

نوروز ۸۶ مبارک 😊

C++ <cmath> Library Functions

Computer Programming Course

Group 3

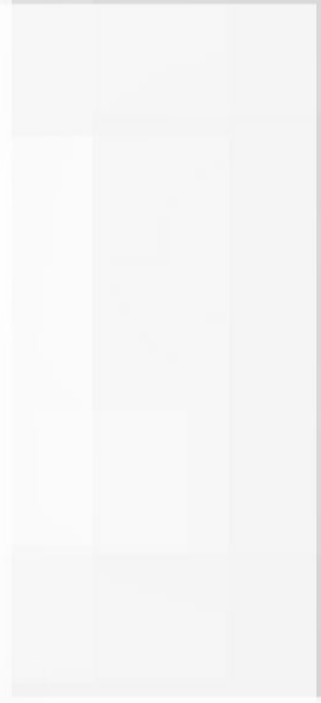
Sharif University of Technology

Instructor: Ehsan Nazerfard

C++ cmath members: fabs()



- ▶ It returns the absolute value of x (i.e. $|x|$).
- ▶ Prototypes:
 - ▣ `float fabs (float x);`
 - ▣ `long double fabs (long double x);`
- ▶ Parameters:
 - **X**: Floating point value.
- ▶ Return Value
 - The absolute value of x .
- ▶ `abs()` is the same as `fabs()`.
- ▶ `fabsf()` & `fabsl()` are defined for float & long double respectively in C.



C++ cmath members: ceil()



- ▶ It returns the smallest integral value that is not less than x .
- ▶ Prototypes:
 - ▣ `float ceil (float x);`
 - ▣ `long double ceil (long double x);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - The smallest integral value not less than x .
- ▶ `ceilf()` & `ceilll()` are defined for float & long double respectively in C.



C++ cmath members: floor()



- ▶ It returns the largest integral value that is not greater than x .
- ▶ Prototypes:
 - ▣ `float floor (float x);`
 - ▣ `long double floor (long double x);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - The largest integral value not greater than x .
- ▶ `floorf()` & `floorl()` are defined for float & long double respectively in C.



C++ cmath members: log()



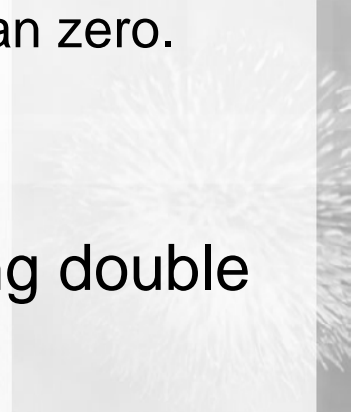
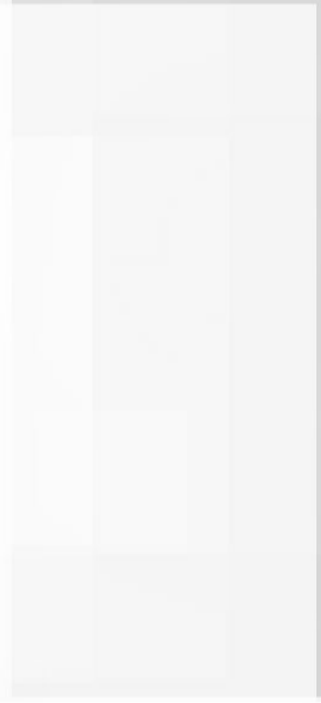
- ▶ It returns the natural logarithm (base-e) of x.
- ▶ Prototypes:
 - ▣ `float log (float x);`
 - ▣ `long double log (long double x);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - Natural logarithm of x.
 - If $(x < 0)$ it returns IND (Indeterminate).
 - If $(x == 0)$ it returns INF (HUGE_VAL).
- ▶ `logf()` & `logl()` are defined for float & long double respectively in C.



