**PRACTICAL # 07**

**OBJECT:**

C# Operator Overloading

**THEORY:**

C# supports operator overloading, which is a way to define the meaning of an operator relative to a user defined class. For instance, if you create a class called Matrix and want to define add operation, you can overload the + operator perform this action.

To overload an operator, use ***operator*** keyword to define an operator method, which defines the action of the operator relative to its class. There are two forms of operator methods: one for unary operators and one for binary operators. The general form for each is shown here:

*// General form for overloading a unary operator*

*public static ret-type operator op(param-type operand){*

*// operations*

*}*

*// General form for overloading a binary operator*

*public static ret-type operator op(param-type1 operand1, param-type1 operand2){*

*// operations*

*}*

**Program:**

**Overloading Binary Operators:**

*// A three-dimensional coordinate class.*

*class ThreeD {*

*int x, y, z; // 3-D coordinates*

*public ThreeD() { x = y = z = 0; }*

*public ThreeD(int i, int j, int k) { x = i; y = j; z = k; }*

*// Overload binary +.*

*public static ThreeD operator +(ThreeD op1, ThreeD op2)*

*{*

*ThreeD result = new ThreeD();*

*/\* This adds together the coordinates of the two points*

*and returns the result. \*/*

*result.x = op1.x + op2.x; // These are integer additions*

*result.y = op1.y + op2.y; // and the + retains its original*

*result.z = op1.z + op2.z; // meaning relative to them.*

*return result;*

*}*

*// Overload binary -.*

*public static ThreeD operator -(ThreeD op1, ThreeD op2)*

*{*

*ThreeD result = new ThreeD();*

*result.x = op1.x - op2.x; // these are integer subtractions*

*result.y = op1.y - op2.y;*

*result.z = op1.z – op2.z;*

*return result;*

*}*

*// Show X, Y, Z coordinates.*

*public void Show()*

*{*

*Console.WriteLine(x + ", " + y + ", " + z);*

*}*

*}*

*static void Main() {*

*ThreeD a = new ThreeD(1, 2, 3);*

*ThreeD b = new ThreeD(10, 10, 10);*

*ThreeD c;*

*c = a + b; // add a and b together*

*Console.Write("Result of a + b: ");*

*c.Show();*

*Console.WriteLine();*

*c = a + b + c; // add a, b, and c together*

*Console.Write("Result of a + b + c: ");*

*c.Show();*

*Console.WriteLine();*

*c = c - a; // subtract a*

*Console.Write("Result of c - a: ");*

*c.Show();*

*Console.WriteLine();*

*c = c - b; // subtract b*

*Console.Write("Result of c - b: ");*

*c.Show();*

*Console.WriteLine();*

*}*

**Overloading Unary Operators:**

*// Overload unary -.*

*public static ThreeD operator -(ThreeD op)*

*{*

*ThreeD result = new ThreeD();*

*result.x = -op.x;*

*result.y = -op.y;*

*result.z = -op.z;*

*return result;*

*}*

//usage

*a = -b*

*// Overload unary ++.*

*public static ThreeD operator ++(ThreeD op)*

*{*

*ThreeD result = new ThreeD();*

*// Return the incremented result.*

*result.x = op.x + 1;*

*result.y = op.y + 1;*

*result.z = op.z + 1;*

*return result;*

*}*

**ACTIVITIES**

**Activity 1**

Define a matrix class, overloading add (+) and multiply (\*) operators.

**REVIEW QUESTIONS**

1. What is operator overloading and how it compares to function overloading?
2. Which keyword is used for operator overloading?
3. Which operators in C# can be overloaded?
4. How can we overload unary operators?