Unit 4
Filters and Regular Expressions

Dr.S.Thenmozhi

Filters and Pipes

- Filter is any command that gets its input from the standard input stream, manipulates the input, and then sends the result to the output stream
- Filters can be used in the left of the pipe, right of the pipe or between two pipes.
- Because filter can send output to the monitor, it can be used in the left of the pipe.
- Because filter can receive input from the keyboard, it can be used in the right of the pipe.

Concatenating Files

- The cat command writes the file contents to the standard output
- It can take multiple files as input
- When multiple files are given, it takes one after the another
- The result becomes as one output and it can be saved in one file also
- The cat command is to catenate multiple file. But when given with one file, it catenates with null file as second.
- It does not give automatic pause after the end of the screen
- It does not check for the filetype before catenating. It just catenates

- Cat is used to create a file. There is only one input which comes from the keyboard.
- As we want to save the contents to the file, we redirect to the file
- End of file is identified with ^d
- Four categories: visual characters, buffered output, missing files, numbered lines
- -v allows us to see control characters with the exception of tab, newline, form feed characters
- -vt – the tabs appear as ^I
- To supress blank lines -s
- Numbered lines - -n

Filtering Beginning of File - head

- Displaying the beginning of a file to the std. output.
- If no files is specified, it receives from the std. input
- It can work with multiple files also
- When used without any options displays first ten lines of a file
- -n to specify the no. of lines to be displayed [counts from beginning]
- head –n 3 stud.dat or head -3 stud.dat
- What does the following command will do?
  - gedit `ls –t | head -1`

[opens up the last edited file]

Filtering End of File - tail

- Displaying the end of a file
- When used without any options displays first ten lines of a file
- -n to specify the no. of lines to be displayed [counts from end]
  - tail –n 3 stud.dat or tail -3 stud.dat
- +n counts from the beginning
  - tail -n +11 stud.dat [displays from line number 11, skips first 10 lines]
- -c extract bytes rather than lines

Filter Columns- cut

- Split a file vertically or column wise
- cut -c   => To cut/extract specific columns
- Ranges can also be specified
- Mutiple columns separated by comma
- cut –c -3,6-22,28-34,55- stud.dat
- -f => to cut fields -d =>specify delimiter
- cut –d "|" -f 2,3 stud.dat or cut –d \| -f 2,3 stud.dat
- What does this command will do?
  - who | cut –d " " –f 1

  [cuts the first field from the result of who command]

Combines Columns- paste

- To merge/paste contents/combines lines together
- It is done vertically
- paste stud1.dat stud2.dat
- pastes both files vertically
- -d – to paste with delimiter
- paste -d “\t |” stud1.dat stud2.dat
- To specify the input coming from std. input put – instead of a filename

Simple Filters - sort

- Orders a file
- It identifies fields and sort on specified fields
- Sort reorders lines in ASCII collating sequence –numerals, uppercase letters, finally lowercase letters
- sort stud.dat
- The sorting sequence can be altered by appropriate options
- Sorting can be done one more than one fields

Simple Filters - sort

- Options
- -t *char* – uses delimiter char to identify fields
- -k n – sorts on the nth field
- -k m,n – sorts on the mth field and then on nth field
- -k m.n – starts on the nth col of mth field
- -u – removes repeated lines
- -n – sorts numerically
- -r – reverse sort order
- -f – case insensitive sort
- -c – checks if file is sorted
- -o *filename* – places output in a file *filename*
- -m *list*- merges sorted files in list

- sort –t "|" –k 2 stud.dat   // sorting on primary key
- sort –t "|" –k 3,3 –k 2,2 stud.dat // sorting on secondary key – start and end should be specified
- sort –t "|" –k 5.7,5.8 stud.dat // specifies the column position of the field specifies 7th ,8th column of the 5th field

Simple Filters - uniq

- Locate repeated and non-repeated lines
- Special tool instead of sort command with –u option
- uniq stud.dat //fetches unique lines in the file to std. Output
- Uniq requires the input as sorted input
- So, uniq can be piped after sort command
- sort stud.dat | uniq
- sort stud.dat | uniq – stud1.dat // two filename one the source and other the destination i.e output will be written into stud1.dat

- uniq Options
- -u – selecting non- repeated lines
- -d – selects the duplicated lines
- -c – counts the frequency of occurrence of all lines

Translating Characters - tr

- Translating characters
- tr filter manipulates on individual characters in a line
- Format: tr options exp1 exp2 std input
- tr takes input only from the std input.
- It does not take filename as argument
- By default it translates each character in expression1 to its mapped counterpart in expression2.
- The first character in first is replaced with the first in the second and so on
- The length of the two expressions should be equal

Translating Characters - tr

- Changing case of text
  - head –n 3 stud.dat | tr A-Z a-z
- Deleting characters -d
  - tr –d ‘|’ < stud.dat | head –n 3   ?????
- Compressing multiple consecutive characters –s (squeezes)
  - tr –s “ “ < stud.dat |head –n 3
- Complementing the action –c
  - Tr –c –d [:digits:] <stud.dat |head –n 3

1. How to display date output with each field in a separate line?
2. Join the splitted lines once again to the same format.
3. Extract the names of the users from /etc/passwd after ignoring the first 10 entries.
4. Sort the file /etc/passwd on GID and UID so that the users with the same GID are placed together. Users with a lower UID should be placed higher in the list.
5. Devise a pipeline sequence which lists the five largest files in the current directory.
6. Select 5 to 10 lines of a file

Answers

1. date | cut –d “ “ –f 1,2,3,4,5,6 –output-delimiter=$’\n’ or date|tr [:space:] ‘\n’
2. paste –d “ “ -s date.lst
3. tail +11 /etc/passwd
4. sort –t “:” –k 3,3 -k4,4 /etc/passwd
5. ls –lS | head -5   [s – based on file size in blocks

   S – based on file size in bytes]
6. tail +5 /etc/passwd |head -5