C++ cmath members: log10()



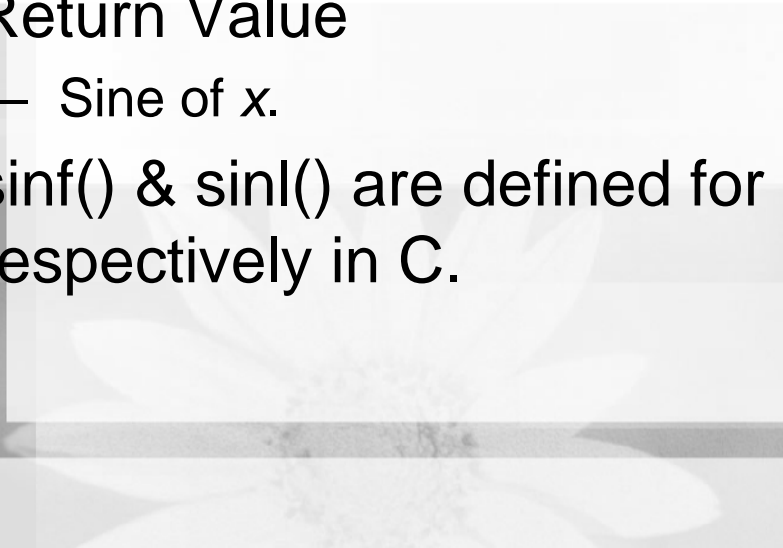
- ▶ It returns the base-10 logarithm of x .
- ▶ Prototypes:
 - ▣ `float log10 (float x);`
 - ▣ `long double log10 (long double x);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - Base-10 logarithm of x , for values of x greater than zero.
 - If ($x < 0$) it returns IND (Indeterminate).
 - If ($x == 0$) it returns INF (HUGE_VAL).
- ▶ `log10f()` & `log10l()` are defined for float & long double respectively in C.



C++ cmath members: sin()



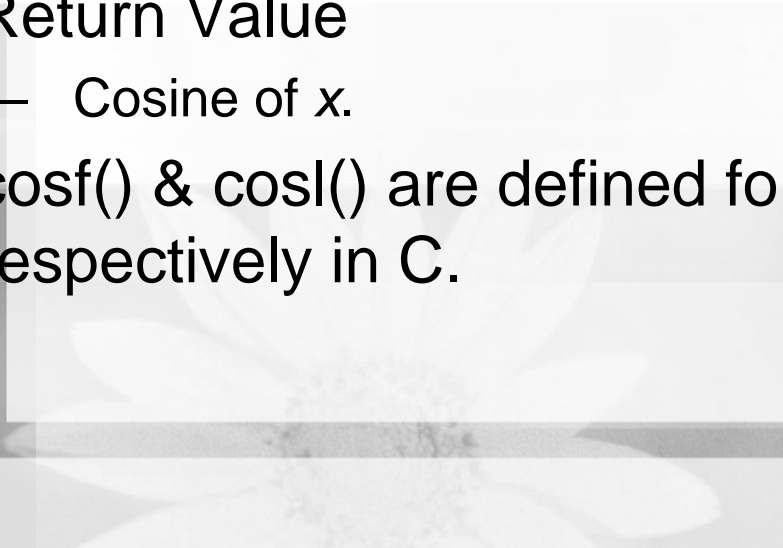
- ▶ It returns the sine of an angle of x radians.
- ▶ Prototypes:
 - ▣ `float sin (float x);`
 - ▣ `long double sin (long double x);`
- ▶ Parameters
 - **X**: Floating point value representing an angle expressed in radians.
- ▶ Return Value
 - Sine of x .
- ▶ `sinf()` & `sinl()` are defined for float & long double respectively in C.



C++ cmath members: cos()



- ▶ It returns the cosine of an angle of x radians.
- ▶ Prototypes:
 - ▣ `float cos (float x);`
 - ▣ `long double cos (long double x);`
- ▶ Parameters
 - **X**: Floating point value representing an angle expressed in radians.
- ▶ Return Value
 - Cosine of x .
- ▶ `cosf()` & `cosl()` are defined for float & long double respectively in C.



