The comparison functions fall into two groups. Functions in the first group (memcmp, strcmp, and strncmp) compare the contents of two character arrays. Functions in the second group (strcoll and strxfrm) are used if the locale needs to be taken into account.

locales ➤25.1

memcmp
strcmp
strncmp

The memcmp, strcmp, and strncmp functions have much in common. All three expect to be passed pointers to character arrays. The characters in the first array are then compared one by one with the characters in the second array. All three functions return as soon as a mismatch is found. Also, all three return a negative, zero, or positive integer, depending on whether the stopping character in the first array was less than, equal to, or greater than the stopping character in the second.

The differences among the three functions have to do with when to stop comparing characters if no mismatch is found. The memcmp function is passed a third argument, n, that limits the number of comparisons performed; it pays no particular attention to null characters. strcmp doesn't have a preset limit, stopping instead when it reaches a null character in either array. (As a result, strcmp works only with null-terminated strings.) strncmp is a blend of memcmp and strcmp; it stops when n comparisons have been performed or a null character is reached in either array.

The following examples illustrate memcmp, strcmp, and strncmp:

```
char s1[] = {'b', 'i', 'g', '\0', 'c', 'a', 'r'};
char s2[] = {'b', 'i', 'g', '\0', 'c', 'a', 't'};

if (memcmp(s1, s2, 3) == 0) ...     /* true  */
if (memcmp(s1, s2, 4) == 0) ...     /* true  */
if (memcmp(s1, s2, 7) == 0) ...     /* false */

if (strcmp(s1, s2) == 0) ...        /* true  */

if (strncmp(s1, s2, 3) == 0) ...    /* true  */
if (strncmp(s1, s2, 4) == 0) ...    /* true  */
if (strncmp(s1, s2, 7) == 0) ...    /* true  */
```

strcoll

The strcoll function is similar to strcmp, but the outcome of the comparison depends on the current locale.

strxfrm

Most of the time, strcoll is fine for performing a locale-dependent string comparison. Occasionally, however, we might need to perform the comparison more than once (a potential problem, since strcoll isn't especially fast) or change the locale without affecting the outcome of the comparison. In these situations, the strxfrm ("string transform") function is available as an alternative to strcoll.

strxfrm transforms its second argument (a string), placing the result in the array pointed to by the first argument. The third argument limits the number of characters written to the array, including the terminating null character. Calling strcmp with two transformed strings should produce the same outcome (negative, zero, or positive) as calling strcoll with the original strings.