Lesson Module Checklist
- Slides
- WB

- Flash cards
- Page numbers
- 1st minute quiz
- Web Calendar summary
- Web book pages
- Commands

- Lab 10 and Final Project uploaded
- riddle file copied to class bin directory
- allscripts updated
- myscript in depot
- flowers in depot
- sample myscripts for Benji and Homer

- Materials uploaded
- Backup slides, CCC info, handouts on flash drive
- Check that backup room headset is charged
- 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/)

CIS 90 - Lesson 12

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

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

Quiz

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

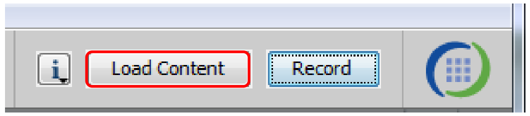
See electronic white board

email answers to: `risimms@cabrillo.edu`

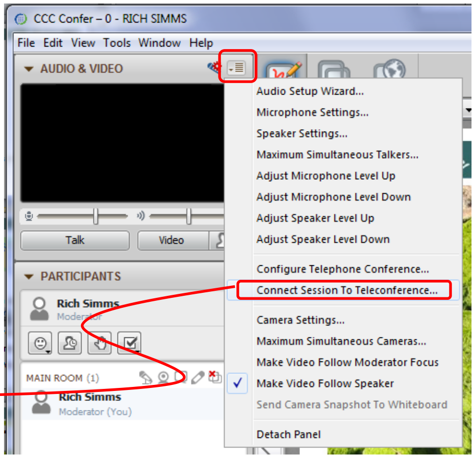
(answers must be emailed within the first few minutes of class for credit)

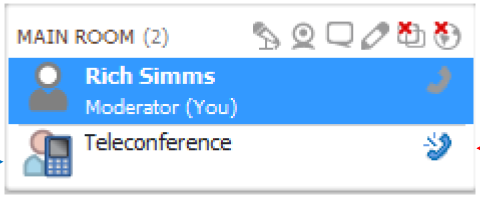
CIS 90 - Lesson 12

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



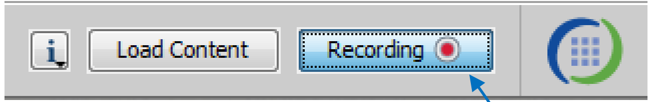
[ ] Connect session to Teleconference


CCC Confer – 0 - RICH SIMMS
File Edit View Tools Window Help
AUDIO & VIDEO
Talk
Video
PARTICIPANTS
Rich Simms
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



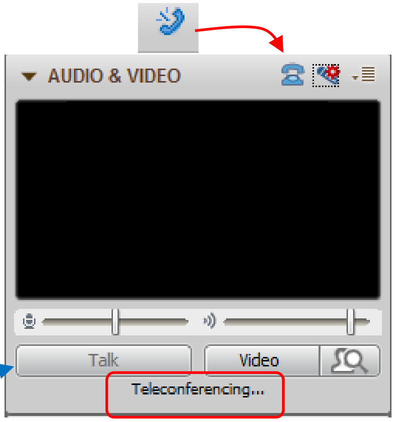
Session now connected to teleconference

[ ] Is recording on?



Red dot means recording

[ ] Use teleconferencing, not mic



Should be greyed out

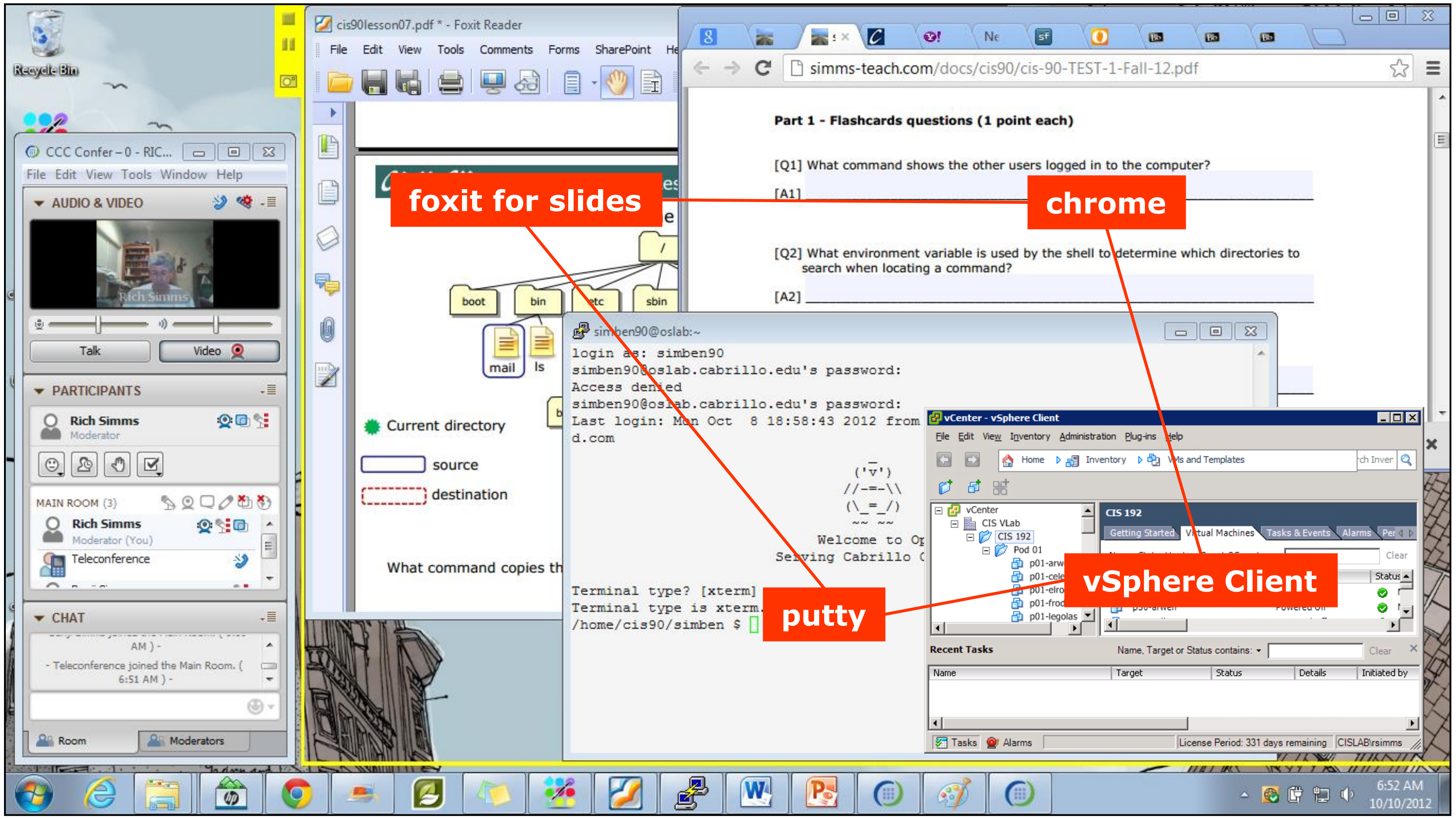
Cabrillo College est. 1959

CIS 90 - Lesson 12

CCC Confer

[ ] **Video (webcam) optional**

[ ] **layout and share apps**



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]

What command copies th

```
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
d.com

                                        ('v')
                                       //-=-\\
                                       (\_=_/)
                                        ~~ ~~
                                   Welcome to Op
                                Serving Cabrillo C

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

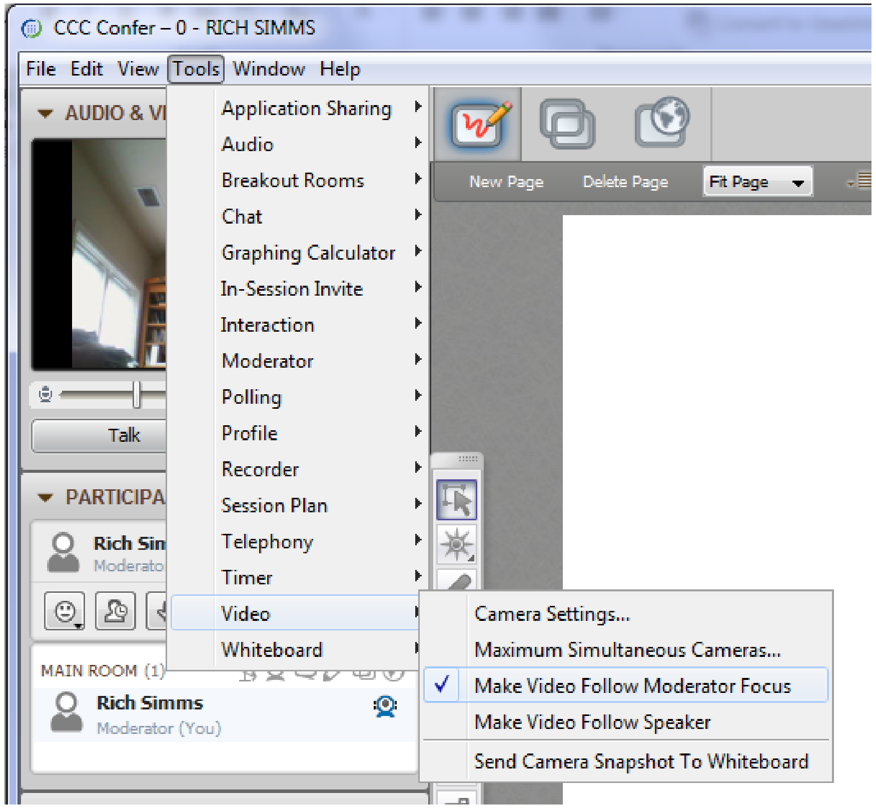
CCC Confer

Don't Forget

[ ] Video (webcam) optional

[ ] Follow moderator

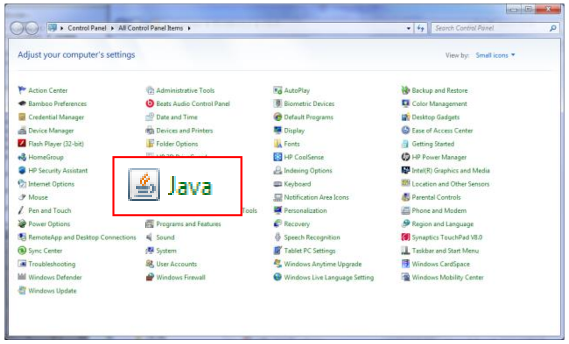
[ ] Double-click on postages stamps



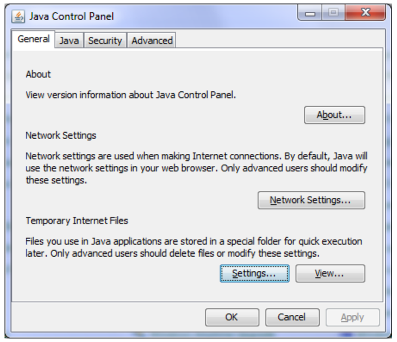
Universal Fix for CCC Confer:

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

Control Panel (small icons)



General Tab > Settings…



About

View version information about Java Control Panel.

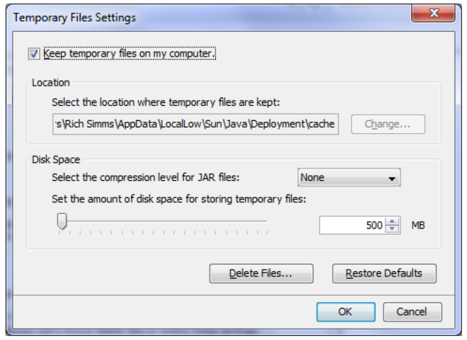
Network Settings

Network settings are used when making Internet connections. By default, Java will use the network settings in your web browser. Only advanced users should modify these settings.

Temporary Internet Files

Files you use in Java applications are stored in a special folder for quick execution later. Only advanced users should delete files or modify these settings.

500MB cache size



[✓] Keep temporary files on my computer.

Location

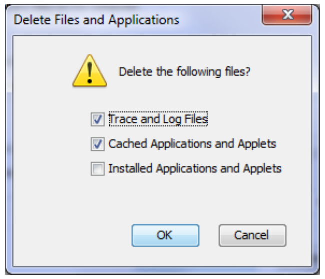
Select the location where temporary files are kept: s\Rich Simms\AppData\LocalLow\Sun\Java\Deployment\cache

Disk Space

Select the compression level for JAR files: None

Set the amount of disk space for storing temporary files: 500 MB

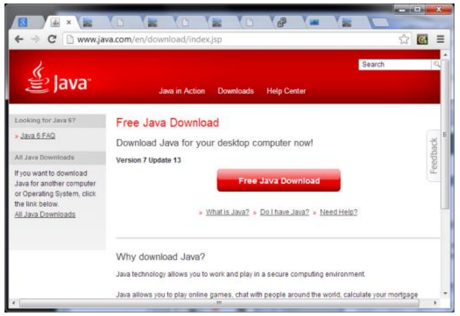
Delete these



Delete the following files?

[✓] Trace and Log Files
[✓] Cached Applications and Applets
[ ] Installed Applications and Applets

Google Java download



If you want to download Java for another computer or Operating System, click the link below.

Free Java Download

Download Java for your desktop computer now!

Version 7 Update 13

Why download Java?

Java technology allows you to work and play in a secure computing environment.

Java allows you to play online games, chat with people around the world, calculate your mortgage

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?

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

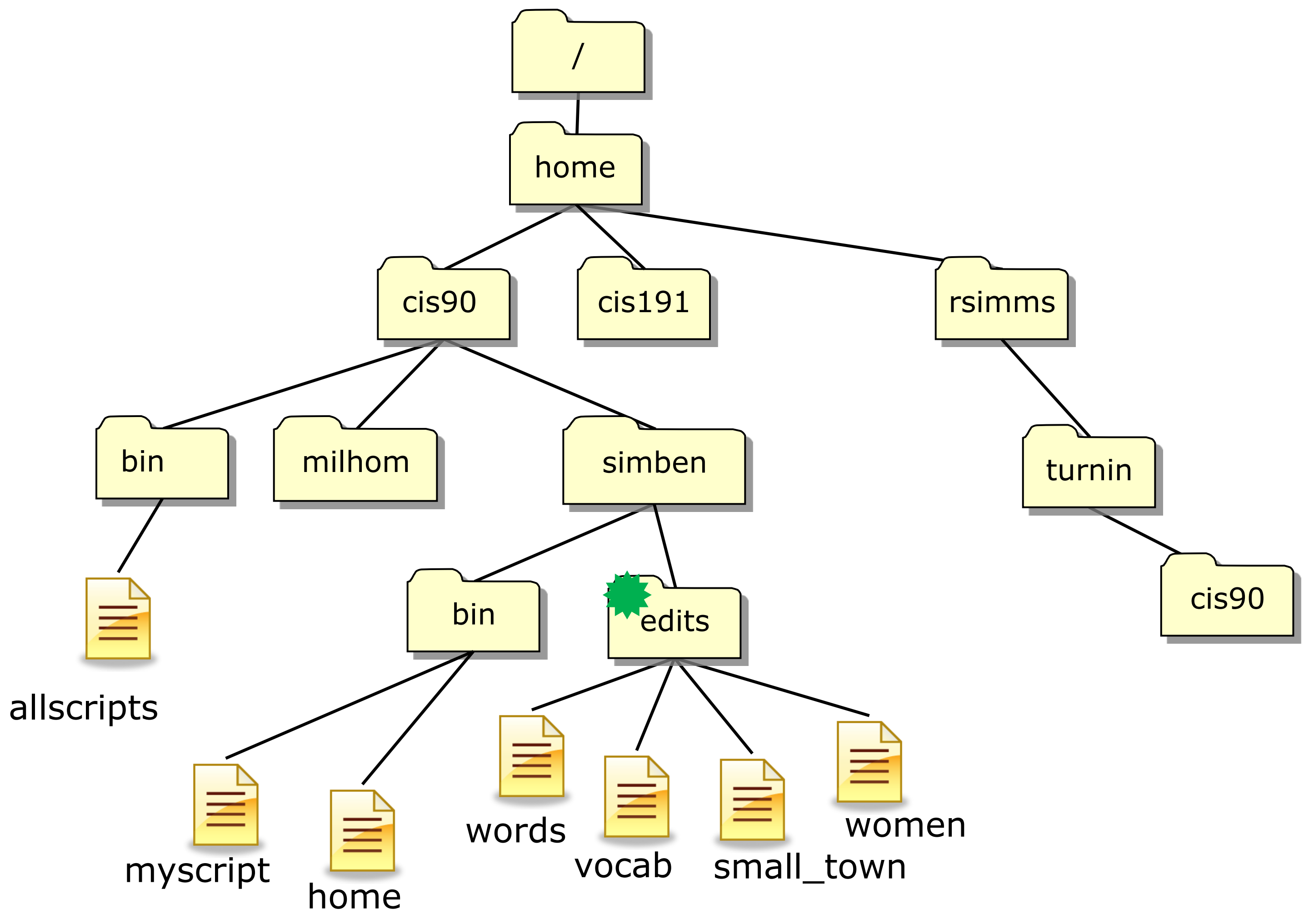
Chinese Proverb	他問一個問題，五分鐘是個傻子，他不問一個問題仍然是一個傻瓜永遠。 *He who asks a question is a fool for five minutes; he who does not ask a question remains a fool forever.*

Submitting Lab 9 & PATHNAMES!

REMINDER

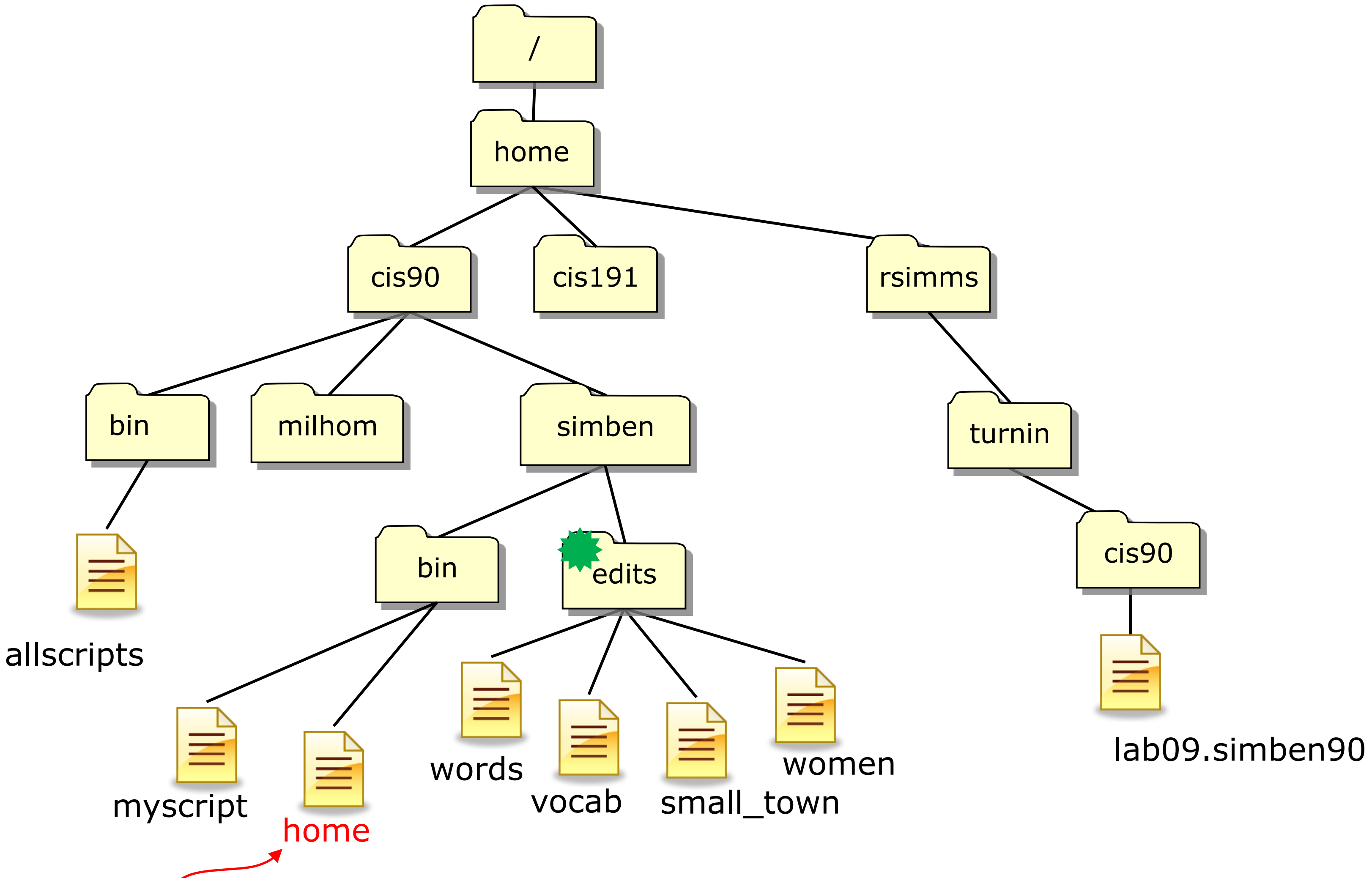
- You must **ALWAYS** use **VALID PATHNAMES** when specifying files as **ARGUMENTS** on a command.
- Pathnames can be relative or absolute.
- A common mistake in the past on Lab 9 is to ignore error messages and not submit all the files requested.

CIS 90 - Lesson 12



