CSCI 501 **Computer Assignment 6** **Fall 2018**

(10 points)

The Date Class

Write a C++ program to implement the Date class, which has the following *private* data members: string month, int day, int year, where month represents the month component of a Date object with valid values: January, February, March, April, May, June, July, August, September, October, November, and December. Clearly, valid values of the day component depend on its month component. For example, for the month of January, a valid value for a day is between 1 and 31, but for the month of February, the valid value is between 1 and 28 (29 if it's a leap year). For the year component, there is no maximum value, but it has the minimum value 1 (not 0).

Put the definition of your Date class in the header file Date.h, and the implementations of its member functions in the source file Date.cc. At the top of your header file, insert the following statement: #include "Date.h".

The Date class should include the following member functions:

- Date (const string& m = MONTH, const int& d = DAY, const int& y = YEAR): This is a *public* constructor that can be used to create a Date object for the month m, day d, and year y, where MONTH = "January", DAY = 1, and YEAR = 2000 are default values.

- void setMonth (const string& m), void setDay (const int& d), void setYear (const int& y): These *public* member functions can be used to set the month, day, and year components of a Date object to the values m, d, and y, respectively, by a client program.

- string getMonth ( ) const, int getDay ( ) const, int getYear ( ) const: These *public* member functions can be used to return the month, day, and year components of a Date object, to a client program.

- void Month ( ): This *public* member function can be used to convert the month component of a Date object by changing its first letter to uppercase and the rest of its letters to lowercase, so the month of a date can be entered as case insensitive.

- bool isValidDate ( ) const: This *public* member function checks a Date object, and if it's a valid date, it returns true; otherwise, it returns false. It calls the *private* member function: bool isValidMonth ( ) const to check if the month component is valid, and it calls the *private* member function: int daysInMonth (const string& m) const to return the number of days in the month m to check if the day component is valid. If the month m is February, the function: daysInMonth ( ) calls the *private* member function: bool isLeapYear ( ) const to get the correct number of days in the month m.

- string toString ( ) const: This *public* member function can be used to convert the month component of a Date object in the form dd-mmm-yyyy, where dd is a one- or two-digit date, mmm the three-letter abbreviation for the month, and yyyy is the four-digit year. For example, if the date is July 20, 1969, then this function should return the string 20-Jul-1969. To help for the conversion, it calls the non-member function: string intToString (const int& n), which takes the integer value n and returns its corresponding value as a string.

- friend istream& operator >> (istream& is, Date& d): This non-member function can be implemented as a friend to the Date class, which overloads the *extraction operator* (>>). It reads the individual date components from the input stream is to Date object d, where a date can be entered either in the form month day year or month day, year. Since the character after the day component is optional, one way to deal with this issue is to use the unget ( ) function for the input stream is. After extracting the day component, you can read one more character, and if that character happens to be a digit, then you can put that character back into the input stream is.

- friend ostream& operator<< (ostream& os, const Date& d): This non-member function can also be implemented as a friend to the Date class, which overloads the *insertion operator* (<<). It gets the individual components from the Date object d and to the output stream os, where d is inserted in os in the form month day, year.

Put the definitions of the components of the default date in your header file Date.h, and you can also include the following vector object for the months of a year: const vector < string > months { “January”, “February”, “March”, “April”, “May”, “June”, “July”, “August”, “September”, “October”, “November”, “December” }, so you can compare a month with the elements of the vector months to determine if the month has a valid value.

A driver program is supplied to test your Date class. The source file of the driver program is prog6.cc. To compile the source files prog6.cc and Date.cc and link their object files with the system library routines, execute: Make N=6. To test your program with the data file prog6.d, execute: Make execute N=6. The files prog6.cc and prog6.d are in the directory: ~cs501/progs/18f/p6. The correct output file prog6.out is also in the same directory.

The main ( ) routine tests the member functions of the Date class for several dates. Some of these dates are generated by the class constructors and some of them are entered from the stdin. For each Date object d, it calls the non-member function: bool printDate (const Date& d). This routine check if d is a valid date, and if it's not, it prints an error message on stderr and it returns false; otherwise, it prints d on stdout and returns true. For a Date d entered from the stdin, if the return value of printDate ( ) is true, it also prints d on stdout in the form dd-mmm-yyyy. The implementation of printDate ( ) is included in the driver program prog6.cc.

When your program is ready, mail its source and header files of your Date class to your TA by executing: mail_prog.501 Date.cc Date.h.