EECS2031

SED A STREAM EDITOR

GREP

- Prints out all the lines in the input that **matches** an expression
- grep [options] pattern [file]
- Options let you do inverse search, ignore case, ……
- grep exits with 0 (found) 1 (not fund) 2 (file not found)
- Regular expressions used in grep, sed, vi, awk to match a pattern

GREP

- The difference between grep, egrep and fgrep.
- Options
  - -n precedes each line by line number
  - -i ignore case
  - -v invert search "show lines that do not match"
- See "`man grep`" for details

REGULAR EXPRESSION

- **“foobar” matches (only) foobar**
- **‘.’ Matches any single character**
  - f.obar matches faobar, fboar, ….
- **[xyz] matches any character in the set**
  - fo[abo]bar matches foabar, fobbar, foobar
- **[^xyz] matches any character that is not in the set**
  - fo[^ab]bar matches focbar, fodbar but not foabar

REGULAR EXPRESSION

- **‘*’ matches 0 or more occurrence of the last char**
  - fo* matches f,fo,foo,fooo,foooo
- **‘?’ matches 0 or 1 occurrence of the last char**
  - fo?bar matches fbar and fobar
- **‘+’ matches one or more occurrence of the last char Extended**
  - fo+bar matches fobar foobar, fooobar, …

REGULAR EXPRESSION

- **‘^’ matches the beginning of a string (line)**
- **‘$’ matches the end of a string (line)**
- **[a-z] matches any character in the range**
- **[0-9] matches any digit in the range**
  - ^[ABC] matches A,B, or C at the beginning of a string
  - ^[^ABC] matches any character at the beginning of a string except A, B, and C
  - ^[^a-z]$ matches any single character string except a lower case letter

REGULAR EXPRESSION

Not all versions of UNIX (NOT OURES)

- **“\<“ and “\>” matches the beginning and end of a word**
- **\{n\} matches n occurrences of the last char**
- **\{n,\} at least n occurrences**
- **\{n,m\} between n and m occurrences**
  - ^A\{4,8\}B matches any line starting with 4,5,6,7, or 8 A’s followed by a B
- **^(\+|-)?[0-9]+\.?[0-9]*$** **what is that? egrep**

REGULAR EXPRESSION

- **Examples**
- **%grep ‘\<north\>’**
- **%egrep ‘e+’**
- **%grep ‘[0-9]([0-9])\{3\}[0-9]\{3\}-[0-9][0-9][0-9][0-9] phone**

SED -- INTRODUCTION

- **Sed is a stream editor**
- **Line by line goes through the editor (filter) where every line may or may not change**
- **There is an interactive editor *ed* that accepts the same commands**
- **All editing commands (could be in a script file) are applied to each line in the file.**
- **The output is sent to the standard output (may be redirected to a file).**

HOW SED WORKS

- Every line of the input file is read into the "pattern space"
- Sed commands are applied to the line one by one.
- After all the commands are applied to the line, the line is sent to the output (some of these commands may result in discarding the line).
- Each command is on the form of address and action
- The address decides if the action will be applied to the line or not.
- If 2 commands are applied at he same line, the second command will be applied to the "possibly" modified line by the first command

SED COMMANDS

- The address can be either a line number or a pattern enclosed between two slashes */pattern/*
- If no pattern, the command is applied to every line
- if one address, the command applied to that line, if 2 addresses, the command applied to the range of addresses.
- take a look at man sed, here are few useful flags
- -n Suppress automatic printing of pattern space
- -e script to follow (multiple edits)
- -f script file

ADDRESSES -- EXAMPLE

- d Delete all the lines
- 2d Delete line 2
- 1,4d Delete lines 1 through 4
- /^$/d Delete all blank lines
- 7,/^$/d Delete lines 7 through the first blank line
- /^$/,$d Delete from the first blank line to the last line
- /a*b/,/[0-9]$/d Delete from the line that contains b, ab, aab, …. to the first line that ends with a digit

SED COMMANDS

a\	Append one or more line to the current line
c\	Change current line with new text
d	Delete line
h	Copy pattern space to holding buffer
H	Append content of pattern space to holding buffer
g	move holding buffer to pattern space (overwrite)
G	like g but append
p	print line
s	substitute
n,q,r,!	

DELETE COMMAND

- sed '3d' file delete the 3rd line
- sed '$d' file delete the last line
- sed '/north/d' file delete all lines that contains north

SUBSTITUTE COMMAND

- sed 's/west/north' file replace the first occurrence of west in every line to north
- sed 's/west/north/g' replace each occurrence of west by north in each line.
- sed –n 's/west/north/p' print only line that contains the word after replacing it by north
- sed –n 's/west/north/gp' print only line that contains the word after replacing it by north but replace every occurrence (g for globally)
- sed –n 's/\(Mar\)got/\1ianne/p' What is that?
- Can have multiple commands sed –e '---' –e '-----' file

READING AND WRITING

- sed '/James/r newfile' file Looks for lines that contains James and right after it, sed read and includes the contents of "newfile"
- sed –e '/James/p' –e '/james/r newfile' file
- sed –e '/james/w newfile' file it writes the lines that contain James into new file

CHANGING THE FILE

- Appending a line after a specific line
- sed –n '/north/ a\ <---Moved------->' file It will append the string "<---Moved--->" after each line that contains "north
- sed –n '/north/ a\

  > <---Moved------->' Another way to do it
- If you want north followed by white space /north[[:space:]] or north[\t]
- Use i\ instead of a\ to insert before the line
- sed '/western/c\

> changed' file change the line contains western to "changed"

OTHER COMMANDS

- sed '/east/{n; s/aa/bb/;} datafile the n commands matches the patter following it to the next line not the current one
- the y command is similar to Unix tr
- sed '1,3y/abcdef/ABCDEF/' datafile Capitalize letters a-f in the first three lines

WRITING TO A FILE

- %sed –n '/north/w newfile' file

HOLDING AND GETTING

- The line is stored in a temporary buffer called the *pattern space* for processing
- Unless deleted or suppressed, line is printed to output
- The pattern space is cleared
- The h command hold the line in the *holding buffer*
- The g(G) command gets the line in the patternspace

HOLDING AND GETTING

- %sed –e '/west/h' –e $G' file
- Put the line contains west in the holding space. Second commands prints it after the end of the last line
- %sed –e '/north/h' –e '/north/d' –e '$g'

AWK