7.4.2 Initializing an `array` in a Declaration with an Initializer List

- The elements of an array also can be initialized in the array declaration by following the array name with an equals sign and a brace-delimited comma-separated list of initializers.
- The program in Fig. 7.4 uses an initializer list to initialize an integer array with five values (line 11) and prints the array in tabular format (lines 13–17).
- If there are *fewer* initializers than elements in the `array`, the remaining `array` elements are initialized to zero.

```
1  // Fig. 7.4: fig07_04.cpp
2  // Initializing an array in a declaration.
3  #include <iostream>
4  #include <iomanip>
5  #include <array>
6  using namespace std;
7
8  int main()
9  {
10    // use list initializer to initialize array n
11    array< int, 5 > n = { 32, 27, 64, 18, 95 };
12
13    cout << "Element" << setw( 13 ) << "Value" << endl;
14
15    // output each array element's value
16    for ( size_t i = 0; i < n.size(); ++i )
17       cout << setw( 7 ) << i << setw( 13 ) << n[ i ] << endl;
18 } // end main
```

Fig. 7.4 | Initializing an `array` in a declaration. (Part 1 of 2.)

```
Element         Value
      0            32
      1            27
      2            64
      3            18
      4            95
```

Fig. 7.4 | Initializing an `array` in a declaration. (Part 2 of 2.)

7.4.3 Specifying an `array`'s Size with a Constant Variable and Setting `array` Elements with Calculations

- Figure 7.5 sets the elements of a 5-element array `s` to the even integers `2`, `4`, `6`, `8` and `10` (lines 15–16) and prints the array in tabular format (lines 18–22).
- Line 11 uses the `const` qualifier to declare a constant variable `arraySize` with the value `5`.
- A constant variable that's used to specify `array`'s size *must* be initialized with a constant expression when it's declared and

```
1  // Fig. 7.5: fig07_05.cpp
2  // Set array s to the even integers from 2 to 10.
3  #include <iostream>
4  #include <iomanip>
5  #include <array>
6  using namespace std;
7
8  int main()
9  {
10    // constant variable can be used to specify array size
11    const size_t arraySize = 5; // must initialize in declaration
12
13    array< int, arraySize > s; // array s has 5 elements
14
15    for ( size_t i = 0; i < s.size(); ++i ) // set the values
16       s[ i ] = 2 + 2 * i;
17
18    cout << "Element" << setw( 13 ) << "Value" << endl;
19
20    // output contents of array s in tabular format
21    for ( size_t j = 0; j < s.size(); ++j )
22       cout << setw( 7 ) << j << setw( 13 ) << s[ j ] << endl;
23 } // end main
```

Fig. 7.5 | Set `array s` to the even integers from 2 to 10. (Part 1 of 2.)

```
Element         Value
      0             2
      1             4
      2             6
      3             8
      4            10
```

Fig. 7.5 | Set `array s` to the even integers from 2 to 10. (Part 2 of 2.)

Common Programming Error 7.2

Not initializing a constant variable when it's declared is a compilation error.

Common Programming Error 7.3

Assigning a value to a constant variable in an executable statement is a compilation error.

```
1  // Fig. 7.6: fig07_06.cpp
2  // Using a properly initialized constant variable.
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8     const int x = 7; // initialized constant variable
9
10    cout << "The value of constant variable x is: " << x << endl;
11 } // end main
```

```
The value of constant variable x is: 7
```

Fig. 7.6 | Using a properly initialized constant variable.

```
1  // Fig. 7.7: fig07_07.cpp
2  // A const variable must be initialized.
3
4  int main()
5  {
6     const int x; // Error: x must be initialized
7
8     x = 7; // Error: cannot modify a const variable
9  } // end main
```

Fig. 7.7 | A `const` variable must be initialized. (Part 1 of 2.)

Microsoft Visual C++ compiler error message:

```
error C2734: 'x' : const object must be initialized if not extern
error C3892: 'x' : you cannot assign to a variable that is const
```

GNU C++ compiler error message:

```
fig07_07.cpp:6:14: error: uninitialized const ‘x’ [-fpermissive]
fig07_07.cpp:8:8: error: assignment of read-only variable ‘x’
```

LLVM compiler error message:

```
Default initialization of an object of const type 'const int'
```

Fig. 7.7 | A `const` variable must be initialized. (Part 2 of 2.)

Good Programming Practice 7.1

Defining the size of an `array` as a constant variable instead of a literal constant makes programs clearer. This technique eliminates so-called magic numbers—numeric values that are not explained. Using a constant variable allows you to provide a name for a literal constant and can help explain the purpose of the value in the program.

7.4.4 Summing the Elements of an `array`

- Often, the elements of an array represent a series of values to be used in a calculation.
- The program in Fig. 7.8 sums the values contained in the four-element integer `array` `a`.

```
1  // Fig. 7.8: fig07_08.cpp
2  // Compute the sum of the elements of an array.
3  #include <iostream>
4  #include <array>
5  using namespace std;
6
7  int main()
8  {
9     const size_t arraySize = 4; // specifies size of array
10    array< int, arraySize > a = { 10, 20, 30, 40 };
11    int total = 0;
12
13    // sum contents of array a
14    for ( size_t i = 0; i < a.size(); ++i )
15       total += a[ i ];
16
17    cout << "Total of array elements: " << total << endl;
18 } // end main
```

```
Total of array elements: 100
```

Fig. 7.8 | Computing the sum of the elements of an `array`.

7.4.5 Using Bar Charts to Display `array` Data Graphically

- Many programs present data to users in a graphical manner.
- One simple way to display numeric data graphically is with a bar chart that shows each numeric value as a bar of asterisks (*).
- Our next program (Fig. 7.9) stores data in an `array` of 11 elements, each corresponding to a grade category.