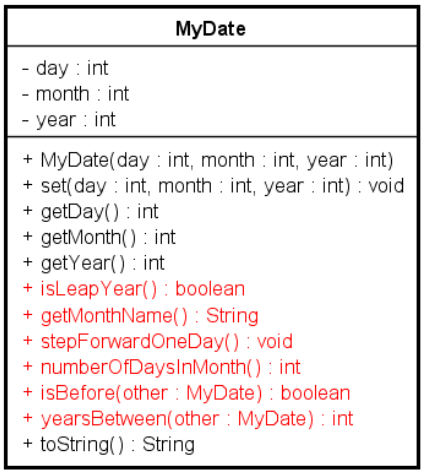
Exercise: MyDate, version 2

Create a new Module and name it `MyDate_v2` – and copy the files from `MyDate_v1`. *If you did not make MyDate_v1, then start with this exercise (given at the end of this document)*



Modify class `MyDate`:

a) In class `MyDate` include a `boolean` method `isLeapYear` that returns `true` if the year is a leap year, otherwise the method returns `false`.

We have a leap year every 4th year except for the 100-years not divisible with 400. This means that we have a leap year if a division with the year and 4 gives a zero reminder and the case that the division with the year and 100 gives a nonzero reminder or the year and 400 also gives a zero reminder. This can be formulated in pseudo code this way:

```
IF (year divisible by 4) AND ((year not divisible by 100) OR (year divisible by 400))
  THEN year is a LEAP YEAR
ELSE
  year is NOT a LEAP YEAR
```

Note: the method may be implemented in one statement.

b) Add a method `numberOfDaysInMonth` returning the number of days in the given month.

Hint: If month is February (month equal to 2) and it is a leap year then there is 29 days. If it is not a leap year, February has 28 days. April, June, September and November (month equal to 4, 6, 9 or 11) all have 30 days. The remaining months January, March, May, July, August, October and December (month equal to 1, 3, 5, 7, 8, 10 or 12) all have 31 days.

c) Modify method `set` now checking for legal dates:

- If a year is negative, the year should be set to its positive equivalent.
- If a month is less than 1 then month should be set to 1 (January).
- If a month is larger than 12 then month should be set to 12 (December).
- If a day is less than 1 it should be set to 1. If the day is larger than the last day in the given month then it should be set to the last day in the month.

Examples:

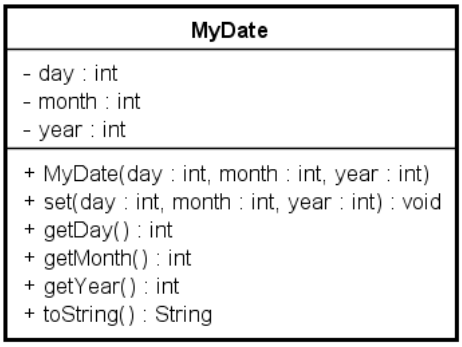
`day=31, month=10; year=-2010` becomes the date `31/10/2010`
`day=31, month=4; year=2010` becomes the date `30/4/2010`
`day=31, month=2; year=2012` becomes the date `29/2/2010`
`day=31, month=2; year=2010` becomes the date `28/2/2010`
`day=0, month=0; year=2010` becomes the date `1/1/2010`

Call `set` from the constructor instead of implementing the same logics in the constructor too.

d) Create a method `getMonthName` returning the month as a string. If month is 1 then “January” is being returned, month = 2 will return “February”, month = 3 will return “March” and so forth.
e) Create a method `stepForwardOneDay` updating the date to the next day. Example 31/01/2000 will be updated to the date 01/02/2000 after calling `stepForwardOneDay`.
f) Create a method `numberOfDaysInMonth` returning the number of days in the month.
g) Create a method `isBefore` returning true if the date is before the date given as argument to the method.
h) Create a method `yearsBetween` returning the number of full years between the date and the argument-date. Example: date1=21/03/2000, date2=24/03/2005, date1.yearsBetween(date2) gives 5 while date1=24/03/2000, date2=21/03/2005, date1.yearsBetween(date2) gives only 4. The method is symmetric i.e. date1.yearsBetween(date2) equals date2.yearsBetween(date1).
i) Modify method `toString()` such that you return a string in the format “dd/mm/yyyy”, i.e. two digits for day and month. Example: 5 January 1996 will be returned as "05/01/1996" (and not as “5/1/1996”)

The following is the MyDate_v1 exercises (AND ONLY NEEDED IF YOU DID NOT MAKE IT)

Exercise: MyDate, version 1



Create a new Module in IntelliJ and name it `MyDate_v1`

Create a class called `MyDate` that represents a date including three pieces of information as instance variables, a month (type `int`), a day (type `int`) and a year (type `int`).

a) A constructor that initializes the three instance variables and assumes that the values provided are correct. (you have to call method `set(…)` in question b)
b) A void method `set(int day, int month, int year)` that sets/changes all three instance variables with the values given as arguments.
c) A get method for each instance variable.
d) A method `toString()` that return a string with the date information in the format “day/month/year”, i.e. the day, month and year separated by forward slashes (`/`). Example: 23 January 1996 (release date for the first Java version) will be returned as "23/1/1996"