Chapter 72. Boost.Operators

Boost.Operators provides numerous classes to automatically overload operators. In Example 72.1, a greater-than operator is automatically added, even though there is no declaration, because the greater-than operator can be implemented using the already defined less-than operator.

Example 72.1. Greater-than operator with `boost::less_than_comparable`

```
#include <boost/operators.hpp>
#include <string>
#include <utility>
#include <iostream>

struct animal : public boost::less_than_comparable<animal>
{
  std::string name;
  int legs;

  animal(std::string n, int l) : name{std::move(n)}, legs{l} {}

  bool operator<(const animal &a) const { return legs < a.legs; }
};

int main()
{
  animal a1{"cat", 4};
  animal a2{"spider", 8};

  std::cout << std::boolalpha << (a2 > a1) << '\n';
}
```

To automatically add operators, derive a class from classes defined by Boost.Operators in `boost/operators.hpp`. If a class is derived from `boost::less_than_comparable`, then `operator>`, `operator<=`, and `operator>=` are automatically defined.

Because many operators can be expressed in terms of other operators, automatic overloading is possible. For example, `boost::less_than_comparable` implements the greater-than operator as the opposite of the less-than operator; if an object isn't less than another, it must be greater, assuming they aren't equal.

If two objects can be equal, use `boost::partially_ordered` as the base class. By defining `operator==`, `boost::partially_ordered` can determine whether less than really means greater than or equal.

In addition to `boost::less_than_comparable` and `boost::partially_ordered`, classes are provided that allow you to overload arithmetic and logical operators. Classes are also available to overload operators usually provided by iterators, pointers, or arrays. Because automatic overloading is only possible once other operators have been defined, the particular operators that must be provided will vary depending on the situation. Consult the documentation for more information.