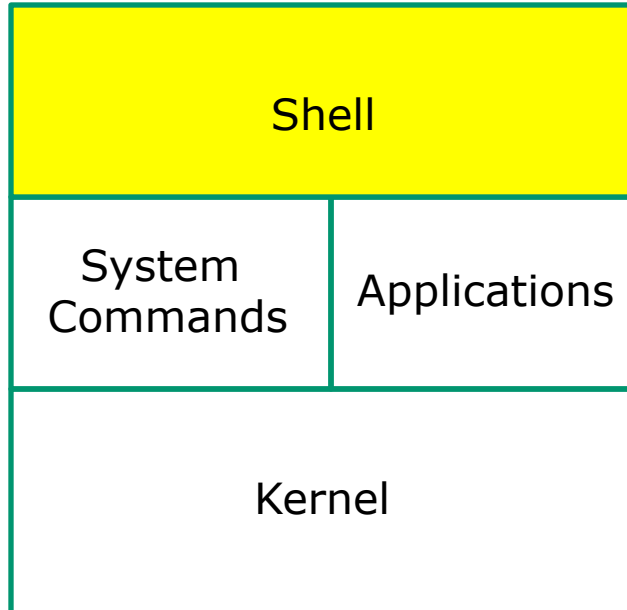
Connect **stderr** to **/dev/null** (the "bit bucket")

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

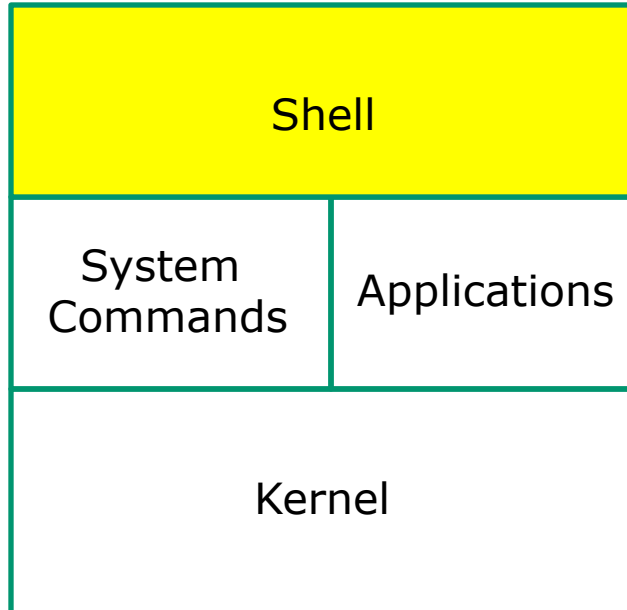
- 1st directory searched: /usr/lib/qt-3.3/bin
- 2nd directory searched: /usr/local/bin
- 3rd directory searched: **/bin**
- 4th directory searched: /usr/bin
- 5th directory searched: /usr/local/sbin
- 6th directory searched: /usr/sbin
- 7th directory searched: /sbin
- 8th directory searched: /home/cis90/simben/./bin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




