sort /usr/share/dict/words > words
diff words /usr/share/dict/words

wget http://web.cs.ucla.edu/classes/winter17/cs35L/assign/assign2.html

1. tr -c 'A-Za-z''[\n*]' < assign2.html > assign2tr1.html
diff assign2.html assign2tr.html
tr -c 'A-Za-z' '[\n*]' replaces any character which is
not a member of the alphabet with a single newline

2. tr -cs 'A-Za-z''[\n*]' < assign2.html > assign2tr2.html
diff assign2.html assign2tr2.html
will replace any repeated
sequence of characters not a member of the alphabet
with a single newline

3. tr -cs 'A-Za-z' '[\n*]' < assign2.html | sort > assign2tr3.html
will sort the lines
of the output of tr -cs 'A-za-z' '[n*]' into
alphabetical order

4. tr -cs 'A-Za-z' '[\n*]' < assign2.html | sort -u > assign2tr4.html
will remove any
repeated lines that occured in 3

5. tr -cs 'A-Za-z' '[\n*]' | sort -u | comm - words
Will compare the output of 5. with the sorted file "words". It prints out three columns, the first column contains lines in the html text file that are in the html text file and not in words.
The second column contains lines that are in words but not in the html text file.
The third column contains lines that are common to both files. Also, all columns are listed in alphabetical order with respect to each other. That is, if the first entry of column 3 beginswith a "b" and the first entry of column 2 begins with a "c" then the first entry of column 2 will appear below the first entry of column 3

6. tr -cs 'A-Za-z' '[\n*]' | sort -u | comm -23 - words
Will suppress Columns 2 and 3 from the output of 5. So this only
lists the lines that are in the html text file, but not in the words file

7. wget http://mauimapp.com/moolelo/hwnwdseng.htm

8. grep -E '<td>.*<\/td>' hwnwdseng.htm > out1.htm

This command will remove everything except for <td>word</td> and some instances of <td></td>

sed '/<td><\/td>/d' out1.htm > out2.htm
This command will leave only <td>word</td> and remove <td></td>

sed 's/<td>\(.*\)<\/td>/\1/g' out2.htm > out3.htm
This will remove the <td> and </td>, leaving only the text in the middle

sed -n 2~2p out3.htm > out4.htm.
This will extract only the Hawaiian words. Print alternate lines starting from second

sed 's/<u>\(.\)<\/u>/\1/g' out4.htm > out5.htm
treat "<u>a</u>" as if it were "a",

tr , '\n' < out5.htm > out6.htm
"H<u>a</u>lau, kula", contain spaces or commas; treat them as multiple words (in this case, as "halau" and "kula"). Example in line 70

sed 's/^[\t]*//' out6.htm > out7.htm
Eliminate blank space at beginning of line

tr ' ' '\n' < out7.htm | sed '/^$/d' > out8.htm
^$ is begining and end of line

sed "s/\`/'/g" out8.htm > out9.htm
treat ` (ASCII grave accent) as if it were ' (ASCII apostrophe, which we use to represent ‘okina)

tr A-Z a-z < out9.htm >out10.htm
Treat upper case letters as if they were lower case

sed "/[^pkmnwlhaeiou']/d" out10.htm > out11.htm
You may find that some of the entries are improperly formatted and contain English rather than Hawaiian; to fix this problem reject any entries that contain non-Hawaiian letters after the abovementioned substitutions are performed

sort -u out11.htm > hwords

grep -E '<td>.*<\/td>' | sed '/<td><\/td>/d' | sed 's/<td>\(.*\)<\/td>/\1/g' | sed -n 2~2p | sed 's/<u>\(.\)<\/u>/\1/g' | tr , '\n' | sed -e 's/^[\t]*//' | tr ' ' '\n' | sed '/^$/d' | sed "s/\`/'/g" | tr A-Z a-z | sed "/[^pkmnwlhaeiou']/d" | sort -u

Hawaiin Spell checker

tr '[:upper:]' '[:lower:]' < infile | tr -cs "pkmnwlhae'iou"
"[\n*]" | sort -u | comm -23 - hwords
Convert upper to lower. Ignore english words. Sort and also Remove duplicate new lines. Compare.

English spell checker.
tr '[:upper:]' '[:lower:]' < infile | tr -cs 'A-Za-z' '[\n*]' | sort -u | comm -23 - hwords

Check your work by running your Hawaiian spelling checker on this web page (which you should also fetch with Wget), and on the Hawaiian dictionary hwords itself.

1. tr '[:upper:]' '[:lower:]' < hwords | tr -cs "pkmnwlhae'iou" "[\n*]" | sort -u | comm -23 - hwords

Should give nothing.