Actually, this notation is a bit confusing. It would be clearer if you could somehow use the relational operator \( > \) to compare the two objects. You can do so with operator overloading, which Chapter 11 discusses.

Meanwhile, there's still the implementation of `topval()` to attend to. That raises a slight problem. Here's a partial implementation that highlights the problem:

```cpp
const Stock & Stock::topval(const Stock & s) const {
    if (s.total_val > total_val)
        return s;       // argument object
    else
        return ?????;  // invoking object
}
```

Here `s.total_val` is the total value for the object passed as an argument, and `total_val` is the total value for the object to which the message is sent. If `s.total_val` is greater than `total_val`, the function returns `s`. Otherwise, it returns the object used to evoke the method.