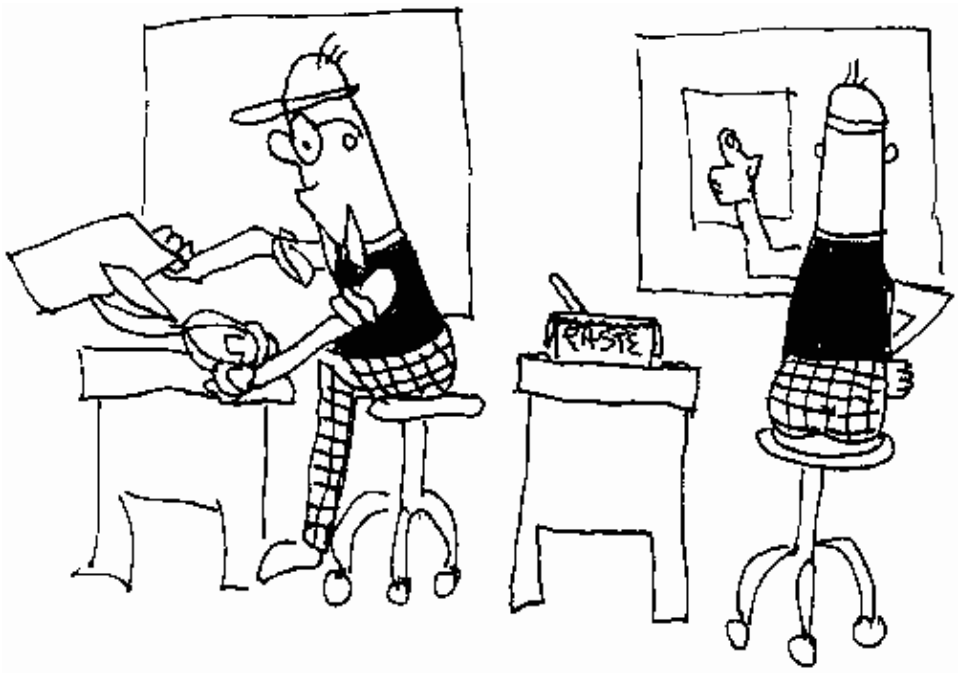
CONTENTS

Chapter 4. sed, the Streamlined Editor



4.1 What Is sed?

Sed is a streamlined, noninteractive editor. It allows you to perform the same kind of editing tasks used in the vi and ex editors. Instead of working interactively with the editor, the sed program lets you type your editing commands at the command line, name the file, and then see the output of the editing command on the screen. The sed editor is nondestructive. It does not change your file unless you save the output with shell redirection. All lines are printed to the screen by default.

The sed editor is useful in shell scripts, where using interactive editors such as vi or ex would require the user of the script to have familiarity with the editor and would expose the user to making unwanted modifications to the open file. You can also put sed commands in a file called a sed script, if you need to perform multiple edits or do not like worrying about quoting the sed commands at the shell command line.[1]

4.2 How Does sed Work?

The sed editor processes a file (or input) one line at a time and sends its output to the screen. Its commands are those you may recognize from the vi and ed/ex editors. Sed stores the line it is currently processing in a

temporary buffer called a pattern space. Once sed is finished processing the line in the pattern space (i.e., executing sed commands on that line), the line in the pattern space is sent to the screen (unless the command was to delete the line or suppress its printing). After the line has been processed, it is removed from the pattern space and the next line is then read into the pattern space, processed, and displayed. Sed ends when the last line of the input file has been processed. By storing each line in a temporary buffer and performing edits on that line, the original file is never altered or destroyed.

4.3 Addressing

You can use addressing to decide which lines you want to edit. The addresses can be in the form of numbers or regular expressions, or a combination of both. Without specifying an address, sed processes all lines of the input file.

When an address consists of a number, the number represents a line number. A dollar sign can be used to represent the last line of the input file. If a comma separates two line numbers, the addresses that will be processed are within that range of lines, including the first and last line in the range. The range may be numbers, regular expressions, or a combination of numbers and regular expressions.

Sed commands tell sed what to do with the line: print it, remove it, change it, and so forth.

FORMAT

```
sed 'command' filename(s)
```

Example 4.1

```
1  % sed '1,3p' myfile
2  % sed −n '/[Jj]ohn/p' datafile
```

EXPLANATION

1. Lines 1 through 3 of myfile are printed.
2. Only lines matching the pattern John or john in myfile are printed.

4.4 Commands and Options

Sed commands tell sed how to process each line of input specified by an address. If an address is not given, sed processes every line of input. (The % is the csh prompt.) See Table 4.1 for a list of sed commands and what they do, and see Table 4.2 for a list of options and how they control sed's behavior.

Example 4.2

```
% sed '1,3d' file
```

EXPLANATION

Sed will delete lines 1 through 3.

Table 4.1. sed Commands

Command Functiona\Appends one or more lines of text to the current line.c\Changes (replaces) text in the current line with new text.dDeletes lines.i\Inserts text above the current line.hCopies the contents of the pattern space to a holding buffer.HAppends the contents of the pattern space to a holding buffer.gGets what is in the holding buffer and copies it into the pattern buffer, overwriting what was there.GGets what is in the holding buffer and copies it into the pattern buffer, appending to what was there.lLists nonprinting characters.pPrints lines.nReads the next input line and starts processing the newline with the next command rather than the first command.qQuits or exits sed.rReads lines from a file.!Applies the command to all lines except the selected ones.sSubstitutes one string for another. Substitution Flags:gGlobally substitutes on a line.pPrints lines.wWrites lines out to a file.xExchanges contents of the holding buffer with the pattern space.yTranslates one character to another (cannot use regular expression metacharacters with y).

Table 4.2. sed Options

Options Function−eAllows multiple edits.−fPrecedes a sed script filename.−nSuppresses default output.

When multiple commands are used or addresses need to be nested within a range of addresses, the commands are enclosed in curly braces and each command is either on a separate line or terminated with semicolons.

