# Part Two

grep, sort, uniq, wc

## Four powerful tools

1. wc - count lines, words, or characters
2. grep - search tool
3. sort - flexible sort tool
4. uniq - find unique lines

## Sample Data

Move to `2nd` folder in workshop directory

There you should find these sample file:

`a.fa`, `b.fa`, `c.fa`

`These files hold protein sequences in fasta format`

## Fasta format

```
>gene_identifier_1 other stuff in the header
ARGAVHNVNVNVNNAHKAHKHKHKHKHK
ARATATYYY
>gene_identifier_2 other stuff in the header
PTPLTPTLPTLTPLRERERETETEEEEEEE
QWEWEQWEWEWEWE
```

## New Commands: wc

**w**ord **c**ount - count lines, words and characters

Options:

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

## wc examples

```
# prints count of characters, words, lines
$ man ls | wc
212 938 7713
# Word count, like in MS Word
$ man bash | wc -w
# Count files in the working directory
$ ls | wc -l
```

# 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>
```

## DIY (1)

Find the examples files and try out the following:

```
$ grep ’YYYY’ c.fa
$ grep ’YYYY’ [abc].fa
$ grep ’chloroplast’ [abc].fa
```

## some grep options

--help list of options and brief explanations

**-c, --count**

**-v, --invert-match**

-i, --ignore-case

-w, --word-regexp

-l, --files-with-match

-A, --after-context

-B, --before-context

-C, --context

-h, --no-filename

-L, --files-without-match

## DIY (2)

Try these on our sample fasta files:

```
$ grep -c ‘>’ [abc].faa
$ grep -w -C1 ‘cytochrome’ [abc].fa
$ grep -v ‘cytochrome’ [abc].fa
$ grep -liw ‘chloroplast’ *.fa
$ grep -Liw ‘chloroplast’ *.fa
```

## DIY (3)

Greping the grep man page

```
man grep | grep -A 3 ‘context’
man grep | grep -B 3 ‘context’
man grep | grep -C 3 ‘context’
```

Try making the match more specific and trying other numbers

## Two more options

```
-E, --extended-regexp
-o, --only-matching
```

These commands require regular expressions to be really useful

## Regular Expressions (1)

| | |
|---|---|
| . | matches any character except a newline |
| * | matches 0 or more of previous character |
| + | matches 1 or more of previous character |
| [xyz] | matches characters x, y and z |
| [^xyz] | matches characters OTHER than x, y and z |
| ^ | anchors match at the BEGINNING of the line |
| $ | anchors match at the END of the line |
| \ | escapes the following special character |

# DIY (3)

Try these on our sample fasta files:

```
$ grep -E ‘[0-9]+’ [abc].fa
$ grep -E ‘[DE]+’ a.fa
$ grep -oE ‘[DE]+’ a.fa
$ grep -oE ‘[DE]’ a.fa | wc -l
$ grep -oE ‘gi\|[0-9]+’
```

# DIY (4): chain grep

```
$ grep -oE ‘[^YS]’ a.faa
$ grep -vE ‘^>’ a.faa | grep -oE ‘[^YS]+’
```

## New Commands: sort

**sorts data line-by-line in various ways**

```
$ grep ‘>’ a.fa | sort
```

## some sort options

`--help` list of options and brief explanations

```
-g, --general-numeric-sort
-n, --numeric-sort
-r, --reverse
-u, --unique
```

## DIY (5)

Find the file named ‘unsorted.tab’

Try sorting this file with different combinations of the above options

For example:

```
$ sort unsorted.tab
$ sort -nr unsorted.tab
```

## Sorting by column

Sorting by column:

```
Sort by column
-k, --key=POS

For now, you can ignore this ...
-t, --field-separator=SEP
```

# DIY (6)

Try sorting the unsorted.tab file by different columns. e.g.

```
sort -k2 unsorted.tab
sort -k3g unsorted.tab
```

try `-h` on column 5 and `-M` on 4

# 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 |

## DIY(7)

```
# The following two are identical
$ sort unsorted.tab | uniq
$ sort -u unsorted.tab
# Try these
$ sort unsorted.tab | uniq -c
$ sort unsorted.tab | uniq -d
$ sort unsorted.tab | uniq -u
```

## Pipeline strategies

```
grep | sort | uniq
grep | sort | uniq | wc
<input> | sort | uniq -c | sort -n
```

`Strategy:` Build the pipelines up incrementally, checking output at each step

## DIY(8)

Go back to DIY(3), pipe the grep output into sort and then into uniq

Test out the combos …