7

Answers to Even-numbered Exercises

2. Which command moves the cursor to the end of the current paragraph? Can you use this command to skip through the buffer in one-paragraph steps?

   The META-} command moves the cursor to the end of the current paragraph. You can use it repeatedly to move through the buffer by paragraphs.

4. After you have been working on a paragraph for a while, most likely some lines will have become too short and others too long. Is there a command to "neaten up" the paragraph without rebreaking all the lines by hand?

   Place the cursor at the beginning of the paragraph and press META-q. Alternatively, you can give the command META-x **fill-paragraph**. You can also give the command META-x **mark-paragraph** (or META-h) while the cursor is on the paragraph you want to reformat, and then give the command META-x **fill-region**.

6. How would you reverse the order of two paragraphs?

   With the cursor on the first paragraph, use META-h to define the paragraph as Region, kill Region with META-x **kill-paragraph**, use META-} to move the cursor to the end of the second paragraph, and use CONTROL-Y to yank the killed paragraph.

8. Imagine that you saw a Usenet posting with something particularly funny in it and saved the posting to a file. How would you incorporate this file into your own buffer? What if you wanted to use only a couple of paragraphs from the posting? How would you add **>** to the beginning of each included line?

   To read a file into the current buffer, move the cursor to where you want to add the file and give the command CONTROL-X **i** (insert-file).

   There are several ways to add only a few paragraphs from a file: You can read the entire file and delete the parts you do not want. Alternatively, you can read the file into its own buffer with CONTROL-X CONTROL-F, kill the part you want, go back to the original buffer with CONTROL-X **b** (switch to buffer), and yank the killed text with CONTROL-Y.

   To place a greater-than sign followed by a SPACE at the beginning of each line of the new text, place the cursor at the beginning of the new text, give the command META-x **query-replace-regexp** ^RETURN >SPACE RETURN, and respond with SPACE to each prompt until you get to the end of the new text; then respond with RETURN.

10. Assume that your buffer contains the C code shown here, with the Major mode set for C and the cursor positioned at the end of the **while** line as shown by the black square:

```
/*
 * Copy string s2 to s1.  s1 must be large enough
 * return s1
 */
char *strcpy(char *s1, char *s2)
{
        char *os1;

        os1 = s1;
        while (*s1++ = *s2++)
                ;
        return os1;
}

/*
 * Copy source into dest, stopping after '\0' is copied, and
 * return a pointer to the '\0' at the end of dest. Then our caller
 * can catenate to the dest  * string without another strlen call.
 */
char *stpcpy (char *dest, char *source)
{
        while ((*dest++ = *source++) != '\0') ■
                ; /* void loop body */
        return (dest - 1);
}
```

a. Which command moves the cursor to the opening brace of **strcpy**? Which command moves the cursor past the closing brace? Can you use these commands to skip through the buffer in one-procedure steps?

With the cursor at the start of the file, give the command CONTROL-META-e to move the cursor past the closing brace of **strcpy**; CONTROL-META-a moves it back to the opening brace. You can use these commands to skip from procedure to procedure.

b. Assume the cursor is just past the closing parenthesis of the **while** condition. How do you move to the matching opening parenthesis? How do you move back to the matching close parenthesis again? Does the same command set work for matched [] (square brackets) and {} (braces)? How does this differ from the vim **%** command?

CONTROL-META-b moves backward over an expression and CONTROL-META-f moves forward over an expression. The expression can be delimited by (), [], or {}.

The vim **%** command requires that you position the cursor on the same line as, and on or to the left of, the closing element of the expression. Then **%** jumps between the opening and closing elements.

c. One procedure is indented in the Berkeley indention style; the other is indented in the GNU style. Which command reindents a line in accordance with the current indention style you have set up? How would you reindent an entire procedure?

Press TAB while the cursor is positioned anywhere on a line to reindent the line to the current indention style. Position the cursor before a pair of matched braces and press CONTROL-META-q to reindent the lines within the braces to the current style.

d. Suppose that you want to write five string procedures and intend to use **strcpy** as a starting point for further editing. How would you make five copies of the **strcpy** procedure?

Move the cursor to the beginning of the word **strcpy** and press CONTROL-SPACE to set Mark. Move the cursor to the line past the closing brace and press META-w to copy Region nondestructively to the Kill Ring. Finally, press CONTROL-Y five times to yank five copies of the killed Region into the Work buffer.

e. How would you compile the code without leaving emacs?

After saving the buffer, give the command META-x **compile**. You will be prompted for a command; respond with the command to compile the file you are working on. The output of the compilation appears in a buffer named ***compilation***.