Lesson Module Status
- Slides – draft
- Properties - done
- Flash cards –
- First minute quiz – done
- Web calendar summary – done
- Web book pages – done
- Commands – done
- Lab 10 and Final Project – done
- Supplies () - na
- Class PC's – na
- Chocolates - bringing

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

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

- Check that room headset is charged – done

CIS 90 - Lesson 12

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

Quiz #9

Please close your books, notes, lesson materials, forum and answer these questions **in the order** shown:

1. What vi command is used to exit vi without saving any of the changes you made?

2. What vi commands are used for copy and paste?

3. How do you send a SIGKILL to one of your own processes?

email answers to: risimms@cabrillo.edu

[ ] Has the phone bridge been added?

[ ] Is recording on?

[ ] Does the phone bridge have the mike?

[ ] Share slides, putty (rsimms, simmsben, roddyduk), and Chrome

[ ] Disable spelling on PowerPoint

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

Previous material and assignment

1. Lab 9 due midnight tonight

2. Five posts due midnight tonight

   Reminder: Only posts between in the CIS 90 ONLINE forum between 4/1 and 5/5 (inclusive) are counted.

3. Questions?
   - vi
   - lab 9
   - test 2
   - pathnames

Test Results

Test 2 Results

Test

01 X
02
03 XX
04 XX
05 X
06 XXXXXXXXX
07 XXXXXX
08
09 XX
10 XXXXXXX
11 XXXXXXX
12 XXXXX
13 XXXXXXXXXX
14 XXXXX
15 XXXXXX

Extra Credit

16 XXXXXX
17 XXXXXXXXX
18 XXXXX
19 XXXX
20 XXXXXXXXXXXXXXX
21 XXX
22 XXXXXXX
23 XXXXXX
24 XXXXXX
25 XX

Test 2 Q6

What single command would list the accounts in /etc/passwd belonging to students in the CIS 90 Online group (cis90ol) and use the bash shell?

cat /etc/passwd

Will list all accounts

cat /etc/passwd | grep cis90ol

List all accounts, from those list just the cis90ol ones

Answer:

cat /etc/passwd | grep cis90ol | grep bash

List all accounts, from those list just the cis90ol ones, from those list just the bash users

Tip, build these pipeline commands one step at a time to see if they are doing what you want.

Test 2 Q7

If the contents of the file named Lost is:

Hugo
Sun
Ben
Sawyer
Jin
Jack
Kate

Make your own Lost file

```
/home/cis90ol/simmsben $ echo Hugo > Lost
/home/cis90ol/simmsben $ echo Sun >> Lost
/home/cis90ol/simmsben $ echo Ben >> Lost
/home/cis90ol/simmsben $ echo Sawyer >> Lost
/home/cis90ol/simmsben $ echo Jin >> Lost
/home/cis90ol/simmsben $ echo Jack >> Lost
/home/cis90ol/simmsben $ echo Kate >> Lost
```

What is the result of doing the following command?

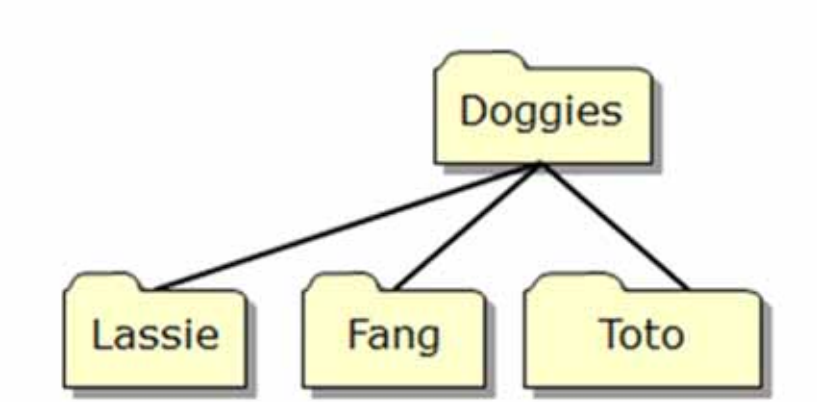
sort -r Lost | tail -3 | head -1 > Jack

```
/home/cis90ol/simmsben $ sort -r Lost | tail -3 | head -1 > Jack
/home/cis90ol/simmsben $ cat Jack
Jack
/home/cis90ol/simmsben $
```

Answer:
A file named Jack is created that contains "Jack"

Test 2 Q10

What single command (no ";"s) creates a new directory named Doggies containing three new sub-directories, named Lassie, Fang and Toto? **cat**



Answer:

mkdir -p Doggies/Lassie Doggies/Fang Doggies/Toto

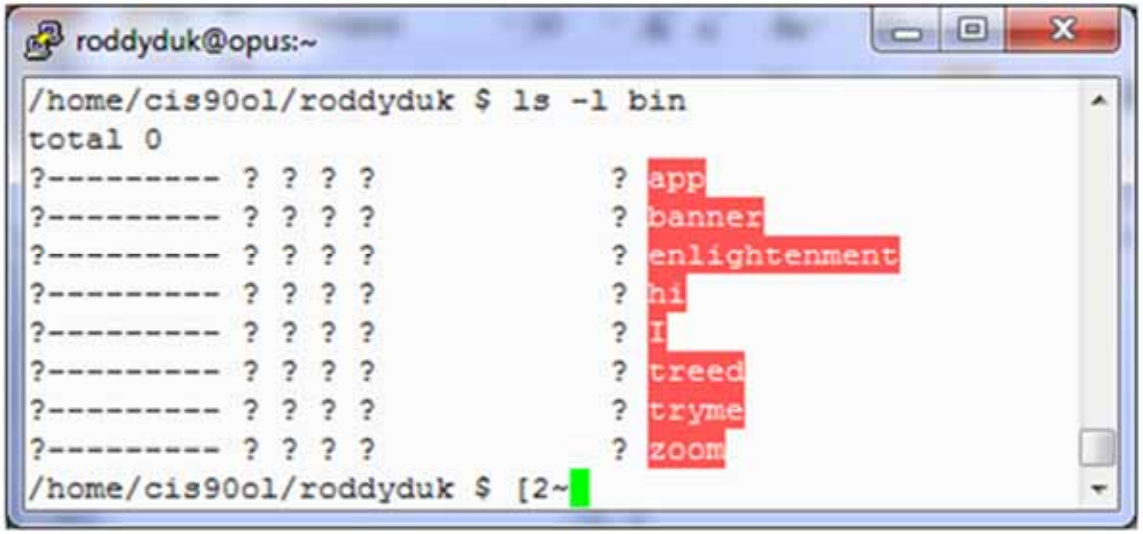
or

mkdir -p Doggies/{Lassie,Fang,Toto}

Use the p (parent) option to create parent directories as needed

Test 2 Q11

11. Duke messed up his *bin* directory. His long listings now look like the following:



Note: No inode information is being displayed. It requires execute permission to access inode information.

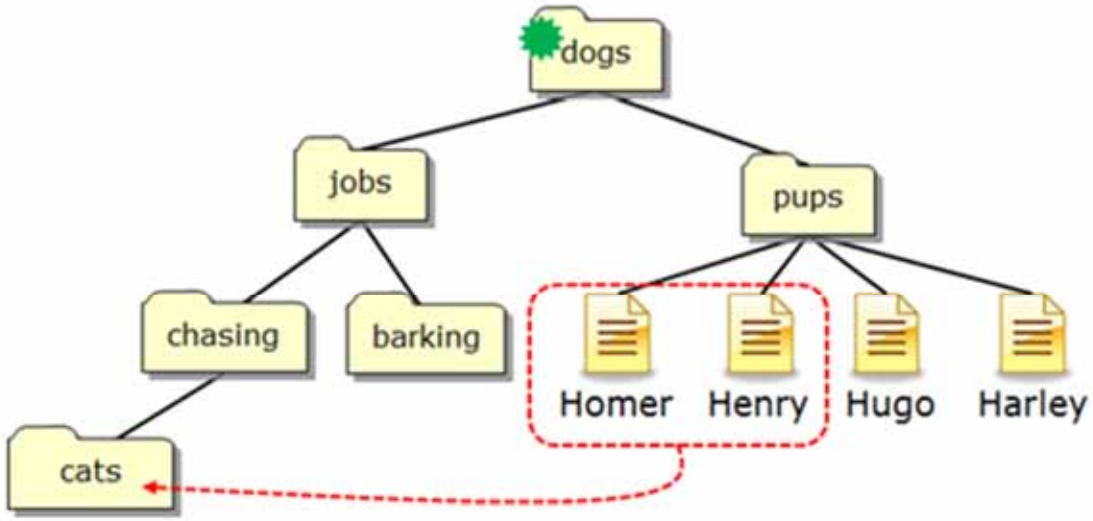
What command should Duke issue so he can see all his file information again?

Answer: chmod u+x bin

Add execute permission to the user (owner) to allow user to access inode information

Test 2 Q12

12. Given this directory structure:



If your current working directory is *dogs*, what single command using filename expansion characters would move just the files *Homer* and *Henry* to the *cats* directory?

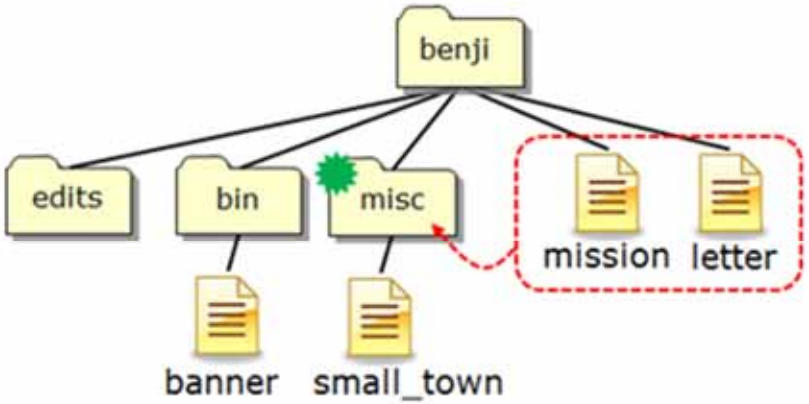
Answer: mv pups/H[oe]* jobs/chasing/cats/

When expanded this becomes two pathnames, one to Homer and one to Henry

relative pathname to cats directory

Test 2 Q13

13. Given this directory structure:



If your current working directory is *misc*, what single command (no ";"s) would copy the *mission* and *letter* files to the *misc* directory?

Answer: cp ../mission ../letter .

relative pathname to mission

relative pathname to letter

"Here" (the . directory is hard linked to the current directory)

Test 2 Q14 (part 1)

14. There is a ASCII English text file in */home/cis90ol/depot* that is 33,939 bytes in size. From your home directory, what single command line **using at least one filename expansion character** and an **environment variable**, could be used to mail yourself and rsimms, a count of the misspelled words in that file, with a subject of "Test 2 Q14"?

```
/home/cis90ol/simmsben $ ls -l ../depot
total 284
-rwxr-xr-x 1 rsimms cis90ol    682 Nov  9 15:22 app
-rw-r--r-- 1 rsimms cis90ol  13674 Jun  2  2009 bho.txt
-rwxr-xr-x 1 rsimms cis90ol    316 Nov 12  2008 dialog
-rw-r--r-- 1 rsimms cis90ol  33939 Nov 10  2008 dickens
-rw-r--r-- 1 rsimms cis90ol 112640 Nov 21 19:51 dogs.tar
drwxr-xr-x 3 rsimms cis90ol   4096 Mar  4  2009 filetypes
drwxr-xr-x 8 rsimms cis90ol   4096 Feb 28 21:10 gillay
-rw-r--r-- 1 rsimms cis90ol     86 Jun  2  2009 hk.txt
-rw-r--r-- 1 rsimms cis90ol    895 Jun  2  2009 index.html
-rw-r--r-- 1 rsimms cis90ol  11096 Jun  2  2009 jfk.txt
-rw-r--r-- 1 rsimms cis90ol   1046 Nov 10  2008 letter
-rw-r--r-- 1 rsimms cis90ol    759 Nov 10  2008 mission
-rw-r--r-- 1 rsimms cis90ol    481 Nov 21 09:33 myscript
-rw-r--r-- 1 rsimms cis90ol      0 Dec  7 13:09 myscripts
lrwxrwxrwx 1 rsimms staff        7 Sep 21  2010 scrooge -> dickens
-rw-r--r-- 1 rsimms staff      611 Mar 17 20:48 sonnet6
/home/cis90ol/simmsben $
```

Note: The dickens file is 33,939 bytes in length

Test 2 Q14 (part 2)

14. There is a ASCII English text file in */home/cis90ol/depot* that is 33,939 bytes in size. From your home directory, what single command line **using at least one filename expansion character** and an **environment variable**, could be used to mail yourself and rsimms, a count of the misspelled words in that file, with a subject of "Test 2 Q14"?

filename expansion characters

environment variable

Answer:

```
spell ../dep*/di* | wc -l | mail -s "Test 2 Q14" rsimms $LOGNAME
```

output misspelled words in dickens file

count them

mail them

Tip, build these pipeline commands one step at a time to see if they are doing what you want.

Test 2 Q15

15. Given the file *expressions* contains these two lines:

```
5+5
9/0
```

What complete command using **bc** would input the math problems in *expressions*, **append** the calculated answers to the file *results* and write any errors to the file *errors*?

stdin *redirected from keyboard to file expressions*

stderr *redirected from terminal to file errors*

bc < expressions >> results 2> errors

stdout *redirected from terminal to append to file results*

Test 2 Q16

16. You have been hired as a parser. You will need to parse command lines and identify the command, options, arguments and any redirection. This includes doing any filename expansion. Parse the command below and complete the section that follows:

file -i /dev/[sh]d* > q16-report 2> q16-errors

(A16)

command: file

option(s): -i

argument(s): /dev/hda
/dev/sda
/dev/sda1
/dev/sda2

redirection: stdout redirected to q16-report
stderr redirected to q16-errors

```
/home/cis90ol/simmsben $ echo /dev/[sh]d*
/dev/hda /dev/sda /dev/sda1 /dev/sda2
```

Note: when filling out the redirection portion above, you must specify the actual file descriptors being redirected.

