Table 22.6
Conversion Specifiers for
...`printf` Functions

Conversion Specifier	*Meaning*
`d`, `i`	Converts an `int` value to decimal form.
`o`, `u`, `x`, `X`	Converts an `unsigned int` value to base 8 (`o`), base 10 (`u`), or base 16 (`x`, `X`). `x` displays the hexadecimal digits `a`–`f` in lower case; `X` displays them in upper case.
`f`, `F`†	Converts a `double` value to decimal form, putting the decimal point in the correct position. If no precision is specified, displays six digits after the decimal point.
`e`, `E`	Converts a `double` value to scientific notation. If no precision is specified, displays six digits after the decimal point. If `e` is chosen, the exponent is preceded by the letter `e`; if `E` is chosen, the exponent is preceded by `E`.
`g`, `G`	`g` converts a `double` value to either `f` form or `e` form. `e` form is selected if the number's exponent is less than –4 *or* greater than or equal to the precision. Trailing zeros are not displayed (unless the `#` flag is used); a decimal point appears only when followed by a digit. `G` chooses between `F` and `E` forms.
`a`†, `A`†	Converts a `double` value to hexadecimal scientific notation using the form `[-]0x`*h*`.`*hhhh*`p±`*d*, where `[-]` is an optional minus sign, the *h*'s represent hex digits, ± is either a plus or minus sign, and *d* is the exponent. *d* is a decimal number that represents a power of 2. If no precision is specified, enough digits are displayed after the decimal point to represent the exact value of the number (if possible). `a` displays the hex digits `a`–`f` in lower case; `A` displays them in upper case. The choice of `a` or `A` also affects the case of the letters `x` and `p`.
`c`	Displays an `int` value as an unsigned character.
`s`	Writes the characters pointed to by the argument. Stops writing when the number of bytes specified by the precision (if present) is reached or a null character is encountered.
`p`	Converts a `void *` value to printable form.
`n`	The corresponding argument must point to an object of type `int`. Stores in this object the number of characters written so far by this call of `...printf`; produces no output.
`%`	Writes the character `%`.

†C99 only

C99 Changes to ...`printf` Conversion Specifications

The conversion specifications for `printf` and `fprintf` have undergone a number of changes in C99:

greatest-width integers ➤27.1

- *Additional length modifiers.* C99 adds the `hh`, `ll`, `j`, `z`, and `t` length modifiers. `hh` and `ll` provide additional length options, `j` allows greatest-width integers to be written, and `z` and `t` make it easier to write values of type `size_t` and `ptrdiff_t`, respectively.