10802 CPP Midterm Exam

Subject: Date Calculation

Contributor: Cheryl Huang

Main testing concept:

Basics

■ C++ BASICS
□ FLOW OF CONTROL
■ FUNCTION BASICS
□ PARAMETERS AND OVERLOADING
□ ARRAYS
■ STRUCTURES AND CLASSES
□ CONSTRUCTORS AND OTHER TOOLS
■ OPERATOR OVERLOADING, FRIENDS, AND REFERENCES
□ STRINGS
□ POINTERS AND DYNAMIC ARRAYS

Functions

□ SEPARATE COMPILATION AND NAMESPACES
□ STREAMS AND FILE I/O
□ RECURSION
□ INHERITANCE
■ POLYMORPHISM AND VIRTUAL FUNCTIONS
□ TEMPLATES
□ LINKED DATA STRUCTURES
□ EXCEPTION HANDLING
□ STANDARD TEMPLATE LIBRARY
□ PATTERNS AND UML

Description:

Please define and implement a class named **`Date`** that has three data members, **`m_year`**, **`m_month`**, and **`m_day`,** and you should also implement the necessary member functions and functions for the following functionalities.:

1. **Define a constructor**: The constructor should be able to give the default values of **`m_year`**, **`m_month`**, and **`m_day`**.

2. **`check()`** : If the date meets the following criteria, return **`true.`** Otherwise, return **`false`**.
   A. The year should be less than or equal to 2020.
   B. The date should follow general calendar rule.
   For example, if **`m_month`** equals 4, and **`m_day`** equals 31, the **`check`** function should return **`false`**. In general, April only has 30 days.
   In this task, you do **NOT** need to consider the leap year, i.e. 2/29. Please consider that February has 28 days.

3. **Define `operator >`** : If the first date argument (the first operand) is later then the second (the second operand), return **`true`**. Otherwise, return **`false`**.

4. **Define `operator <`** : If the first date argument (the first operand) is earlier then the second (the second operand), return **`true`**. Otherwise, return **`false`**.

5. **Define `operator –`** :
   5.1 **Date – Date** : Return the number of days between the first and the second arguments (operands). For example, the number of days between 2020/1/1 and 2020/1/2 is 1, and the number of days between 2020/1/2 and 2020/1/1 is also 1.)
   5.2 **Date – N days** : Return the date that is N days before the input date.

6. **Define `operator +`** :
   6.1 **Date + N days** : Return the date that is N days after the input date.
   6.2 **N days + Date** : Return the date that is N days after the input date.

To be specific, you have to run the **`main`** function in **"Other notes"** section down below correctly and **can't change any code of the main function**.

Input:

Each line contains a list of numbers for execution a command while the first number of each line indicates the executing command. The following describes the details of each command and its inputs:

1. Command **1** represents **date1 > date2**.
   The input order will be **<year1> <month1> <day1> <year2> <month2> <day2>**.
2. Command **2** represents **date1 < date2**. The input order is same as Command 1.
3. Command **3** represents **date1 - date2**. The input order is same as Command 1.
4. Command **4** represents **date - number**.
   The input order will be **<year> <month> <day> <number of days>**.
5. Command **5** represents **date + number**. The input order is same as Command 4.
6. Command **6** represents **number + date**.
   The input order will be **<number of days> <year> <month> <day>**.
7. Input **0** to exit the program.

Note that all of the input numbers will belong to $\mathbb{Z}^+$, and the first number of each line (the command number) will only contains $\{0, 1, 2, 3, 4, 5, 6\}$.
All of the years of dates are expressed in A.D.
The numbers of inputs are separated by spaces.

Output:

Command 1, 2 should output true or false.
Command 3 should output the number of days.
Command 4, 5, 6 should output the date after calculation.

Sample Input / Output:

Sample Input	Sample Output
1 2019 1 3 2020 5 28	false
2 1997 1 13 2018 7 10	true
3 2019 6 30 2019 9 28	90
4 2017 4 12 379	2016/3/29
5 2020 12 31 366	2022/1/1
6 366 2020 12 31	2022/1/1
0	

□ **Easy. Only basic programming syntax and structure are required.**

■ **Medium. Multiple programming grammars and structures are required.**

□ **Hard. Need to use multiple program structures or more complex data types.**

Expected solving time:

35 minutes

Other notes:

You have to run the main function down below correctly and **can't change any code of it**.

```
int main(void)
{
      enum COMMAND { GREATER = 1, SMALLER, BETWEEN, D_SUB_N, D_PLUS_N,
N_PLUS_D };
      int command, y1, m1, d1, y2, m2, d2, num;
      cin >> command;
      while (command != 0)
      {
            switch (command)
            {
```

```
        case GREATER:
        {
            cin >> y1 >> m1 >> d1 >> y2 >> m2 >> d2;
            Date date1(y1, m1, d1), date2(y2, m2, d2);
            if (date1.check() && date2.check())
                cout << (date1 > date2 ? "true" : "false") << endl;
            else
                cout << "Error Input." << endl;
            break;
        }
        case SMALLER:
        {
            cin >> y1 >> m1 >> d1 >> y2 >> m2 >> d2;
            Date date1(y1, m1, d1), date2(y2, m2, d2);
            if (date1.check() && date2.check())
                cout << (date1 < date2 ? "true" : "false") << endl;
            else
                cout << "Error Input." << endl;
            break;
        }
        case BETWEEN:
        {
            cin >> y1 >> m1 >> d1 >> y2 >> m2 >> d2;
            Date date1(y1, m1, d1), date2(y2, m2, d2);
            if (date1.check() && date2.check())
                cout << (date1 - date2) << endl;
            else
                cout << "Error Input." << endl;
            break;
        }
        case D_SUB_N:
        {
            cin >> y1 >> m1 >> d1 >> num;
            Date date(y1, m1, d1);
            if (date.check())
            {
                Date ansDate = date - num;
                cout << ansDate.m_year << "/" << ansDate.m_month << "/" <<
ansDate.m_day << endl;
            }
            else
                cout << "Error Input." << endl;
            break;
        }
        case D_PLUS_N:
        {
            cin >> y1 >> m1 >> d1 >> num;
            Date date(y1, m1, d1);
            if (date.check())
            {
                Date ansDate = date + num;
                cout << ansDate.m_year << "/" << ansDate.m_month << "/" <<
ansDate.m_day << endl;
            }
            else
                cout << "Error Input." << endl;
            break;
        }
        case N_PLUS_D:
        {
            cin >> num >> y1 >> m1 >> d1;
            Date date(y1, m1, d1);
            if (date.check())
            {
                Date ansDate = num + date;
                cout << ansDate.m_year << "/" << ansDate.m_month << "/" <<
ansDate.m_day << endl;
            }
            else
                cout << "Error Input." << endl;
            break;
        }
        }
        cin >> command;
    }
    return 0;
}
```