6

Answers to Even-numbered Exercises

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.

4. While working in vim, with the cursor positioned on the first letter of a word, you 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.

6. Which 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**.

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. If you then 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 the General-Purpose buffer or 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.

10. Use vim to create the **letter_e** file of e's used on page 54. Use as few vim commands as possible. Which vim commands did you use?

    72ieESCAPE**yy7999p**

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 817 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**. Next 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**.

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

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