WeekdaysFunctions

Calculate the day of the week.

Introduction

We all use electronic calendars almost every day and we take it for granted that the calendar knows the weekday for any given date. It is so common place that we don't spend much time thinking about how it is done. This exercise will not only show you how it is done but also that it is not as straight forward as one might think. Luckily, there is a formula for dates after Oct. 15th 1582, derived by C.F. Gauss:

$A = d + [2.6m - 0.2] + y + [y/4] + [c/4] - 2c$

where

$W = Amod7$

with

- $[x]$ is the Gaussoperator: $[x] := x'$ the biggest integer x' such that $x' <= x$ (essentially rounding **down**, when dealing with positive numbers)
- d represents the day ($1 <= d <= 31$)
- y represents the last two digits of the year
- c represents the first two digits of the year (actually all but the last two; c stands for century)
- m represents the month (see Table 1)
- the number W can be used to look up the weekday in Table 2

Table 1: m lookup by month

Month	m
Mar	1
Apr	2
May	3
Jun	4
Jul	5
Aug	6
Sep	7
Oct	8
Nov	9
Dec	10
Jan	11
Feb	12

Table 2: weekday lookup by W

W	weekday
0	Sun
1	Mon
2	Tue
3	Wed
4	Thu
5	Fri

W	weekday
6	Sat

However, there is more to it: Jan and Feb always belong to the 'old' year: Jan 2004 is actually the 11th month of 2003. (This is why Sep, Oct, Nov, and Dec are called something like 'seven', 'eight', 'nine', 'ten').

Example Calculation

Let us calculate the weekday of Jan 1 st of the year 2000. January 2000 is actually month 11 of the year 1999. This leads to the following values for the variables: $d = 1, y = 99, c = 19, m = 11$. Filling in the values we get:

$A = 1 + [2.6 \cdot 11 - 0.2] + 99 + [99/4] + [19/4] - 2 \cdot 19$

$A = 1 + [28.4] + 99 + [24.75] + [4.75] - 38$

$A = 1 + 28 + 99 + 24 + 4 - 38 = 118$

$W = 118 mod 7 = 48 mod 7 = 6$

Looking up 6 in the Table gives the final result: Jan, 1st 2000 was a Saturday.

Assignment

Write a program that asks the user for a calendar date, validates the date and afterwards calculates and prints the weekday of that date. Your program must not accept dates that cannot exist.

Requirements

- The user enters the date in the YYYYMMDD format
- Any impossible date is rejected:
  - dates before Oct. 15th 1582 or after Dec, 31st, 2199
  - impossible month (<1 or >12)
  - impossible days (e.g. any day>31, a day>30 for some months, a day>28 in February of a non-leap year, etc)

Implement the following functions:

1. `public static boolean validate(int year)`
   - returns whether the given year is in the accepted range or exists at all
   - returns true if year is valid, false otherwise
   - the earliest valid date is Oct. 16th 1582, the last date accepted is Dec, 31st, 2199.
2. `public static boolean validate(int year, int month)`
   - returns whether the given year and month combination actually exists/existed and is in the accepted range
   - the earliest valid date is Oct. 16th 1582, the last date accepted is Dec, 31st, 2199.

3. `public static boolean isLeap(int inYear)`

   -returns if the given year is a leap year. Check ExerciseSheet02 `LeapChecker` for more information

4. `public static int nDays(int month, int year)`

   - returns the number of days in month of year

5. `public static boolean validate(int year, int month, int day)`

   - returns whether the given date actually exists/existed
   - returns true if date is valid, false otherwise
   - the earliest valid date is Oct. 16th 1582, the last date accepted is Dec, 31st, 2199.

6. `public static int weekday(int inDay, int inMonth, int inYear)`

   -calculates and returns the number of the weekday of a given date using the formula explained before

   - the date is passed to the function as three ints.
   - returns the weekday according to Table 1

7. `public static String dayName(int W)`

   - returns the name of the Weekday specified by the given number (refer to Table 2 in this Exercise on more information)

Console Example

(text in red is user input)

```
enter date (yyyymmdd): 20000101
Saturday
```

```
enter date (yyyymmdd): 20000001
invalid date (20000001)
```

Hints

- the modulo operator % in Java returns negative numbers if the operands are negative. To look up the day in the weekday table we need a number between 0 and 6. So if the result of *W* is negative you have to add 7.
- use the `main` function in the provided `Main` class for testing (see and rename file `Main.use_this`):

```
public static void main(String[] args) {
    System.out.print("enter date (yyyymmdd): ");
    int date = sc.nextInt();

    int inDay = date % 100, inMonth = date % 10000 / 100, inYear = date / 10000;

    if(validate(inYear, inMonth, inDay)){
        int W = weekday(inDay, inMonth, inYear);
        String weekday = dayName(W);
```

```
            System.out.printf("%s",  weekday);

        }else {
            System.out.printf("invalid date (%d)\n", date);
        }
    }
```

Testing

Test your program (at least) against the test cases provided:

Table 3: some test data for Weekdays problem

Date	expected result
01.13.2000	reject
01.00.2000	reject
32.12.1999	reject
31.11.2004	reject
31.04.2000	reject
30.02.2003	reject
29.02.2003	reject
29.02.1900	reject
31.12.1581	reject
15.11.2004	mon
29.02.2000	tue
16.02.2096	thu
01.02.1980	fri
08.03.1988	(super Tuesday) tue
30.01.1972	(bloody Sunday) sun
25.10.1929	(black Friday) fri
29.02.2004	sun
28.02.2003	fri
03.03.1919	mon