PROGRAMMING ASSIGNMENT 2 CPSC 1020

Practice C++

Calendar Events

DUE: November 20, 2019 midnight

Learning Objectives:
This assignment is designed to provide practice with the following:

- Working with Classes
- Working with Composition “has-a” relationship between classes
- Working with vectors
- Formatting output
- Working with multiple files
- Command-line arguments
- Reading data from files
- Sorting data

Overview:
You are going to read in various appointments from an input file. You will store the appointment information in a vector. You will sort and print out the appointment information. You will write three classes for this assignment as well as several helper functions that work with the three classes.

Classes:

You will have a CalendarEvent, Date, and Time class. In this document, I will give you the class declaration for each of the three classes. You will need to provide both .cpp and .hpp files for each class. You will also provide a functions.cpp and functions.hpp.

This assignment will give you practice using composition. Composition is realized when one class “has-an” instance of another class as one of the data members. The CalendarEvent class will have an instance of Date and an instance of Time as one of the data members. This document will provide the .hpp files. (CalendarEven.hpp, Date.hpp, Time.hpp, and functions.hpp). You are required to implement the class functions as well as the helper functions in the .cpp files. (CalendarEvent.cpp, Data.cpp, Time.cpp, and functions.cpp). You will also provide a driver.cpp. The driver.cpp file should have minimal amount of code in it.

PROGRAMMING ASSIGNMENT 2 CPSC 1020

Practice C++

Calendar Events

DUE: November 20, 2019 midnight

CalendarEvent:

The CalendarEvent class has three private data members: a string for the name of the calendar event, an instance of Date for the date of the event, and an instance for the time of the event.

```
class CalendarEvent
{
   private:
      string eventName;
      Time calTime;
      Date calDate;

   public:

      CalendarEvent(int month, int day, int year, int hour, int minute,
                    string eventName);
      void printCalendar(fstream& out);
      bool isEventDateValid();
      bool isEventTimeValid();
      string getEvent()const;
};
```

Notice there is no default constructor for the CalendarEvent class. You will need to change the provided CalendarEvent constructor so that it will provide the default values in the event an instance of CalendarEvent is created using a default constructor.

void printCalendar(fstream& out) - This function will be used to print the calendar event. It will need to print the string description of the event (eventName), it will then call the Date's print function using the instance of the Date class (calDate), it will also use the instance the Time class (calTime) to call the Time's print function. Notice the parameter for this function is "fstream rather than ifstream or ofstream". Fstream allows you to specify if the file pointer being passed to a function is for input or output. This will become clearer when we go over the functions in the function.hpp file.

bool isEventDateValid() – This function is going to use the instance Date to call the isDateValid function. IsDateValid() will return true or false which will then be returned by isEvenDateValid().

bool isEventTimeValid() - This function is going to use the instance of Time to call the isTimeValid function. IsTimeValid() will return true or false which will then be returned by isEventTimeValid().

string getEvent() – This needs no explanation.

Date:

PROGRAMMING ASSIGNMENT 2 CPSC 1020

Practice C++

Calendar Events

DUE: November 20, 2019 midnight

The date class has three private data members: int month, int day, and int year, each of which are self-explanatory. It also has a public static data member that is an array of strings that represent the string value of each month.

```
class Date
{
   private:
      int month;
      int day;
      int year;


   public:
      Date(int month, int day, int year);
      void setMonth(int);
      void setDay(int);
      void setYear(int);
      void setDate(int, int, int);
      string getStrMonth();
      int getMonth()const;
      int getDay()const;
      int getYear()const;
      void printDate(fstream&);
      bool isDateValid();
      bool isLeapYear();
      static const string ARRAY[13];

};
```

Again, the Date class only has one constructor. You will need to change the provided Date constructor such that it will provide the default values in the event an instance of Date is created using a default constructor.

