Factorials get big fast!

**Program Notes**

The program creates an array to hold the factorial values. Element 0 is 0!, element 1 is 1!, and so on. Because the first two factorials equal 1, the program sets the first two elements of the `factorials` array to 1.0. (Remember, the first element of an array has an index value of 0.) After that, the program uses a loop to set each factorial to the product of the index with the previous factorial. The loop illustrates that you can use the loop counter as a variable in the body of the loop.

The program demonstrates how the `for` loop works hand in hand with arrays by providing a convenient means to access each array member in turn. Also, `formore.cpp` uses `const` to create a symbolic representation (`ArSize`) for the array size. Then, it uses `ArSize` wherever the array size comes into play, such as in the array definition and in the limits for the loops handling the array. Now, if you wish to extend the program to, say, 20 factorials, you just have to set `ArSize` to 20 in the program and recompile. By using a symbolic constant, you avoid having to change every occurrence of 16 to 20 individually.

**Tip**

It's usually a good idea to define a `const` value to represent the number of elements in an array. Use the `const` value in the array declaration and in all other references to the array size, such as in a `for` loop.

The limit `i < ArSize` expression reflects the fact that subscripts for an array with `ArSize` elements run from 0 to `ArSize - 1`, so the array index should stop 1 short of `ArSize`. You could use the test `i <= ArSize - 1` instead, but it looks awkward in comparison.

One program sidelight is that it declares the `const int` variable `ArSize` outside the body of `main()`. As the end of Chapter 4, "Compound Types," mentions, this makes `ArSize` external data. The two consequences of declaring `ArSize` in this fashion are that `ArSize` exists for the duration of the program and that all functions in the program file can use it. In this particular case, the program has just one function, so declaring `ArSize` externally has little practical effect. But multifunction programs often benefit from sharing external