

Computer Science 215
Tools and Techniques for Software Development
Fall 2001
Project 6 – C++ Date Program

Due: Friday, 12/7/2001 Midnight

For this assignment, you will write a program 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.

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:

```
bool equals (Date date);
bool lessThan (Date date);
void format (char *str);
void nextDay (void);
bool leapYear (void);
int monthLength (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:

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

that work similarly to `init_date_1 ()` 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:

```
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 function.

In `date_main.c`, define three variables as follows:

```
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` 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 < 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

print "d2 > d3? "
compare d2 and d3 using the > operator
print "true" or "false" as before
```

Your final output should look like this:

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

d2 < d3? true

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

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