Project #11 (50 points): Classes

Submit Your Solution Using Canvas by 10 PM on Tuesday, 11/27/18

(late submissions will be accepted on 11/27/18 from 10PM to Midnight)

Obtaining Project 11 Input Order and Output:

To view the order of the information input and the format of the output for the project, run the solution by typing the following at a terminal prompt.

Your program's output should look like that produced by the provided sample solution. You can run the sample solution **Project_11_solution** by typing the following at a command prompt in a terminal window**:**

```
/home/work/cpe211/Executables/Project_11/Project_11_solution
```

The comparison script can be run by using the following command

```
/home/work/cpe211data/Project_11/CompareSolution.bash   Project_11.cpp
```

Project 11 Restrictions

Any C++ technique covered in Chapters 1 through 12 is allowed except for global variables ***You are not allowed to use any global variables.*** If necessary, global constants may be used.

Project 11 Description

For this project a Date class will be constructed (contained in the file P11.h) and all function definitions will be written in the file Project_11.cpp. A makefile is provided to compile this project. The source code containing the main function is provided as well. Name of this file is Project_11_main.cpp. On Canvas, download the files P11.h, Project_11_main.cpp and Makefile. Save these files in your Project 11 directory. Do not modify these files. Another file is provided as well. It is Project_11.cpp. Download this and all the code you write will be in this file. This file contains all the function definitions for the methods described in the class in the header file provided.

This project is similar to the Time class described in the classes power point slides –

Note: do not use any variable/parameter in your functions or program that are named month, day or year – these are used as private members of the class and you want make sure that you know which variable or parameter you are talking about.

For this project the following functions are written:

1) For class constructors:
    a. A default constructor called Date() – default date is 1/1/1900
    b. A parameterized constructor Date(int, int, int) – initialize a Date object to the values provided when it is declared(i.e. Date myDate(1,1,1900);
2) For the class Transformers
    a. A parameterized function SetDate(int,int,int) – this sets the date in a Date object (variable of the class Type Date) to the date specified
3) For the class observers –two functions for writing out the date – one format is all numbers the other requires the use of the month name and one function for comparing two dates.
    a. WriteNumberFormat() – writes out the date in format MM/DD/YYYY

b. WriteNameFormat() – writes out the date in format MonthName day, year
c. SameDate(Date) – returns true or false if the class object invoking the function is the same date as the class object passed in as a parameter

After downloading all 4 files (P11.h, Makefile, Project_11_main.cpp and Project_11.cpp), start your work by making empty function stubs in Project_11.cpp. The function headings are determined from the public functions listed in P11.h. Once you have all the function stubs written, you should be able to run the Makefile (type make at the command prompt) to generate the program executable Project_11. You can run the program, but not much happens.

Now add code to each of the functions described above and after completing each function, run the makefile to make sure the program still compiles. If it does not compile fix the errors until it does compile. I recommend writing the two constructors and WriteNumberFormat functions first. Then the rest of the functions can be written in any order.

Project 11 Function Contents

1) The constructor functions set the private members of month, day and year to the default values or the values specified. The order for the Parameterized constructor is Month, Day and Year

2) The SetDate function takes in a date – order is Month, Day and Year – and stores those values in the appropriate private members

3) The WriteNumberFormat function takes the values of the private members month, day and year and outputs them in the form **mm/dd/yyyy**

4) The WriteNameFormat function uses the values of the private members month, day and year and outputs them in **MonthName day, year** format. To get the month name, use a switch statement using month as the switch expression and the enumerators provided in the header file as the case labels.

5) The SameDate function compares the private members month, day and year of the object that invokes the function with the month, day and year members of the object supplied as the argument in the function call. Boolean true is returned if the dates are identical and false is returned otherwise.

When completed, the program output shall match that of the sample solution. There are no input files for this program. All input operations are run from Project_11_main.cpp. **Output that does not match that of the sample solution will lose points.**

All of these functions are relatively short (less than 10 lines) except for the WriteNameFormat function that has several lines due to the 12 case label switch statement – unless you want to create an array and store the month names in an array and then access the correct array element for the name – just remember the month numbers are 1 through 12 and array indexing starts at 0.

Submit your Project_11.cpp file only on Canvas.

Project 11 Helpful Information

- To output days or months less than 10 with a leading 0, use **`cout << setfill('0');`** and then output the month or day in a field width of 2 right justified. After printing out the date, use **`cout << setfill(' ');`** to set the fill character back to a space
- Write one function and then run the Makefile by typing make on the command line. Fix all errors in the function before moving on to the next function.
- **There is very little in the way of output statements in the functions. Matter of fact, only the two write functions have output statements in them. All other output is handled in main which is provided and should not be changed.**