Test 2 Q17

17. On Opus, how many directories and sub-directories are in the /usr/src branch of the UNIX file tree?

```
/home/cis90ol/simmsben $ find /usr/src -type d | wc -l
1780
```

Answer: 1780 (plus or minus 1)

FYI, if you are in CIS 130 and want to use a regular expression:

```
/home/cis90ol/simmsben $ ls -lR /usr/src | grep "^d" | wc -l
1779
```

Test 2 Q20

20. What complete command (with no ";"s) **counts** all the files on Opus belonging to the cis90 user, places a sorted list of them in the file *cis90files*, and redirects error messages to the bit bucket?

```
/home/cis90ol/simmsben $ find / -user cis90 2> /dev/null | sort | tee cis90files | wc -l
115
/home/cis90ol/simmsben $
```

Answer:

Limits the files listed to just those owned by cis90 user

The tee sends the sorted files to both the file cis90files and to stdout

find / -user cis90 2> /dev/null | sort | tee cis90files | wc -l

This lists files starting at / on the UNIX file tree

Permission errors are thrown away by being redirected into the "bit bucket"

Test 2 Q22

22. What complete one-liner command (with ";"s) creates a file named *charlotte*, changes that file's group to be *users*, then removes all permissions for others on the file?

Answer:
touch charlotte; chgrp users charlotte; chmod o-rwx charlotte

```
/home/cis90ol/simmsben $ touch charlotte; chgrp users charlotte; chmod o-rwx
charlotte
/home/cis90ol/simmsben $ ls -l charlotte
-rw-rw---- 1 simmsben users 0 Apr 25 17:07 charlotte
```

Test 2 Q23

23. Using only **echo** commands and file redirection, how could you create a file named *characters* so it would contain the following lines?

Hugo
Sun
Ben
Sawyer
Jin
Jack
Kate

Answer:

```
echo Hugo > characters
echo Sun >> characters
echo Ben >> characters
echo Sawyer >> characters
echo Jin >> characters
echo Jack >> characters
echo Kate >> characters
```

Verify by printing file

```
/home/cis90ol/simmsben $ cat characters
Hugo
Sun
Ben
Sawyer
Jin
Jack
Kate
```

Test 2 Q23

23. Using only **echo** commands and file redirection, how could you create a file named *characters* so it would contain the following lines?

Hugo
Sun
Ben
Sawyer
Jin
Jack
Kate

Alternate answer:

```
echo "Hugo
> Sun
> Ben
> Sawyer
> Jin
> Jack
> Kate" > characters
```

Verify by printing file

```
/home/cis90ol/simmsben $ cat characters
Hugo
Sun
Ben
Sawyer
Jin
Jack
Kate
```

Test 2 Q23

23. Using only **echo** commands and file redirection, how could you create a file named *characters* so it would contain the following lines?

Hugo
Sun
Ben
Sawyer
Jin
Jack
Kate

Alternate answer:

```
echo -e "Hugo\nSun\nBen\nSawyer\nJin\nJack\nKate" > characters
```

Verify by printing file

```
/home/cis90ol/simmsben $ cat characters
Hugo
Sun
Ben
Sawyer
Jin
Jack
Kate
```

Test 2 Q24

24. What command, from your home directory, would print only line 504 of the file *dickens* in the */home/cis90ol/depot* directory? Hint: Use both **head** and **tail**.

Answer:
head -504 ../depot/dickens | tail -1

Print the first 504 lines of Dickens, then out of that, print the last line

Try it on Opus

```
/home/cis90ol/simmsben $ head -504 ../depot/dickens  | tail -1
"My time grows short," observed the Spirit. "Quick!"
/home/cis90ol/simmsben $
```

Housekeeping

Ayshire m[ao]shpit and personal dictionaries

aspell command

Googling "linux aspell personal dictionary" yields this page

Bingo! Thank you Samat Jain

spell command

```
/home/cis90ol/simmsben $ echo Benji lives in Soquel > address
/home/cis90ol/simmsben $ cat address
Benji lives in Soquel
/home/cis90ol/simmsben $ spell address
Soquel

/home/cis90ol/simmsben $ echo "personal_ws-1.1 en 0" > .aspell.en.pws
/home/cis90ol/simmsben $ echo Soquel >> .aspell.en.pws
/home/cis90ol/simmsben $ spell address
/home/cis90ol/simmsben $
```

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

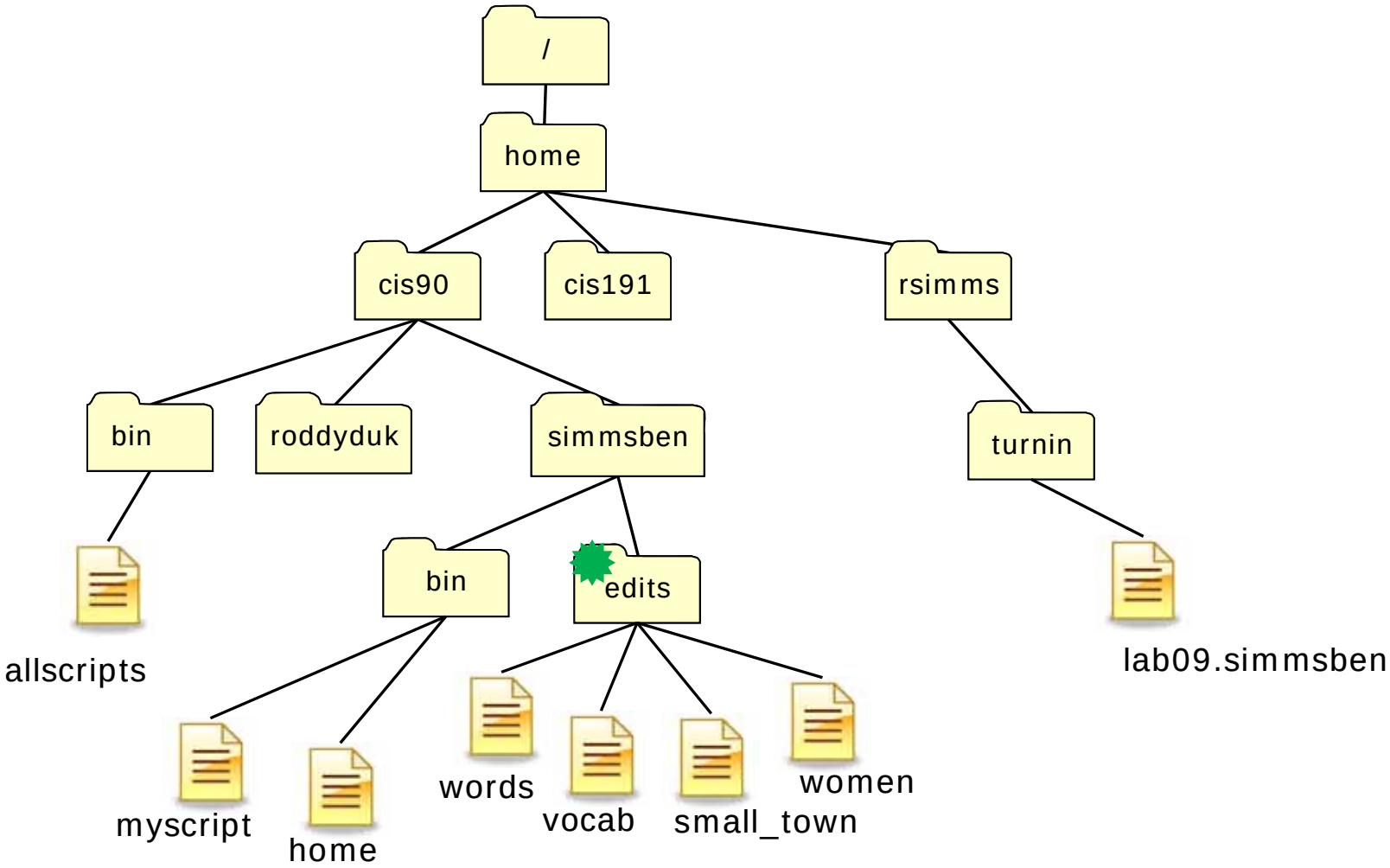
```
cd
echo "personal_ws-1.1 en 0" >  .aspell.en.pws
echo "moshpit" >> .aspell.en.pws
echo "mashpit" >> .aspell.en.pws
echo "Ayshire" >> .aspell.en.pws
cat .aspell.en.pws

cd edits/
spell small_town
```

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

pathnames

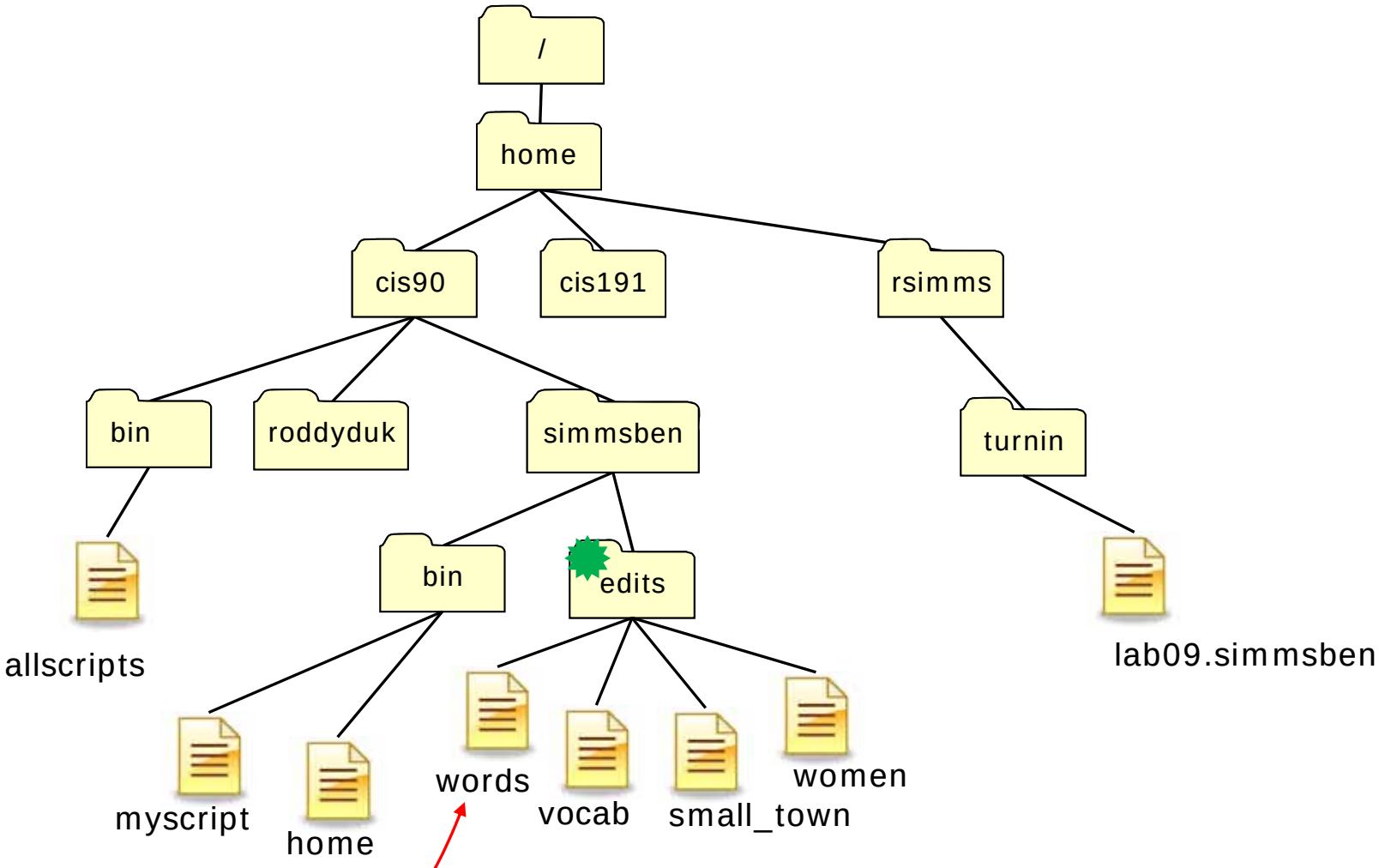
Reminder

- You must always use complete pathnames when specifying files as arguments on a command.
- Pathnames can be relative or absolute.



