Answers to Even-numbered Exercises

7

2. Using sort as a filter, rewrite the following sequence of commands:

```
$ sort list > temp
$ lpr temp
$ rm temp
```

```
$ cat list | sort | lpr
```

4. Assume that the following files are in the working directory:

```
$ ls
intro     notesb    ref2      section1   section3   section4b
notesa    ref1      ref3      section2   section4a  sentrev
```

Give commands for each of the following, using wildcards to express filenames with as few characters as possible.

a. List all files that begin with **section**.

```
$ ls section*
```

b. List the **section1**, **section2**, and **section3** files only.

```
$ ls section[1-3]
```

c. List the **intro** file only.

```
$ ls i*
```

d. List the **section1**, **section3**, **ref1**, and **ref3** files.

```
$ ls *[13]
```

6. Give a command to

a. Redirect the standard output from a sort command into a file named
phone_list. Assume that the input file is named **numbers**.

```
$ sort numbers > phone_list
```

b. Translate all occurrences of characters [and { to the character (, and
translate all occurrences of the characters] and } to the character) in the
file **permdemos.c**. (*Hint:* Refer to the tr man page.)

```
$ cat permdemos.c | tr '[{}]' '(())' or
$ tr '[{}]' '(())' < permdemos.c
```

c. Create a file named **book** that contains the contents of two other files:
part1 and **part2**.

```
$ cat part[12] > book
```

8. Give an example of a command that uses grep

a. With both input and output redirected.

```
$ grep \$Id < *.c > id_list
```

b. With only input redirected.

```
$ grep -i suzi < addresses
```

c. With only output redirected.

```
$ grep -il memo *.txt > memoranda_files
```

d. Within a pipe.

```
$ file /usr/bin/* | grep "Again shell script" | sort -r
```

In which of the preceding is grep used as a filter?

Example **d** uses grep as a filter.

10. When you use the redirect output symbol (**>**) with a command, the shell
creates the output file immediately, before the command is executed.
Demonstrate that this is true.

```
$ ls aaa
ls: aaa: No such file or directory
$ ls xxxxx > aaa
ls: xxxxx: No such file or directory
$ ls aaa
aaa
```

The first of the preceding commands shows that the file **aaa** does not exist
in the working directory. The next command uses ls to attempt to list a
nonexistent file (**xxxxx**) and sends the standard output to **aaa**. The ls
command fails and sends an error message to standard error (you see it on
the screen). Even though the ls command failed, the empty file, named **aaa**,
exists. Because the ls command failed, it did not create the file; the shell
created it before calling ls.

12. Assume that your permissions allow you to write to a file but not to delete it.

a. Give a command to empty the file without invoking an editor.

```
$ filename < /dev/null or
$ cat /dev/null > filename
```

b. Explain how you might have permission to modify a file that you cannot delete.

To delete a file, you must have write and execute permissions to the directory housing the file. To write to a file, you must have write permission to the file and execute permission to the parent directory. When you have write permission only to a file and execute permission only to the directory the file is in, you can modify, but not delete, the file.

14. Why can the **noclobber** variable *not* protect you from overwriting an existing file with cp or mv?

The **noclobber** variable keeps the shell from overwriting a file and does not work on utilities. Thus the **noclobber** variable keeps a redirect symbol (>) from allowing the shell to overwrite a file (the shell redirects output) but has no affect when you ask cp or mv to overwrite a file.

16. Create a file named **answers**, and give the following command:

```
$ > answers.0102 < answers cat
```

Explain what the command does and why. What is a more conventional way of expressing this command?

Reading the command line from left to right, it instructs the shell to redirect standard output to **answers.0102**, redirect standard input to come from **answers**, and execute the cat utility. More conventionally, the same command is expressed as

```
$ cat answers > answers.0102
```