Unit 10
The sed Utility

Unit Objectives

After completing this unit, you should be able to:

- Use the stream edit utility – **sed**
  - Line selection
  - Substitution
  - Delete
  - Print
  - Append, insert, and change
  - Multiple editing
  - And more

sed

There are several ways of running **sed**:

```
sed 'edit-instructions' filename
command | sed 'edit-instructions'
sed -f command.file filename
```

Note: The input file is not changed or overwritten by **sed**!

Line Selection

The **sed** instructions operate on all lines of the input, unless you specify a ***SELECTION*** of lines:

```
sed 'SELECTION edit-instructions'
```

SELECTION can be

- A single line number

  `1` = line `1` of the input

  `$` = the `last line` of the input

- A range of line numbers

  `5,$` = from line `5` to the `end` of the input

- A regular expression to select lines matching a pattern

  `/string/` = selects all lines containing "`string`"

- A range using regular expressions

  `/^on/,/off$/` = from the first line beginning with "`on`" to the first ending in "`off`"

The Substitute Instruction

This instruction changes data:

Syntax: `s/old string/new string/`

Some examples

1. To replace the first occurrence of "Smith" on each line with "Smythe"
   ```
   sed  's/Smith/Smythe/'  phone.list
   ```

2. To replace the same as above using a different delimiter
   ```
   sed  's!Smith!Smythe!g'  phone.list
   ```

3. To replace every match in a line, add the "global"
   ```
   print xxx | sd 's/x/y/'       # responds with yxx
   print xxx | sd 's/x/y/g'      # responds with yyy
   print xxx | sd 's/x/y/2'      # responds with xyx
   ```

Substitutions - Quiz

1. Convert the "phone.list" into just a name list, that is, get rid of the phone numbers

   Desired output:

```
Terrell, Terry
Franklin, Francis
Patterson, Pat
..., ...
```

```
sed  's/_____________________//'  phone.list
```

2. Convert the "phone.list" file to a first-name and number list

   Desired output:

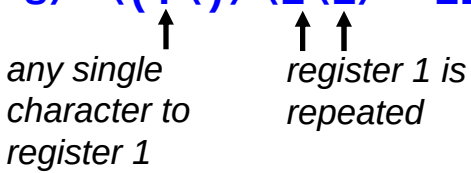
```
Terry                617-7989
Francis              704-3876
Pat                  614-6122
...                  ...
```

```
sed  's/_____________________//'  phone.list
```

sed with Quoted Parentheses

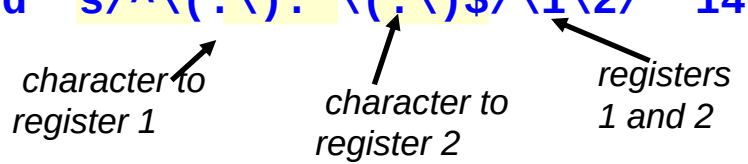
- Repeating the first character

```
$ print "1234" | sed 's/^\(.\)/\1\1/' 11234
$ _
```



- Stripping out all but the first and last characters

```
$ print "1234"|sed 's/^\(.\).*\(.\)$/\1\2/' 14
$ _
```



Now it's your turn...

Working on the "phone.list" file, abbreviate everyone's first name to an initial and a period (use register 1 to store each initial)

```
sed  's/_____________/__________/'  phone.list
```

Delete and Print

This command removes text:

Syntax: `SELECTIONd`

- To delete all lines in the output stream:

```
$ sed  d  phone.list
```

- Delete from line 5 to the end of the file:

```
$ sed  '5,$d'  phone.list
```

- By default `sed` writes out every line it selects
  - Makes print instruction "`p`" by itself redundant:

```
$ cat in.file
line 1
line 2
$ sed p in.file
line 1
line 1
line 2
line 2
$ _
```

Append, Insert, and Change

These instructions add or modify text

Syntax: `SELECTIONx\`
`text`

Where **x** is

i inserts **text** before a single selected line

a appends **text** after a matched line

c changes a range of matched lines into **text**
SELECTION can be a single line or a range
but only one copy of **text** is printed in its place

Command Files

- A **sed** command file consists of one or more **sed** instructions on separate lines
- Command files are useful in many situations:
  - Storing multiple instructions
  - Storing a long complex command
  - For commands which may need to be modified and reused
- Use the **-f** option to use a command file
- Example...

```
$ cat sedscript.sed
  s/ GA/, Georgia/
  s/ FL/, Florida/
  s/ IL/, Illinois/
  s/ TX/, Texas/
  s/ MD/, Maryland/
  s/ DC/, District of Columbia/
$ sed -f sedscript.sed addrs.file > new.addrs.file
$ _
```

A Practical Example

Converting a "BookMaster" script to a "wysiwyg" file

```
:ul.
:li.An unordered list starts with ":ul.".
:li.Each list item is tagged with ":li." - it
appears as an indented bullet point.
:li.The end of the list is marked by ":eul."
:eul.
```

Strategy:

