CECS 282-01
Advanced C++
Program 2 – myDate Object
Due: Feb 22, 2021

Create a C++ class called myDate. It should have the following operations:

- myDate() – default constructor. This will set the date to May 11, 1959
- myDate(int M, int D, int Y) – overloaded constructor. This will set the date to the values passed in through the parameter list represented by Month, Day and Year.
- void display() – display the date in the following format (May 11, 1959) Do NOT print a linefeed after the date.
- void increaseDate(int N) – increment the date by N days.
- void decreaseDate(int N) – decrement the date by N days.
- int daysBetween(myDate D) – return the number of days between this date and the date D. If date D is a future date, the return value will be a positive int. If date D is in the past, the return value will be a negative int.
- int getMonth() – return the month in integer form
- int getDay() – return the day of the month
- int getYear() – return the year
- int dayOfYear() - return the number of days since the current year began. Example Jan 1 is 1, Feb 1 is 32.
- string dayName() – returns Monday, Tuesday, etc according to the day of the week.

Write a driver program that tests each operation. I will provide a test program to you before the due date.

Note:

You can never have a date that is invalid. The only opportunity for this to happen will be with the overloaded constructor. Therefore if any invalid data is passed to this constructor, ignore all data and set the values to the default date.

I have included an explanation of Julian dates. You can make your life much easier by using this formula. The example code is written in FORTRAN.

You will need to use "pass by reference" for some of the functions in the Julian date converter.

Create 2 functions that are NOT class members. Locate these function in the top portion of the myDate.cpp file. Here are the prototypes for these 2 functions:

`int Greg2Julian(int month, int day, int year);` // pass in the Month, Day, Year and return Julian number

`void Julian2Greg(int JD, int & month, int & day, int & year);` // pass in the Julian Date, and get the correct Month, Day and Year through the parameter list – pass by reference

Program #2 summary – myDate

Description:
Create a class that handles dates based on a specific definition of functions. The student will receive a test driver program the day before the program is due. The student is expected to create and test the myDate class based entirely on the specification.

Objectives:
In this assignment, the student is introduced to Julian dates which will help manipulate and manage dates. Formulas are provided that convert a Gregorian date (example: 5/11/1959) to a Julian date which is the number of days that has elapsed since Nov 28, 4768 BC (or 11/28/-4768). The student will need to use pass-by-reference parameter passing to implement the JulianToGregorian function since 3 parameters (month, day and year) have to be modified within the function and returned to the calling routine.
The following concepts will be learned:

- Reference variables (AKA alias variables)
- Pass by reference
- Pass by value
- Creating and Abstract Data Type (a class with associated values and functions)
- Checking for valid objects at creation time

Converting Between Julian Dates and Gregorian Calendar Dates

The Julian date (JD) is a continuous count of days from 1 January 4713 BC (= -4712 January 1), Greenwich mean noon (= 12h UT). For example, AD 1978 January 1, 0h UT is JD 2443509.5 and AD 1978 July 21, 15h UT, is JD 2443711.125.

Conversion of Gregorian calendar date to Julian date for years AD 1801-2099 can be carried out with the following formula:

$$JD = 367K - \langle(7(K+\langle(M+9)/12\rangle))/4\rangle + \langle(275M)/9\rangle + I + 1721013.5 + UT/24 - 0.5\,\mathrm{sign}(100K+M-190002.5) + 0.5$$

where K is the year (1801 <= K <= 2099), M is the month (1 <= M <= 12), I is the day of the month (1 <= I <= 31), and UT is the universal time in hours ("<=" means "less than or equal to"). The last two terms in the formula add up to zero for all dates after 1900 February 28, so these two terms can be omitted for subsequent dates. This formula makes use of the sign and truncation functions described below:

The **sign** function serves to extract the algebraic sign from a number.
Examples: sign(247) = 1; sign(-6.28) = -1.

The **truncation** function < > extracts the integral part of a number.
Examples: <17.835> = 17; <-3.14> = -3.

Example: Compute the JD corresponding to 1877 August 11, 7h30m UT.
Substituting K = 1877, M = 8, I = 11 and UT = 7.5,
JD = 688859 - 3286 + 244 + 11 + 1721013.5 + 0.3125 + 0.5 + 0.5
= 2406842.8125

The formula given above was taken from the U.S. Naval Observatory's no-longer- published *Almanac for Computers* for year 1990.

Also see our Julian date converter data service.

Fliegel and van Flandern (1968) published compact computer algorithms for converting between Julian dates and Gregorian calendar dates. Their algorithms were presented in the Fortran programming language, and take advantage of the truncation feature of integer arithmetic. The following Fortran code modules are based on these algorithms. In this code, YEAR is the full representation of the year, such as 1970, 2000, etc. (not a two-digit abbreviation); MONTH is the month, a number from 1 to 12; DAY is the day of the month, a number in the range 1-31; and JD is the Julian date at Greenwich noon on the specified YEAR, MONTH, and DAY.

Conversion from a Gregorian calendar date to a Julian date. Valid for any Gregorian calendar date producing a Julian date greater than zero:

```
      INTEGER FUNCTION JD (YEAR,MONTH,DAY)
C
C---COMPUTES THE JULIAN DATE (JD) GIVEN A GREGORIAN CALENDAR
C   DATE (YEAR,MONTH,DAY).
C
      INTEGER YEAR,MONTH,DAY,I,J,K
C
      I= YEAR
      J= MONTH
      K= DAY
C
      JD= K-32075+1461*(I+4800+(J-14)/12)/4+367*(J-2-(J-14)/12*12)
     2    /12-3*((I+4900+(J-14)/12)/100)/4
C
      RETURN
      END
```

Conversion from a Julian date to a Gregorian calendar date.

```
      SUBROUTINE GDATE (JD, YEAR,MONTH,DAY)
C
C---COMPUTES THE GREGORIAN CALENDAR DATE (YEAR,MONTH,DAY)
C   GIVEN THE JULIAN DATE (JD).
C
      INTEGER JD,YEAR,MONTH,DAY,I,J,K
C
      L= JD+68569
      N= 4*L/146097
      L= L-(146097*N+3)/4
      I= 4000*(L+1)/1461001
      L= L-1461*I/4+31
      J= 80*L/2447
      K= L-2447*J/80
      L= J/11
      J= J+2-12*L
      I= 100*(N-49)+I+L
C
      YEAR= I
      MONTH= J
      DAY= K
C
      RETURN
      END
```

Example: YEAR = 1970, MONTH = 1, DAY = 1, JD = 2440588.

Reference: Fliegel, H. F. and van Flandern, T. C. (1968). Communications of the ACM, Vol. 11, No. 10 (October, 1968).