# Part Three

## Substitution with sed

# Sample Data

Move into **`section-3/`**, find the following:

`m.tab` - similar to unsorted.tab in Part 2

`ids.txt` - a file of info on imaginary people

`s.fa` - a protein sequence file

# The power of sed

- delete specific lines or ranges of lines
- search and replace (with style)
- extract specific patterns from files

# sed will not hurt your data

sed reads your data and writes to output.

The output will pour into your terminal unless redirected to a pipe or file.

Your original file is perfectly safe

# sed won’t, but YOU can

**NEVER REDIRECT TO ORIGIN**

**--- Pipelines should not be circular ---**

The following will destroy z.txt:

```
prog1 z.txt | prog2 > z.txt    # BAD!!!
```

# sed syntax

```
sed [OPTIONS] <command>

sed [OPTIONS] '[LINE_ADDRESS] PROCEDURE'
```

# sed workflow

**for each** line of input

remove trailing newline character

**if** line matches the address

perform user's commands

**if** **`-n`** option is NOT set

append newline and print

# Addresses - by number

| | |
|---|---|
| **1** | Matches line number 1 |
| **12** | Matches line number 12 |
| **2,5** | Matches lines 2 to 5 |
| **5,$** | Matches lines 5 and on |

# Addresses - by expression

```
/ham/     Matches lines with pattern ‘ham’
/b/,/a/   Matches from lines matching a to b
1,/ham/   Matches lines 1 to matching ‘ham’
/ham/,$   Matches from ‘ham’ to the end
```

# Procedure: deletion (d)

When the line matches the address, sed does not print, rather it moves onto the next line

# Examples 3.1: deletion (d)

The address can be a number or a regular expression:

```
$ sed 'd' m.txt          # delete everything
$ sed '1d' m.txt         # delete 1st line
$ sed '/Fred/d' m.txt    # delete lines containing 'Fred'
$ sed '5,10d' m.txt      # delete lines 5 to 10
$ sed '10,$d' m.txt      # delete lines from 10 on
$ sed '/R/,/T/d' m.txt   # delete lines between R and T
$ sed '/Fred/,/Duffy/d' m.txt
```

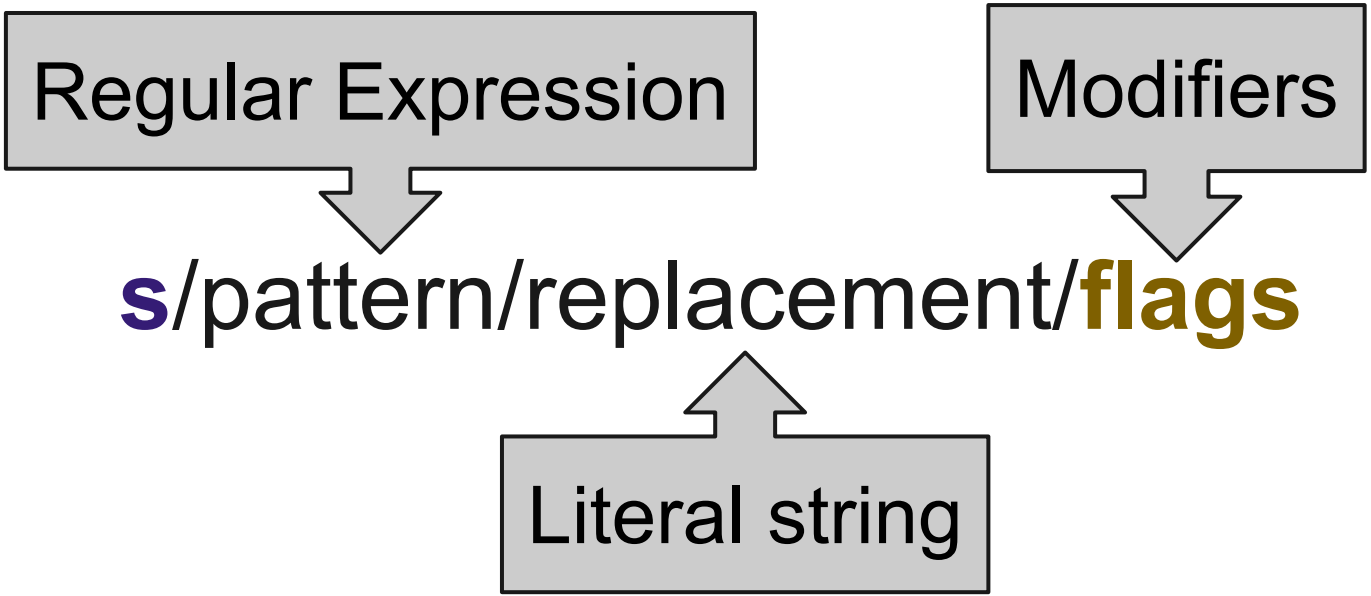
# Regular Expressions (1)

```
.       matches any character except a newline
*       matches 0 or more of the previous char
[...]   matches any of the enclosed
[^...]  matches everything EXCEPT the enclosed
^       anchors match at the BEGINNING of the line
$       anchors match at the END of the line
\       escapes the following special character
```