Execute Step

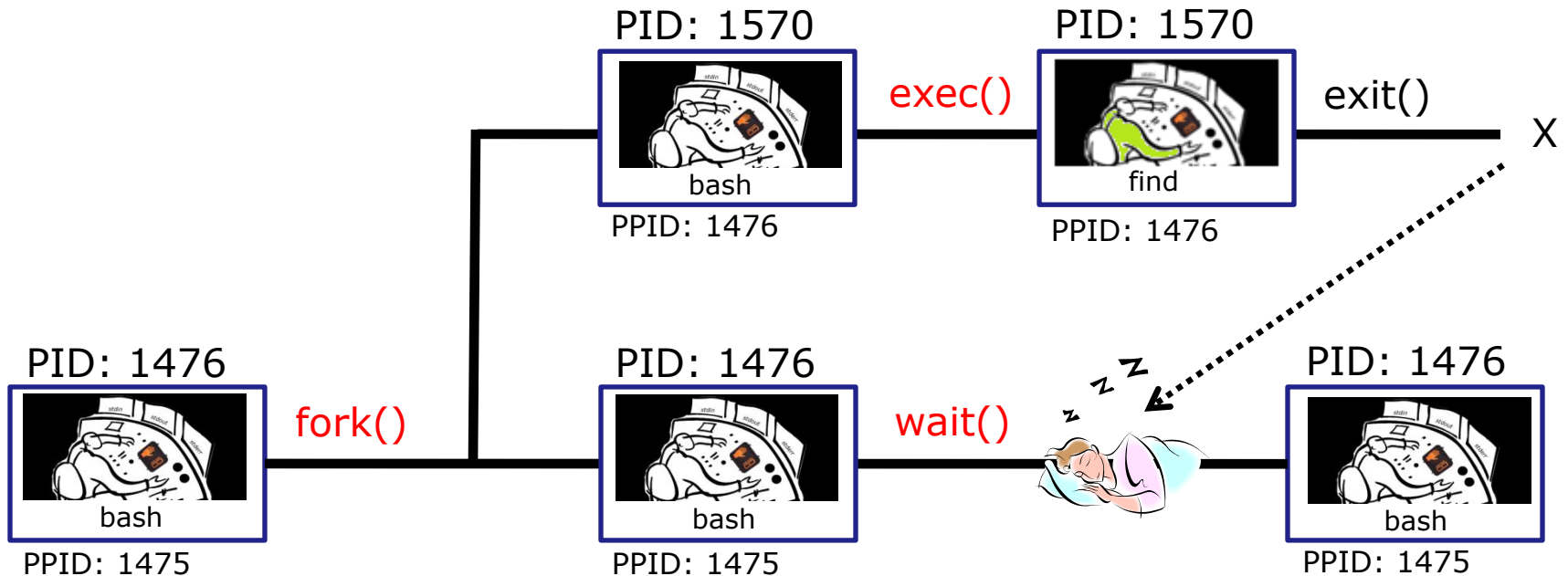


- 1) Prompt
- 2) Parse
- 3) Search
- 4) Execute**
- 5) Nap
- 6) Repeat





Execute Step

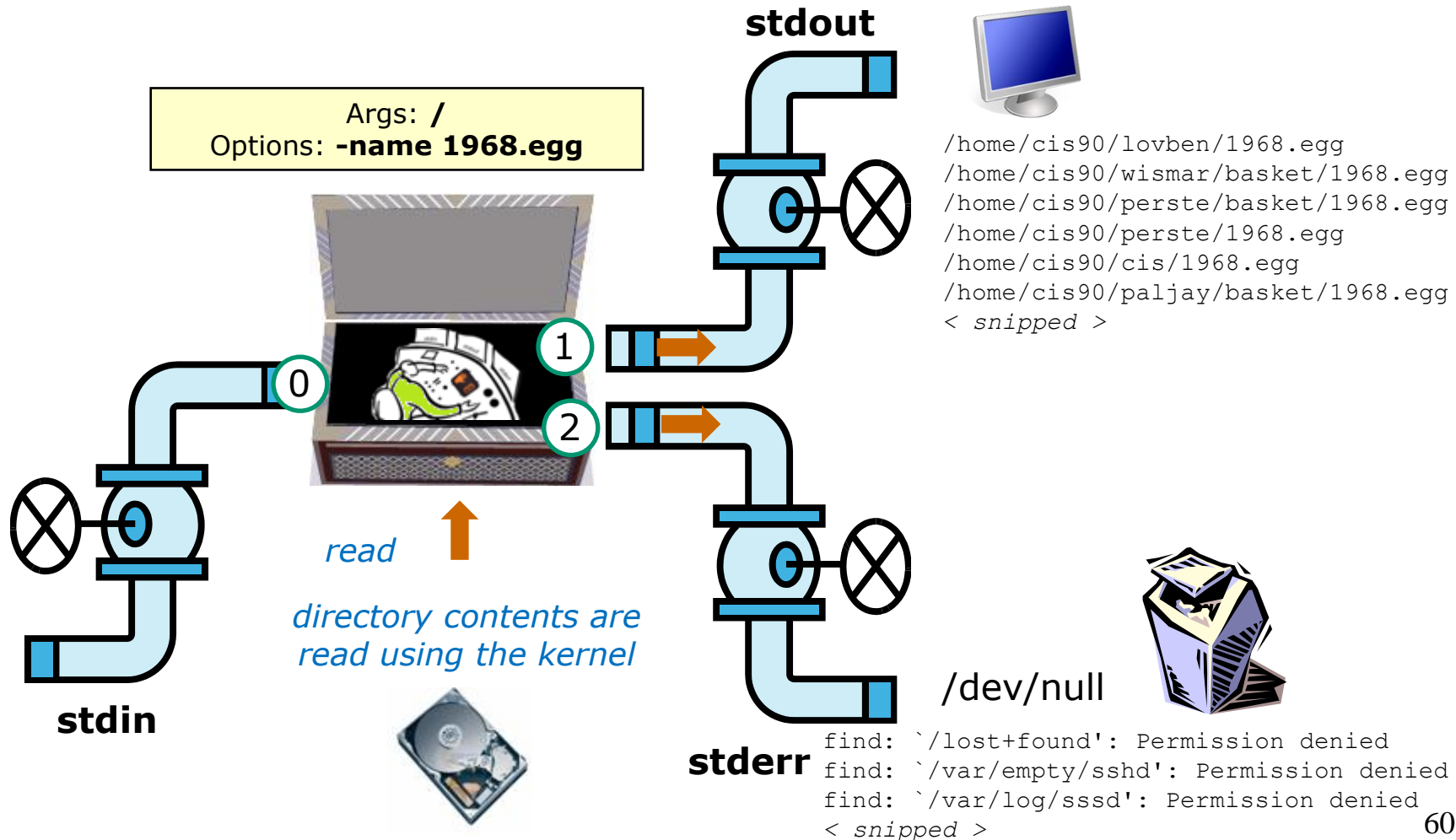


bash executes the **find** command by cloning itself with a **fork()** system call to create a new child process. With an **exec()** system call, the new child process is overlaid with the `find` code instructions. *bash* sleeps by making a **wait()** system call while the `find` child process runs.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

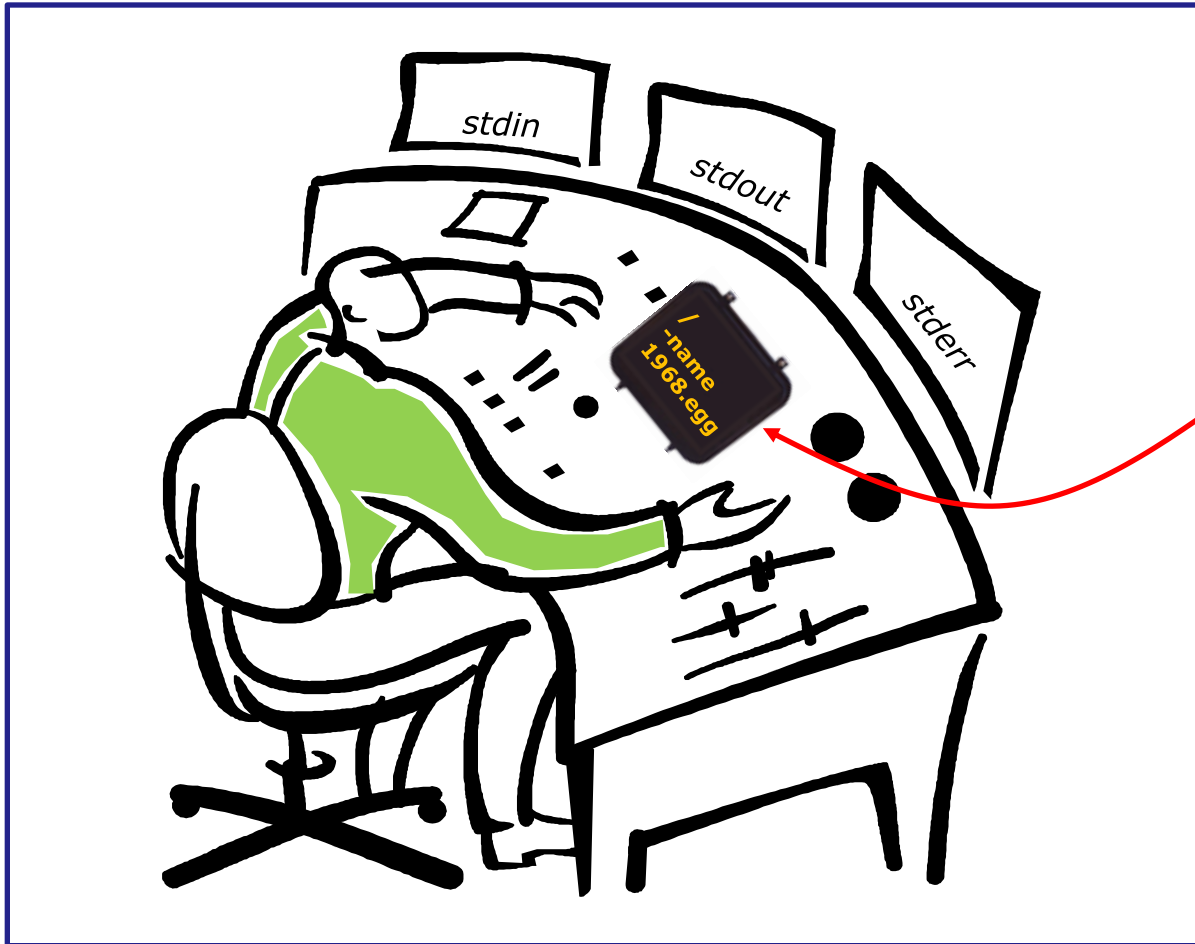


Execute Step

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



This is what the find process might look like



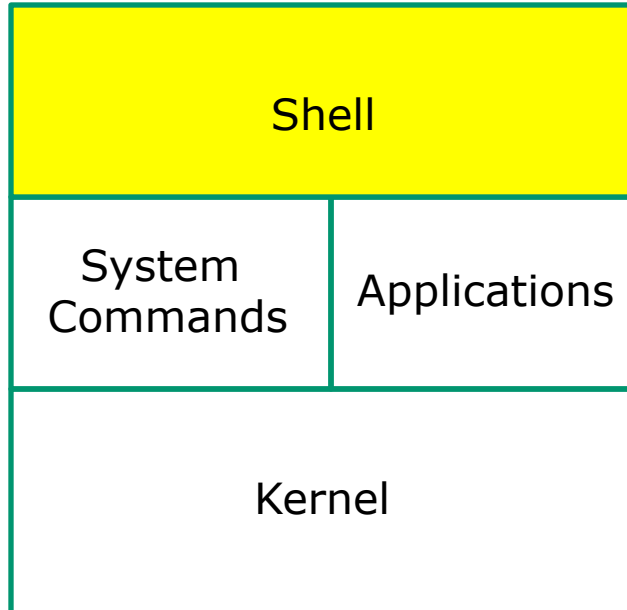
A process:

- 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

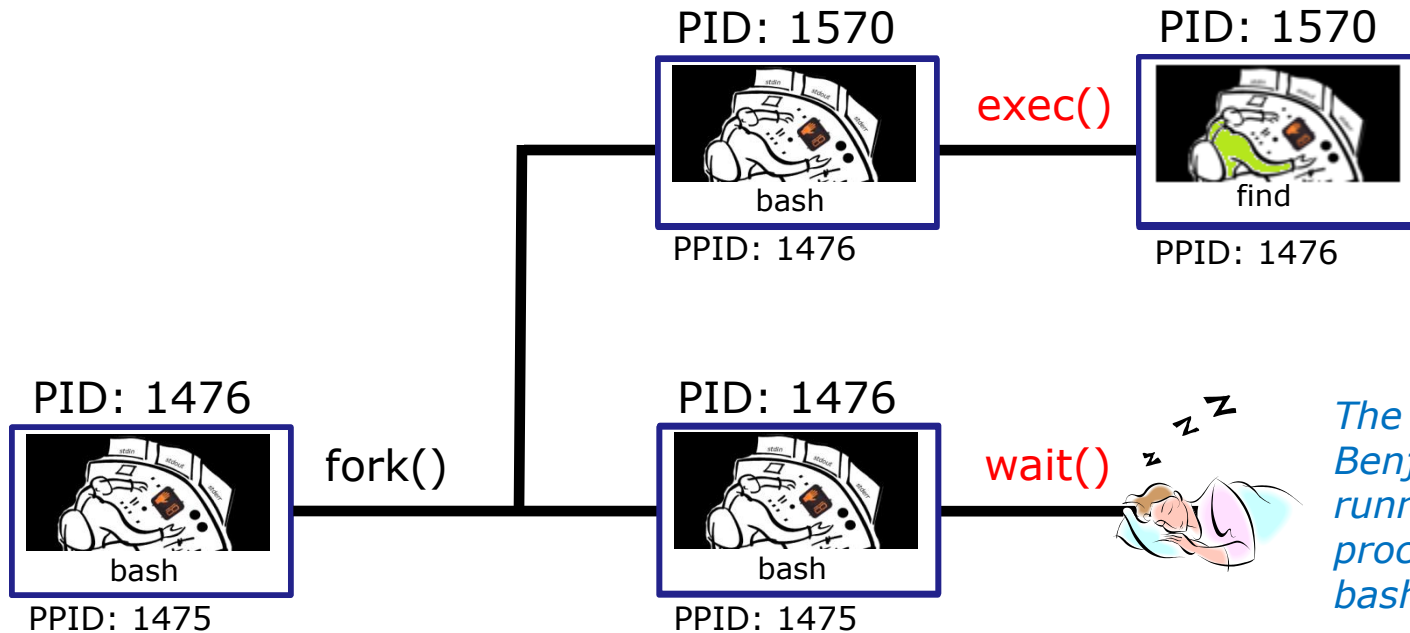


- 1) Prompt
- 2) Parse
- 3) Search
- 4) Execute
- 5) Nap**
- 6) Repeat





Nap Step



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

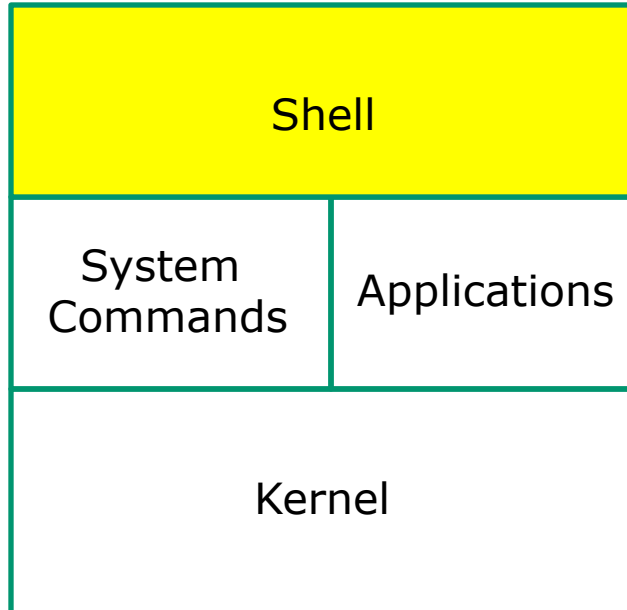
```
[rsimms@oslab ~]$ ps -l -u simben90
```

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

R=Running (PID 1570 find), S=Sleeping (PID 1476 bash)



Repeat Step

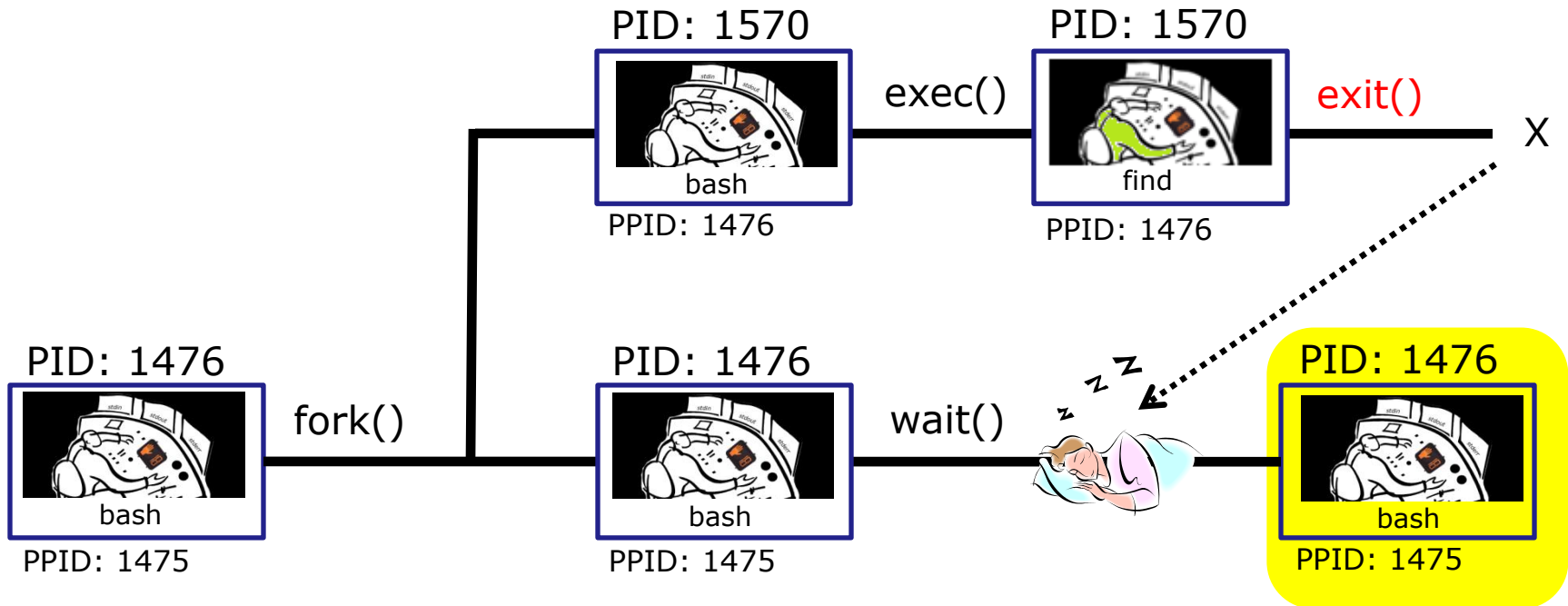


- 1) Prompt
- 2) Parse
- 3) Search
- 4) Execute
- 5) Nap
- 6) Repeat**





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/lib /usr/src**

```
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**

