Unit II File Management commands Filters (Part I)

Sorting Files

- To sort the contents of file

  $ sort filename

  - to sort multiple file together

    $ sort f1 f2 f3

  - to save output to a file named “final”

    $ sort –o final filename

  - to sort only unrepeated data

    $ sort –u filename

Sorting Files

- to sort data from terminal

```
$ sort
```

- to combine the contents of terminal & a file & then perform sorting

```
$ sort – filename
```

- to merge only sorted files

```
$ sort –m f1 f2
```

- to sort the file in reverse/descending order

```
$ sort –r f1
```

Information on Files

- To generate one or more of the following statistics: newline count, word count, and character count

wc filename

A line is any group of characters not containing a newline

A word is a group of characters not containing a space, tab or newline

A character is the smallest unit of information, and includes a space, tab, and newline

Information on Files

- to count only lines

  $ wc –l filename

- to count only words

  $ wc –w filename

- to count only characters

  $ wc –c filename

- to display the length of longest line in a file

  $ wc –L filename

Uniq command

It discards all the successive identical lines except one from the input and writes the output.

- uniq filename
- uniq –c filename -- with count

The options of uniq command are:

c : Count of occurrence of each line.
d : Prints only duplicate lines.
D : Print all duplicate lines
f : Avoid comparing first N fields.
i : Ignore case when comparing.
s : Avoid comparing first N characters.
u : Prints only unique lines.
w : Compare no more than N characters in lines

What is common?

- To find the common lines in two “SORTED” files

  $ comm f1 f2

  - it displays a three-columnar output, first column contains lines unique to f1, second column contains lines unique to f2, while third column contains lines common to both f1 and f2

  - to drop any column, use column number as option

    $ comm -2 f1 f2

    $ comm -12 f1 f2

File Comparison: cmp command

- Used to compare the two files byte by byte
- Helps you to find out whether the two files are identical or not.
- Reports the location of the first mismatch.
- Displays no message and simply returns the prompt if the files compared are identical.

Syntax

cmp [OPTION]... file1 [file2 [skip1 [skip2]]]

File1 and file2 are file names

skip1 and skip2 specify the number of bytes to skip at the beginning of each file which is zero by default

cmp f1 f2

the files are compared byte by byte, if found a mismatch, location is displayed, but if files are identical, no message is displayed & the prompt is returned

cmp -i 5:6 file1.txt file2.txt

Will skip 5 bytes from first file and 6 bytes from second file

cmp -n 30 file1.txt file2.txt

Will search 30 bytes from the starting

diff command

- diff file1 file2
- The first line of the **diff** output will contain:
  - line numbers corresponding to the first file,
  - a letter (**a** for *add*, **c** for *change*, or **d** for *delete*), and
  - line numbers corresponding to the second file.
- Lines preceded by a **<** are lines from the first file;
- lines preceded by **>** are lines from the second file.

pg command

- Displays the contents of text files, one page at a time
- Syntax: pg [-number] [-p string] [-cefnrs] [+line] [+/pattern/] [file...]

Head command

- Print the first N number of data/lines of the given input.
- By default, it prints the first 10 lines of the specified files.
- If more than one file name is provided then data from each file is preceded by its file name.
- **Syntax:**
  - head [OPTION]... [FILE]...

Viewing Files - head

to view first 5 lines from file.txt

$**head -n 5 file.txt**

to view first 10 bytes from the beginning of file

$ head –c 10 filename

*also counts newline character

Try head without any arguments !!!

-Waits for user input

Viewing Files - head

- to view name of the file along with the contents

    $ head –v filename

- to view first few lines of more than 1 file

    $ head f1 f2 f3

- to suppress filenames when multiple files are being used

    $ head –q f1 f2 f3

Tail command

- Print the last N number of data of the given input.
- By default it prints the last 10 lines of the specified files.
- If more than one file name is provided then data from each file is precedes by its file name
- **Syntax:**
  - tail [OPTION]... [FILE]...

Viewing Files - Tail

- To display last few lines of the file

  $ tail filename {by default shows 10 lines}

- to view last n lines

  $ tail –n filename

- to view desired “n” no. of bytes from the ending of file

  $ tail –c n filename {also counts newline character & n is mandatory}

- Try tail without any arguments !!!

Viewing Files - Tail

- to view name of the file along with the contents

    $ tail –v filename

- to view first few lines of more than 1 file

    $ tail f1 f2 f3

- to suppress filenames when multiple files are being used

    $ tail –q f1 f2 f3

Viewing Files-more

- To view files having content not fitting in the screen at a time

  $ more filename

- to view “n” no. of lines

  $ more –n filename

- to prompt the user with the message "[Press space to continue, 'q' to quit.]" and will display "[Press 'h' for instructions.]" when an illegal key is pressed

  $ more –d filename

- to squeeze multiple blank lines into one

  $ more –s filename

Viewing Files

- press “space bar” or “f” to move one page forward
- Press [Enter] to move one line forward
- Press “2f” to move 2 pages forward
- Press “b” to move one page backwards
- Press “3b” to move 3 pages backward
- Press “q” to quit
- Press “h” to get help
- !cmd to execute UNIX command cmd

less command

- *Less* is a command similar to *more*, but which allows backward movement in the file as well as forward movement.
- *less* does not have to read the entire input file before starting, so with large input files it starts up faster than text editors like *vi*

cut command

- To select a list of columns or fields from one or more files.
- Specify either the -c option to cut by column or -f to cut by fields.
- Fields are separated by tabs unless you specify a different field separator with –d
- **Based on column (-c)**
  - cut –c 4 f1 only 4th column
  - cut –c -3 f1 beginning of line to 3rd column
  - cut –c 2-3,6-8 f1 2nd to 3rd & 6th to 8th column
  - cut –c 6-7,9- f1 6th to 7th & 9th to end of the line

Based on fileds (-f)

- cut –f 1,4 f1 (default field separator is tab)
- cut –d “:” –f 2,5-8 f1 give respective fields, for all the lines containing “:” or not
- cut –s –d “:” –f 2,4 f1 give respective fields, only for the lines containing delimiter
- cut - -complement –f1 filename gives all fields except first field
- cut –d “:” –f1,4 - -output-delimiter=$”*“ filename changes the output delimiter to * rather than “:” by default

Paste command

- Merges the lines from multiple files
- paste [options] files-list
  - Options
    - -d : Specify of a list of delimiters.
    - -s : Paste one file at a time instead of in parallel.
- paste file1 file2
- paste –d”|” file1 file2
- paste –s file1 file2
  - The paste command reads each file in sequentially. It reads all the lines from a single file and merges all these lines into a single line.
- Specifying multiple delimiters.
  - paste -d"|," file1 file2 file3

Pipe

- Shells' sequence of interpretation of a command
- Used to connect two commands together so that the output from one program becomes the input of the next program.
- To make a pipe, put a vertical bar (|) on the command line between two commands.

```
echo hi | wc –c
echo “hi” | echo –e “\n welcome” | wc –l
ls –l |sort –r
ls -1 a* | wc –l
echo –n “welcome” ; ls a* | wc -l
```

- Assignment 3.1