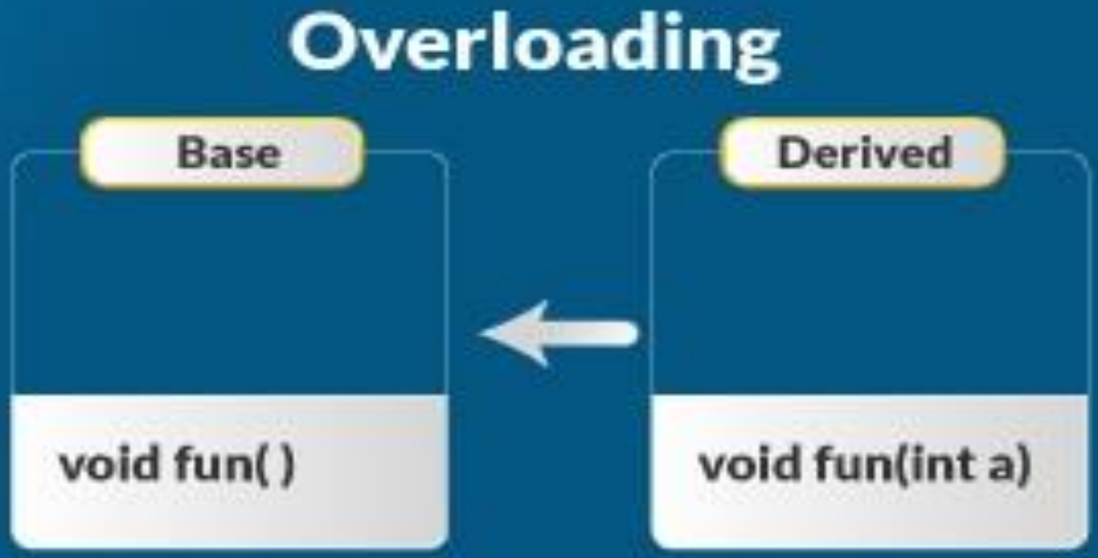
Operator Overloading In C++



By – Mohammad Imran

Operator Overloading

Similar to function overloading, OOPS enables the extra facility to overload some of the inbuilt operators present in C++. An operator can be overloaded by placing a keyword **‘`operator`’** just before the operator symbol.

Operator overloading is a type of static or compile-time polymorphism. C++ supports the compile-time polymorphism. The function overloading and the operator overloading are common examples of compile-time polymorphism.

Operator Overloading

Rules

- ✓ Only built-in operators like (+, -, *, /, etc.) can be overloaded.
- ✓ We cannot overload all of those operators that are not a part of C++ language like ‘$’.
- ✓ We can’t change the arity of the operators. The arity of an operator is the number of operands that the operator takes.
- ✓ We can overload the unary operator as an only unary operator, and we cannot overload it as a binary operator and similarly, We can overload binary operators as an only binary operator, and we cannot overload it as a unary operator.
- ✓ The precedence of the operators remains the same during operator overloading.

By – Mohammad Imran

Operator Overloading

Rules

- ✓ During the operator overloading, we cannot change the actual meaning of an operator. For example, We cannot overload the plus (+) operator to subtract one value form the other value.
- ✓ The operator overloading is not possible for built-in data types. At least one user-defined data types must be there.
- ✓ Some operators like assignment "=", address "&" and comma "," are by default overloaded.
- ✓ When using binary operators overloaded through a member function, the left-hand operand must be an object of the relevant class.

Operator Overloading

Operators that can be overloaded	Examples
Binary Arithmetic	+, -, *, /, %
Unary Arithmetic	+, -, ++, --
Assignment	=, +=,*=, /=,-=, %=
Bitwise	& , | , << , >> , ~ , ^
De-referencing	(->)
Dynamic memory allocation, De-allocation	New, delete
Subscript	[ ]
Function call	()
Logical	&, | |, !
Relational	>, < , = =, <=, >=

By – Mohammad Imran

Operator Overloading

List of operators that cannot be overloaded are mentioned below;

- ✓ Scope Resolution Operator (::)
- ✓ Pointer-to-member Operator (.*)
- ✓ Member Access or Dot operator (.)
- ✓ Ternary or Conditional Operator (?:)
- ✓ Object size Operator (sizeof)
- ✓ Object type Operator (typeid)

By – Mohammad Imran

Operator Overloading

Operator overloading can be done by implementing a function and the function can be a;

- ✓ Member Function
- ✓ Non-Member Function
- ✓ Friend Function

Operator Overloading

The Member Function and Non-Member Function:

Operator overloaded function can be a member function of class X if the Left operand is an object of that class X, but if the Left operand is different, then Operator overloading function must be a non-member function.

The Friend Function:

Operator overloaded function can be made friend function of class X if it needs access to the private and protected members of class X.

By – Mohammad Imran

Operator Overloading

Syntax

```
class <Class_Name>
{
   Return_Type operator Symbol ( Arguments )
   {
      Statements;
   }
};
```

By – Mohammad Imran

Unary Operator

Unary operators are the operators that perform operations on a single operand to produce a new value.