The exclamation point (!) can be used to negate a command. For example,

```
% sed '/Tom/d' file
```

tells sed to delete all lines containing the pattern Tom, whereas

```
% sed '/Tom/!d' file
```

tells sed to delete lines not containing Tom.

The sed options are −e, −f, and, −n. The −e is used for multiple edits at the command line, the −f precedes a sed script filename, and the −n suppresses printing output.

4.5 Error Messages and Exit Status

When sed encounters a syntax error, it sends a pretty straightforward error message to standard error, but if it cannot figure out what you did wrong, sed gets "garbled," which we could guess means confused. The exit status that sed returns to the shell, if its syntax is error−free, is a zero for success and a nonzero integer for failure.[2]

Example 4.3

```
1   % sed '1,3v ' file
    sed: Unrecognized command: 1,3v
    % echo $status (echo $?  if using Korn or Bourne shell)
    2

2   % sed '/^John' file
    sed: Illegal or missing delimiter: /^John
```

```
3   % sed 's/134345/g' file
    sed: Ending delimiter missing on substitution: s/134345/g
```

EXPLANATION

1. The v command is unrecognized by sed. The exit status was 2, indicating that sed exited with a syntax problem.
2. The pattern /^John is missing the closing forward slash.
3. The substitution command, s, contains the search string but not the replacement string.

4.5.1 Metacharacters

Like grep, sed supports a number of special metacharacters to control pattern searching. See Table 4.3

Table 4.3. sed's Regular Expression Metacharacters.

Metacharacter Function ExampleWhat It Matches^Beginning−of−line anchor/^love/Matches all lines beginning with love.$End−of−line anchor/love$/Matches all lines ending with love..Matches one character, but not the newline character/l..e/Matches lines containing an l, followed by two characters, followed by an e.*Matches zero or more characters/ *love/Matches lines with zero or more spaces, followed by the pattern love.[ ]Matches one character in the set/[Ll]ove/Matches lines containing love or Love.[^]Matches one character not in the set/[^A−KM−Z]ove/Matches lines not containing A through K or M through Z followed by ove.\(..\)Saves matched characters s/\(love\)able/\1er/Tags marked portion and saves it as tag number 1. To reference later, use \1 to reference the pattern. May use up to nine tags, starting with the first tag at the leftmost part of the pattern. For example, love is saved in register 1 and remembered in the replacement string. lovable is replaced with lover.&Saves search string so it can be remembered in the replacement string s/love/**&**/The ampersand represents the search string. The string love will be replaced with itself surrounded by asterisks; i.e., love will become **love**.\<Beginning−of−word anchor/\<love/Matches lines containing a word that begins with love.\>End−of−word anchor/love\>/Matches lines containing a word that ends with love.

x\{m\}

x\{m,\}

x\{m,n\}[a]

Repetition of character x: m times

at least m times

at least m and not more than n times

/o\{5\}/Matches if line has 5 occurrences of o, at least 5 occurrences of o, or between 5 and 10 occurrences of o.

[a] Not dependable on all versions of UNIX or all pattern−matching utilities; usually works with vi and grep.

4.6 sed Examples

```
% cat datafile
northwest           NW   Charles Main       3.0    .98    3    43
western             WE   Sharon Gray        5.3    .97    5    23
southwest           SW   Lewis Dalsass      2.7    .8     2    18
southern            SO   Suan Chin          5.1    .95    4    15
southeast           SE   Patricia Hemenway  4.0    .7     4    17
eastern             EA   TB Savage          4.4    .84    5    20
northeast           NE   AM Main Jr.        5.1    .94    3    13
north               NO   Margot Weber       4.5    .89    5     9
central             CT   Ann Stephens       5.7    .94    5    13
```

4.6.1 Printing: The p Command

Example 4.4

```
sed '/north/p' datafile
northwest     NW     Charles Main        3.0  .98   3   34
northwest     NW     Charles Main        3.0  .98   3   34
western       WE     Sharon Gray         5.3  .97   5   23
southwest     SW     Lewis Dalsass       2.7  .8    2   18
southern      SO     Suan Chin           5.1  .95   4   15
southeast     SE     Patricia Hemenway   4.0  .7    4   17
eastern       EA     TB Savage           4.4  .84   5   20
northeast     NE     AM Main Jr.         5.1  .94   3   13
northeast     NE     AM Main Jr.         5.1  .94   3   13
north         NO     Margot Weber        4.5  .89   5   9
north         NO     Margot Weber        4.5  .89   5   9
central       CT     Ann Stephens        5.7  .94   5   13
```

> **EXPLANATION**
>
> Prints all lines to standard output by default. If the pattern north is found, sed will print that line in addition to all the other lines.

Example 4.5

```
sed −n '/north/p' datafile
northwest     NW     Charles Main        3.0  .98   3   34
northeast     NE     AM Main Jr.         5.1  .94   3   13
north         NO     Margot Weber        4.5  .89   5    9
```

> **EXPLANATION**
>
> The −n option suppresses the default behavior of sed when used with the p command. Without the −n option, sed will print duplicate lines of output as shown in the preceding example. Only the lines containing the pattern north are printed when −n is used.

```
% cat datafile
northwest           NW   Charles Main       3.0   .98   3   34
western             WE   Sharon Gray        5.3   .97   5   23
southwest           SW   Lewis Dalsass      2.7   .8    2   18
southern            SO   Suan Chin          5.1   .95   4   15
southeast           SE   Patricia Hemenway  4.0   .7    4   17
eastern             EA   TB Savage          4.4   .84   5   20
northeast           NE   AM Main Jr.        5.1   .94   3   13
north               NO   Margot Weber       4.5   .89   5    9
```

```
central            CT   Ann Stephens       5.7   .94   5   13
```

4.6.2 Deleting: The d Command

Example 4.6

```
sed '3d' datafile
northwest     NW     Charles Main       3.0  .98   3   34
western       WE     Sharon Gray        5.3  .97   5   23
southern      SO     Suan Chin          5.1  .95   4   15
southeast     SE     Patricia Hemenway  4.0  .7    4   17
eastern       EA     TB Savage          4.4  .84   5   20
northeast     NE     AM Main Jr.        5.1  .94   3   13
north         NO     Margot Weber       4.5  .89   5    9
central       CT     Ann Stephens       5.7  .94   5   13
```

EXPLANATION

Deletes the third line. All other lines are printed to the screen by default.