```
cat home words vocab small_town women > /home/rsimms/turnin/cis90/lab09.$LOGNAME
```

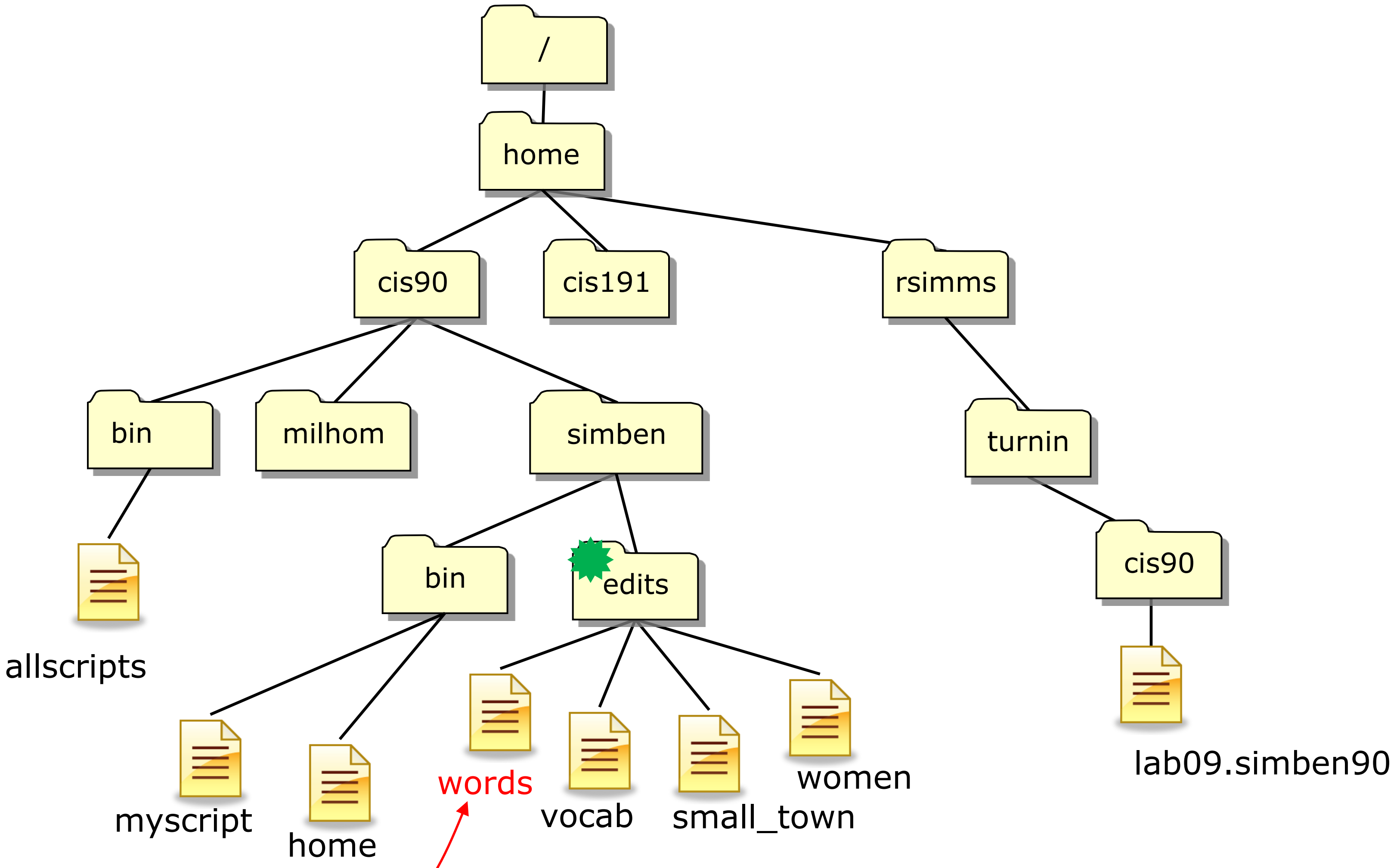
Why will this command NOT work for turning in Lab 9?



```
cat ../bin/home words vocab small_town women > /home/rsimms/turnin/cis90/lab09.$LOGNAME
```

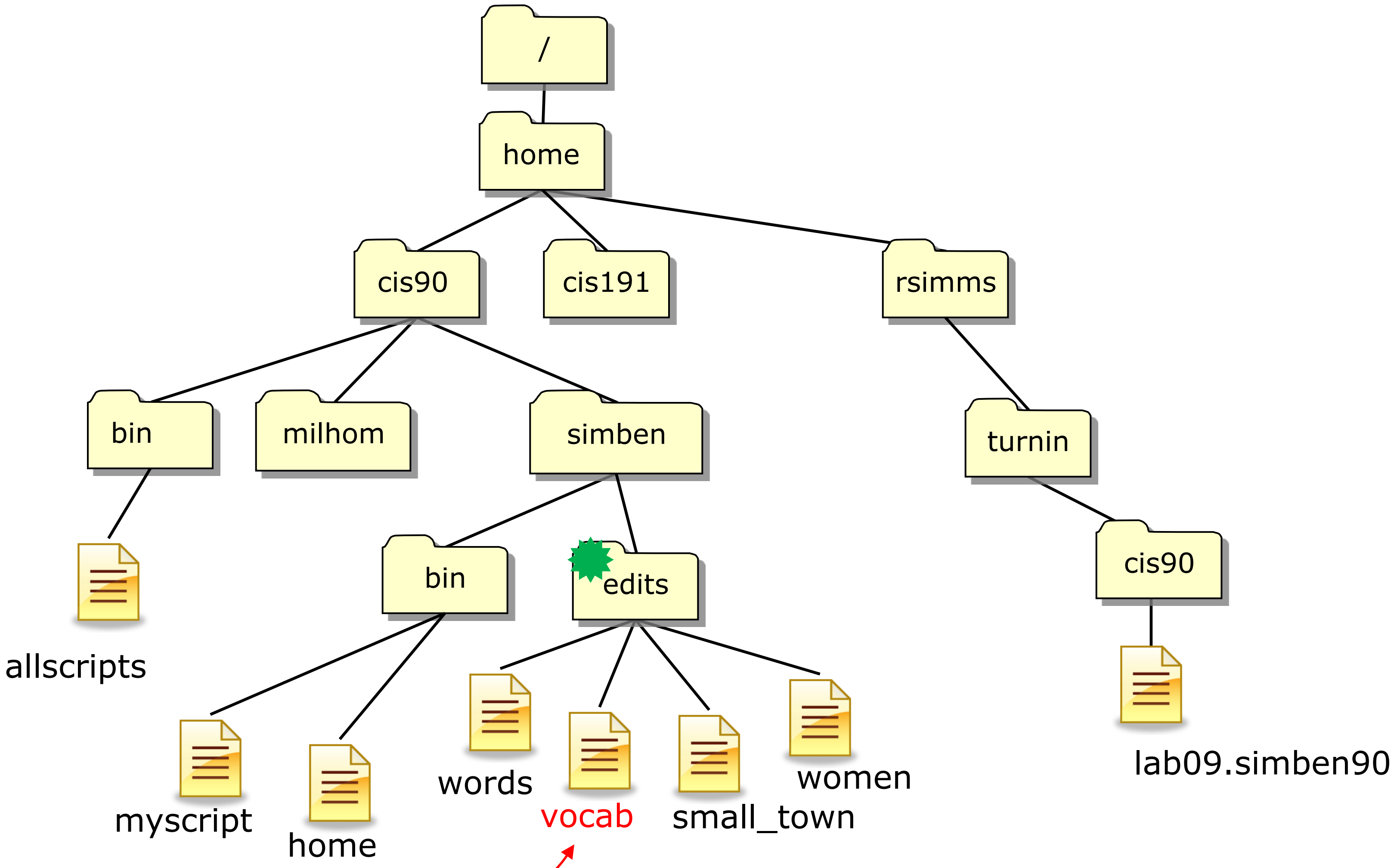
relative pathname to home in the bin directory

CIS 90 - Lesson 12



