Overview

For this assignment, you will write a C++ program that works with dates. The C and Java versions of this code were discussed in class and are available on the webpage.

Description

You must submit three sources files: date.h, date.c, and date_main.c. The header file, date.h, will contain the Date class declaration, which includes the instance variables and function prototypes. date.c should provide function definitions, both public and private, for the Date class. Finally, date_main.c is the driver for the program. You may want to name your executable dt to avoid conflict with the Unix date command.

As in the C and Java versions, your Date class must provide the following functions:

```cpp
bool equals (Date date);
bool lessThan (Date date);
void format (char *str);
void nextDay (void);
bool leapYear (void);
int monthLen (int month, bool leap);
char *monthStr (void);
```

Note that these C++ prototypes are slightly different than in either of the provided programs; however, most of the code in the definitions of these functions will remain the same. Designate the above functions with the proper visibility: public or private.

Additionally, you must provide two constructors:

```cpp
Date (void);
Date (int day, int month, int year);
```

that work similarly to init_date1() and init_date() in date.c, and the two Date constructors in Date.java, respectively.

To give you some experience with working with operator overloading, you must also provide the following operators in the Date class:

```cpp
bool operator== (Date date);
bool operator< (Date date);
bool operator> (Date date);
Date operator++ (int dummy);
```

where the comparators correspond directly to their meanings with integers, and ++ means "add a day." Keep in mind that you already have code that will implement these functions, so the code for these operators may simply contain a call to their corresponding functions.
In `date_main.C`, define three variables as follows:

```c
Date d1 = Date ();
Date d2;
Date *d3;
```

Add an initialization for `d2` in the declaration line above that calls the other `Date` constructor with the values for December 31, 1999 (as in `date.c` and `date.java`). Initialize `d3` later in the code with `new` and the `Date` constructor with values for January 1, 2000.

Follow the exact same steps for printing out the variables, incrementing `d2`, and comparing `d2` and `d3` as in `date.c` and `date.java`. Use `cout` for printing to the screen and add some tags and blank lines to make the output more readable.

Finally add lines that will perform the following:

- increment `d2` using `++`
- get the format string for `d2`
- print the format string
- print "d2 < d1? "
- compare `d2` and `d1` using the `<` operator
- print "true" or "false" as before
- print "d2 == d3? "
- compare `d2` and `d3` using the `==` operator
- print "true" or "false" as before
- print "d2 > d3? "
- compare `d2` and `d3` using the `>` operator
- print "true" or "false" as before

Your final output should look exactly like this:

```
d1:  January 1,  1
d2:  December 31, 1999
d3:  January 1, 2000

 d2 < d1? false
 d2 < d3? true

 d2:  January 1, 2000
 d2 == d3? true

 d2:  January 2, 2000
 d2 < d1? false
 d2 == d3? false
 d2 > d3? true
```

Please provide a correct makefile with your submission, as well as a `README.txt` file describing the option you chose and any special features of your program.