```

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

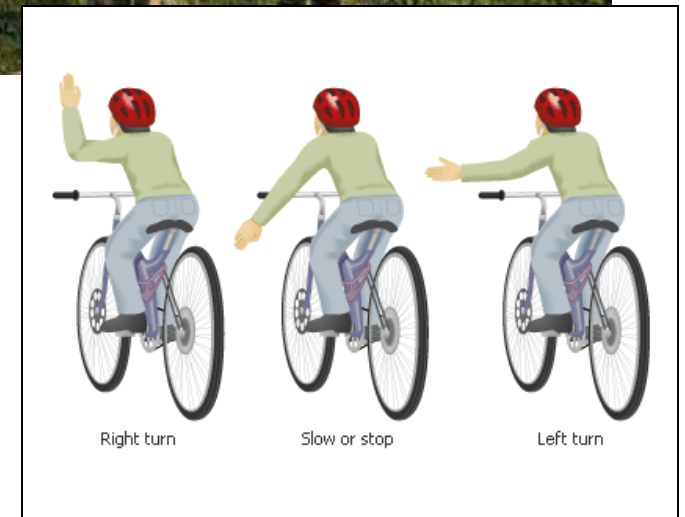
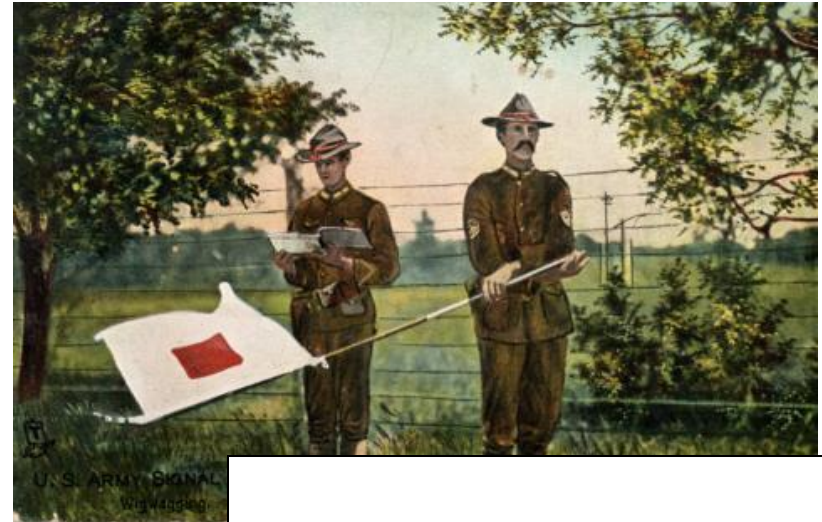
Review of Signals

Signals

PLATE 4

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

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.

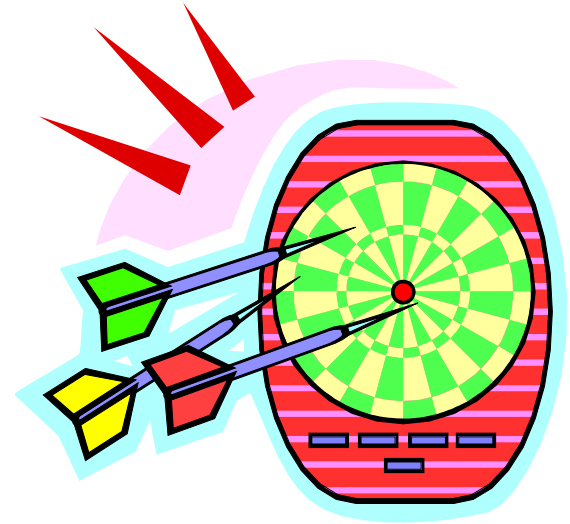


Using special keystrokes

- 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

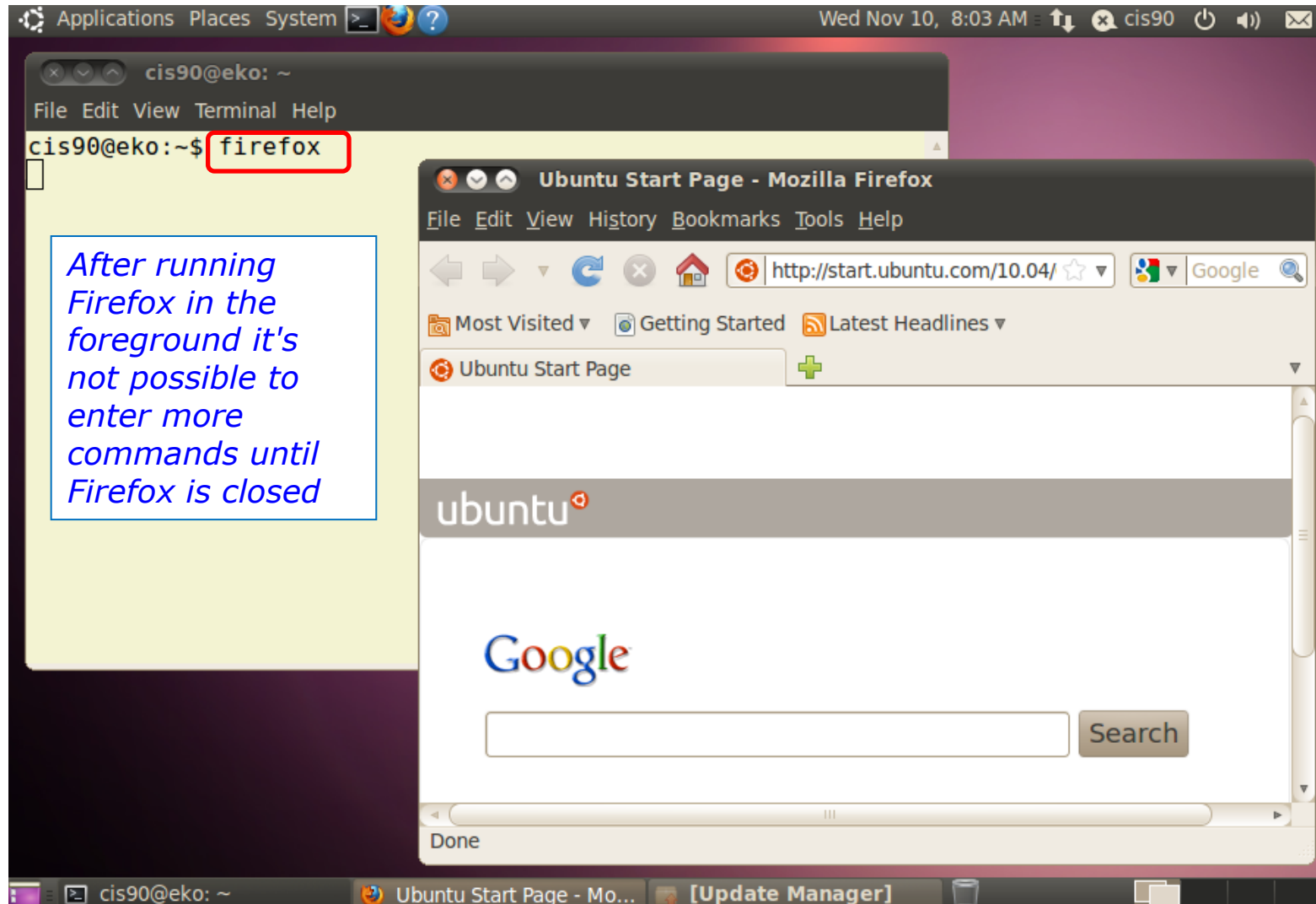
- 1) Run the **annoy** program
- 2) Try sending it a SIGINT with **Ctrl-C**
- 3) Try sending it a SIGQUIT with **Ctrl-**
- 4) Bring up another terminal and try signals 1 through 64
 - Use **ps -u \$LOGNAME** to find the **annoy** *PID*
 - Try **kill -1 *PID***
 - Try **kill -2 *PID***
 - Try **kill -3 *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 desktop environment. In the foreground, a terminal window titled 'cis90@eko: ~' displays the following commands and output:

```

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:~$ 
  
```

The command `firefox &` is highlighted with a red box. Below the terminal output, a blue-bordered box contains the text: "After running Firefox in the background, it is still possible to enter more commands."

In the background, a Mozilla Firefox browser window titled 'Ubuntu Start Page - Mozilla Firefox' is open, displaying the Ubuntu Start Page at <http://start.ubuntu.com/1>. The browser's address bar, search bar, and navigation buttons are visible.

