형식 지정자

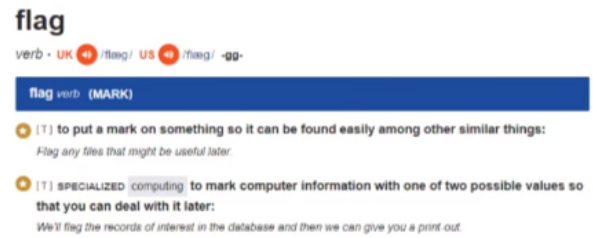
%[flags][width][.precision][length]specifier

```
printf("%+10.5hi", 256);
```

flags

flags	description
-	Left-justify within the given field width; Right justification is the default (see *width* sub-specifier).
+	Forces to preceed the result with a plus or minus sign (+ or -) even for positive numbers. By default, only negative numbers are preceded with a - sign.
(space)	If no sign is going to be written, a blank space is inserted before the value.
#	Used with `o`, `x` or `X` specifiers the value is preceeded with `0`, `0x` or `0X` respectively for values different than zero. Used with `a`, `A`, `e`, `E`, `f`, `F`, `g` or `G` it forces the written output to contain a decimal point even if no more digits follow. By default, if no digits follow, no decimal point is written.
0	Left-pads the number with zeroes (`0`) instead of spaces when padding is specified (see *width* sub-specifier).

http://www.cplusplus.com/reference/cstdio/printf/



width and .precision

width	description
(number)	Minimum number of characters to be printed. If the value to be printed is shorter than this number, the result is padded with blank spaces. The value is not truncated even if the result is larger.
*	The *width* is not specified in the *format* string, but as an additional integer value argument preceding the argument that has to be formatted.

.precision	description
.number	For integer specifiers (`d`, `i`, `o`, `u`, `x`, `X`): *precision* specifies the minimum number of digits to be written. If the value to be written is shorter than this number, the result is padded with leading zeros. The value is not truncated even if the result is longer. A *precision* of `0` means that no character is written for the value `0`. For `a`, `A`, `e`, `E`, `f` and `F` specifiers: this is the number of digits to be printed **after** the decimal point (by default, this is 6). For `g` and `G` specifiers: This is the maximum number of significant digits to be printed. For `s`: this is the maximum number of characters to be printed. By default all characters are printed until the ending null character is encountered. If the period is specified without an explicit value for *precision*, `0` is assumed.
.*	The *precision* is not specified in the *format* string, but as an additional integer value argument preceding the argument that has to be formatted.

http://www.cplusplus.com/reference/cstdio/printf/

length and specifiers

The *length* sub-specifier modifies the length of the data type. This is a chart showing the types used to interpret the corresponding arguments with and without *length* specifier (if a different type is used, the proper type promotion or conversion is performed, if allowed):

	specifiers						
length	`d i`	`u o x X`	`f F e E g G a A`	`c`	`s`	`p`	`n`
(none)	`int`	`unsigned int`	`double`	`int`	`char*`	`void*`	`int*`
`hh`	`signed char`	`unsigned char`					`signed char*`
`h`	`short int`	`unsigned short int`					`short int*`
`l`	`long int`	`unsigned long int`		`wint_t`	`wchar_t*`		`long int*`
`ll`	`long long int`	`unsigned long long int`					`long long int*`
`j`	`intmax_t`	`uintmax_t`					`intmax_t*`
`z`	`size_t`	`size_t`					`size_t*`
`t`	`ptrdiff_t`	`ptrdiff_t`					`ptrdiff_t*`
`L`			`long double`				

Note regarding the `c` specifier: it takes an `int` (or `wint_t`) as argument, but performs the proper conversion to a `char` value (or a `wchar_t`) before formatting it for output.

http://www.cplusplus.com/reference/cstdio/printf/