AP Computer Science — Project 5 - Operators
Dr. Paul Bailey — Tuesday, September 13, 2022

Due Sunday, September 25, 2022, at 11:59 PM.

The purpose of this project is to practice with *operators*. We do this in the context of one of the more complex basic programming problems, dealing with dates.

In Project 4, we saw that an *external date* is a string of the form `MM/DD/YYYY`, where `MM` is a number from 1 to 12 (the month), `DD` is a number from 1 to 31 (the day of the month), and `YYYY` is a number between 1000 and 9999 (the year).

We wish to implement the concept of an *internal date*, which is an integer defined to be the number of days since a given (fixed) date. We wish to be able to convert between internal date (an integer) and external date (a string of the form MM/DD/YYYY).

One standardized form of internal date is *Julian date*, as explained at

`http://en.wikipedia.org/wiki/Julian_day`.

To quote this source, "Julian day number 0 is assigned to the day starting at noon on Monday, January 1, 4713 BC, proleptic Julian calendar." We will use Julian day number as our internal date. The code to convert from one to the another is given. The student is tasked with using these methods.

Create one directory called `P5_Operators` to store the `.java` files. Copy the `Program` and `Tool` classes from Project 4 into this directory.

The `Date` class will contain the date computation methods.

Program 1. Create a new class called `public class Date` in a file called `Date.java`.

Move the `formatDate`, `pullDate`, `checkDate`, and `countMonths` methods from `Tool` to `Date`. After you do this, there should be no date methods in the tool class.

Modify your `Program` to access these methods in the `Date` class instead of the `Tool` class. This sort of rearranging is part of *refactoring*.

Program 2. Add the following methods to the `Date` class.

```
public static int internalDate(int M, int D, int Y)
{
    int a = (14 - M) / 12;
    int y = Y + 4800 - a;
    int m = M + 12 * a - 3;
    int J = D + ((153 * m + 2)/5) + 365 * y + (y / 4) - (y / 100) + (y / 400) - 32045;
    return J;
}

public static int internalDate(String s)
{
    int M = pullDate(s, 1);
    int D = pullDate(s, 2);
    int Y = pullDate(s, 3);
    return internalDate(M, D, Y);
}

public static int currentDate() // current internal date
{
    long unixTime = System.currentTimeMillis();
    int J = (int)((unixTime / 1000L) / 86400L) + 2440588;
    return J;
}
```

Modify the `Program.test1` method to output the internal date of the date that had been input.

To check that you have typed the date conversion correctly, we give some internal dates.

Date	External Date	Internal Date
January 1, 1000	01/01/1000	2086303
July 4, 1776	07/04/1776	2369916
January 1, 1900	01/01/1900	2415021
January 1, 1970	01/01/1970	2440588
September 20, 2021	09/20/2021	2459478
December 25, 5000	12/25/5000	3547631
December 31, 9999	12/31/9999	5373484

Program 3. In the `Date` class, write a method

```
public static int countDays(String externalDate1, String externalDate2)
```

which counts and returns the number of days between to given dates. It should return a positive number.

Add code to the `Program.test2` method to test the `countDays` method.

We will define *internal day of the week* as follows:

Internal DOW	External DOW
1	Sunday
2	Monday
3	Tuesday
4	Wednesday
5	Thursday
6	Friday
7	Saturday

Program 4. In the `Date` class, write a method

```
public static int dayOfWeek(String externalDate)
```

which returns the internal day of the week of the given date. This will use the *modulo* operator and the fact that September 19, 2021 was a Sunday.

Add code to the `Program.test1` method to output the internal day of the week.

Program 5. Add the following method to the `Date` class. It takes an internal date and converts it to an external date.

```
public static String externalDate(int J)
{
    int y = 4716;
    int j = 1401;
    int m = 2;
    int n = 12;
    int r = 4;
    int p = 1461;
    int v = 3;
    int u = 5;
    int s = 153;
    int w = 2;
    int B = 274277;
    int C = -38;
    int f = J + j + (((4 * J + B) / 146097) * 3) / 4 + C;
    int e = r * f + v;
    int g = (e % p) / r;
    int h = u * g + w;
    int D = (h % s) / u + 1;
    int M = ((h / s + m) % n) + 1;
    int Y = (e / p) - y + (n + m - M) / n;
    return formatDate(M, D, Y);
}
```

To test this new method, add code to the `Program` class to take an internal date and write it as an external date.

Program 6. Add the following method to the `Date` class.

```
public static int currentDate() // current internal date
{
    long unixTime = System.currentTimeMillis();
    int J = (int)((unixTime / 1000L) / 86400L) + 2440588;
    return J;
}
```

Modify the `Program.main` method to output the current internal date and the current external date.