Annotate Image

imput or output. Here is a comprehensive table or the format specifiers used in c, categorized by the type of data they handle:

Common Format Specifiers in C

Specifier	Description	Example Usage	GContinue generating
c	Character	'printf("%c", 'A');	Outputs a single character.
`%d`	Signed decimal integer	`printf("%d", 42);`	Outputs a signed integer in base 10.
`%i`	Signed decimal integer (same as `%d`)	`printf("%i", -42);`	Outputs a signed integer in base 10.
`%u`	Unsigned decimal integer	`printf("%u", 42);`	Outputs an unsigned integer in base 10.
`%f`	Decimal floating-point number	`printf("%f", 3.14);`	Outputs a floating-point number in decimal notation (default precision is 6).
`%e`	Scientific notation (lowercase 'e')	`printf("%e", 3.14);`	Outputs a floating-point number in scientific notation.
`%E`	Scientific notation (uppercase 'E')	`print \ \ YE", 3.14);`	Outputs a floating-point number in scientifi notation with 'E'.
`%g`	Use `%f` or `%e` based on the value	`printf("%g", 3.14);`	Outputs a floating- GContinue generating or '%f' format (whichever is snorter).
*\G`	Use "%+" or "%±" based on the value	print+("%G",	Outputs a floating-point number, using '%+' or '%f' format (whichever is shorter).
`%x`	Unsigned hexadecimal integer (lowercase)	`printf("%x", 255);`	Outputs an unsigned integer in hexadecima (base 16) using lowercase letters.
`%X`	Unsigned hexadecimal integer (uppercase)	`printf("%X", 255);`	Outputs an unsigned integer in hexadecima (base 16) using uppercase letters.
`%o`	Unsigned octal integer	`printf("%o", 255);`	Outputs an unsigned integer in octal (base 8).
`%s`	String of characters	`printf("%s", "Hi");`	Outputs a string.
`%p`	Pointer address	`printf("%p", &var);`	Outputs a pointer address in implementation-defined format.
`%%`	Percent sign	`print \ \%");`	Outputs a literal percent sign.

Size Modifiers for Integer Types

GContinue generating

Size modifiers are used in conjunction with the format specifiers for integers to specify the size of the data type:

Modifier	Modifier Specifier Description Example Usa		Example Usage	Explanation
`h`	`%hd`	Short signed integer	`printf("%hd", 32767);`	Outputs a short integer.
`h`	`%hu`	Short unsigned integer	`printf("%hu", 65535);`	Outputs a short unsigned integer.
`h`	`%hx`	Short unsigned hexadecimal integer	`printf("%hx", 255);`	Outputs a short unsigned integer in hexadecimal.
`h`	`%ho`	Short unsigned octal integer	`printf("%ho", 255);`	Outputs a short unsigned integer in octal.
`1`	`%1d`	Long signed integer	ψ, intf("%ld", 123456789L);	Outputs a long
`1`	`%1u`	Long unsigned integer	`printf("%lu", 123456789UL);`	GContinue generating
				unsigned integer.
`1`	`%1x`	Long unsigned hexadecimal integer	`printf("%1x", 255UL);`	Outputs a long unsigned integer in hexadecimal.
`1`	`%10`	Long unsigned octal integer	`printf("%10", 255UL);`	Outputs a long unsigned integer in octal.
`11`	`%11d`	Long long signed integer	`printf("%1ld", 123456789012345LL);`	Outputs a long long integer.
`11`	`%11u`	Long long unsigned integer	`printf("%11u", 123456789012345ULL);`	Outputs a long long unsigned integer.
,11,	`%11x`	Long long unsigned hexadecimal integer	`printf("%11x", 255ULL);`	Outputs a long long unsigned integer in hexadecimal.
`11`	`%110`	Long long unsigned octal integer	`printf("%110", 255ULL);`	GContinue generating octal.
`j`	`%jd`	intmax_t signed integer		

6/15/24, 3:07 PM

Annotate Image

		-		_
`j`	`%ju`	%ju' uintmax_t 'printf("%ju", unsigned integer (uintmax_t)123456789012345ULL);'		Outputs a maximum width unsigned integer.
`z`	`%zd`	size_t signed integer (for signed interpretation)	`printf("%zd", (ssize_t)size);`	Outputs a size_t value (commonly used for size_t to be interpreted as signed).
`z`	`%zu`	size_t unsigned integer	`printf("%zu", size);`	Outputs a size_t value.
`t`	`%td`	ptrdiff_t signed integer	<pre>ntf("%td", (ptrdiff_t)ptrDiff);`</pre>	Outputs a ptrdiff_t value.
`t`	`%tu`	ptrdiff_t unsigned integer	`printf("%tu", (ptrdiff_t)ptrDiff);`	GContinue generating value interpreted as unsigned.

Size Modifiers for Floating-Point Types

Modifier	Specifier	Description	Example Usage	Explanation
`L`	`%Lf`	Long	`printf("%Lf",	Outputs a long double floating-point
		double	3.14L);`	number.

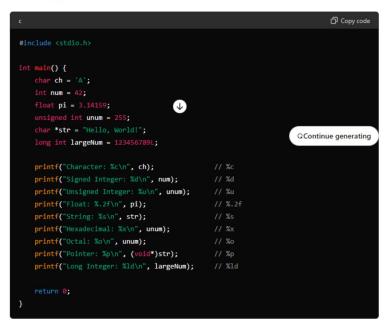
Width and Precision Specifiers

Width and precision specifiers control the minimum number of characters to be printed and the number of digits to appear after the decimal point for floating-point numbers.

Specifier Description '%10d' Minimum width of 10 characters '%.2f' Precision of 2 digits after the decimal point		Example Usage	Explanation Outputs the integer with a minimum width of 10, padding with spaces if necessary.	
		`printf \\ \ 10d", 42);`		
		`printf("%.2f", 3.14159);`	Outputs a floating GContinue generating digits after the decision	
`%10.2ſ`	Minimum width of 10, with 2 digits after the decimal point	`printf("%10.2f", 3.14159);`	Combines width and precision specifiers. Outputs the number with a total width of 10 and 2 digits after the decimal point.	

Example Usage

Here's a C program demonstrating various format specifiers:



This program outputs various types using the $\overset{\cdot}{\psi}$ respective format specifiers, showing how to format characters, integers, floating-point numbers, and more.

These format specifiers allow for

☐ ♥ ♥ ♥

GContinue generating



Message ChatGPT

natGPT can make mistakes. Check important info.