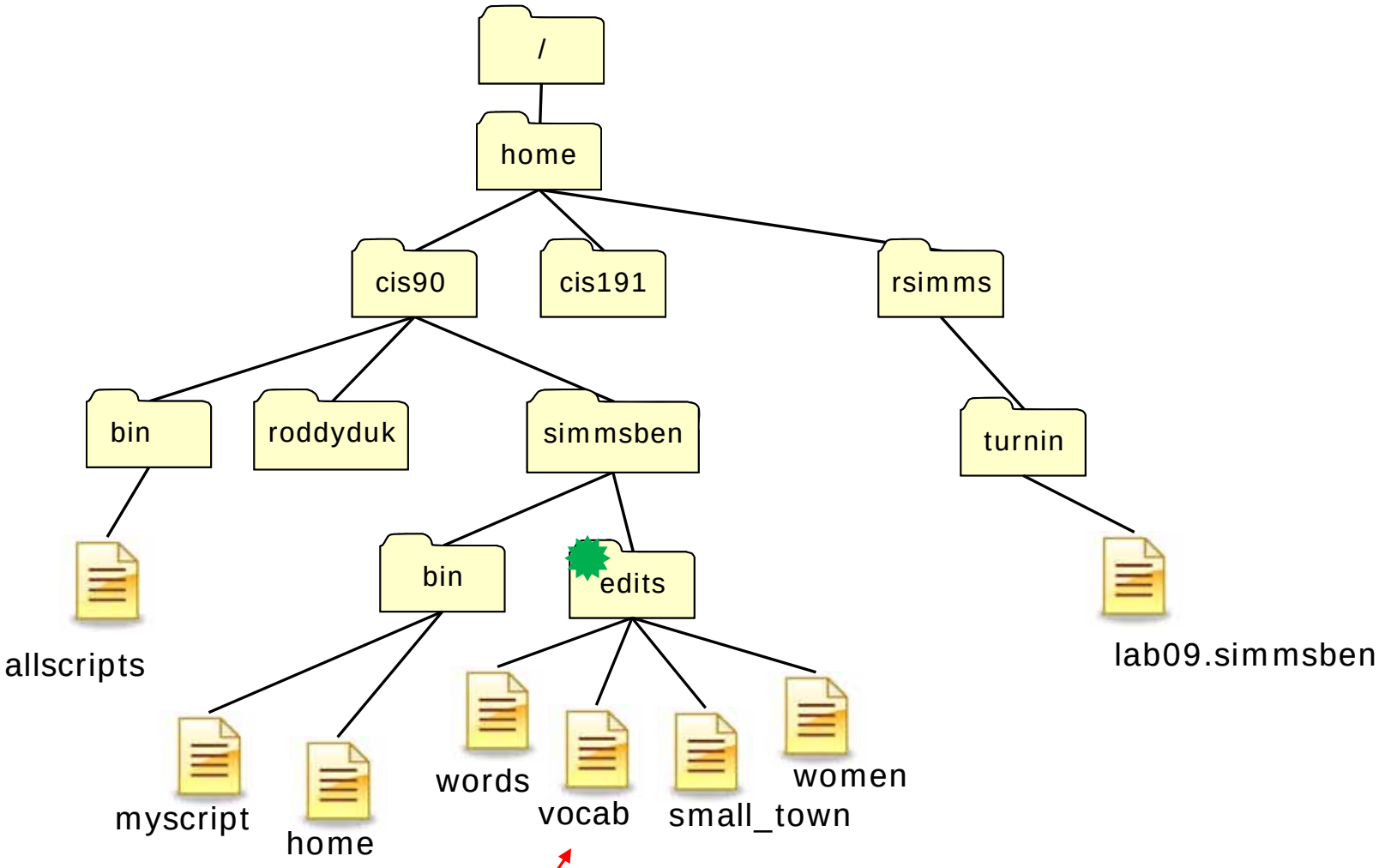
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME

relative pathname



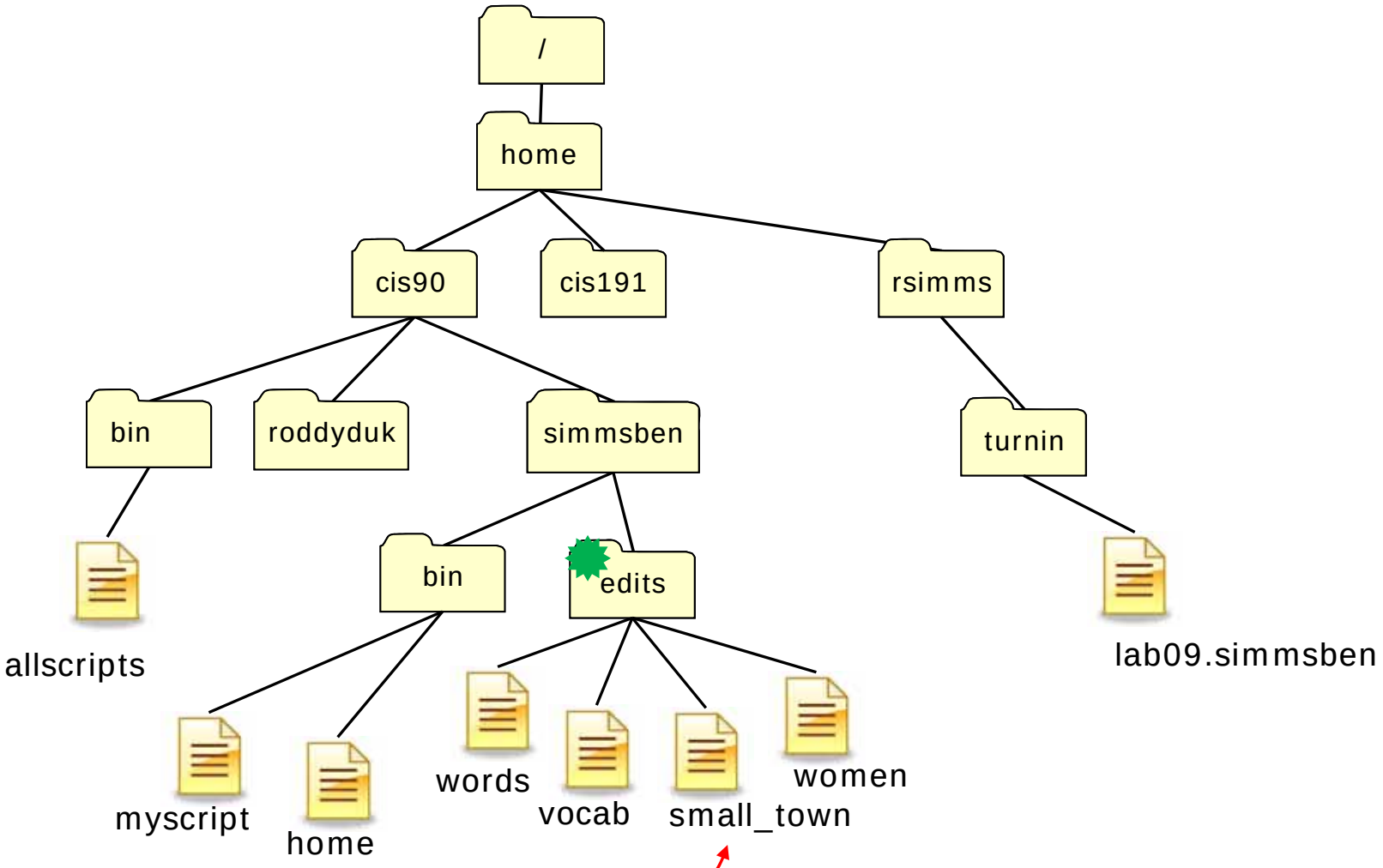
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME

relative pathname



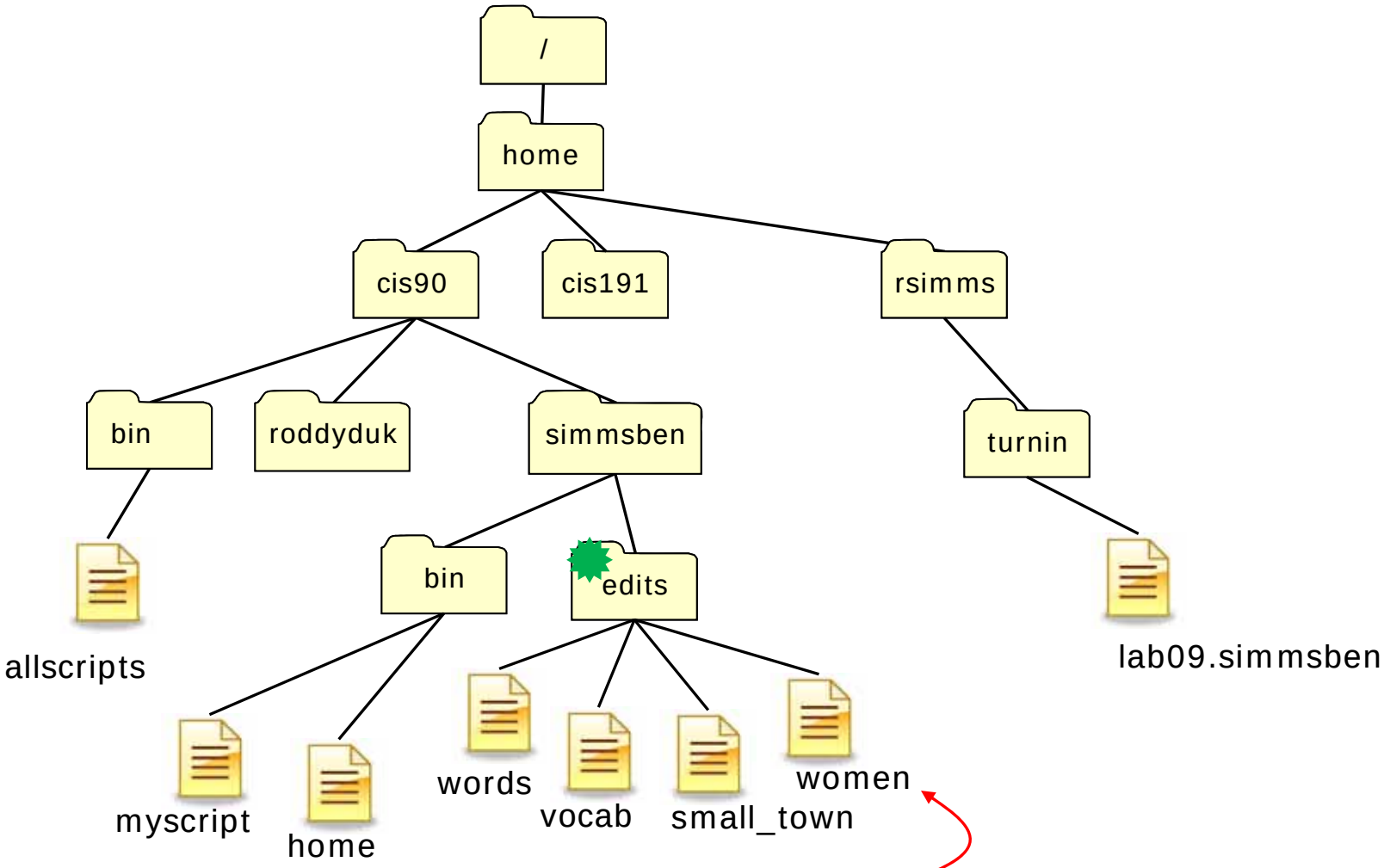
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME

relative pathname



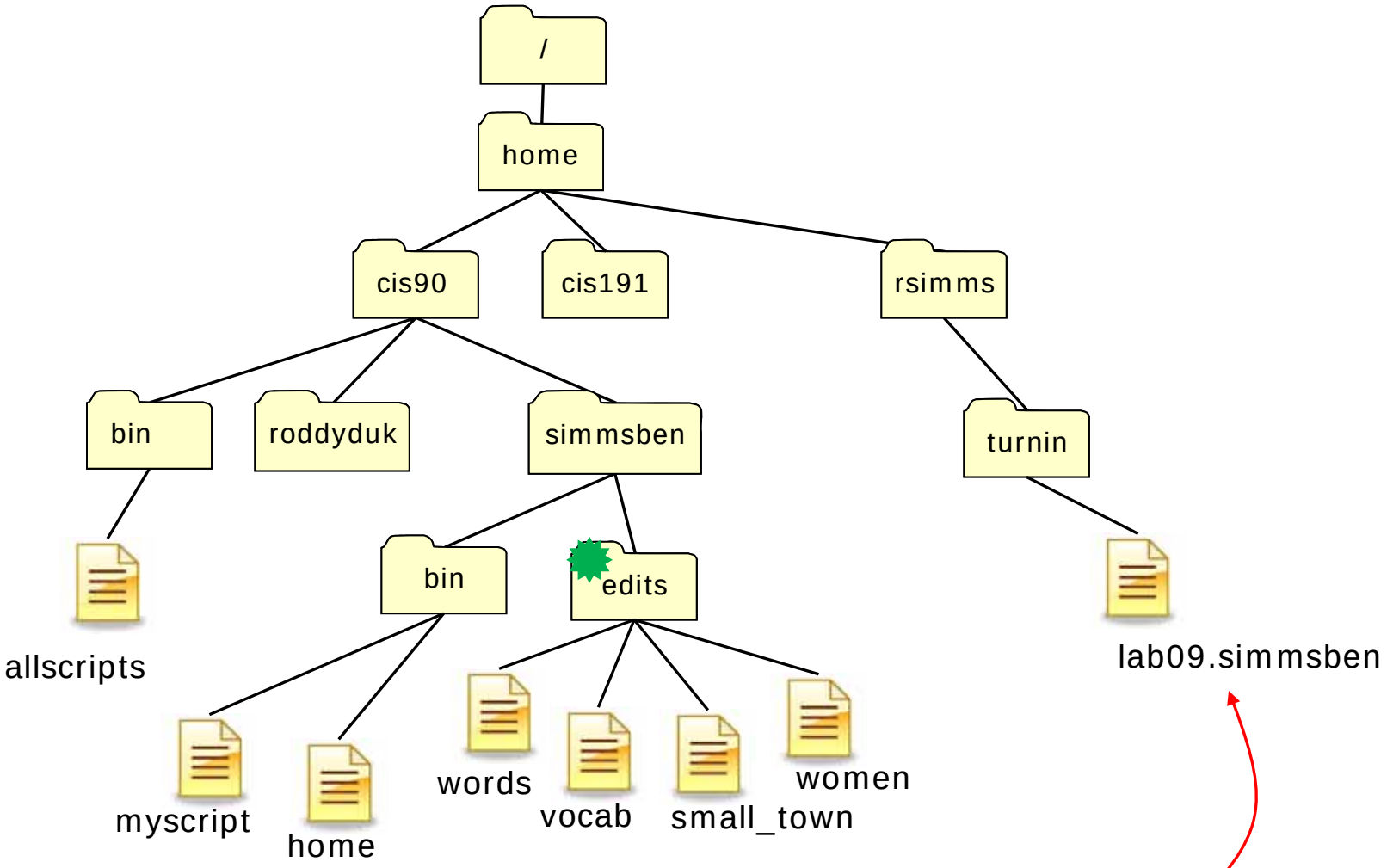
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME

