Overview

In this case study, you will follow the steps of the programming process to develop a program that calculates the growth of money using compound interest. When money is placed in an interest-bearing account, the bank adds the interest earned at a regular interval, often monthly. Each month, you earn interest on a larger amount of money because you are also earning interest on the interest you earned in previous months.

Defining the Problem

We'll begin by defining the problem. As input, the program will ask for several pieces of data.

1. The amount of money placed in the savings account.
2. The interest rate the money will earn in the account.
3. The year that the money is placed in the account.
4. The month that the money is placed in the account.
5. The number of months the money is to remain in the account.

The output will be a table that shows the month and year and the amount of money to which the account has grown by that month.

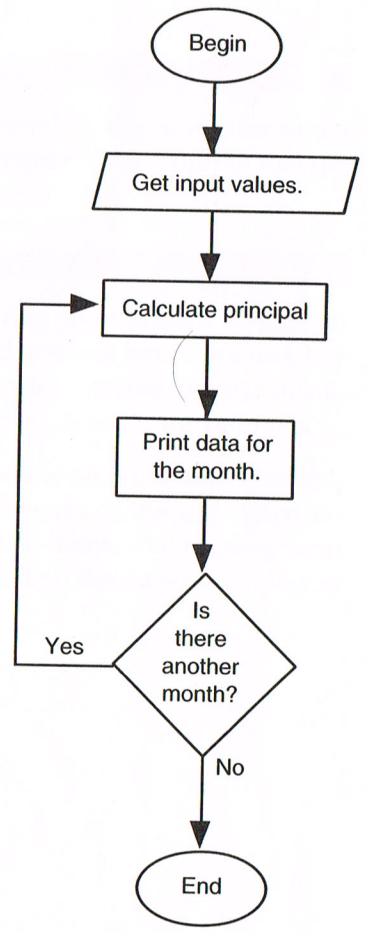

FIGURE II-1
A flowchart helps a programmer visualize the flow of logic in a program.

Developing an Algorithm

The flow of the program is fairly simple. First, it must ask the user for the needed values. Then the program will use a loop as it calculates the new principal amount for each month. The flowchart in Figure II-1 illustrates the flow of logic necessary for the program.

The next step would be to develop a Visual Table of Contents (VTOC) to decide what functions must be written. Functions would need to be written that get the input from the user, that calculate the new account balance, and that print a line of the table, as shown in the VTOC in Figure II-2.

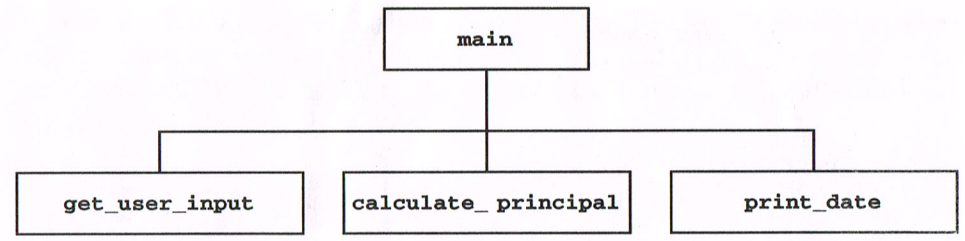

FIGURE II-2
A visual table of contents shows the functions of the program.

Coding the Program

A top-down approach to coding the program is followed. We begin by writing the main function, and then write the functions on which the main function depends. As our flowchart showed, the program must first ask the user to input values. But before we ask the user for those values, there must be variables in which to store the data. The code below shows the main function.

```
// main function
int main()
{
  float principal, interest_rate;
  int first_year, first_month,
      total_months, month_count,
      current_year, current_month;
char WaitTemp[2];

get_user_input(principal, interest_rate, first_year,
               first_month, total_months);

current_year = first_year;
current_month = first_month;

cout.setf(ios::fixed);      // prevent exponential notation
cout.setf(ios::showpoint); // always show decimal point
for (month_count = 0; month_count < total_months; month_count++)
 {
  calculate_principal(principal, interest_rate);
  print_date(current_year, current_month);
  cout << "$" << setprecision(2) << principal << '\n';
  if (current_month < 12)
   {                  // If not yet December, increment month
    current_month++;
   }
  else                // else stop output until Enter is pressed,
   {                  // set month back to January, and increment year
    cout << "\nPress Enter to Continue...\n";
    cin.get(WaitTemp,0);
    cin.ignore(80,'\n');
```