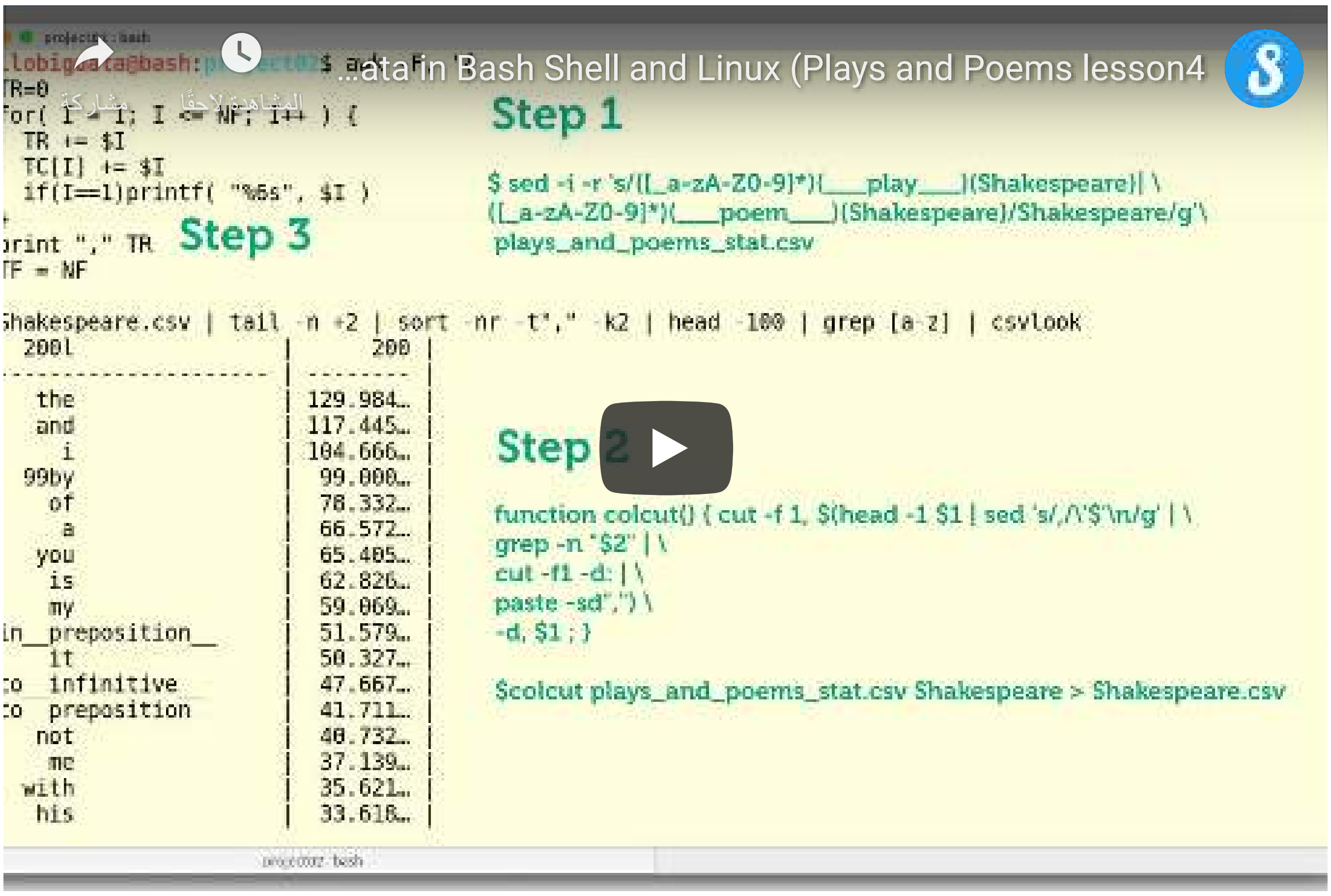
Finding the most frequent words by Shakespeare (Bash functions, sed, awk)

Given a text, what are the most frequent words?

