**Unit - 12**

**OPERATOR OVERLAODING AND TYPE CONVERSION**

When you try to add two user define (class type ) objects the Complier will give us an error. So in order to make the above operation work for the variable of user – defined type. We need to overload the + operator explicitly by making a function to perform the desired task. This property of “ **giving additional meaning to existing operator so that they can work with variable of user-defined types is called operator overloading “**. Providing additional meaning to an existing operator by overloading it doesn’t change the original meaning of the operator but it’s **simply extends the functionality of the operator.**

**Syntax of operator Overloading :-**

**Return\_type classname::operator +( Arguments\_List)**

**{**

}

Where:-

Return\_Type :- name of class

Operator : - it’s a keyword. The complier distinguish an operator function from

Ordinary member function of the class by the keyword **operator.**

Argument\_list:-the no of arguments passed to the operator function.

**IMP Note**: - The no of arguments depends upon whether the overloaded operator is **Unary**

Or **Binary Operator**. If the operator is unary then arguments list is empty and

if it’s a binary operator then argument list contain one parameter.

**Unary operator overloading**

The unary operator works only on a single operand. Some common unary operator are include unary plus(+) , unary mins(-),++ , -- and ! etc.

**Program(1):- Operator overloading of unary operator (++) without return type**

#include<iostream>

using namespace std;

class aa

{

public:

int val;

aa( )

{

val=0;

}

void operator++( ) // operator overloading definition

{

val=val+3;

}

int show( )

{

return val;

}

};

int main()

{

aa obj1,obj2;

cout<<"\nBefore overloading :"<<obj1.show();

cout<<"\nBefore overloading :"<<obj2.show();

++obj1; //operator function calls

++obj2; //same as obj2.operator++( )

++obj2;

cout<<"\nAfter overloading :"<<obj1.show();

cout<<"\nAfter overlaoding :"<<obj2.show();

}

**Program(2):- Operator overloading of unary operator (++) with return type**

#include<iostream>

using namespace std;

class aa

{

public:

int val;

aa()

{

val=0;

}

aa operator++()

{

val=val+3;

}

int show()

{

return val;

}

};

int main()

{

aa obj1,obj2;

cout<<"\nBefore overloading :"<<obj1.show();

cout<<"\nBefore overloading :"<<obj2.show();

++obj1;

obj2=obj1; // data member

cout<<"\nAfter overloading :"<<obj1.show();

cout<<"\nAfter overlaoding :"<<obj2.show();

}