1 Remove lines which contain just ":ul." or ":eul."

2 For lines that start with ":li.", substitute the ":li." with a dash followed by five spaces

```
$ cat  bkm.wysi.sed
/^:e*ul\.$/d
s/^:li\./-     /
$ sed  -f bkm.wysi.sed  bookmaster.file > wysi.file
$ cat wysi.file
-   An unordered list starts with ":ul.".
-   Each list item is tagged with ":li," - it
      appears as an indented bullet point.
-   The end of the list is marked by ":eul."
```

Multiple Editing Instructions

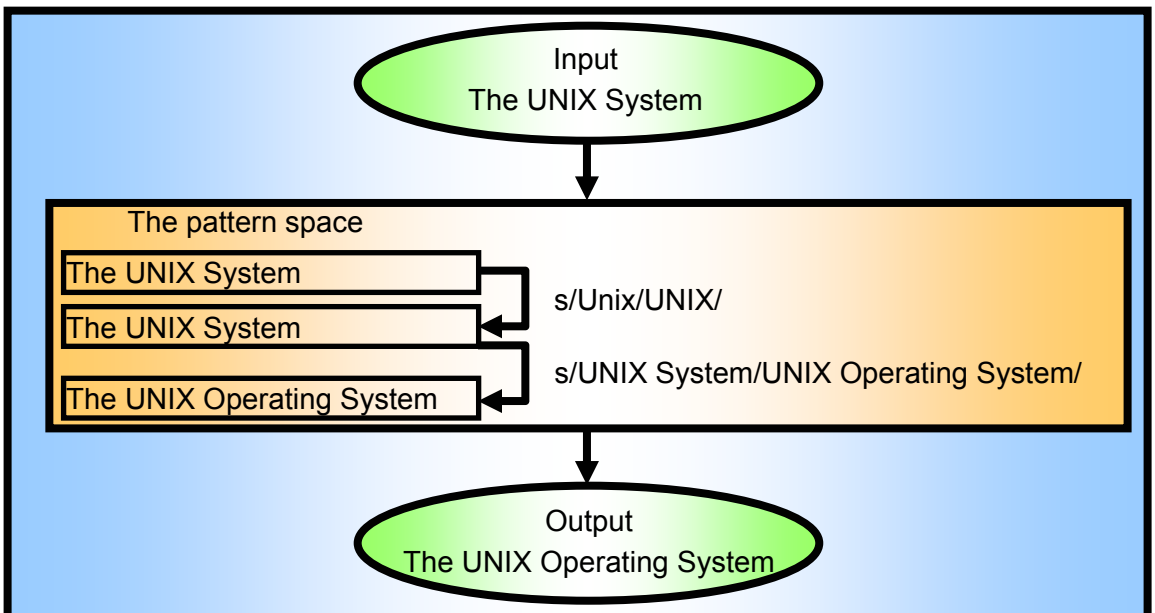
- Multiple instructions can be applied to each line
- Each instruction must be on a separate line

Example:

```
$ sed      '/[1-4]-/s/$/ (Bldng 1) /
>           /[5-9]-/s/$/ (Bldng 2) /'  phone.list

Terrell, Terry          617-7989     (Bldng 2)
Franklin, Francis       704-3876     (Bldng 1)
```

Internal Operation



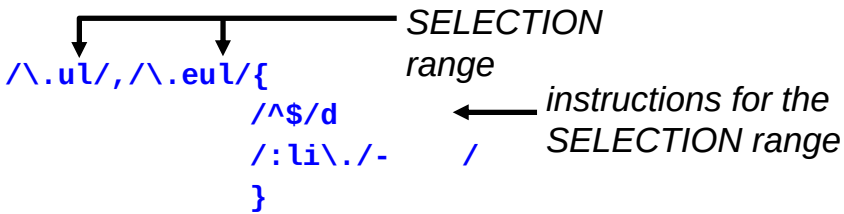
- **sed** applies all editing instructions to a line before it moves on to the next line.
- It holds each input line in a "pattern space" or temporary buffer while editing instructions are applied in sequence.

Grouping Instructions

Braces "**{**" "**}**" are used for two purposes:

- One **SELECTION** inside another (*nest*)
- To apply multiple instructions to the same **SELECTION** range (*group*)

Example...



- The instruction "**/^$/d**" (delete blank lines) will be applied to a range of lines between one that contains an "**.ul**" and up to the first containing an "**.eul**", as will the "**/:li\./- /**"
- The special meaning of the dot preceding "**ul**" and "**eul**" is escaped by the use of a backslash

Checkpoint

1. Write a command line script that displays a **`ps -ef`** with your username as the owner of init.

2. How can I make phone.list appear double spaced?

3. Cat out the sulog (located in **`/var/adm/sulog`**) and change all + to the word "successful" and all - to the word "unsuccessful" using sed.

4. Using **`sed`**, insert **`#!/usr/bin/ksh`** as the first line of a program called program1 and store in program2.

Unit Summary

- Use of **sed** to automate repetitive editing tasks
  - Line selection
  - Substitution
  - Delete
  - Print
  - Append, insert, and change
  - Multiple editing
  - And more