# Examples 3.2: regex

```
$ sed '/[TA]$/d' m.txt      # Remove if ends in T or A
$ sed '/[^TA]$/d' m.txt     # Remove if not ends in T or A
$ sed '/^Scene/d' h1.txt    # Remove if starts with 'Scene'
$ sed '/^\[/d' h1.txt       # Remove if starts '['
```

# substitution (s)



# Examples

```
# replace each line's 1st 'this' with 'that'
sed 's/this/that/'

# replace EVERY 'this' with 'that' (global flag)
sed 's/this/that/g'
```

# Examples 3.3

```
$ sed ‘s/Fred/Franz/’ m.tab
$ sed ‘/Feb/,/Sep/ s/Bob/Larry/’ m.tab
$ sed ‘s/[1-5]/*/g’ m.tab
$ sed ‘s/\[.*\]//’ h*.txt
```

# Exercise 1.1 (ids.txt)

1) cat `ids.txt`, check format (anything weird?)
2) delete ONLY those who are absent
3) delete ONLY those who are present
4) delete all entries after Mark
5) delete entries with an ‘*’ after the name

# Extended Expressions (2)

`(...)` captures the enclosed sequence

`\n` recalls *n*th captured sequence

`+` matches 1 or more of the previous characters

`|` OR

All of these require the **-r** argument (**-E** on mac)

# Examples 3.4

# [A-Z] matches letters A to Z, similar for [a-z] and [0-9]

```
$ sed '/^[A-Z][a-z]+\.$/d' h1.txt
$ sed '/Bob|Fred/d' m.tab
```

# Exercise 1.2

*s.fa* is formatted as so:

```
>gi|<gi>|ref|<ref>| <description> [<species>]
<sequence line 1>
...
<sequence line N>
```

1. Extract the 4 header regions individually
2. Write the gi and ref to a comma-delimited file

# Print only if substituted

**Problem:**

```
$ sed -r 's/^>gi\|([0-9]+).*/\1/' s.fa
```

You want a list of integers, but all the lines in the input still print

**Solution:**

```
$ sed -rn 's/^>gi\|([0-9]+).*/\1/p' s.fa
```

# -n option and p flag

The **-n** option suppresses printing by default

The **p** flag in substitution forces the line to be printed IF a substitution occurs

Therefore, print only if substitution succeeds

# Extraction strategy

To one of more words from a line:

- Start with the term to be extracted

```
sed -rn 's/.*([0-9]+).*/\1/p'
```

- Make pattern unambiguous by adding context

```
sed -r 's/.*id=([0-9]+).*/\1/'
```

- If no context is necessary, just use `grep -o`

```
grep -oE '[0-9]+'
```

# Supplementary

# ! operator, invert selection

Addresses can be negated with **!**

```
1!          Matches lines NOT equal to 1
2,5!        Matches lines NOT between 2 and 5
```

# ! operator

```
# keep ONLY lines between R and T
$ sed '/R/,/T/!d' m.txt
# Write only the Ghost's lines
$ sed '/^Ghost\.$/,/^$/!d' h*.txt
```