ENSC 251 Lab Assignment 1

Lab Assignment Overview:

In the ENSC 251 course, you will work on four lab assignments using the skills learned throughout this course, i.e., Object-Oriented Programming (OOP) with C++. Each lab assignment weighs 10 marks. All lab assignments will be carried out and evaluated in pairs (i.e., two students per group) - I hope by now all of you have already found a partner. Some general grading logistics have been posted on our course website: https://canvas.sfu.ca/courses/70216/pages/lab-logistics.

Please make sure your code can compile and run correctly on the lab computer. If your code cannot compile, then your group can get at most 5 marks. If your code can compile, but cannot run, then your group can get at most 6 marks.

You can divide the work roughly based on the listed tasks; the overall marks for major tasks are also listed for your guidance, but not the detailed marks for every single item. Please include a simple document in each lab (include a txt or pdf file in the .zip file for submission) indicating which parts each member finished. There is some negative marking if you don't complete the listed tasks; details are not listed. Basically, the detailed grading scheme (except the overall marks for major tasks) for each lab will NOT be released before your lab grading is done. Think about you are in a real interview, nobody will tell you what the detailed grading schemes are.

Lab Assignment 1:

In the first two lab assignments, you will build a simple calendar system, where one can book, query, and delete appointments. To make it simpler, we make the following assumptions:

1) The dates supported in the calendar system are from May 9, 2022 to Aug 8, 2022, i.e., the Summer term of 2022.
2) No appointment can be made on weekends and holidays.
3) The time format for each day is using the 24-hour format, from 0:00 to 24:00. We only care about the hour and minute field and ignore the second field.
4) An appointment in the calendar system can only be made at multiples of 30 minutes, and it only supports the same day appointment. For example, May 9th 9:00 to 9:30 appointment is valid, May 9th 10:00 to 12:30 is also valid, May 9th 10:10 to 10:30 is invalid because 10:10 is not a multiple of 30 minutes, May 9th 19:00 to May 10th 9:00 is invalid because it crosses two days.

Let's get started with lab assignment 1 to build some basic elements for this calendar system. **Specifically, in lab assignment 1, you will build the following items.**

1) **[4.5 marks] Implement the *Date* class.** It should have three member variables: ***month*** and ***day***, both of which are ***int*** type; and ***appointed[48]***, which is ***bool*** type array, where appointed[0] denotes if 0:00-0:30 is booked, appointed[1] denotes if 0:30-1:00 is booked, and so on. By default, the appointed array should be initialized as ***false*** before a time slot is booked.
    a. Implement the constructor function(s), get and set functions. For the ***appointed[48]*** array, the get and set function can take an array index as a function argument.

   b. Implement member functions ***isHoliday***, ***isWeekend*** to check if a date is a holiday or a weekend, where no appointment is allowed. More details about the ***isHoliday*** member function are provided in items 3) and 4): basically, use the ***const variable*** holidays and ***equal*** function to implement the ***isHoliday*** member function.
   c. Implement member function ***isValid*** to check if a user input date falls within the valid date range as described in our assumptions. If they do not, print out the error message, and return false.
   d. Implement the ***output*** member function to print out the class information. Add ostream& outs as the function parameter to make this output function more general. In your output function, in addition to print out the member variables, you should also print out whether it is a holiday, weekend, or regular workday. Note your print out information should be easy to understand.
   e. Implement ***printFreeTimeSlots*** member function to print all free time slots of the day that haven't been booked. Implement ***printAppointedTimeSlots*** member function to print all time slots of the day that have already been booked. Note your print out information should be easy to understand, e.g., print out the actual time slots instead of the array indices.

2) **[2.5 marks] Implement the *TimeRange* class.** It should have four member variables: ***beginHour*** and ***beginMinute***, ***endHour*** and ***endMinute***, all of which are ***int*** type. Implement the constructor function(s), get and set functions, and ***output*** function to print out the class information. Similarly, implement member function ***isValid*** to check if the user input beginning time and end time are a multiple of 30 minutes and fall within the valid time range as described in our assumptions.
3) **Use *const variables* to define the three holidays:** Victoria Day (May 23), Canada Day (July 1), and BC Day (August 1).
4) **Use *friend function* to add an *equal* function for two *Date* objects:** Test whether two dates are the same based on the month and day, not including the appointed array. When implementing the friend function, make the function arguments as ***const parameters***. Use the ***const variable*** holidays and ***equal*** function to implement the ***isHoliday*** member function in the Date class.
5) **Make sure both classes are implemented as abstract data type (ADT).**
6) **Make sure your *TimeRange* class is declared and defined in time.hpp and time.cpp, respectively; and the Date class is declared and defined in date.hpp and date.cpp, respectively.**
7) **[2 marks] Test your Date and TimeRange classes in main.cpp:**
   a. Read input from user command line
   b. Use those read data to declare and create Date and TimeRange objects
   c. Print out the Date and TimeRange objects info
   d. Call the set functions to change values of the Date and TimeRange objects, and call the get functions to get the updated values, and print them out again
8) Note that in lab assignment 1, you do NOT have to implement the Calendar book, query, and delete functionalities, which are left to lab assignment 2.

In addition to the actual coding implementation, you need to provide good commenting, naming, and other good coding styles; all these count in your lab assignment marking.

To start the assignment, download the lab1-download.zip from the course website. It includes the following source code: **time.hpp, time.cpp, date.hpp, date.cpp, main.cpp, and Makefile**.

1. After you download lab1-download.zip from the website and put it into a Linux machine
    a. unzip lab1-download.zip //decompress the zip file
    b. mv lab1-download lab1 //rename the directory to lab1
    c. You can see all files under the lab1 directory
2. Your focus should be modifying **time.hpp, time.cpp, date.hpp, date.cpp, main.cpp**.
3. To compile and link your program, type "***make***". You should take a look at the gcc compilation and linking scripts though.
    a. If you develop in another computer other than the lab servers ensc-esil-01 to ensc-esil-24, the compiler we use is gcc version 7.3.0. Keep in mind that your code should be able to compile, run, and demo on one of the lab machines.
4. To run your program, type "***make run***".
5. To clean up your generated files, type "***make clean***".

Note: For grading logistics and remote machine access, please refer to the course website. If you have any questions, please post them on the discussion board.

Assignment Submission:

Your lab assignment 1 will be submitted electronically through Canvas. You will need to submit a single lab1.zip file. Failure to comply with this format will result in a **ZERO** score for this assignment. To zip your files in Linux,

1. Go to your lab1 directory
2. make clean //**make sure you clean your files**
3. cd .. //go one level up
4. zip -r lab1.zip lab //zip all your lab1 files into a single lab1.zip

Submission Deadline:

Your lab assignment 1 is due at **11:59:59pm on Thursday, Jun 2nd, 2022**. You need to meet the deadline: every 10 minutes late for submission, you lose 1 mark; that is, 100 minutes late, you will get zero for this lab.

Lab Demonstration:

You will have to demo your lab assignment 1 to your TA in the following lab sessions that you enrolled on **Monday (Jun 6th, 2022**). Only code from your Canvas submission is allowed in the

lab demo. Each student group has around 10 minutes to explain your code to the TA. If you fail to do the demo (without a medical note), or if it is determined that you do not understand the code being evaluated, you will be awarded zero on this lab assignment. Also please show up in the demo day on time (TA will send out your scheduled time), otherwise you will lose 0.5 mark.