CIS 90 - Lesson 12

Lesson Module Checklist

- Slides –
- Properties -
- Flash cards –
- First minute quiz –
- Web calendar summary –
- Web book pages –
- Commands –
- Lab 10 and Final Project –

- CCC Confer wall paper ready –
- riddle file copied to bin directory
- allscripts updated -

- Materials uploaded –
- Backup slides, CCC info, handouts on flash drive -

- Polycom
- Check that room headset is charged – done

CIS 90 - Lesson 11

Instructor: **Rich Simms**

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

Passcode: **761867**

Sean C.

Don

Carlile

Andrew

Sean Fa.

Carter

Sean Fy.

Dajan

Bryn

Rita

Kelly

Ben

Ray

Michael

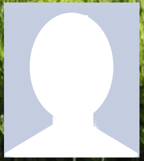
Evan

Josh

Carlos

Gustavo

Jessica


Evie

Jacob

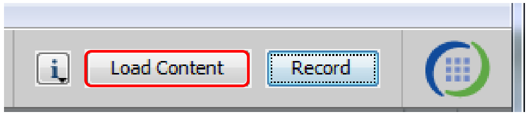
Humberto

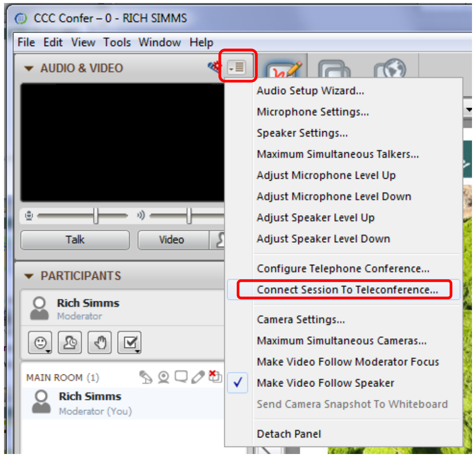
Chad

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

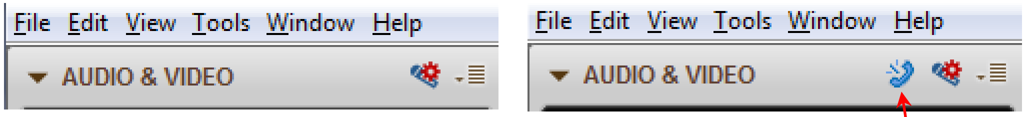
CCC Confer

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



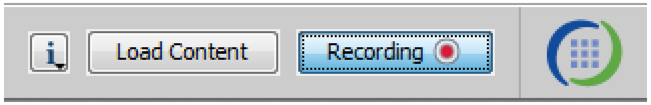

CCC Confer – 0 - RICH SIMMS
File Edit View Tools Window Help
AUDIO & VIDEO
Talk
Video
PARTICIPANTS
Rich Simms
Moderator
MAIN ROOM (1)
Rich Simms
Moderator (You)
Audio Setup Wizard...
Microphone Settings...
Speaker Settings...
Maximum Simultaneous Talkers...
Adjust Microphone Level Up
Adjust Microphone Level Down
Adjust Speaker Level Up
Adjust Speaker Level Down
Configure Telephone Conference...
Connect Session To Teleconference...
Camera Settings...
Maximum Simultaneous Cameras...
Make Video Follow Moderator Focus
✓ Make Video Follow Speaker
Send Camera Snapshot To Whiteboard
Detach Panel

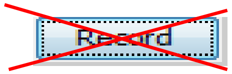
[ ] Connect session to Teleconference


File Edit View Tools Window Help
AUDIO & VIDEO
File Edit View Tools Window Help
AUDIO & VIDEO

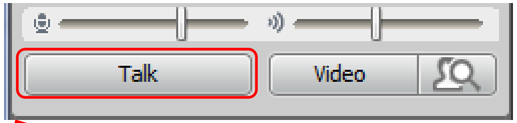
Connected to teleconference

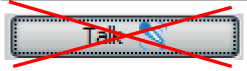
[ ] Is recording on?





[ ] Toggle Talk button to not use Mic

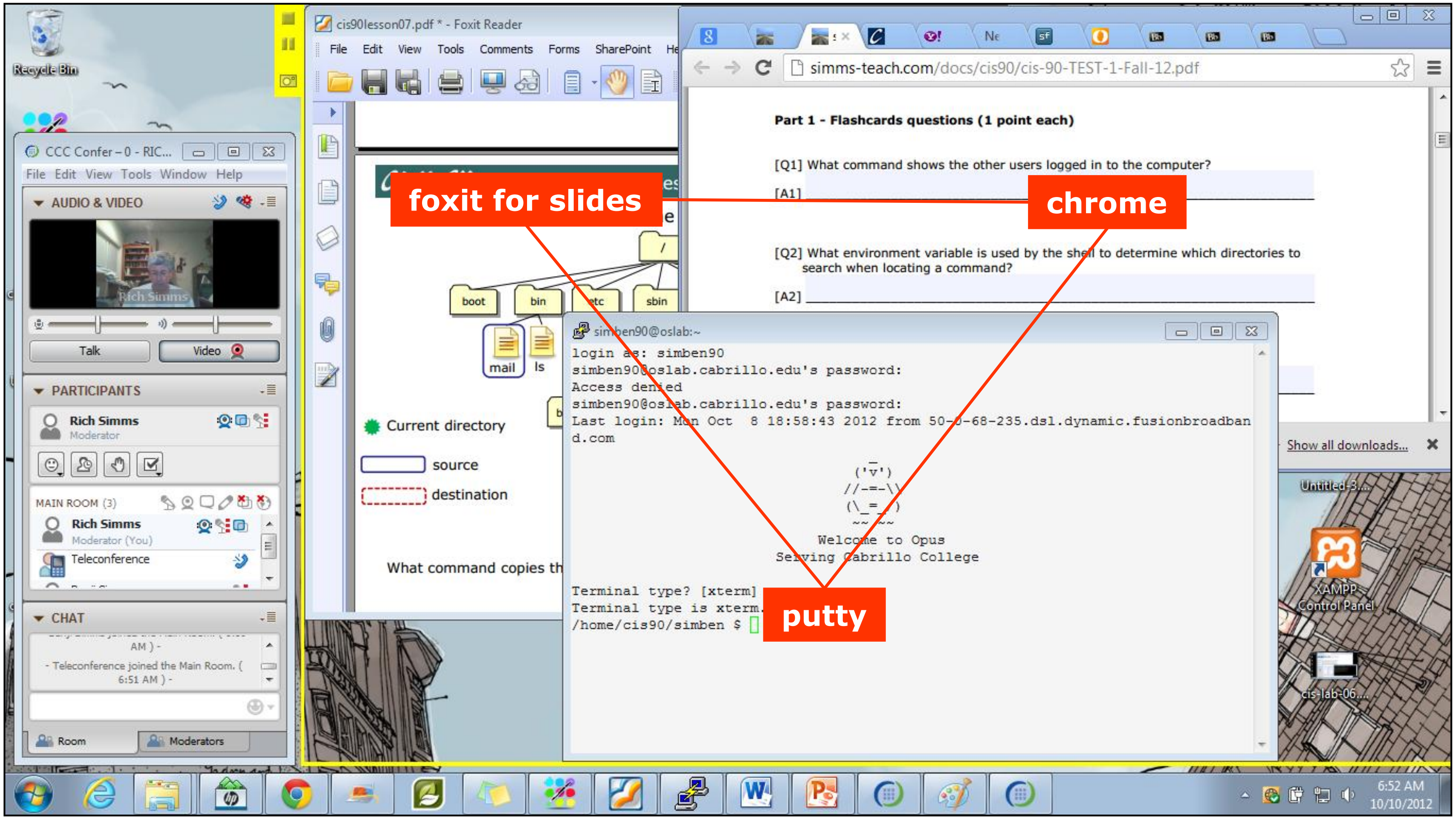




CCC Confer

[ ] Video (webcam) optional

[ ] layout and share apps



foxit for slides

chrome

putty

Part 1 - Flashcards questions (1 point each)

[Q1] What command shows the other users logged in to the computer?

[A1] ______________________________

[Q2] What environment variable is used by the shell to determine which directories to search when locating a command?

