Operator overloading

In C++, we can overload most operators so that they can perform special operations relative to classes that we create. For example, a class that maintains a stack might overload + + operator to perform a push operation and - - to perform a pop.

Important points to remember

- We overload operators by creating operator functions.
- An operator function defines the operations that the overloaded operator will perform relative to the class upon which it will work.
- An operator function is created using the keyword operator.
- Operator functions can be either members or non-members of a class.
- When an operator is overloaded, none of its original meanings are lost.

Some Operators that cannot be overloaded

Scope resolution operator (::)
sizeof operator
Ternary operator (?:)

Operators that cannot be overloaded by using a non-member operator function

Assignment operator (=)
Function call operator ( )
Array indexing operator [ ]
Arrow operator →

Parameters required while calling an operator function

Type of operator	**Member operator function**	**Non-member operator function**
Binary	Only one parameter is required as the first parameter is always available through the *this pointer	Two parameters required as a non-member function does not have a *this pointer
Unary	No parameter is required as the only parameter is implicitly available through the *this pointer.	One parameter is required as a non-member function does not have a *this pointer.