C++ cmath members: tan()

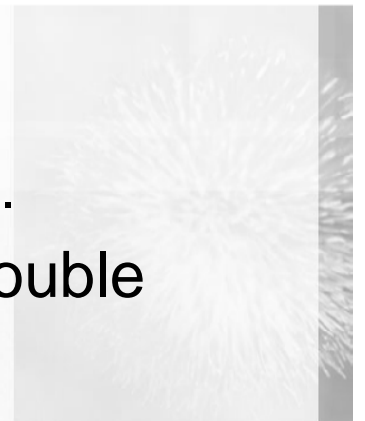


- ▶ It returns the tangent of an angle of x radians.
- ▶ Prototypes:
 - ▣ `float tan (float x);`
 - ▣ `long double tan (long double x);`
- ▶ Parameters
 - **X**: Floating point value representing an angle expressed in radians.
- ▶ Return Value
 - Tangent of x .
- ▶ `tanf()` & `tanl()` are defined for float & long double respectively in C.

C++ cmath members: exp()



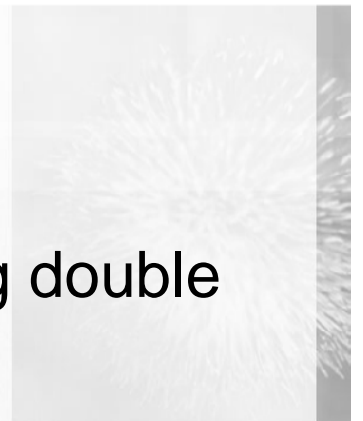
- ▶ It returns the exponential function of x , which is the e number raised to the power x .
- ▶ Prototypes:
 - ▣ `float exp (float x);`
 - ▣ `long double exp (long double x);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - Exponential value of x .
 - If ($|result|$ is too big) it returns INF (HUGE_VAL).
- ▶ `expf()` & `expl()` are defined for float & long double respectively in C.



C++ cmath members: fmod()



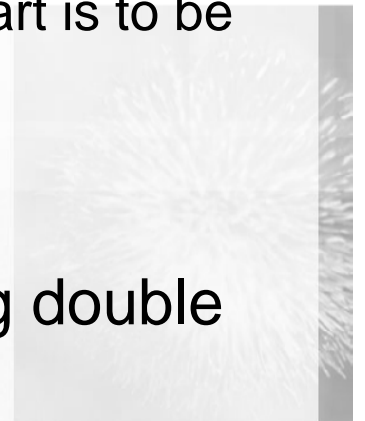
- ▶ It returns the floating-point remainder of *numerator/denominator*.
- ▶ Prototypes:
 - ▣ `float fmod (float numerator, float denominator);`
 - ▣ `long double fmod (long double n, long double d);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - The remainder of dividing the arguments.
 - If `(denominator==0)` it returns IND.
- ▶ `fmodf()` & `fmodl()` are defined for float & long double respectively in C.



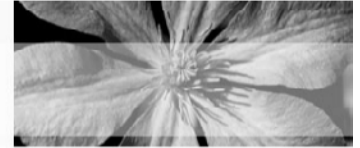
C++ cmath members: modf()



- ▶ It Breaks x into two parts: the integer part (stored in the object pointed by *intpart*) and the fractional part (returned by the function). Each part has the same sign as x .
- ▶ Prototypes:
 - ▣ `long double modf (long double x, long double * intpart);`
 - ▣ `float modf (float x, float * intpart);`
- ▶ Parameters
 - **X**: Floating point value.
 - **intpart**: Pointer to an object where the integral part is to be stored.
- ▶ Return Value
 - The fractional part of x , with the same sign.
- ▶ `modff()` & `modfl()` are defined for float & long double respectively in C.



C++ cmath members: pow()



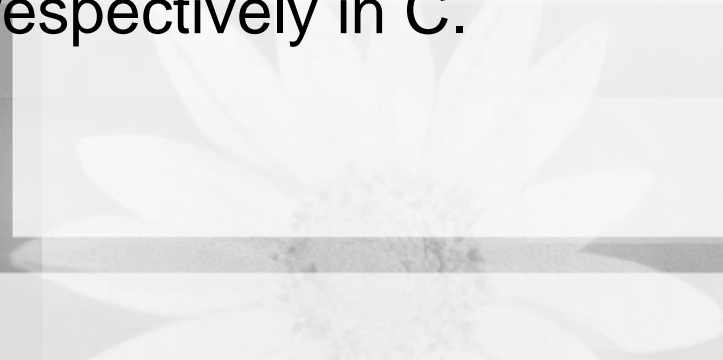
- ▶ It returns *base* raised to the power *exponent*: $base^{exponent}$.
- ▶ Prototypes:
 - ▣ `long double pow (long double b, long double e);`
 - ▣ `float pow (float base, float exponent);`
 - ▣ `float pow (float base, int exponent);`
 - ▣ `double pow (double base, int exponent);`
 - ▣ `long double pow (long double base, int exponent);`
- ▶ Parameters
 - **base**: Floating point value
 - **exponent**: Floating point value.
- ▶ Return Value
 - $base^{exponent}$
 - If `(base==0 && exponent<0)` it returns INF (HUGE_VAL).
 - If `(base<0 && exp is not Integral)` it returns IND.
- ▶ `powf()` & `powl()` are defined for float & long double respectively in C.



