Command Line Preprocessing and Exploration of Shakespeare's Plays

Get the Data

Make a new directory as the working directory and download data there:

```
mkdir shake
cd shake
wget
https://raw.githubusercontent.com/tommyhegarty/tommyhegarty.github.io/mast
er/Shakespeare_data.csv
```

Initial Data Review

The top 10 text lines in the data:

```
cat Shakespeare_data.csv | head -10
```

1. Count the # lines, # plays, and # lines per play

Count the total number of lines:

```
cat Shakespeare_data.csv | wc -l
```

The 2nd column is `Play`. Sort, remove duplicates, and count the number of plays:

```
cat Shakespeare_data.csv | awk -F, '{print $2}' | head -5
cat Shakespeare_data.csv | awk -F, '{print $2}' | sort | uniq -c
```

Count the number of lines per play:

```
cat Shakespeare_data.csv | awk -F, '{print $2}' | sort | uniq -c | sort -
nr
```

2. Identify unique players and count them in each play

1. Select `Play` and `Player` columns;
2. Sort and aggregate for unique Play-Player combinations;

3. Select `Play` column from the aggregated results;
4. Sort and aggregate again to get the number of `Players` per Play.

```
cat Shakespeare_data.csv | awk -F, '{print $2","$5}' | sort | uniq | head
-5
cat Shakespeare_data.csv | awk -F, '{print $2","$5}' | sort | uniq | awk -
F, '{print $1}' | head -5
cat Shakespeare_data.csv | awk -F, '{print $2","$5}' | sort | uniq | awk -
F, '{print $1}' | sort | uniq -c
```

3. Search for play containing certain keywords, e.g. `Brutus`

Search for `Brutus` and count the number of lines containing it:

```
cat Shakespeare_data.csv | grep 'Brutus' | head -5
cat Shakespeare_data.csv | grep 'Brutus' | wc -l
```

Try this too: `cat Shakespeare_data.csv | grep 'Brutus' | less`.

Interested in the 2nd column `Play`:

```
cat Shakespeare_data.csv | grep 'Brutus' | awk -F, '{print $2}' | head -5
```

Sort and output the unique plays containing the term:

```
cat Shakespeare_data.csv | grep 'Brutus' | awk -F, '{print $2}' | sort |
uniq
```

4. Vocabulary and Term Statistics

1. Remove quotes and case-fold;
2. Perl to substitute each `punctuation` with a new line character `\n`;
3. Sort the terms;
4. Aggregate term statistics accordingly;
5. Sort them again in the desc order to find the top-frequent terms.

```
cat Shakespeare_data.csv | awk -F, '{print $6}' | perl -pe
's/\"//g;s/[[:punct:]\s]+/\n/g' | less
cat Shakespeare_data.csv | awk -F, '{print $6}' | perl -pe
's/\"//g;s/[[:punct:]\s]+/\n/g' | sort | uniq -c
cat Shakespeare_data.csv | awk -F, '{print $6}' | perl -pe
's/\"//g;s/[[:punct:]\s]+/\n/g' | sort | uniq -c | sort -nr | head -5
cat Shakespeare_data.csv | awk -F, '{print $6}' | perl -pe
```

```
's/\"//g;s/[[:punct:]\s]+/\n/g;' | perl -pe '$_=lc' | sort | uniq -c |
sort -n
```