The regular setters and getters need no explanation, so I will not spend time on them. Other than to say you may not need this in this project but for practice **I AM REQUIRING YOU TO WRITE THEM**. The function getStrMonth will be used to return the string version of the month. The input document will provide the string in integer form. When printing the variable, you are required to print it in the string form.

void printDate(fstream& out) – This function is used to print the date portion of the calendar event. The format of the date should be in the form of: January 1, 2019

PROGRAMMING ASSIGNMENT 2 CPSC 1020

Practice C++

Calendar Events

DUE: November 20, 2019 midnight

bool isDateValid() – This function will be used to ensure the date is a valid date. Such as, making sure the months are with in 1 – 12, the years are between 1900 – 2020 and the day is between 1 and 31. If the month is February the day can only be 29 if it is leap year. Therefore, you will need to check this. You should also make sure the day given did not exceed the number of days for that month. If the function fells any of the validity checks you are to return false.

bool isLeapYear() – This function will determine if the year read in is a leap year. Look up the rules for determining leap year.

Time:

The time class has two data members, each of which are integers. They represent the hours and minutes of time.

```
class Time
{
   private:
      int hour;
      int minute;

   public:

      Time(int hours,int mins);
      void setHour(int);
      void setMinute(int);
      void setTime(int, int);
      int getHour()const;
      int getMinute()const;
      void printTime(fstream&)const;
      bool isTimeValid();

};
```

Just as with the previous classes discussed, you must modify the Time constructor so that default values are provided through this constructor as well as the regular constructor. The setters and getters are normal, again, you may not use them however, you are required to implement them for practice.

void printTime(fstream& out) prints the information pertaining to time. When printing the time you must convert hours from military time to regular time. The format of the printing hours:minutes 10:00 pm or am. If it is noon you are to print 12:00 noon.

PROGRAMMING ASSIGNMENT 2 CPSC 1020

Practice C++

Calendar Events

DUE: November 20, 2019 midnight

bool isTimeValid() checks that the hours and minutes are between 0 and 24 and minutes are between 0 and 59.

functions:

This file provides three functions, readData, checkArguments, and isOpen.

```
void readData(fstream&, vector<CalendarEvent>&, vector<CalendarEvent>&);
void checkArguments(int argc);
void isOpen(fstream&, char*);
```

void readData(fstream& f, vector<CalendarEvent>& good, vector<CalendarEvent>& bad) reads the information from the data file. You should read the information one set of data at a time. Once you have read a complete set of data (eventName, date, and time) you should create a temporary instance of CalendarEvent. You should then test the data by calling isEventDateValid() and isEventTimeValid() to check the validity of the date and time information. If both of these functions return true then store the calendarEvent on to the good vector otherwise store the calendarEvent on the bad vector. Depending on how you write your readData you may want to do a little research on the various ways to call ignore. We discussed this function earlier in the semester and I mentioned they there are various ways to call the ignore function. Knowing these different ways may be helpful to you.

void checkArguments(int argc) – this function is used to check the number of arguments passed on the command line. You will have 4 arugments: ./executable inputFile goodOutputFile badOutputFile
As stated above if the data is not valid you are to store that calendar event on the bad vector otherwise store the data on the good vector. When printing the information, you will print the good vector to the goodOutputFile. If the information to be printed is in the bad vector that information will be printed to the badOutputFile. If the correct number of files were not passed to the command line argument you should print a message that says “Not enough command line arguments” and exit the program.

void isOpen(fstream&, char*) – this function is used to check if the file opened successfully. This function has two arguments the first is fstream. We learned that input files are usually ifstream, hence “i” for input and ofstream “o” for output. Fstream can be used for either, however when opening the file you must specify if the file is being opened for input or output. The following are links to help you with this.
http://www.cplusplus.com/doc/tutorial/files/
https://www.tutorialspoint.com/cplusplus/cpp_files_streams.htm

If the file did not open correctly you should print a message that tells the user which file did not open, then exit the program. Ex. goodOutput.txt did not open!

Requirements:

1. Implement the classes listed above. (CalendarEvent.hpp, Date.hpp, Time.hpp) You are also to implement the functions in the functions.hpp.

