Useful terminal-linux/bash commands

Some important bash commands, and very useful linux tools
Only basic usage of useful commands in this workshop: read the manual for details
Madhu, Oct 2018 (belur[AT]iitb.ac.in)

Line and word counts:

1. Word count: wc
2. wc -l file.txt (gives line count)
3. Note: ensure file format is unix.
   Use vim filename, and ":set ff=unix", and then ":x"
4. fileformat: "dos" and "neol" is a problem: gives one wrong
   (also dos2unix converts)
5. grep -c "pattern" filename : gives number of LINES having pattern
   (some lines could have that pattern multiple times: so be careful)

grep

1. grep pattern filename.txt (only ascii/plain text files, not binary)
2. grep -c pattern filename.txt (only count)
3. grep pattern file | wc -l : gives number of lines having pattern
4. grep -i pattern filename.txt (case INsensitive) (default is : case SENSITIVE)
5. grep -i -e pattern1 -e pattern2 filename.txt (“or” operation)
6. grep -e pattern1 filename.txt | grep pattern2 (“and” operation)
   lines containing BOTH pattern1 and pattern2 (in any order)
7. grep -n pattern1 filename.txt : prints line-number too
8. grep -v pattern1 filename.txt
   prints those lines withOUT pattern1 (v for “inVert”)
9. grep '\.' file (special characters need ‘escaping by \)
10. Special characters: ' " .\ #

pipe |

1. output of one command goes to next command instead of screen (STDOUT)
2. grep pattern file | wc -l
   Works like: wc -l output-of-grep-pattern-file (but without making new files)
3. helpful for finding repetition across files (using sort, uniq, wc -l)
4. makes linux/bash terminal special and powerful
5. practise this well

sed one liners

1. sed ”s/old-pattern/new-pattern/g” file (g means all occurrence on any line with old-pattern)
2. without g: only first occurrence on such lines
3. Can use any or all of ‘cgi’ (c: ask yes/no, i: case-Insensitive, g: all on such lines)
4. sed syntax above gives output on STDOUT (to be directed to a file (using ‘>’), or piped | to something further)
5. sed -i ”s/old-pattern/new-pattern/g” file (i for in-place. Overwrites existing file)
6. sed -f file-with-sed-commands-without-any-quotes file-to-be-operated-on
7. See “sed one liners” on the net (on sourceforge, primarily by Eric Pement)
8. If special characters, or if ” in pattern, then escape them and use:
   sed ’s/old-pattern/new-pattern’ filename (Use ’ instead of ”)

cut/paste/counts

1. cut -d”,” -f1,3,6-10,14- file.csv
2. paste –delimiter=”,” file1 file2
3. uniq file : gives a ‘uniqed’ file: only *consecutive* repetitions ‘uniqed’
4. sort file — uniq
5. sort file — uniq -d # only duplicates
6. sort file — uniq -c # gives count of each line, can extract duplicates, triplicates
7. cut -d”,” -f1,2 file1 — cat - new-file — sort — cut -d”,” -f1 — uniq -c — grep ’ 1 ’

Shell variables

1. grep pattern file
2. Suppose you need this output for something else (say var=value)
3. Then var=$(command)

Exercise

1. grep pattern file
2. Suppose you need this output for something else (say var=value)
3. Then var=$(command)

View http://www.ee.iitb.ac.in/%7Ebelur/foss/bash/
Download the questions and the sample.xlsx file and answer questions.