CLI Text Processing

Sundeep Agarwal

Know your Environment

```
$ ls *
baz.json  foo.xml  greeting.txt  sample.txt
$ echo *
baz.json foo.xml greeting.txt sample.txt

$ a='foo       123'
$ echo '$a'
$a
$ echo $a
foo 123
$ echo "$a"
foo       123

$ echo foo #123
foo
$ echo 'foo #123'
foo #123
```

Sample file

```
$ cat sample.txt
Hello World!

Good day
How do you do?

Just do it
Believe 42 it!

Today is sunny
Not a bit funny
No doubt you like it too

Much ado about nothing
He he 123 he he
```

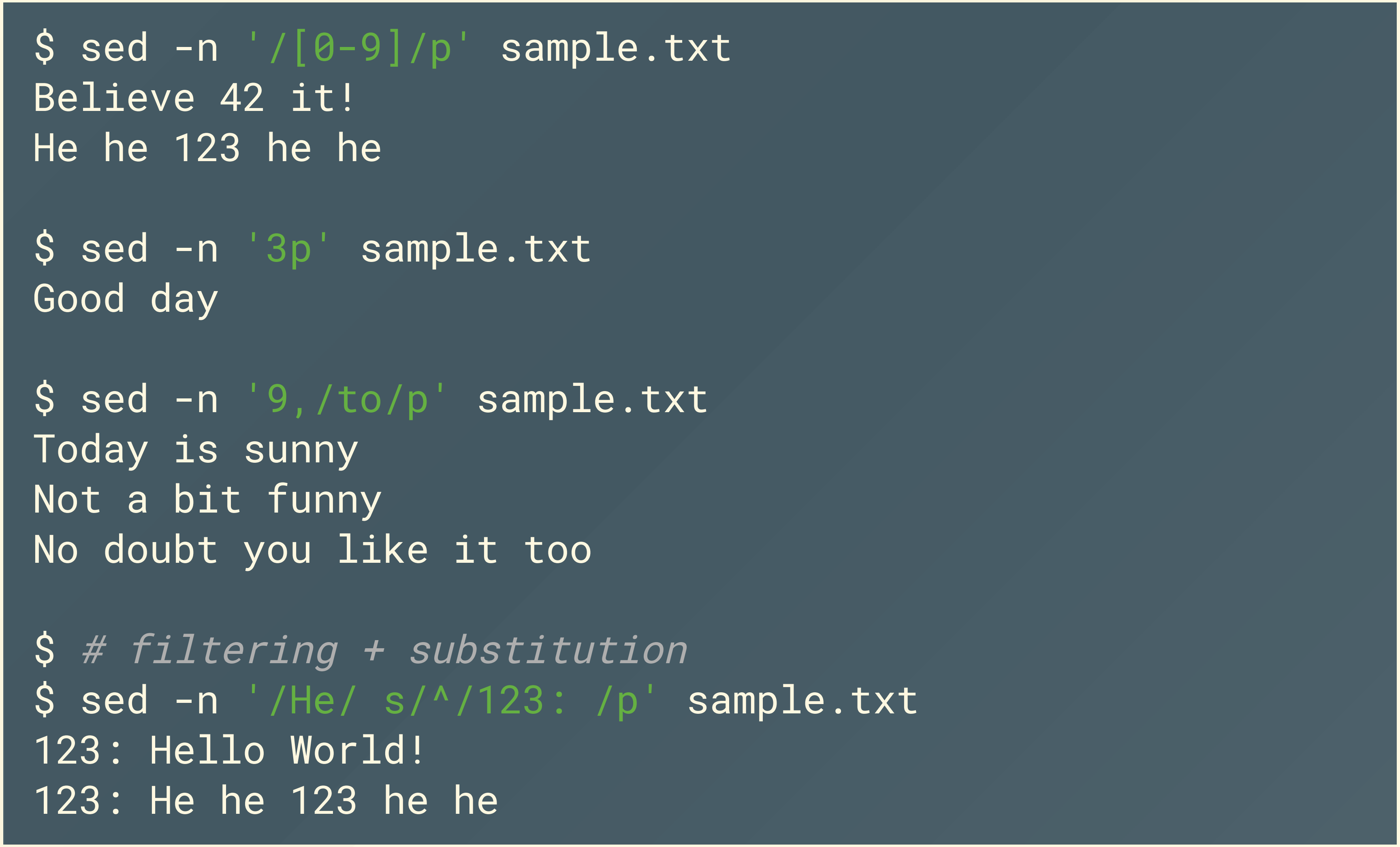
grep

```
$ grep 'day' sample.txt
Good day
Today is sunny

$ grep -i 'to' sample.txt
Today is sunny
No doubt you like it too

$ grep '[0-9]' sample.txt
Believe 42 it!
He he 123 he he

$ grep 'y$' sample.txt
Good day
Today is sunny
Not a bit funny
```

```
$ grep -w 'day' sample.txt
Good day

$ grep -l 'are' *
baz.json
foo.xml

$ grep -oE '[0-9]+' sample.txt
42
123

$ grep -c 'do' sample.txt
4

$ printf 'a.c\nabc' | grep 'a.c'
a.c
abc
$ printf 'a.c\nabc' | grep -F 'a.c'
a.c
```