[A2] ______________________________

```
login as: simben90
simben90@oslab.cabrillo.edu's password:
Access denied
simben90@oslab.cabrillo.edu's password:
Last login: Mon Oct  8 18:58:43 2012 from 50-0-68-235.dsl.dynamic.fusionbroadban
d.com

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

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

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)

The Shell Environment

Objectives	Agenda
• Be able to set, view and unset shell variables • Describe the difference between the set and env commands • Explain the importance of the export command. • Describe three actions that are handled by the .bash_profile file • Define user-defined aliases • Explain the . (dot) command and the exec command.	• Quiz • Housekeeping • Spell checking • vi and /bin/mail • Review pathnames • Final project prep • Variables • The shell environment • Aliases • .bash_profile • .bashrc

Questions

Questions?

- vi
- lab 9
- previous material

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

- Francis Bacon

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

- Mahatma Gandhi

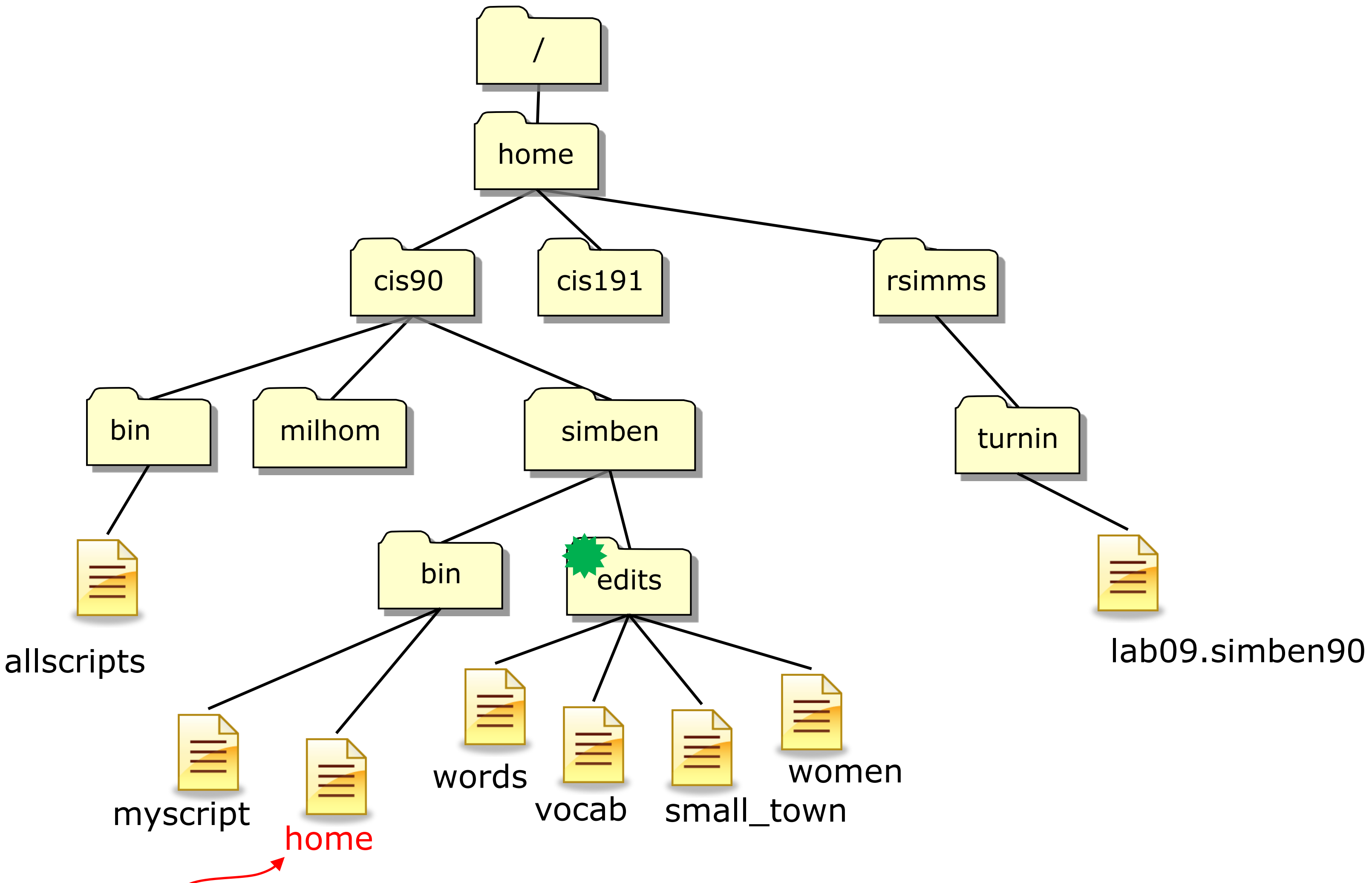
He who asks a question is a fool for five minutes; he who does not ask a question remains a fool forever.

- Chinese Proverb

Submitting Lab 9 & PATHNAMES!

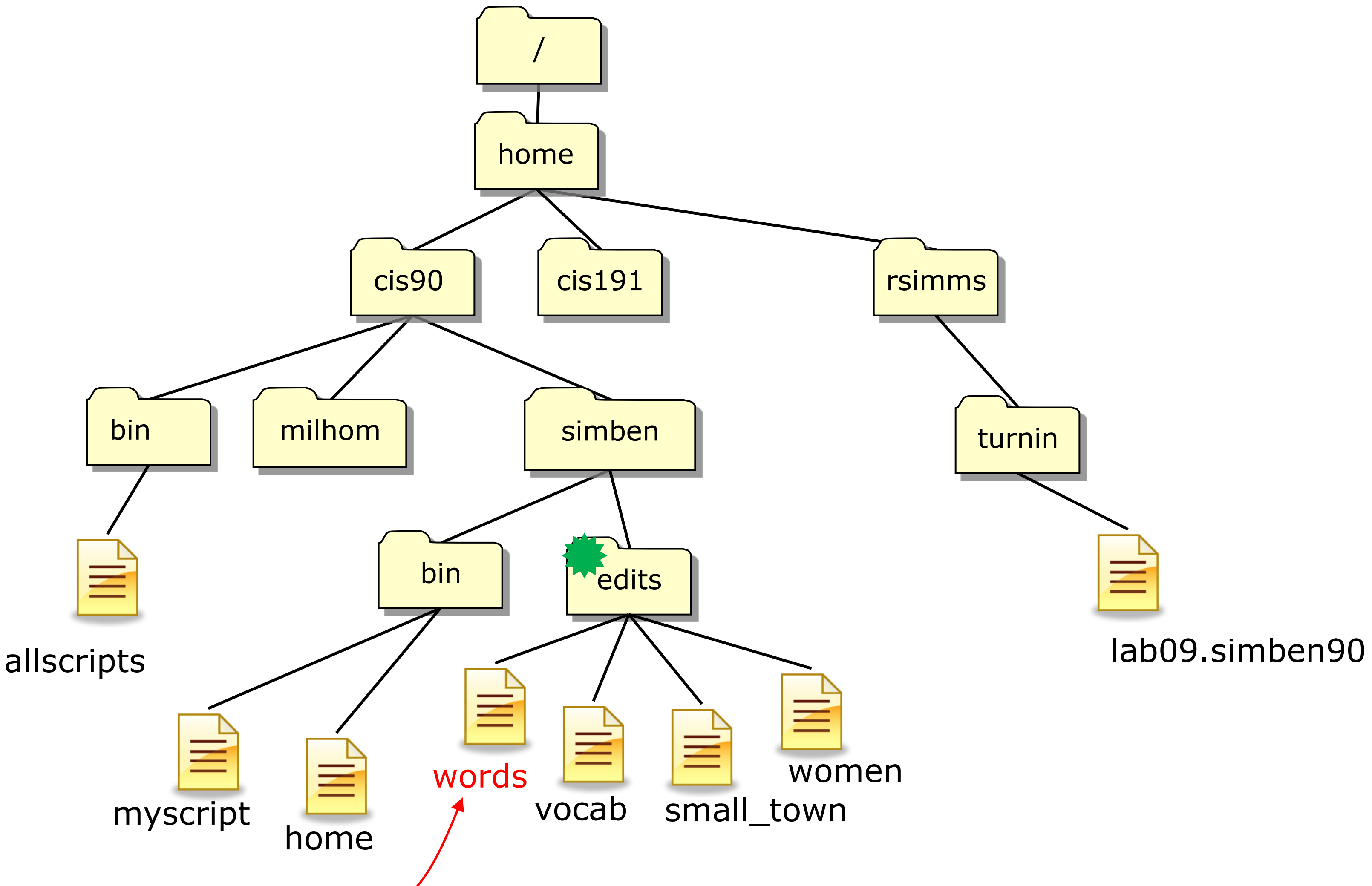
REMINDER

- You must **ALWAYS** use **VALID PATHNAMES** when specifying files as **ARGUMENTS** on a command.
- Pathnames can be relative or absolute.
- A common mistake in the past on Lab 9 is to ignore error messages and not submit all the files requested.



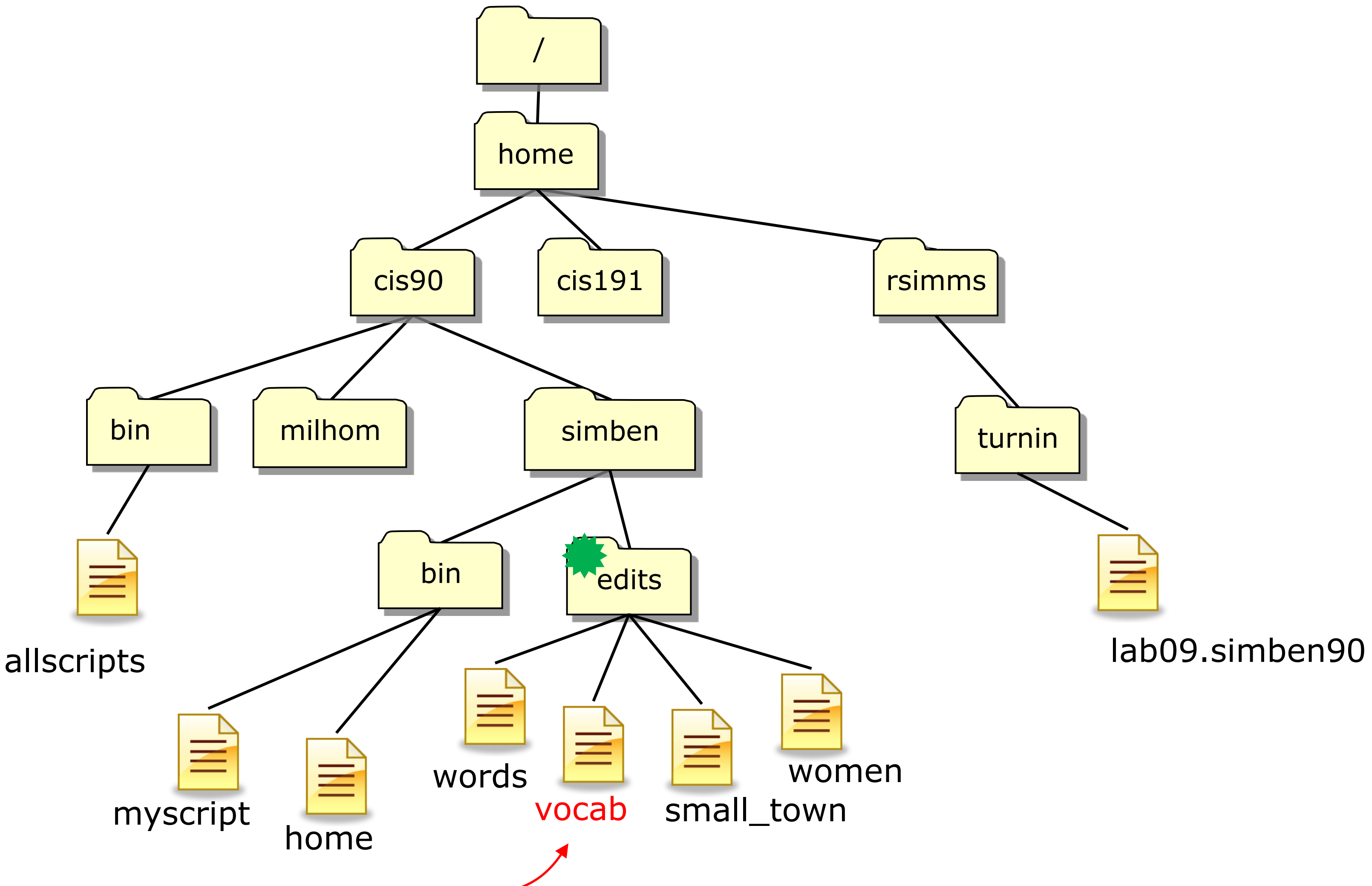
```
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME
```

relative pathname to home in the bin directory



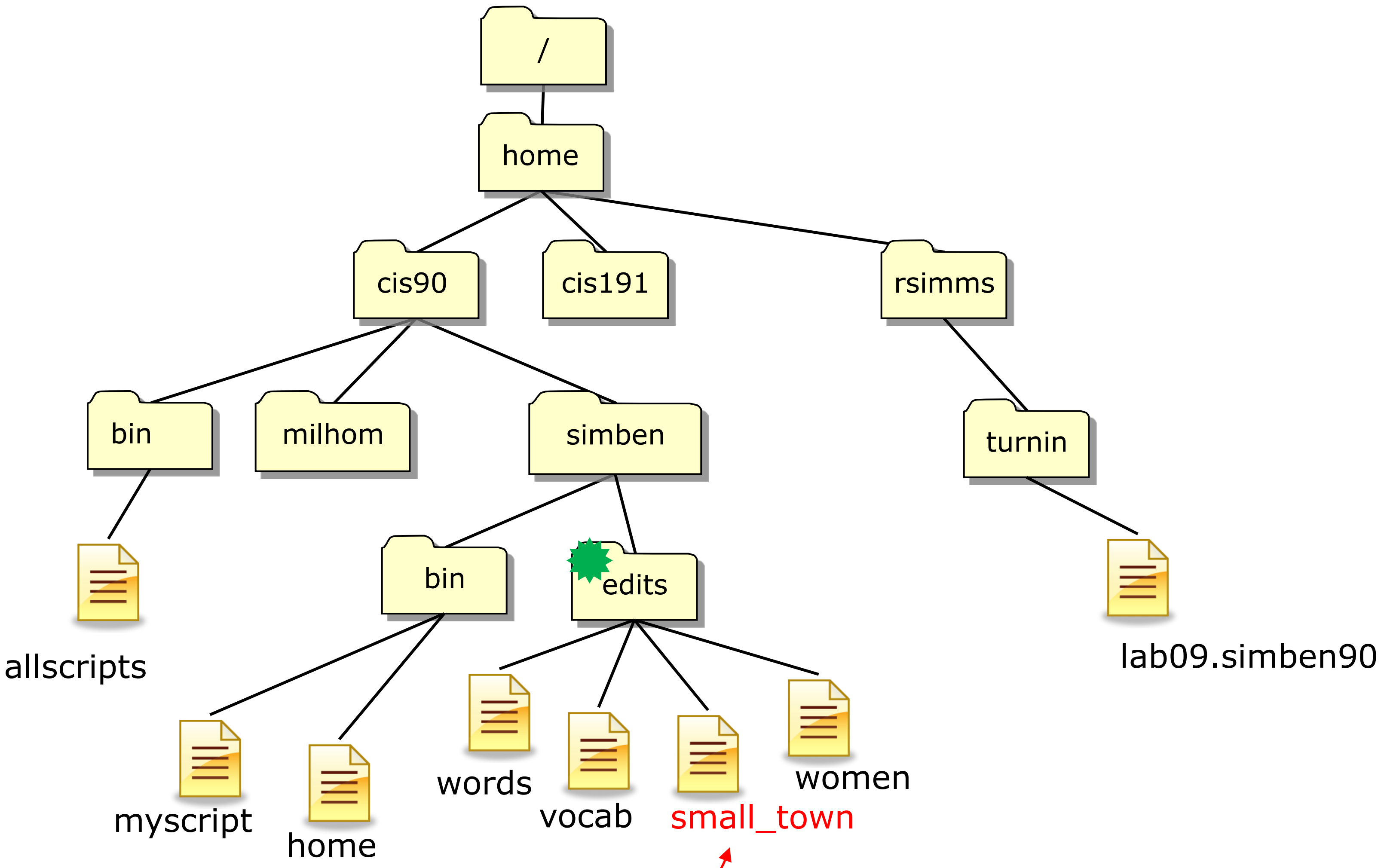
```
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME
```

relative pathname



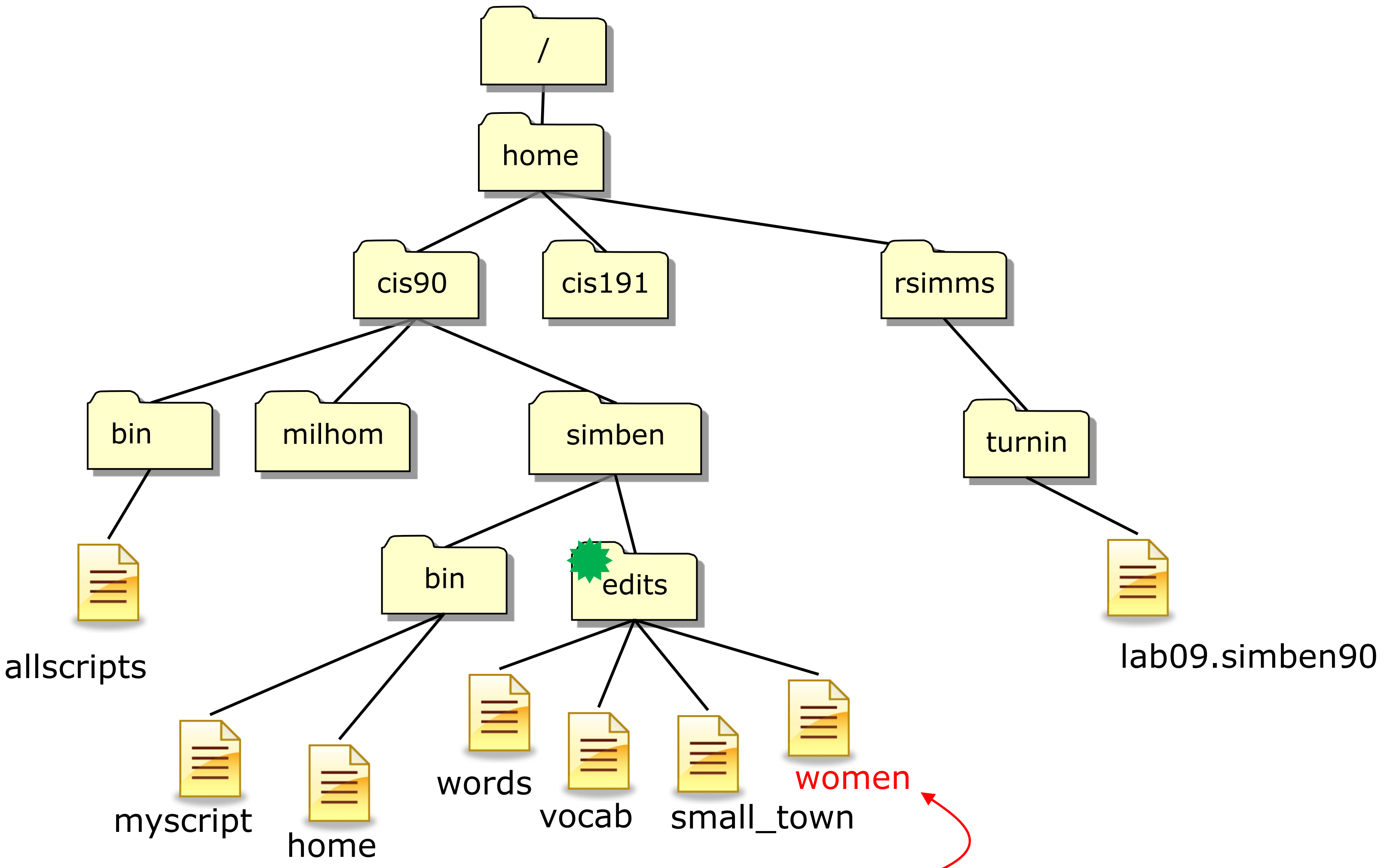
```
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME
```

relative pathname



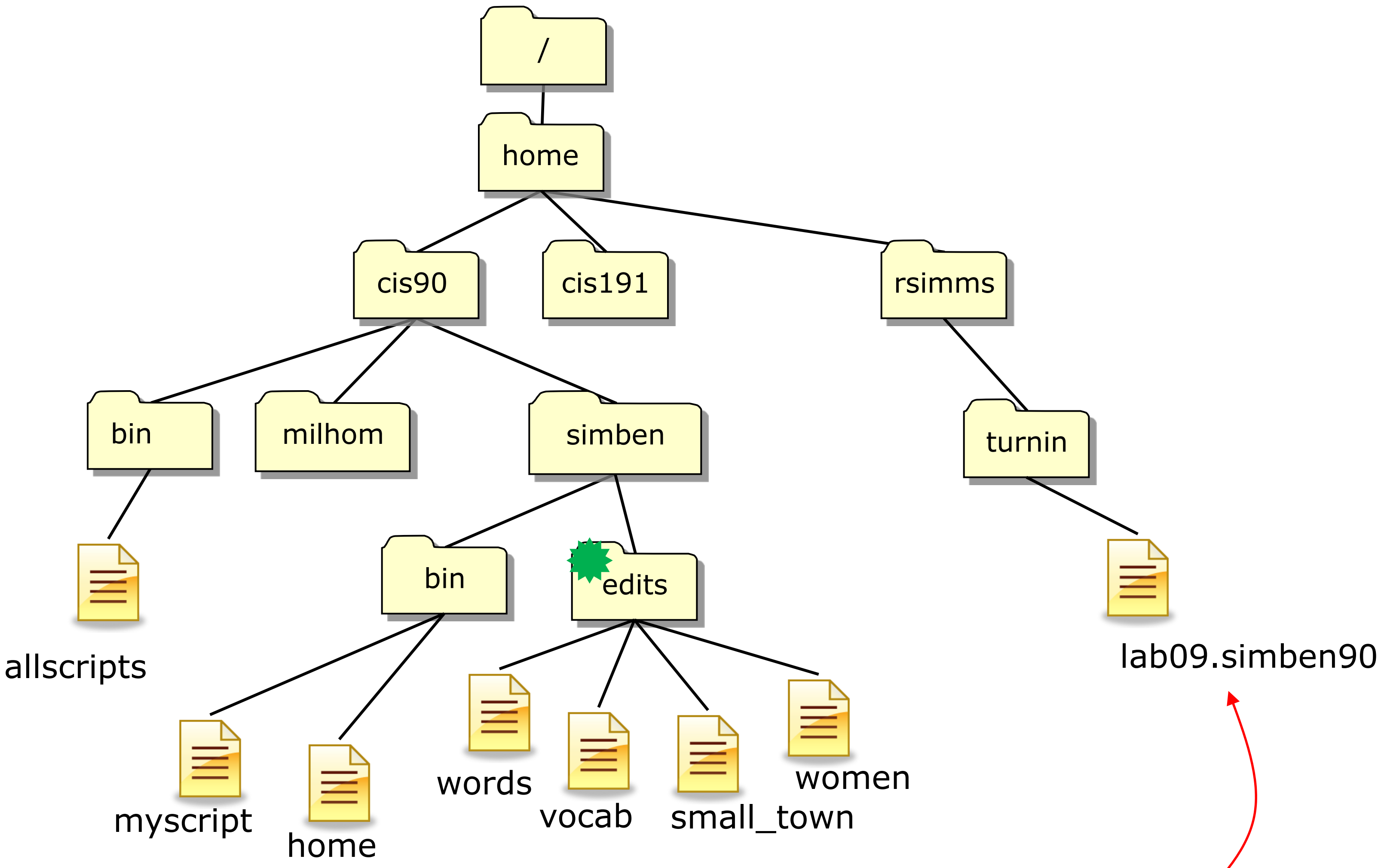
```
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME
```

relative pathname



```
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME
```

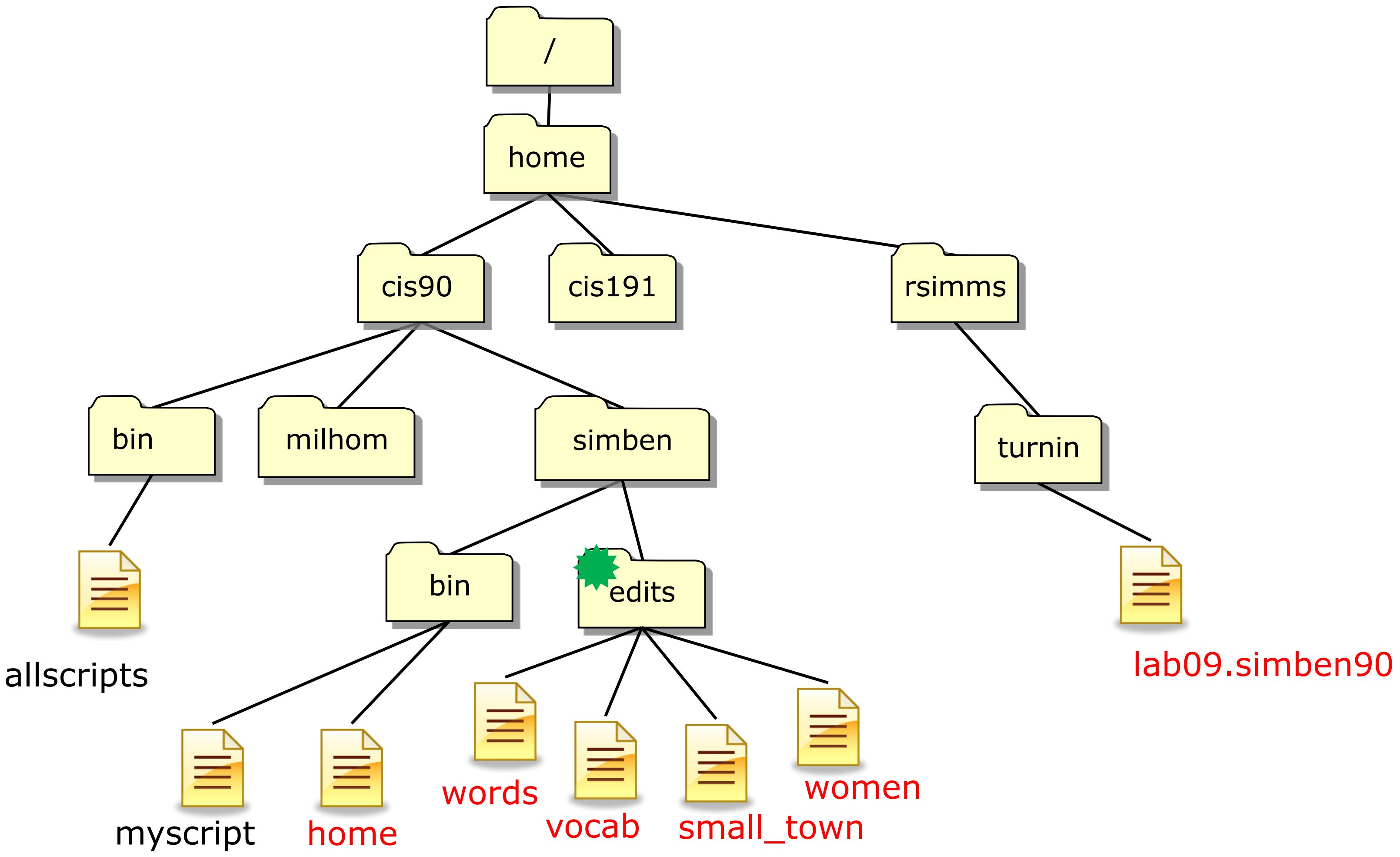
relative pathname



```
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME
```

absolute pathname

CIS 90 - Lesson 12



Doing same thing in two steps

```
cat ../bin/home words vocab small_town woman > lab09
cp lab09 /home/rsimms/turnin/lab09.$LOGNAME
```

Housekeeping

Previous material and assignment

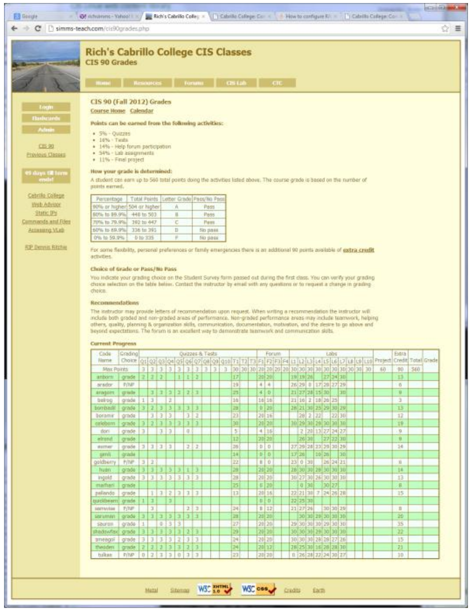
1. Lab 9 due 11:59PM tonight

2. Five posts due 11:59PM tonight

Reminder: Only posts between in the CIS 90 forum between 10/18 and 11/14 (inclusive) are counted.

Managing your grade

Use the web page



Rich's Cabrillo College CIS Classes

CIS 90 Grades

CIS 90 (Fall 2012) Grades

Course Home Calendar

Points can be earned from the following activities:

- 5% - Quizzes
- 16% - Tests
- 14% - Help forum participation
- 54% - Lab assignments
- 11% - Final project

How your grade is determined:

A student can earn up to 560 total points doing the activities listed above. The course grade is based on the number of points earned.

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

For some flexibility, personal preferences or family emergencies there is an additional 90 points available of extra credit activities.

Choice of Grade or Pass/No Pass

You indicate your grading choice on the Student Survey form passed out during the first class. You can verify your grading choice selection on the table below. Contact the instructor by email with any questions or to request a change in grading choice.

Recommendations

The instructor may provide letters of recommendation upon request. When writing a recommendation the instructor will include both graded and non-graded areas of performance. Non-graded performance areas may include teamwork, helping others, quality, planning & organization skills, communication, documentation, motivation, and the desire to go above and beyond expectations. The forum is an excellent way to demonstrate teamwork and communication skills.

Current Progress

Code Name	Grading Choice	Quizzes & Tests													Forum				Labs										Project	Extra Credit	Total	Grade
		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	T1	T2	T3	F1	F2	F3	F4	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10				
Max Points		3	3	3	3	3	3	3	3	3	3	30	30	30	20	20	20	20	30	30	30	30	30	30	30	30	30	30	60	90	560	
anborn	grade	2	2	2		1	1	2				17			20	20			19	19	26		27	24	30					13		
arador	P/NP											19			4	4			26	29	0	17	28	27	29					6		
aragorn	grade		3	3	3	2	2	3				25			4	0			21	27	28	15	30		30					9		
balrog	grade	1	3		2							16			16	16			21	16	2	18	26	25						3		
bombadil	grade	3	2	3	3	3	3	3				28			0	20			28	21	30	25	29	30	29					13		
boromir	grade		3	3	3		3	2				23			20	16				28	2	22		22	30					12		
celeborn	grade	3	2	3	3	3	3	3				30			20	20			30	29	30	29	30	30	30					19		
dori	grade	3		3	3		0					5			4	16				2	20	13	27	24	27					9		
elrond	grade											12			20	20				26	30		27	22	30					9		
eomer	grade	3	3	3	3		2	2				26			0	0			27	29	28	23	29	30	29					14		
gimli	grade											14			0	0			17	28		10	28		30							
goldberry	P/NP	3	2									22			8	0			23	0	30		26	24	21					6		
huan	grade	3	3	3	3	3	1	3				28			20	20			28	30	30	28	30	30	30					14		
ingold	grade	3	3	3	3	3	3	3				28			20	20			30	27	30	26	30	30	30					13		
marhari	grade											25			8	20				0	30		30	27						8		
pallando	grade		1	3	2	3	3	3				13			20	16			22	21	30	7	24	26	28					15		
quickbeam	grade	1	3		3										0	0			22	25	30											
samwise	P/NP		3				2	3				24			8	12			21	27	26		30	30	29					8		
saruman	grade	3	3	3	3	3	3	3				28			20	20				30	30	28	30	30	30					20		
sauron	grade	1		0	3	3						27			20	20			29	30	30	30	29	30	30					35		
shadowfax	grade	3	3	3	3	3	2	3				29			20	20			30	30	30	29	30	30	30					22		
smeagol	grade	3	3	3	3	2	3	3				24			20	20			30	30	30	28	28	27	26					15		
theoden	grade	2	2	2	3	3	2	3				24			20	12			28	25	30	16	28	28	30					21		
tulkas	P/NP	0	2	3	3	0	3	3				23			20	20			0	26	28	22	24	30	27					10		

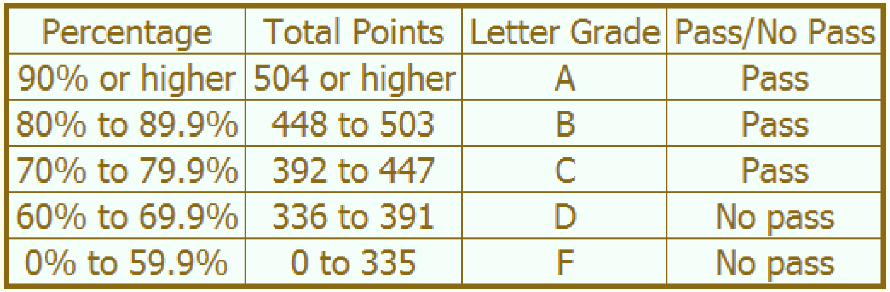
http://simms-teach.com/cis90grades.php

Use Jesse's checkgrades script