& 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

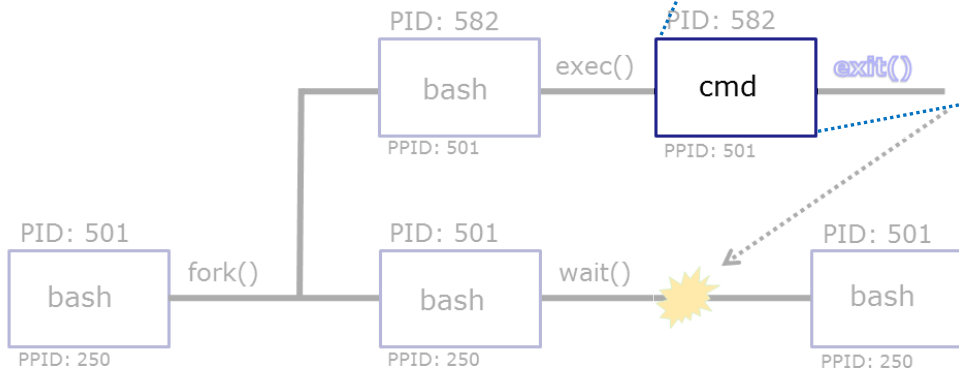
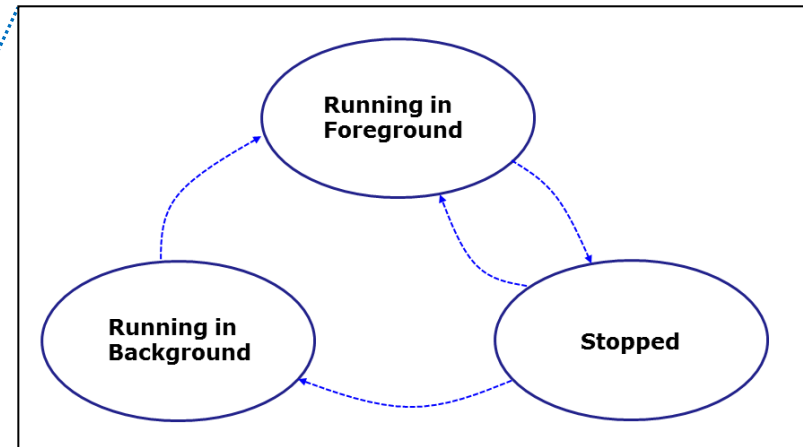
A feature of the bash shell

&	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

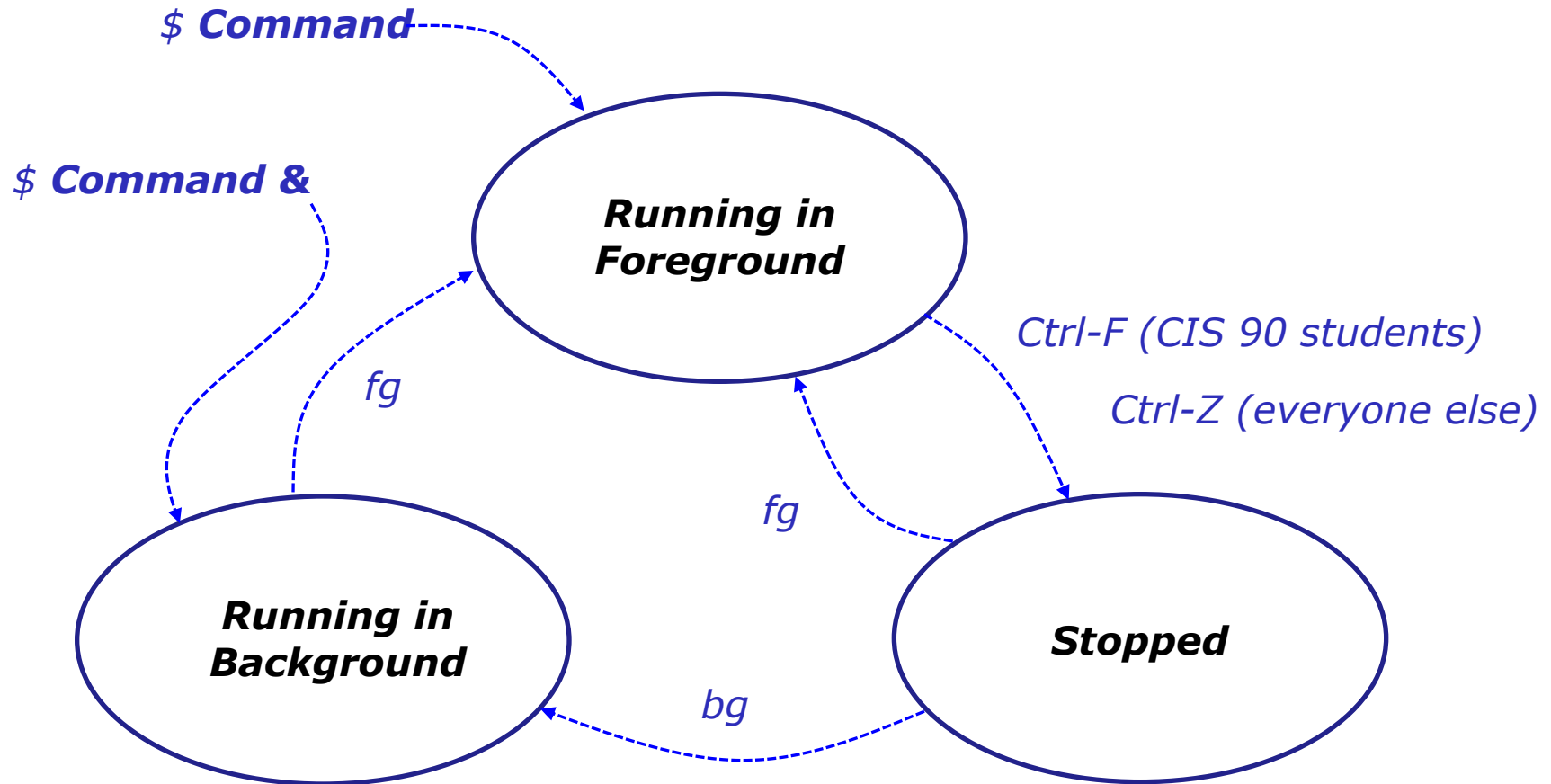
*Use **jobs**, **bg**, **fg** to list and resume jobs in the foreground or background*

