

```
// Program: Convert from Julian Day
// Author: Chuck Stewart
// Purpose: Convert a Julian day in a given year to the associated
// month and day. This demonstrates the use of functions with
// reference parameters, as well as while loops and switch statements.
// It also illustrates a different way to organize the source code in
// a program file, with the main function at the top. This
// necessitates the introduction of function prototypes.

#include <iostream>
using namespace std;

// This is a constant array of integer values. The array entry
// associated with index i is the number of days in the ith month of
// the year. Since C++ array indices start at 0 and month counting
// starts at 1, there are 13 locations in the array, and the
// first location is given a default value of 0.

const int DaysInMonth[13] = { 0, 31, 28, 31, 30, 31, 30,
                             31, 31, 30, 31, 30, 31 };

// Here are "prototypes" for the functions defined below. These
// prototypes are listed here so that the main function will know
// what their interfaces look like. The order and types of the
// parameters must match exactly the order and types of the
// parameters in the actual functions defined below. Note, however,
// that the names of the parameters do not need to match between the
// prototypes and the actual function definitions; the names do
// not even need to be provided in the prototype.

bool is_leap_year( int year );
void month_and_day( int julian_day, int year, int & month, int & day );
void output_month_name( int month );

// Now the main function is defined. It does very little other than
// input / output and call the function to convert the Julian date
// to a month and day. The choice of whether to place the main
// function at the beginning or end of the source code file is
// a matter of taste only.
//
// Any function called by the main function must have its prototype
// appear before the main function. This is in fact true anywhere in
// the code: if function A calls function B, then at least a
// prototype for B (if not the actual definition of B) must appear
// before the code for A. Eventually we will be placing some of
// these prototypes in separate "header" files.

int main() {
    cout << "Please enter two integers giving the Julian date and the year: ";
    int julian, year;
    cin >> julian >> year;

    int month, day_in_month;
    month_and_day( julian, year, month, day_in_month );
    cout << "The date is ";
    output_month_name( month );
    cout << " " << day_in_month << ", " << year << endl;
    return 0;
}
```

```
// Function returns true if the given year is a leap year and
// returns false otherwise.
bool is_leap_year( int year ) {
    return year % 4 == 0 && ( year % 100 != 0 || year % 400 == 0 );
}

// Compute the month and day corresponding to the Julian day within
// the given year.
void month_and_day( int julian_day, int year, int & month, int & day ) {
    bool month_found = false;
    month = 1;

    // Loop through the months, subtracting the days in this month
    // from the Julian day, until the month is found where the
    // remaining days is less than or equal to the total days in the
    // month.
    while ( !month_found ) {
        // Calculate the days in this month by looking it up in the
        // array. Add one if it is a leap year.
        int days_this_month = DaysInMonth[month];
        if ( month == 2 && is_leap_year(year) )
            ++ days_this_month;

        if ( julian_day <= days_this_month )
            month_found = true; // Done!
        else {
            julian_day -= days_this_month;
            ++ month;
        }
    }
    day = julian_day;
}

// Output a string giving the name of the month.
void output_month_name( int month ) {
    switch (month) {
        case 1: cout << "January"; break;
        case 2: cout << "February"; break;
        case 3: cout << "March"; break;
        case 4: cout << "April"; break;
        case 5: cout << "May"; break;
        case 6: cout << "June"; break;
        case 7: cout << "July"; break;
        case 8: cout << "August"; break;
        case 9: cout << "September"; break;
        case 10: cout << "October"; break;
        case 11: cout << "November"; break;
        case 12: cout << "December"; break;
        default: cout << "Illegal month";
    };
}
```