```
            current_month = 1;
            current_year++;
         }
      }
   return 0;
}
```

After the variables are declared, the main function calls **get_user_input** to get the necessary values from the user. Once those values are returned to the main function, **current_year** and **current_month** are initialized to the initial values entered by the user.

Next, two format options are set to ensure that the output does not appear in exponential notation, and that the decimal point always appears. Now we're ready for the loop that prints the output.

We'll use a for loop to print the table because we know the number of times we need to iterate. A loop counter variable named **month_count** is used by the for loop.

The first statement in the loop calls a function named **calculate_principal** that takes the principal and interest rate and returns a new principal amount. The principal will grow with each iteration of the loop.

Next, the current year and current month are passed to a function called **print_date**, which prints the name of the month followed by the year. The principal is then printed on the same line of the screen.

The remainder of the loop is an if structure used to increment the month and, when necessary, the year. Each time the year is incremented, the output pauses to allow the user time to read the output.

EXERCISE II-1 CREATING THE main FUNCTION

1. Enter the following lines into a blank editor screen.

```
// COMPOUND.CPP
// By Jonathan Kleid
// Calculates future value of an amount of money placed in an interest
// bearing account over time.

#include <iostream.h>
#include <iomanip.h>

// function prototypes
void get_user_input(float &principal, float &interest_rate,
                    int &first_year, int &first_month, int &total_months);
void calculate_principal(float &principal, float interest_rate);
void print_date(int year, int month);
```

2. Add the main function at the bottom of the source code file.
3. Save the source code file as *COMPOUND.CPP* and leave it open for the next exercise.

Now that we have analyzed the main function, let's look at the functions required to finish the program.

The **get_user_input** function (shown below) receives variables by reference, prompts the user for values for the variables, and returns the values to the main function. The function uses do while loops to repeat prompts until valid values are input.

```
// Function that gets the input values from the user.
void get_user_input(float &principal, float &interest_rate,
                    int &first_year, int &first_month, int &total_months)
{
  cout << "Enter the starting principal: ";
  cin  >> principal;
  cout << "Enter the current interest rate (Ex. 0.09 for 9%): ";
  cin  >> interest_rate;
  do
   {
     cout << "Enter the first year: ";
     cin  >> first_year;
     if ((first_year < 1900) || (first_year > 2050))
      {
       cout << "Invalid year.\n";
      }
   } while((first_year < 1900) || (first_year > 2050));

  do
   {
     cout << "Enter the first month (1 for Jan., 2 for Feb., etc...): ";
     cin  >> first_month;
     if ((first_month < 1) || (first_month > 12))
      {
       cout << "Invalid month.\n";
      }
   } while((first_month < 1) || (first_month > 12));

  cout << "Enter the total number of months: ";
  cin  >> total_months;
  cin.ignore(80,'\n');
}
```

EXERCISE II-2 ADDING THE USER INPUT FUNCTION

1. Add the **get_user_input** function to the source code on your screen.
2. Leave the source code file open for the next exercise.

The **print_date** function (shown below) is primarily a switch structure that prints the name of the month based on the month number that is passed to the function. The year and month are passed by value because there is no need to return the values to the main function. After the switch structure prints the month, a statement prints the year.

```
// Function that prints the month and year to the screen.
void print_date(int year, int month)
{
  switch(month)
```