The only problem with this arrangement is that b must be declared as a variable and then initialized prior to the call. If b isn't needed for any other purpose, it can be mildly annoying to create it solely for the purpose of calling sum_array.

In C99, we can avoid this annoyance by using a ***compound literal:*** an unnamed array that's created "on the fly" by simply specifying which elements it contains. The following call of sum_array has a compound literal (shown in **bold**) as its first argument:

```
total = sum_array((int []){3, 0, 3, 4, 1}, 5);
```

In this example, the compound literal creates an array containing the five integers 3, 0, 3, 4, and 1. We didn't specify the length of the array, so it's determined by the number of elements in the literal. We also have the option of specifying a length explicitly: (int [4]){1, 9, 2, 1} is equivalent to (int []){1, 9, 2, 1}.

In general, a compound literal consists of a type name within parentheses, followed by a set of values enclosed by braces. A compound literal resembles a cast applied to an initializer. In fact, compound literals and initializers obey the same rules. A compound literal may contain designators, just like a designated initializer, and it may fail to provide full initialization (in which case any uninitialized elements default to zero). For example, the literal (int [10]){8, 6} has 10 elements; the first two have the values 8 and 6, and the remaining elements have the value 0.

designated initializers ➤*8.1*

Compound literals created inside a function may contain arbitrary expressions, not just constants. For example, we could write

```
total = sum_array((int []){2 * i, i + j, j * k}, 3);
```

where i, j, and k are variables. This aspect of compound literals greatly enhances their usefulness.

lvalues ➤*4.2*

A compound literal is an lvalue, so the values of its elements can be changed. If desired, a compound literal can be made "read-only" by adding the word const to its type, as in (const int []){5, 4}.

9.4 The return Statement

A non-void function must use the return statement to specify what value it will return. The return statement has the form

return statement

return *expression* ;

The expression is often just a constant or variable:

```
return 0;
return status;
```