C++ cmath members: sqrt()



- ▶ It returns the square root of the x.
- ▶ Prototypes:
 - ▣ `float sqrt (float x);`
 - ▣ `long double sqrt (long double x);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - Square root of x.
- ▶ `sqrtf()` & `sqrtl()` are defined for float & long double respectively in C.



cmath members: HUGE_VAL



▶ HUGE_VAL

- If the magnitude of the result is so large that it cannot be represented in an object of the return type, a range error occurs, returning HUGE_VAL with the appropriate sign.

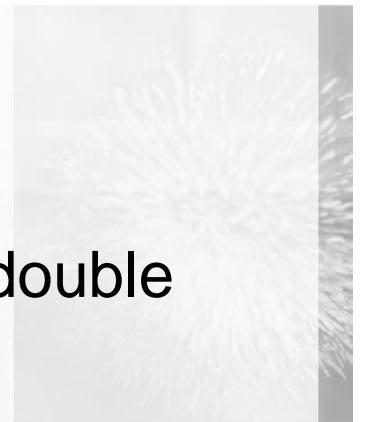
▶ Examples

- `cout << HUGE_VAL;` → `1.#INF`
- `cout << pow(10, 3000.0);` → `-1.#INF`
- `cout << sqrt(-4);` → `1.#IND`
- `cout << (float)1/0;` → `1.#INF`
- `cout << (float)0/0;` → `-1.#IND`
- `cout << log(-5.0);` → `-1.#INF`
- `cout << acos(3);` → `-1.#IND`
- `cout << isinf(1e+300 / 1e-9 ? "INF" : "NInf");`
- **INF & IND stand for Infinity & Indeterminate respectively.**

C++ cmath members: asin()



- ▶ It returns the principal value of the arc sine of x , expressed in radians.
- ▶ Prototypes:
 - ▣ `float asin (float x);`
 - ▣ `long double asin (long double x);`
- ▶ Parameters
 - **X**: Floating point value in the interval $[-1,+1]$.
- ▶ Return Value
 - Arc sine of x , in the interval $[-\pi/2,+\pi/2]$ radians.
 - If $(x < -1 \ || \ x > 1)$ it returns IND.
- ▶ `asinf()` & `asinl()` are defined for float & long double respectively in C.



C++ cmath members: acos()

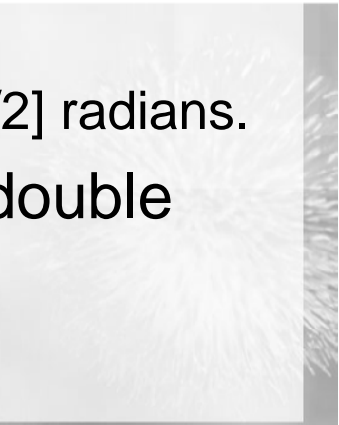
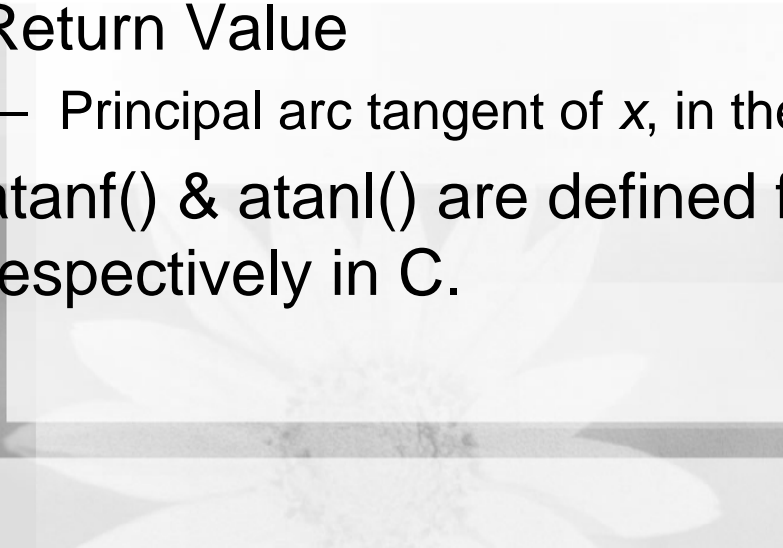