Finding the most frequent words for a given text (e.g., Knight_of_the_Burning_Pestle) is easy, we can build a function `toptokens()`, which is nothing but the `topcrimes()` function developed in our previous project. Let's watch the following video lecture first:



```
projects : bash
llobig ca@bash:p ect02$ awk
TR=0
for( I = 1; I <= NF; I++ ) {
  TR += $I
  TC[I] += $I
  if(I==1)printf( "%6s", $I )
}
print "," TR
TF = NF
```

```
$ sed -i -r 's/([_a-zA-Z0-9]*)(___play___)(Shakespeare)| \
([_a-zA-Z0-9]*)(___poem___)(Shakespeare)/Shakespeare/g'\
 plays_and_poems_stat.csv
```

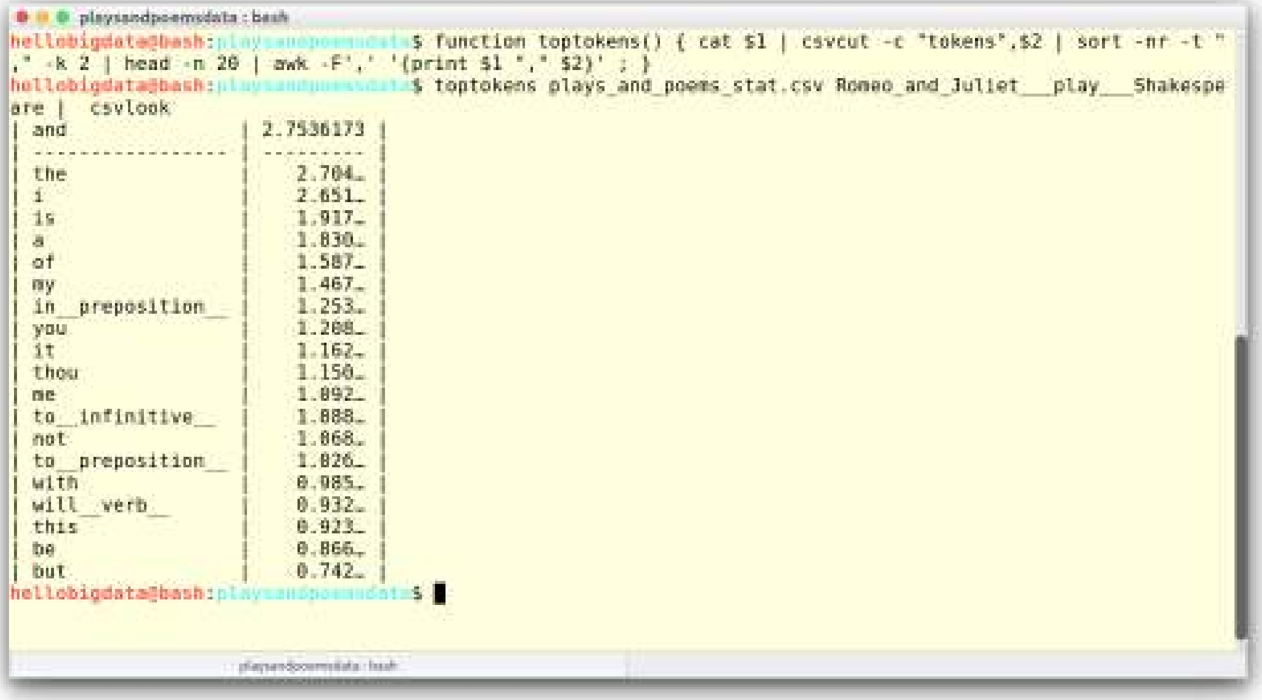
```
Shakespeare.csv | tail -n +2 | sort -nr -t"," -k2 | head -100 | grep [a-z] | csvlook
  2001                     |      200 |
-------------------------- | -------- |
   the                     | 129.984… |
   and                     | 117.445… |
     i                     | 104.666… |
  99by                     |  99.000… |
    of                     |  78.332… |
     a                     |  66.572… |
   you                     |  65.405… |
    is                     |  62.826… |
    my                     |  59.069… |
in__preposition__          |  51.579… |
    it                     |  50.327… |
to  infinitive             |  47.667… |
to  preposition            |  41.711… |
   not                     |  40.732… |
    me                     |  37.139… |
  with                     |  35.621… |
   his                     |  33.618… |
```

```
function colcut() { cut -f 1, $(head -1 $1 | sed 's/,/\'$'\n/g' | \
grep -n "$2" | \
cut -f1 -d: | \
paste -sd",") \
-d, $1 ; }

$colcut plays_and_poems_stat.csv Shakespeare > Shakespeare.csv
```

Video lecture: Finding the most frequent words by Shakespeare (complex)