Example 4.7

```
sed '3,$d' datafile
northwest      NW    Charles Main       3.0  .98   3   34
western        WE    Sharon Gray        5.3  .97   5   23
```

EXPLANATION

The third line through the last line are deleted. The remaining lines are printed. The dollar sign ($) represents the last line of the file. The comma is called the range operator. In this example, the range of addresses starts at line 3 and ends at the last line, which is represented by the dollar sign ($).

Example 4.8

```
sed '$d' datafile
northwest     NW     Charles Main        3.0  .98   3  34
western       WE     Sharon Gray         5.3  .97   5  23
southwest     SW     Lewis Dalsass       2.7  .8    2  18
southern      SO     Suan Chin           5.1  .95   4  15
southeast     SE     Patricia Hemenway   4.0  .7    4  17
eastern       EA     TB Savage           4.4  .84   5  20
northeast     NE     AM Main Jr.         5.1  .94   3  13
north         NO     Margot Weber        4.5  .89   5   9
```

EXPLANATION

Deletes the last line. The dollar sign ($) represents the last line. The default is to print all of the lines except those affected by the d command.

Example 4.9

```
sed '/north/d' datafile
western       WE     Sharon Gray         5.3  .97   5   23
southwest     SW     Lewis Dalsass       2.7  .8    2   18
southern      SO     Suan Chin           5.1  .95   4   15
southeast     SE     Patricia Hemenway   4.0  .7    4   17
```

```
eastern       EA     TB Savage          4.4  .84   5   20
central       CT     Ann Stevens        5.7  .94   5   13
```

EXPLANATION

All lines containing the pattern north are deleted. The remaining lines are printed.

4.6.3 Substitution: The s Command

Example 4.10

```
sed 's/west/north/g' datafile
northnorth    NW     Charles Main       3.0  .98   3   34
northern      WE     Sharon Gray        5.3  .97   5   23
southnorth    SW     Lewis Dalsass      2.7  .8    2   18
southern      SO     Suan Chin          5.1  .95   4   15
southeast     SE     Patricia Hemenway  4.0  .7    4   17
eastern       EA     TB Savage          4.4  .84   5   20
northeast     NE     AM Main Jr.        5.1  .94   3   13
north         NO     Margot Weber       4.5  .89   5    9
central       CT     Ann Stephens       5.7  .94   5   13
```

EXPLANATION

The s command is for substitution. The g flag at the end of the command indicates that the substitution is global across the line; that is, if multiple occurrences of west are found, all of them will be replaced with north. Without the g command, only the first occurrence of west on each line would be replaced with north.

```
% cat datafile
northwest          NW   Charles Main       3.0   .98   3   34
western            WE   Sharon Gray        5.3   .97   5   23
southwest          SW   Lewis Dalsass      2.7   .8    2   18
southern           SO   Suan Chin          5.1   .95   4   15
southeast          SE   Patricia Hemenway  4.0   .7    4   17
eastern            EA   TB Savage          4.4   .84   5   20
northeast          NE   AM Main Jr.        5.1   .94   3   13
north              NO   Margot Weber       4.5   .89   5   9
central            CT   Ann Stephens       5.7   .94   5   13
```

Example 4.11

```
sed −n 's/^west/north/p' datafile
northern      WE     Sharon Gray        5.3  .97   5   23
```

EXPLANATION

The s command is for substitution. The −n option with the p flag at the end of the command tells sed to print only those lines where the substitution occurred; that is, if west is found at the beginning of the line and is replaced with north, just those lines are printed.

Example 4.12

```
sed 's/[0–9][0–9]$/&.5/' datafile
northwest     NW     Charles Main       3.0  .98   3   34.5
western       WE     Sharon Gray        5 3  .97   5   23.5
southwest     SW     Lewis Dalsass      2.7  .8    2   18.5
```

```
southern        SO      Suan Chin           5.1   .95    4    15.5
southeast       SE      Patricia Hemenway   4.0   .7     4    17.5
eastern         EA      TB Savage           4.4   .84    5    20.5
northeast       NE      AM Main Jr.         5.1   .94    3    13.5
north           NO      Margot Weber        4.5   .89    5    9
central         CT      Ann Stephens        5.7   .94    5    13.5
```

EXPLANATION

The ampersand[3] (&) in the replacement string represents exactly what was found in the search string. Each line that ends in two digits will be replaced by itself, and .5 will be appended to it.

Example 4.13

```
sed −n 's/Hemenway/Jones/gp' datafile
southeast       SE      Patricia Jones      4.0   .7    4     17
```

EXPLANATION

All occurrences of Hemenway are replaced with Jones, and only the lines that changed are printed. The −n option combined with the p command suppresses the default output. The g stands for global substitution across the line.

Example 4.14

```
sed −n 's/\(Mar\)got/\1ianne/p' datafile
north           NO      Marianne Weber      4.5   .89    5     9
```

EXPLANATION

The pattern Mar is enclosed in parentheses and saved as tag 1 in a special register. It will be referenced in the replacement string as \1. Margot is then replaced with Marianne.

