# 5

## Answers to Even-numbered Exercises

2. How can you keep other users from using write to communicate with you? Why would you want to?

   Give the command **mesg n** to keep ordinary users from writing to your terminal. You can take this action when you do not want to be disturbed or when you are viewing something on your screen that you do not want overwritten.

4. How can you find out which utilities are available on your system for editing files? Which utilities are available for editing on your system?

   Give the command **apropos editor**. Typical editors are vim, ex, ed, and gedit.

6. What happens when you use diff to compare two binary files that are not identical? (You can use gzip to create the binary files.) Explain why the diff output for binary files is different from the diff output for ASCII files.

   When you compare binary files with diff, the utility displays a message saying the files differ when the files differ or no message when the files are the same. The diff utility compares ASCII files on a line-by-line basis; it is not designed to compare binary files on a byte-by-byte basis. Use cmp to compare binary files in that manner.

8. What is the result of giving the which utility the name of a command that resides in a directory that is *not* in your search path?

   The which utility displays a message saying that the command you are looking for is not in the list of directories in your search path. For example,

```
$ which me
/usr/bin/which: no me in (/usr/bin/which: no me in
(/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/home/sam/bin)
```

10. Experiment by calling the file utility with the names of files in **/usr/bin**. How many different types of files are there?

    Approximately 20.

12. Re-create the **colors.1** and **colors.2** files used in Figure 5-8 on page 147. Test your files by running **diff –u** on them. Do you get the same results as in the figure?

    ```
    $ cat colors.1
    red
    green
    yellow
    pink
    purple
    orange

    $ cat colors.2
    red
    blue
    green
    yellow
    orange
    ```

14. Repeat exercise 5 using the file **phone.gz**, a compressed version of the list of names and phone numbers. Consider more than one approach to answer each question, and explain how you made your choices.

    You can either decompress the file using gunzip, giving the same commands as exercise 5 used once the file is decompressed, or use zcat and a pipe to display the results without creating an intermediate file as shown here:

    ```
      $ zcat phone.gz | grep "Ace Electronics"
      $ zcat phone.gz | sort
      $ zcat phone.gz | uniq
      $ zcat phone.gz | sort -u
    ```

    Which technique you use makes a significant difference only if **phone.gz** is large, in which case it becomes an issue of what you are doing and making a tradeoff between using more CPU (processor) time and less hard disk storage, or vice versa.

    When you are giving a single command one time only, using a pipe is more efficient. When you want to give more than one command or want to give one command repeatedly, it is more efficient to decompress the file once using gunzip and then process it repeatedly with grep, sort, or uniq, assuming you have sufficient disk space. The most inefficient technique as far as disk space goes and the most efficient as far as CPU (and your) time goes is to put the output of grep, sort, or uniq in new files using a redirect output symbol:

    ```
      $ zcat phone.gz | sort > phone.sort
    ```

16. Older email programs were not able to handle binary files. Suppose that you are emailing a file that has been compressed with gzip, which produces a binary file, and the recipient is using an old email program. Refer to the man page on uuencode, which converts a binary file to ASCII. Learn about the utility and how to use it.

a. Convert a compressed file to ASCII using uuencode. Is the encoded file larger or smaller than the compressed file? Explain. (If uuencode is not on the local system, you can install it using yum [page 490]; it is part of the **sharutils** package.)

The following command converts the file **memo.gz** to ASCII using uuencode. The **.uuencode** filename extension is not required.

```
$ uuencode memo.gz > memo.gz.uuencode
```

The resulting ASCII file is larger than the original binary file because uuencode includes control information.

b. Would it ever make sense to use uuencode on a file before compressing it? Explain.

No purpose is served by using uuencode to convert a binary file to ASCII before compressing it because compressing an ASCII file creates a binary file.