For example, if we want to grab the most frequent words in the Romeo and Juliet play, we can execute the following:

```
function toptokens() { cat $1 | \
csvcut -c "tokens" $2 | \
csvcut -c "tokens",$2 | \
sort -nr -t "," -k 2 | \
head -n 20 | \
awk -F',' '{print $1 "," $2}' ; }

toptokens plays_and_poems_stat.csv "Romeo_and_Juliet___play___Shakespeare" |  csvlook
```



```
hellobigdata@bash:playsandpoemsdata$ function toptokens() { cat $1 | csvcut -c "tokens",$2 | sort -nr -t "
," -k 2 | head -n 20 | awk -F',' '{print $1 "," $2}' ; }
hellobigdata@bash:playsandpoemsdata$ toptokens plays_and_poems_stat.csv Romeo_and_Juliet___play___Shakespe
are |  csvlook
| and                | 2.7536173 |
| ------------------ | --------- |
| the                |    2.704… |
| i                  |    2.651… |
| is                 |    1.917… |
| a                  |    1.830… |
| of                 |    1.587… |
| my                 |    1.467… |
| in__preposition__  |    1.253… |
| you                |    1.208… |
| it                 |    1.162… |
| thou               |    1.150… |
| me                 |    1.092… |
| to__infinitive__   |    1.088… |
| not                |    1.068… |
| to__preposition__  |    1.026… |
| with               |    0.985… |
| will__verb__       |    0.932… |
| this               |    0.923… |
| be                 |    0.866… |
| but                |    0.742… |
hellobigdata@bash:playsandpoemsdata$ 
```

The top 20 frequent words in the work "Romeo and Juliet"

Given an author, what are the most frequent words?

This is slightly complicated! becuase we again need to perform several steps:

- For the given author, trim out the plays/ poems names, indclding text types (i.e., `plays | poems`)
- Combine all the columns, i.e., sum horizontally the frequencies of words for all the texts of that author
- Sort the words, based on the accumulated frequencies on all works by that author.

Don’t be scared! we will take you there.

Step 1. Trim out the plays/ poems names, for a given author:

Let’s consider that the author in question is Shakespeare. The following `awk` based regular expression will trim out all the bit before the name of the author. If you look closely, you will see that inside the `sed` regex, it’s actually finding the pattern of plays `OR` (`|`) poems names that end with the string “Shakespeare” and then replacing inplace (due to the `-i`) the whole

"Shakespeare" and then replacing inplace (due to the `-i -r`) the whole

matched pattern e.g., `Romeo_and_Juliet___play___Shakespeare` with the string `Shakespeare`:

```
$ sed -i -r 's/([_a-zA-Z0-9]*)(___play___)(Shakespeare)| \
([_a-zA-Z0-9]*)(___poem___)(Shakespeare)/Shakespeare/g' \
plays_and_poems_stat.csv
```

At this stage, we have a file, where all the Shakespeare works have renamed to "Shakespeare".

Step 2 Separate all the works of "Shakespeare"

In this step, we build a function (`colcut()`), which, given the column title (e.g., "Shakespeare") spit out all the columns with that title including the first column (tokens), which we will write onto a file (`Shakepeare.csv`). Also note the use of the new command `paste`, which merges lines of files and writes to standard output lines consisting of sequentially corresponding lines of each given file.

```
function colcut() { cut -f 1, $(head -1 $1 | sed 's/,/\'$'\n/g' | \
grep -n "$2" | \
cut -f1 -d: | \
paste -sd",") \
-d, $1 ; }
```

We use this function as follows:

```
colcut plays_and_poems_stat.csv Shakespeare > Shakespeare.csv
```