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

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 5/5/2012

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*

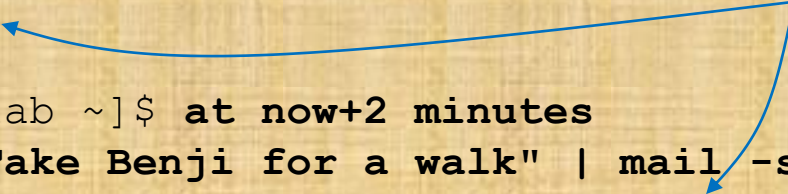
```
/home/cis90/simben $ jobs
```

*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/4  
  
[rsimms@oslab ~]$ at now+2 minutes  
at> echo "Take Benji for a walk" | mail -s "walk the dog" $LOGNAME  
at> echo "Read your mail" > /dev/pts/4  
at> <EOT>  
job 11 at 2012-11-05 11:02  
[rsimms@oslab ~]$ atq  
11      2012-11-05 11:02 a rsimms  
[rsimms@oslab ~]$
```



These should match

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.*

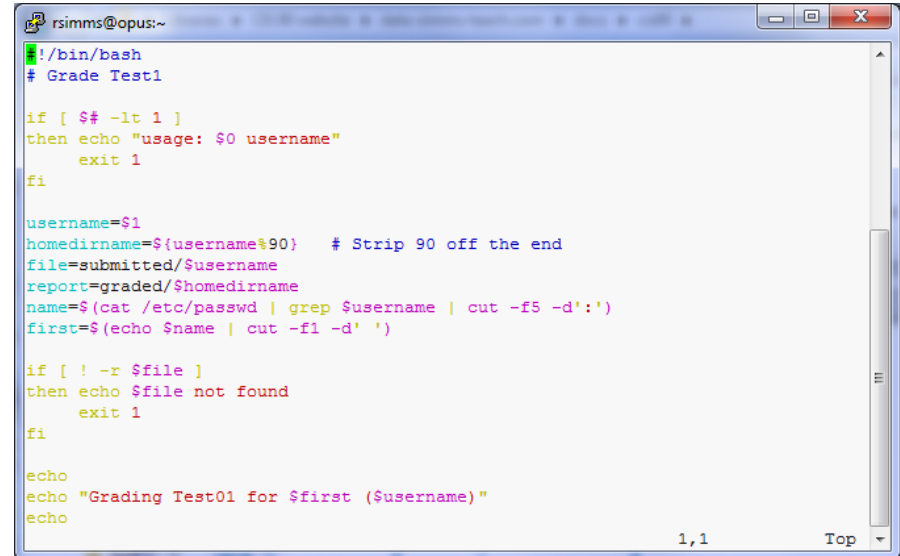
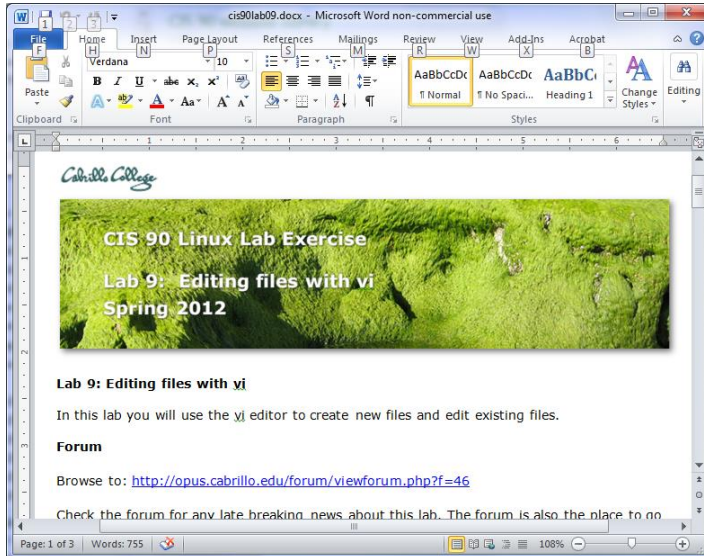
Linux

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:

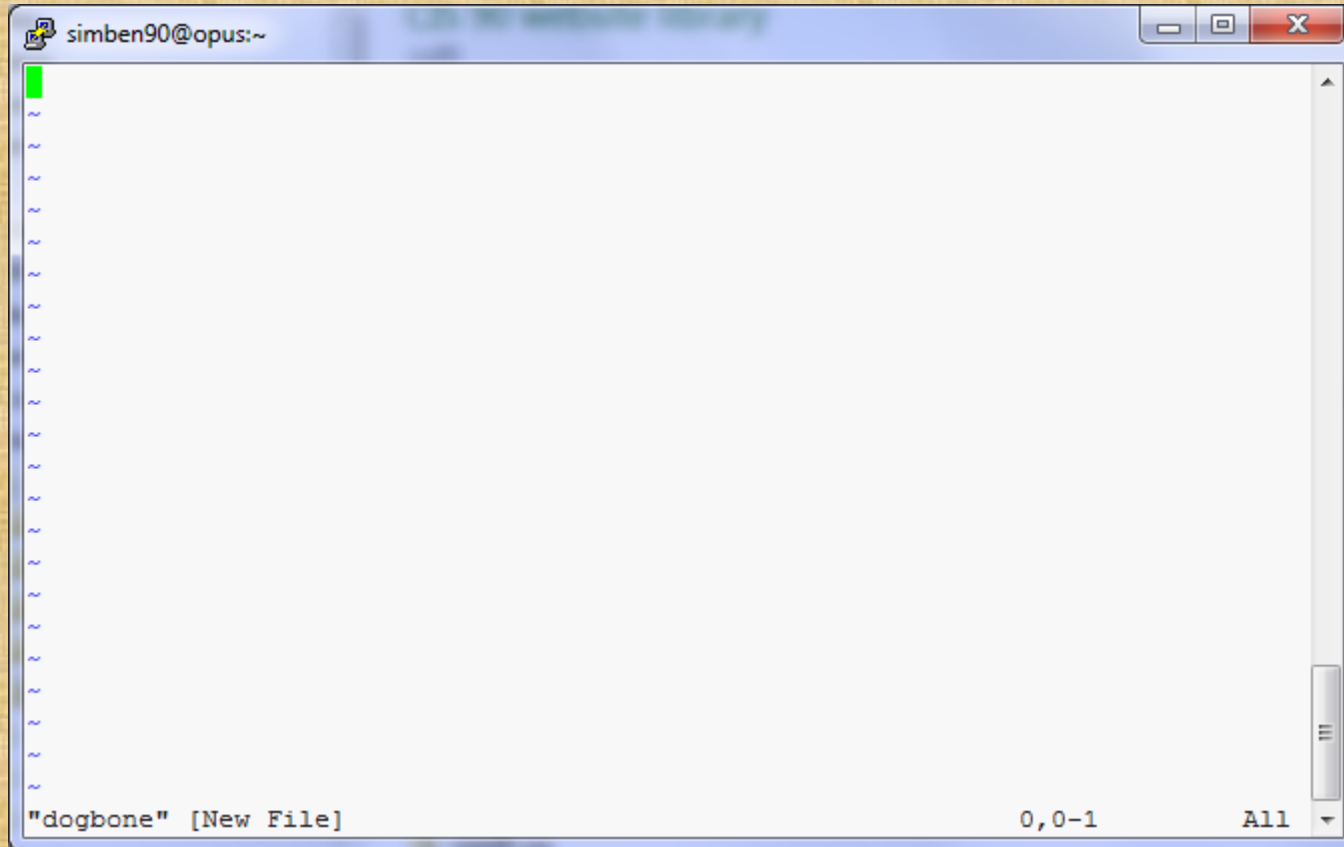
- The original vi code was written by Bill Joy for BSD Unix
- Bill Joy co-founded Sun Microsystems in 1982
- vi (for "visual")
- vim is an enhanced version of vi

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