```
cat ../bin/home words vocab small_town women > /home/rsimms/turnin/cis90/lab09.$LOGNAME
```

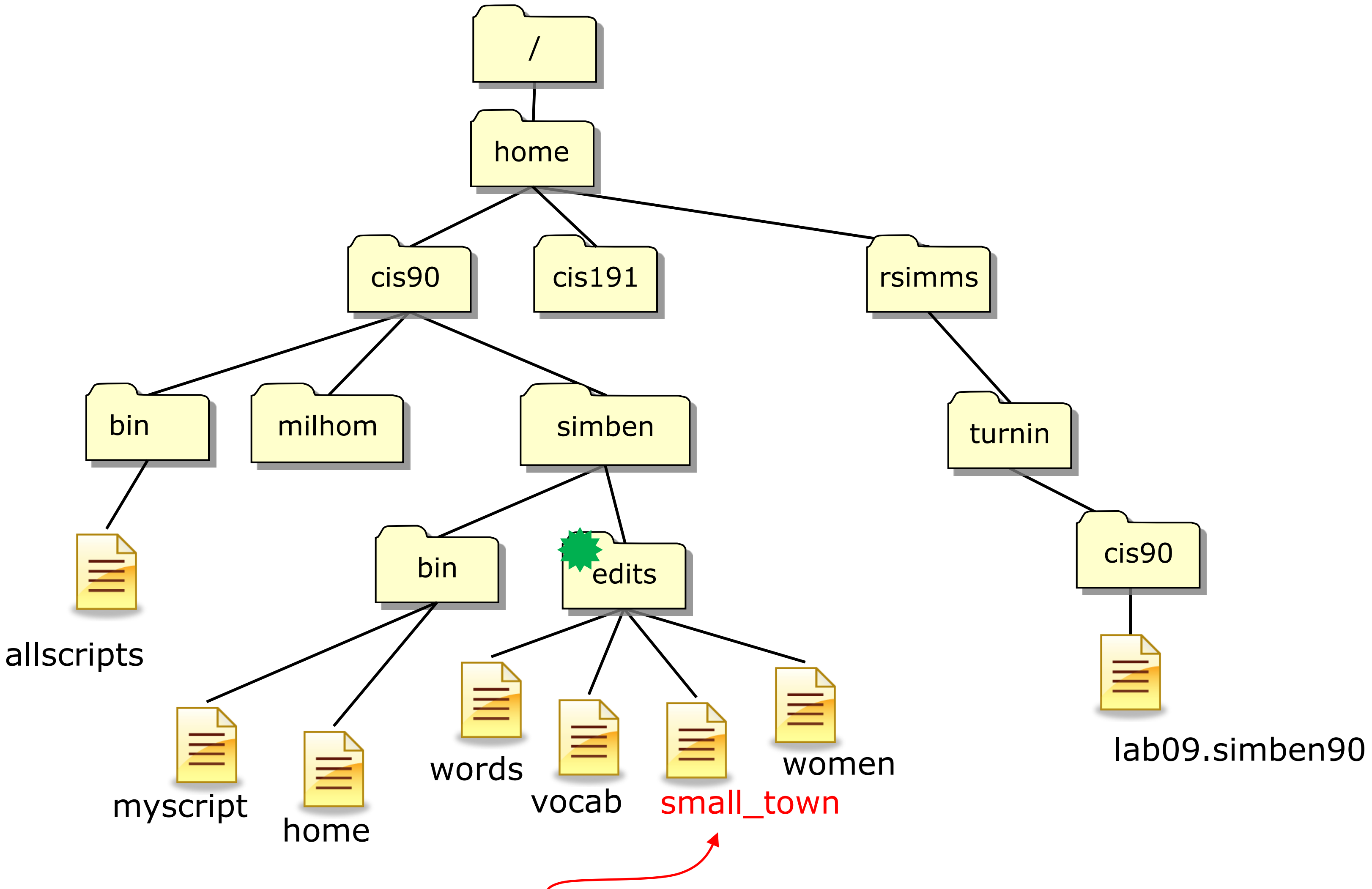
relative pathname



```
cat ../bin/home words vocab small_town women > /home/rsimms/turnin/cis90/lab09.$LOGNAME
```

relative pathname

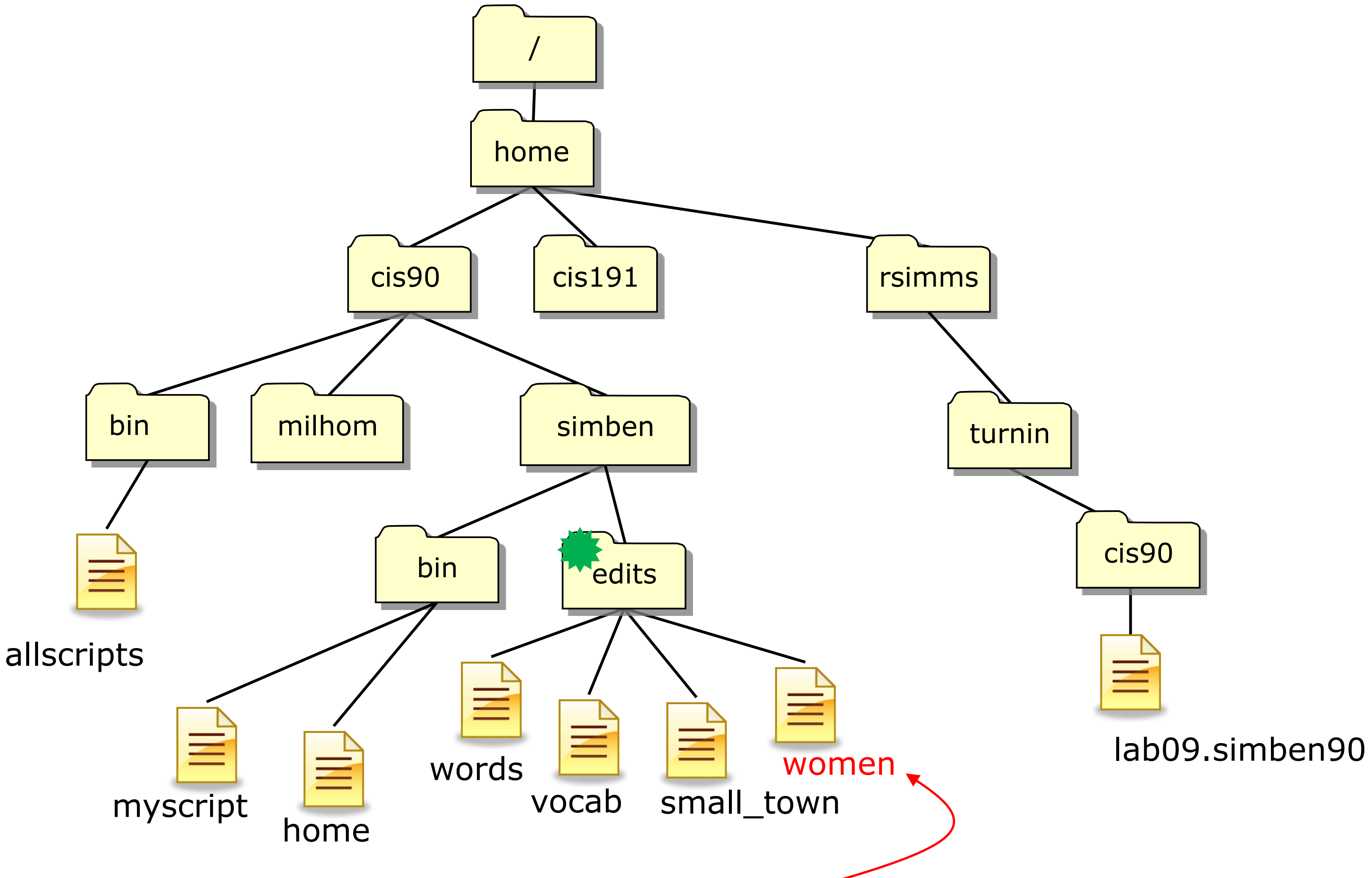
CIS 90 - Lesson 12



```
cat ../bin/home words vocab small_town women > /home/rsimms/turnin/cis90/lab09.$LOGNAME
```

relative pathname

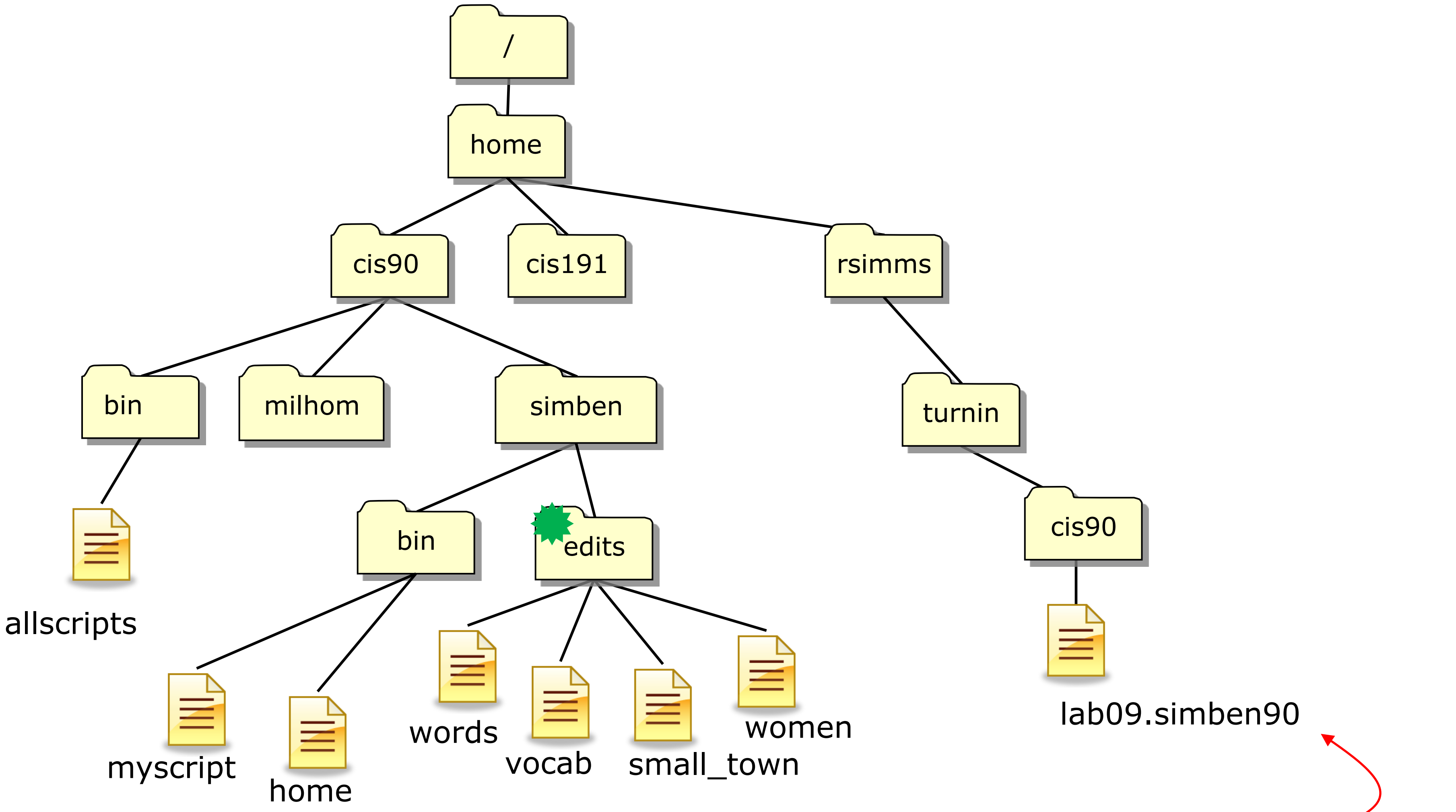
CIS 90 - Lesson 12



```
cat ../bin/home words vocab small_town women > /home/rsimms/turnin/cis90/lab09.$LOGNAME
```

relative pathname

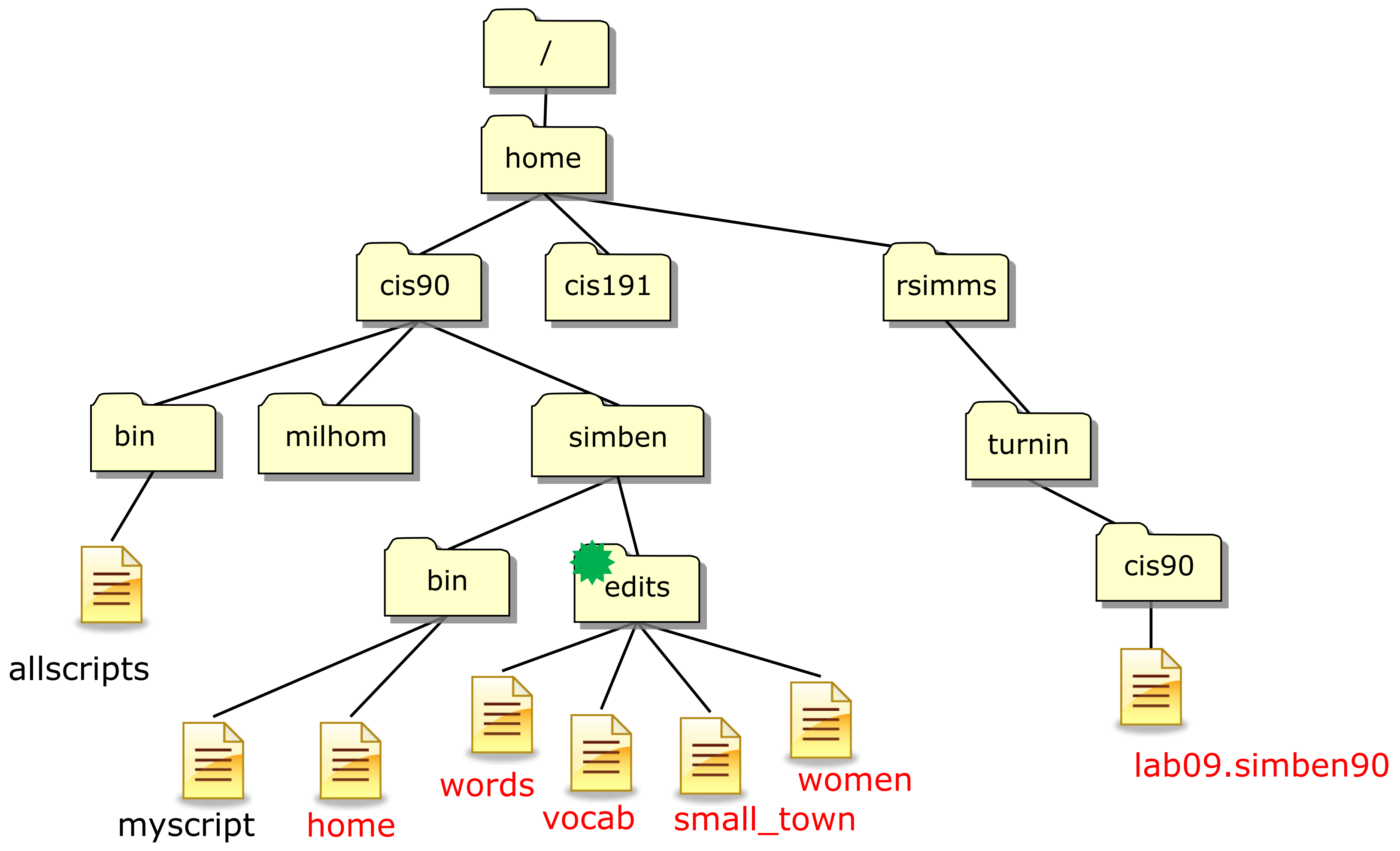
CIS 90 - Lesson 12



