Lesson 29 Operator overloading (OOP)

In C++, we can change the way operators work for user-defined types like objects and structures. This is known as operator overloading.

To overload an operator, we use a special operator function. We define the function inside the class or structure whose objects/variables we want the overloaded operator to work with.

Example: **++** Operator Overloading

```
#include <iostream>

using namespace std;


class Count {

  private:

   int value;


  public:


   // Constructor to initialize count to 5

   Count() : value(5) {}
```

```
    // Overload ++ when used as prefix
    void operator ++ () {
        ++value;
    }

    void display() {
        cout << "Count: " << value << endl;
    }
};

int main() {
    Count count1;

    // Call the "void operator ++ ()" function
    ++count1;

    count1.display();
    return 0;
}
```

Output:

```
count : 6
```