maids = (cooks = 4) + 3;

The expression cooks = 4 has the value 4, so maids is assigned the value 7. However, just because C++ permits this behavior doesn't mean you should encourage it. But the same rule that makes this peculiar statement possible also makes the following useful statement possible:

x = y = z = 0;

This is a fast way to set several variables to the same value. The precedence table (Appendix D, "Operator Precedence") reveals that assignment associates right-to-left, so first 0 is assigned to z, and then the value of z = 0 is assigned to y, and so on.

Finally, as mentioned before, relational expressions such as x < y evaluate to the bool values true or false. The short program in Listing 5.3 illustrates some points about expression values. The << operator has higher precedence than the operators used in the expressions, so the code uses parentheses to enforce the correct order.

**Listing 5.3 express.cpp**

```cpp
// express.cpp -- values of expressions
#include <iostream>
using namespace std;
int main()
{  
    int x;

    cout << "The expression x = 100 has the value ";
    cout << (x = 100) << "\n";
    cout << "Now x = " << x << "\n";
    cout << "The expression x < 3 has the value ";
    cout << (x < 3) << "\n";
    cout << "The expression x > 3 has the value ";
    cout << (x > 3) << "\n";
    cout.setf(ios_base::boolalpha);  //a newer C++ feature
    cout << "The expression x < 3 has the value ";
    cout << (x < 3) << "\n";
```