show_array() treats the array as read-only data. Suppose you accidentally violate this restriction by doing something like the following in the show_array() function:

```
ar[0] += 10;
```

Then, the compiler will put a stop to your wrongful ways. Borland C++, for example, gives an error message like this (edited slightly):

```
Cannot modify a const object in function
    show_array(const double *, int)
```

The message reminds us that C++ interprets the declaration `const double ar[]` to mean `const double *ar`. Thus, the declaration really says that `ar` points to a constant value. We'll discuss this in detail when we finish with the current example. Meanwhile, here is the code for the show_array() function:

```c
void show_array(const double ar[], int n)
{
    for (int i = 0; i < n; i++)
    {
        cout << "Property #" << (i + 1) << ": 
        cout << ar[i] << "\n";
    }
}
```

**Modifying the Array**

The third operation for our array is multiplying each element by the same revaluation factor. You need to pass three arguments to the function: the factor, the array, and the number of elements. No return value is needed, so the function can look like this:

```c
void revalue(double r, double ar[], int n)
{
    for (int i = 0; i < n; i++)
        ar[i] *= r;
}
```