Part Two

grep, sort, uniq, wc, tr, and column

New Commands: wc

word **c**ount - counts words, characters, lines

Options:

```
-l, --lines  line count
-w, --words  word count
-m, --chars  character count
-L, --max-line-length
```

wc examples

```
$ wc a.txt
 6 12 24 a.txt # characters, words, lines
# Count files in directory
$ ls | wc -l
# Word count of the bash man page
$ man bash | wc -w
```

grep

grep - a general, line-by-line search tool

- prints lines matching the search pattern
- for multiple files, tells which files matched
- has lots of very powerful options

Syntax:

```
$ grep [options] <pattern> <files>
$ <in> | grep [options] <pattern>
```

grep: examples (1)

```
$ grep 'Carmen Sandiego' world.txt
saw Carmen Sandiego eating lobsters with
$ grep 'ad' a.txt b.txt c.txt
a.txt:the road was peppered with potholes
a.txt:his pothole addled brain couldn't
b.txt:severed heads are not good for
```

some grep options

`--help` list of options and brief explanations

`-E, --extended-regexp`	`-B, --before-context`
`-i, --ignore-case`	`-A, --after-context`
`-v, --invert-match`	`-C, --context`
`-h, --no-filename`	`-L, --files-without-match`
`-o, --only-matching`	`-l, --files-with-match`
`-c, --count`	`-w, --word-regexp`

grep examples (2)

```
# Counts the number of lines that match
grep -c ‘omic’ dictionary.txt
# prints 3 lines of context around the match
grep -iC 3 ‘error’ log.txt
# Lists all files containing the word ‘gene’
grep -liw ‘gene’ *.txt
# prints all files EXCEPT those matching
grep -Li ‘mays’ *.fasta
```

Regular Expressions (1)

```
.      matches any character except a newline
*      matches 0 or more of previous character
+      matches 1 or more of previous character
[]     matches any character enclosed
[^]    matches any character NOT enclosed
^      anchors match at the BEGINNING of the line
$      anchors match at the END of the line
\      escapes the following special character
```

grep examples (3)

```
# lists each match (e.g. AT3G30720) on its own line
$ grep -oE 'AT[0-9]G[0-9]+' at.faa
# counts the occurrences of C and G in a.fna
$ grep -oE ‘[CG]’ a.fna | wc -l
# Count empty lines
$ grep -cE ‘^$’
```

Regular Expressions (2)

```
a|b     matches patterns a OR b
{x}     matches x of the previous char
{x,y}   matches between x and y of the previous char
+       matches 1 or more of the previous char
()      group the enclosed, useful with ‘|’
```

All of these require the **-E**

grep examples (4)

```
# Print all phone numbers from directory
grep -Eo ‘[0-9]{3}-[0-9]{4}’ directory.txt
# Print lines that match Bob OR Jerry
grep -E ‘Bob|Jerry’ a.txt
# Print lines where GT is repeated 5 or more times
grep -E ‘(GT){5,}’ a.fna
```

New Commands: sort

sorts data line-by-line in various ways

```
$ echo ‘a\nc\nb’ | sort
a
b
c
```

some sort options

`--help` list of options and brief explanations

`-g, --general-numeric-sort` (scientific notation)

`-n, --numeric-sort` `-R, --random-sort`

`-r, --reverse` `-f, --ignore-case`

Sorting by column:

`-k, --key=POS1[,POS2]`

`-t, --field-separator=SEP`

sort examples

```
# Reverse numeric sort on columns 4 to 6
$ sort -rnk 4,6 a.txt
# General numeric sort on column 4
$ sort -gk 4,4 b.txt
# Sort on 6 and on 4 in TAB separate file
$ sort -t $'\t' -nk 6,6 -k 4,4 c.tsv
```

New Commands: uniq

deals with unique lines in various ways

INPUT MUST ALREADY BE SORTED

So sort ALWAYS appears upstream of uniq

uniq options

`--help` list of options and brief explanations

`-c, --count` count occurences of each line

`-d, --repeated` print only duplicated lines

`-u, --unique` print only uniq lines

`-i, --ignore-case`

```
sort first-names.txt | uniq -c | sort -rnk 1
```

uniq examples

```
# Sort the names by frequency
sort firstnames.txt | uniq -c | sort -rnk 1,1
# Count the number of uniq names
sort first-names.txt | uniq | wc -l
# Count the names that occur only once
sort first-names.txt | uniq -u | wc -l
```

New Command tr

Translate character sets

```
# Translates a -> x, b -> y and x -> z
tr ‘abc’ ‘xyz’ < myfile.txt
# Delete characters ‘:’, ‘<>’, ‘.’
tr -d ‘:<>.’ < myfile.txt
```

New Command: column

Format output data

```
# align file whitspace columns
cat a.tab | column -t
# set column delimiter
cat a.tab | column -s $’\a’ -t
# fill rows first
cat a.tab | column -x
```

Your Turn

First download the material from github

```
git clone https://github.com/zbwrnz/adv-unix-workshop
```

Navigate to `1st` folder, read the note