Subject Name: **Object Oriented Programming Using C++**

Subject Code: **BCA-301 N**

Subject Topic: **Introduction to Operator Overloading**

Abhishek Dwivedi

Assistant Professor

Department of Computer Application

UIET, CSJM University, Kanpur

Operator Overloading in C++

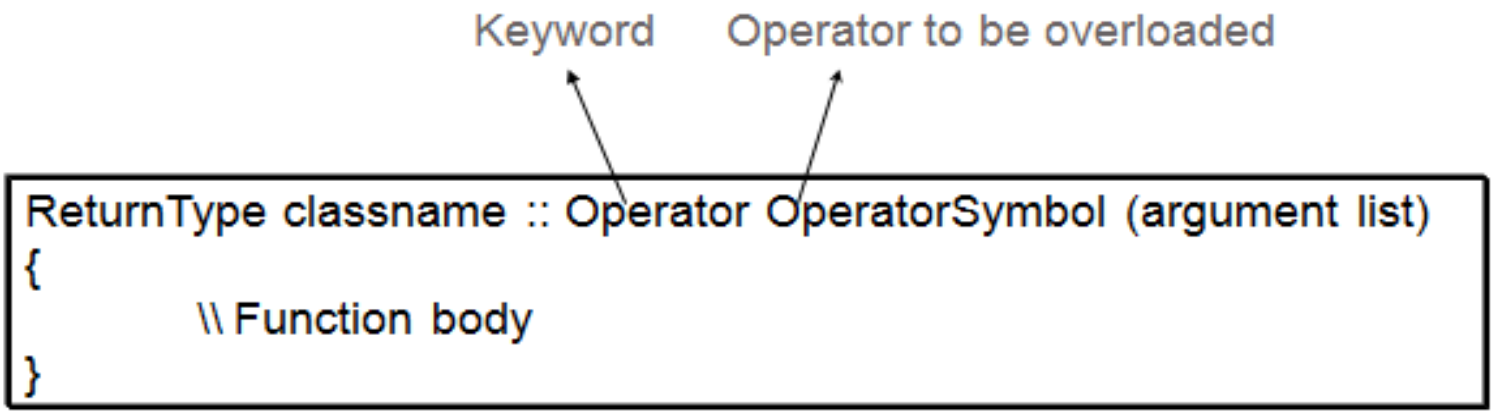
- Operator overloading is an important concept in C++. It is a type of polymorphism in which an operator is overloaded to give user defined meaning to it. Overloaded operator is used to perform operation on user-defined data type. For example '+' operator can be overloaded to perform addition on various data types, like for Integer, String(concatenation) etc.

Operator that are not overloaded

- Almost any operator can be overloaded in C++. However there are few operator which can not be overloaded. Operator that are not overloaded are follows

1. scope operator - **::**
2. sizeof
3. member selector - **.**
4. member pointer selector - ***
5. ternary operator - **?:**

Operator Overloading Syntax



Implementing Operator Overloading in C++

- Operator overloading can be done by implementing a function which can be :

1. Member Function
2. Non-Member Function
3. Friend Function

- Operator overloading function can be a member function if the Left operand is an Object of that class, but if the Left operand is different, then Operator overloading function must be a non-member function.
- Operator overloading function can be made friend function if it needs access to the private and protected members of class.

Restrictions on Operator Overloading in C++

- Following are some restrictions to be kept in mind while implementing operator overloading.

1. Precedence and Associativity of an operator cannot be changed.

2. Numbers of Operands cannot be changed. Unary operator remains unary, binary remains binary etc.

3. No new operators can be created, only existing operators can be overloaded.

Overloading Arithmetic Operator in C++

- Almost all arithmetic operator can be overloaded to perform arithmetic operation on user-defined data type. Here we have overloading the + operator, to add two Time(hh:mm:ss) objects.

```
class Time
 {
    public:
     int h,m,s;
    Time()
    { h=0, m=0; s=0; }
    void setTime();
    void show()
    { cout<< h<< ":"<< m<< ":"<< s; }
    Time operator+(time);       //overloading '+' operator
 };
```

```
Time Time::operator+(Time  t1) //operator function
 {
Time t;
int a,b;
a = s+t1.s;
t.s = a%60;
b = (a/60)+m+t1.m;
t.m = b%60;
 t.h = (b/60)+h+t1.h;
 t.h = t.h%12;
 return t;
 }
```

```
void time::setTime()
{
cout << "\n Enter the hour(0-11) ";
cin >> h;
cout << "\n Enter the minute(0-59) ";
cin >> m;
cout << "\n Enter the second(0-59) ";
 cin >> s;
 }
```

```
void main()
{
Time t1,t2,t3;
cout << "\n Enter the first time ";
 t1.setTime();
cout << "\n Enter the second time ";
t2.setTime();
t3 = t1 + t2;                    //adding of two time object using '+' operator
cout << "\n First time ";
 t1.show();
 cout << "\n Second time ";
 t2.show();
cout << "\n Sum of times ";
t3.show();
getch();
 }
```

References:

- www.studytonight.com
- www.tutorialpoint.com
- www.geeksforgeeks.org
- “Object oriented programming in C++”, Robert Lafore
- “Object oriented programming with C++”, E.Balagurusamy