- ▶ It returns the principal value of the arc cosine of x , expressed in radians.
- ▶ Prototypes:
 - ▣ `float acos (float x);`
 - ▣ `long double acos (long double x);`
- ▶ Parameters
 - **X**: Floating point value in the interval $[-1,+1]$.
- ▶ Return Value
 - Arc arc cosine of x , in the interval $[0,\pi]$ radians.
 - If $(x < -1 \ || \ x > 1)$ it returns IND.
- ▶ `acosf()` & `acosl()` are defined for float & long double respectively in C.

C++ cmath members: atan()



- ▶ It returns the principal value of the arc tangent of x , expressed in radians.
- ▶ Prototypes:
 - ▣ `float atan (float x);`
 - ▣ `long double atan (long double x);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - Principal arc tangent of x , in the interval $[-\pi/2, +\pi/2]$ radians.
- ▶ `atanf()` & `atanl()` are defined for float & long double respectively in C.



C++ cmath members: atan2()



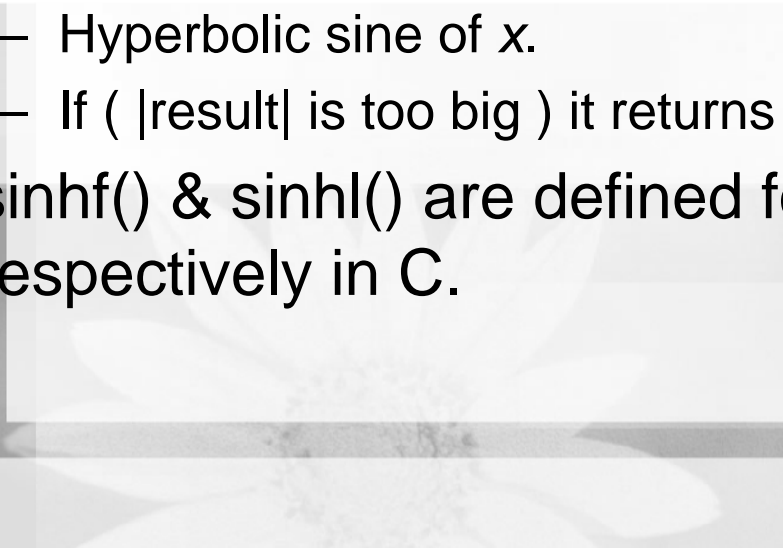
- ▶ It returns the principal value of the arc tangent of y/x , expressed in radians.
- ▶ Prototypes:
 - ▣ `long double atan2 (long double y, long double x);`
 - ▣ `float atan2 (float y, float x);`
- ▶ Parameters
 - **y**: Floating point value representing an y-coordinate.
 - **x**: Floating point value representing an x-coordinate.
- ▶ Return Value
 - Principal arc tangent of y/x , in the interval $[-\pi, +\pi]$ radians.
- ▶ `atan2f()` & `atan2l()` are defined for float & long double respectively in C.