relative pathname



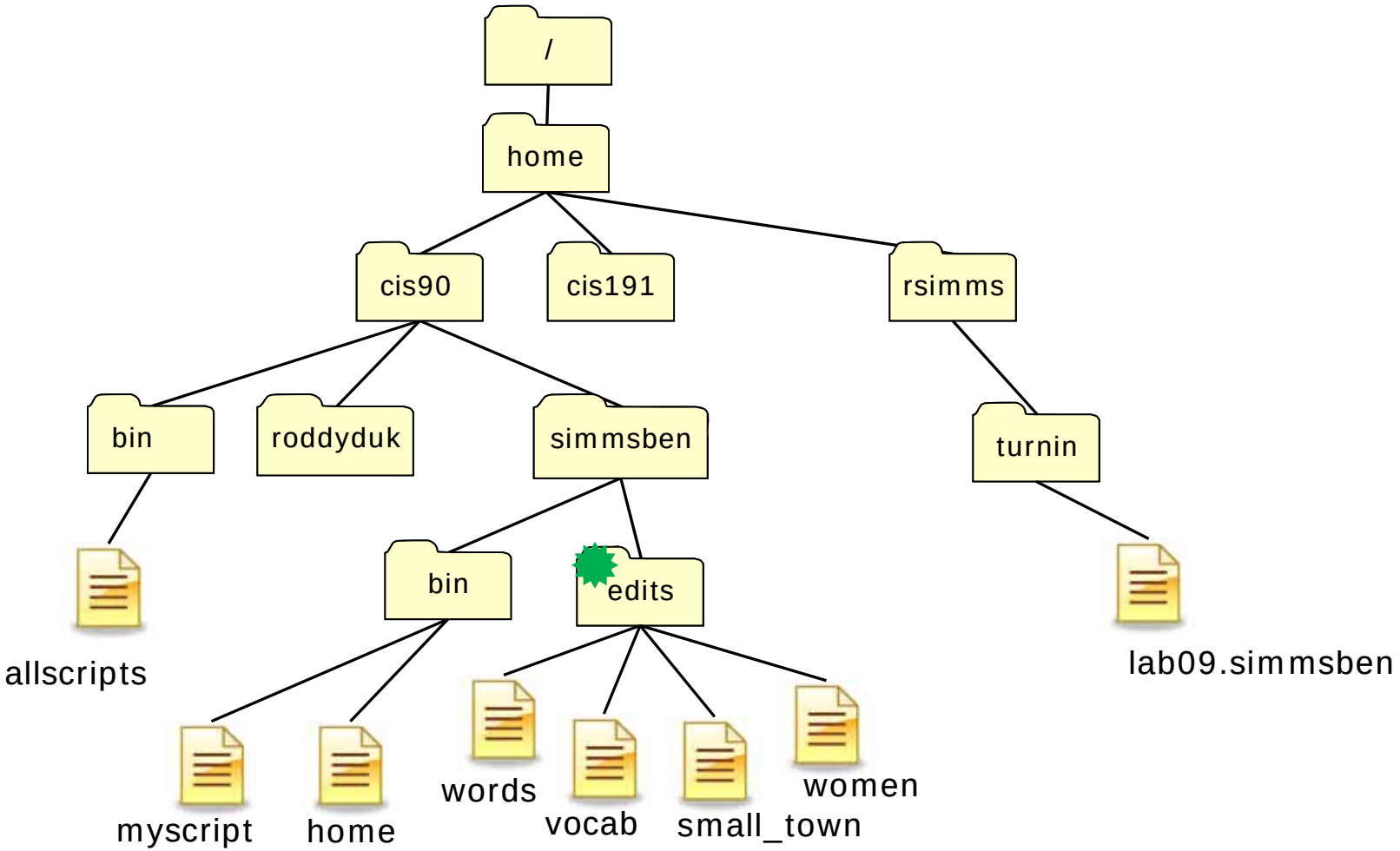
cat ../bin/home words vocab small_town woman > /home/rsimms/turnin/lab09.$LOGNAME

relative pathname



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

absolute pathname



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

vi

Best Practice - /bin/mail and vi

```
/home/cis90/simmsben $ mail roddyduk
Subject: Good bones
Hey Duke,
I really appreciate thatbone you sent me last week.
Let me knwo if you want to go mark some fench posts
this weekend.
Later,
Ben
```

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

/bin/mail and vi

```
/home/cis90/simmsben $ mail roddyduk
Subject: Good bones
Hey Duke,
I really appreciate thatbone you sent me last week.
Let me knwo if you want to go mark some fench posts
this weekend.
Later,
Ben
~v
```

Well … you could try the ~v command

/bin/mail and vi

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

/bin/mail and vi

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

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

/bin/mail and vi

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

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

&
```

The message Duke reads has all the typos fixed.

/bin/mail and vi

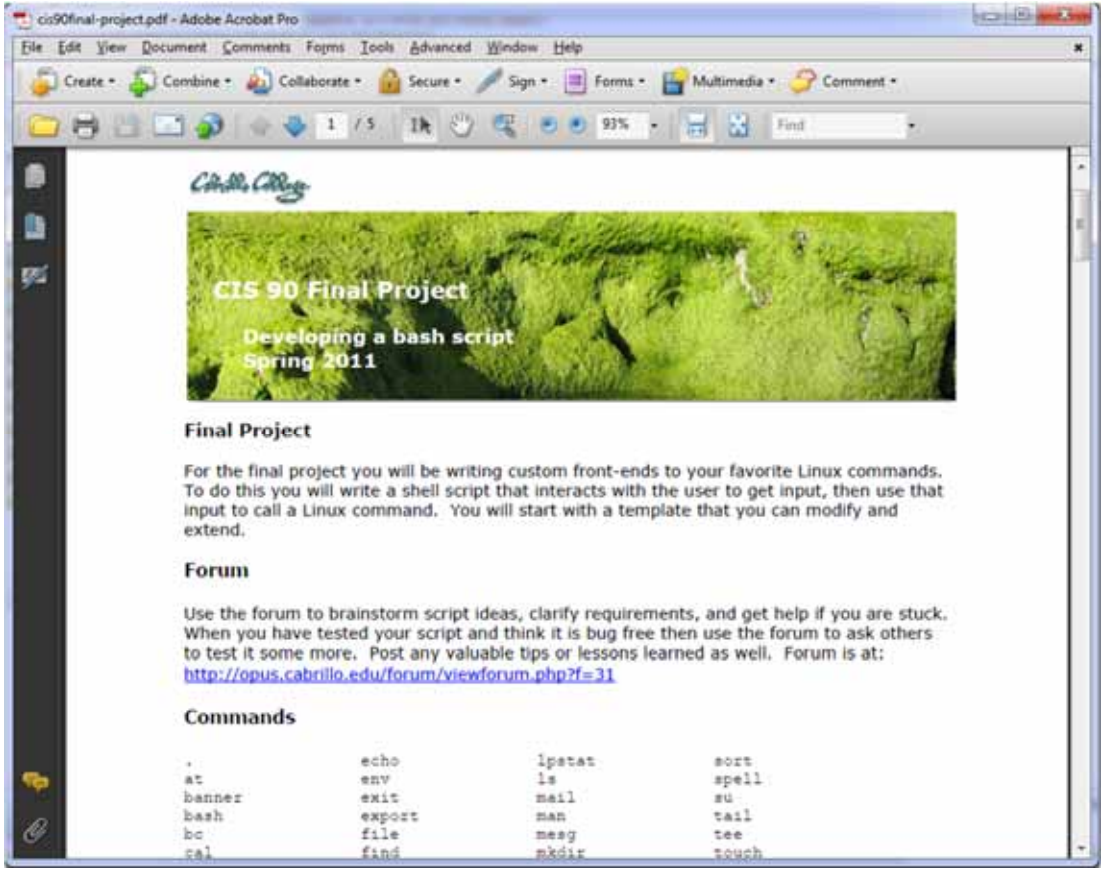
Try it!

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

cc: yourself so you can verify what you sent.

final project

Final Project

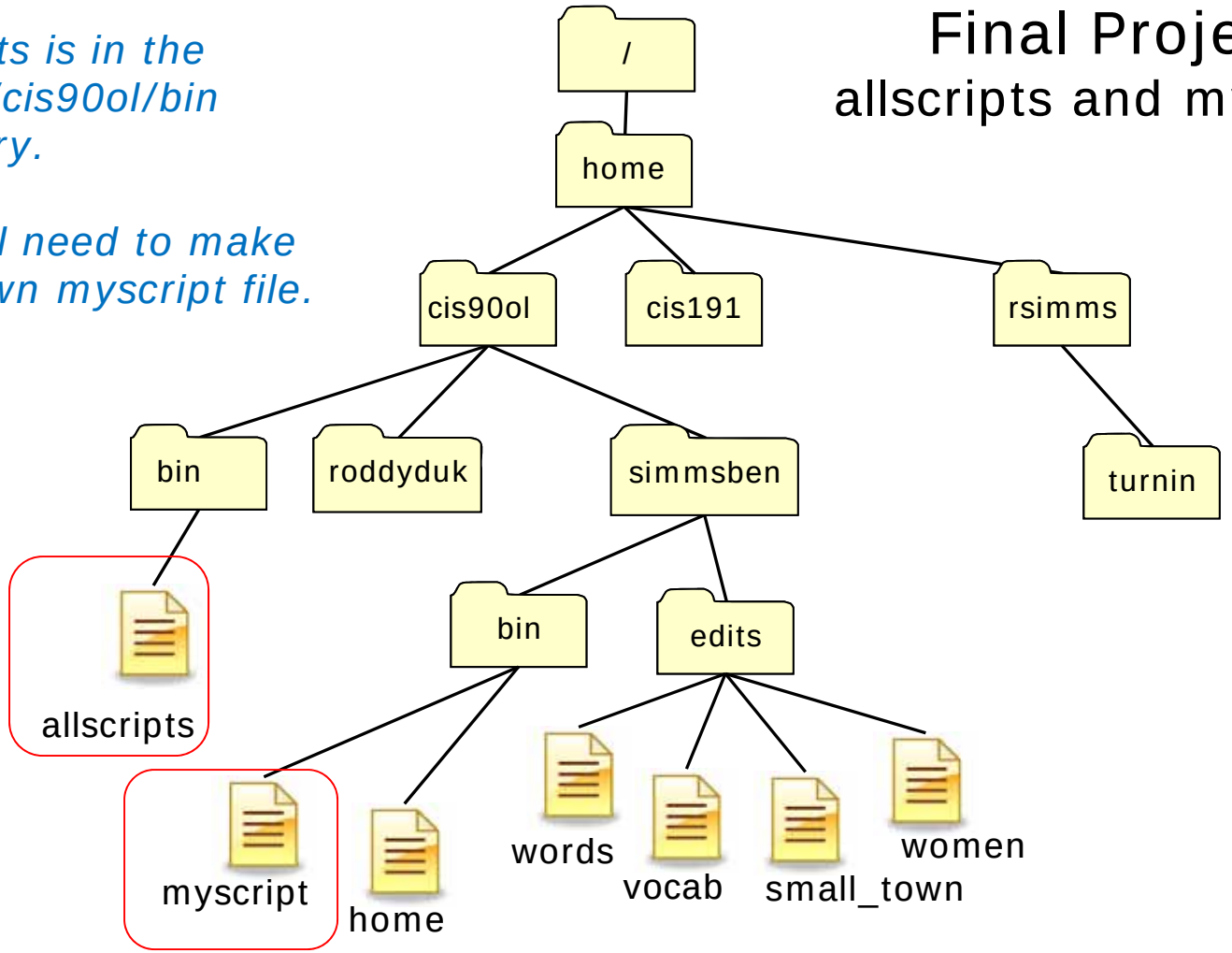


You now have the necessary skills to begin the final project!

Final Project
allscripts and myscript

allscripts is in the /home/cis90ol/bin directory.

You will need to make your own myscript file.




