CSE 374 Lecture 7

Regex and Sed

- Homework 1 due tonight
- Review previous lectures via website
- Office Hours This Week
  - Monday: Cynthia 11-12
  - Wednesday: Simon 4-5
  - Thursday: Dixon 3-4
  - Friday: Andrew 12-1

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

Grep Review / Breakout Room

Grep - a program to do matching using regular expressions

```
grep '([0-9]*)' numberslist
```

1. `Wget https://courses.cs.washington.edu/courses/cse374/20sp/lectures/numberslist`
2. Un-mute, introduce yourself
3. Work with room-partners to find a regular expression that finds all 14 phone numbers
4. If you find it, paste solution in main-room chat

What is ‘sed’?

Run ‘man sed’ now!

Stream editor: makes basic text transformations on an input stream

Use ‘sed *command* file[s]’

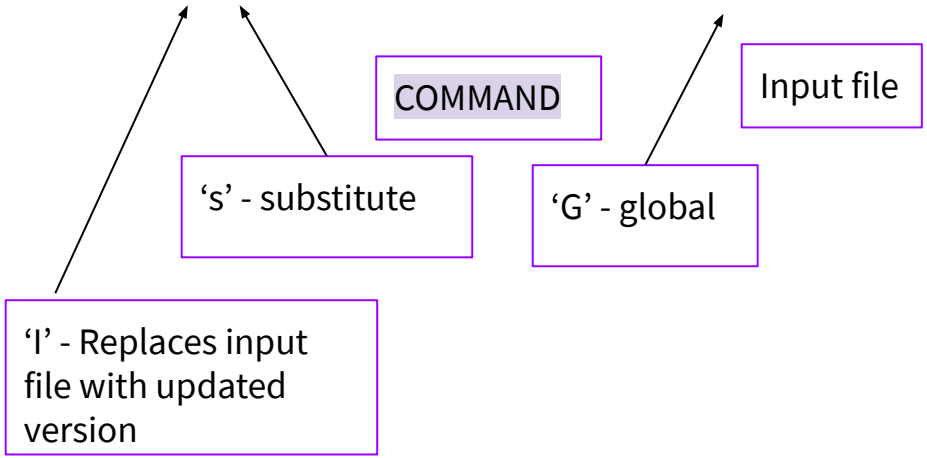
Changes line by line, one pass through

Basic usage: sed

$ sed [OPTIONS] [COMMAND] [FILE]

$ input_stream | sed [COMMAND]

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



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

Other types of commands

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

‘d’ : delete this pattern space and continue

‘y’ : transliterate characters

‘a’: append text

‘i’ : insert text

‘c’ : replace text

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

```
$ echo hello world | sed
'y/abcdefghij/0123456789/'
74llo worl3$
```

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

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

Addresses

Addresses apply sed 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)

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 possibly (‘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