- ***Length modifier*** (optional). The presence of a length modifier indicates that the item to be displayed has a type that's longer or shorter than is normal for a particular conversion specification. (For example, `%d` normally refers to an `int` value; `%hd` is used to display a `short int` and `%ld` is used to display a `long int`.) Table 22.5 lists each length modifier, the conversion specifiers with which it may be used, and the type indicated by the combination of the two. (Any combination of length modifier and conversion specifier not shown in the table causes undefined behavior.)

Table 22.5 Length Modifiers for ...`printf` Functions

Length Modifier	*Conversion Specifiers*	*Meaning*
`hh`†	d, i, o, u, x, X	`signed char`, `unsigned char`
	n	`signed char *`
`h`	d, i, o, u, x, X	`short int`, `unsigned short int`
	n	`short int *`
`l` *(ell)*	d, i, o, u, x, X	`long int`, `unsigned long int`
	n	`long int *`
	c	`wint_t`
	s	`wchar_t *`
	a, A, e, E, f, F, g, G	no effect
`ll`† *(ell-ell)*	d, i, o, u, x, X	`long long int`, `unsigned long long int`
	n	`long long int *`
`j`†	d, i, o, u, x, X	`intmax_t`, `uintmax_t`
	n	`intmax_t *`
`z`†	d, i, o, u, x, X	`size_t`
	n	`size_t *`
`t`†	d, i, o, u, x, X	`ptrdiff_t`
	n	`ptrdiff_t *`
`L`	a, A, e, E, f, F, g, G	`long double`

†C99 only

- ***Conversion specifier***. The conversion specifier must be one of the characters listed in Table 22.6. Notice that `f`, `F`, `e`, `E`, `g`, `G`, `a`, and `A` are all designed to write `double` values. However, they work fine with `float` values as well; thanks to the default argument promotions, `float` arguments are converted automatically to `double` when passed to a function with a variable number of arguments. Similarly, a character passed to ...`printf` is converted automatically to `int`, so the `c` conversion works properly.

default argument promotions ➤9.3

Be careful to follow the rules described here; the effect of using an invalid conversion specification is undefined.