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

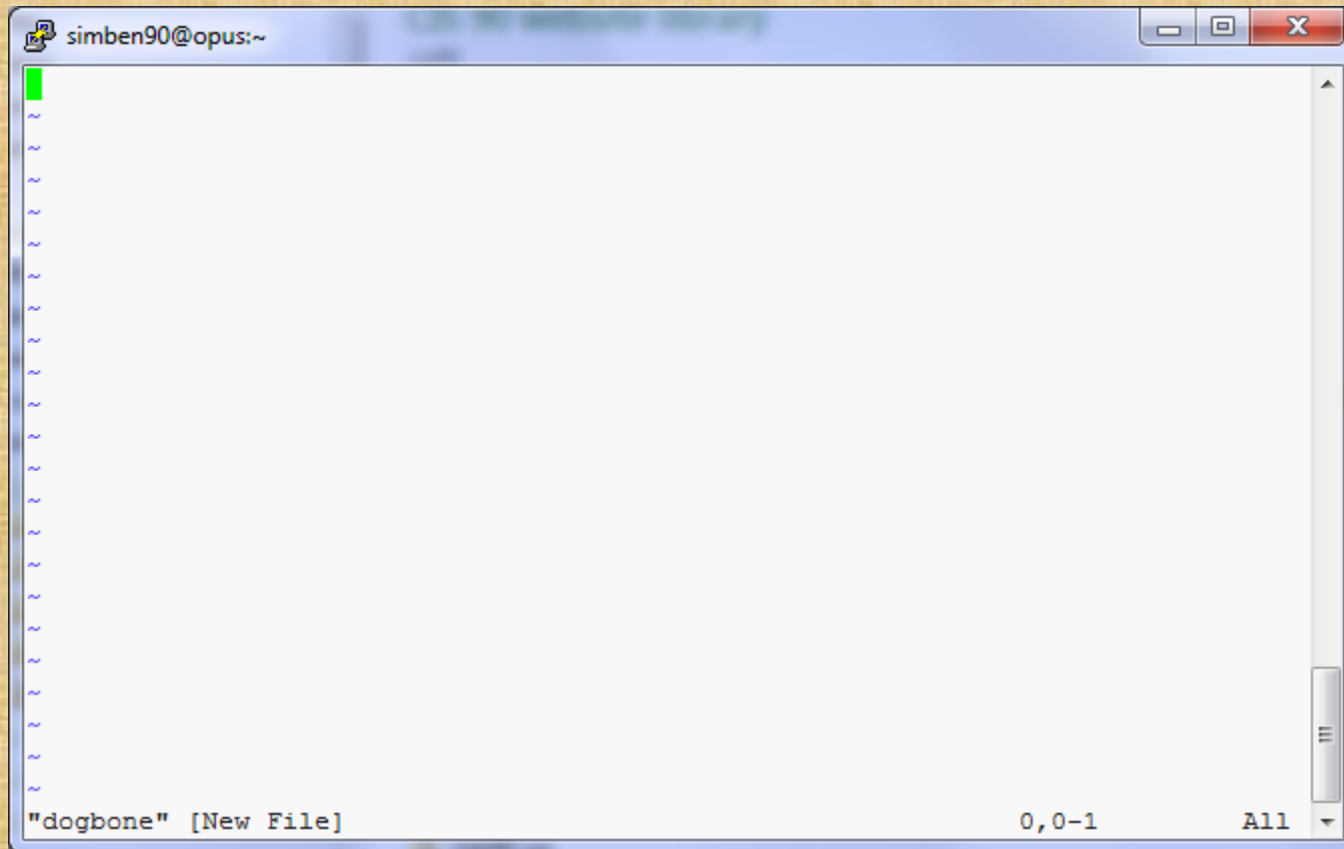
Type this

See this ...



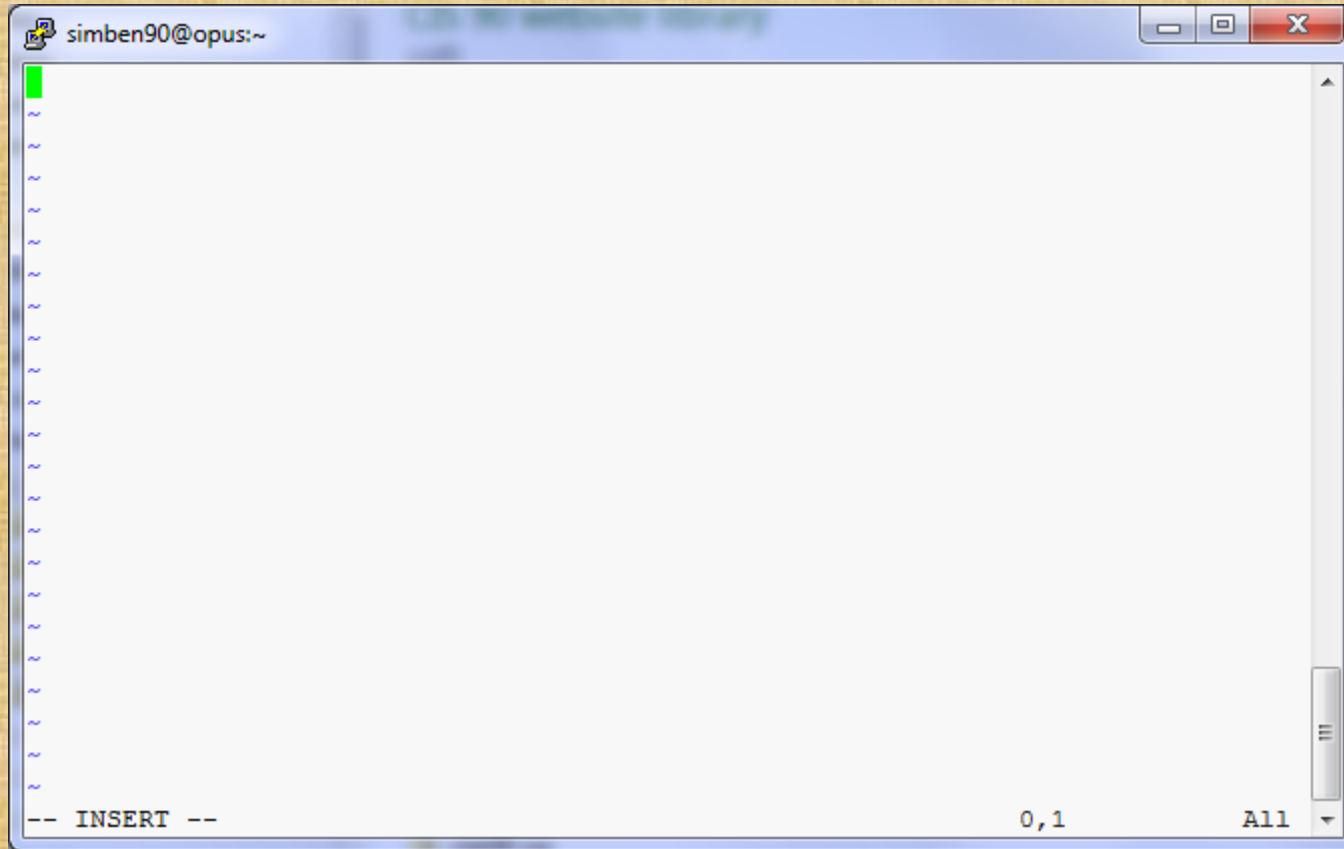
Take your hands OFF THE MOUSE – don't use it in vi!

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



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

See this ...

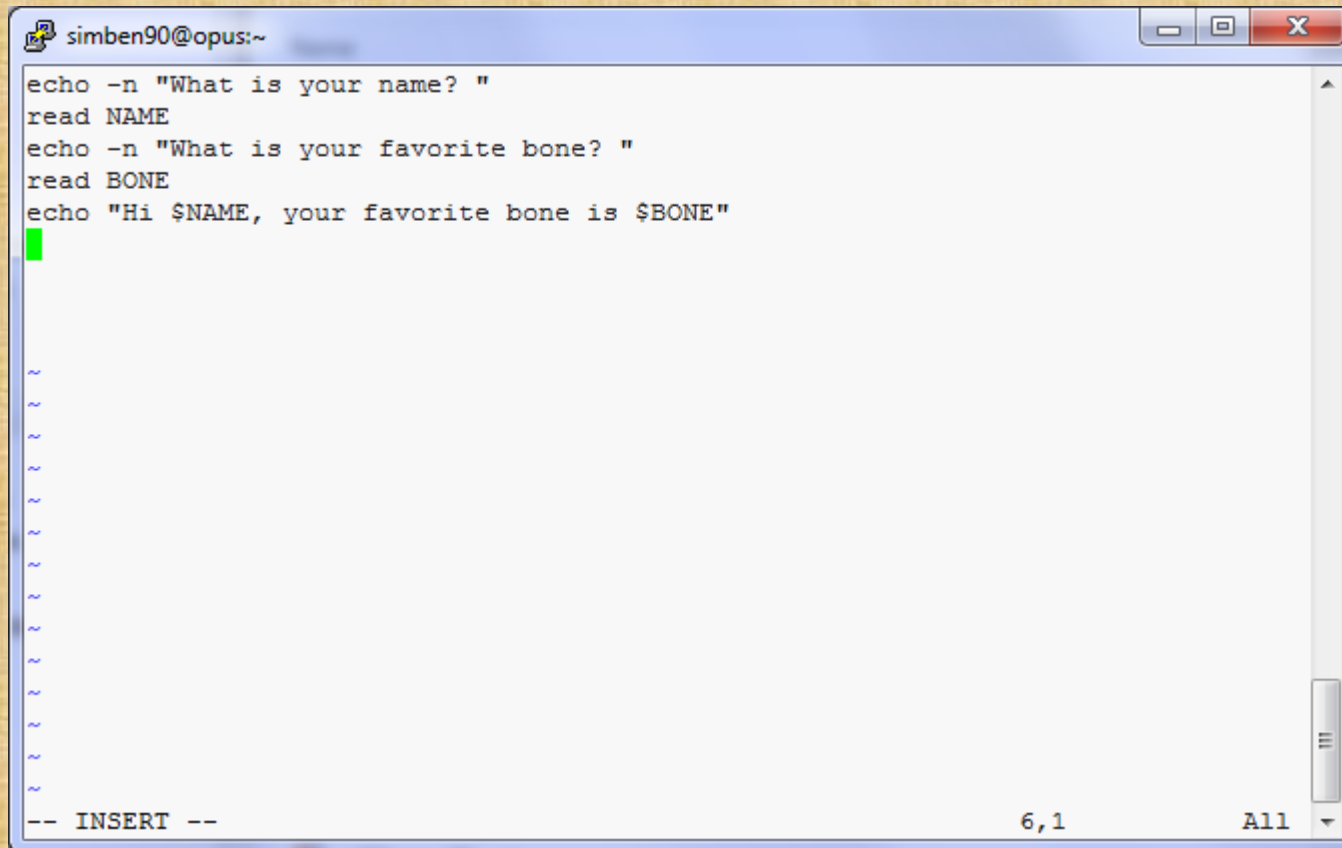


Keep your hands OFF THE MOUSE – don't use it in vi!

[illegible]

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!

A screenshot of a terminal window titled "simben90@opus:~". The window contains a shell script with three lines: `echo -n "What is your name? "`, `read NAME`, `echo -n "What is your favorite bone? "`, `read BONE`, and `echo "Hi $NAME, your favorite bone is $BONE"`. A green cursor is visible on the line following the last command. Below the script, there are several tilde (~) characters representing blank lines. The bottom right corner shows the page number "6,0-1" and the word "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"
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
6,0-1 All
```

