



Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'cosh.3p' command

\$ man cosh.3p

COSH(3P) POSIX Programmer's Manual COSH(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

cosh, coshf, coshl ? hyperbolic cosine functions

SYNOPSIS

```
#include <math.h>

double cosh(double x);

float coshf(float x);

long double coshl(long double x);
```

DESCRIPTION

The functionality described on this reference page is aligned with the ISO C standard. Any conflict between the requirements described here and the ISO C standard is unintentional. This volume of POSIX.1?2017 defers to the ISO C standard.

These functions shall compute the hyperbolic cosine of their argument x.

An application wishing to check for error situations should set errno to zero and call feclearexcept(FE_ALL_EXCEPT) before calling these functions. On return, if errno is non-zero or fetestexcept(FE_INVALID |

FE_DIVBYZERO | FE_OVERFLOW | FE_UNDERFLOW) is non-zero, an error has occurred.

RETURN VALUE

Upon successful completion, these functions shall return the hyperbolic cosine of x .

If the correct value would cause overflow, a range error shall occur and `cosh()`, `coshf()`, and `coshl()` shall return the value of the macro `HUGE_VAL`, `HUGE_VALF`, and `HUGE_VALL`, respectively.

If x is NaN, a NaN shall be returned.

If x is ± 0 , the value 1.0 shall be returned.

If x is $\pm \text{Inf}$, $\pm \text{Inf}$ shall be returned.

ERRORS

These functions shall fail if:

Range Error The result would cause an overflow.

If the integer expression `(math_errhandling & MATH_ERRNO)` is non-zero, then `errno` shall be set to `[ERANGE]`. If the integer expression `(math_errhandling & MATH_ERREXCEPT)` is non-zero, then the overflow floating-point exception shall be raised.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

On error, the expressions `(math_errhandling & MATH_ERRNO)` and `(math_errhandling & MATH_ERREXCEPT)` are independent of each other, but at least one of them must be non-zero.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

`acosh()`, `feclearexcept()`, `fetestexcept()`, `isnan()`, `sinh()`, `tanh()`

The Base Definitions volume of POSIX.1?2017, Section 4.20, Treatment of

COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1-2017, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 7, 2018 Edition, Copyright (C) 2018 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.opengroup.org/unix/online.html> .

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.kernel.org/doc/man-pages/reporting_bugs.html .