Types of Unary Operators

- ✓ Increment (++)
- ✓ Decrement (––)
- ✓ Unary Minus (–)
- ✓ Unary Plus (+)

Overloading Unary Operator (++) Pre

Example - 1

```
#include<iostream>
using namespace std;
class Overloading
{
    int data;
   public:
    Overloading()
    {
        data = 0;
    }
    void operator ++()
    {
        ++data;
    }
    void show()
    {
        cout << "Data = " << data << endl;
    }
};
int main()
{
    Overloading t1;
    t1.show();
    ++t1;
    t1.show();
    return 0;
}
```

OUTPUT

```
Data = 0
Data = 1
```

By – Mohammad Imran

Overloading Unary Operator (-) Example - 2

```
#include <iostream>
using namespace std;
class Distance
{
  private:
   int feet;
   int inches;
  public:
   Distance()
   {
      feet = 0;
      inches = 0;
   }
```

```
Distance(int f, int i)
{
   feet = f;
   inches = i;
}
void displayDistance()
{
   cout << "Feet: " << feet << endl;
   cout << "Inches:" << inches <<endl;
}
```

By – Mohammad Imran

Overloading Unary Operator (-)

Example - 2

```
   void operator -()
   {
      feet = -feet;
      inches = -inches;
   }
};
int main()
{
      Distance D1(11, 10), D2(-5, 15);
      -D1;
      D1.displayDistance();
      -D2;
      D2.displayDistance();
      return 0;
}
```

OUTPUT

```
Feet = -11
Inches = -10
Feet = 5
Inches = -15
```

By – Mohammad Imran

Overloading Binary Operator (+)

Example - 3

```
#include <iostream>
using namespace std;
class Employee
{
   public:
    int salary;
    Employee( int sal )
    {
       salary = sal;
    }
    void print( )
    {
       cout<< salary <<endl;
    }
```

```
   Employee operator +( Employee n )
   {
      return salary + n.salary;
   }
};


int main()
{
   Employee e1(20000);
   Employee e2(25000);
   Employee e3 = e1 + e2;
   cout<<"Salary Sum ="<< e3.salary;
   return 0;
}
```

OUTPUT

```
Salary Sum = 45000
```

By – Mohammad Imran

Operator Overloading

Relational Operator

Like unary and binary operator we can overload relational operator as well. Relational operator works with integer after overload these operator it will work with object also.

We can overload relational operator

<, >, <=, >=, ==

By – Mohammad Imran

Overloading Relational Operator (>)

Example - 4

```
#include <iostream>
using namespace std;
class Employee
{
   public:
    int salary;
    Employee( int sal )
    {
       salary = sal;
    }
    void print( )
    {
       cout << salary << endl;
    }
```

```
    bool operator > ( Employee n )
    {
       if(salary > n.salary)
       {
          return true;
       }
       else
       {
          return false;
       }
    }
};
```

By – Mohammad Imran

Overloading Relational Operator (>)

Example - 4

```
int main()
{
   Employee e1(20000);
   Employee e2(25000);
   if(e1 > e2)
      cout << "Employee e1 salary is more than employee e2.";
   else
      cout << "Employee e1 salary is less than employee e2.";
   return 0;
}
```

OUTPUT

```
Employee e1 salary is less than employee e2.
```

By – Mohammad Imran

Operator Overloading With String

Like overloading operator with numbers we can overload some operator to compare string also

We can overload relational operator with string

<, >, ==

Overloading Operator (==) with String

Example - 5

```
#include <iostream>
using namespace std;
class String
{
    char str[20];
   public:
    void getdata()
    {
        gets(str);
    }
    int operator ==(String s)
    {
        if(!strcmp(str,s.str))
            return 1;
        return 0;
    }
};
```

By – Mohammad Imran

Overloading Operator (==) with String

Example - 5

```
int main()
{
   String s1,s2;
   cout << "Enter first string :: ";
   s1.getdata();
   cout << "Enter second string :: ";
   s2.getdata();
   if(s1 == s2)
   {
      cout << "Strigs are Equal\n";
   }
   else
   {
      cout << "Strings are Not Equal\n";
   }
   return 0;
}
```

OUTPUT

```
Enter first string: Hello
Enter second string: Hello
Strings are Equal
```

By – Mohammad Imran

Overloading Unary Operator (-) with Friend Function

Example - 6

```
#include <iostream>
using namespace std;
class Sign
{
  private:
int num;
  public:
   Sign()
   {
     num = 0;
   }
   Sign(int s)
   {
     num = s;
   }
   void displaySign()
   {
     cout << "Num sign = " << num << endl;
   }
   friend void operator -(Sign);
};
void operator -(Sign d)
{
   d.num = -d.num;
   cout << "Num sign = " << d.num << endl;
}
```

By – Mohammad Imran

Overloading Unary Operator (-) with Friend Function

Example - 6

```
int main()
{
     Sign s1(11);
     s1.displaySign();
     -s1;
     return 0;
}
```

OUTPUT

```
Num sign = 11
Num sign = -11
```

By – Mohammad Imran

Coding Questions

By – Mohammad Imran

Question - 1

Write a C++ program to overload unary pre-increment operator (++) to increment by default 2 when used with object.

Sample Output

```
X = 10
++obj
X = 12
```

Click here to see code

By – Mohammad Imran

Question - 2

Write a C++ program to overload binary minus (-) operator using reference and friend function.

Click here to see code

By – Mohammad Imran

Question - 3

Write a C++ program to overload unary (++) operator as a post increment with object.

Click here to see code

By – Mohammad Imran

QUIZ

By – Mohammad Imran

Quiz - 1

```
#include <iostream>
using namespace std;
int main()
{
   int x = 10;
   char y = 'a';
   x = x + y;
   float z = x + 1.0;
   cout << "x = " << x << endl;
   cout << "y = " << y << endl;
   cout << "z = " << z << endl;
   return 0;
}
```

ANSWER

```
x = 107
y = a
z = 108
```

By – Mohammad Imran