CIST 1400, Introduction to Computer Programming
Programming Assignment

Program 7 : 70 points

Overview of Assignment

For this assignment, you will be processing two dates entered by the user and processing the number of days that have elapsed between the two dates. As an example, it would let you enter your birthdate and today's date and tell you how many days old you are. You will also be calculating some other data based on the number of elapsed days.

About the Dates

We will be processing each date as straight `int` values for the month, the day and the year. An example interaction might look like this for entering the first date, which would be February 23rd, 2016:

```
Enter first date now:
Enter Month: 2
Enter Day  : 23
Enter Year : 2016
```

We will not be processing our dates as `String` data.

When you enter a date, you will need to check to see that it is a valid date. So, you will need to check the following conditions:

- The month should fall in the range of 1 through 12.
- The day should fall in the range of 1 through the appropriate value for the month (31 for January, 30 for April, 28 or 29 for February, etc).
- The year should fall in the range 1900 through 2400.

All range validation on those values should be inclusive (i.e., the year can range from 1900 through 2400, including those endpoints).

Entering and Validating the Dates

Your program will start by asking the user for the first date in the format outlined above, and if the date is not valid for some reason, you will prompt for a date again. Your program shouldn't progress past this until the user has entered a valid date.

If any part of the date is incorrect, your program should prompt for an entirely new date (month, day and year) and not just an updated part of the date. For instance, if the date entered was `0` for month, `13` for day and `2016` for year, then the program would identify the month as being a bad value and prompt for all three values again, not just a new month value. In other words, if the month is bad, your program will not just prompt for a new month but rather an entirely new date to test.

Once you have a valid first date, you will use a similar process to input and validate a second date.

Duration Calcuation

Once you have the two valid dates, you will need to determine how many days have elapsed between them. The one "catch" is that the dates can be entered in any order, so that will need to be handled in your program. The result will always be a non-negative number of days.

Here are some examples. It is assumed that you would enter the date in the correct format in your program; they are listed here in MM/DD/YYYY format for easier presentation.

First Date	Second Date	Days Elapsed	Notes
2/23/2016	2/23/2016	0	Same Day
2/23/2016	2/24/2016	1	Same year First date before second date
2/24/2016	2/23/2016	1	Same year Second date before first date
2/23/2016	2/23/2017	366	Different years First date before second date Leap Day of 2/29/2016 factored in
2/23/2017	2/23/2016	366	Different years Second date before first date Leap Day of 2/29/2016 factored in
2/23/2016	2/23/2015	365	Different years Second date before first date No Leap Day factored in
1/1/1900	1/1/2400	182621	Different years All leap years calculated with 366 days
2/29/2012	4/1/2112	36556	Allowable leap day for first date
2/23/2016	11/30/1955	22000	22000 days, it's not a lot

Additional Results to Calculate

After calcualting the number of days elapsed between two dates, you will also calculate the following information:

- The number of weeks and days between the two dates.
- The number of hours between the two dates.
- The number of minutes between the two dates.

Output Format

Display your output in this order and in a format very similar to this:

```
The total number of days between the dates is 22000
That's a total of 3142 weeks and 6 days.
That's 528000 hours.
That's 31680000 minutes.
```

Goals

- Write a fairly involved program utilizing all of the concepts we have covered so far in class.

Class and File Naming

Name your class `NetID_Dates` and source file `NetID_Dates.java`, using your UNO NetID. For example, account `jsmith` would create a class `jsmith_Dates` in a file named `jsmith_Dates.java`.

Points to Think About

- Because this program only utilizes material from chapters 1 through 4 of the Deitel book, there will be lots of repetitive code in this program.
- Obviously if you're calculating how many days are between two dates that are many years apart, part of the calculation is adding either 365 or 366 for those years that aren't or are leap years between the two dates.
- What about the number of days from those years the dates are in, though? If you can determine which day of the year a particular date is (i.e., the 146th day of the year), then you can more easily add in the remaining days of the earlier date and the days of the later date.

Testing Your Program

- Testing your program with the dates listed previously in the assignment and in the sample runs will be a good bellwether about the correctness of your program.

