Math Library Functions

Function	Description	Example
`ceil( x )`	rounds x to the smallest integer not less than x	`ceil( 9.2 )` is 10.0 `ceil( -9.8 )` is -9.0
`cos( x )`	trigonometric cosine of x (x in radians)	`cos( 0.0 )` is 1.0
`exp( x )`	exponential function e^x	`exp( 1.0 )` is 2.718282 `exp( 2.0 )` is 7.389056
`fabs( x )`	absolute value of x	`fabs( 5.1 )` is 5.1 `fabs( 0.0 )` is 0.0 `fabs( -8.76 )` is 8.76
`floor( x )`	rounds x to the largest integer not greater than x	`floor( 9.2 )` is 9.0 `floor( -9.8 )` is -10.0
`fmod( x, y )`	remainder of x/y as a floating-point number	`fmod( 2.6, 1.2 )` is 0.2
`log( x )`	natural logarithm of x (base e)	`log( 2.718282 )` is 1.0 `log( 7.389056 )` is 2.0
`log10( x )`	logarithm of x (base 10)	`log10( 10.0 )` is 1.0 `log10( 100.0 )` is 2.0

Friend Function

- The friend function is declared using the **friend** keyword inside the body of the class. By using the keyword 'friend' compiler knows that the given function is a friend function.
- A friend function in C++ is defined as a function that can **access private, protected** and **public** members of a class.
- A friend function is a function that is declared outside the scope of a class.

- Example:

```
class className
{
... .. ...
//friend function declaration
friend return_type functionName(argument/s);
}
// friend function definition
return_type functionName(argument/s)
{
... .. ...    /* Private and protected data
               of className can be accessed from this function
               because it is a friend function
               */
}
```

Virtual Function

- A virtual function is a member function which is declared within a base class and is re-defined (Overridden) by a derived class.

Syntax:

- virtual void function_name()

Rules for Virtual Functions

1. Virtual functions cannot be static.
2. A virtual function can be a friend function of another class.
3. Virtual functions should be accessed using pointer or reference of base class type to achieve run time polymorphism.
4. The prototype of virtual functions should be the same in the base as well as derived class.

Friend Function	Virtual Function
It is non-member functions that usually have private access to class representation.	It is a base class function that can be overridden by a derived class.
It is used to access private and protected classes.	It is used to ensure that the correct function is called for an object no matter what expression is used to make a function class.
It is declared outside the class scope. It is declared using the '*friend*' keyword.	It is declared within the base class and is usually redefined by a derived class. It is declared using a '*virtual*' keyword.
It is generally used to give non-member function access to hidden members of a class.	It is generally required to tell the compiler to execute dynamic linkage of late binding on function.
They support sharing information of class that was previously hidden, provides method of escaping data hiding restrictions of C++, can access members without inheriting class, etc.	They support object-oriented programming, ensures that function is overridden, can be friend of other function, etc.
It can access private members of the class even while not being a member of that class.	It is used so that polymorphism can work.