

H

header An entity whose contents are made available via the `#include` pre-processor directive. A header does not need to exist as a text file—it can be stored as a binary file or defined internally within the translator. Typically, a header contains function declarations and macro definitions. It also may contain `typedef` and structure, union, and enumeration declarations. Headers can be nested up to some implementation-defined depth (at least 8 in C89, 15 in C99). Standard headers are accessed using the following notation:

```
#include <header-name>
```

while programmer-defined headers are accessed using

```
#include "header-name"
```

header, name of A header name preprocessing token which has one of the following forms:

```
<h-char-sequence>  
"q-char-sequence"
```

where *h-char-sequence* may contain any member of the source character set except the new-line character and `>`; and where *q-char-sequence* may contain any member of the source character set except the new-line character and `"`.

Header names are only recognized as such in the context of a `#include` directive.

If the characters `'`, `\`, `"`, or `/*` occur in the sequence between the `<` and `>` delimiters, the behavior is undefined. Similarly, if the characters `'`, `\`, or `/*` occur in the sequence between the `"` delimiters, the behavior is undefined. As such, sequences of characters that resemble escape sequences cause undefined behavior. (MS-DOS/Windows directory names use a backslash to separate subdirectories. Therefore, a directive such as `#include "\abc.h"` has undefined behavior.)

header, standard A header that is required by Standard C. Each library routine defined by Standard C is declared in a corresponding standard header, the names and purposes of which are as follows:

<i>Header</i>	<i>Purpose</i>
assert.h	Program diagnostic purposes
complex.h ^{C99}	Complex arithmetic
ctype.h	Character testing and conversion
errno.h	Various error-checking facilities
fenv.h ^{C99}	Floating-point environment
float.h	Floating type characteristics
inttypes.h ^{C99}	Integer format conversion
iso646.h ^{C95}	Alternate Token Spellings
limits.h	Integer type sizes
locale.h	Internationalization support
math.h	Math functions
setjmp.h	Nonlocal jump facility
signal.h	Signal handling
stdarg.h	Variable argument support
stdbool.h ^{C99}	Boolean type and values
stddef.h	Miscellaneous
stdint.h ^{C99}	Integer types
stdio.h	Input/output functions
stdlib.h	General utilities
string.h	String functions
tgmath.h ^{C99}	Type-generic math
time.h	Date and time functions
wchar.h ^{C95}	Multibyte and wide-character processing
wctype.h ^{C95}	Wide-character classification and mapping

All external identifiers declared in the standard headers are reserved, whether or not their associated header is referenced.

Standard headers may be included in any order and multiple times in the same scope without producing ill effects. The exception is `assert.h`, which, if included multiple times, can behave differently depending on the existence of the macro `NDEBUG`.

You should include a standard header only at file scope in your program.

heap Memory that may be dynamically allocated and released by a user program using the library functions `calloc`, `free`, `malloc`, and `realloc`.

hexadecimal constant *See* constant, integer.

hexadecimal escape sequence A sequence of the form `\x..h` that represents the character having a bit pattern with value `h..h` hexadecimal. As shown, the length of the sequence of hexadecimal digits is not limited by Standard C, while that for an octal sequence is.

horizontal-tab character One of the white space characters allowed in source text.

horizontal-tab escape sequence An escape sequence, `\t`, which represents the horizontal tab character.

hosted environment^{C89} *See* environment, hosted.

HUGE_VAL^{C89} A macro, defined in `math.h`, that expands to a positive `double` expression that is not necessarily representable as a `float`. It is returned from some math functions to indicate an unrepresentably large value. On some implementations, it may equal infinity. Prior to C99, it was not required to be a constant expression. The name `HUGE_VAL` replaced what many implementations previously called `HUGE`. *See also* `HUGE_VALF` and `HUGE_VALL`.

HUGE_VALF^{C99} A macro, defined in `math.h`, that expands to a positive `float` constant expression and is returned from some math functions to indicate an unrepresentably large value. On some implementations, it may equal infinity. *See also* `HUGE_VAL` and `HUGE_VALL`.

HUGE_VALL^{C99} A macro, defined in `math.h`, that expands to a positive `long double` constant expression and is returned from some math functions to indicate an unrepresentably large value. On some implementations, it may equal infinity. *See also* `HUGE_VAL` and `HUGE_VALF`.

hyperbolic functions The `math.h` functions `acosh`, `asinh`, `atanh`, `cosh`, `sinh`, and `tanh`, and their `float` and `long double` counterparts, for both floating and complex types.

hypot[f|l]^{C99} A function that computes the hypotenuse of a right triangle given the two sides `x` and `y`, without undue underflow or overflow.

```
#include <math.h>
double hypot(double x, double y);
float hypotf(float x, float y);
long double hypotl(long double x, long double y);
```

A range error may occur.

