Any function which is not declared as void should include a **return** statement. The value or expression in the return statement is that which is returned to the calling function. In the **celsius_to_fahrenheit** function, the value stored in **fahr** is returned to the calling function.

The program below shows how you can use this function.

```
#include<iostream.h>
int main()
 {
  float fahrenheit;
  float celsius = 22.5;

  fahrenheit = celsius_to_fahrenheit(celsius);

  cout << celsius << " C = " << fahrenheit << " F\n";
  return 0;
 }
```

Note

You can use any data type when declaring a function.

The statement **fahrenheit = celsius_to_fahrenheit(celsius);** calls the **celsius_to_fahrenheit** function and passes the value in the variable **celsius** to the function. The function returns the temperature in Fahrenheit degrees and the calling statement assigns the Fahrenheit temperature to the variable **fahrenheit**.

The **celsius_to_fahrenheit** function could be rewritten to include only one statement and return the same result. Any valid expression can appear in the parentheses of the return statement. In this case, the local variable **fahr** is eliminated by performing the calculation in the return statement.

```
float celsius_to_fahrenheit(float celsius)
 {
  return(celsius * (9.0/5.0) + 32.0);
 }
```

On the Net

Important Points Regarding Return

1. *The return statement does not require that the value being returned be placed in parentheses. You may, however, want to get into the habit of placing variables and expressions in parentheses to make the code more readable.*
2. *A function can return only one value using return. Use passing by reference to return multiple values from a function.*
3. *When a return statement is encountered, the function will exit and return the value specified, even if other program lines exist below the return.*
4. *A function can have more than one return statement to help simplify an algorithm.*
5. *The calling function is not required to use or even to capture the value returned from a function it calls.*

To learn more about the ways return can be used, and to see examples of the points listed above, see http://www.ProgramCPP.com. See topic 9.2.1.

PITFALLS

When the last line of a function is reached, or when a **return()** *statement is executed, the function ends and the program returns to the calling function and begins executing statements from where it left. Do not end functions with a call back to the original function or the function will not terminate properly. Continually calling functions without returning from them will eventually cause the program to crash.*

If you call the main function at the end of a function you wrote, the main function will begin with the first statement, rather than beginning with the statement following the call to your function.

EXERCISE 9-7 USING RETURN

1. Open *CTOF.CPP*. The complete Celsius to Fahrenheit program appears.
2. Compile, link, and run the program. After you have seen it work, close the source code file.

DIVIDING THE SERIES PROGRAM INTO FUNCTIONS

Now that you have practiced creating functions and moving data to and from them, let's take another look at the program from Exercise 9-1. Earlier in this chapter, you studied a VTOC of the program divided into functions. Refer back to the first few pages of this chapter if you need to review the program or VTOC.

EXERCISE 9-8 MULTI-FUNCTION PROGRAM

1. Retrieve the source code file *SERIES2.CPP*. Analyze the source code to see that the program is divided into functions.
2. Compile and run the program to see that it has the same result as the single-function version you ran in Exercise 9-1.
3. Close the source code file.

More About Function Prototypes

A function prototype consists of the function's return type, name and argument list. In this chapter, the function prototypes specified the parameter names in the argument list. However, this is not necessary as long as the type is specified. For example, the prototype for the **celsius_to_fahrenheit** *function could be written as:*

```
float celsius_to_fahrenheit(float);
```

The prototype for the **get_values** *function could be written as:*

```
void get_values(float &, float &);
```