MCIS-4135
Homework Assignment 7

Purpose:

This assignment will give you the opportunity to continue abstract data type programming in C++. The assignment is written a lot like a recipe, with the hope that you will admire the result, and review the steps that you took to get there.

This problem is worked by modifying the solution to Homework Assignment 6. Please start with those three files, and make modifications as described below.

What to hand in:

Hand in listings of the program which must be captured in 3 or more files: a header file, a test program, and at least one implementation file. Also, hand in the output of a sample run (or runs) of the program. All of these must be formatted in a reasonable style. Choose sufficient test cases to ensure your functions work properly.

Problems:

More Dating Skills:

1. Create a "project" in Visual C++ (or the equivalent in whatever environment you are using). We will create 3 files in this project, one header file called date.h, and two C++ source files, called date.cpp and hw7.cpp. Begin by copying the corresponding files from homework 6 into these names (copy hw6.cpp to hw7.cpp).
2. Modify date.h according to the following rules, for each function in the header:
   a) Move the function into the structure
   b) Remove the "Date" from the function's name.
   c) Remove the function's first Date parameter (if any)
3. Modify date.cpp according to the following rules for each function:
   a) Put the scope operator between after the letters Date in each function.
   b) Remove the function's first Date parameter (if any)
   c) Where the date parameter is referred to inside the function (e.g., d.day), remove the parameter name and "dot".
4. Modify hw7.cpp according to the following rules for each call
   a) Change every call like DateXXXX(Date a,b,c) to a.XXXX(b,c);
   b) Change every call like Date a = DateXXXX(b,c) to a.XXXX(b,c);
5. Compile and run your final program, capturing the output for submission.

Notes:

For step 2, note that this will change

```
struct Date {
  ...
};
void DatePrint(Date d);
```

to

```
struct Date {
  ...
  void Print();
};
```

For step 3, this will change

```
void DatePrint(Date d) {
  cout << d.month << '/' << d.day << '/' << d.year;
}
```

to

```
void Date::Print() {
  cout << month << '/' << day << '/' << year;
}
```

For step 4, this will change

```
Date d;
d = DateSet(3,2,2004);
```

to

```
Date d;
d.Set(3,2,2004);
```

and

```
DatePrint(d);
```

to

```
d.Print();
```

After making these changes, you should be able to recompile the program, run it, and get the same results as the previous exercise.

The really important thing to notice is that your program is now organized around a bunch of instances (Date variables) that are having "messages" sent to them (methods invoked on them).

For contrast note that last week we had functions "in charge", and we had to make sure they had the right parameters. This week, we have the objects in charge.

Evaluation Criteria:

Code will be evaluated both for proper style and for correct content. Your code should be in 3 (or more) files: date.h, date.cpp, and hw7.cpp (date.cpp could be subdivided into many other files if you choose).

30	Interface file: proper definitions and content, date type, compliant set of operations
35	Implementation file: proper use of C++ for implementation
35	Test program: full test of interfaces, including boundary conditions. Inclusion of results. ***Note: If your program doesn't compile, hand in the error listing as the results!***