Sobrecarga de Operadores

C++ permite sobreescribir operadores, con el fin de mejorar la legibilidad del código.

Los operadores que podemos sobrecargar son : :+ - * / % ^ & | ~ ! = < > += -= *= /= %= ^= &= |= << >> >>= <<= == != <= >= <=> && || ++ -- , ->* -> ( ) [ ]

Veamos un ejemplo:

```
Fraccion& Fraccion::operator*=(const Fraccion& rhs)
{
   int new_n = n * rhs.n/gcd(n * rhs.n, d * rhs.d);
   d = d * rhs.d/gcd(n * rhs.n, d * rhs.d);
   n = new_n;
   return *this;
}
```

Restricciones

•Los operadores :: (resolución del alcance),. (acceso de miembros),. * (acceso de miembros a través del puntero a miembro) y?: (condicional ternario) no se pueden sobrecargar.

•No es posible cambiar la precedencia, la agrupación o el número de operandos de los operadores.

•La sobrecarga del operador -> debe devolver un puntero sin formato o devolver un objeto (por referencia o por valor) para el que el operador -> está a su vez sobrecargado.

•Las sobrecargas de operadores && y || pierden la evaluación lazy.

•&&, || y, (coma) pierden sus propiedades especiales de secuenciación cuando se sobrecargan y se comportan como llamadas de función normales incluso cuando se usan sin la notación de llamada de función.

Sobrecarga del operador << y >>

•Las sobrecargas de operator >> y operator << que toman std :: istream & o std :: ostream & como argumento de la izquierda y retorna el mismo tipo:

```
std::ostream& operator<<(std::ostream& os, const T& obj)
{
    // write obj to stream
    return os;
}

std::istream& operator>>(std::istream& is, T& obj)
{
    // read obj from stream
    if( /* T could not be constructed */ )
        is.setstate(std::ios::failbit);
    return is;
}
```