```
anborn: 71% (262 of 364 points)
arador: 54% (198 of 364 points)
aragorn: 67% (245 of 364 points)
balrog: 46% (168 of 364 points)
bombadil: 91% (332 of 364 points)
boromir: 65% (238 of 364 points)
celeborn: 104% (380 of 364 points)
dori: 52% (191 of 364 points)
elrond: 65% (237 of 364 points)
eomer: 84% (307 of 364 points)
gimli: 34% (125 of 364 points)
goldberry: 59% (218 of 364 points)
huan: 108% (394 of 364 points)
ingold: 97% (354 of 364 points)
marhari: 59% (215 of 364 points)
pallando: 76% (278 of 364 points)
samwise: 72% (265 of 364 points)
saruman: 96% (353 of 364 points)
sauron: 103% (376 of 364 points)
shadowfax: 105% (385 of 364 points)
smeagol: 96% (351 of 364 points)
theoden: 93% (340 of 364 points)
tulkas: 80% (294 of 364 points)
```

As of November 10, 2012

Managing your grade



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

Points gone by

- 8 quizzes - 24 points
- 2 tests - 60 points
- 2 forum periods - 40 points
- 8 labs - 240 points

364 points

Points yet to earn

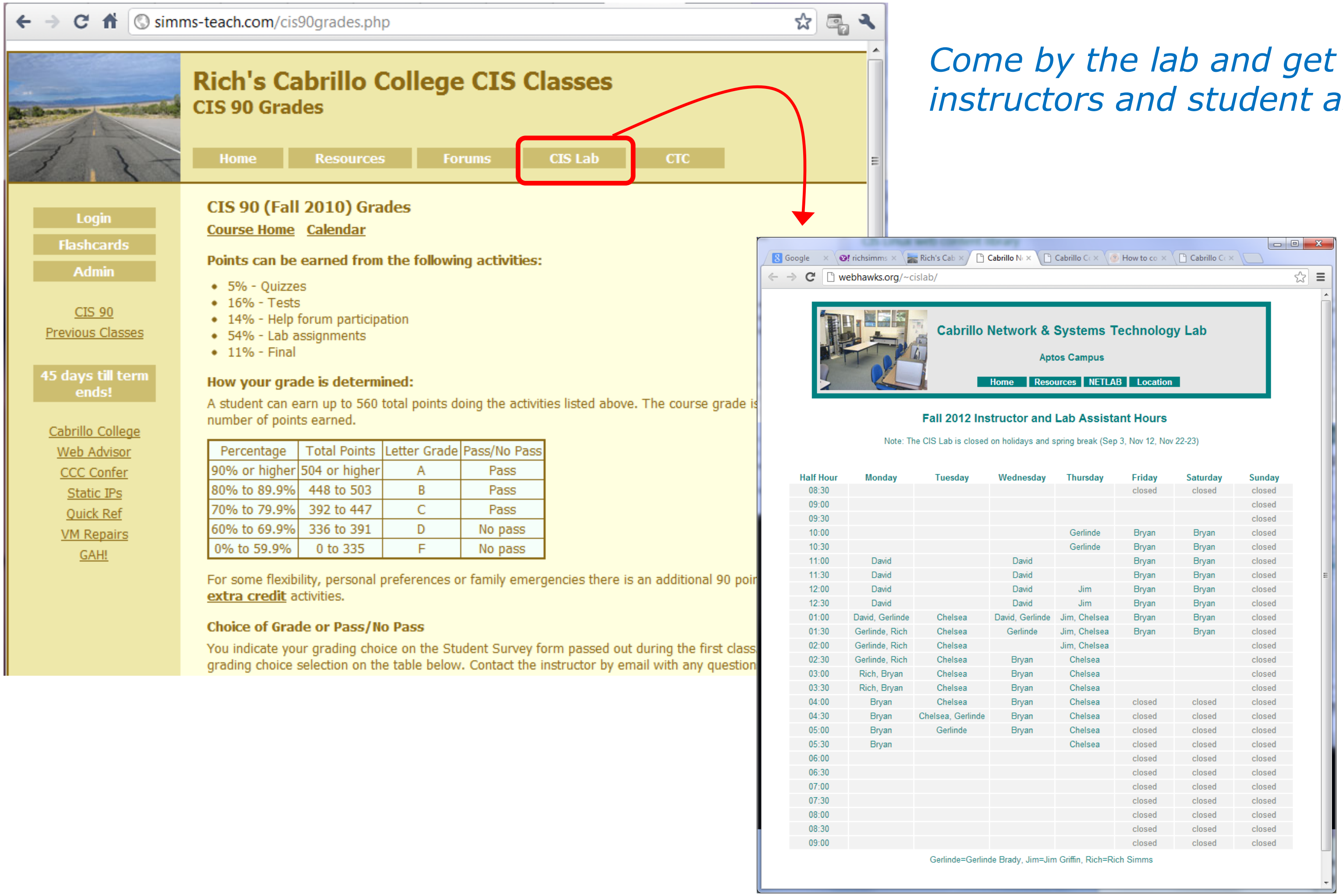
- 2 quizzes - 6 points
- 1 test - 30 points
- 2 forum periods - 40 points
- 2 labs - 60 points
- 1 final project - 60 points

196 points

- Plus extra credit - up to 90 points

Managing your grade

Getting extra help for CIS 90



simms-teach.com/cis90grades.php

Rich's Cabrillo College CIS Classes

CIS 90 Grades

CIS 90
Previous Classes

45 days till term ends!

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

CIS 90 (Fall 2010) Grades

Course Home Calendar

Points can be earned from the following activities:

- 5% - Quizzes
- 16% - Tests
- 14% - Help forum participation
- 54% - Lab assignments
- 11% - Final

How your grade is determined:

A student can earn up to 560 total points doing the activities listed above. The course grade is number of points earned.

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

For some flexibility, personal preferences or family emergencies there is an additional 90 poi extra credit activities.

Choice of Grade or Pass/No Pass

You indicate your grading choice on the Student Survey form passed out during the first class grading choice selection on the table below. Contact the instructor by email with any question

webhawks.org/~cislab/

Cabrillo Network & Systems Technology Lab

Aptos Campus

Fall 2012 Instructor and Lab Assistant Hours

Note: The CIS Lab is closed on holidays and spring break (Sep 3, Nov 12, Nov 22-23)

Half Hour	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
08:30					closed	closed	closed
09:00							closed
09:30							closed
10:00				Gerlinde	Bryan	Bryan	closed
10:30				Gerlinde	Bryan	Bryan	closed
11:00	David		David		Bryan	Bryan	closed
11:30	David		David		Bryan	Bryan	closed
12:00	David		David	Jim	Bryan	Bryan	closed
12:30	David		David	Jim	Bryan	Bryan	closed
01:00	David, Gerlinde	Chelsea	David, Gerlinde	Jim, Chelsea	Bryan	Bryan	closed
01:30	Gerlinde, Rich	Chelsea	Gerlinde	Jim, Chelsea	Bryan	Bryan	closed
02:00	Gerlinde, Rich	Chelsea		Jim, Chelsea			closed
02:30	Gerlinde, Rich	Chelsea	Bryan	Chelsea			closed
03:00	Rich, Bryan	Chelsea	Bryan	Chelsea			closed
03:30	Rich, Bryan	Chelsea	Bryan	Chelsea			closed
04:00	Bryan	Chelsea	Bryan	Chelsea	closed	closed	closed
04:30	Bryan	Chelsea, Gerlinde	Bryan	Chelsea	closed	closed	closed
05:00	Bryan	Gerlinde	Bryan	Chelsea	closed	closed	closed
05:30	Bryan			Chelsea	closed	closed	closed
06:00					closed	closed	closed
06:30					closed	closed	closed
07:00					closed	closed	closed
07:30					closed	closed	closed
08:00					closed	closed	closed
08:30					closed	closed	closed
09:00					closed	closed	closed

Gerlinde=Gerlinde Brady, Jim=Jim Griffin, Rich=Rich Simms

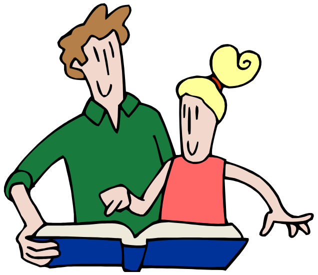
Come by the lab and get help from instructors and student assistants

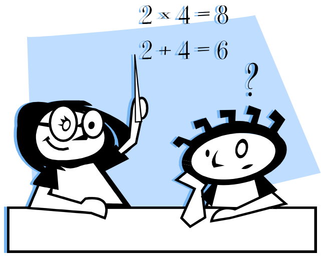
Managing your grade

Getting extra help for CIS 90

- Rich’s Office Hours Wed 4:20-5:10pm in Room 2501 (right after class) or TBA (contact me)

- Ask questions on the Forum at:
  http://opus.cabrillo.edu/forum/





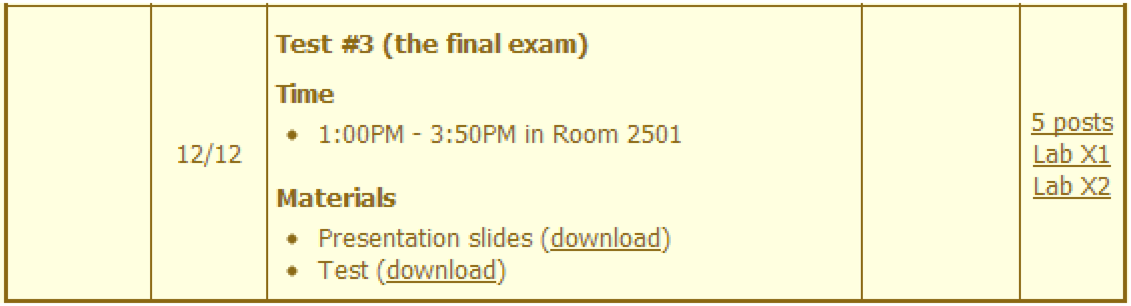
Final Exam

Can **not** be taken online using CCC Confer

It will be held in room 2501 on Wednesday, Dec 12th from 1:00 to 3:50PM

If you know you can't make this date you will need to contact the instructor, in advance, to arrange an exam **EARLIER** in the week.

No makeups after the Wednesday exam



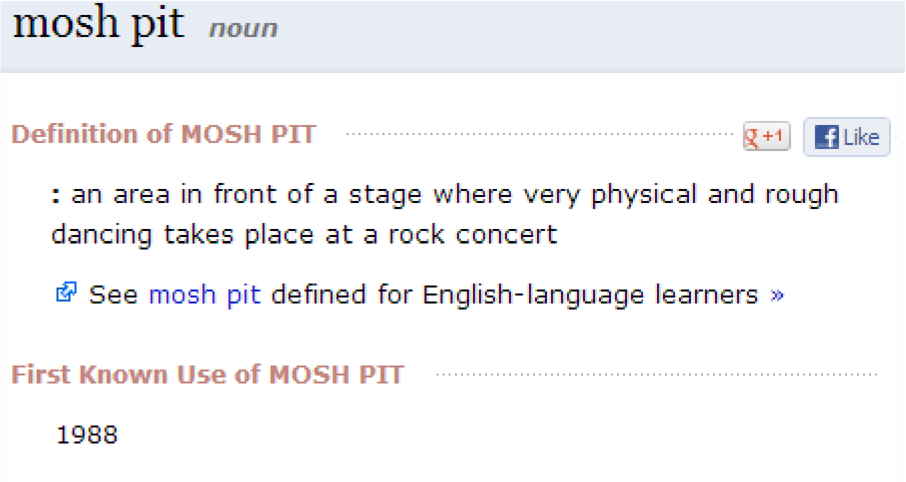
	12/12	**Test #3 (the final exam)** **Time** • 1:00PM - 3:50PM in Room 2501 **Materials** • Presentation slides (download) • Test (download)		5 posts Lab X1 Lab X2

Ayshire
moshpit
and personal
dictionaries

moshpit?

Merriam-Webster

m-w.com



mosh pit *noun*

Definition of MOSH PIT

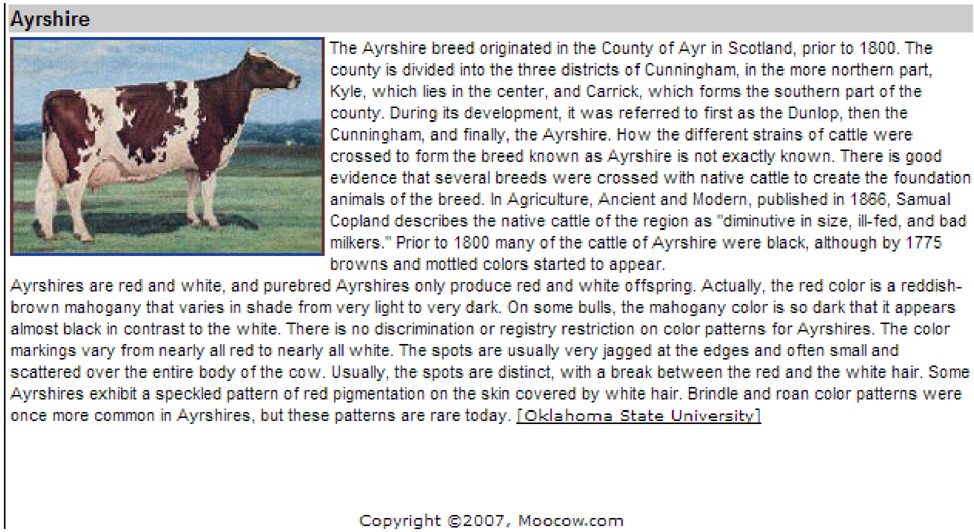
: an area in front of a stage where very physical and rough dancing takes place at a rock concert

See mosh pit defined for English-language learners »

First Known Use of MOSH PIT

1988

Ayshire?



Ayrshire

The Ayrshire breed originated in the County of Ayr in Scotland, prior to 1800. The county is divided into the three districts of Cunningham, in the more northern part, Kyle, which lies in the center, and Carrick, which forms the southern part of the county. During its development, it was referred to first as the Dunlop, then the Cunningham, and finally, the Ayrshire. How the different strains of cattle were crossed to form the breed known as Ayrshire is not exactly known. There is good evidence that several breeds were crossed with native cattle to create the foundation animals of the breed. In Agriculture, Ancient and Modern, published in 1866, Samual Copland describes the native cattle of the region as "diminutive in size, ill-fed, and bad milkers." Prior to 1800 many of the cattle of Ayrshire were black, although by 1775 browns and mottled colors started to appear.

Ayrshires are red and white, and purebred Ayrshires only produce red and white offspring. Actually, the red color is a reddish-brown mahogany that varies in shade from very light to very dark. On some bulls, the mahogany color is so dark that it appears almost black in contrast to the white. There is no discrimination or registry restriction on color patterns for Ayrshires. The color markings vary from nearly all red to nearly all white. The spots are usually very jagged at the edges and often small and scattered over the entire body of the cow. Usually, the spots are distinct, with a break between the red and the white hair. Some Ayrshires exhibit a speckled pattern of red pigmentation on the skin covered by white hair. Brindle and roan color patterns were once more common in Ayrshires, but these patterns are rare today. [Oklahoma State University]

Adding words to the UNIX dictionary

```
/home/cis90/simben $ echo Benji lives in Soquel > address
/home/cis90/simben $ cat address
Benji lives in Soquel
/home/cis90/simben $ spell address
Soquel

/home/cis90/simben $ echo "personal_ws-1.1 en 0" > .aspell.en.pws
/home/cis90/simben $ echo Soquel >> .aspell.en.pws
/home/cis90/simben $ spell address
/home/cis90/simben $
```

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

This is FYI and not required for Lab 9

Make a Personal Dictionary

```
cd
echo "personal_ws-1.1 en 0" >  .aspell.en.pws
echo "moshpit" >> .aspell.en.pws
echo "Ayshire" >> .aspell.en.pws
cat .aspell.en.pws

cd edits/
spell small_town
```

Note: You should still leave the two words Ayshire and moshpit (or mashpit) in the file words when you submit Lab 9

Lab 9
Subtle Things

(but very important)

In Lab 9 you create a script named home in your edits/ directory

```
/home/cis90/simben/edits $ cat home
cd
clear
echo This is the home directory of $LOGNAME
echo =======================================
ls -F
```

WHY?

From your home directory

```
/home/cis90/simben $ home
-bash: home: command not found
```

Move home from edits/ to bin/

```
/home/cis90/simben $ mv edits/home bin/
```

From your home directory, the script does not work until it is moved from edits/ into bin/

Again, from your home directory

```
/home/cis90/simben $ home
This is the home directory of simben90
====================================
bag/          etc/               lab07        monster2   snap2
bigfile       expressions        lab07.bak    monster3   tempdir/
```

< snipped >

```
edits/        lab05.graded       mistakes     results
errors        lab06.graded       monster1     snap1*
```

QUESTION: Why does the script work after moving it from the edits/ directory to the bin/ directory?

Remember!

