CS6150 - Advanced Programming Lab

July to Nov, 2023

Instructor: Arun Rajkumar

Lab 2

Score: The assignment corresponds to **5** points (out of 100) towards the final grade.

The assignment will contribute to **5 points** towards your final exam. You are expected to submit it by **11:59 PM on 12th August** (Saturday). You should submit the assignment as a zip file containing your .cpp files and your .h files named as #roll_number_Lab2_filename.cpp, #rollnumber_Lab2_filename.h *(example: CS23M001_Lab2_filename.cpp*). However, you must submit a version of your code today before **5 PM** implementing at least the first 3 functionalities listed below - Failure to do this will result in a loss of 1 point.

A Note on Academic Integrity: It is expected that you adhere to the highest standards of academic integrity. This includes no internet help, no plagiarism and no sharing of your code with anyone. Any violation of this will result in immediate reporting to the institute disciplinary committee.

To check against input output you can use https://codeforces.com/problemset/customtest or https://ide.codingblocks.com.

In this assignment you will create a C++ class that helps your client manage their calendar needs. The client has asked you specifically to include the following functionalities:

To be submitted before 5 PM today:

- **The client wishes to obtain the Day, Month and Year of a given date.**
- **The client wishes to know the next weekday of a given date in the year 2023.**
- **The client wishes to know the next holiday of any given date in the year 2023.**

Your task is to implement a class called **Calendar.** You should implement a **Date** class also. Following are the class methods you should implement in order to solve this problem.

- **string Calendar::getDay(Date d) { return "Monday/Tuesday/...."}**
- **string Calendar::getMonth(Date d);**
- **string Calendar::getYear(Date d);**
- **Date Calendar::getNextWeekDay(Date d) {}**
- **Date Calendar::getNextHoliday(Date d) {}**

Date class must have the following members

- **string** dayOfWeek, // stores what day it is. Monday, Tuesday etc.
- **int** month, // any number between [1...12]
- **int** dayOfMonth // any number between [1...31]
- **int** year // any year
- **Constructor Date(int month, int year, int dayOfMonth);**

To calculate the "dayOfWeek" you can implement the logic inside **Date** class and compute dayOfWeek later.

To be submitted either today or before 11:59 PM on August 12th.

- The client wishes to be able to provide any valid date and get the corresponding day.
    - Create a new method called **string Calendar::getDay(Date d) {}**
- The client wishes to be able to set a particular date as a holiday. The client expects all Saturdays and Sundays to be set as holidays by default.
    - Create a new method called **void Calendar::addHoliday(Date d) {}**

- The client wishes to perform addition and subtraction operators to be enabled for the dates. Specifically, the client wishes to be able to add a date with an integer to obtain a valid date that comes the specified integer number of days post the supplied date.
    - Create a new method called **Date Calendar::iterateDates(Date d, int t) {}**
    - **Shift from Date d by t days.** Then return the date.
- The client wishes to know the number of holidays in a given month of a specified year.
    - Create a new method called **int Calendar::getNumberOfHolidays(int month, int year);**
- The client wishes to know if a date preceded a different date. Example (23/08/2023 < 22/08/2023) is true but (23/08/2023 < 24/08/2023) is false.
    - Create a new method called **bool Calendar::isPreceded(Date d1, Date d2);** Return true **if d1 < d2** else false.
- The client wishes to add a note to a given date. For instance the client might want to add a note "Meeting with Alan Turing" to the date 23/08/2023.
    - Create a new method called **void Calendar::addNoteToDate(Date d, string note);**
- The client wishes to be able to remove a note that was added to a date.
    - Create a new method called **bool Calendar::removeNoteFromDate(Date d);** Return true if successfully deleted, otherwise false, if there is no note stored for that date.
- The client wishes to display all notes for the next n days of a given date for a given n.
    - Create a new Data Structure Called **Notes.** Notes should have two member variables, **string content,** and **Date date.**
    - Create a new method called **std::vector<Notes> Calendar::getNotesFromDateTillN(Date d, int n);**
- The client wishes to be able to compute the number of days between any two valid dates.
    - Create a new method called **int Calendar::getDistanceBetweenDates(Date d1, Date d2);**
- The client wishes to print the calendar for any given month of a specified year.
    - Create a new method called **void Calendar::printMonthOfYear(int month, int year);** Should print (instead of returning anything) all the dates line by line, each line should have one week aligned by sunday.
- The client wishes to check if two given users, who have their own meetings noted in their respective calendars, have any set of common days in a given month of an year where they are free to schedule a meeting and if so list the dates (of course, they can't have meetings on holidays).

- Two different users will have two different calendar objects.
- Write a global function **std::vector<Date> getFreeDates(Calendar c1, Calendar c2, int month, int year);**
- **Should return a list of dates** when these Calendars c1 and c2 do not have conflicts.