```
cat ../bin/home words vocab small_town women > /home/rsimms/turnin/cis90/lab09.$LOGNAME
```

absolute pathname

CIS 90 - Lesson 12



A much better way to do this:

```
cat ../bin/home words vocab small_town women > lab09
less lab09
cp lab09 /home/rsimms/turnin/cis90/lab09.$LOGNAME
```

Lets you review your work so you know what you are turning in

Housekeeping

Previous material and assignment

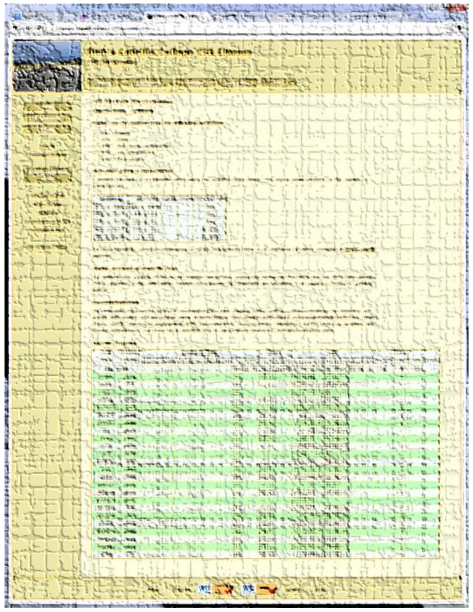
1. Lab 9 due 11:59PM tonight

2. Five posts due 11:59PM tonight

Reminder:
Only posts between in the CIS 90 forum between 4/12 and 5/9 (inclusive) are counted.

Managing your grade

Use the web page



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

Use Jesse's checkgrades script

```
adaldrida: 73% (268 of 364 points)
anborn: 101% (368 of 364 points)
arador: 50% (184 of 364 points)
aragorn: 58% (213 of 364 points)
balrog: 0% (0 of 364 points)
bilbo: 90% (330 of 364 points)
bombadil: 7% (28 of 364 points)
celebrian: 65% (237 of 364 points)
cirdan: 52% (190 of 364 points)
durin: 86% (316 of 364 points)
dwalin: 92% (335 of 364 points)
elrond: 104% (380 of 364 points)
eomer: 101% (370 of 364 points)
faramir: 106% (389 of 364 points)
frodo: 100% (364 of 364 points)
gimli: 64% (233 of 364 points)
goldberry: 86% (314 of 364 points)
gwaihir: 72% (264 of 364 points)
haldir: 64% (235 of 364 points)
ingold: 89% (327 of 364 points)
ioreth: 95% (346 of 364 points)
legolas: 101% (371 of 364 points)
marhari: 91% (334 of 364 points)
pallando: 77% (282 of 364 points)
quickbeam: 46% (171 of 364 points)
samwise: 96% (352 of 364 points)
sauron: 91% (333 of 364 points)
shadowfax: 78% (284 of 364 points)
strider: 103% (377 of 364 points)
theoden: 91% (332 of 364 points)
treebeard: 95% (349 of 364 points)
tulkas: 98% (360 of 364 points)
```

As of May 8, 2013

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

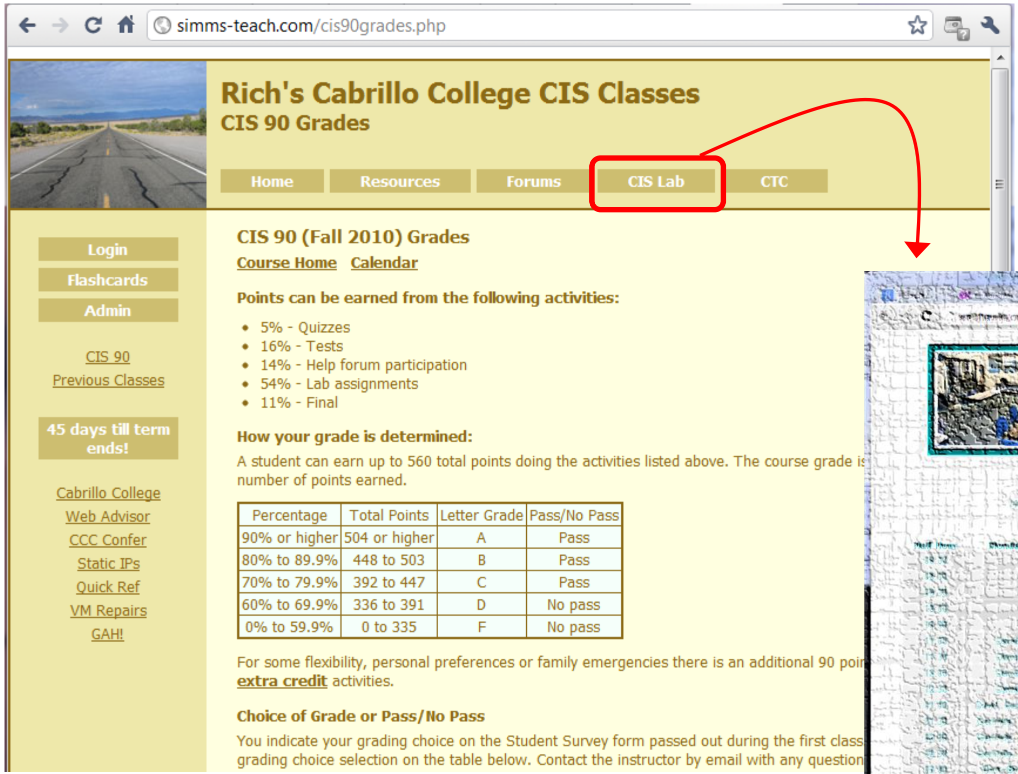
Points yet to earn
- 2 quizzes - 6 points
- 1 test - 30 points
- 2 forum periods - 40 points
- 2 labs - 60 points
- 1 final project - 60 points

} 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 (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

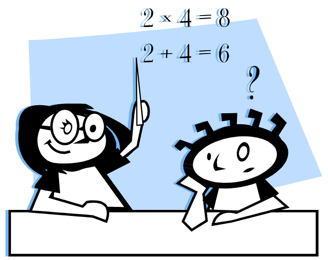
Cabrillo Network & Systems Technology Lab
Fall 2012 Instructor and Lab Assistant Hours

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 in Room 2501 (right after class) or TBA (contact me)
- Ask questions on the Forum at:
  http://oslab.cabrillo.edu/forum/

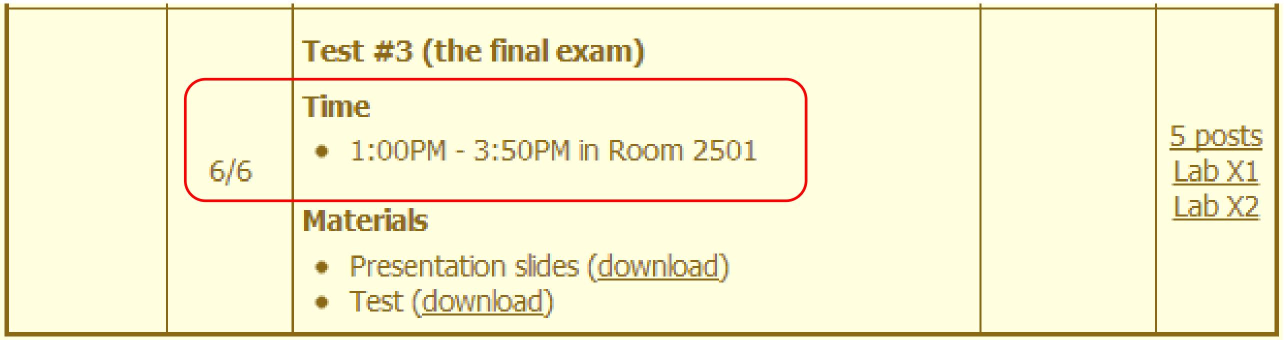


Final Exam

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

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



	6/6	**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

A Tangent on Spell (from last lesson)

Soquel is not in 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
```

Question: How can we add Soquel to the UNIX dictionary so it is ignored in future spell checks?

Question: How can we add Soquel to the UNIX dictionary so it is ignored in future spell checks?

```
/home/cis90/simben $ man spell
No manual entry for spell
```

Hmmm. No man page for spell - weird!

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

Where is it on our path?

```
/home/cis90/simben $ file usr/bin/spell
/usr/bin/spell: Bourne shell script text executable
```

So what kind of file is it?

```
/home/cis90/simben $ cat /usr/bin/spell
#!/bin/sh

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

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

Ah ha, it's a script, so lets look at it …

Well … son of a gun, the actual command is ***aspell!***

Question: How can we add Soquel to the UNIX dictionary so it is ignored in future spell checks?

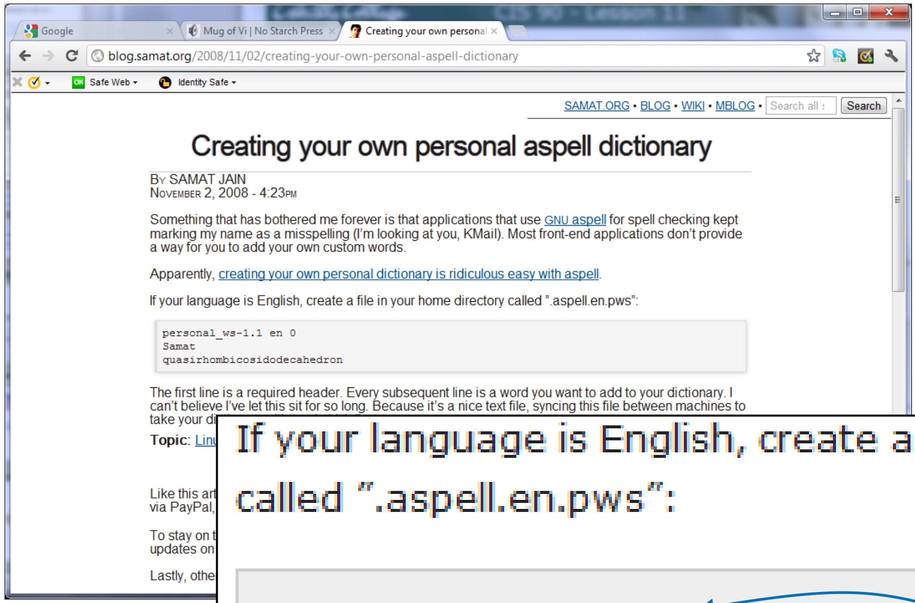
```
ASPELL(1)              Aspell Abbreviated User's Manual              ASPELL(1)

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.

<snipped>

       --home-dir=<directory>
              Directory Location for personal wordlist files.

       --per-conf=<file name>
              Personal configuration file.  This file overrides options found in the
              global config file.
```

There must be a way to add Soquel … the man page indicates it is possible but has no examples … lets try google instead

Googling "linux aspell personal dictionary"

Bingo! Thank you Samat Jain!

http://blog.samat.org/2008/11/02/creating-your-own-personal-aspell-dictionary


Google
Mug of Vi | No Starch Press
Creating your own personal
blog.samat.org/2008/11/02/creating-your-own-personal-aspell-dictionary
Safe Web
Identity Safe
SAMAT.ORG • BLOG • WIKI • MBLOG •
Search all
Search

Creating your own personal aspell dictionary

By SAMAT JAIN
November 2, 2008 - 4:23pm

Something that has bothered me forever is that applications that use GNU aspell for spell checking kept marking my name as a misspelling (I'm looking at you, KMail). Most front-end applications don't provide a way for you to add your own custom words.

Apparently, creating your own personal dictionary is ridiculous easy with aspell.

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

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

The first line is a required header. Every subsequent line is a word you want to add to your dictionary. I can't believe I've let this sit for so long. Because it's a nice text file, syncing this file between machines to take your di

Topic: Linu

Like this ar
via PayPal,

To stay on t
updates on

Lastly, othe

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

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

Add this line to the top

Now add any words you wish for the aspell program to ignore when doing spelling checks

Adding words to the UNIX dictionary

```
/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 Soquel to your own custom dictionary to be used with the spell command

This is FYI and not required for Lab 9

```
/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

Ayshire moshpit and personal dictionaries

moshpit?

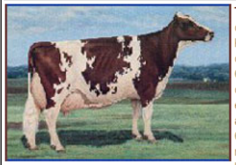
Ayshire?

urban DICTIONARY

1. **moshpit**

a place at a gig where you can dance with however the f**k bleeped you want with a bunch of people you don't know. the dancing will often include punches aimed in the air NOT at the person nearest to you however usually results in full contact. can be dangerous however everyone with a ticket should feel welcome in the mosh pit.

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]

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

Add more to your custom word list

```
cd
echo "moshpit" >> .aspell.en.pws
echo "Ayshire" >> .aspell.en.pws

spell edits/small_town
```

Note: Please 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/
```