```
 /home/cis90/simben $ home
-bash: home: command not found
```

"Step 3 – Search" of the Shell's six steps.

If the shell is unable to locate the command on the path it prints "command not found"

Because

```
/home/cis90/simben $ echo $PATH
/usr/lib/qt-
3.3/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/s
bin:/home/cis90/simben/../bin:/home/cis90/simben/bin:.
```

By moving the script into the user's local bin directory, which is on the path, the command can now be run from anywhere on the system

vi and /bin/mail (review)

Best Practice - /bin/mail and vi

```
/home/cis90/simben $ mail rodduk90
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
```

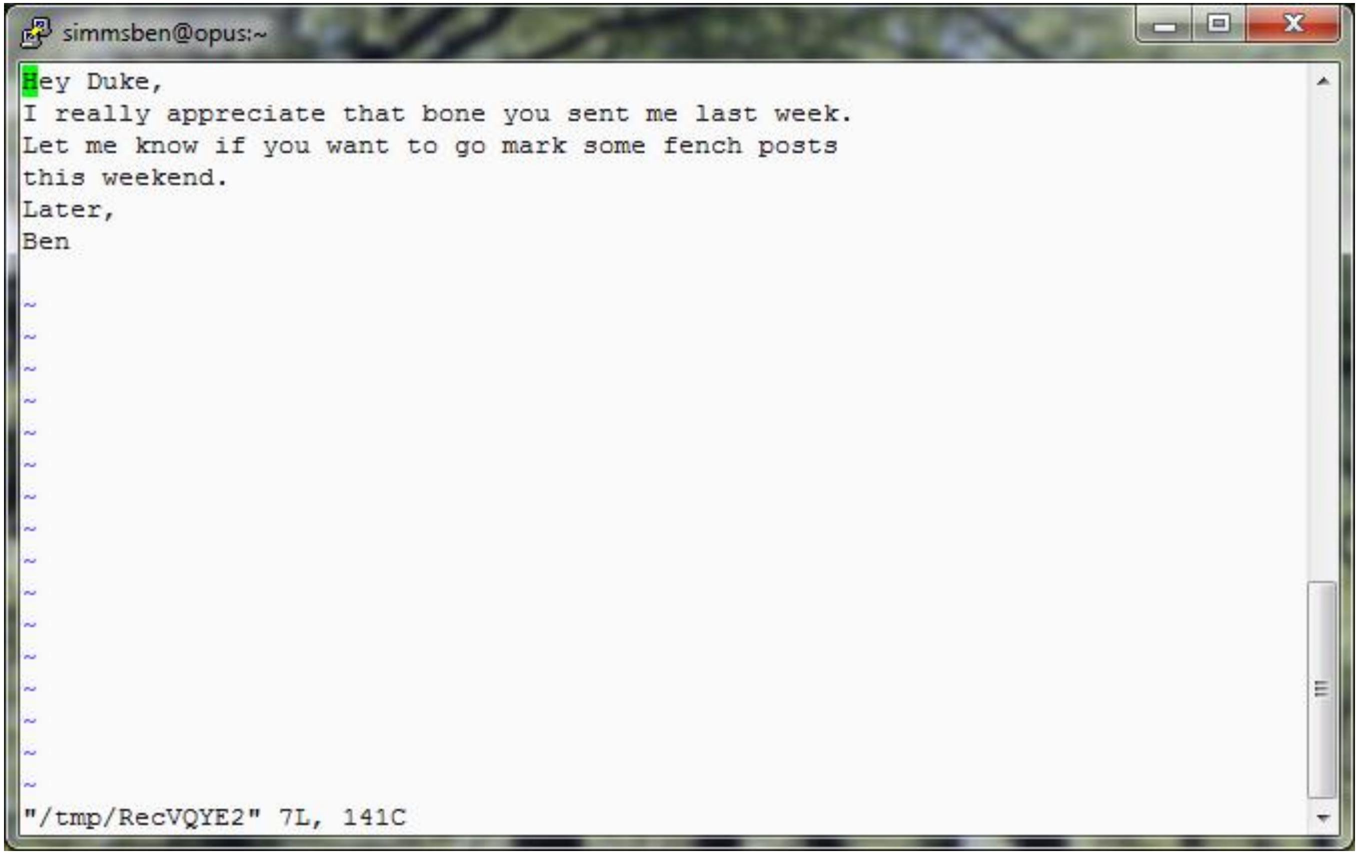
You are composing a message and you spot some typos …
CRUD … what can you do?

/bin/mail and vi

```
/home/cis90/simben $ mail rodduk90
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



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

~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/tmp/RecVQYE2" 7L, 141C
```

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 rodduk90
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/simben $
```

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

/bin/mail and vi

```
/home/cis90/rodduk $ mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/rodduk90": 1 message 1 unread
>U  1 simben90@opus.cabril  Mon Nov 10 20:25  22/782   "Good bones"
& 1
Message 1:
From simben90@opus.cabrillo.edu  Mon Nov 10 20:25:32 2008
Date: Mon, 10 Nov 2008 20:25:32 -0800
From: Benji Simms <simben90@opus.cabrillo.edu>
To: rodduk90@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.

/bin/mail and vi

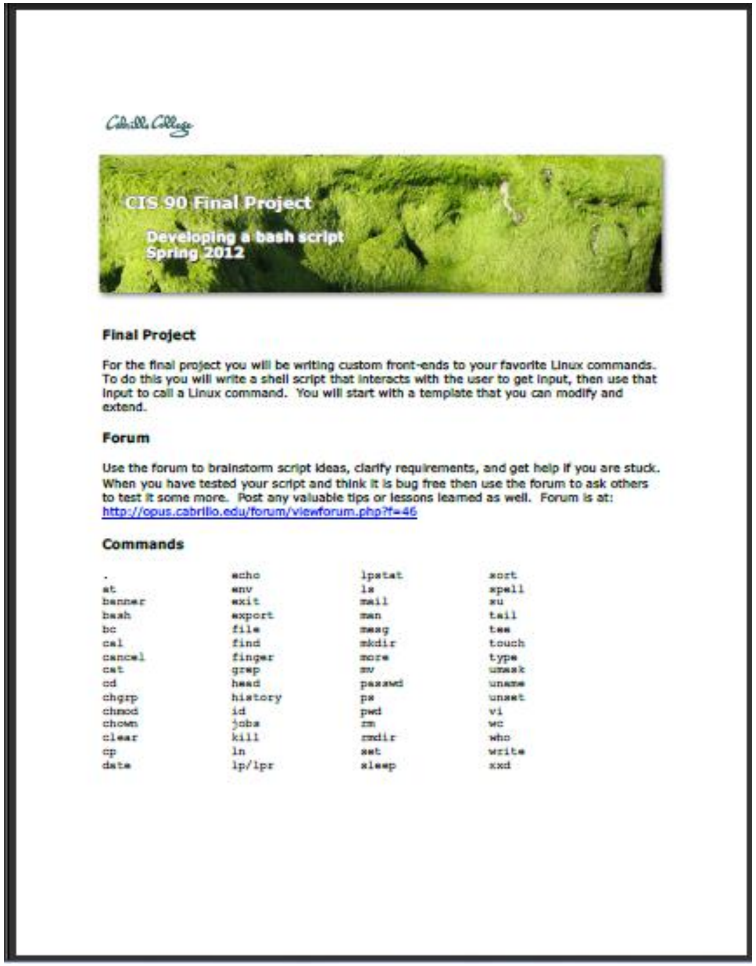
Try it!

Use /bin/mail and send me (rsimms) a message that you have made or corrected using the ~v command.

cc: yourself so you can verify what you sent.

final project preview

Final Project



CIS 90 Final Project

Developing a bash script
Spring 2012

Final Project

For the final project you will be writing custom front-ends to your favorite Linux commands. To do this you will write a shell script that interacts with the user to get input, then use that input to call a Linux command. You will start with a template that you can modify and extend.

Forum

Use the forum to brainstorm script ideas, clarify requirements, and get help if you are stuck. When you have tested your script and think it is bug free then use the forum to ask others to test it some more. Post any valuable tips or lessons learned as well. Forum is at: http://opus.cabrillo.edu/forum/viewforum.php?f=46

Commands

```
.            echo         lpstat       sort
at           env          ls           spell
banner       exit         mail         su
bash         export       man          tail
bc           file         mesg         tee
cal          find         mkdir        touch
cancel       finger       more         type
cat          grep         mv           umask
cd           head         passwd       uname
chgrp        history      ps           unset
chmod        id           pwd          vi
chown        jobs         rm           wc
clear        kill         rmdir        who
cp           ln           set          write
date         lp/lpr       sleep        xxd
```

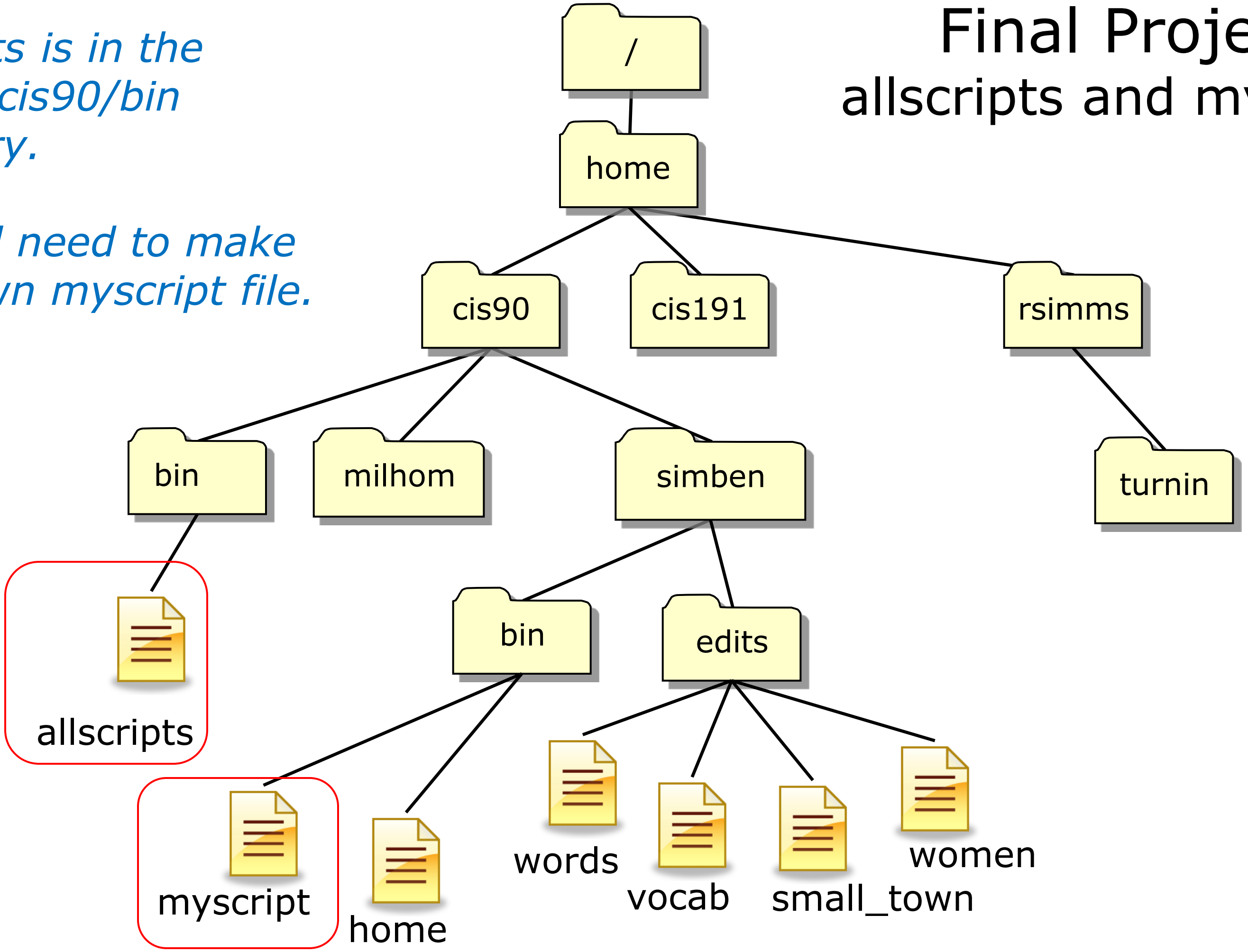
You now have the necessary skills to begin the final project!

Final Project

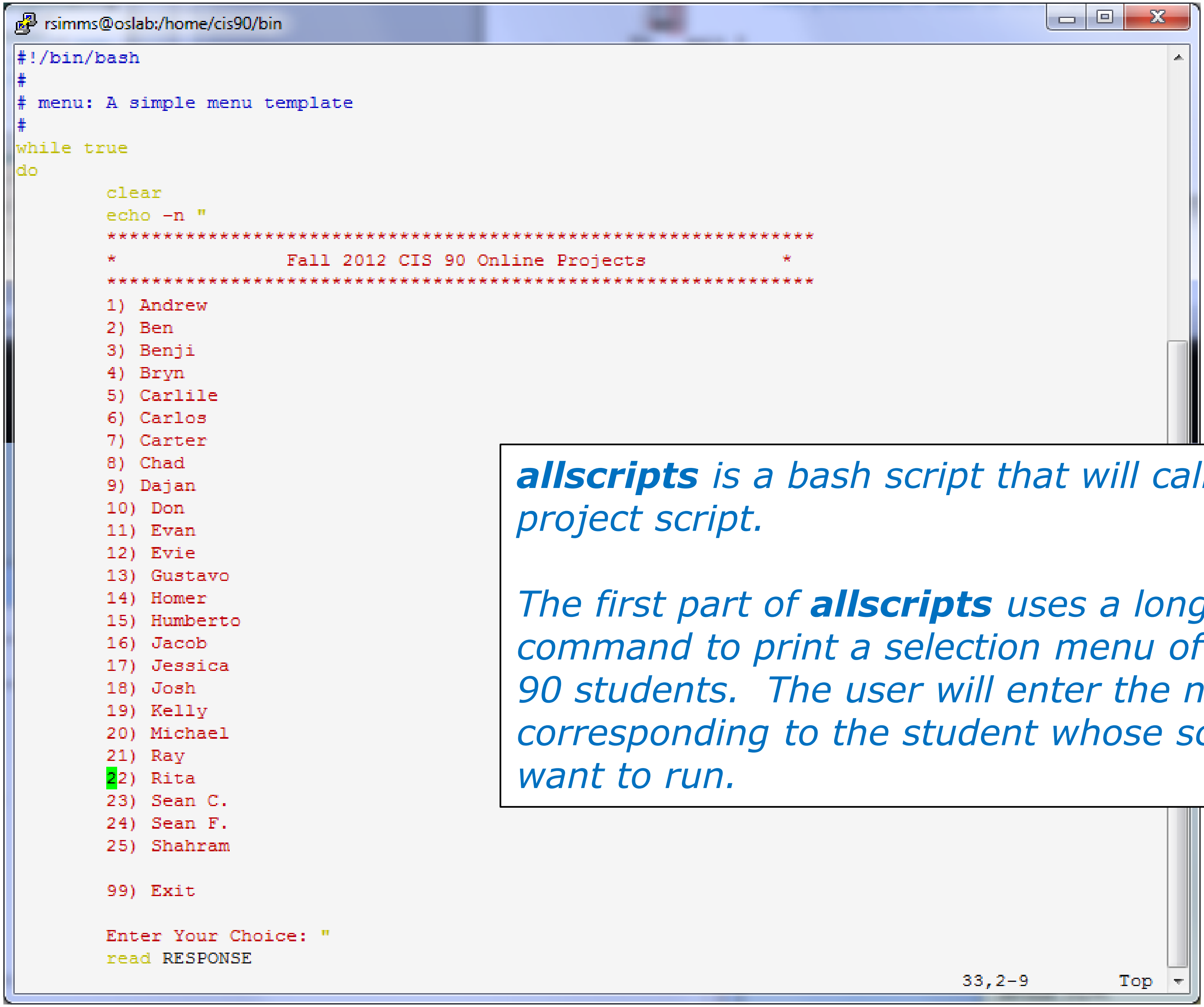
allscripts and myscript

allscripts is in the /home/cis90/bin directory.

You will need to make your own myscript file.



```
/home/cis90/rodduk $ ls -l /home/cis90/bin/allscripts  bin/myscript
-rwxr-xr-x 1 simben90 cis90 4296 Nov 13 13:07 bin/myscript
-rwxr-xr-x 1 rsimms   staff 4381 Nov 13 18:17 /home/cis90/bin/allscripts
```

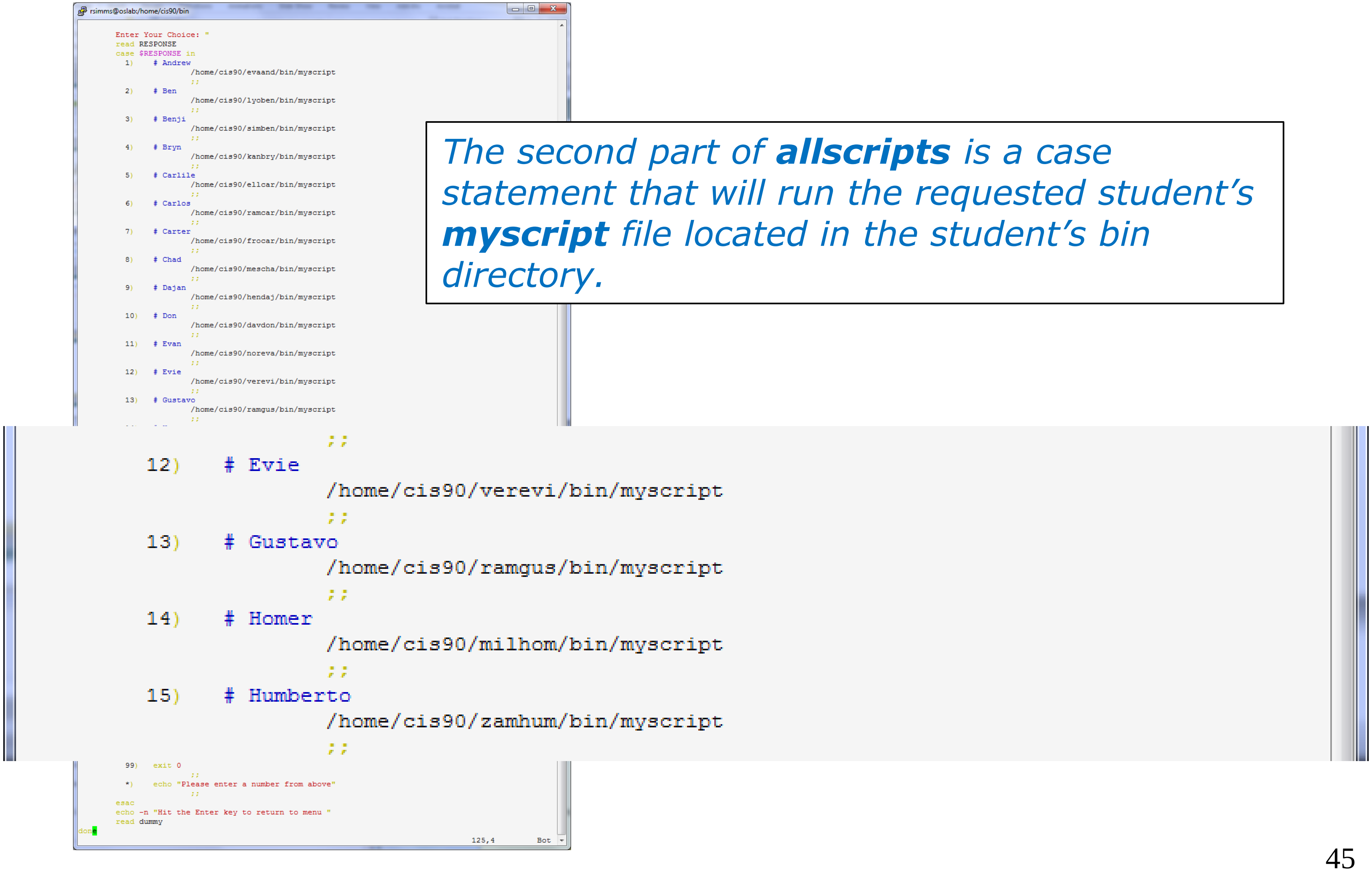


