Answers to Even-Numbered Exercises 10

from page 469

1. How can you cause vim to enter Input mode? How can you make vim revert to Command mode?

2. What is the Work buffer? Name two ways of writing the contents of the Work buffer to the disk.

   The Work buffer is the area of memory where vim stores the text you are editing. A **:w** command writes the contents of the work buffer to disk but does not end your editing session. A **ZZ** command writes the contents of the work buffer to disk and ends your editing session.

3. Suppose that you are editing a file that contains the following paragraph and the cursor is on the second tilde (~):

   ```
   The vim editor has a command, tilde (~),
   that changes lowercase letters to
   uppercase and vice versa.
   The ~ command works with a Unit of Measure or
   a Repeat Factor, so you can change
   the case of more than one character at a time.
   ```

   How can you

   a. Move the cursor to the end of the paragraph?

   b. Move the cursor to the beginning of the word `Unit`?

   c. Change the word `character` to `letter`?

4. In vim, with the cursor positioned on the first letter of a word, give the command **x** followed by **p**. Explain what happens.

   The commands exchange the first two letters of the word. First the **x** command copies the character the cursor is on to the General-Purpose buffer and deletes the character, leaving the cursor on the character to the right of where the deleted character was. Then the **p** command inserts the contents of the General-Purpose buffer after the character the cursor is on.

5. What are the differences between the following commands?

   a. **i** and **I**

   b. **a** and **A**

   c. **o** and **O**

   d. **r** and **R**

   e. **u** and U

6. What command would you use to search backward through the Work buffer for lines that start with the word `it`?

   Give the command **?^it**RETURN to search backward (?) for a line beginning with (^) `it`.

7. What command substitutes all occurrences of the phrase `this week` with the phrase `next week`?

8. Consider the following scenario: You start vim to edit an existing file. You make many changes to the file and then realize that you deleted a critical section of the file early in your editing session. You want to get that section back but do not want to lose all the other changes you made. What would you do?

   This problem assumes that you have not written out the Work buffer since you deleted the critical section. There are a few ways to approach this problem. To be safe, make copies of the Work buffer and the original file under names other than the name of the original file. That way, if you make a mistake, you can easily start over. For example, give the command **:wq changedfile** to save the work buffer as **changedfile**, and exit from vim. Then use cp to copy the original file to, for example, **file.orig** and **changedfile** to **changedfile.orig**. Start vim with the following command, which instructs it to edit the original file first and the modified file second:

   ```
   $ vim originalfile changedfile
   ```

Once you are editing the original file, search for and copy the part of the file you want to save into a Named buffer. For example, to save five lines, starting with the line the cursor is on, into the Named buffer **a**, give the command **"a5yy**. Then edit the modified file by giving the command **:n!**RETURN (edit the next file without writing out the Work buffer). Position the cursor where you want to insert the text, and give the command **"ap** or **"aP**, depending on where you want to place the copied text.

9. Consider the following scenario: Alex puts the following line in his **.login** file:

```
setenv EXINIT 'set number wrapmargin=10 showmode'
```

Then Alex creates a **.vimrc** file in the directory **/home/alex/literature** with the following line in it:

```
set nonumber
```

What will the parameter settings be when Alex runs vim while the working directory is **/home/alex/bin**? What will they be when he runs vim from the directory **/home/alex/literature**? What will they be when he edits the file **/home/alex/literature/promo**?

10. Use vim to create the **letter_e** file of e's used on page 73. Use as few vim commands as possible. What vim commands did you use?

72ieESCAPE**yy7999p**

Advanced Exercises

11. What commands can you use to take a paragraph from one file and insert it in a second file?

12. Create a file that contains the following list, and then execute commands from within vim to sort the list and display it in two columns. (*Hint:* Refer to page 1286 in Part III for more information on pr.)

```
Command mode
Input mode
Last Line mode
Work buffer
General-Purpose buffer
Named buffer
Regular Expression
Search String
```

```
Replacement String
Startup File
Repeat Factor
```

Sort the list by placing the cursor on the first character of the Work buffer and giving the command **!Gsort**. Then you can use pr to display the file in two columns (the **–t** option causes pr not to print a header and trailer). Place the cursor on the first character of the file, and give the command **!Gpr –2 –t**.

13. How do the Named buffers differ from the General-Purpose buffer?

14. Assuming that your version of vim does not support multiple Undo commands, if you delete a line of text and then delete another line and then a third line, what commands would you use to recover the first two lines that you deleted?

    "2p and **"3p** (or **"2P** and **"3P**)

15. What command would you use to swap the words hither and yon on any line with any number of words between them? (You need not worry about special punctuation, just upper- and lowercase letters and spaces.)