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 >

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

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

Answer: The edits directory is not on the path but the local bin/ directory is

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

Remember the six steps of the shell

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

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
```

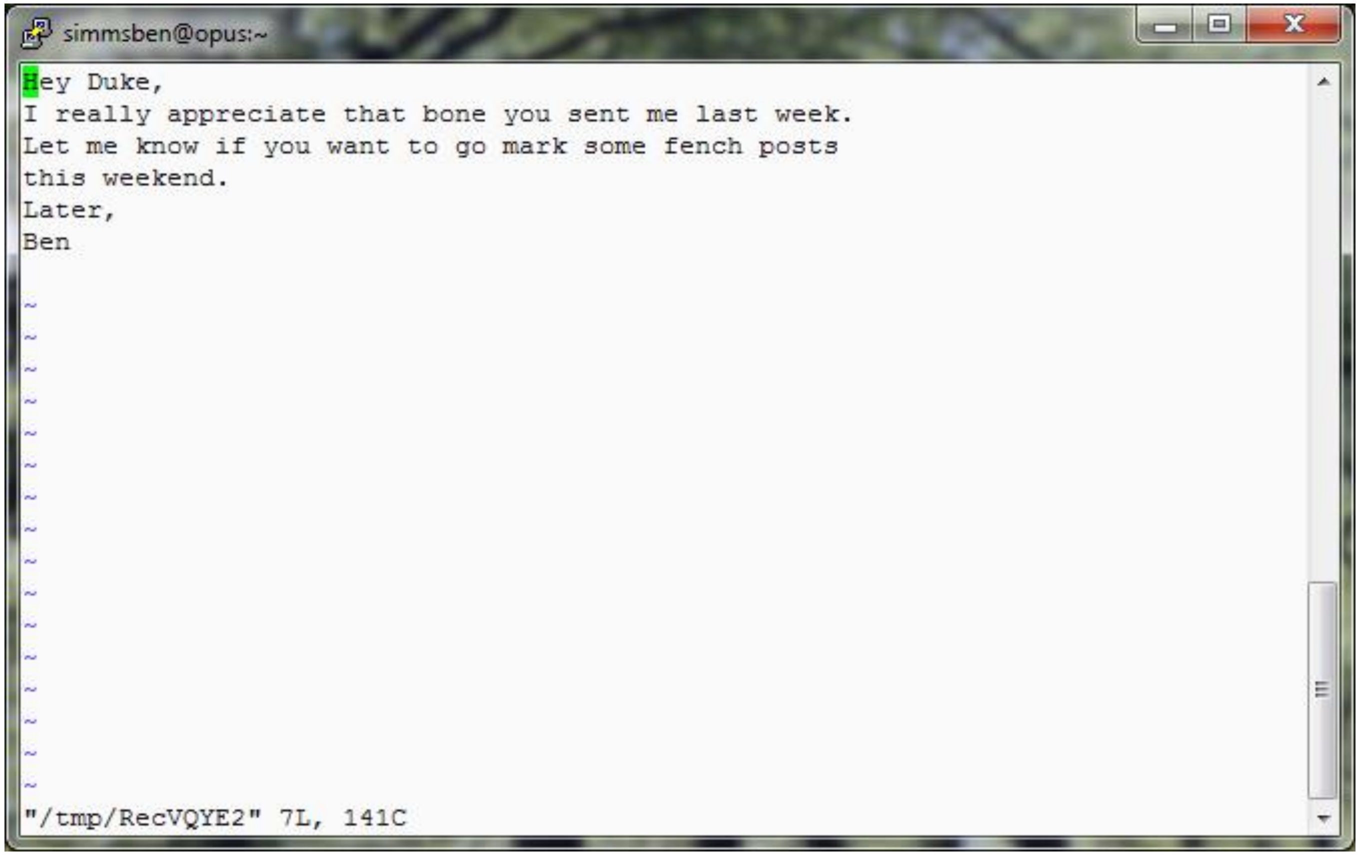
You are composing a message and you spot some typos …
CRUD … what can you do?

/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. <Esc>: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!

Activity

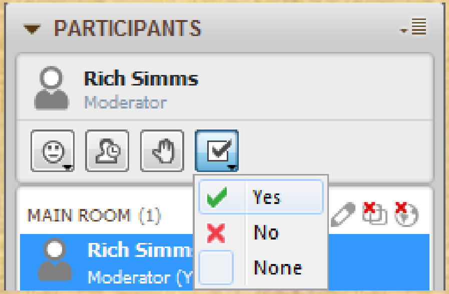
Try it!

Use /bin/mail and send yourself a message:

mail $LOGNAME

Type a few lines into the message then use the **~v** command to correct or change them.

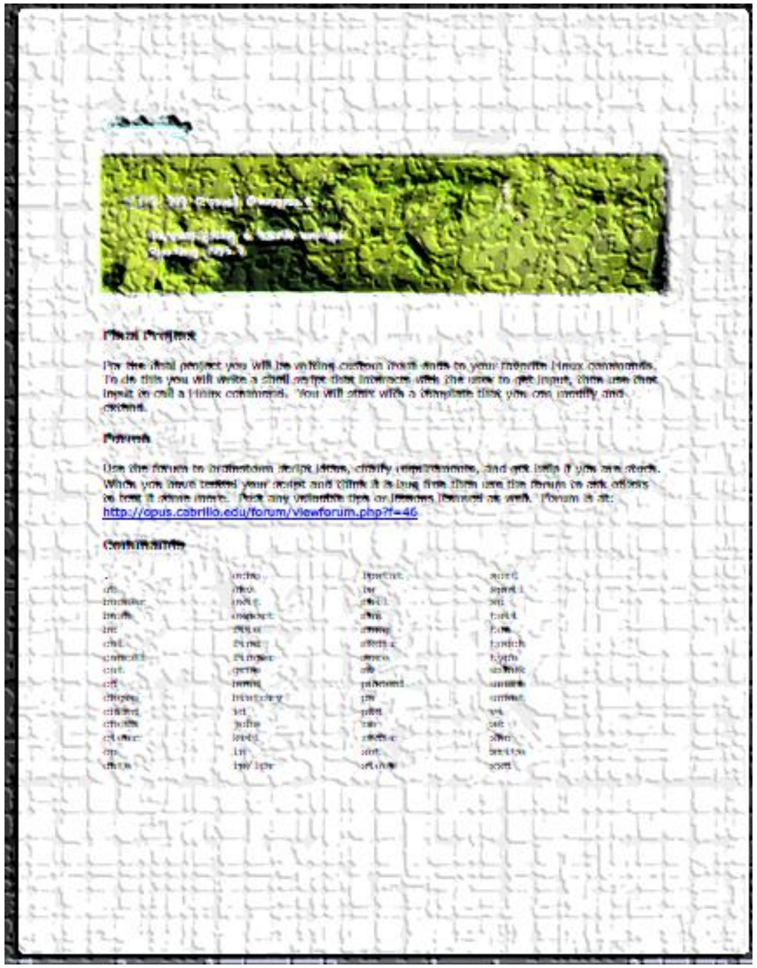
Read the email you sent yourself to see if your changes worked.



Did it work?
Start this activity by putting a red x in CCC Confer.
If you get it to work correctly change your red x to a green checkmark

final project preview

Final Project



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

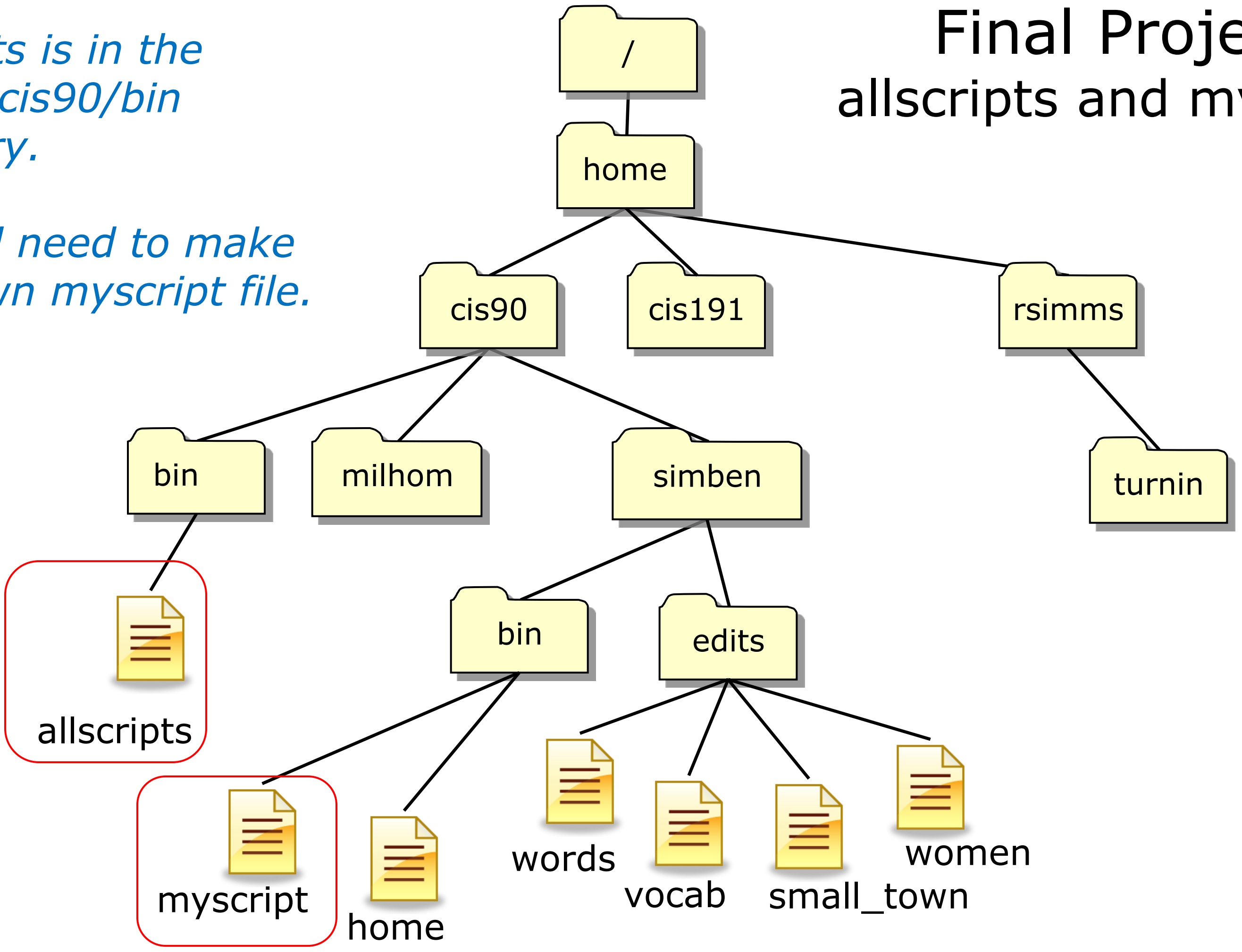
[illegible]

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

Final Project
allscripts and myscript

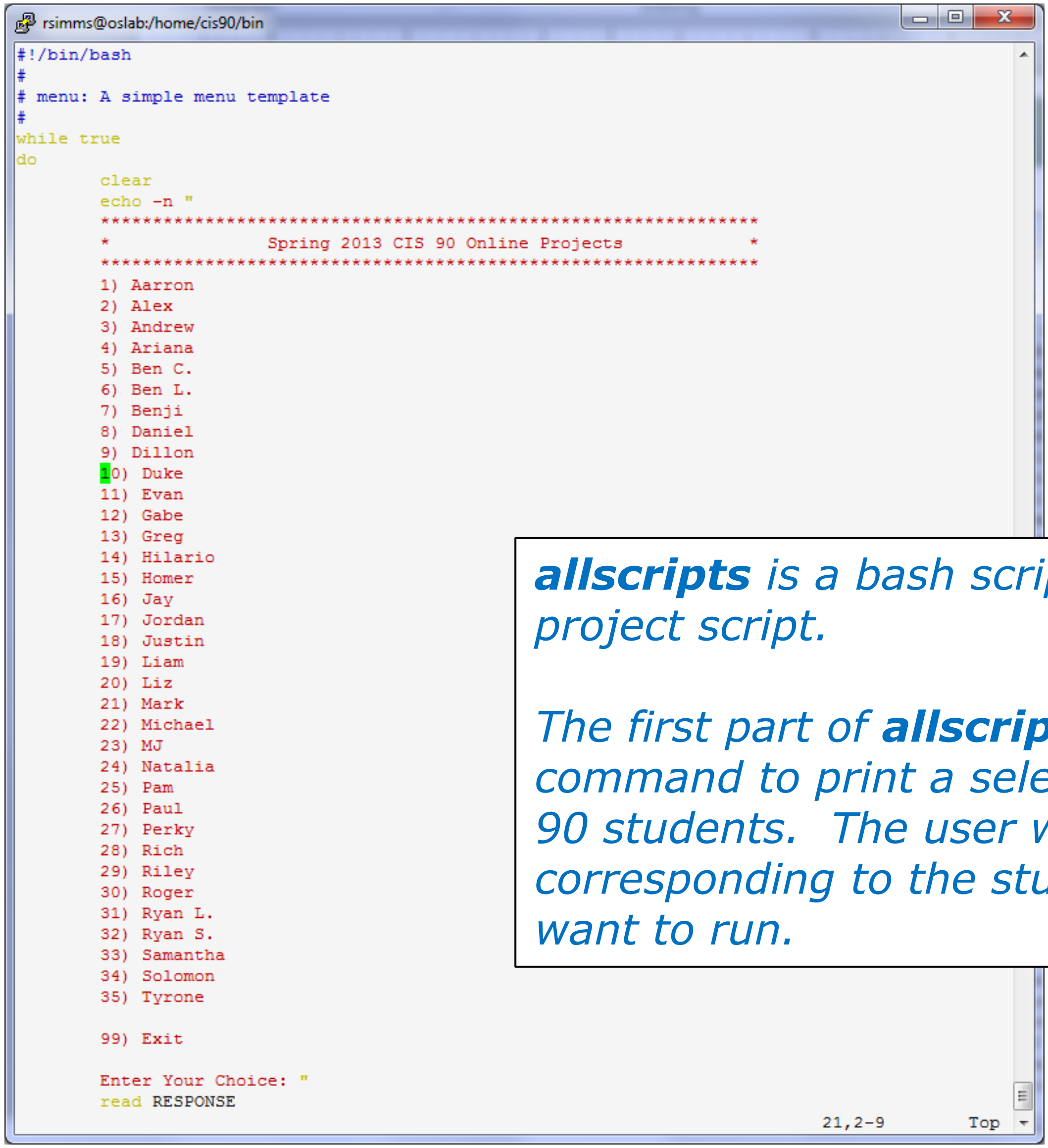
allscripts is in the /home/cis90/bin directory.

You will need to make your own myscript file.



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

CIS 90 - Lesson 12

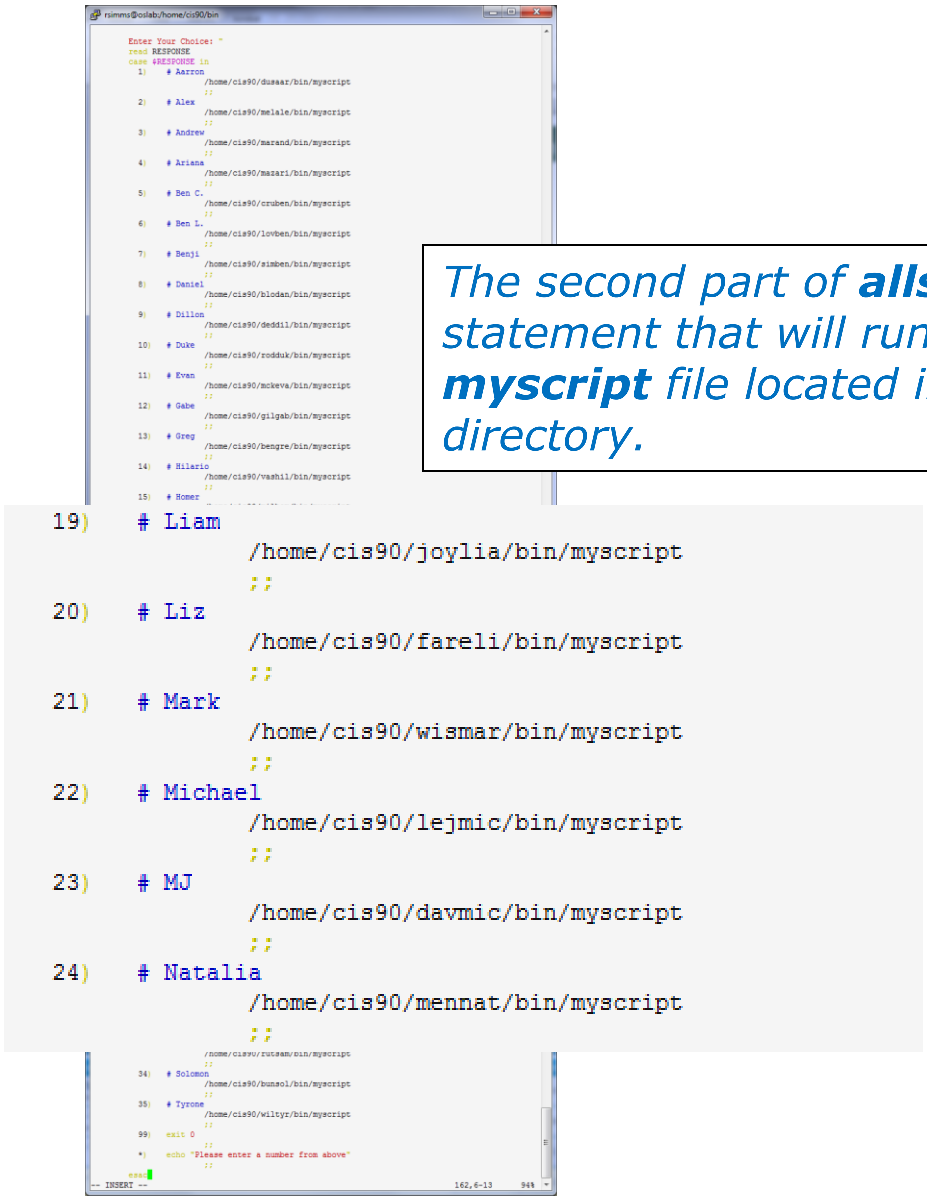


rsimms@oslab:/home/cis90/bin

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
        ****************************************************************
        *                Spring 2013 CIS 90 Online Projects            *
        ****************************************************************
        1) Aarron
        2) Alex
        3) Andrew
        4) Ariana
        5) Ben C.
        6) Ben L.
        7) Benji
        8) Daniel
        9) Dillon
        10) Duke
        11) Evan
        12) Gabe
        13) Greg
        14) Hilario
        15) Homer
        16) Jay
        17) Jordan
        18) Justin
        19) Liam
        20) Liz
        21) Mark
        22) Michael
        23) MJ
        24) Natalia
        25) Pam
        26) Paul
        27) Perky
        28) Rich
        29) Riley
        30) Roger
        31) Ryan L.
        32) Ryan S.
        33) Samantha
        34) Solomon
        35) Tyrone

        99) Exit

        Enter Your Choice: "
        read RESPONSE
                                                                  21,2-9        Top
```