```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
        ***************************************************************
        *               Fall 2012 CIS 90 Online Projects             *
        ***************************************************************
        1) Andrew
        2) Ben
        3) Benji
        4) Bryn
        5) Carlile
        6) Carlos
        7) Carter
        8) Chad
        9) Dajan
        10) Don
        11) Evan
        12) Evie
        13) Gustavo
        14) Homer
        15) Humberto
        16) Jacob
        17) Jessica
        18) Josh
        19) Kelly
        20) Michael
        21) Ray
        22) Rita
        23) Sean C.
        24) Sean F.
        25) Shahram

        99) Exit

        Enter Your Choice: "
        read RESPONSE
                                                                                  33,2-9        Top
```

__allscripts__ is a bash script that will call your project script.

The first part of __allscripts__ uses a long __echo__ command to print a selection menu of the CIS 90 students. The user will enter the number corresponding to the student whose script they want to run.

*The second part of **allscripts** is a case statement that will run the requested student's **myscript** file located in the student's bin directory.*



rsimms@oslab:/home/cis90/bin

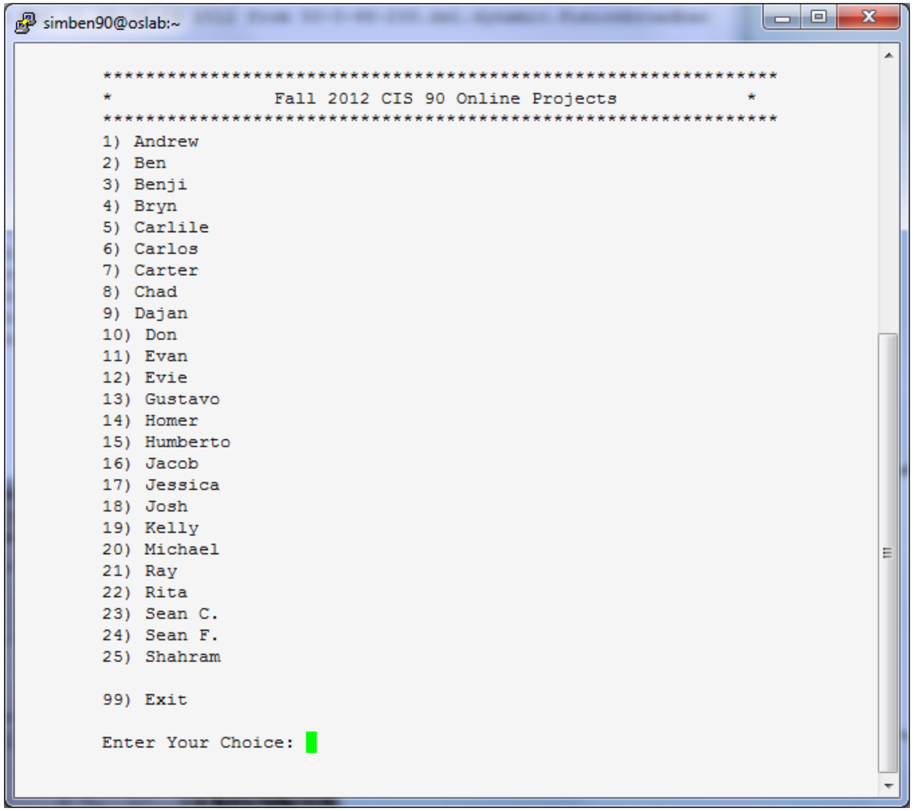
```
        Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Andrew
                        /home/cis90/evaand/bin/myscript
                        ;;
          2)    # Ben
                        /home/cis90/lyoben/bin/myscript
                        ;;
          3)    # Benji
                        /home/cis90/simben/bin/myscript
                        ;;
          4)    # Bryn
                        /home/cis90/kanbry/bin/myscript
                        ;;
          5)    # Carlile
                        /home/cis90/ellcar/bin/myscript
                        ;;
          6)    # Carlos
                        /home/cis90/ramcar/bin/myscript
                        ;;
          7)    # Carter
                        /home/cis90/frocar/bin/myscript
                        ;;
          8)    # Chad
                        /home/cis90/mescha/bin/myscript
                        ;;
          9)    # Dajan
                        /home/cis90/hendaj/bin/myscript
                        ;;
          10)   # Don
                        /home/cis90/davdon/bin/myscript
                        ;;
          11)   # Evan
                        /home/cis90/noreva/bin/myscript
                        ;;
          12)   # Evie
                        /home/cis90/verevi/bin/myscript
                        ;;
          13)   # Gustavo
                        /home/cis90/ramgus/bin/myscript
                        ;;
```

```
                        ;;
          12)   # Evie
                        /home/cis90/verevi/bin/myscript
                        ;;
          13)   # Gustavo
                        /home/cis90/ramgus/bin/myscript
                        ;;
          14)   # Homer
                        /home/cis90/milhom/bin/myscript
                        ;;
          15)   # Humberto
                        /home/cis90/zamhum/bin/myscript
                        ;;
```

```
          99)   exit 0
                        ;;
          *)    echo "Please enter a number from above"
                        ;;
        esac
        echo -n "Hit the Enter key to return to menu "
        read dummy
done
                                                   125,4         Bot
```

Final Project
allscripts (continued)

Running **/home/cis90/bin/allscripts** looks like this



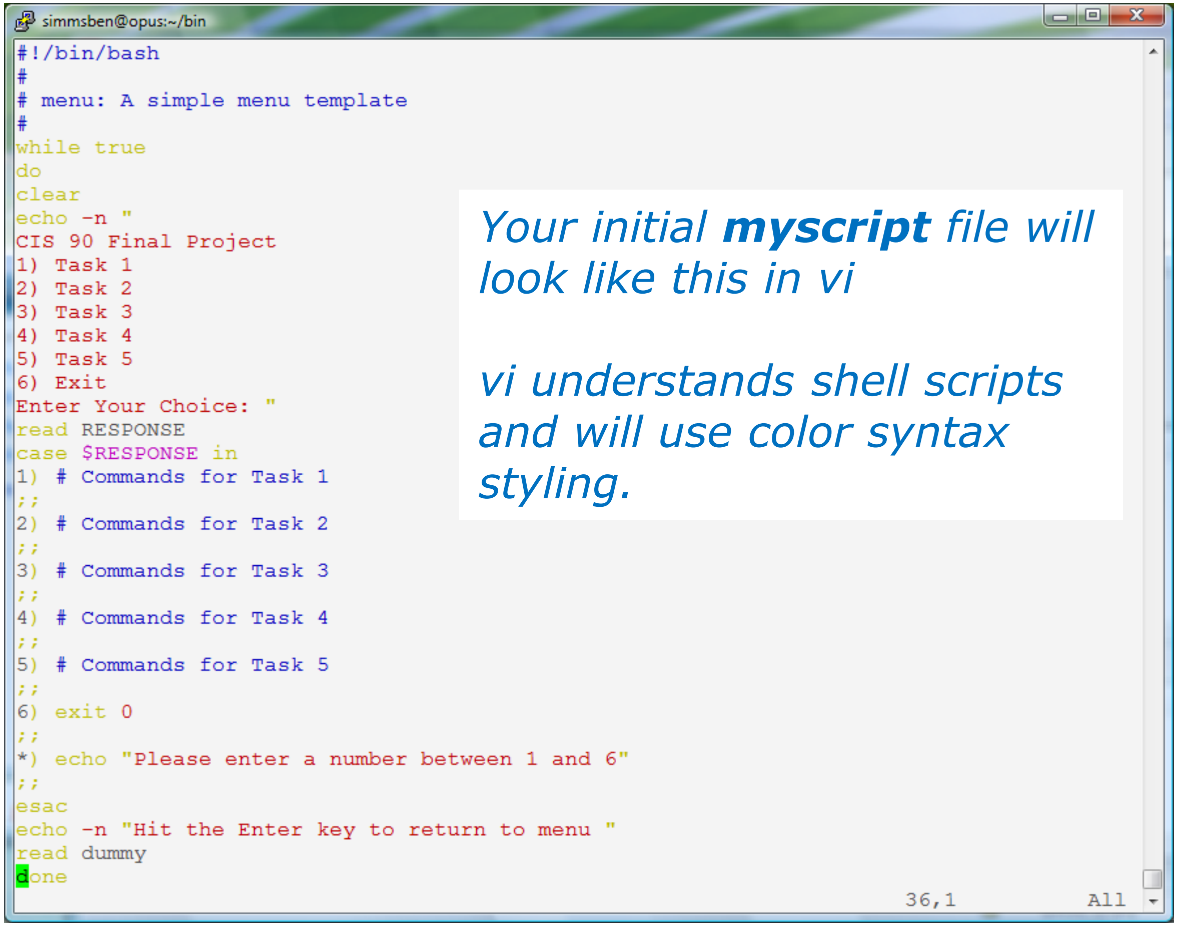
```
***************************************************************
*               Fall 2012 CIS 90 Online Projects            *
***************************************************************
1) Andrew
2) Ben
3) Benji
4) Bryn
5) Carlile
6) Carlos
7) Carter
8) Chad
9) Dajan
10) Don
11) Evan
12) Evie
13) Gustavo
14) Homer
15) Humberto
16) Jacob
17) Jessica
18) Josh
19) Kelly
20) Michael
21) Ray
22) Rita
23) Sean C.
24) Sean F.
25) Shahram

99) Exit

Enter Your Choice:
```

*This script has been updated with everyone's name and pathnames to each student's **myscript** file*

Final Project

myscript

/home/cis90/${LOGNAME%90}/bin/myscript



```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
clear
echo -n "
CIS 90 Final Project
1) Task 1
2) Task 2
3) Task 3
4) Task 4
5) Task 5
6) Exit
Enter Your Choice: "
read RESPONSE
case $RESPONSE in
1) # Commands for Task 1
;;
2) # Commands for Task 2
;;
3) # Commands for Task 3
;;
4) # Commands for Task 4
;;
5) # Commands for Task 5
;;
6) exit 0
;;
*) echo "Please enter a number between 1 and 6"
;;
esac
echo -n "Hit the Enter key to return to menu "
read dummy
done
                                                            36,1          All
```

*Your initial **myscript** file will look like this in vi*

vi understands shell scripts and will use color syntax styling.

*Every student needs to create a **myscript** file in their bin directory.*

*Use vi to create the **myscript** file and copy and paste the template code from the Final Project into it.*

Final Project

/home/cis90/${LOGNAME%90}/bin/myscript

Getting Started

1) On Opus, cd to your bin directory and enter:

   vi myscript

   then type **i** to enter insert mode

2) In your web browser, view the CIS 90 calendar page and click on the project link for Lesson 15. Select the template code and copy it to the clipboard.

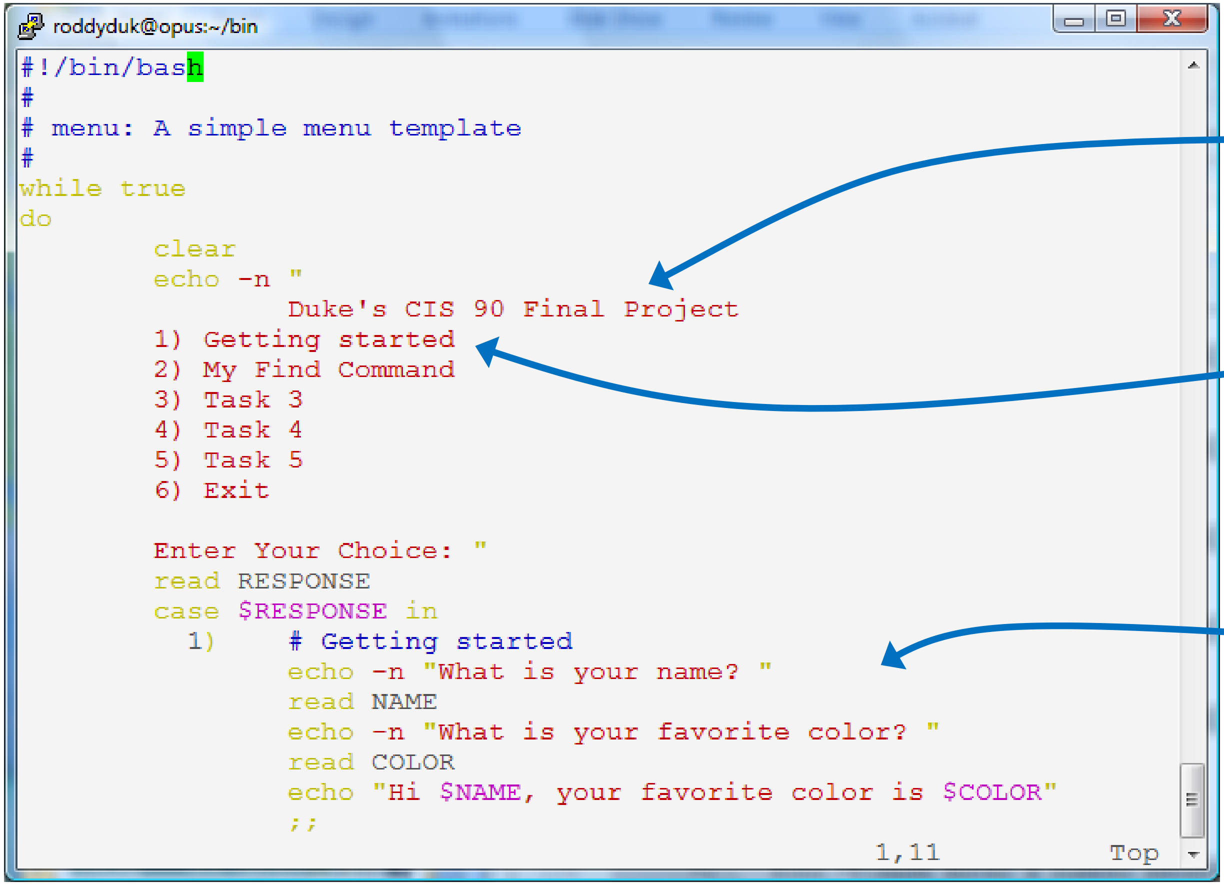
3) Click back on the vi session and click the right mouse button to paste the template code.

4) Save the code with **Esc** and the **:wq**

5) Give myscript execute permissions with **chmod +x myscript**

Final Project

/home/cis90/${LOGNAME%90}/bin/myscript



roddyduk@opus:~/bin

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
                Duke's CIS 90 Final Project
        1) Getting started
        2) My Find Command
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Getting started
                echo -n "What is your name? "
                read NAME
                echo -n "What is your favorite color? "
                read COLOR
                echo "Hi $NAME, your favorite color is $COLOR"
                ;;
                                                   1,11          Top
```

Customize your menu title

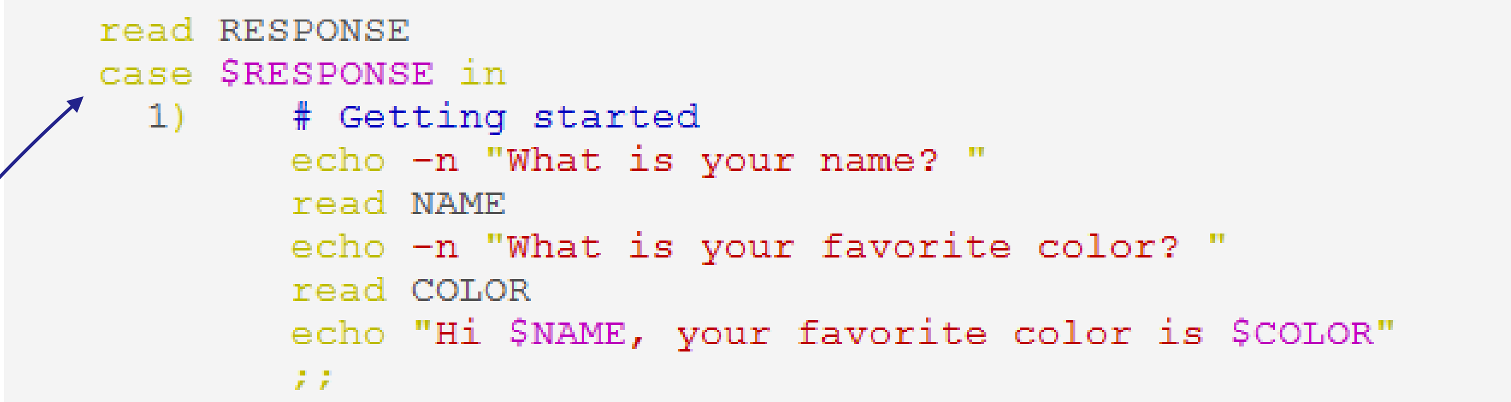
Add a menu entry

Add some sample dialog code using variables

Final Project

/home/cis90/${LOGNAME%90}/bin/myscript

A new command



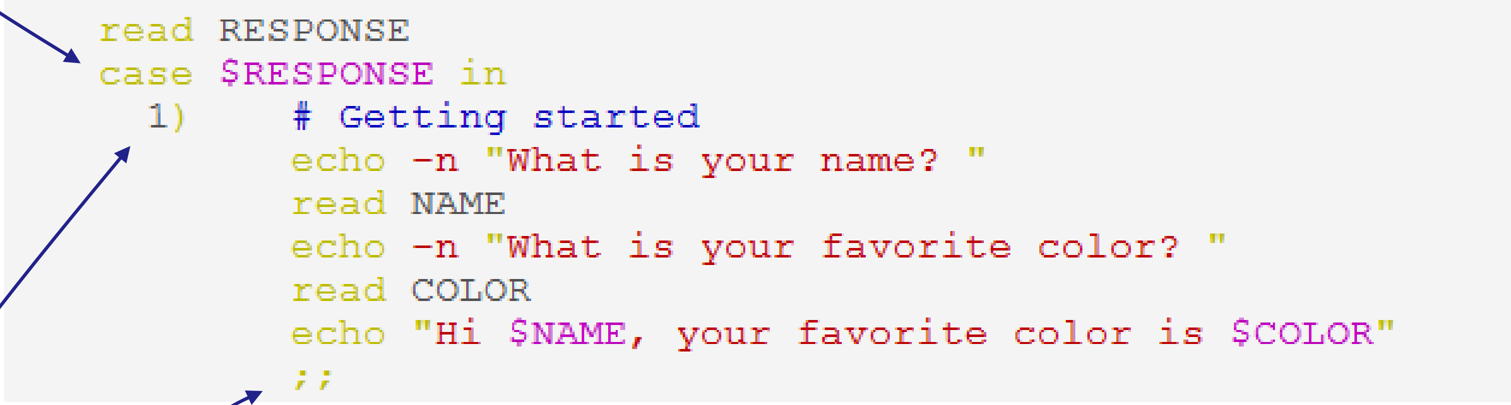
```
read RESPONSE
case $RESPONSE in
  1)    # Getting started
        echo -n "What is your name? "
        read NAME
        echo -n "What is your favorite color? "
        read COLOR
        echo "Hi $NAME, your favorite color is $COLOR"
        ;;
```

another new command

Final Project

/home/cis90/${LOGNAME%90}/bin/myscript

case statement begins here



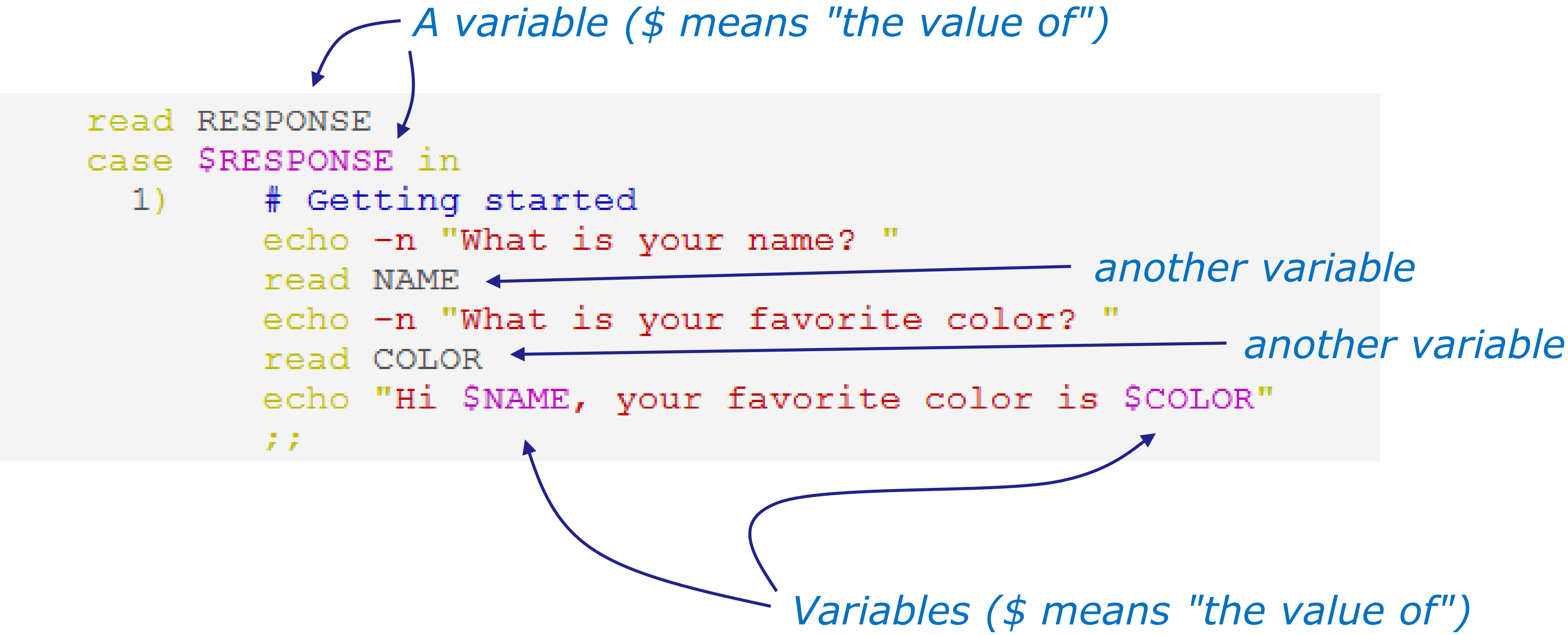
```
read RESPONSE
case $RESPONSE in
  1)    # Getting started
        echo -n "What is your name? "
        read NAME
        echo -n "What is your favorite color? "
        read COLOR
        echo "Hi $NAME, your favorite color is $COLOR"
        ;;
```

