relational operators to compare strings. Instead, you can go to the C-style string library and use the `strcmp()` function to compare strings. This function takes two string addresses as arguments. That means the arguments can be pointers, string constants, or character array names. If the two strings are identical, the function returns the value zero. If the first string precedes the second alphabetically, `strcmp()` returns a negative value, and if the first string follows the second alphabetically, `strcmp()` returns a positive value. Actually, "in the system collating sequence" is more accurate than "alphabetically." This means that characters are compared according to the system code for characters. For example, in ASCII code, all uppercase letters have smaller codes than the lowercase letters, so uppercase precedes lowercase in the collating sequence. Therefore, the string "Zoo" precedes the string "aviary". The fact that comparisons are based on code values also means that uppercase and lowercase letters differ, so the string "FOO" is different from the "foo" string.

In some languages, such as BASIC and standard Pascal, strings stored in differently sized arrays are necessarily unequal to each other. But C-style strings are defined by the terminating null character, not by the size of the containing array. This means that two strings can be identical even if they are contained in differently sized arrays:

```c
char big[80] = "Daffy"; // 5 letters plus \0
char little[6] = "Daffy"; // 5 letters plus \0
```

By the way, although you can't use relational operators to compare strings, you can use them to compare characters, because characters actually are integer types. So,

```c
for (ch = 'a'; ch <= 'z'; ch++)
    cout << ch;
```

is valid code, at least for the ASCII character set, for displaying the characters of the alphabet.

Listing 5.11 uses `strcmp()` in the test condition of a `for` loop. The program displays a word, changes its first letter, displays the word again, and keeps going until `strcmp()` determines the word is the same as the string "mate". Note that the listing includes the `cstring` file because it provides a function prototype for `strcmp()`.

**Listing 5.11 compstr.cpp**