```
/home/cis90ol/roddyduk $ ls -l /home/cis90ol/bin/allscripts bin/myscript
-rwxr-xr-x 1 roddyduk cis90 4296 Nov 13 13:07 bin/myscript
-rwxr-xr-x 1 rsimms   staff 4381 Nov 13 18:17 /home/cis90ol/bin/allscripts
```

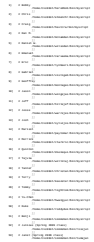
Final Project
/home/cis90ol/bin/allscripts

```
#!/bin/bash
#
# menu: A simple menu template
#
while true
do
        clear
        echo -n "
        ********************************************************************
        *                  Fall 2011 CIS 90 Online Projects                *
        ********************************************************************
        1) Bobby         7) Eric       13) Josh       19) Terry
```

More menu lines (reduced in size to fit on page)

Print a menu with each student's name

```
     Enter Your Choice: "
        read RESPONSE
        case $RESPONSE in
          1)    # Bobby
                        /home/cis90ol/herodbob/bin/myscript
                        ;;
```



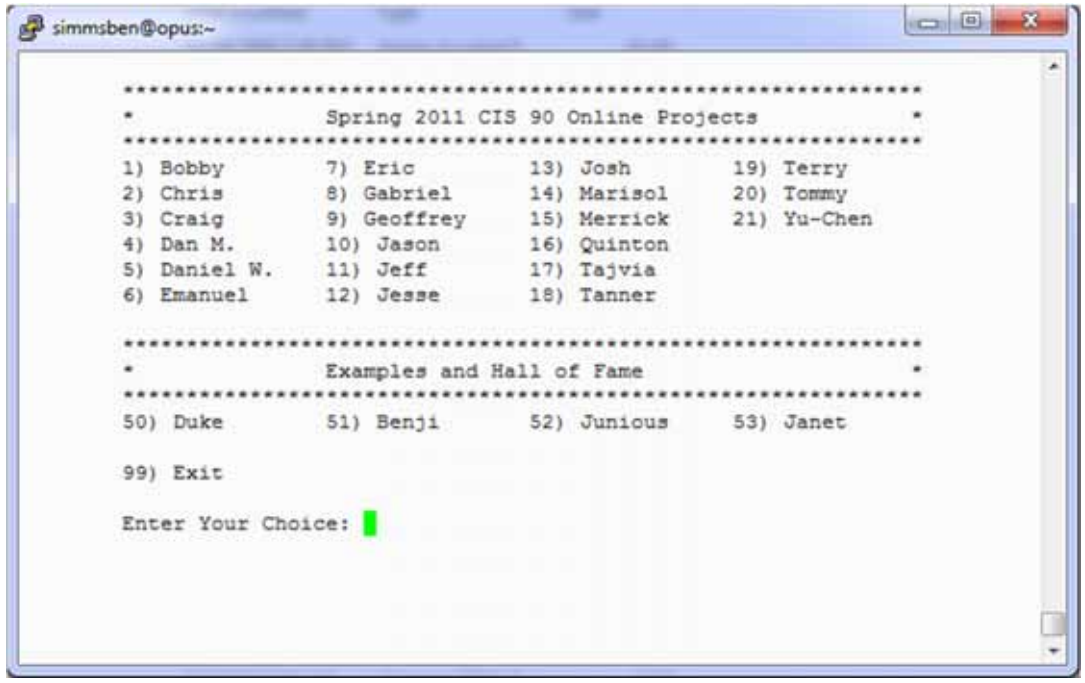
For every student name in the menu, there is a pathname to that student's myscript file

The rest of the pathnames to student's myscript files (reduced in size to fit on page)

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

Final Project
allscripts (continued)

Running **/home/cis90ol/bin/allscripts** looks like this



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

Final Project
myscript

/home/cis90ol/$LOGNAME/bin/myscript

simmsben@opus:~/bin

```
#!/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 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/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.

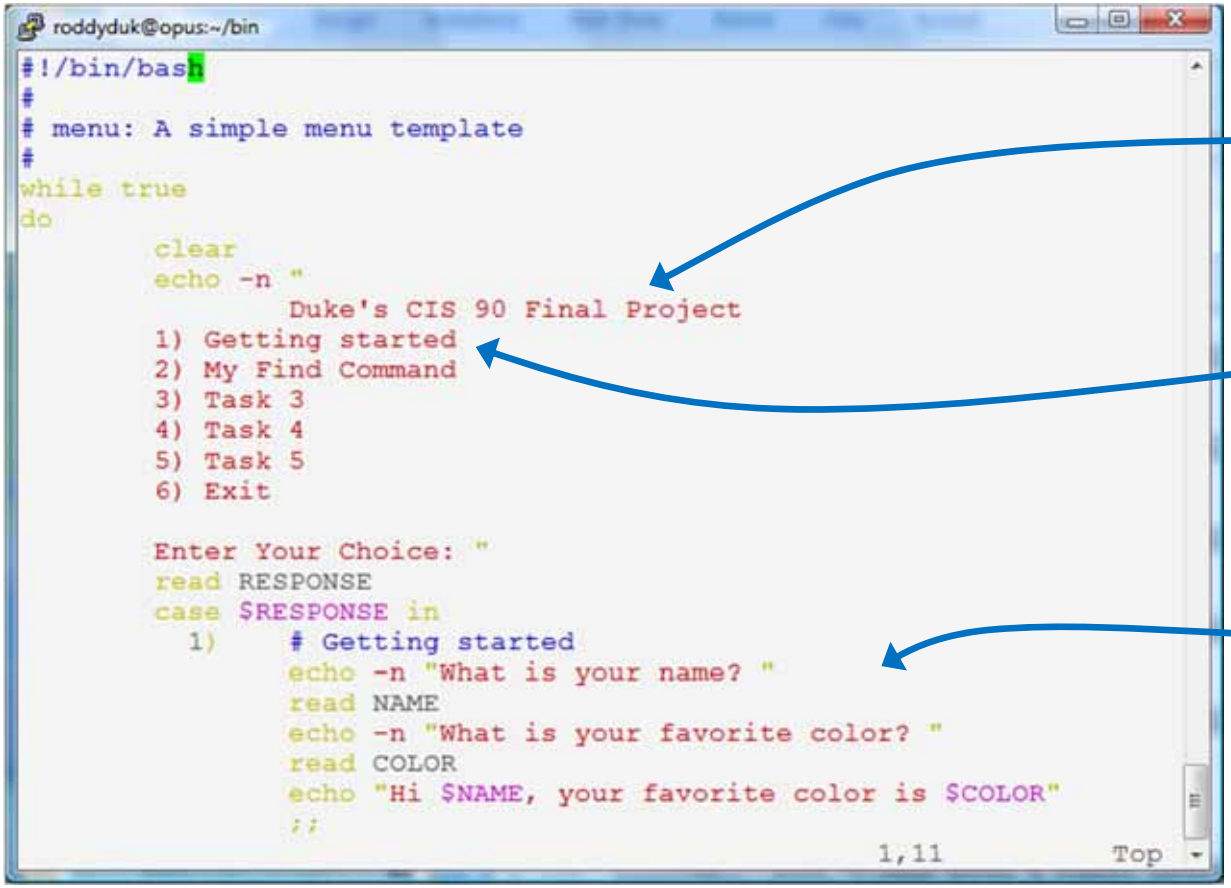
3) Click back on the vi session and click the right mouse button to paste the template code.

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

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

Final Project

/home/cis90/$LOGNAME/bin/myscript



Customize your menu title

Add a menu entry

Add some sample dialog code using variables

Final Project

/home/cis90/$LOGNAME/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/bin/myscript

case statement begins here

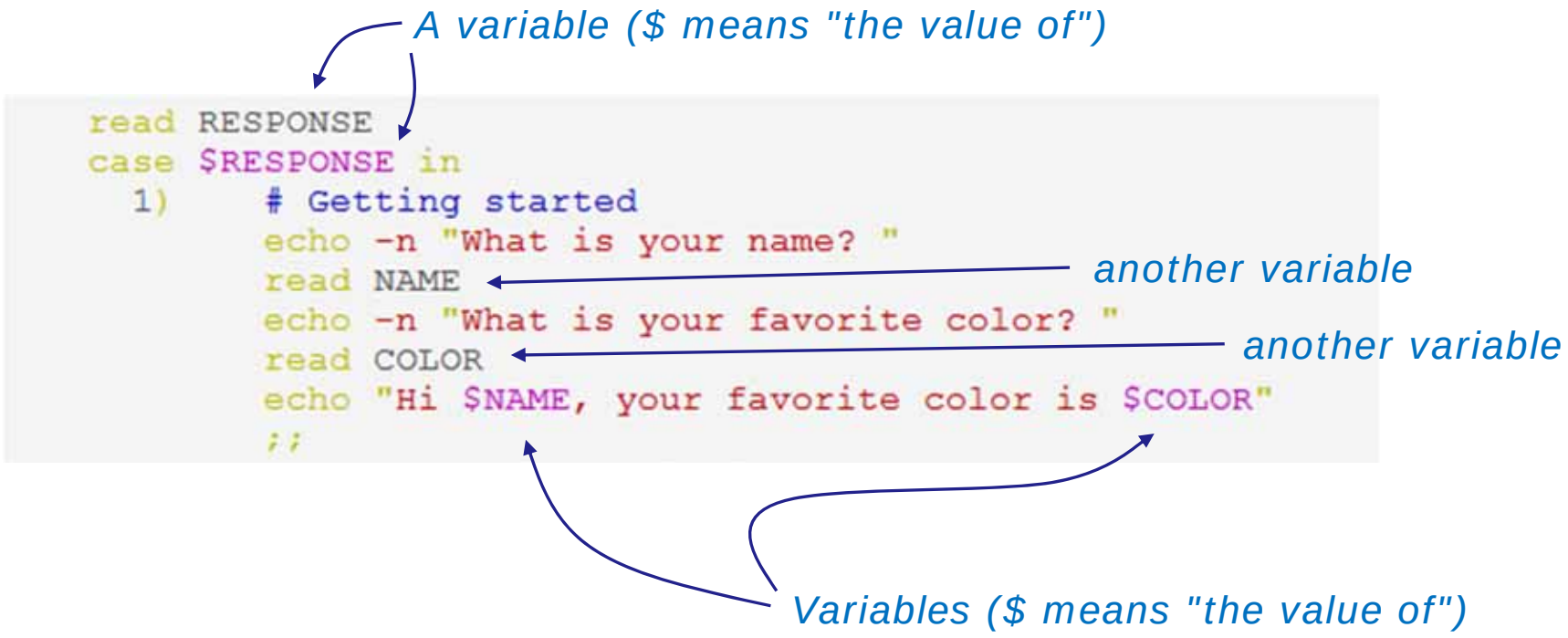
```
read RESPONSE
case $RESPONSE in
  1)    # Getting started
        echo -n "What is your name? "
        read NAME
        echo -n "What is your favorite color? "
        read COLOR
        echo "Hi $NAME, your favorite color is $COLOR"
        ;;
```

First case ends here

First case of case statement starts here

Final Project

/home/cis90/$LOGNAME/bin/myscript



Final Project

/home/cis90/$LOGNAME/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/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

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

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

Showing values of variables

Use echo **$**_____ to show value of a variable

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

[rsimms@nosmo ~]$ echo $TERM
xterm

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

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

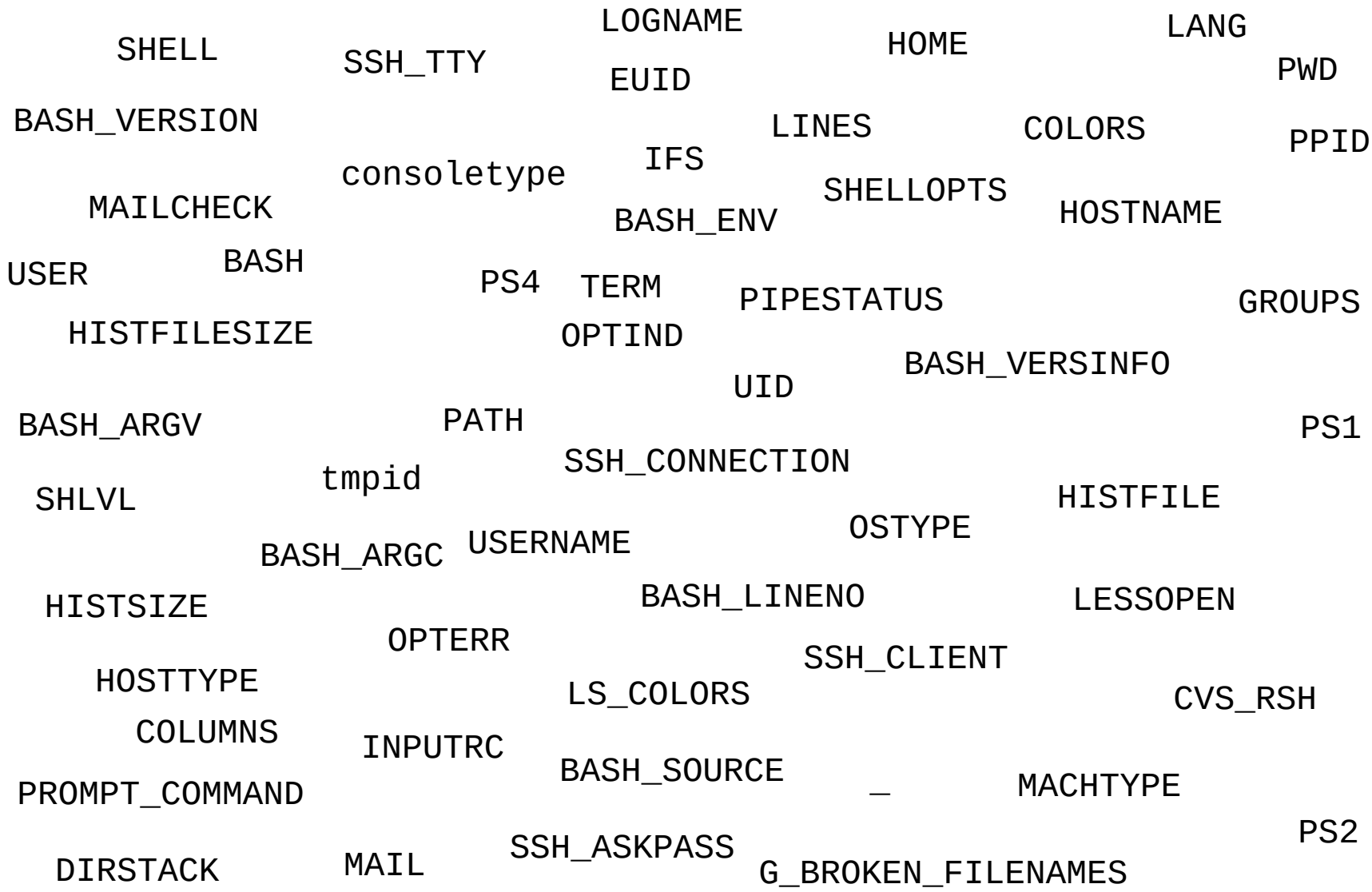
Shell Variables

Changing values of variables

Use **=** (no spaces) to change value

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


/home/cis90ol/simmsben/bin $ river="The Amazon"
/home/cis90ol/simmsben/bin $ echo $river
The Amazon
/home/cis90ol/simmsben/bin $ echo river
river
/home/cis90ol/simmsben/bin $
```

Shell Variables



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

Shell Variables