First case ends here

First case of case statement starts here

Final Project

/home/cis90/${LOGNAME%90}/bin/myscript



A variable ($ means "the value of")

```
    read RESPONSE
    case $RESPONSE in
      1)     # Getting started
             echo -n "What is your name? "
             read NAME
             echo -n "What is your favorite color? "
             read COLOR
             echo "Hi $NAME, your favorite color is $COLOR"
             ;;
```

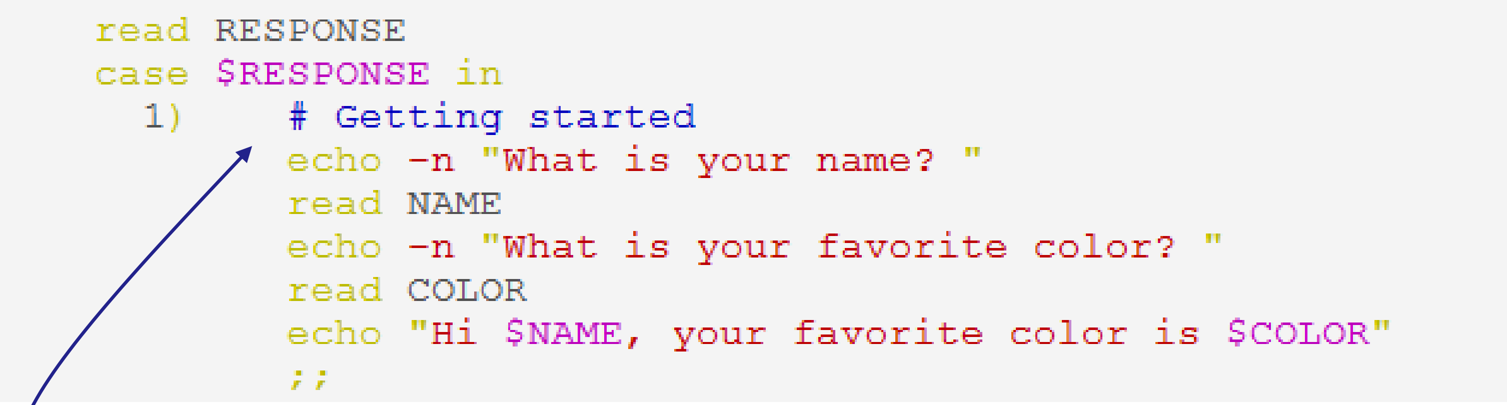
another variable

another variable

Variables ($ means "the value of")

Final Project

/home/cis90/${LOGNAME%90}/bin/myscript

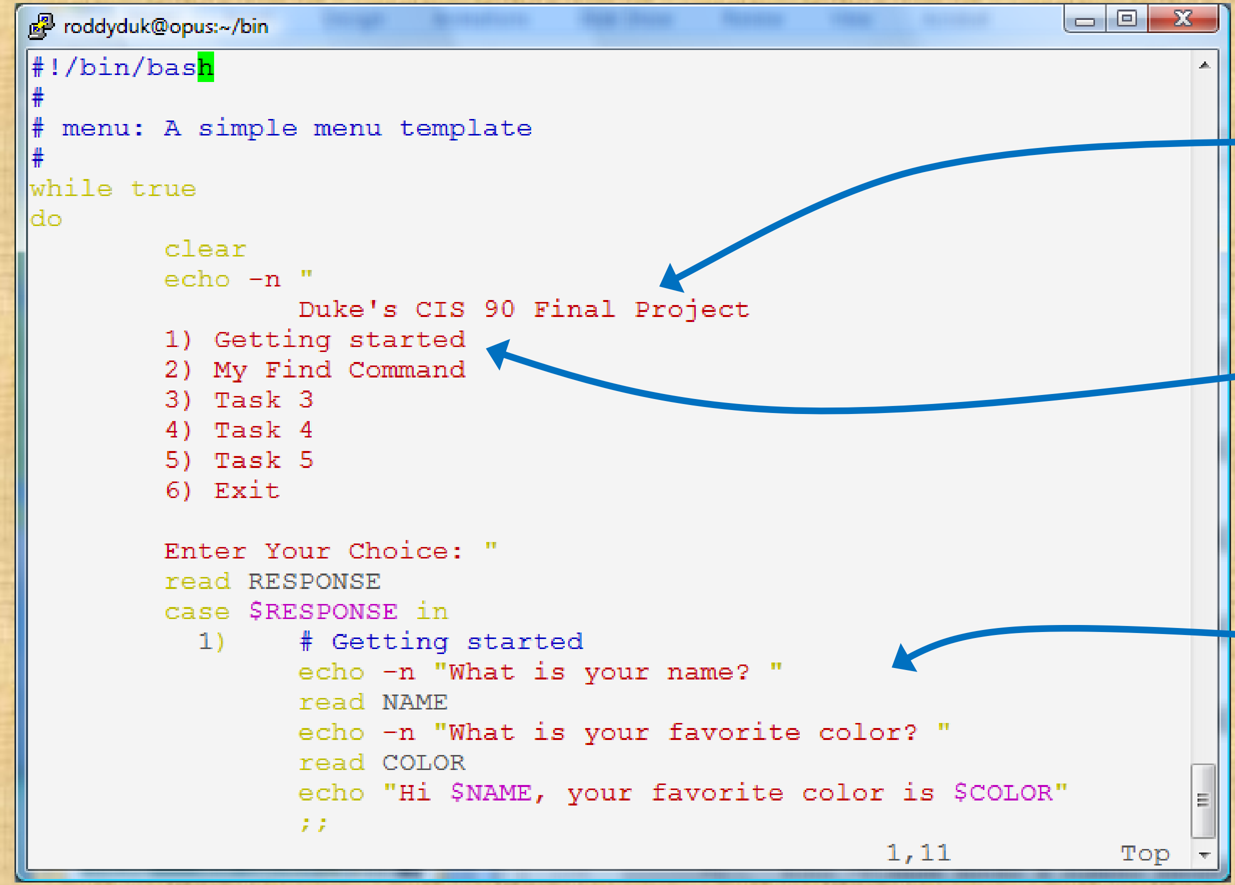


```
    read RESPONSE
    case $RESPONSE in
      1)     # Getting started
             echo -n "What is your name? "
             read NAME
             echo -n "What is your favorite color? "
             read COLOR
             echo "Hi $NAME, your favorite color is $COLOR"
             ;;
```

Comments begin with a #

Final Project

/home/cis90/${LOGNAME%90}/bin/myscript



```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
                Duke's CIS 90 Final Project
        1) Getting started
        2) My Find Command
        3) Task 3
        4) Task 4
        5) Task 5
        6) Exit

        Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Getting started
                echo -n "What is your name? "
                read NAME
                echo -n "What is your favorite color? "
                read COLOR
                echo "Hi $NAME, your favorite color is $COLOR"
                ;;
                                                        1,11          Top
```

Customize your menu title

Customize the first menu entry

Add this sample dialog code using variables

*When finished, test both the **myscript** and **allscripts** "commands"*

Shell Variables

Shell Variables

LOGNAME LANG
HOME
SHELL SSH_TTY
PWD
EUID
BASH_VERSION LINES COLORS
PPID
IFS
consoletype
SHELLOPTS
MAILCHECK HOSTNAME
BASH_ENV
BASH
USER PS4 TERM PIPESTATUS GROUPS
HISTFILESIZE OPTIND
BASH_VERSINFO
UID
PATH
BASH_ARGV PS1
SSH_CONNECTION
tmpid
HISTFILE
SHLVL
OSTYPE
USERNAME
BASH_ARGC
BASH_LINENO LESSOPEN
HISTSIZE
OPTERR
SSH_CLIENT
HOSTTYPE
LS_COLORS CVS_RSH
COLUMNS
INPUTRC
BASH_SOURCE
MACHTYPE
PROMPT_COMMAND _
PS2
SSH_ASKPASS
MAIL DIRSTACK G_BROKEN_FILENAMES

*See all shell variables by typing **set***

Shell Variables

- Shell variables names consist of alpha-numeric characters.
- Variables defined by the Operating System are uppercase, e.g. TERM, PS1, PATH
- The **set** command will display all the shell's current variables and their values.
- Shell variables are initialized using the assignment operator:
  For example: **TERM=vt100**
  Note: Quotes must be used for white space: **VALUE="any value"**
- Variables may be viewed using the echo command:
  e.g. **echo $TERM**
  The $ in front of a variable name denotes the value of that variable.
- To remove a variable, use the unset command: **unset PS1**
- Shell variables hold their values for the duration of the session i.e. until the shell is exited

Shell Variables

```
/home/cis90/simben/Poems $ set
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
BASH_ENV=/home/cis90/simben/.bashrc
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="3" [1]="2" [2]="25" [3]="1"
[4]="release" [5]="i686-redhat-linux-gnu")
BASH_VERSION='3.2.25(1)-release'
COLORS=/etc/DIR_COLORS.xterm
COLUMNS=80
CVS_RSH=ssh
DIRSTACK=()
EUID=1160
GROUPS=()
G_BROKEN_FILENAMES=1
HISTFILE=/home/cis90/simben/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/home/cis90/simben
HOSTNAME=opus.cabrillo.edu
HOSTTYPE=i686
IFS=$' \t\n'
IGNOREEOF=10
INPUTRC=/etc/inputrc
LANG=en_US.UTF-8
LESSOPEN='|/usr/bin/lesspipe.sh %s'
LINES=24
LOGNAME=simben
LS_COLORS='no=00:fi=00:di=00;34:ln=00;36:pi=40;33:so=00;35
:bd=40;33;01:cd=40;33;01:or=01;05;37;41:mi=01;05;37;41:ex=
00;32:*.cmd=00;32:*.exe=00;32:*.com=00;32:*.btm=00;32:*.ba
t=00;32:*.sh=00;32:*.csh=00;32:*.tar=00;31:*.tgz=00;31:*.a
rj=00;31:*.taz=00;31:*.lzh=00;31:*.zip=00;31:*.z=00;31:*.Z
=00;31:*.gz=00;31:*.bz2=00;31:*.bz=00;31:*.tz=00;31:*.rpm=
00;31:*.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=00;35:*.x
bm=00;35:*.xpm=00;35:*.png=00;35:*.tif=00;35:'
MACHTYPE=i686-redhat-linux-gnu
MAIL=/var/spool/mail/simben
MAILCHECK=60
OLDPWD=/home/cis90/simben
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/
cis90/simben/../bin:/home/cis90/simben/bin:.
PIPESTATUS=([0]="0")
PPID=26514
PROMPT_COMMAND='echo -ne
"\033]0;${USER}@${HOSTNAME%%.*}:${PWD/#$HOME/~}"; echo -ne
"\007"'
PS1='$PWD $'
PS2='> '
PS4='+ '
PWD=/home/cis90/simben/Poems
SHELL=/bin/bash
SHELLOPTS=braceexpand:emacs:hashall:histexpand:ignoreeof:i
nteractive-comments:monitor
SHLVL=1
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
TERM=xterm
UID=1160
USER=simben
USERNAME=
_=env
consoletype=pty
```

The set command, with no arguments, will show all shell variables and their values

Showing the values of variables

Use: **echo $***varname*

Example 1

```
[rsimms@nosmo ~]$ echo $PATH
/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/home/rsimms/bin
```

Example 2

```
[rsimms@nosmo ~]$ echo $TERM
xterm
```

Example 3

```
[rsimms@nosmo ~]$ echo $HOME
/home/rsimms
```

Example 4

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

Using the echo command to show the values of variables

Setting the values of variables

Use: *varname***=***value*
(no spaces please around the =)

Example 1

```
[rsimms@nosmo ~]$ PS1="By your command >"
By your command >
By your command >PS1="What can I do for you $LOGNAME? "
What can I do for you rsimms?
What can I do for you rsimms?
```

Example 2

```
/home/cis90/simben/bin $ river="The Amazon"
/home/cis90/simben/bin $ echo $river
The Amazon
/home/cis90/simben/bin $ echo river
river
```

Creating Shell Variables

1

```
/home/cis90/simmen/bin $ echo $defrost $ac $fan

/home/cis90/simmen/bin $
```

the value of a variable that has not been created is null

2

```
/home/cis90/simmen/bin $ defrost=on
/home/cis90/simmen/bin $ ac=off
/home/cis90/simmen/bin $ fan=medium
```

create some new shell variables and assign values

3

```
/home/cis90/simmen/bin $ echo $defrost $ac $fan
on off medium
```

*print the **values** of the shell variables*

```
/home/cis90/simmen/bin $ echo defrost ac fan
defrost ac fan
```

*print the **names** of the shell variables*

Shell Variables

```
/home/cis90/simben $ defrost=on
/home/cis90/simben $ ac=off
/home/cis90/simben $ fan=medium
/home/cis90/simben $ set
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
BASH_ENV=/home/cis90/simben/.bashrc
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="3" [1]="2" [2]="25" [3]="1" [4]="release" [5]="i686-redhat-linux-gnu")
BASH_VERSION='3.2.25(1)-release'
COLORS=/etc/DIR_COLORS.xterm
COLUMNS=84
CVS_RSH=ssh
DIRSTACK=()
EUID=1156
GROUPS=()
G_BROKEN_FILENAMES=1
HISTFILE=/home/cis90/simben/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/home/cis90/simben
HOSTNAME=opus.cabrillo.edu
HOSTTYPE=i686
IFS=$' \t\n'
IGNOREEOF=10
INPUTRC=/etc/inputrc
LANG=en_US.UTF-8
LESSOPEN='|/usr/bin/lesspipe.sh %s'
LINES=39
LOGNAME=simben
LS_COLORS='no=00:fi=00:di=00;34:ln=00;36:pi=40;33:so=00;35:bd=40;33;01:cd=40;33;01:or=01;05;37;41:mi=01;05;37;41:ex=00;32:*.cmd=00;32:*.exe=00;32:*.com=00;32:*.btm=00;32:*.bat=00;32:*.sh=00;32:*.csh=00;32:*.tar=00;31:*.tgz=00;31:*.arj=00;31:*.taz=00;31:*
.lzh=00;31:*.zip=00;31:*.z=00;31:*.Z=00;31:*.gz=00;31:*.bz2=00;31:*.bz=00;31:*.tz=00;31:*.rpm=00;31:*.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=00;35:*.xbm=00;35:*.xpm=00;35:*.png=00;35:*.tif=00;35:'
MACHTYPE=i686-redhat-linux-gnu
MAIL=/var/spool/mail/simben
MAILCHECK=60
OLDPWD=/home/cis90/simben/edits
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/cis90/simben/../bin:/home/cis90/simben/bin:.
PIPESTATUS=([0]="0")
PPID=7254
PROMPT_COMMAND='echo -ne "\033]0;${USER}@${HOSTNAME%%.*}:${PWD/#$HOME/~}"; echo -ne "\007"'
PS1='$PWD $ '
PS2='> '
PS4='+ '
PWD=/home/cis90/simben
SHELL=/bin/bash
SHELLOPTS=braceexpand:emacs:hashall:histexpand:ignoreeof:interactive-comments:monitor
SHLVL=1
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
SSH_CLIENT='63.249.103.107 19509 22'
SSH_CONNECTION='63.249.103.107 19509 207.62.186.9 22'
SSH_TTY=/dev/pts/1
TERM=xterm
UID=1156
USER=simben
USERNAME=
_=
ac=off
consoletype=pty
defrost=on
fan=medium
```

Note: Any new variables you initialize will show up in the output of the ***set*** *command*

font reduced for the other variables to fit on slide

Shell Variables

Using grep to find a variable in the output of the set command

```
/home/cis90/simben $ set | grep defrost
defrost=on
```

The output of the set command is piped to the grep command which displays only lines containing “defrost”

Class Activity

Create and initialize three new variables:

```
defrost=on
ac=off
fan=medium
```

Show the names of the variables:

```
echo  defrost  ac  fan
```

Show the values of the variables:

```
echo  $defrost  $ac  $fan
```

Display all variables and locate yours:

```
set
set | grep defrost
set | grep ac
set | grep fan
```

Removing Shell Variables

To remove a variable, use the unset command: **unset PS1**

```
/home/cis90/simben $ echo  $defrost  $ac  $fan
on off medium
```

show values

```
/home/cis90/simben $ unset defrost
/home/cis90/simben $ echo  $defrost  $ac  $fan
off medium
```

remove one of the variables

```
/home/cis90/simben $ unset ac fan
/home/cis90/simben $ echo  $defrost  $ac  $fan

/home/cis90/simben $
```

remove remaining variables

Class Exercise

Delete your three new variables:

```
unset  defrost
unset  ac  fan
```

Show the names of the variables:

```
echo  defrost  ac  fan
```

Show the values of the variables:

```
echo  $defrost  $ac  $fan
```

Shell Variables

Variables are often used in scripts when you need a placeholder to store some data

1

```
/home/cis90/simben $ vi funscript
/home/cis90/simben $ cat funscript
#!/bin/bash
echo -n "Turn the Air Conditioning on or off? "
read ac
echo "Air Conditioning set to $ac"
exit
```

Create a script that uses a variable named "ac" to hold the status of an air conditioner.

Prompt the user and input what they type into the this variable.

