SED and AWK

Manipulation

SED

- sed is a special purpose editor
- takes commands from
  - a script
    or
  - command line

- it **cannot** be used interactively

- All input
  - comes from standard input
  - goes to standard output

SED Files

- changes are not made to the edit file
- the input file, along with any changes, is written to standard output
- so a file is manipulated and written to standard output

SED Processing

- sed goes through the file **a line at a time**
- if no specific address is specified it operates on all lines

SED Syntax

sed [-n] [-e] [‘command’] [file…]
sed [-n] [-f scriptfile] [file…]

- -n - only print lines specified with the ‘p’ command or the ‘p’ flag of the substitute (‘s’) command
- -e command - the next argument is an editing command rather than a filename
- -f scriptfile - a filename containing editing commands

SED Processing

- All editing commands in a script are applied in order to each line of input
- All editing lines of a script are applied to all lines of the edit file unless line addressing restricts the lines affected by the command
- The original file is unchanged
- Editing commands modify a **copy** of the original line which is sent to standard output

SED Commands

[address[, address]][!]command [arguments]

Addressing:

- a line number
- a pattern, in slashes (/pattern/)
  - described using regular expressions

If no pattern is specified, the command will be applied to all lines of the input file

SED Command Addresses

- Usually two addresses

If only one address is given, the command operates only on that line

- Two comma separated addresses specify a range that consists of lines between the first and second address, inclusively
- The ! operator can be used to negate an address, ie; address!command causes command to be applied to all lines that do not match address
- Braces { } can be used to apply multiple commands to an address

Multiple Commands

```
[/pattern/[,/pattern/]]{
        command1
        command2
        command3
        }
```

- { must be the l**ast character on a line**
- } must be **on a line by itself**
- there must be **no spaces following the braces**

Examples

d deletes the current line
6d deletes line 6
/^$/d deletes all blank lines
1,10d deletes lines 1 through 10
1,/^$/d deletes from line 1 through the first blank line
/^$/,/$/d deletes from the first blank line through the last line of the file
/^$/,10d deletes from the first blank line through line 10
/^Co*t/,/[0-9]$/d deletes from the first line that begins with Cot, Coot, Cooot, etc through the first line that ends with a digit

SED Commands

SED is very extensive.

Here are some basic commands:

s - substitute
a - append
i - insert
c - change
d - delete
h,H - put pattern space into hold space
g,G - Get hold space

Substitution

[address(es)]s/pattern/replacement/[flags]

pattern - search pattern
replacement - replacement string for pattern
flags - optionally any of the following

- n a number from 1 to 512 indicating which occurrence of pattern should be replaced
- g global replace
- p print contents of pattern space
- w file write the contents of pattern space to file

Examples

s/cat/dog/
Substitute dog for the first occurrence of cat
s/Tom/Dick/2
Substitutes Dick for the second occurrence of Tom

s/wood/plastic/p
Substitutes plastic for the first occurrence of wood and outputs (prints) pattern space
s/Mr/Dr/g
Substitutes Dr for every occurrence of Mr in pattern space

Append, Insert, and Change

Syntax for these commands is a little strange because they must be specified on multiple lines

```
append    [address]a\
          text
insert    [address]i\
          text
change    [address(es)]c\
          text
```

Append, Insert,

Append puts text after the current line

Insert puts text before the current line

These commands requires a **** following it to “escape” the NEWLINE

- text must begin on the next line.
- for multiple lines, use for ESCAPE all lines excpet last on

Let's examine an online resource:

http://www.ibm.com/developerworks/linux/library/l-sed2.html