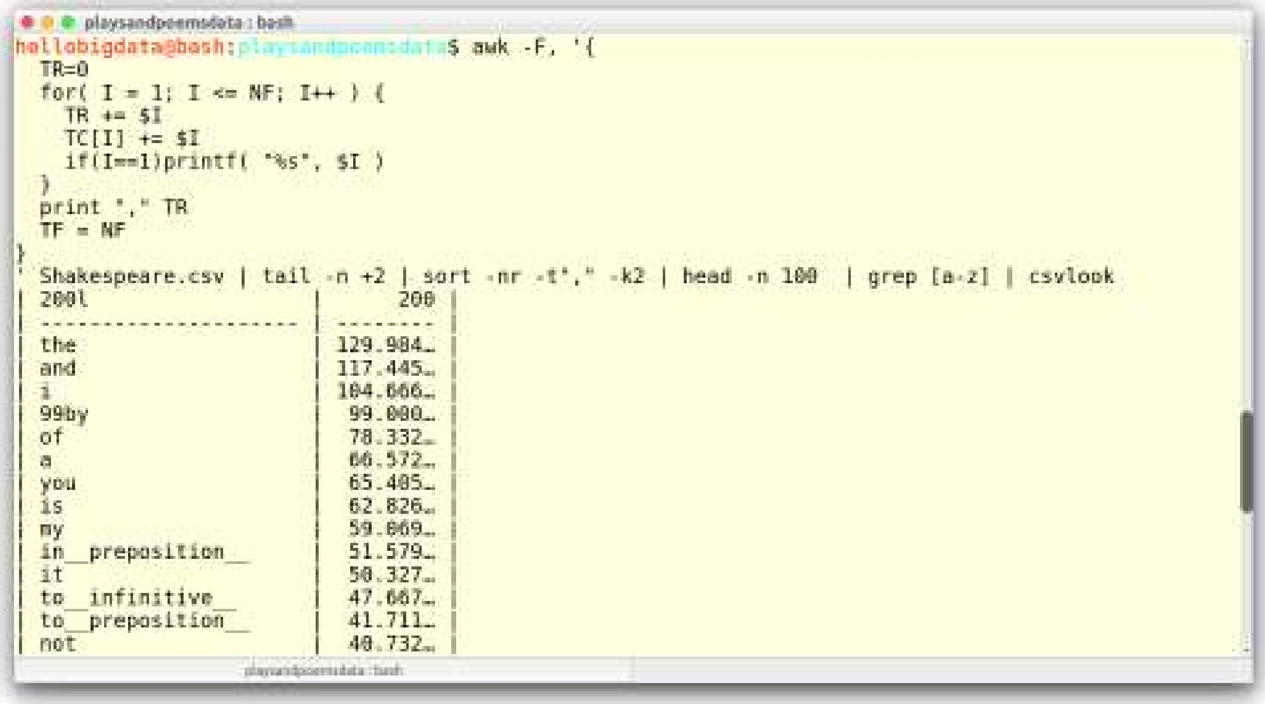
Note that we can not use `csvcut` because it can not handle multiple columns with 'same' title, which is our case (`Shakespeare`).

Step 3. Combine/sum horizontally all the columns with same titles (e.g., `Shakespeare`).

Finally, our final bit of code looks like below. We apply the following `awk` code to the `Shakespeare.csv` file which will do the trick for us!

```
awk -F, '{
  TR=0
  for( I = 1; I <= NF; I++ ) {
    TR += $I

    TC[I] += $I
    if(I==1)printf( "%6s", $I )
  }
  print "," TR
  TF = NF
}
' Shakespeare.csv | tail -n +2 | sort -nr -t"," -k2
```

This small `awk` code will combine and sum horizontally all the columns (for any number of columns). Note that at the end we again sort the output based on the second column (i.e., combined and summed frequencies).



```
hellobigdata@bash:playsandpoems:data$ awk -F, '{
  TR=0
  for( I = 1; I <= NF; I++ ) {
    TR += $I
    TC[I] += $I
    if(I==1)printf( "%s", $I )
  }
  print "," TR
  TF = NF
}
' Shakespeare.csv | tail -n +2 | sort -nr -t"," -k2 | head -n 100  | grep [a-z] | csvlook
| 200l                  |      200 |
| --------------------- | -------- |
| the                   | 129.984… |
| and                   | 117.445… |
| i                     | 104.666… |
| 99by                  |  99.000… |
| of                    |  78.332… |
| a                     |  66.572… |
| you                   |  65.405… |
| is                    |  62.826… |
| my                    |  59.069… |
| in__preposition__     |  51.579… |
| it                    |  50.327… |
| to__infinitive__      |  47.667… |
| to__preposition__     |  41.711… |
| not                   |  40.732… |
```

The final output will look like below:

Note that due to some garbage characters (e.g., page numbers) in the data set, we excluded tokens that are numbers. We only have shown word tokens, using a `grep [a-z]'` at the end of the command. There we go, the most frequent five words in all Shakespearean works:

- `the`,
- `and`,
- `I`,
- `of`, and
- `a`.