```
/home/cis90/simmsben/Poems $set
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
BASH_ENV=/home/cis90/simmsben/.bashrc
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="3" [1]="2" [2]="25" [3]="1"
[4]="release" [5]="i686-redhat-linux-gnu")
BASH_VERSION='3.2.25(1)-release'
COLORS=/etc/DIR_COLORS.xterm
COLUMNS=80
CVS_RSH=ssh
DIRSTACK=()
EUID=1160
GROUPS=()
G_BROKEN_FILENAMES=1
HISTFILE=/home/cis90/simmsben/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/home/cis90/simmsben
HOSTNAME=opus.cabrillo.edu
HOSTTYPE=i686
IFS=$' \t\n'
IGNOREEOF=10
INPUTRC=/etc/inputrc
LANG=en_US.UTF-8
LESSOPEN='|/usr/bin/lesspipe.sh %s'
LINES=24
LOGNAME=simmsben
```

The set command, by itself, will show all the shell variables

```
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/simmsben
MAILCHECK=60
OLDPWD=/home/cis90/simmsben
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/
cis90/simmsben/../bin:/home/cis90/simmsben/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/simmsben/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=simmsben
USERNAME=
_=env
consoletype=pty
```

Shell Variables

1

```
/home/cis90ol/simmsben/bin $ echo $defrost $ac $fan

/home/cis90ol/simmsben/bin $
```

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

2

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

create some new shell variables and assign values

3

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

/home/cis90/roddyduk $ echo defrost ac fan
defrost ac fan
```

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

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

Class Exercise

Create and intialize 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
```

Shell Variables

```
/home/cis90/roddyduk $ unset defrost
/home/cis90/roddyduk $ unset ac fan
```

delete the new shell variables

```
/home/cis90/roddyduk $ echo $defrost $ac $fan

/home/cis90/roddyduk $
```

Non-existing variables have null values

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

```
/home/cis90/roddyduk $ defrost=on
/home/cis90/roddyduk $ ac=off
/home/cis90/roddyduk $ fan=medium
/home/cis90/roddyduk $ set
```

```
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
BASH_ENV=/home/cis90/roddyduk/.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/roddyduk/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/home/cis90/roddyduk
HOSTNAME=opus.cabrillo.edu
HOSTTYPE=i686
IFS=$' \t\n'
IGNOREEOF=10
INPUTRC=/etc/inputrc
LANG=en_US.UTF-8
LESSOPEN='|/usr/bin/lesspipe.sh %s'
LINES=39
LOGNAME=roddyduk
LS_COLORS='no=00:fi=00:di=00;34:ln=00;36:pi=40;33:so=00;35:bd=40;33;01:cd=40;33;01:or=01;05;37;41:mi=01;05;37;41:ex=00;32:*.cmd=00;32:*.exe=00;32:*.com=00;32:*.btm=00;32:*.bat=00;32:*.sh=00;32:*.csh=00;32:*.tar=00;31:*.tgz=00;31:*.arj=00;31:*.taz=00;31:*
.lzh=00;31:*.zip=00;31:*.z=00;31:*.Z=00;31:*.gz=00;31:*.bz2=00;31:*.bz=00;31:*.tz=00;31:*.rpm=00;31:*.cpio=00;31:*.jpg=00;35:*.gif=00;35:*.bmp=00;35:*.xbm=00;35:*.xpm=00;35:*.png=00;35:*.tif=00;35:'
MACHTYPE=i686-redhat-linux-gnu
MAIL=/var/spool/mail/roddyduk
MAILCHECK=60
OLDPWD=/home/cis90/roddyduk/edits
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/cis90/roddyduk/../bin:/home/cis90/roddyduk/bin:.
PIPESTATUS=([0]="0")
PPID=7254
PROMPT_COMMAND='echo -ne "\033]0;${USER}@${HOSTNAME%%.*}:${PWD/#$HOME/~}"; echo -ne "\007"'
PS1='$PWD $ '
PS2='> '
PS4='+ '
PWD=/home/cis90/roddyduk
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=roddyduk
USERNAME=
_=
ac=off
consoletype=pty
defrost=on
fan=medium
```

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/roddyduk $ set | grep defrost
defrost=on
```

In this example:

- *the **set** command outputs all the shell variables to stdout*
- *which is piped to the stdin of the grep command*
- *the **grep** command examines each line one at a time and only outputs the lines containing the string "defrost"*

Shell Variables

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

1

```
/home/cis90/roddyduk $ vi funscript
/home/cis90/roddyduk $ 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/roddyduk $ chmod +x funscript
```

Add execute permissions so the script can be run

3

```
/home/cis90/roddyduk $ ./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

Environment Variables

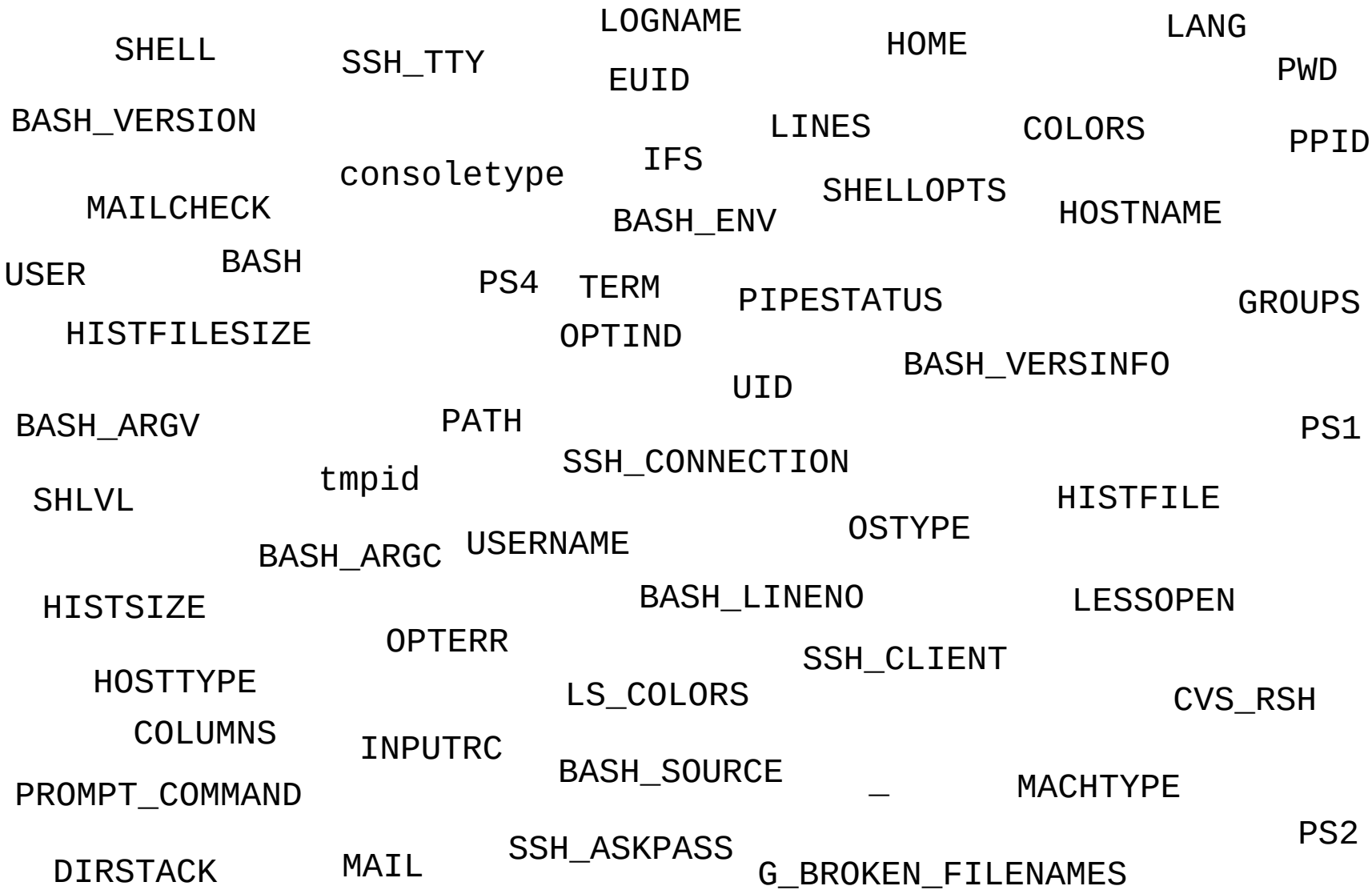
- A subset of the shell variables are environment variables.
- Environment variables are shell variables that have been *exported*.
- The **env** command will display the current environment variables and their values. Using the **export** command by itself will also show all the environment variables.
- The **export** command is used to make a shell variable into an environment variable.

  dog=benji; export dog
  or **export dog=benji**

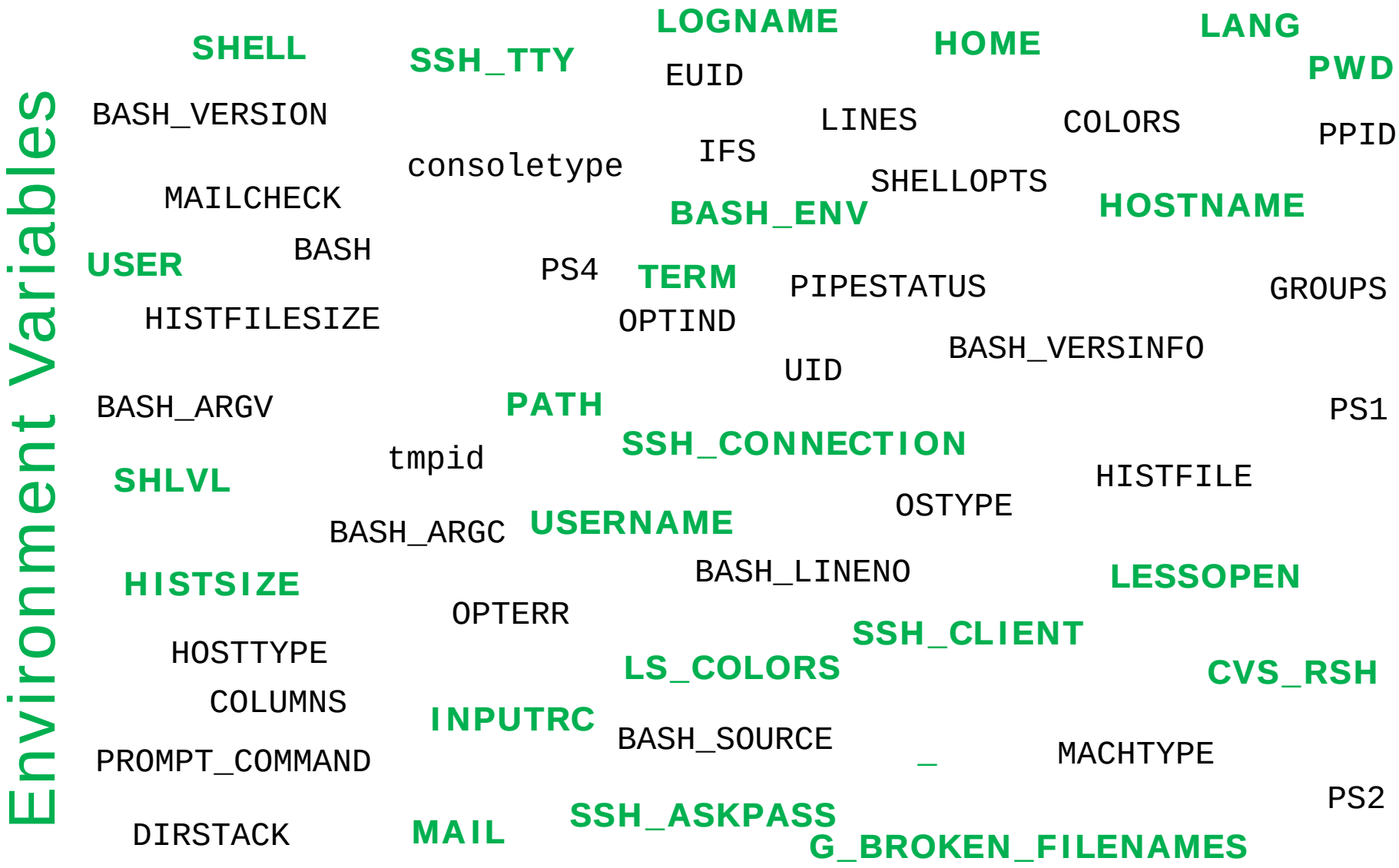
- The **export -n** command is used to make an environment variable back into a normal shell variable. E.g. **export -n dog** makes dog back into a regular shell variable.

Child processes are provided copies of the parent's environment variables. Any changes made by the child will not effect the parent's copies.

Shell Variables



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



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

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.

Shell (Environment) Variables

env command – show all environment variables

