Welcome
Linux 2022

www.adafrolabs.com

Command line filters form an advanced group of commands, which are used to quickly obtain specific information from or about files. Filters can be also used to process output from other commands (including other filters) through chaining multiple commands together.

Most commonly used filters include:

more and less

These commands are familiar to you already - they are used to process text, allowing the user to view it one page at a time.

head and tail

These commands provide respectively the beginning or the end of the file. By default 10 lines are shown, but any number of lines can be requested. For example, the following command will display the last 18 lines from file contacts. txt:

tail -18 contacts.txt

This command will sort input. By default sorting is done in the a -z order (ascending), but the -r option can be used for reverse order. Numerical values can be sorted using the **-n** option

uniq

This command displays unique lines. Input needs to be sorted, as only adjacent lines are compared. Duplicated lines can be displayed using the -d option and matches can be counted instead of being displayed using the -c option, for example:

uniq -dc somefile - will count duplicated lines in somefile

.

wc

The wc (word count) is used to count input lines, words and characters. By default all 3 stats are provided, but options such as 4, -w, and -c can narrow the results.

cut

This feature rich command allows you to select columns from a file. You can choose specific characters using the -b option, or cut at the delimiter-based fields using the -d option.

Examples:

cut -b 2-5 somefile - will display character(byte) 2,3,4 and 5 in somefile

cut -b 1-6, 8-12 somefile - will display characters 1 through 6 and 8 through 12 in somefile

cut -d, f2 somefile - will display the second column (field) in somefile

cut -d” ” -f3 somefile - will display third field in a space-delimited somefile

grep

This is the #1 matching/searching toot in the *nix world. While we will only limit ourselves to literal matches, grep supports a powerful pattern matching mechanism called regular . expressions, allowing command line users to perform complex ad-hoc searches. Grep by default displays input lines with matches in them and is case-sensitive. Common grep options include: -i (ignore case), -c (count matches instead of displaying them) and -v (reVerse match, providing non-matching tines). Of course, options can be combined, for example:

grep -cv "Alice" somefile - will count lines in somefile which do not have "Alice" in them

Using a text editor such as a nano, create a file called friends with the following information inside(friends names and their birth dates). You may copy and paste from the on-line version of this lab:

Trevor,04,03,1996
Collin, 07,17,1996
Kermit, 06,02,1991
Alex,11,23,1995
Trish,03,27,2000
Unity,05,27,1999
Benson,06,15,1998
Nadia,02,27,1993
Gloria,01,28,1991

Using the information above, Internet resources and help from a classmate, provide Linux commands solving the following tasks on the above file. The file contains are there to help you out, but your solutions should be applicable even if the records were different.

- Count the number of friends:

  Command: _____

- Display records for friends born in 1991:

  Command: _____

- Display the name and year of birth for everyone:

  Command: _____

- Display the last 3 records:

  Command: _____

- Show all records in alphabetical order:

  Command: _____