Example 4.15

```
sed 's#3#88#g' datafile
northwest       NW      Charles Main        88.0   .98    88    884
western         WE      Sharon Gray         5.88   .97     5    288
southwest       SW      Lewis Dalsass       2.7    .8      2    18
southern        SO      Suan Chin           5.1    .95     4    15
southeast       SE      Patricia Hemenway   4.0    .7      4    17
eastern         EA      TB Savage           4.4    .84     5    20
northeast       NE      AM Main Jr.         5.1    .94    88    188
north           NO      Margot Weber        4.5    .89     5    9
central         CT      Ann Stephens        5.7    .94     5    188
```

EXPLANATION

The character after the s command is the delimiter between the search string and the replacement string. The delimiter character is a forward slash by default, but can be changed (only when the s command is used). Whatever character follows the s command is the new string delimiter. This technique can be useful when searching for patterns containing a forward slash, such as pathnames or birthdays.

```
% cat datafile
northwest           NW   Charles Main       3.0    .98    3     34
western             WE   Sharon Gray        5.3    .97    5     23
```

```
southwest          SW   Lewis Dalsass      2.7   .8    2    18
southern           SO   Suan Chin          5.1   .95   4    15
southeast          SE   Patricia Hemenway  4.0   .7    4    17
eastern            EA   TB Savage          4.4   .84   5    20
northeast          NE   AM Main Jr.        5.1   .94   3    13
north              NO   Margot Weber       4.5   .89   5     9
central            CT   Ann Stephens       5.7   .94   5    13
```

4.6.4 Range of Selected Lines: The Comma

Example 4.16

```
sed -n '/west/,/east/p' datafile
northwest      NW     Charles Main       3.0  .98   3    34
western        WE     Sharon Gray        5.3  .97   5    23
southwest      SW     Lewis Dalsass      2.7  .8    2    18
southern       SO     Suan Chin          5.1  .95   4    15
southeast      SE     Patricia Hemenway  4.0  .7    4    17
```

EXPLANATION

All lines in the range of patterns between west and east are printed. If west were to appear on a line after east, the lines from west to the next east or to the end of file, whichever comes first, would be printed. The arrows mark the range.

Example 4.17

```
sed -n '5,/^northeast/p' datafile
southeast      SE     Patricia Hemenway  4.0  .7    4    17
eastern        EA     TB Savage          4.4  .84   5    20
northeast      NE     AM Main Jr.        5.1  .94   3    13
```

EXPLANATION

Prints the lines from line 5 through the first line that begins with northeast.

Example 4.18

```
sed '/west/,/east/s/$/**VACA**/' datafile
northwest      NW     Charles Main       3.0  .98   3    34**VACA**
western        WE     Sharon Gray        5.3  .97   5    23**VACA**
southwest      SW     Lewis Dalsass      2.7  .8    2    18**VACA**
southern       SO     Suan Chin          5.1  .95   4    15**VACA**
southeast      SE     Patricia Hemenway  4.0  .7    4    17**VACA**
eastern        EA     TB Savage          4.4  .84   5    20
northeast      NE     AM Main Jr.        5.1  .94   3    13
north          NO     Margot Weber       4.5  .89   5    9
central        CT     Ann Stephens       5.7  .94   5    13
```

EXPLANATION

For lines in the range between the patterns east and west, the end of line ($) is replaced with the string **VACA**. The newline is moved over to the end of the new string. The arrows mark the range.

4.6.5 Multiple Edits: The e Command

Example 4.19

```
sed −e '1,3d' −e 's/Hemenway/Jones/' datafile
southern      SO     Suan Chin          5.1  .95   4   15
southeast     SE     Patricia Jones     4.0  .7    4   17
eastern       EA     TB Savage          4.4  .84   5   20
northeast     NE     AM Main            5.1  .94   3   13
north         NO     Margot Weber       4.5  .89   5   9
central       CT     Ann Stephens       5.7  .94   5   13
```

EXPLANATION

The −e option allows multiple edits. The first edit removes lines 1 through 3. The second edit substitutes Hemenway with Jones. Since both edits are done on a per−line basis (i.e., both commands are executed on the current line in the pattern space), the order of the edits may affect the outcome. For example, if both commands had performed substitutions on the line, the first substitution could affect the second substitution.

```
% cat datafile
northwest          NW   Charles Main       3.0   .98   3   34
western            WE   Sharon Gray        5.3   .97   5   23
southwest          SW   Lewis Dalsass      2.7   .8    2   18
southern           SO   Suan Chin          5.1   .95   4   15
southeast          SE   Patricia Hemenway  4.0   .7    4   17
eastern            EA   TB Savage          4.4   .84   5   20
northeast          NE   AM Main Jr.        5.1   .94   3   13
north              NO   Margot Weber       4.5   .89   5    9
central            CT   Ann Stephens       5.7   .94   5   13
```

4.6.6 Reading from Files: The r Command

Example 4.20

```
% cat newfile
              ___________________________________
              | ***SUAN HAS LEFT THE COMPANY*** |
              |_________________________________|
% sed '/Suan/r newfile' datafile
northwest     NW     Charles Main       3.0  .98   3   34
western       WE     Sharon Gray        5.3  .97   5   23
southwest     SW     Lewis Dalsass      2.7  .8    2   18
southern      SO     Suan Chin          5.1  .95   4   15
              ___________________________________
              | ***SUAN HAS LEFT THE COMPANY***   |
              |___________________________________|
southeast     SE     Patricia Hemenway  4.0  .7    4   17
eastern       EA     TB Savage          4.4  .84   5   20
northeast     NE     AM Main            5.1  .94   3   13
north         NO     Margot Weber       4.5  .89   5   9
central       CT     Ann Stephens       5.7  .94   5   13
```

EXPLANATION

The r command reads specified lines from a file.The contents of newfile are read into the input file datafile, after the line where the pattern Suan is matched. If Suan had appeared on more than one line, the contents of newfile would have been read in under each occurrence.

4.6.7 Writing to Files: The w Command

Example 4.21

```
sed −n '/north/w newfile' datafile
cat newfile
northwest     NW     Charles Main       3.0  .98   3  34
northeast     NE     AM Main Jr.        5.1  .94   3  13
north         NO     Margot Weber       4.5  .89   5   9
```

EXPLANATION

The w command writes specified lines to a file. All lines containing the pattern north are written to a file called newfile.