Grading Notes

- You must have an Honor Pledge on file for the course in order for this program to be graded.
- You must have header documentation at the top of your program. Header documentation is worth 5% of your grade.
- You may only use material through lecture 12 of the class. Using material beyond lecture 12 will result your assignment being returned as not acceptable.
- Even though we have used `String` data in our previous programs, processing dates as `String` data is not allowed on this assignment. All input for this assignment will be `int` data.
- Make sure your output follows the format outlined earlier in the assignment. See also the "Sample Runs" for examples.

Thoughts on Developing This Program

When developing a larger program like this with lots of different parts, it's best to focus on a single task at a time and make sure it's working before moving forward with the rest of the assignment.

For example, in this program, you'll obviously want to make sure you can read a date from the user, so start small by simply inputting the date and then outputting it again to make sure you have read and processed the date correctly.

After that, start thinking about how you would validate an individual date and how you would write code to continuously prompt the user for a date until a good one was entered. Implement code to handle those conditions.

Once you have a single date validation section working, replicate and adapt that code for your second date.

Next, you'll need to work on calculating the elapsed number of days between the dates.

As a first step, you can work on dates in the same years and in ascending order (i.e., first date entered comes before second date entered). That's the simplest situation you need to handle.

Calculating the day of the year for each date, such as February 1st being the 32nd day of the year will enable you to do basic subtraction to calculate the number of elapsed days. Make sure to take into account potential leap days for dates after February.

Once you have tackled this for dates in order, enhance the code to handle dates in the same year that are in reverse order.

Your last major task will be handling dates in different years. The concept is similar to handling dates in the same year except that you need to find the number of remaining days in the earlier date, add the number of complete years between the two dates and then finally add in the day of the year for the later date. Again, you will need to handle the dates being entered in either order.

Finally, calculate the additional results that need to be generated and output those in the format specified earlier in the assignment.

Sample Run

```
Enter first date now:
Enter Month: 1
Enter Day  : 1
Enter Year : 1900

Enter the second date:
Enter Month: 1
Enter Day  : 1
Enter Year : 2400

The first date is: 1/1/1900
The other date is: 1/1/2400

The total number of days between the dates is 182621
That's a total of 26088 weeks and 5 days.
That's 4382904 hours.
That's 262974240 minutes.
```

Sample Run

```
Enter first date now:
Enter Month: 2
Enter Day  : 23
Enter Year : 2016

Enter the second date:
Enter Month: 11
Enter Day  : 30
Enter Year : 1955

The first date is: 2/23/2016
The other date is: 11/30/1955

The total number of days between the dates is 22000
That's a total of 3142 weeks and 6 days.
That's 528000 hours.
That's 31680000 minutes.
```

Sample Run

```
Enter first date now:
Enter Month: 0
Enter Day  : 1
Enter Year : 2016
0 is not a valid month.
Enter Month: 13
Enter Day  : 1
Enter Year : 2016
13 is not a valid month.
Enter Month: -1
Enter Day  : 1
Enter Year : 2016
-1 is not a valid month.
Enter Month: 1
Enter Day  : 0
Enter Year : 2016
0 is not a valid day.
Enter Month: 1
Enter Day  : 32
Enter Year : 2016
32 is not a valid day.
Enter Month: 2
Enter Day  : 30
Enter Year : 2016
30 is not a valid day.
Enter Month: 2
Enter Day  : 29
Enter Year : 2015
29 is not a valid day.
Enter Month: 2
Enter Day  : 23
Enter Year : 1899
1899 is not a valid year.
Enter Month: 2
Enter Day  : 23
Enter Year : 2401
2401 is not a valid year.
Enter Month: 2
Enter Day  : 24
Enter Year : 2016
```

```
Enter the second date:
Enter Month: 1
Enter Day  : 1
Enter Year : 1899
1899 is not a valid year.
Enter Month: 1
Enter Day  : 1
Enter Year : 2401
2401 is not a valid year.
Enter Month: 0
Enter Day  : 1
Enter Year : 2016
0 is not a valid month.
Enter Month: 1
Enter Day  : 1
Enter Year : 5000
5000 is not a valid year.
Enter Month: 2
Enter Day  : 29
Enter Year : 1900
29 is not a valid day.
Enter Month: 2
Enter Day  : 28
Enter Year : 1900

The first date is: 2/24/2016
The other date is: 2/28/1900

The total number of days between the dates is 42364
That's a total of 6052 weeks and 0 days.
That's 1016736 hours.
That's 61004160 minutes.
```