Redundant Parentheses

As in algebra, it's acceptable to place *unnecessary* parentheses in an expression to make the expression clearer. These are called **redundant parentheses**. For example, the preceding assignment statement could be parenthesized as follows:

```
y = ( a * x * x ) + ( b * x ) + c;
```

2.7 Decision Making: Equality and Relational Operators

We now introduce a simple version of C++'s **`if` statement** that allows a program to take alternative action based on whether a **condition** is true or false. If the condition is *true*, the statement in the body of the `if` statement *is* executed. If the condition is *false*, the body statement *is not* executed. We'll see an example shortly.

Conditions in `if` statements can be formed by using the **relational operators** and **equality operators** summarized in Fig. 2.12. The relational operators all have the same level of precedence and associate left to right. The equality operators both have the same level of precedence, which is *lower* than that of the relational operators, and associate left to right.

Algebraic relational or equality operator	C++ relational or equality operator	Sample C++ condition	Meaning of C++ condition
Relational operators			
>	>	`x > y`	x is greater than y
<	<	`x < y`	x is less than y
≥	>=	`x >= y`	x is greater than or equal to y
≤	<=	`x <= y`	x is less than or equal to y
Equality operators			
=	==	`x == y`	x is equal to y
≠	!=	`x != y`	x is not equal to y

Fig. 2.12 | Relational and equality operators.

Common Programming Error 2.3

Reversing the order of the pair of symbols in the operators `!=`, `>=` and `<=` (by writing them as `=!`, `=>` and `=<`, respectively) is normally a syntax error. In some cases, writing `!=` as `=!` will not be a syntax error, but almost certainly will be a logic error that has an effect at execution time. You'll understand why when you learn about logical operators in Chapter 5. A fatal logic error causes a program to fail and terminate prematurely. A nonfatal logic error allows a program to continue executing, but usually produces incorrect results.

Common Programming Error 2.4

Confusing the equality operator `==` with the assignment operator `=` results in logic errors. We like to read the equality operator as "is equal to" or "double equals," and the assignment operator as "gets" or "gets the value of" or "is assigned the value of." As you'll see in Section 5.9, confusing these operators may not necessarily cause an easy-to-recognize syntax error, but may cause subtle logic errors.