sed

```
$ cat greeting.txt
Hi there
Have a nice day

$ sed 's/^Hi.*/Hello World!/' greeting.txt
Hello World!
Have a nice day

$ sed -i.bkp 's/[vnt][a-z]*/4/g' greeting.txt
$ cat greeting.txt
Hi 4
Ha4 a 4 day
```



```
$ sed -n '/[0-9]/p' sample.txt
Believe 42 it!
He he 123 he he

$ sed -n '3p' sample.txt
Good day

$ sed -n '9,/to/p' sample.txt
Today is sunny
Not a bit funny
No doubt you like it too

$ # filtering + substitution
$ sed -n '/He/ s/^/123: /p' sample.txt
123: Hello World!
123: He he 123 he he
```

cut

```
$ printf 'foo\tbar\t123\tbaz\n' | cut -f2
bar

$ echo 'foo:bar:123:baz' | cut -d: -f1,4
foo:baz

$ echo 'one;two;three;four' |
       cut -d';' --output-delimiter=',' -f1,3-
one,three,four

$ echo '1,2,3,4,5,6,7' | cut -d, --complement -f1,3
2,4,5,6,7

$ echo 'hi there' | cut -c5-
here
```

awk

```
$ echo 'a b cde' | awk '{print $2}'
b

$ echo 'foo:123:bar:789' | awk -F: '{print $NF}'
789
$ echo 'foo:123:bar:789' | awk -F: '{print $(NF-1)}'
bar

$ echo 'a762b45c' | awk -F'[0-9]+' '{print $1, $NF}'
a c

$ echo '1:2:3:4' | awk -F: -v OFS='-' '{print $2,$4}'
2-4

$ echo 'abc 123 cab' | awk '{gsub(/[a-z]/,"X",$1)} 1'
XXX 123 cab
```

```
$ awk '/do/ && !/it/' sample.txt
How do you do?
Much ado about nothing

$ awk '$1 ~ /u/{print $1}' sample.txt
Just
Much

$ awk 'NR==4' sample.txt
How do you do?

$ awk -v RS= '/bit/' sample.txt
Today is sunny
Not a bit funny
No doubt you like it too
```

