- *Conversion specifier.* The conversion specifier must be one of the characters listed in Table 22.12.

Table 22.12
Conversion Specifiers for ...scanf Functions

Conversion Specifier	*Meaning*
d	Matches a decimal integer; the corresponding argument is assumed to have type int *.
i	Matches an integer; the corresponding argument is assumed to have type int *. The integer is assumed to be in base 10 unless it begins with 0 (indicating octal) or with 0x or 0X (hexadecimal).
o	Matches an octal integer; the corresponding argument is assumed to have type unsigned int *.
u	Matches a decimal integer; the corresponding argument is assumed to have type unsigned int *.
x, X	Matches a hexadecimal integer; the corresponding argument is assumed to have type unsigned int *.
a†, A†, e, E, f, F†, g, G	Matches a floating-point number; the corresponding argument is assumed to have type float *. In C99, the number can be infinity or NaN.
c	Matches *n* characters, where *n* is the maximum field width, or one character if no field width is specified. The corresponding argument is assumed to be a pointer to a character array (or a character object, if no field width is specified). Doesn't add a null character at the end.
s	Matches a sequence of non-white-space characters, then adds a null character at the end. The corresponding argument is assumed to be a pointer to a character array.
[	Matches a nonempty sequence of characters from a scanset, then adds a null character at the end. The corresponding argument is assumed to be a pointer to a character array.
p	Matches a pointer value in the form that ...printf would have written it. The corresponding argument is assumed to be a pointer to a void * object.
n	The corresponding argument must point to an object of type int. Stores in this object the number of characters read so far by this call of ...scanf. No input is consumed and the return value of ...scanf isn't affected.
%	Matches the character %.

†C99 only

Numeric data items can always begin with a sign (+ or -). The o, u, x, and X specifiers convert the item to unsigned form, however, so they're not normally used to read negative numbers.

The [specifier is a more complicated (and more flexible) version of the s specifier. A complete conversion specification using [has the form %[*set*] or %[^*set*], where *set* can be any set of characters. (If] is one of the characters in *set*, however, it must come first.) %[*set*] matches any sequence of characters in *set* (the *scanset*). %[^*set*] matches any sequence of characters *not* in *set* (in other words, the scanset consists of all characters not in *set*). For example, %[abc]