

CSE 374 Lecture 7

Regex and Sed



- Homework 1 && Test 1 due tonight
- Review previous lectures via website
 - Also, text book on web
- Videos on webpage
- ALL grades will be reviewed
- HW/2 posted - writing shell scripts

What have we done so far?

- Use a shell
- Run and combine commands
 - emacs
 - Write shell scripts
 - Regular expressions
- Klaatu, VM
- Pwd, ls, chmod
 - <|>>
- Apropos, man
 - grep

Last Friday - Regular expression and Grep

Can you write a regular expression to identify every phone number?

`\` : escape following character

`'` : matches any single character at least once

`p1|p2` : matches p₁ OR p₂

`'*'` : matches zero or more of the previous p

`'?'` : matches zero or one of the previous p (`\p`)

`'+'` : matches one or more of previous p (`pp*`)

`()` : group patterns for order of operations

`{}` : repeat n times

`[]` : contain literals to be matched (single or range)

`^` : Anchors to beginning of line

`$` : anchors to end of line

`<>` : word boundaries

D., Mark	(206) 901-2345
E., Clarence	+1-206-789-0123
E., Philip	1-206-890-1234
G., Timnit	(206) 4569012
H., Grace	+1 206.345.6789
H., Margaret	(206) 567-8901
J., Katherine	206 456 7890
L., Ada	(206) 123-4567
L., Jerry	2061235678
O., Ellen	206 2346789
T., Alan	206-234-5678
W., Jeannette	206 678.9012



Save (ctrl-s)

New


by [gskinner](#)

Alternation

×

+

*

{1,3}

?

?

the preceding token, effectively

Expression

↔ JavaScript

/(\+?1)?[\-\.]?(\(?[0-9]{3}\)?)[\-\.]?([0-9]{3})[\-\.]?([0-9]{4})/g

Text

Tests NEW

D., Mark

(206) 901-2345

E., Clarence

+1-206-789-0123

E., Philip

1-206-890-1234

G., Bill

206-12-5678

G., Timnit

(206) 4569012

H., Grace

+1 206.345.6789

H., Margaret

(206) 567-8901

J., Katherine

206 456 7890

Tools

Replace

List

De

What is 'sed'?

Run 'man sed' now!

Stream editor: makes basic text transformations on an input stream

Use 'sed *command* file[s]'

Changes line by line, one pass through

Basic usage: sed

```
$ sed [OPTIONS] [COMMAND] [FILE]
```

```
$ input_stream | sed [COMMAND]
```

```
$ sed -i 's/original/replacement/g' test.txt
```

's' - substitute

COMMAND

Input file

'G' - global

'i' - Replaces input
file with updated
version

Useful options:

-i : replace input file with edited version

-e : allows for multiple commands -
applies each left to right (sed -e
's/a/A/' -e 's/b/B/' <old >new)

-f : reads command from a file

-n : suppresses output except when told
otherwise

Omitting file applies [COMMAND] to
stdin

Sed cycle

1. Read one line from input stream
2. Put in pattern space without trailing /n
3. Execute command
 - a. commands with address are only executed if address is verified
4. Pattern space is printed to the output stream

Addresses

Addresses apply only to specific lines. Address comes before command.

Number : only that line number

\$: last line of input

First~step : every 'step' lines starting with 'first'

/regexp/ : only lines matching the regular expression

l1,l2: range - between line that matches l1, and line that matches l2 (l1&l2 can be numbers or regex)

Other types of commands

‘P’ : print this line (often used with ‘-n’ to suppress printing of non-marked lines)

```
sed -n 's/pattern/&/p' <file
```

‘d’ : delete this pattern space and continue

```
$ echo hello world | sed  
'y/abcdefghijklmnopqrstuvwxyz/0123456789/'  
7411o wor13$
```

‘y’ : transliterate characters

```
$ seq 3 | sed '2i hello'  
1  
hello  
2  
3
```

‘a’ : append text

‘i’ : insert text

```
$ seq 10 | sed '2,9c hello'  
1  
hello  
10
```

‘c’ : replace text

sed - more ideas

- Sed encounters one line at a time, and does one pass of the input.
- Delimiter '/' can be changed to anything, like '_' or ':' - may help if COMMAND contains many '/'
- Multi-line editing is possible, but painful, with sed (with 'hold buffer'). Use another scripting program (like 'awk').
- Branches are also possible ('b' and 't' commands)
- Use backreferences (\1, \2 etc) to refer back to regex gathered with \(to \)

What about 'awk'

Or perl? Or ed? Or ruby?

Special purpose language for text editing on an input stream. More programming concepts, used for bigger commands.

Many scripting choices, often with more functionality. Sed stands as the quickest, easiest, and standard on *nix systems for simple commands.

Up next

Introduction to C