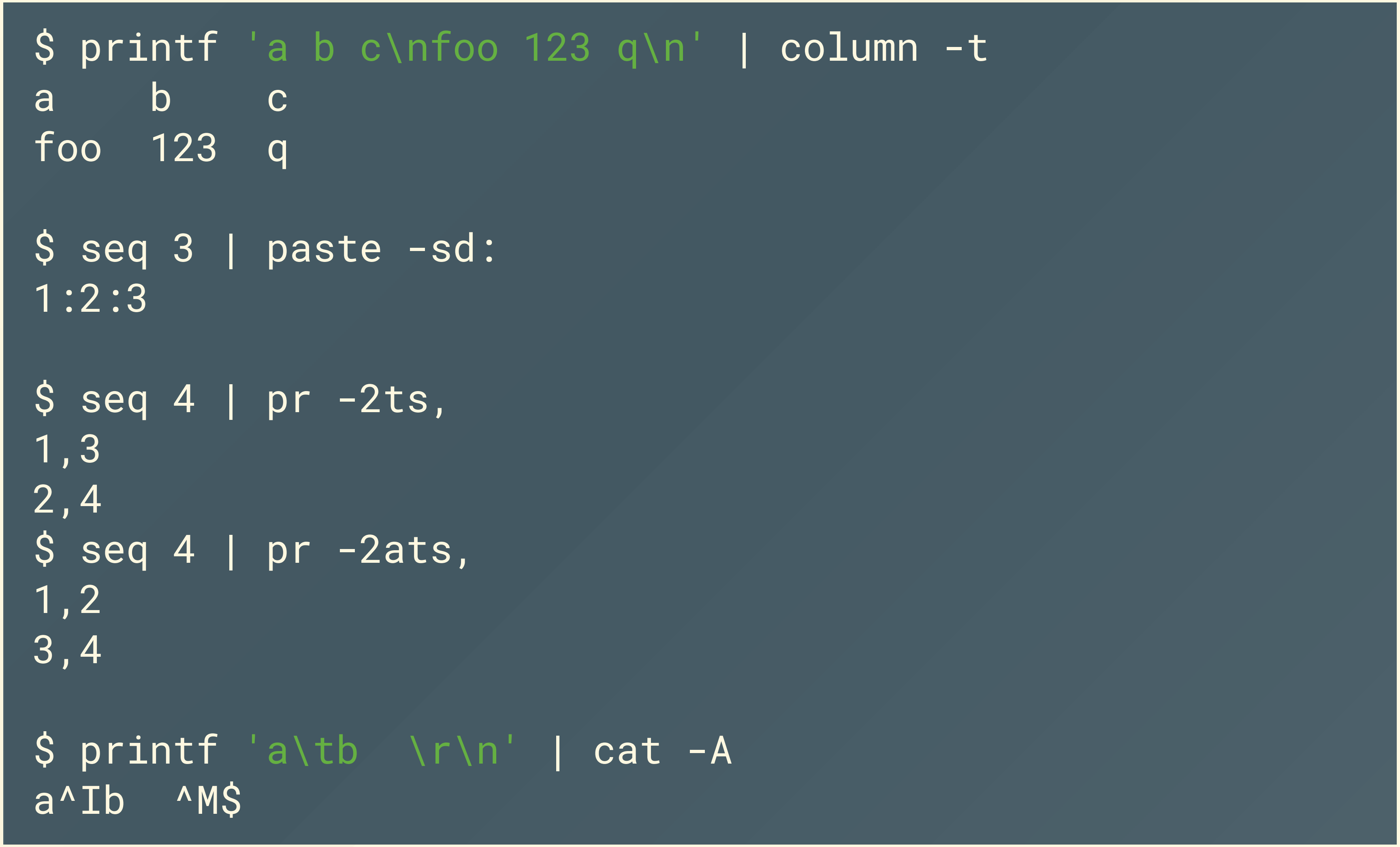
perl

```
$ perl -e 'print "Hello Perl\n"'
Hello Perl

$ echo 'foo baz' | perl -pe 's/foo/123/'
123 baz

$ perl -ne 'print if /\d/' sample.txt
Believe 42 it!
He he 123 he he

$ echo 'foo _foo 1foo' | perl -pe 's/(?<!_)foo/baz/g'
baz _foo 1baz

$ s='I like "mango" and "guava"'
$ echo "$s" | perl -pe 's/"[^"]+"(*SKIP)(*F)|\w+/\U$&/g'
I LIKE "mango" AND "guava"
```

```
$ echo 'a b cde' | perl -lane 'print $F[1]'
b
$ echo 'foo:123:bar:789' | perl -F: -lane 'print $F[-1]'
789


$ echo '1,2,3,4,5,6' | perl -MList::Util=shuffle \
       -F, -lane 'print join ",",shuffle @F'
4,3,2,1,5,6

$ seq 2 | perl -0777 -ne 'print $_ x 3'
1
2
1
2
1
2
```

Misc

```
$ echo 'Hello World' | tr 'a-zA-Z' 'n-za-mN-ZA-M'
Uryyb Jbeyq
$ echo '2017-03-21' | tr -d '-'
20170321

$ printf 'z\nd\nk\n' | sort
d
k
z
$ printf '20\n2' | sort -n
2
20

$ seq 3 | shuf
2
3
1
```



```
$ printf 'a b c\nfoo 123 q\n' | column -t
a    b    c
foo  123  q

$ seq 3 | paste -sd:
1:2:3

$ seq 4 | pr -2ts,
1,3
2,4
$ seq 4 | pr -2ats,
1,2
3,4

$ printf 'a\tb  \r\n' | cat -A
a^Ib  ^M$
```

xml

```
$ cat foo.xml
<foo>
    <abc attr="good">Hi there</abc>
    <xyz attr="bad">I am good. How are you?</xyz>
</foo>
```

```
$ xmlstarlet el foo.xml
foo
foo/abc
foo/xyz
$ xmlstarlet el -a foo.xml
foo
foo/abc
foo/abc/@attr
foo/xyz
foo/xyz/@attr
```

```
$ xmlstarlet sel -t -v 'foo/abc' foo.xml
Hi there

$ xmlstarlet sel -t -v 'foo/xyz/@attr' foo.xml
bad

$ xmlstarlet sel -t -v 'foo/*' foo.xml
Hi there
I am good. How are you?

$ xmlstarlet sel -t -v 'foo/*[@attr="bad"]' foo.xml
I am good. How are you?
```

- http://xmlstar.sourceforge.net/doc/UG/xmlstarlet-ug.html

json

```
$ cat baz.json
{
   "abc": {
      "@attr": "good",
      "text": "Hi there"
   },
   "xyz": {
      "@attr": "bad",
      "text": "I am good. How are you?"
   }
}
```

```
$ jq '.abc.text' baz.json
"Hi there"

$ jq '.[] | .text' baz.json
"Hi there"
"I am good. How are you?"

$ jq '.abc' baz.json
{
  "@attr": "good",
  "text": "Hi there"
}

$ jq '.[] | select(."@attr"=="good") | .text' baz.json
"Hi there"
```

- https://stedolan.github.io/jq/manual/

Further Reading

- Command line text processing
- BashFAQ
- Linux Curated resources
- Why does my regular expression work in X but not in Y?

Presentation prepared using marp