4.6.8 Appending: The a Command

Example 4.22

```
sed '/^north  /a\\
−−−>THE NORTH SALES DISTRICT HAS MOVED<−−−' datafile
northwest     NW     Charles Main       3.0  .98   3   34
western       WE     Sharon Gray        5.3  .97   5   23
southwest     SW     Lewis Dalsass      2.7  .8    2   18
southern      SO     Suan Chin          5.1  .95   4   15
southeast     SE     Patricia Hemenway  4.0  .7    4   17
eastern       EA     TB Savage          4.4  .84   5   20
northeast     NE     AM Main Jr.        5.1  .94   3   13
north         NO     Margot Weber       4.5  .89   5   9
−−−>THE NORTH SALES DISTRICT HAS MOVED<−−−
central       CT     Ann Stephens       5.7  .94   5   13
```

EXPLANATION

The a command is the append command. The string −−−>THE NORTH SALES DISTRICT HAS MOVED<−−− is appended after lines beginning with the pattern north, when north is followed by a space. The text that will be appended must be on the line following the append command.

Sed requires a backslash after the a command. The second backslash is used by the C shell to escape the newline so that its closing quote can be on the next line.[4] If more than one line is appended, each line, except the last one, must also end in a backslash.

```
% cat datafile
northwest          NW   Charles Main       3.0   .98   3   34
western            WE   Sharon Gray        5.3   .97   5   23
southwest          SW   Lewis Dalsass      2.7   .8    2   18
southern           SO   Suan Chin          5.1   .95   4   15
southeast          SE   Patricia Hemenway  4.0   .7    4   17
eastern            EA   TB Savage          4.4   .84   5   20
northeast          NE   AM Main Jr.        5.1   .94   3   13
north              NO   Margot Weber       4.5   .89   5    9
```

```
central           CT   Ann Stephens       5.7   .94   5   13
```

4.6.9 Inserting: The i Command

Example 4.23

```
sed '/eastern/i\\
NEW ENGLAND REGION\\
-----------------------------------' datafile
northwest     NW     Charles Main       3.0  .98   3   34
western       WE     Sharon Gray        5.3  .97   5   23
southwest     SW     Lewis Dalsass      2.7  .8    2   18
southern      SO     Suan Chin          5.1  .95   4   15
southeast     SE     Patricia Hemenway  4.0  .7    4   17
NEW ENGLAND REGION
-----------------------------------
eastern       EA     TB Savage          4.4  .84   5   20
northeast     NE     AM Main Jr.        5.1  .94   3   13
north         NO     Margot Weber       4.5  .89   5   9
central       CT     Ann Stephens       5.7  .94   5   13
```

EXPLANATION

The i command is the insert command. If the pattern eastern is matched, the i command causes the text following the backslash to be inserted above the line containing eastern. A backslash is required after each line to be inserted, except the last one. (The extra backslash is for the C shell.)

4.6.10 Next: The n Command

Example 4.24

```
sed '/eastern/{ n; s/AM/Archie/; }' datafile
northwest     NW     Charles Main       3.0  .98   3   34
western       WE     Sharon Gray        5.3  .97   5   23
southwest     SW     Lewis Dalsass      2.7  .8    2   18
southern      SO     Suan Chin          5.1  .95   4   15
southeast     SE     Patricia Hemenway  4.0  .7    4   17
eastern       EA     TB Savage          4.4  .84   5   20
northeast     NE     Archie Main Jr.    5.1  .94   3   13
north         NO     Margot Weber       4.5  .89   5   9
central       CT     Ann Stephens       5.7  .94   5   13
```

EXPLANATION

If the pattern eastern is matched on a line, the n command causes sed to get the next line of input (the line with AM Main Jr.), replace the pattern space with this line, substitute (s) AM with Archie, print the line, and continue.

4.6.11 Transform: The y Command

Example 4.25

```
sed '1,3y/abcdefghijklmnopqrstuvwxyz/ABCDEFGHIJKL
 MNOPQRSTUVWXYZ/'  datafile
NORTHWEST     NW     CHARLES MAIN       3.0  .98   3   34
WESTERN       WE     SHARON GRAY        5.3  .97   5   23
SOUTHWEST     SW     LEWIS DALSASS      2.7  .8    2   18
southern      SO     Suan Chin          5.1  .95   4   15
southeast     SE     Patricia Hemenway  4.0  .7    4   17
eastern       EA     TB Savage          4.4  .84   5   20
northeast     NE     AM Main Jr.        5.1  .94   3   13
north         NO     Margot Weber       4.5  .89   5   9
central       CT     Ann Stephens       5.7  .94   5   13
```

EXPLANATION

For lines 1 through 3, the y command translates all lowercase letters to uppercase letters. Regular expression metacharacters do not work with this command.

```
% cat datafile
northwest           NW   Charles Main       3.0   .98   3   34
western             WE   Sharon Gray        5.3   .97   5   23
southwest           SW   Lewis Dalsass      2.7   .8    2   18
southern            SO   Suan Chin          5.1   .95   4   15
southeast           SE   Patricia Hemenway  4.0   .7    4   17
eastern             EA   TB Savage          4.4   .84   5   20
northeast           NE   AM Main Jr.        5.1   .94   3   13
north               NO   Margot Weber       4.5   .89   5   9
central             CT   Ann Stephens       5.7   .94   5   13
```

4.6.12 Quit: The q Command

Example 4.26

```
sed '5q' datafile
northwest     NW     Charles Main       3.0  .98   3   34
western       WE     Sharon Gray        5.3  .97   5   23
southwest     SW     Lewis Dalsass      2.7  .8    2   18
southern      SO     Suan Chin          5.1  .95   4   15
southeast     SE     Patricia Hemenway  4.0  .7    4   17
```

EXPLANATION

After the line 5 is printed, the q command causes the sed program to quit.

Example 4.27

```
sed  '/Lewis/{ s/Lewis/Joseph/;q; }' datafile
northwest     NW      Charles Main       3.0  .98   3   34
western       WE      Sharon Gray        5.3  .97   5   23
southwest     SW      Joseph Dalsass     2.7  .8    2   18
```

EXPLANATION

When the pattern Lewis is matched on a line, first the substitution command (s) replaces Lewis with Joseph, and then the q command causes the sed program to quit.

