Sobrecarga de Operadores

En C++, la declaración y definición de una sobrecarga de operador es muy similar a la declaración y definición de una función cualquiera.

La sobrecarga de operadores, de modo similar que la sobrecarga de funciones, permite al programador dar nuevos significados a los símbolos de los operadores ya existentes en C++.

El resultado debe ser la creación de un código más fácil de leer y usar.

El ejemplo más sencillo de una sobrecarga de operadores nos lo da el lenguaje mismo, es decir, en una operación aritmética (por ejemplo, una **suma**) el compilador determina el tipo de operación requerida de acuerdo con el tipo de datos involucrados.

```
int a, b;
double x, y;
float r;
r = a + b;
r = a + x;
r = x + y;
```

La sobrecarga de operadores nos permitirá hacer cosas como estas, observe en el siguiente ejemplo como los operadores señalados están funcionando de una manera atípica de lo acostumbrado. Este funcionamiento es permitido al sobrecargar los operadores dentro de la clase.

```
void main()
{
	Fraccion *f1 = new Fraccion (2,6);
	Fraccion *f2 = new Fraccion (3,7);
	Fraccion res;
	Coleccion cole(10); // supongamos una colección de fracciones

	res =  (*f1) + (*f2);
	cout<< res;

	res =(*f1)  - (*f2);
	cout<< res ;

	res =(*f1)  * (*f2);
	cout<< res;

	res++;
	cout<< res;

	res--;
	cout<< res;
```

```
        cole[0]= (*f1);
        cole[1]= (*f2);
        cout<< cole;
 }
```

Los operadores se sobrecargan escribiendo una función de manera habitual, excepto que el nombre de la función será la palabra clave **operator,** seguida por el símbolo del operador que se desee sobrecargar.

Por ejemplo el nombre de la función operator+ sirve para sobrecargar el operador de suma (+), veamos el siguiente prototipo de método:

```
Persona&  operator + (const Persona &);
```

Restricciones en los operadores sobrecargados

1.- Sólo se pueden sobrecargar operadores definidos en C++.
2.- La sobrecarga de operadores sólo funciona con objetos del alguna clase
3.- No se puede cambiar la preferencia los operadores definidos en C++.
4.- No se puede hacer funcionar a un operador binario como unitario ni viceversa.
5.- No se puede sobrecargar un operador que funcione exclusivamente con apuntadores.
6.- Es responsabilidad del programador definir correctamente el significado de las funciones sobrecargadas.

Pueden sobrecargarse los operadores:

+, -, * , /, %, ^ , &, |, (,), , <= , >= , <>, == , != , &&, | |, = , +=, -=, *= , /= , %= , ^ = , &= , |= , <<= ,>>= , [], (),->, new y delete.

Estos no se pueden sobrecargar

".", ".*", "::" y "?:".

Los operadores =, [], ->, (), new y delete, sólo pueden ser sobrecargados cuando se definen como miembros de una clase.

Antes de sobrecargar cualquier operador para una clase se deben tener en cuenta los siguientes factores:

- Los operadores **binarios** se declaran con un solo parámetro, ya que el primer parámetro es pasado por el programa como **this**, es decir, un puntero al mismo objeto.
- Los operadores **unarios** se declaran sin paramétros, ya que el único parámetro es pasado por el programa como **this**.

  Nota: Los operadores binarios son aquellos que poseen dos partes (izquierda y derecha), por ejemplo, una operación de suma requiere dos operandos (x + y). Los operadores unarios son aquellos que poseen solo una parte, por ejemplo, una operación de incremento (x ++).

Los métodos de los operadores de una clase específica, se llaman sólo cuando el operando de la izquierda de del operador binario, es un objeto de esa clase, o cuando el operando de un operador unario es un objeto de esa clase.