```
[roddyduk@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=roddyduk
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/roddyduk/../bin:/home/cis90/roddyduk/bin:
.
MAIL=/var/spool/mail/roddyduk
PWD=/home/cis90/roddyduk
INPUTRC=/etc/inputrc
LANG=en_US.UTF-8
fan=medium
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
HOME=/home/cis90/roddyduk
SHLVL=2
BASH_ENV=/home/cis90/roddyduk/.bashrc
LOGNAME=roddyduk
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

```
[roddyduk@opus ~]$ export
declare -x BASH_ENV="/home/cis90/roddyduk/.bashrc"
declare -x CVS_RSH="ssh"
declare -x G_BROKEN_FILENAMES="1"
declare -x HISTSIZE="1000"
declare -x HOME="/home/cis90/roddyduk"
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="roddyduk"
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/roddyduk"
declare -x OLDPWD
declare -x
PATH="/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/cis90/roddyduk/../bin:/home/cis90/roddyduk/bin:."
declare -x PWD="/home/cis90/roddyduk"
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="roddyduk"
declare -x USERNAME=""
```

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

Shell (Environment) Variables

export command – show all exported variables

To create your own environment variable use the export command

1

```
/home/cis90/roddyduk $ env | wc –l
24
/home/cis90/roddyduk $ export | wc -l
24
```

There are currently 24 environment (exported) variables

2

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

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

3

```
/home/cis90/roddyduk $ env | wc -l
25
/home/cis90/roddyduk $ export | wc -l
25
```

Now there are 25 environment variables

4

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

use grep to show the new environment variable

Flashback PS1

bash shell tip

Change PS1 to change the shell prompt

PS1 settings	Result
PS1='$PWD $'	/home/cis90/simmsben/Poems $
PS1="\w $"	~/Poems $
PS1="\W $"	Poems $
PS1="\u@\h $"	simmsben@opus $
PS1='\u@\h $PWD $'	simmsben@opus /home/cis90/simmsben/Poems $
PS1='\u@\$HOSTNAME $PWD $'	simmsben@opus.cabrillo.edu /home/cis90/simmsben/Poems $
PS1='\u \! $PWD $'	simmsben 825 /home/cis90/simmsben/Poems $
PS1="[\u@\h \W/\$"	[simmsben@opus Poems/$
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** changes as you change directories!

bash shell tip

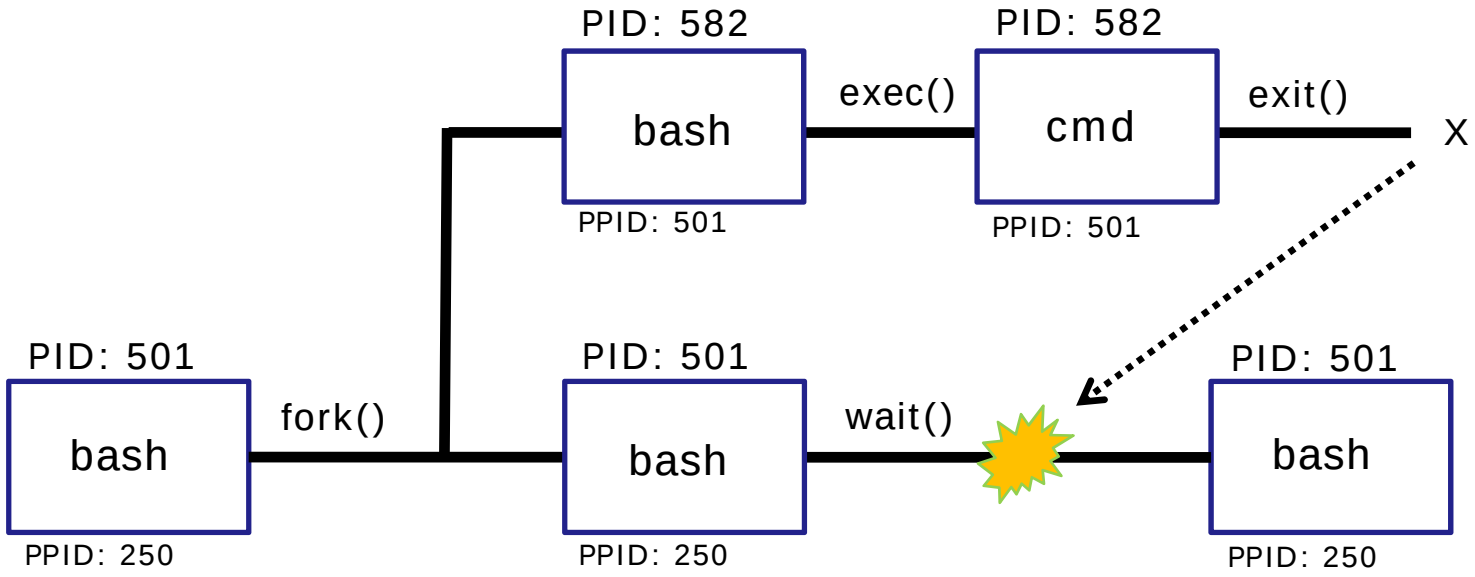
changing the prompt

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.

Shell Environment

exporting variables

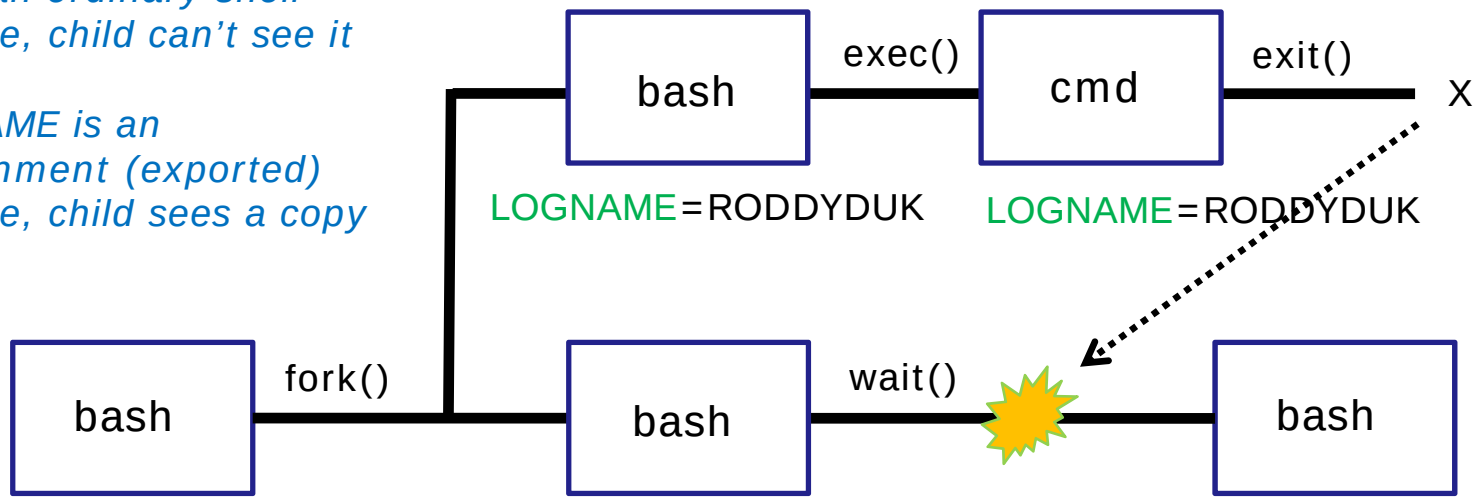


- When a shell forks a child, not all of the variables are passed on to the child.
- Only the parent's exported variables (the environment variables) are passed to the child.

Example: Only exported variables are available to the child

fan is an ordinary shell variable, child can't see it

LOGNAME is an environment (exported) variable, child sees a copy of it



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

The child gets the value of LOGNAME only.

Example: Only exported variables are available to the child

1

```
/home/cis90/roddyduk $ fan=medium
/home/cis90/roddyduk $ echo $fan $LOGNAME
medium roddyduk
```

fan is a shell variable
LOGNAME is an environment (exported) variable

parent

2

```
/home/cis90/roddyduk $ env | grep fan
/home/cis90/roddyduk $ set | grep fan
fan=medium
/home/cis90/roddyduk $ env | grep LOGNAME
LOGNAME=roddyduk
/home/cis90/roddyduk $ set | grep LOGNAME
LOGNAME=roddyduk
```

*fan only shows up in **set** output*

*LOGNAME shows up in both **env** and **set** output*

child

3

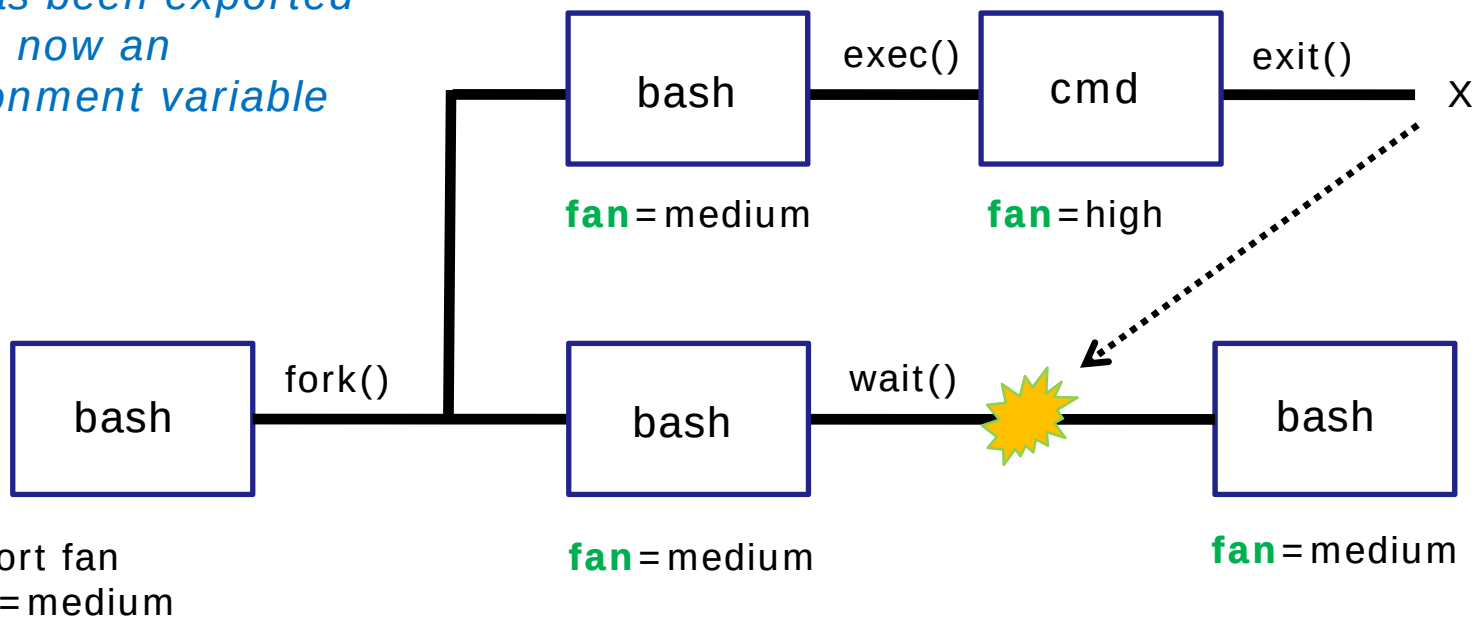
```
/home/cis90/roddyduk $ bash
[roddyduk@opus ~]$ echo $fan $LOGNAME
roddyduk
[roddyduk@opus ~]$ exit
exit
/home/cis90/roddyduk $
```

bash command runs another shell as a child process. LOGNAME is available but fan is not.

Only LOGNAME, an environment (exported) variable, is available to the child process

Example: Changes made by the child do not effect the parent

fan has been exported and is now an environment variable



- The child gets copies of the parent's environment variables which the child can change. However, any changes made by the child have no effect on the parent's variables.