4.6.13 Holding and Getting: The h and g Commands

Example 4.28

```
sed -e '/northeast/h'  -e  '$G'  datafile
northwest      NW     Charles Main       3.0  .98   3   34
western        WE     Sharon Gray        5.3  .97   5   23
southwest      SW     Lewis Dalsass      2.7  .8    2   18
southern       SO     Suan Chin          5.1  .95   4   15
southeast      SE     Patricia Hemenway  4.0  .7    4   17
eastern        EA     TB Savage          4.4  .84   5   20
northeast      NE     AM Main Jr.        5.1  .94   3   13
north          NO     Margot Weber       4.5  .89   5   9
central        CT     Ann Stephens       5.7  .94   5   13
northeast      NE     AM Main Jr.        5.1  .94   3   13
```

EXPLANATION

As sed processes the file, each line is stored in a temporary buffer called the pattern space. Unless the line is deleted or suppressed from printing, the line will be printed to the screen after it is processed. The pattern space is then cleared and the next line of input is stored there for processing. In this example, after the line containing the pattern northeast is found, it is placed in the pattern space and the h command copies it and places it into another special buffer called the holding buffer. In the second sed instruction, when the last line is reached ($) the G command tells sed to get the line from the holding buffer and put it back in the pattern space buffer, appending it to the line that is currently stored there, in this case, the last line. Simply stated: Any line containing the pattern northeast will be copied and appended to the end of the file. (See Figure 4.1.)

Figure 4.1. The pattern space and holding buffer. See Example 4.31.

```
% cat datafile
northwest           NW   Charles Main       3.0   .98   3   34
western             WE   Sharon Gray        5.3   .97   5   23
southwest           SW   Lewis Dalsass      2.7   .8    2   18
southern            SO   Suan Chin          5.1   .95   4   15
southeast           SE   Patricia Hemenway  4.0   .7    4   17
eastern             EA   TB Savage          4.4   .84   5   20
northeast           NE   AM Main Jr.        5.1   .94   3   13
north               NO   Margot Weber       4.5   .89   5    9
central             CT   Ann Stephens       5.7   .94   5   13
```

Example 4.29

```
sed -e '/WE/{h; d; }' -e '/CT/{G; }' datafile
northwest      NW     Charles Main       3.0  .98   3   34
southwest      SW     Lewis Dalsass      2.7  .8    2   18
southern       SO     Suan Chin          5.1  .95   4   15
southeast      SE     Patricia Hemenway  4.0  .7    4   17
eastern        EA     TB Savage          4.4  .84   5   20
northeast      NE     AM Main Jr.        5.1  .94   3   13
north          NO     Margot Weber       4.5  .89   5    9
central        CT     Ann Stephens       5.7  .94   5   13
western        WE     Sharon Gray        5.3  .97   5   23
```

EXPLANATION

If the pattern WE is found on a line, the h command causes the line to be copied from the pattern space into a holding buffer. When stored in the holding buffer, the line can be retrieved (G or g command) at a later time. In this example, when the pattern WE is found, the line in which it was

found is stored in the pattern buffer first. The h command then puts a copy of the line in the holding buffer. The d command deletes the copy in the pattern buffer. The second command searches for CT in a line, and when it is found, sed gets (G) the line that was stored in the holding buffer and appends it to the line currently in the pattern space. Simply stated: The line containing WE is moved and appended after the line containing CT. (See "Holding and Exchanging: The h and x Commands".)

Example 4.30

```
sed -e '/northeast/h'  -e  '$g'  datafile
northwest     NW     Charles Main       3.0  .98   3   34
western       WE     Sharon Gray        5.3  .97   5   23
southwest     SW     Lewis Dalsass      2.7  .8    2   18
southern      SO     Suan Chin          5.1  .95   4   15
southeast     SE     Patricia Hemenway  4.0  .7    4   17
eastern       EA     TB Savage          4.4  .84   5   20
northeast     NE     AM Main Jr.        5.1  .94   3   13
north         NO     Margot Weber       4.5  .89   5    9
northeast     NE     AM Main Jr.        5.1  .94   3   13
```

EXPLANATION

As sed processes the file, each line is stored in a temporary buffer called the pattern space. Unless the line is deleted or suppressed from printing, the line will be printed to the screen after it is processed. The pattern space is then cleared and the next line of input is stored there for processing. In this example, after the line containing the pattern northeast is found, it is placed in the pattern space. The h command takes a copy of it and places it in another special buffer called the holding buffer. In the second sed instruction, when the last line is reached ($), the g command tells sed to get the line from the holding buffer and put it back in the pattern space buffer, replacing the line that is currently stored there, in this case, the last line. Simply stated: The line containing the pattern northeast is copied and moved to overwrite the last line in the file.

```
% cat datafile
northwest           NW   Charles Main       3.0   .98   3   34
western             WE   Sharon Gray        5.3   .97   5   23
southwest           SW   Lewis Dalsass      2.7   .8    2   18
southern            SO   Suan Chin          5.1   .95   4   15
southeast           SE   Patricia Hemenway  4.0   .7    4   17
eastern             EA   TB Savage          4.4   .84   5   20
northeast           NE   AM Main Jr.        5.1   .94   3   13
north               NO   Margot Weber       4.5   .89   5   9
central             CT   Ann Stephens       5.7   .94   5   13
```

Example 4.31

```
sed −e '/WE/{h; d; }' −e '/CT/{g; }' datafile
northwest     NW      Charles Main       3.0  .98   3   34
southwest     SW      Lewis Dalsass      2.7  .8    2   18
southern      SO      Suan Chin          5.1  .95   4   15
southeast     SE      Patricia Hemenway  4.0  .7    4   17
eastern       EA      TB Savage          4.4  .84   5   20
northeast     NE      AM Main Jr.        5.1  .94   3   13
north         NO      Margot Weber       4.5  .89   5   9
western       WE      Sharon Gray        5.3  .97   5   23
```

EXPLANATION

If the pattern WE is found, the h command copies the line into the holding buffer; the d command deletes the line in the pattern space. When the pattern CT is found, the g command gets the copy in the holding buffer and overwrites the line currently in the pattern space. Simply stated: Any line containing the pattern WE will be moved to overwrite lines containing CT. (See Figure 4.1.)

