

NAME

ceil, ceilf, ceill – ceiling function: smallest integral value not less than argument

LIBRARY

Math library (*libm*, *-lm*)

SYNOPSIS

```
#include <math.h>
```

```
double ceil(double x);
```

```
float ceilf(float x);
```

```
long double ceill(long double x);
```

Feature Test Macro Requirements for glibc (see **feature_test_macros(7)**):

```
ceilf(), ceill():
```

```
_ISOC99_SOURCE || _POSIX_C_SOURCE >= 200112L
```

```
    /* Since glibc 2.19: */ _DEFAULT_SOURCE
```

```
    /* glibc <= 2.19: */ _BSD_SOURCE || _SVID_SOURCE
```

DESCRIPTION

These functions return the smallest integral value that is not less than *x*.

For example, *ceil(0.5)* is 1.0, and *ceil(-0.5)* is 0.0.

RETURN VALUE

These functions return the ceiling of *x*.

If *x* is integral, +0, -0, NaN, or infinite, *x* itself is returned.

ERRORS

No errors occur.

ATTRIBUTES

For an explanation of the terms used in this section, see **attributes(7)**.

Interface	Attribute	Value
ceil(), ceilf(), ceill()	Thread safety	MT-Safe

STANDARDS

C11, POSIX.1-2008.

HISTORY

C99, POSIX.1-2001.

The variant returning *double* also conforms to SVr4, 4.3BSD, C89.

SUSv2 and POSIX.1-2001 contain text about overflow (which might set *errno* to **ERANGE**, or raise an **FE_OVERFLOW** exception). In practice, the result cannot overflow on any current machine, so this error-handling stuff was just nonsense. (More precisely, overflow can happen only when the maximum value of the exponent is smaller than the number of mantissa bits. For the IEEE-754 standard 32-bit and 64-bit floating-point numbers the maximum value of the exponent is 127 (respectively, 1023), and the number of mantissa bits including the implicit bit is 24 (respectively, 53).) This was removed in POSIX.1-2008.

The integral value returned by these functions may be too large to store in an integer type (*int*, *long*, etc.). To avoid an overflow, which will produce undefined results, an application should perform a range check on the returned value before assigning it to an integer type.

SEE ALSO

floor(3), **lrint(3)**, **nearbyint(3)**, **rint(3)**, **round(3)**, **trunc(3)**