UMKC School of Computing and Engineering

Assignment 5 Due: Sunday November 3

Date

For this assignment you'll write a simple class hierarchy and use functions, static members and overloaded operators to accomplish needed requirements.

The assignment problem:

You are asked to develop an application to display the date as a combination of month name, day and year. Add/ subtract a day to/from the date. Also you need to calculate the difference between two dates.

Implementation:

Note: the number of days in each month is not fixed and we have leap years with 366 days instead of 365 days.

- Date class will have 3 member variables (month, day, and year)
- Default constructor to set the date to 1/1/1900 (this is the origin of the calendar)
- Overloaded Constructor that takes 3 parameters (month,day,year) to set the values and make sure that it is a valid date.
- Static member data for daysInMonth[] ,daysOfweek[] , monthsOfyear[], startWeekday →(you can set it to 1), startYear→ (set it to 1900).
- bool isValid() const : this member function will check if a date is a valid one or not.
    - Rules for valid date:
        - Month should be between 1 and 12 (Jan to Dec)
        - Day in each month should be between 1 and the days (28, 30, and 31) and between 1 and 29 if the month is Feb and the year is a leap year.
        - Year should be greater than or equal to the starting year (1900)
- bool isLeap(int year) const : This function will check if the year is a leap year or not.
- int findTotalDays() const: is used to calculate the number of days that has passed since January 1st 1900 and a given date. For example, if the input date to this function is February 3rd 1901, the output of the function should be "398 days". If the input date to this function is March 13th 1905, the output of the function should be "1897 days"
- Date& operator++() : overload pre-increment to increment a day
- Date& operator--() : overload pre-decrement to decrement a day
- Date operator++(int): overload post-increment to increment a day
- Date operator--(int) : overload post-decrement to decrement a day
- Date& operator+=(int days) : add days to the given date
- Date& operator-=(int days) : subtract days from the given date.
- bool operator==(const Date& right) const : checks if two date objects are the same
- bool operator!=(const Date& right) const : checks if two date objects are not the same.
- Date& operator=(const Date& right) : assign right to LHS
- friend int operator-(const Date& date1, const Date& date2) : returns the difference between date1 and date 2 (number of days)
- friend ostream& operator<<(ostream& output, const Date& date) : outputs the day of the week, month of the year ,day , and year of date. e.g: "Tue Oct 22 2019"
- void plusReset() function and void minusReset() will be used to adjust the new date if necessary. (As when using the increment and the decrement operators to go to next day or previous day, this may change the month or year of the date).

- **DO NOT CREATE ANY FUNCTIONS EITHER THAN THE ONES I MENTIONED ABOVE.**

- **You can use the following 'main.cpp' to test your code (copy and paste):**

```
int main()
{
	//create two dates and print them
	Date date1(10, 22, 2019);
	Date date2(10, 15, 1944);
	cout << "date1: " << date1 << endl;
	cout << "date2: " << date2<<endl;

	//create two more dates, increment them and print them
	Date date3 = date1;
	Date date4 = date2;
	date3++;
	date4++;
	cout << "date3: " << date3<<endl;
	cout << "date4: " << date4<<endl;

	//Add and subtract days to and from previous dates
	date3 += 20;
	date4 -= 130;
	cout << "date3 after change: " << date3<<endl;
	cout << "date4 after change: " << date4<<endl;

	//find the difference and print the number of days
	cout << "Difference between date3 and date4: " << date3 - date4 << " days" <<
endl;
}
```

Output Sample:

```
date1: Tue Oct 22 2019

date2: Sun Oct 15 1944

date3: Wed Oct 23 2019

date4: Mon Oct 16 1944

date3 after change: Tue Nov 12 2019

date4 after change: Thr Jun 8 1944

Difference between date3 and date4: 27550 days
Press any key to continue . . .
```

Submission:

- Your application must be implemented as an OOP (uses a class, header file, and a main program). Otherwise, it will receive no score.
- Zip up your entire project folder and submit the zip file to Canvas by the deadline.