4.6.14 Holding and Exchanging: The h and x Commands

Example 4.32

```
sed −e '/Patricia/h'  −e '/Margot/x' datafile
northwest     NW      Charles Main       3.0  .98   3   34
western       WE      Sharon Gray        5.3  .97   5   23
southwest     SW      Lewis Dalsass      2.7  .8    2   18
southern      SO      Suan Chin          5.1  .95   4   15
southeast     SE      Patricia Hemenway  4.0  .7    4   17
eastern       EA      TB Savage          4.4  .84   5   20
northeast     NE      AM Main Jr.        5.1  .94   3   13
southeast     SE      Patricia Hemenway  4.0  .7    4   17
central       CT      Ann Stephens       5.7  .94   5   13
```

EXPLANATION

The x command exchanges (swaps) the contents of the holding buffer with the current pattern space. When the line containing the pattern Patricia is found, it will be stored in the holding buffer. When the line containing Margot is found, the pattern space will be exchanged for the line in the holding buffer. Simply stated: The line containing Margot will be replaced with the line containing Patricia.

4.7 sed Scripting

A sed script is a list of sed commands in a file. To let sed know your commands are in a file, when invoking sed at the command line, use the −f option followed by the name of the sed script. Sed is very particular about the way you type lines into the script. There cannot be any trailing whitespace or text at the end of the command. If commands are not placed on a line by themselves, they must be terminated with a semicolon. A line from the input file is copied into the pattern buffer, and all commands in the sed script are executed on that line. After the line has been processed, the next line from the input file is placed in the pattern buffer, and all commands in the script are executed on that line. Sed gets "garbled" if your syntax is incorrect.

The nice thing about sed scripts is that you don't have to worry about the shell's interaction as you do when at the command line. Quotes are not needed to protect sed commands from interpretation by the shell. In fact, you cannot use quotes in a sed script at all, unless they are part of a search pattern.

```
% cat datafile
northwest           NW   Charles Main       3.0   .98   3   34
western             WE   Sharon Gray        5.3   .97   5   23
southwest           SW   Lewis Dalsass      2.7   .8    2   18
southern            SO   Suan Chin          5.1   .95   4   15
southeast           SE   Patricia Hemenway  4.0   .7    4   17
eastern             EA   TB Savage          4.4   .84   5   20
northeast           NE   AM Main Jr.        5.1   .94   3   13
north               NO   Margot Weber       4.5   .89   5    9
central             CT   Ann Stephens       5.7   .94   5   13
```

4.7.1 sed Script Examples

Example 4.33

```
% cat sedding1 (Look at the contents of the sed script.)
1  # My first sed script by Jack Sprat
2  /Lewis/a\
3     Lewis is the TOP Salesperson for April!!\
      Lewis is moving to the southern district next month.\
4     CONGRATULATIONS!
5  /Margot/c\
      ******************\
      MARGOT HAS RETIRED\
      ******************
6  1i\
   EMPLOYEE DATABASE\
   ---------------------
7  $d

% sed −f sedding1 datafile   (Execute the sed script commands; the
                               input file is datafile.)
EMPLOYEE DATABASE
---------------------
northwest     NW     Charles Main       3.0  .98   3   34
western       WE     Sharon Gray        5.3  .97   5   23
southwest     SW     Lewis Dalsass      2.7  .8    2   18
Lewis is the TOP Salesperson for April!!
Lewis is moving to the southern district next month
CONGRATULATIONS!
southern      SO     Suan Chin          5.1  .95   4   15
southeast     SE     Patricia Hemenway  4.0  .7    4   17
eastern       EA     TB Savage          4.4  .84   5   20
northeast     NE     AM Main Jr.        5.1  .94   3   13
******************
MARGOT HAS RETIRED
*****************
```

EXPLANATION

1. This line is a comment. Comments must be on lines by themselves and start with a pound sign (#).
2. If a line contains the pattern Lewis, the next three lines are appended to that line.
3. Each line being appended, except the last one, is terminated with a backslash. The backslash must be followed immediately with a newline. If there is any trailing text, even one space after the newline, sed will complain.
4. The last line to be appended does not have the terminating backslash. This indicates to sed that this is the last line to be appended and that the next line is another command.
5. Any lines containing the pattern Margot will be replaced (c command) with the next three lines of text.
6. The next two lines will be inserted (i command) above line 1.
7. The last line ($) will be deleted.

Example 4.34

```
%  cat sedding2 (Look at the contents of the sed script.)
# This script demonstrates the use of curly braces to nest addresses
```

```
# and commands. Comments are preceded by a pound sign(#) and must
# be on a line by themselves. Commands are terminated with a newline
# or semicolon.  If there is any text after a command, even one
# space, you receive an error message:
#      sed: Extra text at end of command:

1  /western/, /southeast/{
      /^ *$/d
      /Suan/{ h; d; }
   }
2   /Ann/g
3   s/TB \(Savage\)/Thomas \1/

% sed −f sedding2 datafile
northwest     NW     Charles Main       3.0  .98   3   34
western       WE     Sharon Gray        5.3  .97   5   23
southwest     SW     Lewis Dalsass      2.7  .8    2   18
southeast     SE     Patricia Hemenway  4.0  .7    4   17
eastern       EA     Thomas Savage      4.4  .84   5   20
northeast     NE     AM Main Jr.        5.1  .94   3   13
north         NO     Margot Weber       4.5  .89   5    9
southern      SO     Suan Chin          5.1  .95   4   15
```

EXPLANATION

1. In the range of lines starting at western and ending at southeast, blank lines are deleted, and lines matching Suan are copied from the pattern buffer into the holding buffer, then deleted from the pattern buffer.
2. When the pattern Ann is matched, the g command copies the line in the holding buffer to the pattern buffer, overwriting what is in the pattern buffer.
3. All lines containing the pattern TB Savage are replaced with Thomas and the pattern that was tagged, Savage. In the search string, Savage is enclosed in escaped parentheses, tagging the enclosed string so that it can be used again. It is tag number 1, referenced by \1.

