Week 2
Shell Scripting, RegEx, and Streams

16 January 2019

CS 35L Lab 4

Jeremy Rotman

Announcements

- Assignment #1 **was** due January 12 by 11:55pm
  - You can still submit the assignment
  - If you submit before 11:55pm tonight, it's only an 8% penalty!
- Assignment #2 is due January 23 (Wednesday) by 11:55pm

Questions?

Outline

- ➔ RegEx examples
- ➔ Bash Arrays
- ➔ More Useful Commands
- ➔ Homework 2 Tips

RegExamples

Assume for the following that we are using egrep (grep -E) to search for lines in a file

RegExamples

What would you use to match lines that begin with a “th” not case specific?

RegExamples

What would you use to match lines that begin with a “th” not case specific?

^[Tt][Hh]

RegExamples

How about lines that end with punctuation?

RegExamples

How about lines that end with punctuation?

[[:punct:]]$

RegExamples

What would the following RegEx give you?

[[:digit:]][ ]+[[:digit:]]

RegExamples

What would the following RegEx give you?

[[:digit:]][ ]+[[:digit:]]

Any line where there are two digits separated by 1 or more spaces

RegExamples

Given the RegEx: ((do*t) \2)+

Which of the following would match?

A. doot doot
B. doot doot doot
C. doot
D. doot doot Mr. Skeltal

RegExamples

Given the RegEx: ((do*t) \2)+

Which of the following would match?

A. doot doot
B. doot doot doot
C. doot
D. doot doot Mr. Skeltal

RegExamples

Given the RegEx: [Hh]ello.*[Ww]orld

Which of the following would match?

A. Hello World
B. helloworld
C. Hello to the most beautiful world I have ever seen
D. Hi World

RegExamples

Given the RegEx: [Hh]ello.*[Ww]orld

Which of the following would match?

A. Hello World
B. helloworld
C. Hello to the most beautiful world I have ever seen
D. Hi World

RegExamples

Given the Line: LEEEEEEEEEEROY JENKINS

Which of the following regular expressions would match this?

A. LE{3}ROY JENKINS
B. LE{2,30}ROY JENKINS
C. LE?ROY JENKINS
D. L[a-z]*ROY JENKINS

RegExamples

Given the Line: LEEEEEEEEEEROY JENKINS

Which of the following regular expressions would match this?

A. LE{3}ROY JENKINS
B. LE{2,30}ROY JENKINS
C. LE?ROY JENKINS
D. L[a-z]*ROY JENKINS

Bash Arrays

- `declare -a hw`
  - Creates an array named hw
- `hw[0]="hello"`
  - Sets the first (zero-based indexing) element of hw to “hello”
- `for x in ${hw[@]}`
  - Iterates over all elements of hw
  - @ references all members of the array
- `${hw[@]}`
  - References the array

Bash Arrays

- `${#hw[@]}`
  - Length of the array
- `${#hw[0]}`
  - Length of element 0 in the array

Command: head

Outputs the first part of files

```
head [OPTION]... [FILE]...
```

➔ Useful Option

- `-n` *K*
  - Print the first *K* lines (instead of default 10)
  - Use -*K* to print all but last *K* lines

Command: find

- Another useful option
  - `-exec COMMAND {} +`
    - Can be used to run commands on all results of find
    - `find . -type f -exec file '{}' \;`
      - Run the file command on every file in or below the current directory
      - '{}' is used to reference the file currently being processed
      - ; ends parsing for arguments to the command

Command: grep

- A few more potentially useful grep options
  - -m *NUM*
    - Stop reading a file after *NUM* matching lines
  - -H
    - Print the filename for each match (prefix to line match)
    - Default if grep is given multiple files
  - -n
    - Prefix the line of output with 1-based line number within its input file

Command: wc

Prints newline, word, and byte counts for a file

- Useful Option
  - -l
    - Print only the newline count (I.E. prints the number of lines)
- Helpful note
  - If you try to save the number of lines in a file to a variable:
    - LINE_COUNT=`wc -l file.txt`
      - This will not work as expected, it includes a prefix indicating filename
    - LINE_COUNT=`cat file.txt | wc -l`
      - This should save just the number

Homework #2

- You are working on a project that has many text files
  - Some are in plain ASCII, and some are in UTF-8, a superset of ASCII
- UTF-8 files that contain characters outside of ASCII are supposed to have a first line including:
  - `-*- coding: utf-8 -*-`
- Unfortunately, people are bad at including this heading
- We are going to write scripts to identify files that should or shouldn't have the heading
  - And determine if the heading is there

Homework #2

- For all of the scripts
  - Accept 1 or more arguments (filenames)
    - If you receive a directory, recursively look at all files under the directory or its subdirectories
  - Although UTF-8 is a superset of ASCII, we will refer to UTF-8 files as a file that contains any UTF-8 specific characters

Homework #2

- 4 Scripts in total
  - find-ascii-text
    - Output a line for each argument that names an ASCII text file
  - find-utf-8-text
    - Output a line for each argument that names a UTF-8 text file
  - find-missing-utf-8-header
    - Output a line for each argument that names a UTF-8 text file that lacks the “`-*- coding: utf-8 -*-`” line in the first line
  - find-extra-utf-8-header
    - Output a line for each argument that names an ASCII text file but includes the “`-*- coding: utf8 -*-`” line in the first line

Homework #2

- ➔ You do not need to worry about cases where you receive no arguments
- ➔ Be prepared to handle file arguments that have special characters
  - ◆ Spaces, *'s, and leading -'s
  - ◆ You do not need to worry about filenames with newlines

Homework #2 Hints

- `test [ expression ]`
  - `[ -d pathname ]`
    - True if the pathname resolves to an existing directory
  - `[ -f pathname ]`
    - True if the pathname resolves to an existing directory entry for a regular file
- `head -n1 filename`
  - This will output just the first line of *filename*