PROGRAMMING ASSIGNMENT 2 CPSC 1020

Practice C++

Calendar Events

DUE: November 20, 2019 midnight

2. Sort the good vector and bad vectors by date.

3. Print the calendar information for the good vector. The format for the output at a minimal should be as follows:

```
**********
Pre-registration
January 4, 2019  2:09 pm
**********

**********
Faculty Meeting
January 4, 2019  12:00 noon
**********

**********
Doctor Appointment Foot
January 8, 2019  3:45 pm
**********

**********
Haircut
January 9, 2019  9:02 am
**********

**********
TA Meeting
February 4, 2019  11:30 am
**********
```

   Feel free to be creative with the final output. I enjoy seeing how creative some students can be with their output. I may even give a few EC points if you are creative with your output.
4. Print the calendar information for the bad vector in the same format as above.
5. You must use an **iterator** when printing the events in the calendar for the good and bad vectors.
6. Create a makefile that has a working make and make run. When I type make run your make file should first compile your files then run the program.
7. All of the above are the minimal you are required to do. Feel free to make minimal changes to the functions. However, I will deduct points if you make do not fulfill the minimal requirements for this assignment. As an example, if you need to change the prototype of a function feel free to. But don't do something just do shorten or make the assignment easier, by forgoing some of the listed requirements. An example would be putting all data in public space, etc.

PROGRAMMING ASSIGNMENT 2 CPSC 1020

Practice C++

Calendar Events

DUE: November 20, 2019 midnight

Extra Credit: 5 points

After sorting the calendar events by date, sort the events by time, such that your calendar entries are sorted by both time and date.

Formatting, Compiling, and Hand-in Requirements:
Tar your file **PA2.tar.gz** and submit the file through **hand-in to the folder PA2.** If you are doing the EC you must submit a tarred file called **PA2EC.tar.gz** through **hand-in to the folder PA2EC.**
PLEASE SUBMIT TO ONLY ONE HAND-IN FOLDER.

You should submit the following files.

- driver.cpp
- functions.cpp
- functions.hpp
- CalendarEvent.hpp
- CalendarEvent.cpp
- Date.hpp
- Date.cpp
- Time.hpp
- Time.cpp
- makefile
- Readme that has the following information:
  - problems you encountered with this assignment
  - how you solved the problems
  - your thoughts about the assignment.
- With the exception of the makefile, you should have a header in each of your files that contains the following information. If you neglect to place a header in a file, you will receive a 5-point deduction.

```
/**************************/
 *Your name                *
 *CPSC1020 Fall19          *
 *Section: <Your section>. *
 *UserName:                *
 *Instructor:  Dr. Yvon Feaster  *
/**************************/
```

- Your program should compile with no warnings and no errors on the School of Computing servers. There will be a substantial point deduction if your program has compile errors. If you made a substantial effort to complete the program, but you have compile errors, you will receive no more than 30 on the assignment. If the program compiles but has warnings, there will be a

PROGRAMMING ASSIGNMENT 2 CPSC 1020

Practice C++

Calendar Events

DUE: November 20, 2019 midnight

minimal of 10-point reduction. (The more warnings you have the higher the deductions.)

- ❖ You should use meaningful variable names throughout the program.
- ❖ Your code should be well documented. (comments – see example below)
- ❖ There should be no line of code longer than 80 characters.
- ❖ You should use proper and consistent indention.

Failure to do any of the above items will result in a deduction of points for each offense.

Here are some guide lines for documenting the code in this assignment.

Before each function**, in the .hpp files**, you should have a detailed description of what the overall function does. To borrow from another student's code, here is an example of overall function description.

```
/* Parameters: img - image_t pointer array holding the image data for
 *                   each of the input files
 * Return:       output - image_t struct containing output image data
 * This function averages every pixels rbg values from each of the
 * input images and puts those averages into a single output image
 */
```

You are required to have this type of comment block before each function.
Also, if you include comments in the body of the function (and you should) they should be placed above the line of code not beside the code.
Example:

Good

```
//This is a comment
if(something)
{
      do something;
}
```

Bad

```
if(something) //This is a comment
{
      do something;
}
```