grep options let you modify the search. Table 5-1 lists some of the options.

Table 5-1. Some grep options

Option	Description
-v	Print all lines that do not match pattern.
-n	Print the matched line and its line number.
-l	Print only the names of files with matching lines (lowercase letter "L").
-c	Print only the count of matching lines.
-i	Match either upper- or lowercase.

Next, let's use a regular expression that tells **grep** to find lines with *carol*, followed by zero or more other characters (abbreviated in a regular expression as "`.*`"),* then followed by *Aug*:

```
$ ls -l | grep "carol.*Aug"
-rw-rw-r--   1 carol doc      1605 Aug 23 07:35 macros
$
```

For more about regular expressions, see the references in the section "Documentation" (Chapter 8).

sort

The **sort** program arranges lines of text alphabetically or numerically. The following example sorts the lines in the *food* file (from the section "Printing Files" in Chapter 4) alphabetically. **sort** doesn't modify the file itself; it reads the file and writes the sorted text to the standard output.

```
$ sort food
Afghani Cuisine
Bangkok Wok
Big Apple Deli
Isle of Java
Mandalay
Sushi and Sashimi
Sweet Tooth
Tio Pepe's Peppers
```

* Note that the regular expression for "zero or more characters," "`.*`", is different than the corresponding filename wildcard "`*`". See the section "File and Directory Wildcards" in Chapter 4. We can't cover regular expressions in enough depth here to explain the difference—though more-detailed books do. As a rule of thumb, remember that the first argument to **grep** is a regular expression; other arguments, if any, are filenames that can use wildcards.