C Library math.h functions

The **math.h** header defines various mathematical functions and one macro. All the functions available in this library take **double** as an argument and return **double** as the result. Let us discuss some important functions one by one.

1. **double ceil(double x)**: The C library function double ceil(double x) returns the smallest integer value greater than or equal to x.

```
syntax : double ceil(double x)
```

```c
// C code to illustrate
// the use of ceil function.
#include <stdio.h>
#include <math.h>

int main ()
{
float val1, val2, val3, val4;

val1 = 1.6;
val2 = 1.2;
val3 = -2.8;
val4 = -2.3;

printf ("value1 = %.1lf\n", ceil(val1));
printf ("value2 = %.1lf\n", ceil(val2));
printf ("value3 = %.1lf\n", ceil(val3));
printf ("value4 = %.1lf\n", ceil(val4));

return(0);
}
```

```
value1 = 2.0
value2 = 2.0
value3 = -2.0
value4 = -2.0
```

2. **double floor(double x):** The C library function double floor(double x) returns the largest integer value less than or equal to x.

```
syntax : double floor(double x)
```

```c
// C code to illustrate
// the use of floor function
#include <stdio.h>
#include <math.h>

int main ()
{
   float val1, val2, val3, val4;

   val1 = 1.6;
   val2 = 1.2;
   val3 = -2.8;
   val4 = -2.3;

   printf("Value1 = %.1lf\n", floor(val1));
   printf("Value2 = %.1lf\n", floor(val2));
   printf("Value3 = %.1lf\n", floor(val3));
   printf("Value4 = %.1lf\n", floor(val4));

   return(0);
}
```

```
Value1 = 1.0
Value2 = 1.0
Value3 = -3.0
Value4 = -3.0
```

3. **double fabs(double x)**: The C library function double fabs(double x) returns the absolute value of x.

```
syntax : double fabs(double x)
```

```c
// C code to illustrate
// the use of fabs function
#include <stdio.h>
#include <math.h>

int main ()
{
   int a, b;
   a = 1234;
   b = -344;

   printf("The absolute value of %d is %lf\n", a, fabs(a));
   printf("The absolute value of %d is %lf\n", b, fabs(b));

    return(0);
 }
```

```
The absolute value of 1234 is 1234.000000
The absolute value of -344 is 344.000000
```

4. **double log(double x)**: The C library function double log(double x) returns the natural logarithm (base-e logarithm) of x.

```
syntax : double log(double x)
```

```c
// C code to illustrate
// the use of log function

#include <stdio.h>
#include <math.h>

int main ()
{
   double x, ret;
   x = 2.7;

   /* finding log(2.7) */
   ret = log(x);
   printf("log(%lf) = %lf", x, ret);

   return(0);
}
```

```
log(2.700000) = 0.993252
```

5. **double log10(double x)**: The C library function double log10(double x) returns the common logarithm (base-10 logarithm) of x.

```
syntax : double log10(double x)
```

```c
// C code to illustrate
// the use of log10 function
#include <stdio.h>
#include <math.h>

int main ()
{
   double x, ret;
   x = 10000;

   /* finding value of log1010000 */
   ret = log10(x);
   printf("log10(%lf) = %lf\n", x, ret);

   return(0);
}
```

```
log10(10000.000000) = 4.000000
```

6. **double fmod(double x, double y)** : The C library function double fmod(double x, double y) returns the remainder of x divided by y.

```
syntax : double fmod(double x, double y)
```

```c
// C code to illustrate
// the use of fmod function
#include <stdio.h>
#include <math.h>

int main ()
{
   float a, b;
   int c;
   a = 8.2;
   b = 5.7;
   c = 3;
   printf("Remainder of %f / %d is %lf\n", a, c, fmod(a, c));
   printf("Remainder of %f / %f is %lf\n", a, b, fmod(a, b));

   return(0);
}
```

```
Remainder of 8.200000 / 3 is 2.200000
Remainder of 8.200000 / 5.700000 is 2.500000
```

