Regular expressions

Part II

SED Command

- Stands for **s**tream **ed**itor
- Reads the specified file or standard input, modifying it as per the commands.
- All the input you feed into SED passes through and goes to STDOUT.
- It does not change the input file.

General Syntax

- /pattern/action
  - **pattern** is a regular expression,
  - **action** is the commands as given below:
    - **p**: Prints the line
    - **d**: Delete the line
    - **s/pattern1/pattern2/ [option]:** Substitutes the first occurrence of pattern1 with pattern2
    - **a** append
    - **c** change

Running sed

- sed OPTIONS... [SCRIPT] [INPUTFILE...]
- Options
  - -n to suppress output
  - -e expression
  - -f *script-file*
  - *-E extended regular expression*
- sed [-n] [-e] 'command(s)' files
  - sed -e '1d' -e '2d' -e '5d' books.txt -- deletes lines 1,2,5
- sed [-n] -f scriptfile files
  - echo -e "1d\n2d\n5d" > commands.txt
  - sed -f commands.txt books.txt

The sed Addresses

- Addresses are either particular locations in a file or a range where a particular editing command should be applied.
- When the sed encounters no addresses, it performs its operations on every line in the file.
- Addressing in sed is done in two ways:
  - **Line addressing:** By one or two line numbers (like 2,4)
  - **Context addressing:** By specifying a ‘/’ enclosed pattern which occurs in a line (like /to/)

- **$** represents last line of the file
- **'m,n[action]'** Lines starting from m to n (if m<n)
- **'m,n[action]'** only n^{th} line (if m>n) because the sed does not work in reverse direction
- **'m,+n[action]'** matches lines starting from m and continues to next n number of lines
- **'m,n![action]'** matches everythin except lines from m to n
- '**m~n[action]'** apply the action to m^{th} line and step over the next nth line. Sed continues to apply this pattern until the end of the file.
- **'m,[action]' or ',n[action]'** This generates the syntax error

Printing Operation

- Syntax:
  sed –n ‘address**p**’ filename
  sed –n ‘/RE/**p**’ filename
- Example:
  - sed –n ‘2,5p’ file print the lines from 2 to 5
  - sed –n ‘/ab/p’ file print the lines which contains the pattern ‘ab’
  - sed ‘/ab/,$p’ print the lines containing pattern ab to till the last line.

Deletion Operation

- Syntax:
  sed ‘address**d**’ filename
  sed ‘/RE/**d**’ filename
- Example
  - sed ‘2,5d’ file delete the lines from 2 to 5
  - sed ‘/ab/d’ file delete the lines which contains the pattern ‘ab’

The Substitution Command

Syntax:
sed 'ADDRESS**s**/RE/REPLACEMENT/FLAGS' file
sed '/PATTERN/**s**/RE/REPLACEMENT/FLAGS' file

- s is a substitute command
- RE is regular expression to match
- REPLACEMENT is a value to replace
- FLAGS can be any of the following
  - g Replaces all matches
  - n (any number) Replaces only n^{th} match
  - p If substitution was made, then prints the pattern space
  - w FILENAME If substitution was made, then writes result to FILENAME
  - i match RE in a case-insensitive manner

- Replace the occurrence of abc with xyz

  sed 's/abc/xyz/g‘

(By default sed replaces only the first occurrence of ab in each line. To replace all the occurrence use flag g.)

- Print the result to a new file

  sed –n ‘s/ab/xy/w newfile’ file

- Print only substituted lines

  sed –n ‘s/ab/xy/p’ file

- Replace all the occurrence of ab with xy from 5^{th} line onwards

  sed ‘5,$s/ab/xy/g’ file

Using an Alternative String Separator

- Suppose you have to do a substitution on a string that includes the forward slash character.
- Example replace all the occurrence of ‘/root’ with ‘/admin’
  - sed 's:/root:/admin:g‘ file

Replacing with Empty Space

- Use an empty substitution string to delete the particular word from the file.
- sed ‘s/ab//g’ file

Using back reference

Make all letters to appear within {}

sed ‘s/[a-zA-Z]/{&}/’ file

Ex. Abc is replaced as {a}{b}{c}

Make all the digits to appear within parenthesis.

sed ‘s/[0-9]\+/(&)/’ file

For example replace 99 to (99)

Add hi at the end of every line

sed ‘s/.*/&hi/’ file

Using Regular Expression

^ Matches the beginning of lines
$ Matches the end of lines
. Matches any single character
* Matches zero or more occurrences of the previous character
\+ Matches one or more occurrences of the previous character
\? Matches zero or one occurrence of the previous character
[chars] Matches any one of the characters given in chars, You can use the - character to indicate a range of characters. [^chars] matches any character except [chars]

- \{i\} matches exactly i sequences.
- \{i,j\} Matches between i and j, inclusive, sequences.
- \{i,\}Matches more than or equal to i sequences.
- \(regexp\)Groups the inner regexp as a whole
- regexp1\|regexp2 Matches either regexp1 or regexp2.
- Regexp1regexp2 matches the concatenation of regexp1 and regexp2

File write Operation in sed

Syntax:

sed ‘address**w** outputfile’ filename

sed ‘/RE/**w** outputfile’ filename

Append Operation in sed

Syntax:

sed 'ADDRESS **a**\Line which you want to append' filename

sed '/PATTERN/**a**\Line which you want to append' filename

Insert Operation in sed

Syntax:

sed 'ADDRESS **i**\Line which you want to insert' filename

sed '/PATTERN/**i**\Line which you want to insert' filename

Change/Replace Operation in sed

Syntax:

sed 'ADDRESS **c**\Line with which you want to replace' filename

sed '/PATTERN/**c**\Line with which you want to replace' filename

Print line numbers using sed

- Syntax:
- sed –n 'ADDRESS=' filename
- sed –n '/RE/=' filename
- sed –n ‘/RE1/,/RE2/=’ filename
- Examples
  - Print the line numbers which contains the pattern abc
    - sed -n '/abc/=' input.txt
  - Print the line numbers which begin with pattern abc to till the line contains the pattern xyz
    - sed –n ‘/abc/,/xyz/=’ input.txt