2

```
/home/cis90/simben $ chmod +x funscript
```

Add execute permissions so the script can be run

3

```
/home/cis90/simben $ ./funscript
Turn the Air Conditioning on or off? off
Air Conditioning set to off
```

Run the script

Class Exercise

Now make this little user dialog script:

```
vi funscript
```

insert the following lines then save

```
#!/bin/bash
echo -n "Turn the Air Conditioning on or off? "
read ac
echo "Air Conditioning set to $ac"
exit
```

```
chmod +x funscript
```

```
./funscript
```

Environment Variables

Shell Variables

SHELL SSH_TTY LOGNAME EUID HOME LANG PWD

BASH_VERSION LINES COLORS PPID

consoletype IFS SHELLOPTS

MAILCHECK BASH_ENV HOSTNAME

USER BASH PS4 TERM PIPESTATUS GROUPS

HISTFILESIZE OPTIND BASH_VERSINFO

UID

BASH_ARGV PATH PS1

SSH_CONNECTION

SHLVL tmpid HISTFILE

BASH_ARGC USERNAME OSTYPE

HISTSIZE BASH_LINENO LESSOPEN

OPTERR SSH_CLIENT

HOSTTYPE LS_COLORS CVS_RSH

COLUMNS INPUTRC BASH_SOURCE

PROMPT_COMMAND _ MACHTYPE

PS2

DIRSTACK MAIL SSH_ASKPASS G_BROKEN_FILENAMES

*Use the **set** command to show all shell variables*

Environment Variables

LOGNAME **LANG** **HOME** **SHELL** **SSH_TTY** **PWD**

EUID

BASH_VERSION LINES COLORS PPID

IFS consoletype SHELLOPTS

MAILCHECK **HOSTNAME** **BASH_ENV**

BASH **USER** PS4 **TERM** PIPESTATUS GROUPS

HISTFILESIZE OPTIND

BASH_VERSINFO

UID

PATH BASH_ARGV PS1

SSH_CONNECTION

tmpid HISTFILE **SHLVL**

OSTYPE **USERNAME** BASH_ARGC

BASH_LINENO **LESSOPEN** **HISTSIZE**

OPTERR **SSH_CLIENT**

HOSTTYPE **LS_COLORS** **CVS_RSH**

COLUMNS **INPUTRC**

BASH_SOURCE MACHTYPE **_** PROMPT_COMMAND

PS2

SSH_ASKPASS **MAIL** DIRSTACK

G_BROKEN_FILENAMES

*Use the **env** to see which of the shell variables have been exported and therefore environment variables (shown in bold/green above)*

Environment Variables

- Environment variables are a special subset of the shell variables.
- Environment variables are shell variables that have been *exported*.
- The **env** command will display the current environment variables and their values. Using the **export** command with no arguments will also show all the environment variables.
- The **export** command is used to make a shell variable into an environment variable.

  dog=benji; export dog
  or **export dog=benji**
- The **export -n** command is used to make an environment variable back into a normal shell variable. E.g. **export -n dog** makes dog back into a regular shell variable.

Child processes are provided copies of the parent's environment variables.

Any changes made by the child will not affect the parent's copies.

Shell (Environment) Variables

env command – show all environment variables

```
[simben@opus ~]$ env
HOSTNAME=opus.cabrillo.edu
SHELL=/bin/bash
TERM=xterm
HISTSIZE=1000
SSH_CLIENT=63.249.103.107 20807 22
SSH_TTY=/dev/pts/0
USER=simben
LS_COLORS=no=00:fi=00:di=00;34:ln=00;36:pi=40;33:so=00;35:bd=40;33;01:cd=40;33;01:or=01;05;37;41:mi=01;05
;37;41:ex=00;32:*.cmd=00;32:*.exe=00;32:*.com=00;32:*.btm=00;32:*.bat=00;32:*.sh=00;32:*.csh=00;32:*.tar=
00;31:*.tgz=00;31:*.arj=00;31:*.taz=00;31:*.lzh=00;31:*.zip=00;31:*.z=00;31:*.Z=00;31:*.gz=00;31:*.bz2=00
;31:*.bz=00;31:*.tz=00;31:*.rpm=00;31:*.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=00;35:*.xbm=00;35:*.xpm=
00;35:*.png=00;35:*.tif=00;35:
USERNAME=
PATH=/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/cis90/simben/../bin:/home/cis90/simben/bin:.
MAIL=/var/spool/mail/simben
PWD=/home/cis90/simben
INPUTRC=/etc/inputrc
LANG=en_US.UTF-8
fan=medium
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
HOME=/home/cis90/simben
SHLVL=2
BASH_ENV=/home/cis90/simben/.bashrc
LOGNAME=simben
CVS_RSH=ssh
SSH_CONNECTION=63.249.103.107 20807 207.62.186.9 22
LESSOPEN=|/usr/bin/lesspipe.sh %s
G_BROKEN_FILENAMES=1
_=/bin/env
```

The env command by itself will list all the environment (exported) variables

Shell (Environment) Variables

export command – show all exported variables

```
[simben@opus ~]$ export
declare -x BASH_ENV="/home/cis90/simben/.bashrc"
declare -x CVS_RSH="ssh"
declare -x G_BROKEN_FILENAMES="1"
declare -x HISTSIZE="1000"
declare -x HOME="/home/cis90/simben"
declare -x HOSTNAME="opus.cabrillo.edu"
declare -x INPUTRC="/etc/inputrc"
declare -x LANG="en_US.UTF-8"
declare -x LESSOPEN="|/usr/bin/lesspipe.sh %s"
declare -x LOGNAME="simben"
declare -x
LS_COLORS="no=00:fi=00:di=00;34:ln=00;36:pi=40;33:so=00;35:bd=40;33;01:cd=40;33;01:or=01;05;37;41:mi=01;05;37
;41:ex=00;32:*.cmd=00;32:*.exe=00;32:*.com=00;32:*.btm=00;32:*.bat=00;32:*.sh=00;32:*.csh=00;32:*.tar=00;31:*
.tgz=00;31:*.arj=00;31:*.taz=00;31:*.lzh=00;31:*.zip=00;31:*.z=00;31:*.Z=00;31:*.gz=00;31:*.bz2=00;31:*.bz=00
;31:*.tz=00;31:*.rpm=00;31:*.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=00;35:*.xbm=00;35:*.xpm=00;35:*.png=00;
35:*.tif=00;35:"
declare -x MAIL="/var/spool/mail/simben"
declare -x OLDPWD
declare -x
PATH="/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/cis90/simben/../bin:/home/cis90/simben/bin:."
declare -x PWD="/home/cis90/simben"
declare -x SHELL="/bin/bash"
declare -x SHLVL="2"
declare -x SSH_ASKPASS="/usr/libexec/openssh/gnome-ssh-askpass"
declare -x SSH_CLIENT="63.249.103.107 20807 22"
declare -x SSH_CONNECTION="63.249.103.107 20807 207.62.186.9 22"
declare -x SSH_TTY="/dev/pts/0"
declare -x TERM="xterm"
declare -x USER="simben"
declare -x USERNAME=""
```

*The **export** command by itself will list all the exported (environment) variables.*

*Similar to **env** command but different output format*

Shell (Environment) Variables

export command – show all exported variables

*To create your own environment variable use the **export** command*

1

```
/home/cis90/simben $ env | wc –l
29
/home/cis90/simben $ export | wc -l
29
```

There are currently 24 environment (exported) variables

2

```
/home/cis90/simben $ fan=medium
/home/cis90/simben $ export fan
```

Create a new shell variable named fan and export it so it becomes an environment variable

3

```
/home/cis90/simben $ env | wc -l
30
/home/cis90/simben $ export | wc -l
30
```

Now there are 25 environment variables

4

```
[simben@opus ~]$ export | grep fan
declare -x fan="medium"
[simben@opus ~]$ env | grep fan
fan=medium
[simben@opus ~]$ set | grep fan
fan=medium
```

use grep to show fan is an exported shell variable

Shell Environment

The Shell Environment

- The shell environment can be customized using the environment variables.
- Commands in the shell environment can be customized using aliases.
- Aliases and environment variable settings can be made permanent using the hidden *.bash_profile* and *.bashrc* files in the users home directory.
- Environment variables are exported so they are available to child processes.

Shell (Environment) Variables

Some famous environment variables

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

Customizing the shell prompt with PS1

PS1 settings	Result
`PS1='$PWD $'`	`/home/cis90/simben/Poems $`
`PS1="\w $"`	`~/Poems $`
`PS1="\W $"`	`Poems $`
`PS1="\u@\h $"`	`simben90@opus $`
`PS1='\u@\h $PWD $'`	`simben90@opus /home/cis90/simben/Poems $`
`PS1='\u@\$HOSTNAME $PWD $'`	`simben90@opus.cabrillo.edu /home/cis90/simben/Poems $`
`PS1='\u \! $PWD $'`	`simben90 825 /home/cis90/simben/Poems $`
`PS1="[\u@\h \W/\$"`	`[simben90@opus Poems/$`
`PS1="Enter command: "`	`Enter command:`

Important: Use single quotes around variables that change. For example if you use $PWD with double quotes, the prompt will **not** change as you change directories!

bash shell tip

changing the prompt

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

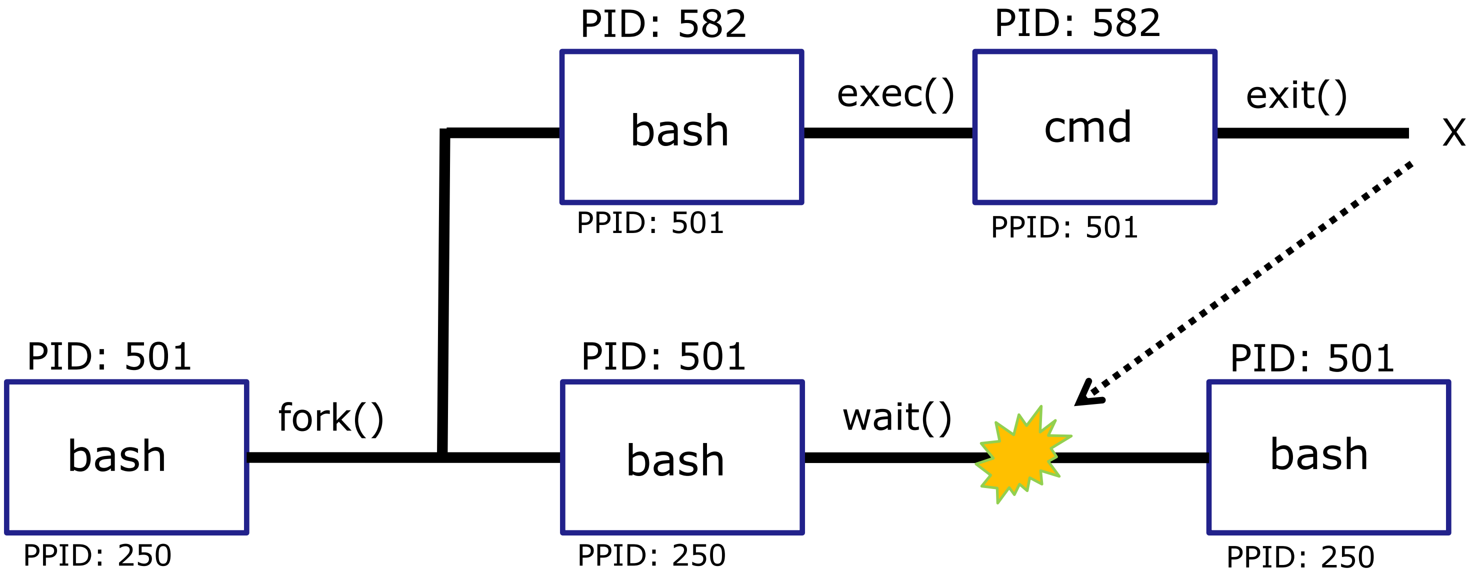
The prompt string can have any combination of text, variables and these codes.

variables and child processes

The rules of the road for variables

- When a shell forks a child, only copies of exported variables are made available to the child.
- A child can modify the variables it receives but those modifications will not change the parent’s variables.

exporting variables



- When a shell forks a child, only copies of exported variables are made available to the child.
- A child can modify the variables it receives but those modifications will not change the parent’s variables.

The rules of the road for variables

- When a shell forks a child, only copies of exported variables are made available to the child.

- A child can modify the variables it receives but those modifications will not change the parent’s variables.

Only exported variables are available to the child

1

```
/home/cis90/simben $ window=down
/home/cis90/simben $ echo $window $LOGNAME
down simben90
```

Create a new variable named window

2

parent

```
/home/cis90/simben $ env | grep window
/home/cis90/simben $ set | grep window
window=down
/home/cis90/simben $ env | grep LOGNAME
LOGNAME=simben90
/home/cis90/simben $ set | grep LOGNAME
LOGNAME=simben90
```

*window is a shell variable that has **not** been exported.*

The environment variable LOGNAME has been exported.

3

child

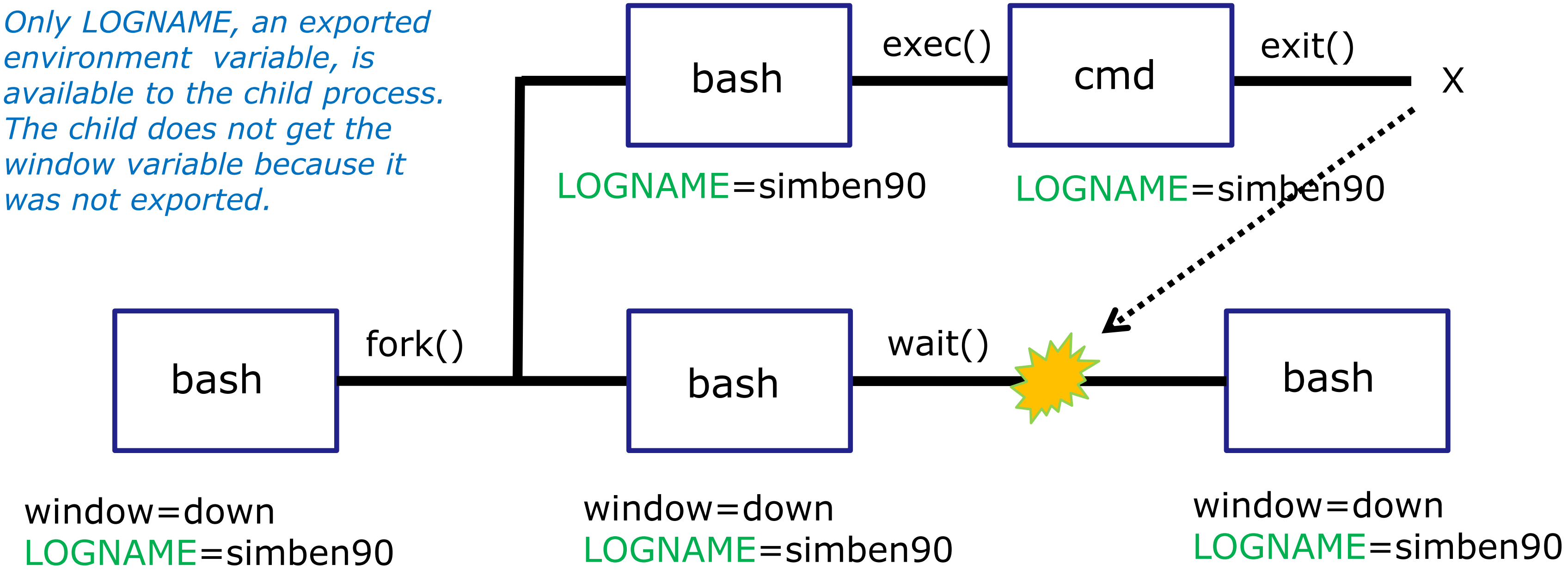
```
/home/cis90/simben $ bash
[simben@opus ~]$ echo $window $LOGNAME
simben90
[simben@opus ~]$ exit
exit
/home/cis90/simben $
```

Running the bash command starts another bash process as a child of the current bash process. LOGNAME has a value, but there is no window variable.

IMPORTANT OBSERVATION: Only LOGNAME, an exported environment variable, is available to the child process. The child does not get the window variable because it was not exported.

Only exported variables are available to the child

IMPORTANT OBSERVATION: Only LOGNAME, an exported environment variable, is available to the child process. The child does not get the window variable because it was not exported.



- When a shell forks a child, not all of the variables are passed on to the child.
- Only copies of the parent's exported variables are passed to the child.

The rules of the road for variables

- When a shell forks a child, only copies of exported variables are made available to the child.

- A child can modify the variables it receives but those modifications will not change the parent’s variables.

Changes made by the child do not affect the parent

1 parent

```
/home/cis90/simben $ echo $window
down
/home/cis90/simben $ export window
```

export window so it is available to children

2 child

```
/home/cis90/simben $ bash
[simben@opus ~]$ echo $window
down
```

a copy of window is now available to the child process

3 child

```
[simben@opus ~]$ window=up
[simben@opus ~]$ echo $window
up
[simben@opus ~]$ exit
exit
```

the child modifies the window variable

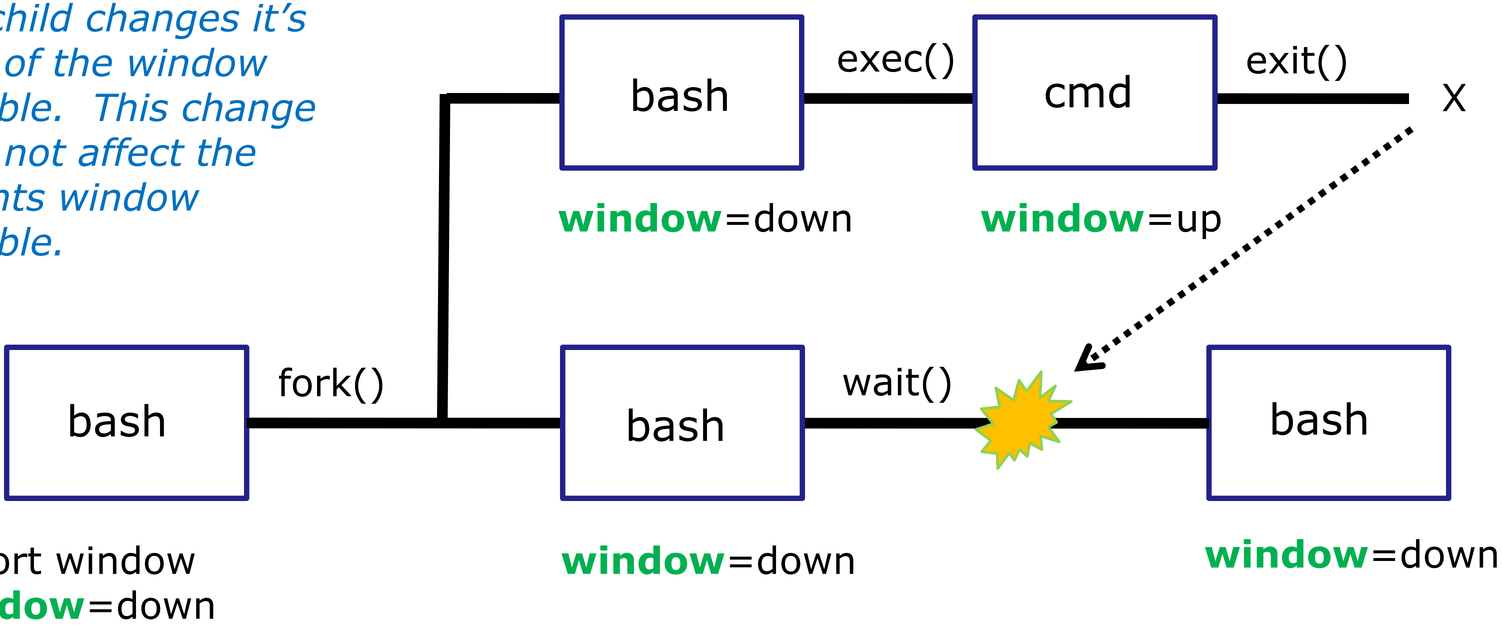
4 parent

```
/home/cis90/simben $ echo $window
down
```

The modifications made by the child do not affect the parent's variable

Changes made by the child do not affect the parent

The child changes it's copy of the window variable. This change does not affect the parents window variable.



