We saw in Section 23.6 that `strtok` searches a string for a "token"—a sequence of characters that doesn't include certain delimiting characters. The call `strtok(s1, s2)` scans the `s1` string for a nonempty sequence of characters that are *not* in the `s2` string. `strtok` marks the end of the token by storing a null character in `s1` just after the last character in the token; it then returns a pointer to the first character in the token.

Later calls of `strtok` can find additional tokens in the same string. The call `strtok(NULL, s2)` continues the search begun by the previous `strtok` call. As before, `strtok` marks the end of the token with a null character, and then returns a pointer to the beginning of the token. The process can be repeated until `strtok` returns a null pointer, indicating that no token was found.

One problem with `strtok` is that it uses a static variable to keep track of a search, which makes it impossible to use `strtok` to conduct simultaneous searches on two or more strings. Thanks to its extra parameter, `wcstok` doesn't have this problem.

The first two parameters to `wcstok` are the same as for `strtok` (except that they point to wide strings, of course). The third parameter, `ptr`, will point to a variable of type `wchar_t *`. The function will save information in this variable that enables later calls of `wcstok` to continue scanning the same string (when the first argument is a null pointer). When the search is resumed by a subsequent call of `wcstok`, a pointer to the same variable should be supplied as the third argument; the value of this variable must not be changed between calls of `wcstok`.

To see how `wcstok` works, let's redo the example of Section 23.6. Assume that `str`, `p`, and `q` are declared as follows:

```
wchar_t str[] = L" April  28,1998";
wchar_t *p, *q;
```

Our initial call of `wcstok` will pass `str` as the first argument:

```
p = wcstok(str, L" \t", &q);
```

`p` now points to the first character in `April`, which is followed by a null wide character. Calling `wcstok` with a null pointer as its first argument and `&q` as the third argument causes it to resume the search from where it left off:

```
p = wcstok(NULL, L" \t,", &q);
```

After this call, `p` points to the first character in `28`, which is now terminated by a null wide character. A final call of `wcstok` locates the year:

```
p = wcstok(NULL, L" \t", &q);
```

`p` now points to the first character in `1998`.

Miscellaneous Functions

```
size_t wcslen(const wchar_t *s);
wchar_t *wmemset(wchar_t *s, wchar_t c, size_t n);
```