Common Sed Commands

Sunday, June 11, 2017 8:46 PM

tr -c 'A-Za-z' '[\n*]' < assign2.html : After running this, all non
all non letter characters were replaced with a new line. So, if there
are multiple non letter characters in a row then there will be
multiple new lines.

tr -cs 'A-Za-z' '[\n*]' < assign2.html : After running this command, it
outputs a similar to the previous command except that if there are multiple
non letter characters in a row, they are replaced with only one new line.
There is no blank lines between lines with text.

tr -cs 'A-Za-z' '[\n*]' < assign2.html | sort : The output of this command has
the same characteristics of the previous command, but now
the output is sorted alphabetically. I initially
put < assign2.html at the end, but the terminal somewhat
froze and I had to use C-c to exit. Putting it before piping
to sort fixed the issue.

tr -cs 'A-Za-z' '[\n*]' < assign2.html | sort -u : This
outputted the same result as the previous command,
but using the -u option removed any duplicate words
that are spelled using the same characters. If, there
are different characters used to spell the same word
they are kept. For example, for all, All, and ALL,
none of the words are deleted, but if it was all,
all, and all, only one of the all words is kept.

tr -cs 'A-Za-z' '[\n*]' < assign2.html | sort -u | comm - words : This command outputs 3 columns.
The first column outputs the words that only appear in assign2.html. The
second outputs the words that only appear in words, and the third column
outputs the words that are in both files.

tr -cs 'A-Za-z' '[\n*]' < assign2.html | sort -u | comm -23 - words : This command outputs only the words that are unique to assign2.html. the -23 in the comm command suppresses the 2nd and 3rd columns, so they are not shown.

grep '<td>.*</td>' $1 | : This command searches for all the lines containing the English and Hawaiian words. These are between <td> and </td>. Initially, I did not put the . before the * and only lines that were <td></td> were outputted. Adding the * fixed it. The $1 is the file passed in to be changed.

sed 's/<[^>]>//g' | : This command deletes all the HTML tags from the text.

sed "s/\`/\'/g" | : This command replaces the okina with a comma. When trying to come up with the command, the " were initially ', but this caused the terminal to no longer let me enter commands. Also, I initially did not put \`. It did not have the \ before the `, causing the terminal to again freeze up.

sed 's/^ *//g' | : This command deletes all the leading spaces for each line. Initially, I used sed 's/^\t*//g' to delete the spaces since I thought it was a tab rather than just a space. Also, I thought if I added [] around the \t it would work, but it did not either. Thus, I just had the command look for a space at the beginning of the line and delete it.

sed 's/[,[:space:]]/\n/g' | : This command finds all the commas and spaces and changes them to new lines. This allows the words separated by a comma or space to be treated as multiple words. Initially, I used [,\s], but it would replace some spaces with newlines and it would not for others. Making it [,[:space:]] allowed the command to work.

sed '/^$/d' : This command deletes all the empty lines.

sed '/^\s*$/d' | : This command gets rid of all the empty lines.
Initially, I used the command, sed 's/^\s*$//g' and nothing happened. It seemed to replace the empty line with another empty line. By using the delete function with d, the empty lines were replaced.

tr '[:upper:]' '[:lower:]' | : This changes the uppercase letters into lowercase letters

sed "/[^pk'mnwlhaeiou]/d" | : This deletes any line that does not contain the Hawaiian letters. The ^ inside the brackets makes it so the sed command deletes the characters that are not in the set listed.

sort -u : This sorts the word list alphabetically and removes duplicates. Then, it saves the output into hwords.