

# CS-344 - Unix Operating System Fundamentals

## Lecture 7 Specifying Instructions to the Shell

Based on slides created by  
Dr. Bangalore for the  
Spring 2005 offering of  
the course

### Declaring and Using Variables in the Shell

- Declaring Variables: `variable_name = value`
- Display Variables: `echo $variable_name`
- Examples:
  - `a=Hello`
  - `b=date`
  - `$b`
  - `c='date'`

10/13/2006

3

### Declaring and Using Variables in the Shell

- Difference between single and double quotes
  - Double quotes ignore most special characters, but command and variable substitution is performed
  - Single quotes, no substitution performed:
  - Examples:
    - `d="You are currently logged on to `hostname`"`  
You are currently logged on to vulcan5.cis.uab.edu
    - `e="Today's date is: $b" (b=date)`
    - `f="Today's date is: $c" (c='date')`
    - `echo $e $f`  
Today's date is: date Today's date is: Fri Oct 13 10:03:37 CDT 2006

10/13/2006

4

### Overwriting Existing Files

- When output is redirected to a file, if the file already exists it's overwritten.
- To avoid overwriting existing files set the `noclobber` variable: `set -o noclobber`
- When `noclobber` variable is set, the shell complains that the file exists
- To overwrite a file when `noclobber` variable is set, use `>!` instead of `>` for output redirection
- Example: `date >! filename`

10/13/2006

5

### Avoiding Accidental Removal of Files

- Unlike Windows, in UNIX when files are deleted they cannot be undeleted
- `noclobber` feature is an instruction to the shell not to the UNIX commands (e.g., `cp`, `mv`, `rm`)
- To avoid accidental removal of files use `-i` option with `cp`, `mv`, `rm`
- set alias for `cp`, `mv`, `rm`, etc. to avoid using `-i` every time that you use those commands
  - `alias rm='rm -i'`
  - `alias cp='cp -i'`
  - `alias mv='mv -i'`
- To override an alias use `"\command"` instead of command (e.g., `\rm filename`)

10/13/2006

6

## Redirecting Error Messages

- Utilities write to the error stream when an error occurs during their execution
  - `ls -l filename` (if the filename does not exist or the file permissions are not sufficient)
- We have seen output redirection using ">" and ">>". To redirect error messages use ">2" or ">>2" (in bash)
- The shell assigns a 1 for the standard output stream and 2 for the standard error stream ("*>*" is same as "*1>*")
- To redirect both error and output together use ">&1" after the filename
  - *Print stdout to screen, stderr to outerr*  
`ls -l myfile xyz 2> outerr`
  - *Print stdout AND stderr to outerr*  
`ls -l myfile xyz > outerr 2>&1`

10/13/2006

7

## Communicative Shell

- The expansion and modification made by the shell when multiple commands or special characters are used can be viewed by starting the shell with `-x` option
- We can start a new shell by typing "`bash -x`" or execute a script "`bash -x myscript`" from the current shell

```
$ bash -x
$ tr '\t' ':' < quizscores | sort
+ tr '\t' :
+ sort
.....
.....
```

```
$ bash -x
$c="You are currently logged on to 'hostname'"
++ hostname
+c="You are currently logged on to blazer5"
$ echo $c
+c= You are currently logged on to blazer5
You are currently logged on to blazer5
```

10/13/2006

8

## Shell Command Line Expansion (I)

- Wildcard character – \*
  - Matches zero or more characters
  - List all files that end with .java – `ls *.java`
  - List all files that start with cs – `ls cs*`
- Matching character – ?
  - Matches one character
  - List all files that start with chapter followed by any character – `ls chapter?`
- \* and ? are special characters called meta-characters. These characters are interpreted by the shell and should not be used in filenames

10/13/2006

9

## Shell Command Line Expansion (II)

- To specify filename within a specified range use `[]`
  - Only one character is matched
  - List all files that start with chapter and has a number following it within the range 1-5 – `ls chapter[1-5]`
- To match and create multiple filenames from one pattern use `{}`
  - Curly braces match files specified in the braces, but will NOT expand ranges
  - List all files that start with chapter and has specific numbers following it – `ls chapter{2,3,4}`
- All meta-characters can be combined together
  - `ls *[0-9]?{java,c,cpp}`

10/13/2006

10

## Local & Environment Variables

- Use "`export variable_name`"
  - `a=Hello`
  - `export a`
  - `export b=World`
  - `export PATH=${PATH}:/bin`
- Remove variable – `unset`
  - `unset a`
- List all variables – `set`
- List environment variables – `env`

10/13/2006

11

## Miscellaneous

- `view`: similar to `vi` with the read-only flag set
- `split`: split long files into smaller files
  - `split -l linecount filename suffix`
    - `split -l 20 longfile file+`

10/13/2006

12

## Locating Files (find)

- Locate each file with a specific file-name and display its pathname.
  - `find ~ -name file-name -print`
  - `~` → starting directory (in this case the user's home directory)
  - `-name` → find all files with the specified name `file-name`
  - `-print` → output the full path name for each match.

10/13/2006

13

## Installing software

- Steps involved:
  - Download
  - `./configure`
  - `make`
  - `make install`
- Download and install Apache HTTP server
  - Download tar gzip file to one of the vulcan machines using the following comand:  
`wget http://apache.us.lucid.dk/httpd/httpd-2.2.3.tar.gz`
  - Install the software in your home directory
  - Add Apache to your path
- To recover the files of a gzip/tar file:
  - `gunzip -c httpd-2.2.3.tar.gz | tar -xvf -`

10/13/2006

14