allscripts *is a bash script that will run 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.*



rsimms@oslab:/home/cis90/bin

```
        Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Aarron
                        /home/cis90/dusaar/bin/myscript
                        ;;
          2)    # Alex
                        /home/cis90/melale/bin/myscript
                        ;;
          3)    # Andrew
                        /home/cis90/marand/bin/myscript
                        ;;
          4)    # Ariana
                        /home/cis90/mazari/bin/myscript
                        ;;
          5)    # Ben C.
                        /home/cis90/cruben/bin/myscript
                        ;;
          6)    # Ben L.
                        /home/cis90/lovben/bin/myscript
                        ;;
          7)    # Benji
                        /home/cis90/simben/bin/myscript
                        ;;
          8)    # Daniel
                        /home/cis90/blodan/bin/myscript
                        ;;
          9)    # Dillon
                        /home/cis90/deddil/bin/myscript
                        ;;
          10)   # Duke
                        /home/cis90/rodduk/bin/myscript
                        ;;
          11)   # Evan
                        /home/cis90/mckeva/bin/myscript
                        ;;
          12)   # Gabe
                        /home/cis90/gilgab/bin/myscript
                        ;;
          13)   # Greg
                        /home/cis90/bengre/bin/myscript
                        ;;
          14)   # Hilario
                        /home/cis90/vashil/bin/myscript
                        ;;
          15)   # Homer
```

```
   19)    # Liam
                  /home/cis90/joylia/bin/myscript
                  ;;
   20)    # Liz
                  /home/cis90/fareli/bin/myscript
                  ;;
   21)    # Mark
                  /home/cis90/wismar/bin/myscript
                  ;;
   22)    # Michael
                  /home/cis90/lejmic/bin/myscript
                  ;;
   23)    # MJ
                  /home/cis90/davmic/bin/myscript
                  ;;
   24)    # Natalia
                  /home/cis90/mennat/bin/myscript
                  ;;
```

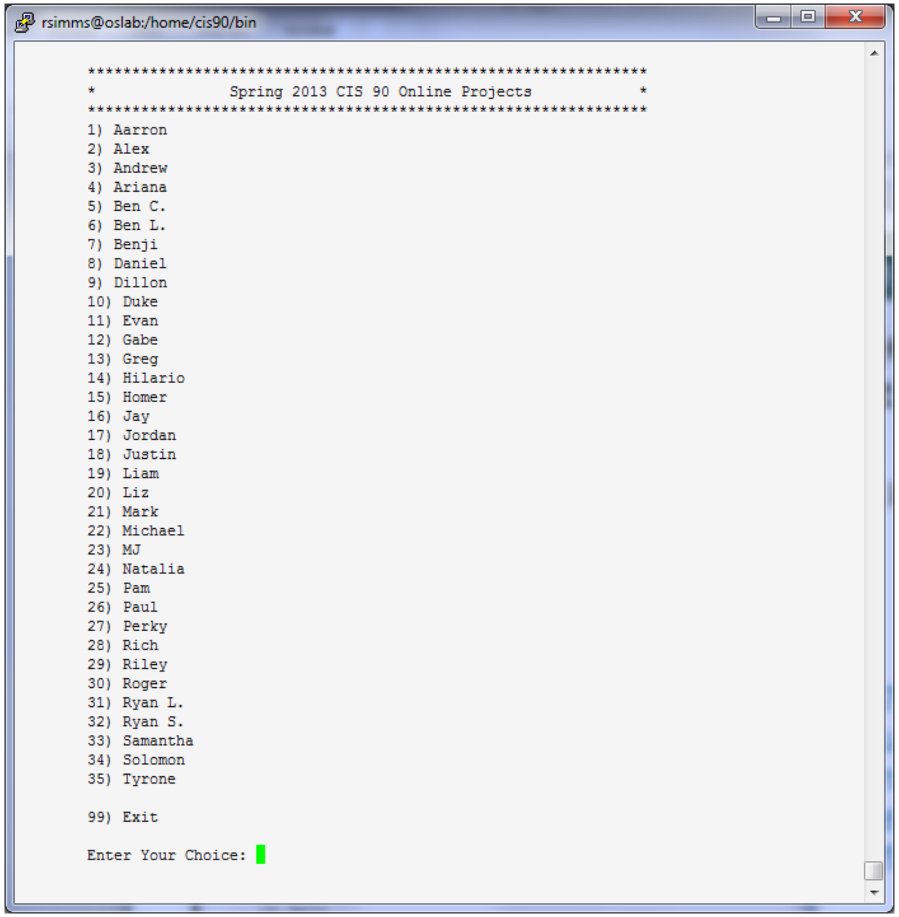
```
                        /home/cis90/rutsam/bin/myscript
                        ;;
          34)   # Solomon
                        /home/cis90/bunsol/bin/myscript
                        ;;
          35)   # Tyrone
                        /home/cis90/wiltyr/bin/myscript
                        ;;
          99)   exit 0
                        ;;
          *)    echo "Please enter a number from above"
                        ;;
        esac
-- INSERT --                                              162,6-13      94%
```

*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.*

Note the use of an absolute path to run each students script

Final Project
allscripts (continued)

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

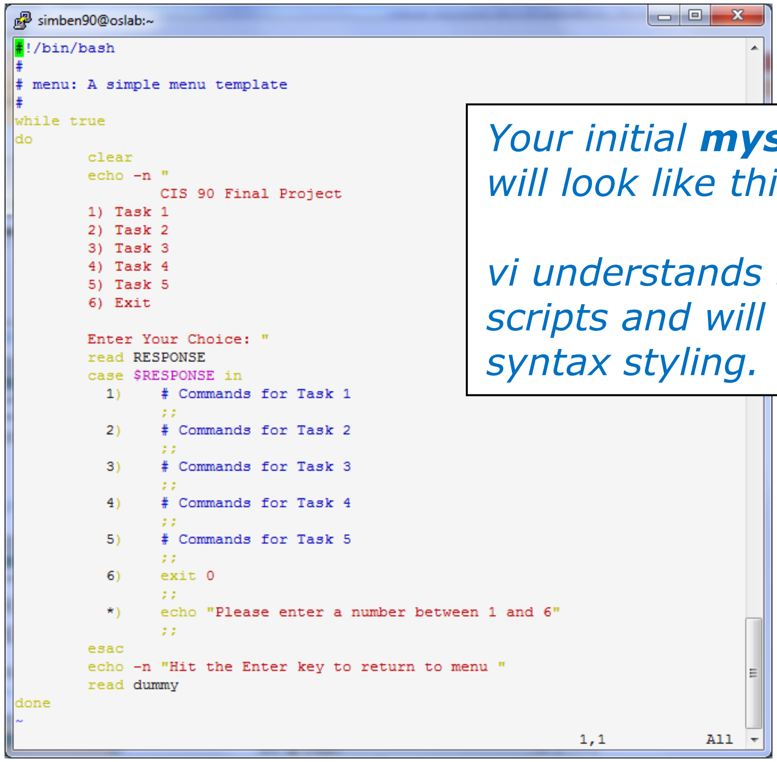

rsimms@oslab:/home/cis90/bin

```
*******************************************************************
*                Spring 2013 CIS 90 Online Projects              *
*******************************************************************
1) Aarron
2) Alex
3) Andrew
4) Ariana
5) Ben C.
6) Ben L.
7) Benji
8) Daniel
9) Dillon
10) Duke
11) Evan
12) Gabe
13) Greg
14) Hilario
15) Homer
16) Jay
17) Jordan
18) Justin
19) Liam
20) Liz
21) Mark
22) Michael
23) MJ
24) Natalia
25) Pam
26) Paul
27) Perky
28) Rich
29) Riley
30) Roger
31) Ryan L.
32) Ryan S.
33) Samantha
34) Solomon
35) Tyrone

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


simben90@oslab:~