4.7.2 Review

Table 4.4 lists sed commands and what they do.

Table 4.4. sed Review

CommandWhat It Doessed −n '/sentimental/p' filexPrints to the screen all lines containing sentimental. The file filex does not change. Without the −n option, all lines with sentimental will be printed twice.sed '1,3d' filex > newfilexDeletes lines 1, 2, and 3 from filex and saves changes in newfilex.sed '/[Dd]aniel/d' filexDeletes lines containing Daniel or daniel.sed −n '15,20p' filexPrints only lines 15 through 20.sed '1,10s/Montana/MT/g' filexSubstitutes Montana with MT globally in lines 1 through 10.

```
sed '/March/\!d' filex    (csh)
sed '/March/!d' filex      (sh)
```

Deletes all lines not containing March. (The backslash is used only in the csh to escape the history character.)sed '/report/s/5/8/' filexChanges the first occurrence of 5 to 8 on all lines containing report.sed 's/…./' filexDeletes the first four characters of each line.sed 's/…$//' filexDeletes the last three characters of

each line.sed '/east/,/west/s/North/South/' filexFor any lines falling in the range from east to west, substitutes North with South.sed −n '/Time off/w timefile' filexWrites all lines containing Time off to the file timefile.sed 's/\([Oo]ccur\)ence/\1rence/' fileSubstitutes either Occurence or occurence with Occurrence or occurrence.sed −n 'l' filexPrints all lines showing nonprinting characters as \nn, where nn is the octal value of the character, and showing tabs as >.

UNIX TOOLS LAB EXERCISE

Lab 2: sed Exercise

Steve Blenheim:238−923−7366:95 Latham Lane, Easton, PA 83755:11/12/56:20300

Betty Boop:245−836−8357:635 Cutesy Lane, Hollywood, CA 91464:6/23/23:14500

Igor Chevsky:385−375−8395:3567 Populus Place, Caldwell, NJ 23875:6/18/68:23400

Norma Corder:397−857−2735:74 Pine Street, Dearborn, MI 23874:3/28/45:245700

Jennifer Cowan:548−834−2348:583 Laurel Ave., Kingsville, TX 83745:10/1/35:58900

Jon DeLoach:408−253−3122:123 Park St., San Jose, CA 04086:7/25/53:85100

Karen Evich:284−758−2857:23 Edgecliff Place, Lincoln, NB 92743:7/25/53:85100

Karen Evich:284−758−2867:23 Edgecliff Place, Lincoln, NB 92743:11/3/35:58200

Karen Evich:284−758−2867:23 Edgecliff Place, Lincoln, NB 92743:11/3/35:58200

Fred Fardbarkle:674−843−1385:20 Parak Lane, DeLuth, MN 23850:4/12/23:780900

Fred Fardbarkle:674−843−1385:20 Parak Lane, DeLuth, MN 23850:4/12/23:780900

Lori Gortz:327−832−5728:3465 Mirlo Street, Peabody, MA 34756:10/2/65:35200

Paco Gutierrez:835−365−1284:454 Easy Street, Decatur, IL 75732:2/28/53:123500

Ephram Hardy:293−259−5395:235 CarltonLane, Joliet, IL 73858:8/12/20:56700

James Ikeda:834−938−8376:23445 Aster Ave., Allentown, NJ 83745:12/1/38:45000

Barbara Kertz:385−573−8326:832 Ponce Drive, Gary, IN 83756:12/1/46:268500

Lesley Kirstin:408−456−1234:4 Harvard Square, Boston, MA 02133:4/22/62:52600

William Kopf:846−836−2837:6937 Ware Road, Milton, PA 93756:9/21/46:43500

Sir Lancelot:837−835−8257:474 Camelot Boulevard, Bath, WY 28356:5/13/69:24500

Jesse Neal:408−233−8971:45 Rose Terrace, San Francisco, CA 92303:2/3/36:25000

Zippy Pinhead:834−823−8319:2356 Bizarro Ave., Farmount, IL 84357:1/1/67:89500

Arthur Putie:923−835−8745:23 Wimp Lane, Kensington, DL 38758:8/31/69:126000

Popeye Sailor:156−454−3322:945 Bluto Street, Anywhere, USA 29358:3/19/35:22350

Jose Santiago:385−898−8357:38 Fife Way, Abilene, TX 39673:1/5/58:95600

Tommy Savage:408−724−0140:1222 Oxbow Court, Sunnyvale, CA 94087:5/19/66:34200

Yukio Takeshida:387−827−1095:13 Uno Lane, Ashville, NC 23556:7/1/29:57000

Vinh Tranh:438−910−7449:8235 Maple Street, Wilmington, VM 29085:9/23/63:68900

(Refer to the database called datebook on the CD.)

1: Change Jon's name to Jonathan.

2: Delete the first three lines.

3: Print lines 5 through 10.

4: Delete lines containing Lane.

5: Print all lines where the birthdays are in November or December.

6: Append three stars to the end of lines starting with Fred.

7: Replace the line containing Jose with JOSE HAS RETIRED.

8: Change Popeye's birthday to 11/14/46.

9: Delete all blank lines.

10: Write a sed script that will:

a. Insert above the first line the title PERSONNEL FILE.
b. Remove the salaries ending in 500.
c. Print the contents of the file with the last names and first names reversed.
d. Append at the end of the file THE END.

[1] Remember, the shell will try to evaluate any metacharacters or whitespace when a command is typed at the command line; any characters in the sed command that could be interpreted by the shell must be quoted.

[2] For a complete list of diagnostics, see the UNIX man page for sed.

[3] To represent a literal ampersand in the replacement string, it must be escaped: \&.

[4] The Bourne and Korn shells do not require the second backslash to escape the newline because they do not require quotes to be matched on the same line, only that they match.

CONTENTS