Part Two

grep, sort, uniq, wc

Review: Piping and Redirection

Redirection

```
Given A and B are programs and f is a file

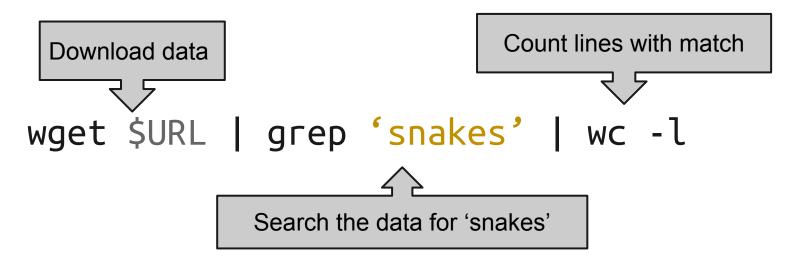
A | B    Pipe output of A to input of B

A > f    Overwrite f with A's output

A >> f    Append A's output to end of f
```

Pipelines

Many UNIX tools can be linked into piplines



Warnings

```
# a.txt will be empty after both commands
$ head a.txt > a.txt
$ head a.txt | A | B > a.txt
```

Never open a file for both reading and writing in one pipeline

Sample data and exercises

Move to section-2/

You should see the following:

- 1. h[12345].txt (5 files)
- 2. script.sh
- 3. solutions.sh
- 4. unsorted.tab

Four powerful tools

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

WC

word count - 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 lines, words, and bytes
$ wc h*.txt
# Word count, like in MS Word
$ man bash | wc -w
# Count files in the working directory
$ ls | wc -l
```

grep

- grep a 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>
```

Examples 2.1

\$ grep 'primrose' h*.txt

```
$ grep 'not to be' h*.txt

If your shell is not coloring the matches,
run the following command:
```

\$ alias grep='grep --color=auto'

Exercise 2.1

Practice wc and basic grep

- Navigate to section-2/
- Make script.sh executable (chmod 755)
- Open file script.sh
- Follow the instructions for Exercise 2.1

some grep options

```
--help list of options and brief explanations
-c, --count -A, --after-context
-v, --invert-match -B, --before-context
-i, --ignore-case -C, --context
-w, --word-regexp -h, --no-filename
-l, --files-with-match -L, --files-without-match
```

Examples 2.3

```
$ grep -c 'Scene' h*.txt
$ grep -C1 'rose' h*.txt
$ grep -wC1 'rose' h*.txt
$ grep -liw 'rose' h*.txt
$ grep -Liw 'rose' h*.txt
$ grep -v 'ACT' h*.txt
```

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
```

Example 2.4

```
$ grep -E '[a-z]+able' h*.txt
$ grep -oE '[a-z]+able' h*.txt
$ grep -E '^\[.*\]$' h*.txt
$ grep -oE '\[.*\]' h*.txt
$ grep -hoE '\[.*\]' h*.txt
```

New Commands: sort

sorts data line-by-line in various ways

\$ sort unsorted.txt

some sort options

- --help list of options and brief explanations-g, --general-numeric-sort
- -n, --numeric-sort
- -r, --reverse
- -u, --unique

Example 2.5

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

Sorting by column

Sorting by column:

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

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

Example 2.6

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
```

-u, --unique print only uniq lines

Example 2.7

```
# 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

Exercise 2.2

Practice building pipelines

Navigate to section-2/

Follow the instructions for Exercise 2.2