```
#!/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
~
                                                            1,1           All
```

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

vi understands shell scripts and will use color syntax styling.

Every student will be creating a ***myscript*** *file in their bin directory for the final project.*

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

Getting Started

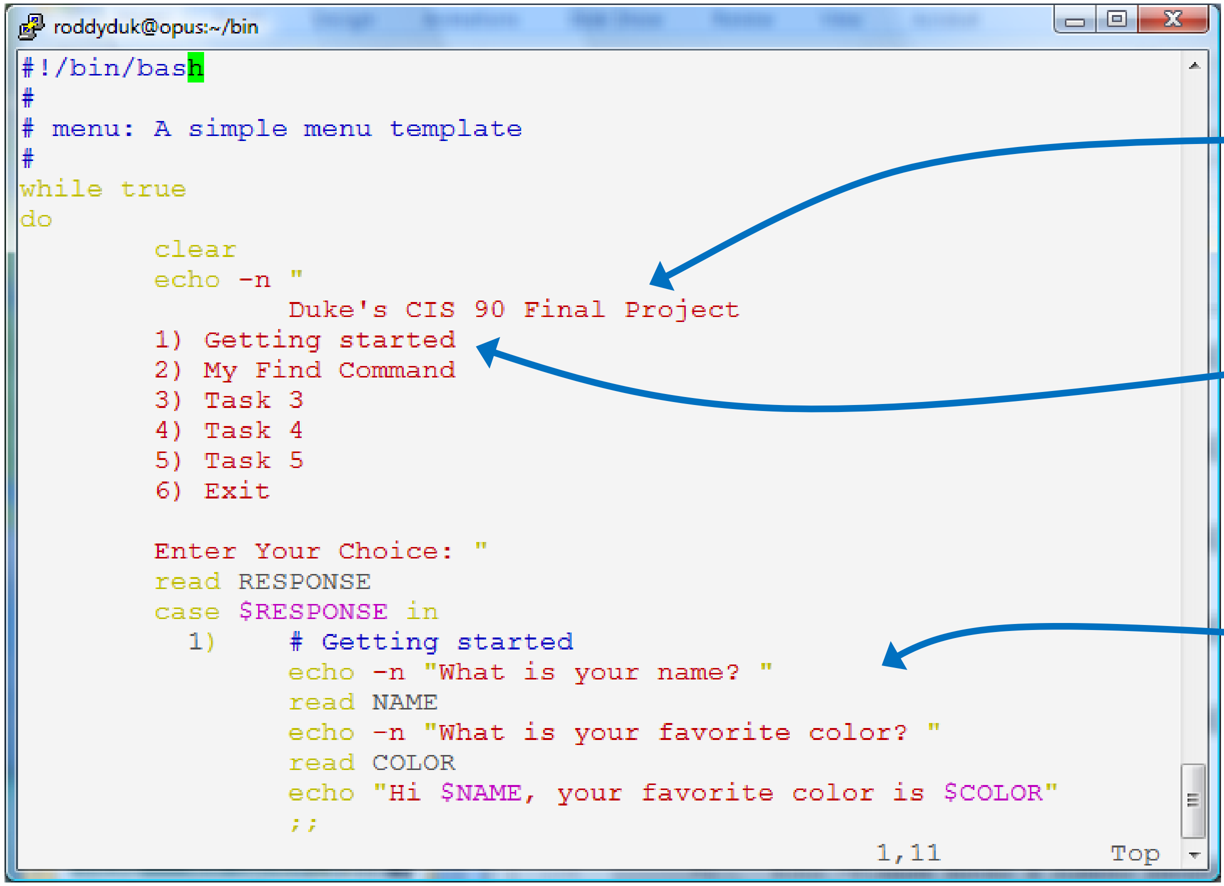
1) On Opus, cd to your home directory and enter:
```
cp  ../depot/myscript  bin/
```

2) Give your script execute permissions with:
```
chmod +x bin/myscript
```

3) Run the script:
```
myscript
```

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

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

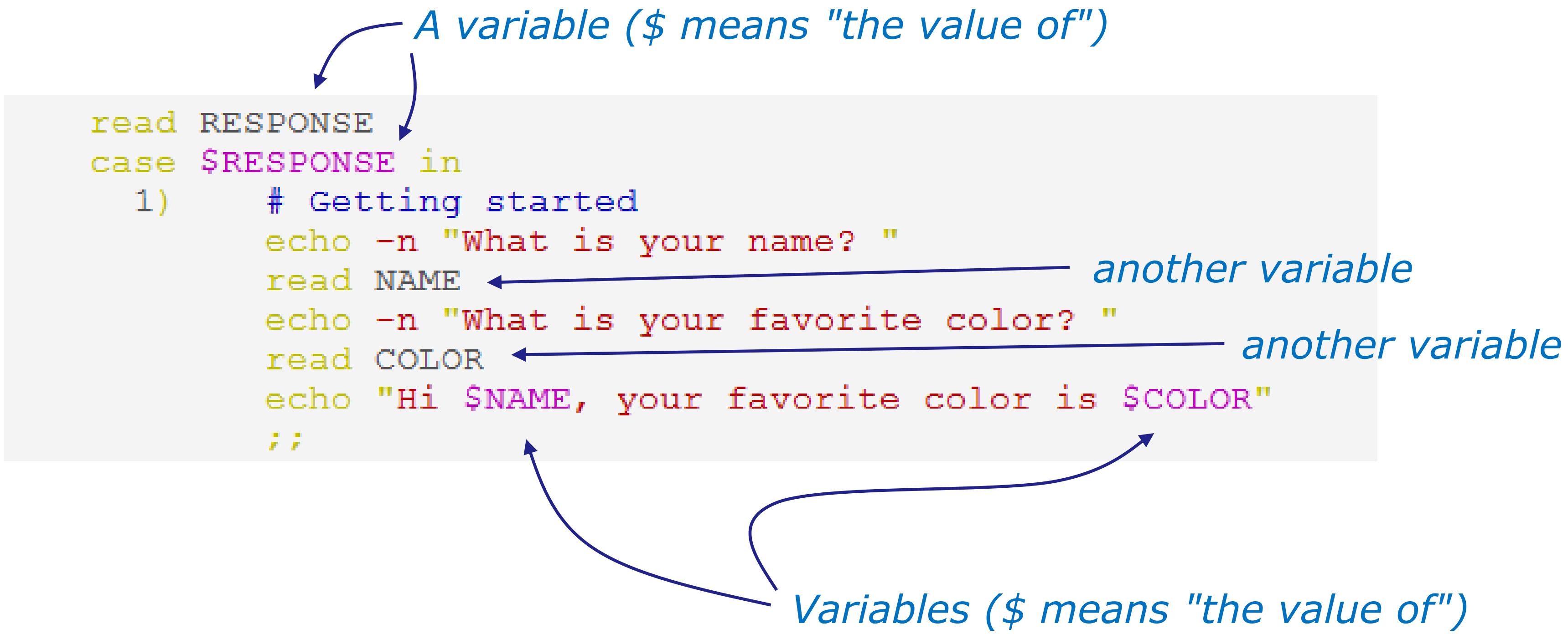
```
read RESPONSE
case $RESPONSE in
  1)     # Getting started
         echo -n "What is your name? "
         read NAME
         echo -n "What is your favorite color? "
         read COLOR
         echo "Hi $NAME, your favorite color is $COLOR"
         ;;
```

First case ends here

First case of case statement starts here

Final Project

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



```
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"
         ;;
```

A variable ($ means "the value of")
another variable
another variable
Variables ($ means "the value of")

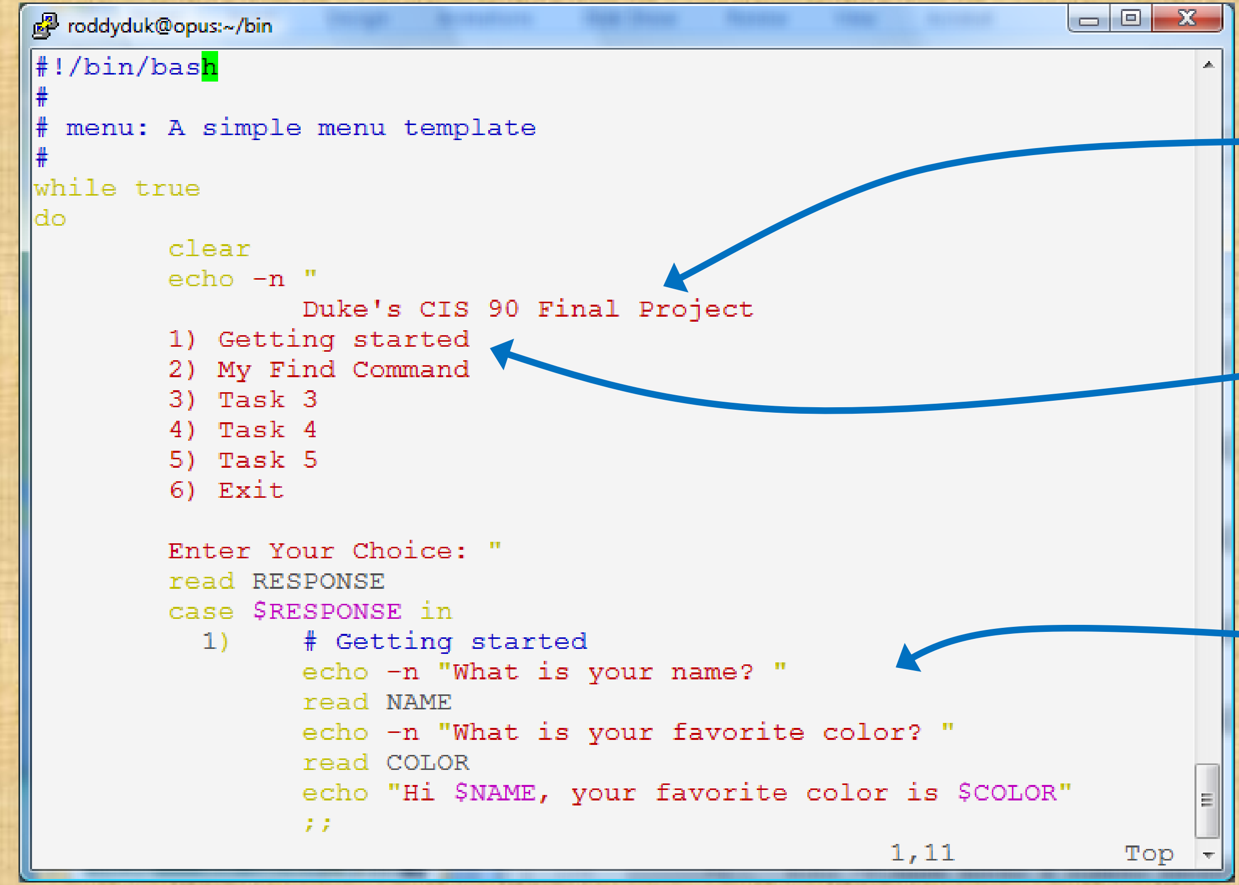
Final Project

/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 SHELL HOME SSH_TTY PWD EUID

BASH_VERSION LINES COLORS PPID IFS consoletype SHELLOPTS MAILCHECK HOSTNAME BASH_ENV

USER BASH PS4 TERM PIPESTATUS GROUPS HISTFILESIZE OPTIND BASH_VERSINFO UID

BASH_ARGV PATH PS1 SSH_CONNECTION tmpid SHLVL HISTFILE OSTYPE BASH_ARGC USERNAME

BASH_LINENO HISTSIZE LESSOPEN OPTERR SSH_CLIENT HOSTTYPE LS_COLORS CVS_RSH COLUMNS INPUTRC

BASH_SOURCE PROMPT_COMMAND _ MACHTYPE PS2 SSH_ASKPASS DIRSTACK MAIL 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

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

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

Environment 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

tmpid HISTFILE

SHLVL OSTYPE

BASH_ARGC **USERNAME**

BASH_LINENO **LESSOPEN**

HISTSIZE OPTERR

SSH_CLIENT

HOSTTYPE **LS_COLORS** **CVS_RSH**

COLUMNS **INPUTRC** BASH_SOURCE **_** MACHTYPE

PROMPT_COMMAND PS2

DIRSTACK **MAIL** **SSH_ASKPASS** **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

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

Shell (Environment) Variables

export command - show all exported variables

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

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

