CS 35L

Software Construction Laboratory

Lecture 2.2

10th October, 2019

Logistics

- Assignment 2 Deadline
  - Deadline - Monday, 14th October, 11:55pm
- If you are looking for PTE's or wanting to switch labs, continue to write your name on the sheet of paper
- Assignment 10
  - Will create a sheet for presentations from Week 3
- Hardware requirement for Week 8
  - Seeed Studio BeagleBone Green Wireless Development Board
  - Buy individual boards

Review - Previous Lab

- Shell Scripting
  - Basics
  - Conditional and Iterative Statements
- Regular Expressions
  - BRE vs ERE
  - Examples

Sed

- stream editor, modifies the input as specified by the command(s)
- Can be used for:
  - Printing specific lines
    - sed –n ‘1p’ sedFile.txt
    - sed –n ‘1,5p’ sedFile.txt
    - sed –n ‘1~2p’ sedFile.txt
- Deleting text
  - sed '1~2d' sedFile.txt
- Substituting text - s/regex/replacement/flags
  - sed ‘s/sed/ned/’ sedfile.txt
  - sed ‘s/sed/ned/g’ sedfile.txt
  - sed 's/<[^>]*>//g' a.html
  - sed ‘s/regExpr/replText/’ filename

Sed - contd..

- sed –n 12,18p sedfile.txt
- sed 12,18d sedfile.txt
- sed ‘1~3d’ sedfile.txt
- sed ‘1,5 s/line/Line/g’ sedfile.txt
- sed ‘/pattern/d’ sedfile.txt

Grep command

- A Unix command to search files/text for the occurrence of a string of characters that matches a specified pattern
- Usage:
  - grep [option(s)] pattern [file(s)]
- grep –c ‘<[^>]*>’ sedfile.txt
- grep –n ‘<[^>]*>’ sedfile.txt
- ls –l | grep “.txt”

Grep Continued

Lab2.txt:

- This is the second lab.
- We are studying commands.
- Their uses are many.
- THIS LAB IS TEACHING US COMPUTER SCIENCE.
- Soon this can help us do great things there.

- $grep -i ”THis" lab2.txt
- $grep -c ”this" lab2.txt
- $grep -w ”us" lab2.txt
- $grep –e “their” –e “there” lab2.txt

awk

- awk is more than a command; it’s a programming language by itself
- Utility/language for data extraction
- awk views a text file as records and fields
- Usage:
  - Awk options ‘/search pattern/ {Actions}’ file
- Examples:
  - awk ‘{print;}’ file.txt   // print the file line by line; default behaviour
  - awk ‘/Hello/ {print;}’ file.txt // prints lines which matches Hello

Regex Question

- Which regular expression would match all subdirectories within a directory?

Lab Assignment 2

- hwnwdseng.htm -> buildwords -> hwords
- Buildwords
  - Read from STDIN and perform work on input
- Store the output in hwords
  - E.g. cat hwnwdseng.htm | sh buildwords > hwords

Lab Assignment 2 contd…

- How to construct buildwords?
  - Extract lines which contain words (both English and Hawaiian) (Hint: <td> tag)
  - Get lines with Hawaiian words
    - Even numbered lines(how do we access the line number of records?)
  - sed 's/<[^>]*>//g' a.html to remove all HTML tags
  - Remove leading space
    - sed 's/^\s*//g’
  - Substitute space in between words to newline
  - Delete all commas
  - Delete entries which have any character other than Hawaiian
  - Sort unique

Assignment 2 - Laboratory

- Build a spelling checker for the Hawaiian language
  - Get familiar with sort, comm and tr commands!
- Steps:
  - Download a copy of web page containing basic English-to-Hawaiian dictionary
  - Extract only the Hawaiian words from the web page to build a simple Hawaiian dictionary. Save it to a file called hwords (site scraping)
  - Automate site scraping: buildwords script (cat hwnwdseng.htm | buildwords > hwords)
  - Modify the command in the lab assignment to act as a spelling checker for Hawaiian
  - Use your spelling checker to check hwords and the lab web page for spelling mistakes

Useful Text Processing Tools

- wc: outputs a one-line report of lines, words, and bytes
- head: extract top of files
- tail: extracts bottom of files
- tr: translate or delete characters
- grep: print lines matching a pattern
- sort: sort lines of text files
- sed: filtering and transforming text

Lab2.log

- .log is the same as .txt – no difference
- Ex:
  - 1. I used wget to download the webpage
  - 2. I ….
  - 3. Answer to #3 here
- Should read basically like a lab journal
- Keep things concise!

Lab Hints

- Run your script on seasnet servers before submitting to CCLE
- sed '/patternstart/,/patternstop/d‘
  - delete patternstart to patternstop, works across multiple lines

    will delete all lines starting with patternstart to patternstop
- The Hawaiian words html page uses \r and \n for new lines
  - od –c hwnwdseng.htm to see the ASCII characters
- You can delete blank white spaces such as tab or space using
  - tr -d '[:blank:]’
  - Use tr -s to squeeze multiple new lines into one
- sed 's/<[^>]*>//g' a.html to remove all HTML tags

POSIX Bracket Expressions

Class	**Matching characters**	**Class**	**Matching characters**
[:alnum:]	Alphanumeric characters	**[:lower:]**	**Lowercase characters**
[:alpha:]	Alphabetic characters	[:print:]	Printable characters
[:blank:]	**Space and tab characters**	[:punct:]	Punctuation characters
[:cntrl:]	Control characters	[:space:]	Whitespace characters
[:digit:]	Numeric characters	**[:upper:]**	**Uppercase characters**
[:graph:]	Nonspace characters	[:xdigit:]	Hexadecimal digits

Buildwords

- Hawaiian.html -> buildwords -> hwords
- Buildwords
  - Read from STDIN and perform work on input
  - Output to STDOUT
- Ex: $ ./buildwords < hawaiian.html > hwords

Assignment 2 - Homework

Useful grep Options

- • -l and -L
  - Suppress normal output; instead print the name of each input file from which output would normally have been printed. The scanning will stop on the first match.
  - Uppercase L prints all complementary filenames
- -v
  - Invert sense of matching
- -o
  - Print only results that match completely

Questions?