**sed – stream editor**

**sed [OPTIONS] [SCRIPT]** - "stream editor" that allows you to filter and transform text.  
For example:

sed 's/test/example/g' substitutes every occurrence of 'test' with 'example'

**Key Ideas:**

* sed is a **line** editor; it edits one line at a time.
* A line is read from stdin into a *pattern space* buffer, operated on, and then written to stdout.
* You can restrict which lines are possibly modified using "line restrictions"
* There is a *hold buffer* that can be used to temporarily hold input lines.
* sed works on a stream of lines, possibly from multiple files. The lines are numbered continuously starting at 1. If you want the lines to restart on each new file, use the –s option.

stdin

stdout

pattern space

hold space

n, N

p, P

x

h

H

G

g

**sed's Algorithm:**

lineCounter = 0

while (a line exists in the input stream) {

lineCounter++; if (-s and EOF) lineCounter = 1;

copy the line to the pattern space buffer

for each command in the sed script:

execute the command on the text in the pattern space buffer

write the pattern space buffer to stdout.

}

sed OPTIONS

|  |  |
| --- | --- |
| -f scriptFile | Specify a file that contains the sed commands |
| -r | Enable extended regular expressions (always use this option!) |
| -n | Do not print the pattern space to stdout (unless a sed command specifically says to) |
| -i | In-place – changes the original file. (Use this very carefully!) |
| -e command | A sed command to execute. (Not needed if there is only one command.) |

sed 's/pattern/replacementText/gNIp' - substitution command

|  |  |
| --- | --- |
| g | Replace every match of the pattern in the line |
| N | Replace the Nth match of the pattern in the line |
| I | Ignore case |
| p | Output the line to stdout if the pattern was matched. |

Notes:

* The first character after the 's' is the delimiter. Use something other than / if the pattern contains /'s.
* & is the matching text. Use & in the replacement text to include the match in the replacement text.
* \N is the matching text of a capture group. (\1, \2, \3, … \9). E.g., /([^ ]+) \1/ finds duplicate words.

sed COMMANDS (which can be preceeded by an optional line restriction)

|  |  |  |
| --- | --- | --- |
| Range | s | Substitute /pattern/replacementText/ |
| Address | q | Quit sed; all other lines in the input are ignored |
| Address | a\ text | Append a line |
| Address | i\ text | Insert a new line |
| Range | c\ text | Change (replace) a line |
| - | = | Add a line number to the pattern space buffer. (This is tricky. Look up examples!) |
| Range | y | Character for character substitution (like tr) |

Restricting which lines to modify: (Use ! on the command to invert the restriction.)

|  |  |
| --- | --- |
| *N* | A specific line number; 1 is the first line, $ is the last line; an Address |
| N,N | A RANGE of line numbers; 1,$ is the entire file, which is the default. |
| /pattern/ | Any line that contains the pattern |
| /pattern1/,/pattern2/ | A line that contains pattern1 turns editing on; a line that contains pattern2 turns editing off. |
| N,/pattern2/ | At line N, turn editing on; when pattern2 is found, turn editing off. |
| /pattern1/,N | When pattern1 is found, turn editing on; stop editing at line N. |

Buffer manipulation: (refer to the diagram on the previous page)

|  |  |
| --- | --- |
| *n* | Output *pattern space*; empty *pattern space*, read in the next line |
| N | Do not output *pattern space*; do not empty *pattern space*; add the next line to *pattern space* |
| p | Output entire *pattern space*. |
| P | Outputs the first part of the *pattern space* up to the first newLine character |
| x | Exchange the *pattern space* with the *hold space*. |
| h | Copies entire *pattern space* to the *hold space*. |
| H | Appends the entire *pattern space* to the *hold space*. |
| g | Copies the *hold space* to the *pattern space*. |
| G | Appends the *hold space* to the *pattern space*. |

sed examples

|  |  |
| --- | --- |
| sed –r '1,/^$/ d' | Delete all lines up to the first blank line |
| sed –r '/^#/ d' | Delete all lines that start with a # |
| sed –r '/^$/ p' | Every time you find an empty line, output it twice |
| sed –n '1,10 p' | Emulates head (outputs the first 10 lines of the input) |
| sed '11 q' | Emulates head (quits sed on line 11, which outputs the first 10 lines) |
| sed '11,$ d' | Emulates head (deletes all lines from line 11 to the end of the file) |
| sed –rn '/pattern/ p' | Emulates grep (only outputs lines that contain the matching pattern) |
| sed –rn '/pattern/ !p' | Emulates grep –v (output all lines that don't match pattern) |
| sed –r '/pattern/ G' | Insert a blank line after every line that contains the pattern. |
| sed –r 's/^[ \t]\*//' | Delete all white space at the beginning of every line. |
| sed 's/^[ \t]\*//;s/[ \t]\*$//' | Delete all white space at the beginning and end of every line. Note that a semicolon separates two separate commands. |