# sed Stream editor

**START HERE**

`sed 's/<old>/<new>/' <file>`
Replace first match per line

`sed -i 's/<old>/<new>/g' <file>`
Edit file in place

`sed 's/<old>/<new>/g' <file>`
Replace all matches

`sed -n '<N>,<M>p' <file>`
Print a line range

`sed -n '<N>p' <file>`
Print line N only

`sed -E 's/(<re>)/[\1]/' <file>`
Extended regex, group

`sed '<N>d' <file>`
Delete line N

`sed '/<re>/d' <file>`
Delete matching lines

**Read line** into pattern space → run script → **Commands** address? command → auto-print → **Output** stdout, or -i file

Repeated for every input line

## CONCEPTS

sed reads input one line at a time, runs your script on it, and prints the result. Without -i the file itself is never changed.

The hold space is a second buffer that survives between lines; h, H, g, G and x move text between it and the pattern space.

macOS ships BSD sed: -i needs an argument (-i ''), and \U, \n in replacements and 0~2 are not supported. This sheet is GNU sed.

## SUBSTITUTE

`sed 's/<re>/<new>/' <file>`
Replace first match on each line

`sed 's/<re>/<new>/g' <file>`
Replace every match

`sed 's/<re>/<new>/<N>' <file>`
Replace only the Nth match

`sed 's/<re>/<new>/gI' <file>`
Ignore case

`sed 's/<re>/&!/' <file>`
& inserts the whole match

`sed 's|<path>|<new>|g' <file>`
Pick another delimiter for slashes

`sed -E 's/(<a>)(<b>)/\2\1/' <file>`
Swap groups with backreferences

`sed -E 's/<re>/\U&/' <file>`
Uppercase the match (\L lower, \u first)

*\E ends a \U or \L span. GNU only.*

`sed 's/<re>/<new>/w <out>' <file>`
Write changed lines to a file

## SUBSTITUTE (cont.)

`sed 's/<re>/<new>/;t;s/^/NO: /' <file>`
Branch if a substitution happened

## OPTIONS

`sed -n`
Print only when told to (p)

`sed -i[<suffix>]`
Edit in place, optional backup

*-i.bak keeps <file>.bak. Do not put a space between -i and the suffix.*

`sed -E`
Extended regex: (), +, ?, |

`sed -e '<cmd>' -e '<cmd>'`
Combine several commands

`sed -f <script.sed> <file>`
Read commands from a file

`sed -s`
Treat each file as separate (for $, 1)

`sed -z`
Split input on NUL, not newline

*With -z a whole file is one line, so s/\n/ /g works across lines.*

`sed --debug`
Show the parsed script and each step

`sed --posix`
Disable GNU extensions

`sed -u`
Unbuffered, for pipes that stream

## ADDRESSES

`sed '<N>cmd'`
Act on line N

`sed '$cmd'`
Act on the last line

`sed '<N>,<M>cmd'`
Act on lines N to M

`sed '/<re>/cmd'`
Act on lines matching a regex

`sed '/<re>/!cmd'`
Act on lines not matching

## ADDRESSES (cont.)

`sed '/<a>/,/<b>/cmd'`
From the first match to the next

`sed '<N>,+<K>cmd'`
Line N and the K lines after

`sed '<N>~<K>cmd'`
Every Kth line, starting at N

*0~2 means every even line; 1~2 every odd line.*

`sed '0,/<re>/cmd'`
Up to and including the first match

*Lets an empty s//x/ hit only the first match in the file.*

`sed '/<re>/I cmd'`
Case-insensitive address

## COMMANDS

`p`
Print the pattern space

`d`
Delete line, start next cycle

`q / Q`
Quit (Q without printing)

`a\ <text>`
Append text after the line

`i\ <text>`
Insert text before the line

`c\ <text>`
Replace the line with text

`y/<abc>/<xyz>/`
Transliterate characters

`=`
Print the line number

`r <file>`
Append a file's content

`w <file>`
Write the line to a file

`l`
Show line unambiguously (\t, $)

`n / N`
Next line (replace / append)

`{ cmd; cmd }`
Group commands under one address

## COMMANDS (cont.)

`h / H / g / G / x`
Hold space: copy / append / get / get-append / swap

`D / P`
Delete / print up to first newline

`b <label> / t <label> / :<label>`
Branch / branch if substituted / label

## REGEX NOTES

`^ $ . *`
Start, end, any char, repeat

`[abc] [^abc] [a-z]`
Character classes

`\+ \? \| \(\)`
BRE needs backslashes (GNU)

`+ ? | () {n,m}`
Plain in -E mode

`\b \w \s \n`
Word edge, word char, space, newline (GNU)

`[[:digit:]] [[:space:]]`
POSIX classes

`\(<re>\) \1`
Group and backreference in BRE

`\. \* \\ \[`
Escape a special character

## RECIPES

**Delete blank lines and trailing spaces**

```
sed -e '/^$/d' -e 's/[[:space:]]*$//' <file>
```

**Replace across many files (GNU)**

```
grep -rlZ '<old>' <dir> \
  | xargs -0 sed -i 's/<old>/<new>/g'
```

**Print a block between two markers**

```
sed -n '/<start>/,/<end>/p' <file>
```

**Join every two lines / number lines**

```
sed '$!N;s/\n/ /' <file>
sed = <file> | sed 'N;s/\n/\t/'
```