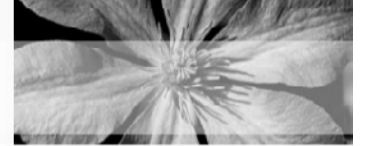
C++ cmath members: sinh()



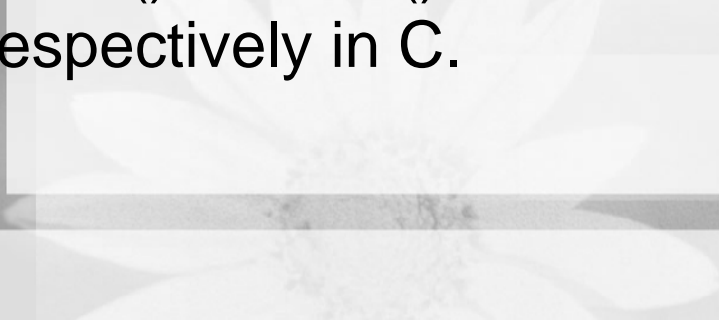
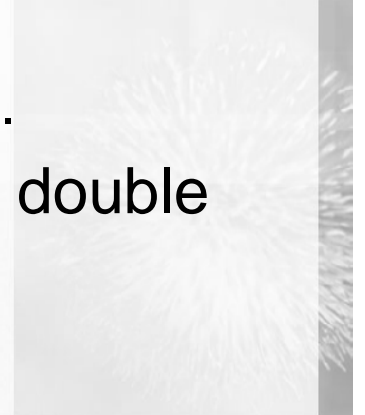
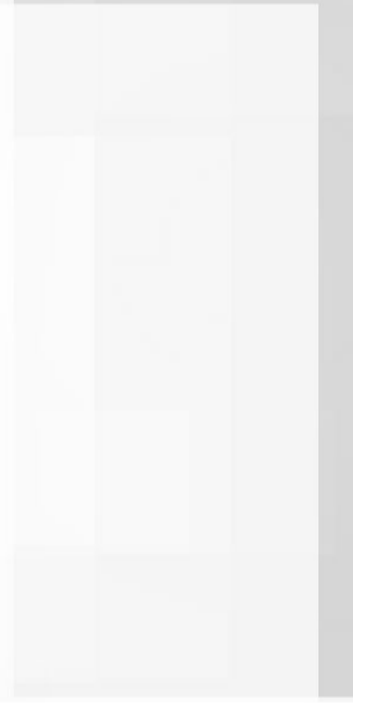
- ▶ It returns the hyperbolic sine of x .
- ▶ Prototypes:
 - ▣ `float sinh (float x);`
 - ▣ `long double sinh (long double x);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - Hyperbolic sine of x .
 - If ($|result|$ is too big) it returns INF (HUGE_VAL).
- ▶ `sinhf()` & `sinhl()` are defined for float & long double respectively in C.



C++ cmath members: cosh()



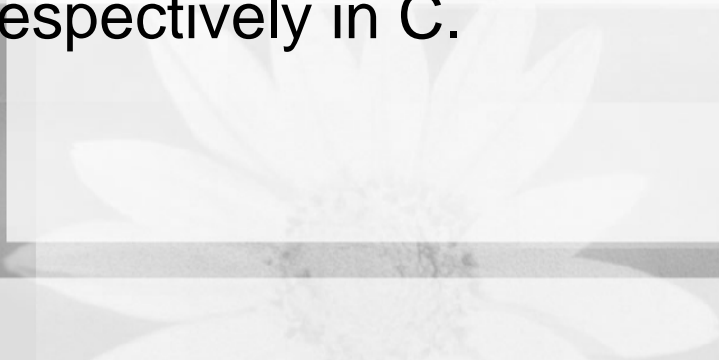
- ▶ It returns the hyperbolic cosine of x .
- ▶ Prototypes:
 - ▣ `float cosh (float x);`
 - ▣ `long double cosh (long double x);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - Hyperbolic cosine of x .
 - If ($|result|$ is too big) it returns INF (HUGE_VAL).
- ▶ `coshf()` & `coshl()` are defined for float & long double respectively in C.