```
[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: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=""
```

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 29 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 30 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 environment (exported) shell variable

use grep to show fan is a 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)

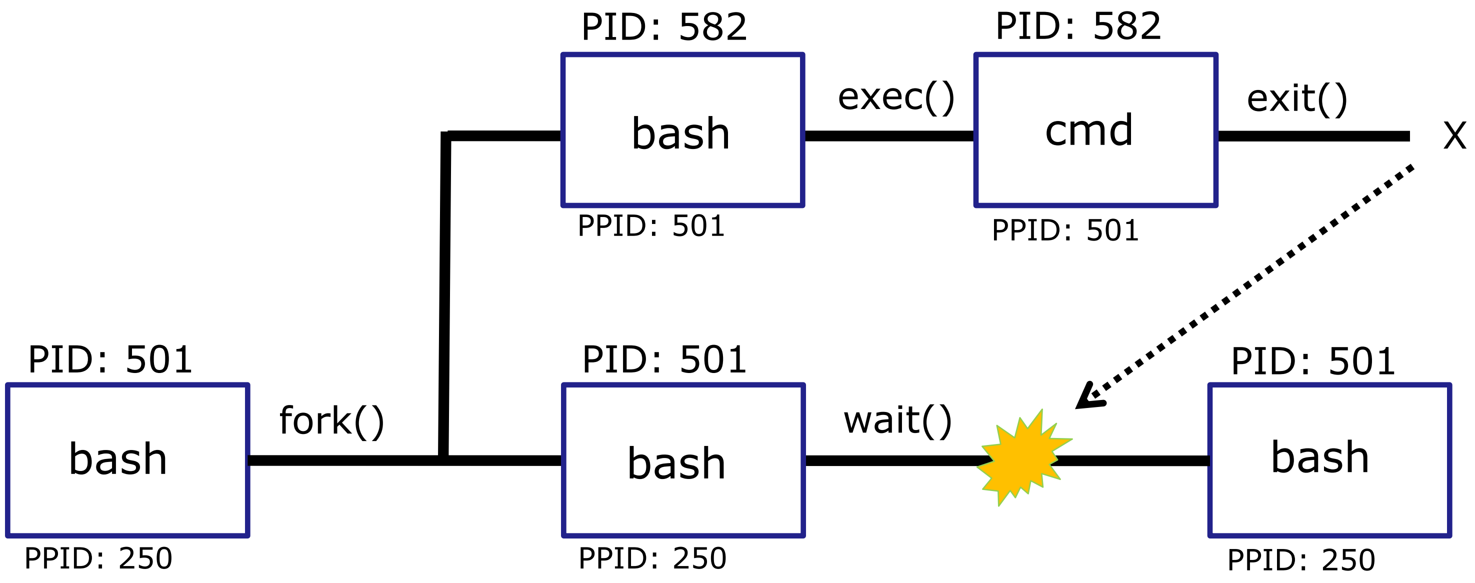
The prompt string can have any combination of text, variables and these codes.

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.*

LOGNAME is an environment variable that has been exported.

3 child

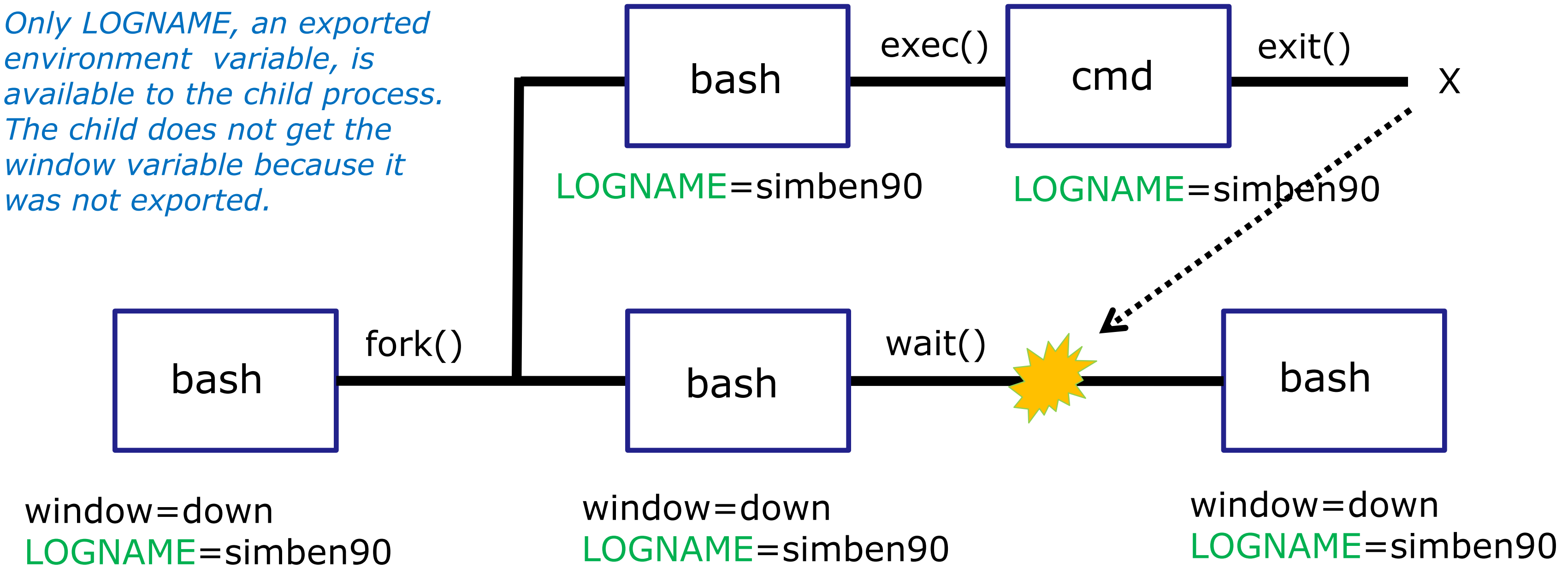
```
/home/cis90/simben $ bash
[simben@opus ~]$ echo $window $LOGNAME
simben90
[simben@opus ~]$ exit
exit
```

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

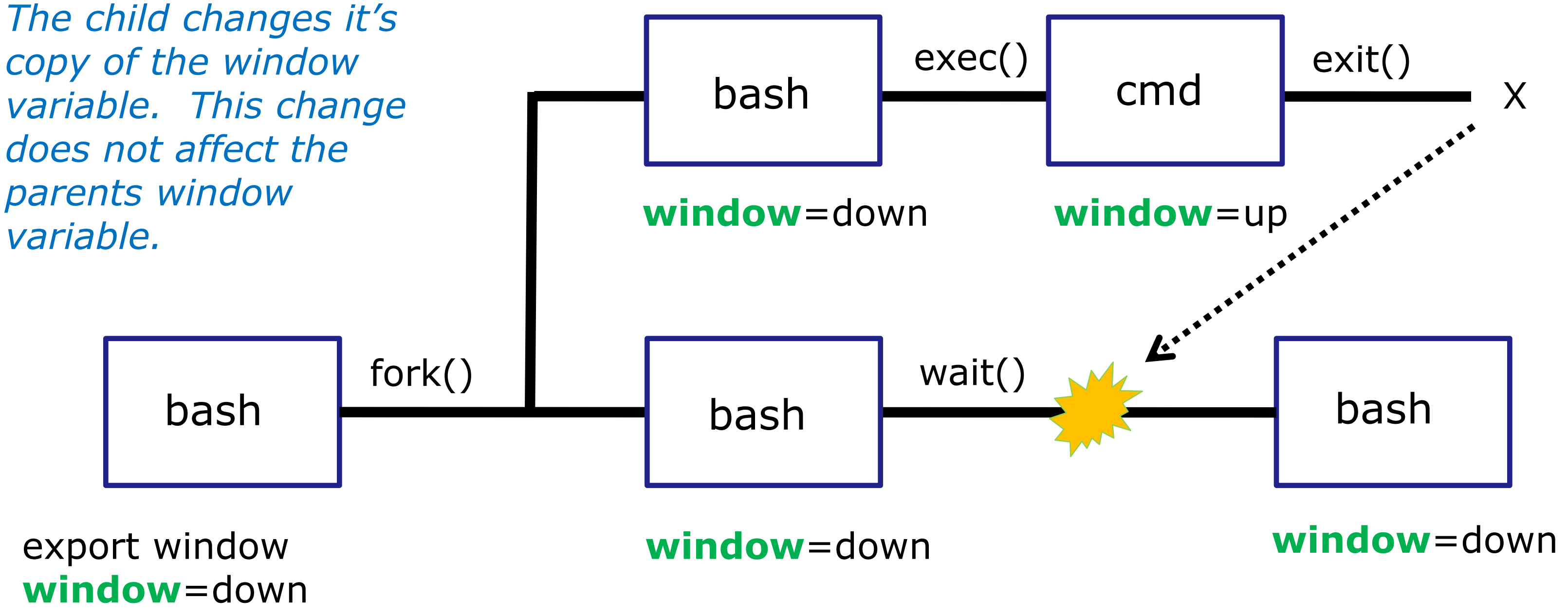
4 parent

```
/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/turnin/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 small_town

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 -n5 ~/edits/small_town'
/home/cis90/simben $ alias s
alias s='clear; head -n5 ~/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 mira=show
```

*Make **show** an alias of **cat***
*Make **mira** 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 $ mira 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 **mira letter** will cat out the letter file*

3

```
/home/cis90/simben $ unalias show
/home/cis90/simben $ alias mira
alias view='show'
/home/cis90/simben $ mira 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

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

2
```
$ p
on medium off
```

3
```
$ ac=off
```

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

single

1
```
$ alias p='echo $ac $fan $defrost'
$ alias p
alias p='echo $ac $fan $defrost'
```

2
```
$ p
on medium off
```

3
```
$ ac=off
```

4
```
$ p
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 probe=file**
- **probe /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 environment 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

The special prompt used for CIS 90 students is specified

variables are exported

umask value is set

EOF's are ignored

Suspend character redefined from Z to F

Terminal type is requested and 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***

$\left.\begin{array}{l} \\ \\ \end{array}\right\}$ *equivalent*

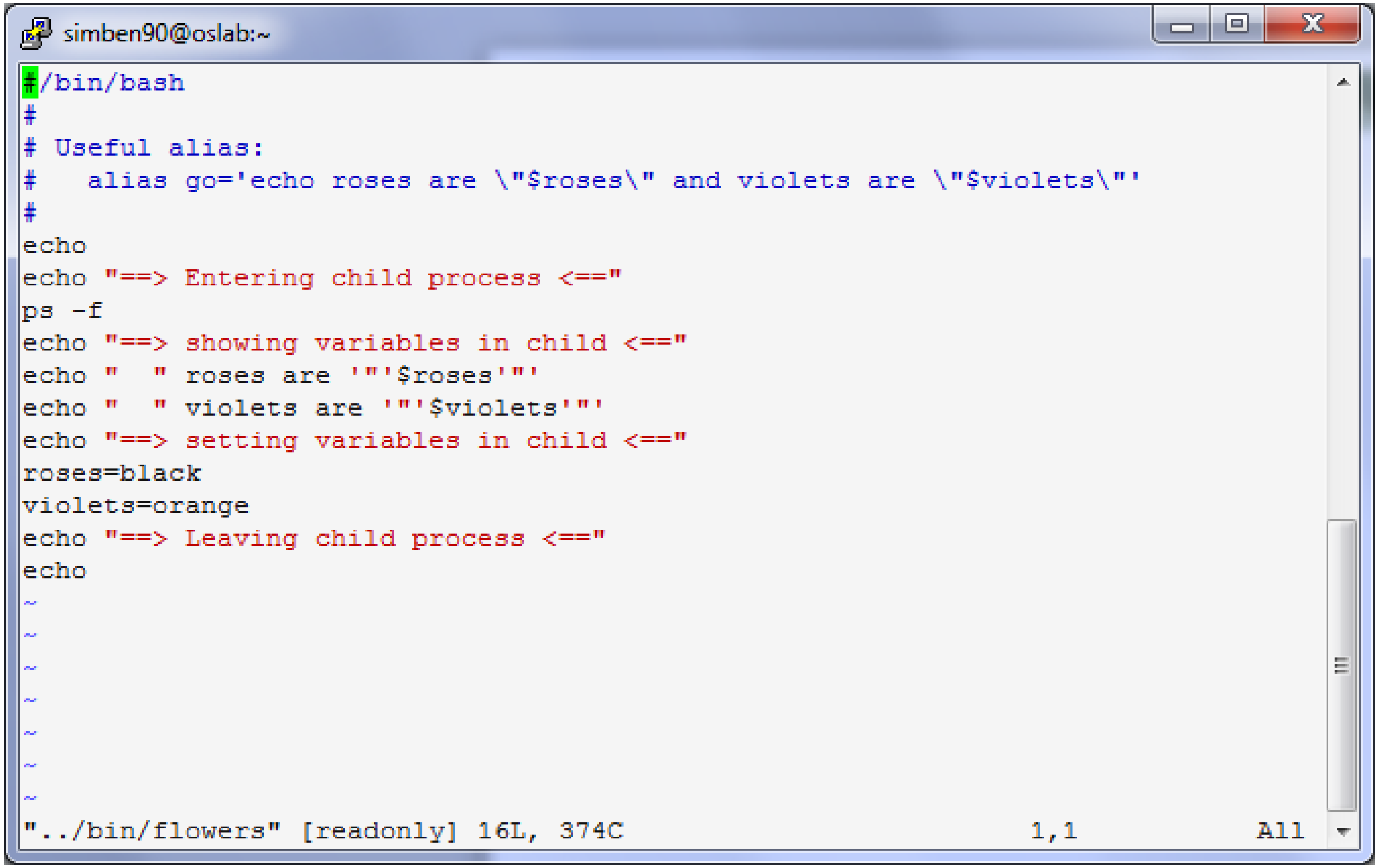
In this example, the commands in the file script are run by the parent shell, and therefore, any changes made to the environment will last for the duration of the login session.

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?

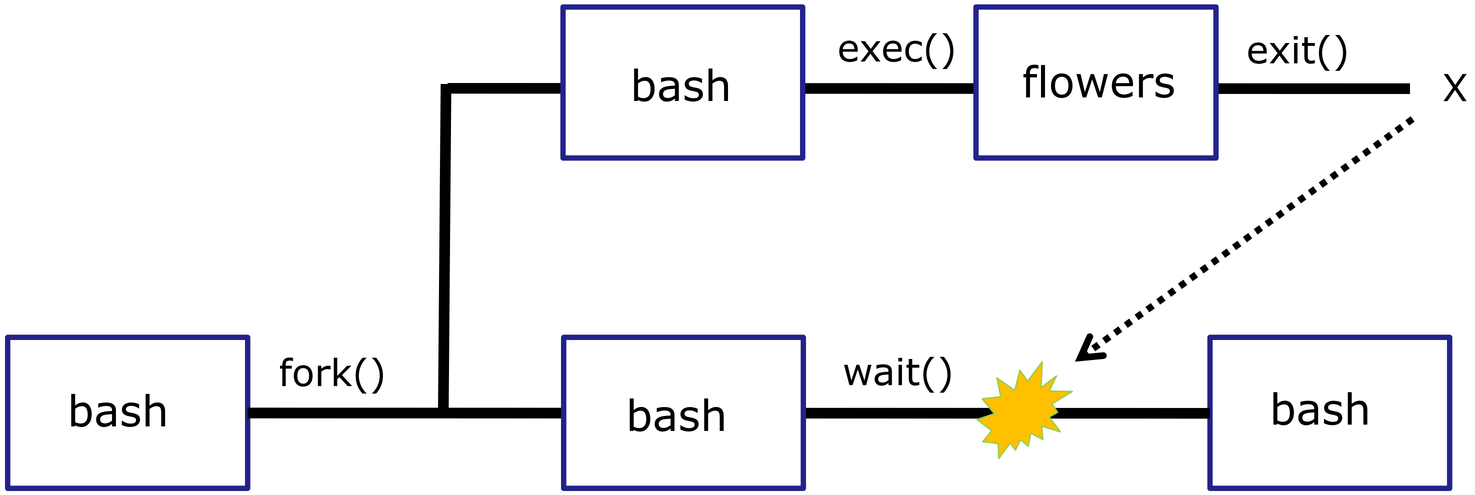
/home/cis90/bin/flowers



simben90@oslab:~

```
#/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"

/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
==> showing variables in child <==
   roses are ""
   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 does not see roses or violets

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

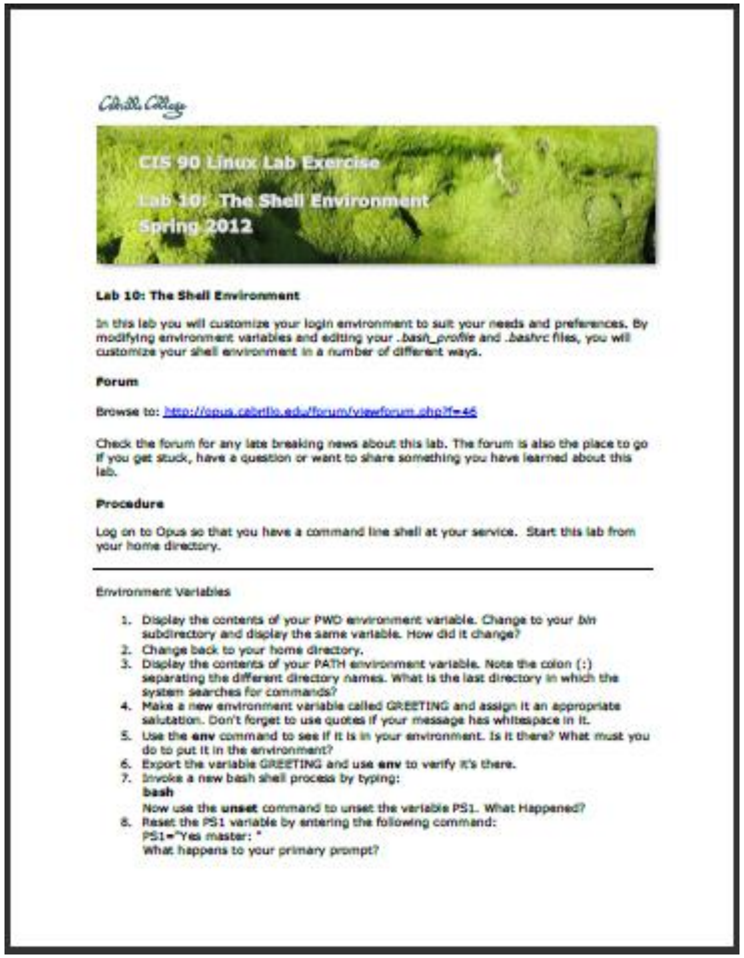
script is not running as child

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 2 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