**CS 271**
**Programming Assignment # 10**

You are expected to read Chapter 18 before beginning this assignment. Also, view the video by Derek Banas on C++ Classes.

C++ style input and output is required throughout this lab.

The following C language features are prohibited:

- printf
- scanf
- C library names
- \t in output statements
- \n in output statements

Submit the following 4 files. Don't zip or tar them.

makefile
Date.h
Date.cpp
PA10.cpp

Programs must compile.
Programs that have syntax errors receive a grade of zero.

- Documentation and Style - This is a large part of the grade on this assignment. See the "Documentation and Style Guidelines" document in the Lecture Notes Module. The TA will be checking for:
  - Header and inline comments
  - Indentation
  - Spacing
  - Use of blank lines
- Follow course conventions for identifiers (variable names, function names, class names).
- You may use "this" in your code only if it is absolutely necessary.
- Do not use "std::" in your code.

## Makefile

A makefile is required for this assignment.
You must have separate targets for

- all
- PA10
- PA10.o
- Date.o

Resources

You can re-use much of the code from the Date class that you wrote for PA 8.

You can get a lot of good ideas and code from the Date class in Chapter 18 (sec 18.8)

Exception Handling

In this assignment, you will modify functions that do data validation so that instead of ignoring bad data, you will throw an "invalid_argument" exception.

You will need

```
#include <stdexcept>
```

Implementing the PA 10 Date Class

You can begin with the Date class you wrote in PA 8. You will need to modify some of the functions, remove some, and add new functions to match the specifications of this assignment.

Make sure that you remove any functions from PA 8 that are not listed in the specification for this assignment.

| **Date** |
| --- |
| - month : unsigned int<br>- day : unsigned int<br>- year : unsigned int |
| <<constructor>> Date( )<br><<constructor>> Date( m : unsigned int, d : unsigned int, y : unsigned int )<br>+ setDate( m : unsigned int, d : unsigned int, y : unsigned int )<br>+ getMonth( ) : int<br>+ getDay( ) : unsigned int<br>+ getYear( ) : unsigned int<br>+ operator++( n : int ) : Date<br>+ operator++( ) : Date &<br>+ operator+=( nd : unsigned int ) : Date &<br>- helpIncrement( ) : void<br>- endOfMonth( ) : bool<br>- leapYear( ) : bool<br><<friend>> ostream& operator<<( output : ostream &, d : const Date & )<br><<friend>> istream& operator>>( input : istream&, d : Date & ) |

Additional specifications for the member functions:

1) default constructor Date( ) - initializes Date object to 1/1/1980
   Note: This is different from the constructor in the textbook. This constructor does not have default arguments.

2) another constructor Date( int m, int d, int y ) - initializes Date object using the parameters.

3) There are no mutators. There is one function called setDate which accepts 3 parameters corresponding to month, day and year. The setDate function does not have default arguments.
   Note: setDate must check month and day to make sure they are compatible.
   Note: setDate must allow 29 days in February during leap years.
   Note: setDate must throw an exception if the month, day, and year do not represent a valid date.

4) 3 accessors: an accessor for month, an accessor for day, and an accessor for year.
   Note: Apply "const" to all accessors.

5) Overloaded operator functions for prefix increment, postfix increment, and the shortcut assignment +=.
   Note: prefix increment will add one day to the calling object and change month and year if necessary.
   Note: postfix increment will add one day to the calling object and change month and year if necessary.
   Note: += will add days to the calling object and change month and year if necessary.

6) private functions: helpIncrement, endOfMonth, leapYear.
   Note: See the UML class diagram. The specifications for endOfMonth and leapYear are different than what is shown in the textbook.

Additional specifications for non-member functions:

The overloaded output operator should have "const" on its second parameter. The operator must be written in a way that allows cascading of <<.

Output format example: October 15, 2018

The overloaded input operator will input 3 integers for month, day, and year (in that order) and call the setDate function. The operator must be written in a way that allows cascading of >>.

PA10.cpp

Test all of the Date class functions thoroughly. There are 14 functions. One call to each function is NOT thorough testing.

Submit the following 4 files. Don't zip or tar them.

makefile
Date.h
Date.cpp
PA10.cpp