106

Type a :

A screenshot of a terminal window titled "simben90@opus:~". The window contains a shell script with three lines: `echo -n "What is your name? "`, `read NAME`, `echo -n "What is your favorite bone? "`, `read BONE`, and `echo "Hi $NAME, your favorite bone is $BONE"`. Below the script, there are several tilde (~) characters representing input or output, followed by a green cursor at the bottom left. The terminal has standard window controls (minimize, maximize, close) in the top right corner.

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

A screenshot of a terminal window titled "simben90@opus:~". The window contains a shell script with four 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"
```


Below the script, there are several tilde (~) characters representing input from the user. At the bottom left, the prompt ":wg" is visible next to a green cursor. The terminal has standard window controls (minimize, maximize, close) in the top right corner.

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!

:

```

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

The Mug of Vi - Mozilla Firefox

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

[See mug text](#)

Done

Big Mug Label - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://nostarch.com/mug_big.htm

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

Cabrillo College Home Page (0 unread) Yahoo! Mail, richsimms Big Mug Label 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

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

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

Done

http://nostarch.com/mug.htm

/bin/mail and vi

```
/home/cis90/simmsben $ mail roddyduk
```

```
Subject: Good bones
```

```
Hey Duke,
```

```
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/simmsben $ mail roddyduk
```

```
Subject: Good bones
```

```
Hey Duke,
```

```
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

[illegible]

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/simmsben $ mail roddyduk
Subject: Good bones
Hey Duke,
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)
.
Cc:
/home/cis90/simmsben $
```

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

/bin/mail and vi

```
/home/cis90/roddyduk $ mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/roddyduk": 1 message 1 unread
>U  1 simmsben@opus.cabrill  Mon Nov 10 20:25  22/782  "Good bones"
& 1
Message 1:
From simmsben@opus.cabrillo.edu  Mon Nov 10 20:25:32 2008
Date: Mon, 10 Nov 2008 20:25:32 -0800
From: Benji Simms <simmsben@opus.cabrillo.edu>
To: roddyduk@opus.cabrillo.edu
Subject: Good bones
```

Hey Duke,
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,
Ben

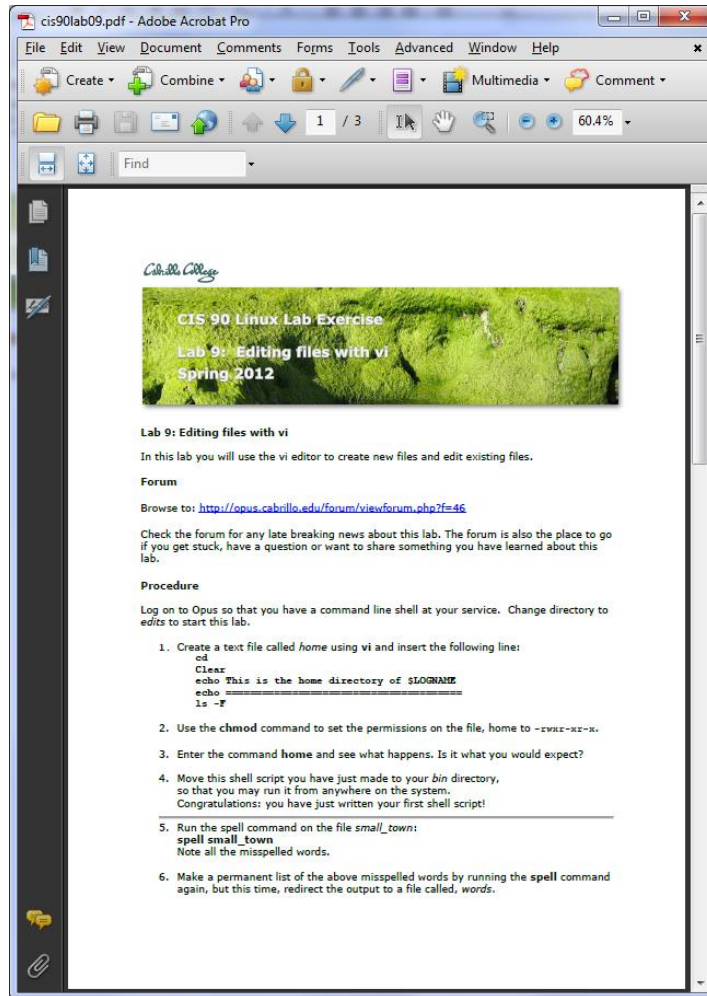
*The message Duke reads has all the
typos fixed.*

Fix an email message before sending

```
/home/cis90/simben/edits $ mail rsimms
Subject: test of vi
sdkfjas;dfkjas;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

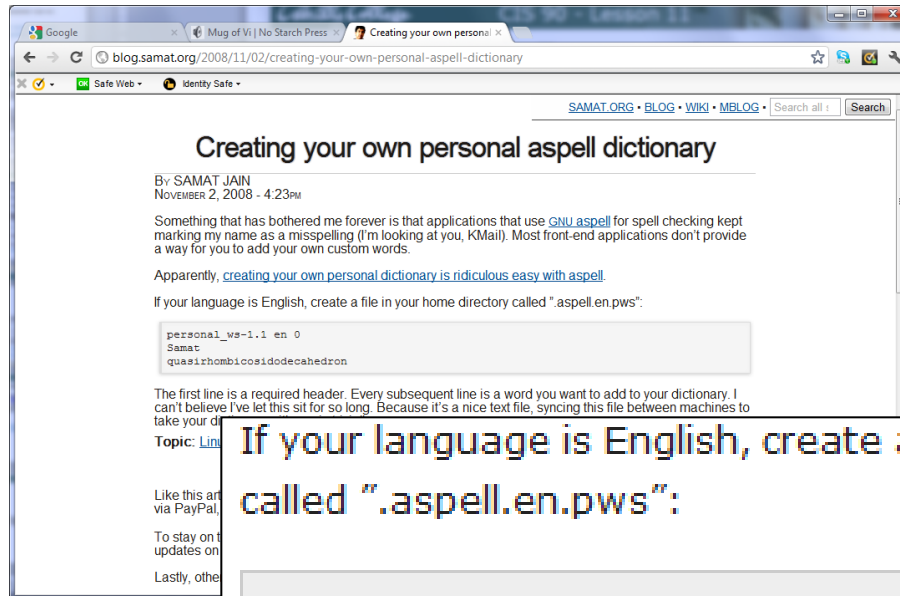
display the help message

-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>; swch = <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>;
swch = <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