Consider program **p01** in file **p01.cpp** below. Program **p01** contains **class** *Date*. The object of this example is to show how to split file **p01.cpp** into files **p01.cpp**, **Date01.h**, **Date01.cpp**, and **p01make**.

```
#include <iostream>
#include <fstream>
#include <iomanip>
#include <string>
using namespace std;
class Date {
    int M,D,Y;              //Month, Day, Year
public:
    Date(int m,int d,int y)
        :M(m-1),D(d),Y(y)
    {}
    void Print(ostream& o)
    {   static string Month[]=
        {"January","February","March","April","May","June"
        "July","August","September","October","November","December"
        };
        o << Month[M];
        o << " ";
        o << D;
        o << ", ";
        o << Y;
    }
};
int main()
{   Date SashaBDay(8,2,1995);
    cout << endl;
    SashaBDay.Print(cout);
    cout << endl;
    return 0;
}
```

Figure 1. File **p01.cpp**.

Program p01 prints

```
August 2, 1995
```

Table 1 lists file names together with a description of their contents

File	Description
p01.cpp	File **p01.cpp** exercises **class** *Date*. Variable *SashaBDay* is initialized to 8-2-1995 and printed.
Date01.h	File **Date01.h** contains the interface for **class** *Date*. Member data and member function prototypes appear in the interface.
Date01.cpp	File **Date01.cpp** contains the implementations of member functions of class Date. Member function names are qualified with the class name (*Date*) and the global resolution operator (**::**).
p01make	File **p01make** contains instructions for creating executable file **p01**. Instructions in file **p01make** are executed by the UNIX utility ***make***.

Table 1. Files of project **p01** and their descriptions.

File **p01.cpp** has been revised to its new contents below. Note the absence of **class** *Date*. Note the inclusion of file **Date01.h**. **class** *Date* defines the new type *Date*.

```
//-------------------------------------------------------
//C++ include files
//-------------------------------------------------------
#include <iostream>
#include <fstream>
#include <iomanip>
#include <string>
using namespace std;
//-------------------------------------------------------
//Application include files
//-------------------------------------------------------
#include "Date01.h"
//-------------------------------------------------------
//Function main exercises class Date
//-------------------------------------------------------
int main()
{    Date SashaBDay(8,2,1995);
     cout << endl;
     SashaBDay.Print(cout);
     cout << endl;
     return 0;
}
```

Figure 2. File **p01.cpp** revised.

File **Date01.h** defines the interface for **class** *Date*. The interface is *used* in file **p01.cpp** where a variable of type *Date* is declared. The interface is also included in file **Date01.cpp**. The interface is *validated* when file **Date01.cpp** is compiled. Any differences between member function prototypes in file **Date01.h** and member function implementations in file **Date01.cpp** are noted in compilation errors.

```
#ifndef Date01_h
#define Date01_h 1
//-------------------------------------------------------
//C++ include files
//-------------------------------------------------------
#include <iostream>
#include <fstream>
#include <iomanip>
using namespace std;
//-------------------------------------------------------
//Application include files
//-------------------------------------------------------
class Date {
     int M,D,Y;              //Month, Day, Year
public:
     Date(int m,int d,int y);
     void Print(ostream& o);
};
#endif
```

Figure 3. File **Date01.h**.

Note the two macro directives that are given at the beginning of file **Date01.h**. Note the directive on the last line of file **Date01.h**. The directives are
#ifndef *Date01_h*
#define *Date01_h* **1**

…

#endif
The purpose of the two directives is to prevent the file from being included more than once. The C++ compiler prohibits multiple declarations of the same identifier. If file **Date01.h** is included more than once in a file compiled by the C++ compiler will cause compilation errors. In particular identifier *Date* will be declared two or more times.

The directive **#ifndef** *Date01_h* says "Is identifier *Date01_h* undefined?" If the answer is yes, then the second directive is executed. The second directive, **#define** *Date01_h* **1**, says define and assign a **1** to the identifier *Date01_h*. If the answer is no, then skip all remaining source until the directive **#endif**. Do not include the source between the directives **#ifndef** *Date01_h* and **#endif**.

File **Date01.cpp** contains the implementations of member functions of **class** *Date*.

```
//-------------------------------------------------------------
//C++ include files
//-------------------------------------------------------------
#include <iostream>
#include <fstream>
#include <iomanip>
#include <string>
using namespace std;
//-------------------------------------------------------------
//Application include files
//-------------------------------------------------------------
#include "Date01.h"
//-------------------------------------------------------------
Date::Date(int m,int d,int y):M(m-1),D(d),Y(y){}
void Date::Print(ostream& o)
{    static string Month[]=
          {"January","February","March","April","May","June"
          ,"July","August","September","October","November","December"
          };
     o << Month[M];
     o << " ";
     o << D;
     o << ", ";
     o << Y;
}
```

Figure 4. File **Date01.cpp**.

File **p01make** contains instructions for creating executable file **p01**. Instructions in file **p01make** are accepted and executed by the UNIX utility ***make***. Refer to lecture note 27 for a discussion of the C++ compiler, project files, and the ***make*** utility. To invoke file p01make enter the command given on the command line as shown.

$ make –f p01make

```
#-------------------------------------------------------------
# File p01make contains instructions for the make utility that
# create executable file p01.
# Program p01 exercises class date.
#-------------------------------------------------------------
# Author:  Thomas R. Turner
# E-Mail:  trturner@ucok.edu
# Date:    April, 2005
#-------------------------------------------------------------
# Copyright April, 2005 by Thomas R. Turner.
# Do not reproduce without permission from Thomas R. Turner.
#-------------------------------------------------------------
# Object files
#-------------------------------------------------------------
obj    =      p01.o Date01.o
#-------------------------------------------------------------
# Link object files into program p01.
#-------------------------------------------------------------
p01:           ${obj}
               g++ -o p01 ${obj} -lm
#-------------------------------------------------------------
# Compile p01.cpp that exercises class Date
#-------------------------------------------------------------
p01.o:         p01.cpp Date01.h
               g++ -c -g p01.cpp
#-------------------------------------------------------------
# Compile Date01.cpp that implements member function of class Date
#-------------------------------------------------------------
Date01.o:    Date01.cpp Date01.h
               g++ -c -g Date01.cpp
```

Figure 5. File **p01make.**