7. **double sqrt(double x)**: The C library function double sqrt(double x) returns the square root of x.

```
syntax : double sqrt(double x)
```

```c
// C code to illustrate
// the use of sqrt function
#include <stdio.h>
#include <math.h>

int main ()
{

   printf("Square root of %lf is %lf\n", 225.0, sqrt(225.0) );
   printf("Square root of %lf is %lf\n", 300.0, sqrt(300.0) );

   return(0);
}
```

```
Square root of 225.000000 is 15.000000
Square root of 300.000000 is 17.320508
```

8. **double pow(double x, double y)**: The C library function double pow(double x, double y) returns x raised to the power of y i.e. xy.

```
syntax : double pow(double x, double y)
```

```c
// C code to illustrate
// the use of pow function
#include <stdio.h>
#include <math.h>

int main ()
{
   printf("Value 8.0 ^ 3 = %lf\n", pow(8.0, 3));

   printf("Value 3.05 ^ 1.98 = %lf", pow(3.05, 1.98));

   return(0);
}
```

```
Value 8.0 ^ 3 = 512.000000
Value 3.05 ^ 1.98 = 9.097324
```

9. **double modf(double x, double *integer)**: The C library function double modf(double x, double *integer) returns the fraction component (part after the decimal), and sets integer to the integer component.

```
syntax : double modf(double x, double *integer)
```

```c
// C code to illustrate
// the use of modf function
#include<stdio.h>
#include<math.h>

int main ()
{
   double x, fractpart, intpart;

   x = 8.123456;
   fractpart = modf(x, &intpart);

   printf("Integral part = %lf\n", intpart);
   printf("Fraction Part = %lf \n", fractpart);
```

```
    return(0);
}
```

Output:

```
Integral part = 8.000000
Fraction Part = 0.123456
```

10. **double exp(double x)**: The C library function double exp(double x) returns the value of e raised to the xth power.

```
syntax : double exp(double x)
```

```
// C code to illustrate
// the use of exp function
#include <stdio.h>
#include <math.h>

int main ()
{
   double x = 0;

   printf("The exponential value of %lf is %lf\n", x, exp(x));
   printf("The exponential value of %lf is %lf\n", x+1, exp(x+1));
   printf("The exponential value of %lf is %lf\n", x+2, exp(x+2));

   return(0);
}
```

```
The exponential value of 0.000000 is 1.000000
The exponential value of 1.000000 is 2.718282
The exponential value of 2.000000 is 7.389056
```

11. **double cos(double x)** : The C library function double cos(double x) returns the cosine of a radian angle x.

```
syntax : double cos(double x)
```

Note : Same syntax can be used for other trignometric functions like sin, tan etc.

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

#define PI 3.14159265
```

```c
int main ()
{
   double x, ret, val;

   x = 60.0;
   val = PI / 180.0;
   ret = cos( x*val );
   printf("The cosine of %lf is %lf degrees\n", x, ret);

   x = 90.0;
   val = PI / 180.0;
   ret = cos( x*val );
   printf("The cosine of %lf is %lf degrees\n", x, ret);

   return(0);
}
```

Output:

```
The cosine of 60.000000 is 0.500000 degrees
The cosine of 90.000000 is 0.000000 degrees
```

12. **double acos(double x) :** The C library function double acos(double x) returns the arc cosine of x in radians.

```
syntax : double acos(double x)
```

Note : Same syntax can be used for other arc trignometric functions like asin, atan etc.

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

#define PI 3.14159265

int main ()
{
   double x, ret, val;

   x = 0.9;
   val = 180.0 / PI;

   ret = acos(x) * val;
   printf("The arc cosine of %lf is %lf degrees", x, ret);

   return(0);
}
```

```
The arc cosine of 0.900000 is 25.855040 degrees
```

13. **double tanh(double x):** The C library function double tanh(double x) returns the hyperbolic tangent of x.

```
syntax : double tanh(double x)
```

Note : Same syntax can be used for other hyperbolic trignometric functions like sinh, cosh etc.

```c
// C code to illustrate
// the use of tanh function
#include <stdio.h>
#include <math.h>

int main ()
{
   double x, ret;
   x = 0.5;

   ret = tanh(x);
   printf("The hyperbolic tangent of %lf is %lf degrees", x, ret);

   return(0);
}
```

Output:

```
The hyperbolic tangent of 0.500000 is 0.462117 degrees
```