- A child can modify the variables it receives but those modifications will not change the parent's variables.

aliases

alias command

(a shell builtin)

```
alias [-p] [name[=value] ...]
              Alias with no arguments or with the -p option prints the list
              of aliases in the form alias name=value on  standard  output.
              When  arguments  are  supplied,  an alias is defined for each
              name whose value is given.  A trailing space in  value causes
              the  next  word to be checked for alias substitution when the
              alias is expanded.  For each name in the  argument  list  for
              which  no  value is supplied, the name and value of the alias
              is printed.  Alias returns true unless a name  is  given  for
              which no alias has been defined.

              Note  aliases  are not expanded by default in non-interactive
              shell, and it can be enabled by  setting  the  expand_aliases
              shell option using shopt.
```

Now you can give your own name to commands!

alias command

Example: Make a new name for the cp command

1

```
/home/cis90/simben $ alias copy=cp
/home/cis90/simben $ copy lab09 /home/rsimms/cis90/lab09.$LOGNAME
/home/cis90/simben $
```

2

```
/home/cis90/simben $ type copy
copy is aliased to `cp'
/home/cis90/simben $
```

*The **type** command shows that copy is an alias*

3

```
/home/cis90/simben $ alias copy
alias copy='cp'
/home/cis90/simben $
```

*The **alias** command (without an "=" sign) shows what the alias is*

4

```
/home/cis90/simben $ unalias copy
/home/cis90/simben $ alias copy
-bash: alias: copy: not found
```

*Use **unalias** command to remove an alias*

alias command

Example: Make an alias, called s, that prints the first 5 lines of smalltown

1

```
/home/cis90/simben $ alias s="clear; head –n5 ~/edits/small_town"
/home/cis90/simben $ s
HOW SMALL IS SMALL?

YOU KNOW WHEN YOU'RE IN A SMALL TOWN WHEN...
   The airport runaway is terraced.
   The polka is more popular than a moshpit on Saturday night.
/home/cis90/simben $
```

2

```
/home/cis90/simben $ type s
s is aliased to `clear; head -10 ~/edits/small_town'
/home/cis90/simben $ alias s
alias s='clear; head -10 ~/edits/small_town'
```

*The **type** and **alias** commands show that s is an alias*

3

```
/home/cis90/simben $ unalias s
/home/cis90/simben $
```

*Use **unalias** command to remove an alias*

alias an alias

Yes, an alias can be made using another alias

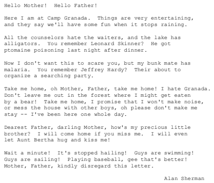
1

```
/home/cis90/simben $ alias show=cat
/home/cis90/simben $ alias view=show
```

*Make **show** an alias of **cat***
*Make **view** and alias of **show***

2

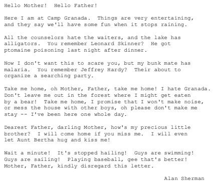
```
/home/cis90/simben $ show letter
```



```
Hello Mother!  Hello Father!

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

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!

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

                                              Alan Sherman
```

reduced sized to fit on page

```
/home/cis90/simben $ view letter
```



```
Hello Mother!  Hello Father!

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

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!

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

                                              Alan Sherman
```

reduced sized to fit on page

*Now, either **show letter** or **view letter** will cat out the letter file*

3

```
/home/cis90/simben $ unalias show
/home/cis90/simben $ alias view
alias view='show'
/home/cis90/simben $ view letter
-bash: show: command not found
/home/cis90/simben $
```

It can be broken too

single and double quotes (**very subtle**)

You can control whether bash does filename expansion when you create the alias or … when the alias is used

```
$ ac=on
$ fan=medium
$ defrost=off
```

double *single*

```
1   $ alias p="echo $ac $fan $defrost"     $ alias p='echo $ac $fan $defrost'
    $ alias p                               $ alias p
    alias p='echo on medium off'            alias p='echo $ac $fan $defrost'

2   $ p                                     $ p
    on medium off                           on medium off

3   $ ac=off                                $ ac=off

4   $ p                                     $ p
    on medium off                           off medium off
```

Note: using single quotes prevents bash from expanding the variables when creating up the alias

Class Exercise

Make some aliases

For example:

- **alias mypath="echo $PATH"**
- **mypath**

- **alias details=file**
- **details /usr/bin/spell**

Now invent 1-2 of your own

bash startup files

bash startup files

only executed when logging in

/etc/profile (system wide)

- adds root's special path

/etc/profile.d/*.sh (system wide)

- kerberos directories added to path
- adds color, vi aliases
- language, character sets

.bash_profile (user specific)

- set up your path, prompt and other environement variables

.bashrc (user specific)

- add your new aliases here

Edit these files to customize your shell environment

/etc/bashrc (system wide)

- changes umask to 0002 for regular users
- sets final prompt string

.bash_profile

.bash_profile

- The *.bash_profile* is a shell script that sets up a user's shell environment.
- This script is executed each time the user logs in.
- The *.bash_profile* is used for initializing shell variables and running basic commands like umask or set -o options.
- This script also runs the users *.bashrc* file

.bash_profile for CIS 90 (runs only at login)

```
[simben@opus ~]$ cat .bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/../bin:$HOME/bin:.
BASH_ENV=$HOME/.bashrc
USERNAME=""
PS1='$PWD $ '
export USERNAME BASH_ENV PATH
umask 002
set -o ignoreeof
stty susp ^F
eval `tset -s -m vt100:vt100 -m :\?${TERM:-ansi} -r -Q `

[simben@opus ~]$
```

sources the .bashrc file

Appends the CIS 90 bin, the user's bin and the "current" directories to the path

Prompt (PS1) used in CIS 90 is specified

variables are exported

umask value is set

EOF's are ignored

Suspend character redefined from Z to F

Terminal type is set

.bashrc

.bashrc

- The .*bashrc* is a shell script that is executed during user login and whenever a new shell is invoked

- Good place to add user defined aliases

.bashrc

The *.bashrc* is a shell script that is executed during user login and whenever a new shell is invoked. This file usually contains the user defined aliases.

```
[simben@opus ~]$ cat .bashrc
# .bashrc

# User specific aliases and functions

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
alias print="echo -e"
[simben@opus ~]$
```

sources the /etc/bashrc file

creates a print alias, the -e option enables interpretation of backslash escapes

Class Exercise

Modify .bashrc

Add a new permanent alias to your bash environment

```
  alias me="finger $LOGNAME"
```

When finished logout and login again and verify the alias is permanent.

. and exec

. and exec

In normal execution of a UNIX command, shell-script or binary, the child process in unable to affect the login shell environment.

Sometimes it is desirable to run a shell script that will initialize or change shell variables in the parent environment. To do this, the shell (bash) provides a **.** (dot) or **source** command, which instructs the shell to execute the shell script itself, without spawning a child process to run the script, and then continue on where it left off.

. ***myscript***
source ***myscript*** } *equivalent*

In this example, the commands in the file script are run by the parent shell, and therefore, any changes made to the environment will last for the duration of the login session.

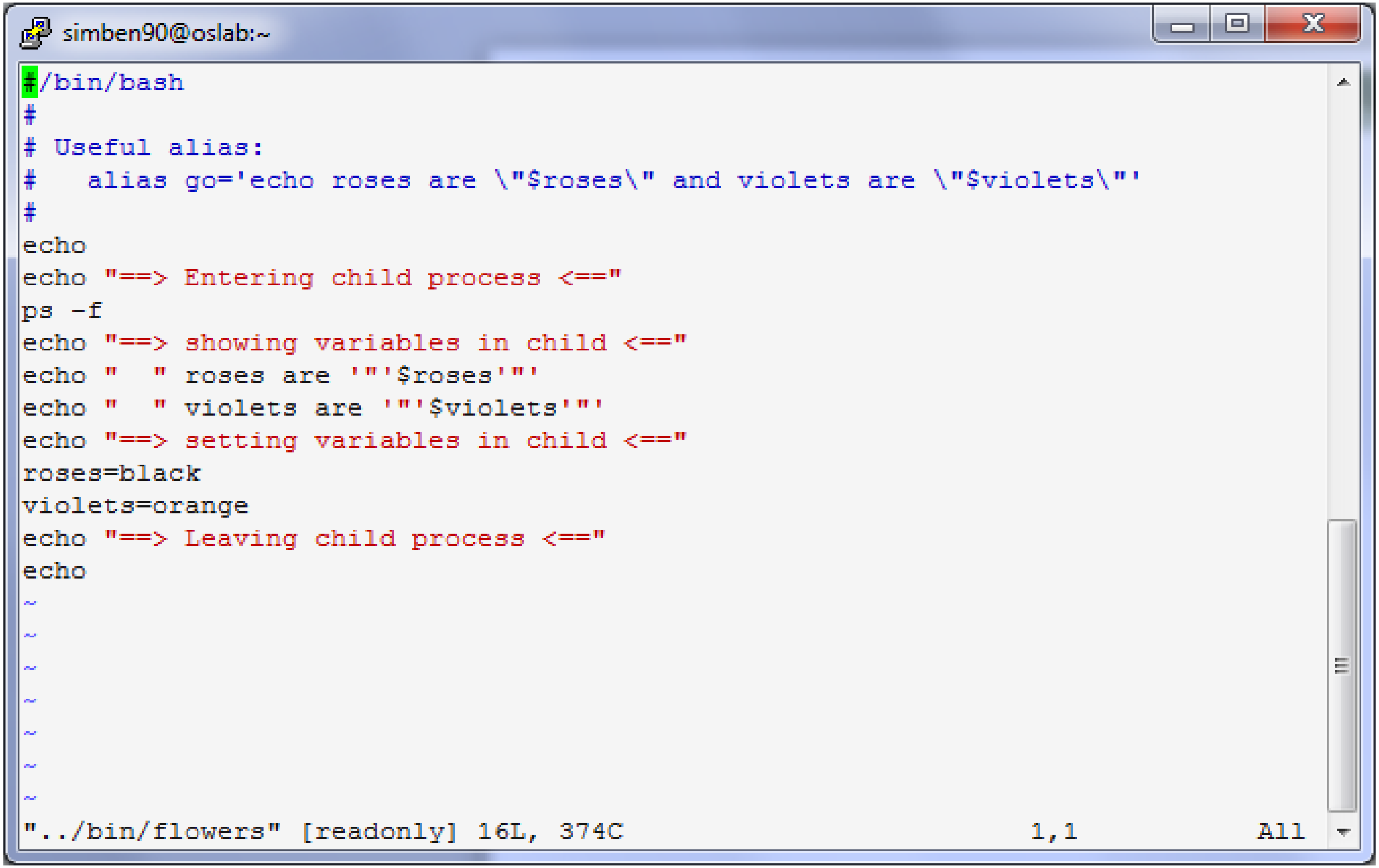
If a UNIX command is run using the exec command, the bash code in the process is overlaid by the command code, when finished the process will terminate

exec clear

This will have the effect of clearing the screen and logging off the computer

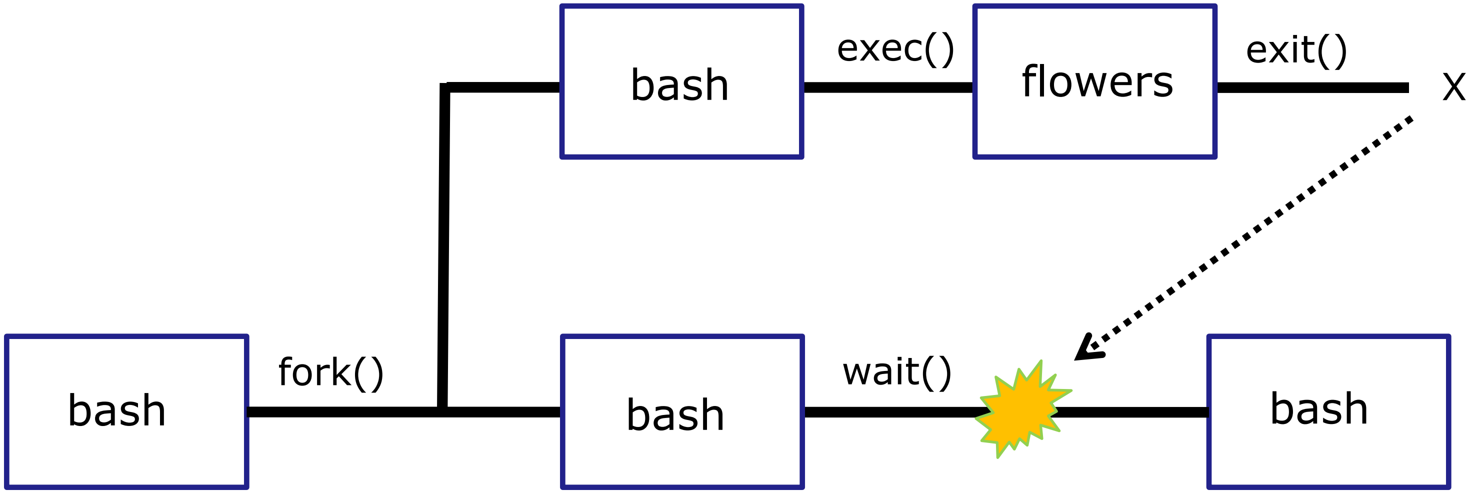
grok this lesson?

/home/cis90/bin/flowers



```
#/bin/bash
#
# Useful alias:
#    alias go='echo roses are \"$roses\" and violets are \"$violets\"'
#
echo
echo "==> Entering child process <=="
ps -f
echo "==> showing variables in child <=="
echo "   " roses are '"'$roses'"'
echo "   " violets are '"'$violets'"'
echo "==> setting variables in child <=="
roses=black
violets=orange
echo "==> Leaving child process <=="
echo
~
~
~
~
~
~
~
"../bin/flowers" [readonly] 16L, 374C                         1,1           All
```

running the flowers script



Use the flowers script in /home/cis90/bin to test your understanding of variables and child processes

Create alias to show variables

```
/home/cis90/simben $ alias go='echo roses are \"$roses\" and
violets are \"$violets\"'

/home/cis90/simben $ go
roses are "" and violets are ""
```

Copy and paste the alias command in the comments of flowers.

This alias shows the value of the roses and violets variables by typing ***go***

Create and initialize variables

```
/home/cis90/simben $ roses=red
/home/cis90/simben $ go
roses are "red" and violets are ""
```

Now the roses variable has been created and initialized

```
/home/cis90/simben $ violets=blue
/home/cis90/simben $ go
roses are "red" and violets are "blue“
```

Now the violets variable has been created and initialized

Unset variables

```
/home/cis90/simben $ unset roses
/home/cis90/simben $ go
roses are "" and violets are "blue"
```

Now the roses variable no longer exists

```
/home/cis90/simben $ unset violets
/home/cis90/simben $ go
roses are "" and violets are ""
```

Now the violets variable no longer exists

Create and initialize variables again

```
/home/cis90/simben $ roses=red; violets=blue
/home/cis90/simben $ go
roses are "red" and violets are "blue"
```

Now both variables have been created and initialized again

Run flowers script as a child process

(variables not exported)

```
/home/cis90/simben $ go
roses are "red" and violets are "blue"
```

The parent sees roses and violets

```
/home/cis90/simben $ flowers

==> Entering child process <==
UID        PID  PPID  C STIME TTY          TIME CMD
simben90 20864 20863  0 07:50 pts/0    00:00:00 -bash
simben90 20956 20864  0 08:10 pts/0    00:00:00 -bash
simben90 20963 20956  3 08:10 pts/0    00:00:00 ps -f
```

parent

child (flowers)

ps command

```
==> showing variables in child <==
   roses are ""
   violets are ""
==> setting variables in child <==
==> Leaving child process <==
```

The child does not see roses or violets

```
/home/cis90/simben $ go
roses are "red" and violets are "blue"
```

The variables are unchanged after running flowers script

Run flowers script as a child process

(roses variable exported)

```
/home/cis90/simben $ export roses
/home/cis90/simben $ go
roses are "red" and violets are "blue"

/home/cis90/simben $ flowers

==> Entering child process <==
UID        PID  PPID  C STIME TTY          TIME CMD
simben90 20864 20863  0 07:50 pts/0    00:00:00 -bash
simben90 21023 20864  0 08:22 pts/0    00:00:00 -bash
simben90 21030 21023  1 08:22 pts/0    00:00:00 ps -f
==> showing variables in child <==
   roses are "red"
   violets are ""
==> setting variables in child <==
==> Leaving child process <==

/home/cis90/simben $ go
roses are "red" and violets are "blue"
```

The parent sees roses and violets

parent

child (flowers)

ps command

The child now sees roses since it was exported

The variables are unchanged after running flowers script

Run flowers script as a child process
(scripted sourced)

```
/home/cis90/simben $ go
roses are "red" and violets are "blue"

/home/cis90/simben $ source flowers

==> Entering child process <==
UID        PID  PPID  C STIME TTY          TIME CMD
simben90 20864 20863  0 07:50 pts/0    00:00:00 -bash
simben90 21043 20864  0 08:24 pts/0    00:00:00 ps -f
==> showing variables in child <==
   roses are "red"
   violets are "blue"
==> setting variables in child <==
==> Leaving child process <==

/home/cis90/simben $ go
roses are "black" and violets are "orange"
/home/cis90/simben $
```

The parent sees roses and violets

script is not running as child

The script now sees roses and violets because it is running in the parent process

The variables are changed after running flowers script

Wrap up

Lab 10 - the last one!

CIS 90 Linux Lab Exercise

Lab 10: The Shell Environment
Spring 2012

Lab 10: The Shell Environment

In this lab you will customize your login environment to suit your needs and preferences. By modifying environment variables and editing your *.bash_profile* and *.bashrc* files, you will customize your shell environment in a number of different ways.

Forum

Browse to: http://opus.cabrillo.edu/forum/viewforum.php?f=46

Check the forum for any late breaking news about this lab. The forum is also the place to go if you get stuck, have a question or want to share something you have learned about this lab.

Procedure

Log on to Opus so that you have a command line shell at your service. Start this lab from your home directory.

Environment Variables

1. Display the contents of your PWD environment variable. Change to your *bin* subdirectory and display the same variable. How did it change?
2. Change back to your home directory.
3. Display the contents of your PATH environment variable. Note the colon (:) separating the different directory names. What is the last directory in which the system searches for commands?
4. Make a new environment variable called GREETING and assign it an appropriate salutation. Don't forget to use quotes if your message has whitespace in it.
5. Use the **env** command to see if it is in your environment. Is it there? What must you do to put it in the environment?
6. Export the variable GREETING and use **env** to verify it's there.
7. Invoke a new bash shell process by typing:
   bash
   Now use the **unset** command to unset the variable PS1. What Happened?
8. Reset the PS1 variable by entering the following command:
   PS1="Yes master: "
   What happens to your primary prompt?

Extra Credit Special

1) Why did the prompt change?

```
/home/cis90/simben $ bash
[simben@opus ~]$ exit
exit
/home/cis90/simben $
```

2) What command could be issued prior to the bash command above that would prevent the prompt from changing?

For 3 points extra credit, email risimms@cabrillo.edu answers to both questions before the next class starts

New commands:

.	- source the commands
alias	- create or show an alias
unalias	- remove an alias
set	- show all variables
env	- show environment variables
export	- export variable so child can use
exec	- replace with new code
source	- same as .

New Files and Directories:

.bash_profile	- executed at login
.bashrc	- executed at login and new shells

Next Class

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

Lab 10

Quiz questions for next class:

- How do you make an alias setting permanent?
- What must you do to a variable so a child can use it?
- How would you use an alias to make a command named copy … that would do what the cp command does?

Backup