TP n° 4

Exercice. Représenter des dates en Java

On veut créer une classe `Journee` qui représente une date [1] Une date est composée d'un jour, d'un mois, et d'une année.

1. Construire une telle classe `Journee`, avec le constructeur et les accesseurs pour tous les attributs.
2. Écrire une fonction `toString` (le nom est important) qui retourne une représentation textuelle de la date au format AAAA-JJ-MM ; on ne se préoccupera pas d'éventuels '0' pour les nombres inférieurs à 10, il s'agit avant tout de faire une fonction qui nous permette de débugger. En effet, cette fonction sera utilisée automatiquement par Java pour transformer une `Journee` en `String` :
   ```
   System.out.println("Ma date: "+j);
   ```
3. Compiler cette classe `Journee`.
4. Afin de tester cette classe, on va utiliser une seconde classe `Calendrier`, qui comportera la méthode principale `main`. Dans un premier temps, on entrera à la main des valeurs d'entrée dans ce main, et on affichera les résultats à l'aide de `System.out.println` ; par exemple :
   ```
   Journee j = new Journee(18,1,2016);
   System.out.println(j.getJour()); // doit afficher "18"
   System.out.println("Ma date: "+j); // doit afficher "Ma date: 2016-1-18"
   ```
   a) Créer cette classe `Calendrier` avec une méthode `main` comme ci-dessus.
   b) Compiler et exécuter la classe `Calendrier`.
   c) Ajouter d'autres tests pour vérifier que les accesseurs fonctionnent correctement. À chaque fois, compiler et exécuter la classe `Calendrier`.

Dans les questions qui suivent, on utilisera à chaque fois la méthode `main` pour faire les tests. On devra donc recompiler les deux classes (`Journee` et `Calendrier`) avant d'exécuter `Calendrier`.

5. Dans le classe `Journee`, construire une opération de classe qui teste si une année est bissextile. Une année est bissextile si elle est multiple de 4 (par exemple 2016 qui est bissextile), mais pas si elle est multiple de 100 (par exemple 1900 n'était pas bissextile), hormis dans les cas où elle est multiple de 400 (c'est pour cela que 2000 était bissextile).
6. Écrire une opération de classe qui retourne pour un mois et une année le nombre de jours qu'il y a dans le mois. Tester avec février 2015, avril 2015, janvier 2016 et février 2016.
7. Écrire un opération qui détermine si une `Journee` est valide, c'est à dire si elle correspond à une vraie date. Tester avec le 32/12/2015, le 05/13/2016, le 29/02/2016, le 31/07/2016, le 29/02/2017.
8. Écrire une opération qui donne le jour de la semaine correspondant à une date. Pour cela, on utilise la formule suivante :

   $\text{jour\_dans\_semaine} = (\text{jour} + \text{code\_mois} + \text{annee\_sans\_siècle} + \lfloor \frac{\text{annee\_sans\_siècle}}{4} \rfloor + \text{code\_siècle}) \bmod 7$

   Ensuite, si jour_dans_semaine=0, c'est que l'on est samedi ; si jour_dans_semaine=1, c'est que l'on est dimanche, etc.
   On fournit des fonctions qui donnent le code_mois à partir du mois et de l'année et le code_siècle à partir du siècle dans la classe `AideCalendrier`. Pour pouvoir les utiliser, il faut récupérer le fichier `AideCalendrier.class` et le mettre dans le même dossier que `Journee.java` et `Calendrier.java`. Il n'y rien à compiler et vous n'avez pas accès au code de `AideCalendrier`, seulement aux signatures des fonctions :
   ```
   public static int codeSiecle(int siecle) {}
   public static int codeMois(int mois, int annee) {}
   ```
   Notez que ici « siècle » signifie « l'année divisée par 100 », donc le « siècle » de 1998 est 19, même si 1998 est au 20e siècle. L'année_sans_siècle de 1998 est 98.

1. Java dispose d'une classe `Date`, mais il n'est pas autorisé de s'en servir pour ce TP !

9. Écrire une méthode qui écrit une date joliment, en se servant du jour de la semaine, par exemple : « lundi 18 janvier 2016 » ; on pourra utiliser la fonction `moisFr` de la classe `AideCalendrier` qui transforme un numéro de mois en sa valeur en français :

```
public static String moisFr(int mois) {}
```

On veut maintenant se servir de la classe Calendrier pour réellement faire l'interface avec l'utilisateur.

10. Écrire une opération de classe qui à partir d'une chaîne de caractères (`String` que l'on suppose de la forme AAAA-MM-JJ, construit un objet `Journee` correspondant. Pour vous aider :
    - Un objet String dispose d'une opération `toCharArray` qui retourne un tableau des caractères formant la chaîne : `"Bonjour".toCharArray()` ⇝ `['B','o','n','j','o','u','r']`
    - Même si l'on suppose que l'entrée est correcte, il est utile de s'assurer que le tableau de caractères obtenu a la bonne longueur, afin d'éviter des erreurs si l'on souhaite accéder à une valeur de ce tableau.
    - Une méthode simple pour convertir un caractère en un chiffre : `int chiffre = caractere-'0';`
    - Même si l'on suppose que l'entrée est correcte, on peut tout de même vérifier que le chiffre obtenu est bien entre 0 et 9 ; s'il ne l'est pas, retourner une date invalide, par exemple 0000-00-00.
11. Écrire une fonction qui prend en entrée une chaîne de caractères (`String` que l'on suppose de la forme AAAA-MM-JJ et qui retourne un message d'erreur si la date est invalide et une jolie version de la date si elle est valide. On peut y joindre la date telle que donnée en entrée.
    Par exemple :
    - `"2015-02-29: date invalide!"`
    - `"2016-02-29: lundi 29 février 2016"`
12. Écrire une vraie fonction `main` qui pour chaque date donnée en argument dans la ligne de commande sous la forme AAAA-MM-JJ affiche une erreur ou la date joliment selon si elle est valide ou non. On pourra également mettre un petit message d'aide si l'utilisateur n'avait donné aucune date. Par exemple :

```
etudiant@etudiant01$ java Calendrier
Usage: donnez des dates au format AAAA-MM-JJ, notre programme les affichera joliement.
etudiant@etudiant01$ java Calendrier 2016-01-18 2015-02-29 2016-02-29
2016-01-18: lundi 18 janvier 2016
2015-02-29: date invalide!
2016-02-29: lundi 29 février 2016
etudiant@etudiant01$ java Calendrier 2016-02-030 2015-1-1 2015-01-01
2016-02-030: date invalide!
2015-1-1: date invalide!
2015-01-01: jeudi 1 janvier 2015
```