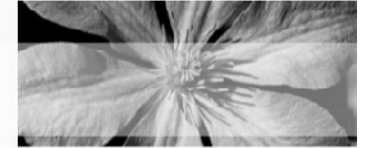
C++ cmath members: tanh()



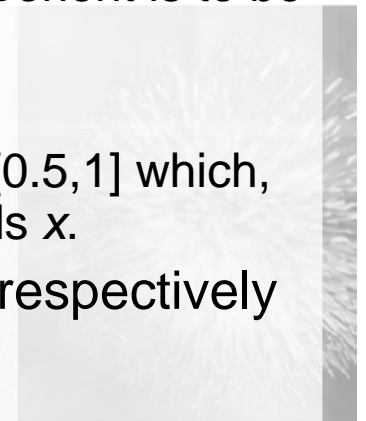
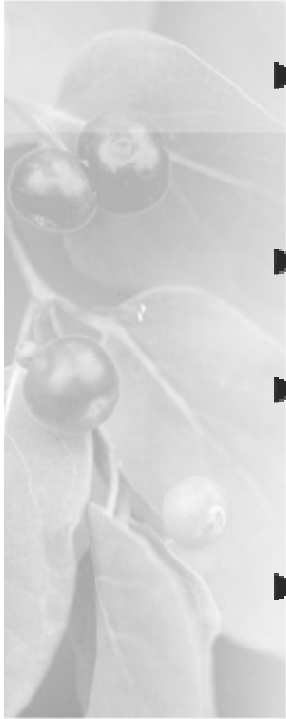
- ▶ It returns the hyperbolic tangent of x .
- ▶ Prototypes:
 - ▣ `float tanh (float x);`
 - ▣ `long double tanh (long double x);`
- ▶ Parameters
 - **X**: Floating point value.
- ▶ Return Value
 - Hyperbolic tangent of x .
- ▶ `tanhf()` & `tanhl()` are defined for float & long double respectively in C.



C++ cmath members: frexp()



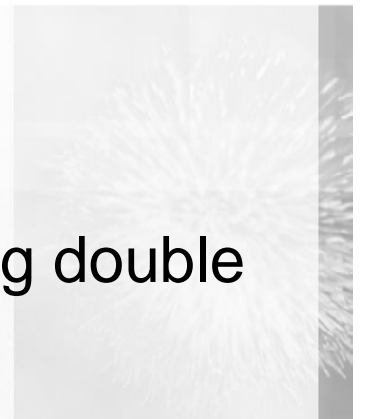
- ▶ It Breaks the floating point number x into its binary significand (a floating point value between 0.5 and 1.0) and an integral exponent for 2, such that $x = \text{significand} * 2^{\text{exponent}}$
- ▶ The exponent is stored in the location pointed by exp , and the significand is the value returned by the function.
- ▶ Prototypes:
 - ▣ `float frexp (float x, int * exp);`
 - ▣ `long double frexp (long double x, int * exp);`
- ▶ Parameters
 - **X**: Floating point value to be computed.
 - **exp**: Pointer to an int object where the value of the exponent is to be stored.
- ▶ Return Value
 - The value X , is the floating point value in the interval [0.5,1] which, once multiplied by 2 raised to the power of $*exp*$, yields x .
- ▶ `frexpf()` & `frexpl()` are defined for float & long double respectively in C.



C++ cmath members: ldexp()



- ▶ It returns the resulting floating point value from multiplying x (the significand) by 2 raised to the power of exp (the exponent).
- ▶ Prototypes:
 - ▣ `float ldexp (float x, int exp);`
 - ▣ `long double ldexp (long double x, int exp);`
- ▶ Parameters
 - **X**: Floating point value representing the significand.
 - **exp**: Value of the exponent.
- ▶ Return Value
 - The function returns $x * 2^{exp}$
- ▶ `ldexpf()` & `ldexpl()` are defined for float & long double respectively in C.



Conclusion



- ▶ It is recommended to use cast operator when using `<cmath>` functions to avoid call ambiguity. For example use `pow(3,6.0)` instead of `pow(3,6)` which results an ambiguous call error.

<code>#include <header.h></code>	<u>examples:</u> <code><math.h></code> <code><iostream.h></code>	In standard C++ this style is deprecated / antiquated ☹
<code>#include <<u>c</u>header></code>	<code><cmath></code>	C++ style for <u>C headers</u> 😊😊
<code>#include <header></code>	<code><iostream></code>	C++ style 😊😊😊

Reference



- ▶ <http://www.cplusplus.com/reference/clibrary/cmath/>
- ▶ Note: The prototypes appeared in this document are based on Dev-C++ 4.9.9.2 IDE.