The child gets a copy of the fan variable. The child can make changes to this copy but any changes made will not effect the parent's variable.

environment variables

1 *parent*

```
/home/cis90/roddyduk $ echo $fan
medium
/home/cis90/roddyduk $ export fan
```

export fan so it is available to children

2 *child*

```
/home/cis90/roddyduk $ bash
[roddyduk@opus ~]$ echo $fan
medium
```

fan is now available to the child process

3 *child*

```
[roddyduk@opus ~]$ fan=high
[roddyduk@opus ~]$ echo $fan
high
[roddyduk@opus ~]$ exit
exit
```

the child modifies the variable

4 *parent*

```
/home/cis90/roddyduk $ echo $fan
medium
```

any changes made by a child will not effect the parent's variable

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

2
```
/home/cis90/roddyduk $ type copy
copy is aliased to `cp'
/home/cis90/roddyduk $
```

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

3
```
/home/cis90/roddyduk $ alias copy
alias copy='cp'
/home/cis90/roddyduk $
```

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

4
```
/home/cis90/roddyduk $ unalias copy
/home/cis90/roddyduk $ 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 10 lines of smalltown

1

```
/home/cis90/roddyduk $ alias s="clear; head -10 ~/edits/small_town"
/home/cis90/roddyduk $ 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 mashpit on on Saturday night.
   Third Street is on the edge of town.
   Every sport is played on dirt.
   The editor and publisher of the newspaper carries a camera at all times.
   You don't use your turn signal because everyone knows where you are
      going knows where you are going.
/home/cis90/roddyduk $
```

2

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

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

3

```
/home/cis90/roddyduk $ unalias s
/home/cis90/roddyduk $
```

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

alias an alias

Yes, an alias can be made using another alias

1

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

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

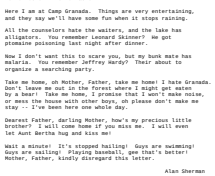
```
/home/cis90/roddyduk $ show letter
```

reduced sized to fit on page

2

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

```
/home/cis90/roddyduk $ view letter
```



reduced sized to fit on page

3

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

It can be broken too

single and double quotes

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
alias p='echo on medium off'
```

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

2

```
$ p
on medium off
```

```
$ p
on medium off
```

3

```
$ ac=off
```

```
$ ac=off
```

4

```
$ p
on medium off
```

```
$ 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 details=file**
- **details /usr/bin/spell**

Now invent 2-3 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 (runs only at login)

```
[roddyduk@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 `

[roddyduk@opus ~]$
```

sources the .bashrc file

Adds the user's bin and current directories to the path

These variables are set

Prompt (PS1) is defined

These 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. e.g.

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

# User specific aliases and functions

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
alias print="echo -e"
[roddyduk@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 desireable 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*** or **source** ***myscript***

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.

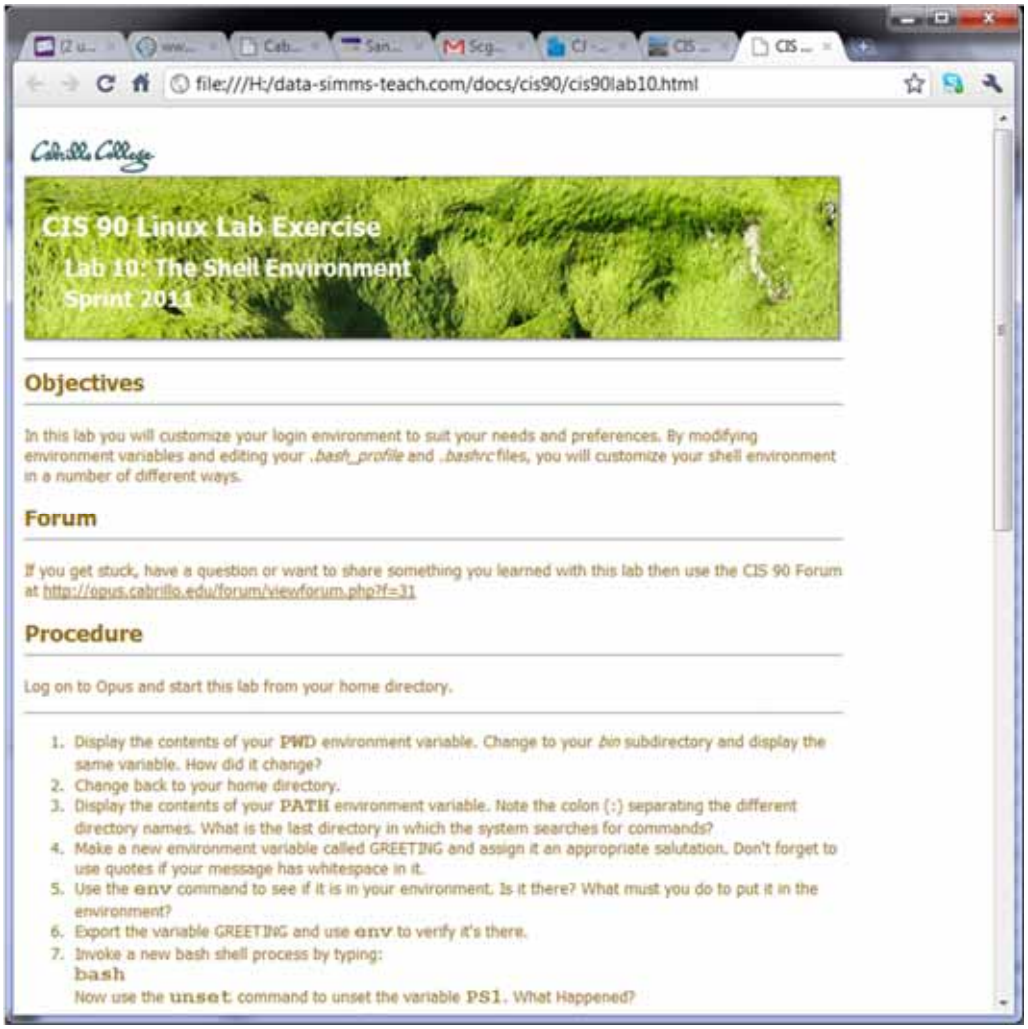
If a UNIX command is run using the exec command, the bash code in the process is replaced by the command code, when finished the process will terminate when that command exits

exec clear

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

Wrap up

Lab 10 - the last one!



file:///H:/data-simms-teach.com/docs/cis90/cis90lab10.html

Cabrillo College

CIS 90 Linux Lab Exercise

Lab 10: The Shell Environment

Sprint 2011

Objectives

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

If you get stuck, have a question or want to share something you learned with this lab then use the CIS 90 Forum at http://opus.cabrillo.edu/forum/viewforum.php?f=31

Procedure

Log on to Opus and start this lab from your home directory.

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?

Extra Credit Special

1) Why did the prompt change?

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

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

For 5 points extra credit, email risimms@cabrillo.edu your complete answer by noon tomorrow (May 6, 2011)

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

spell command

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

spell command flags CIS as misspelled word.

How can we add CIS to the dictionary?

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

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

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

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

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

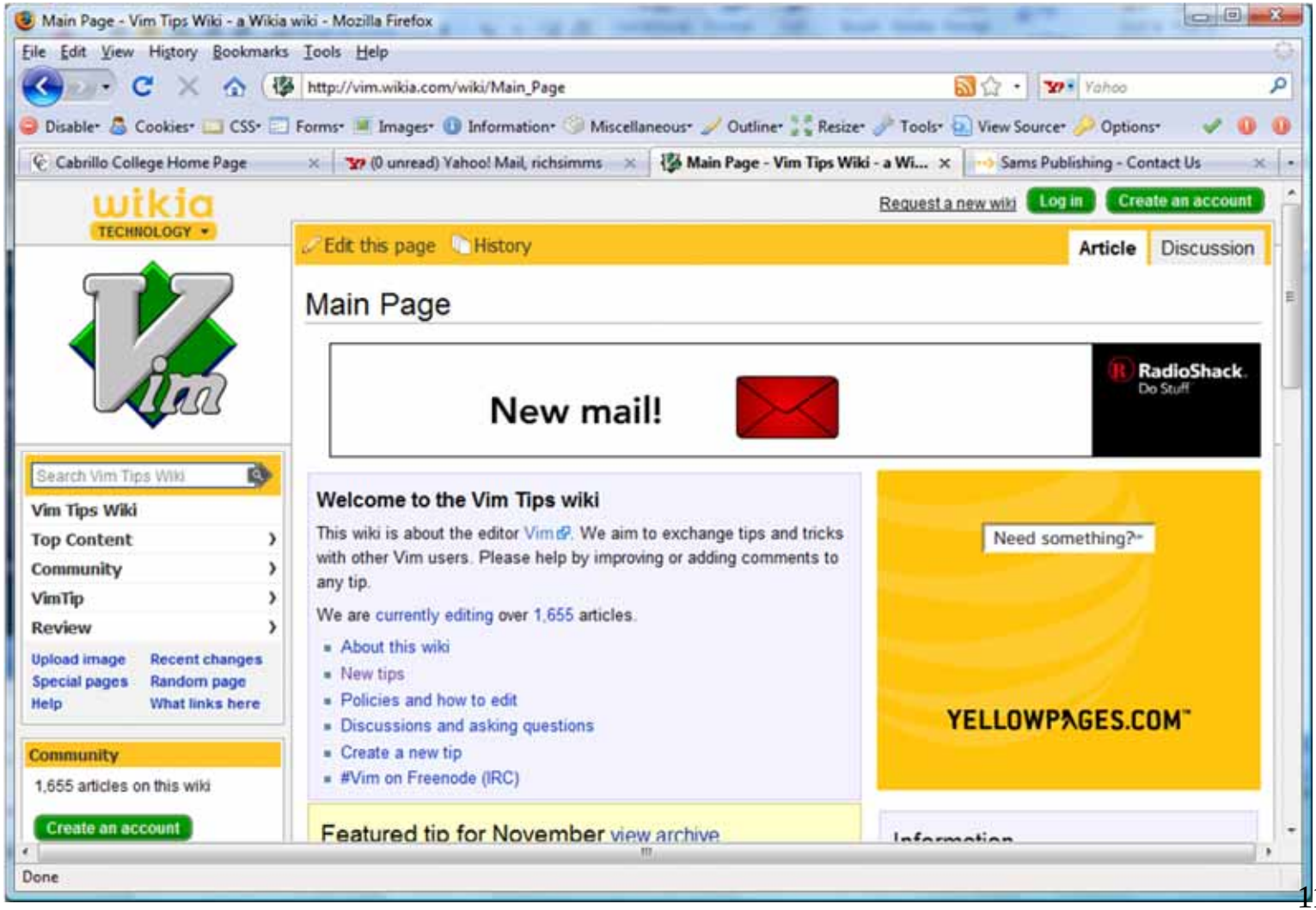
*OK, the actual command is **aspell***

aspell command

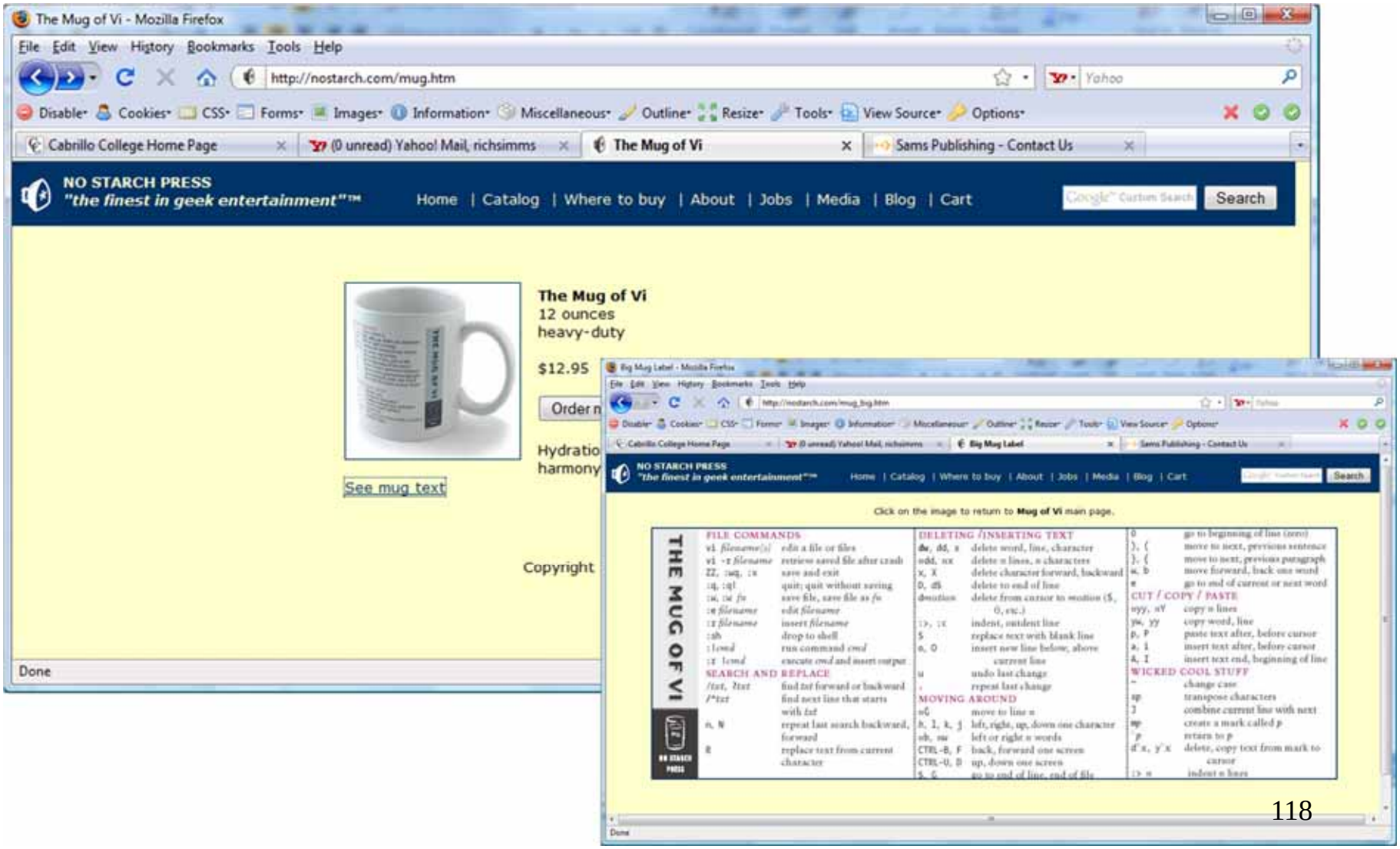
```
ASPELL(1)                Aspell Abbreviated User's Manual                ASPELL(1)

NAME
       aspell - interactive spell checker

SYNOPSIS
       aspell [options] <command>

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

COMMANDS
       <command> is one of:

       -?,help
              display the help message

       -c,check file
              to spell-check a file
```

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

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



The Mug of vi


The Mug of Vi - Mozilla Firefox
http://nostarch.com/mug.htm
NO STARCH PRESS
"the finest in geek entertainment"™
Home | Catalog | Where to buy | About | Jobs | Media | Blog | Cart
Search
The Mug of Vi
12 ounces
heavy-duty
$12.95
See mug text
Copyright
Done
Click on the image to return to Mug of Vi main page.
THE MUG OF VI
FILE COMMANDS
DELETING / INSERTING TEXT
SEARCH AND REPLACE
MOVING AROUND
CUT / COPY / PASTE
WICKED COOL STUFF