Interface, interface, classe anonyme et énumération

Exercice 1 - Vive l'europe, les plaques changent.

Et oui l'europe c'est l'harmonie. Même les plaques d'immatriculation changent. Le nouveau format des plaques d'immatriculation sera :

```
AA 111 AA
```

(2 lettres maximum / 3 chiffres maximum / 2 lettres maximum)

1 Dans un premier temps proposez une nouvelle implémentation de la classe `NumberPlate` nommée `EuropeanNumberPlate`.

2 En fait, on souhaite pouvoir gérer la transition, c-a-d avoir un parc de voiture pouvant posséder des plaques à l'ancienne norme ou à la nouvelle.
Comment doit-on modifier `Car`, `NumberPlate` et `EuropeanNumberPlate` ?

3 Implémenter la solution retenue.

4 On souhaite pouvoir créer des plaques d'immatriculation en utilisant une méthode possédant un paramètre de type `boolean` permettant de créer des plaques de l'ancien ou du nouveau type.
Indiquer quel est le profil de cette méthode ?
Dans quelle classe doit-on mettre cette méthode ?

Exercice 2 - Manipulation de fichier.

Dans cet exercice on propose de d'utiliser le package `java.io`. Ce pakage contient une serie de classes qui permettent de manipuler les entrées sorties, les fichiers ainsi que la serialisation.

1 Affichez les fichiers et répertoires d'un répertoire en utilisant la classe `java.io.File`, penser à lire la documentation de `java.io.File`.

2 La méthode `listFiles()` est surchargée, rappelez ce que cela veut dire.

3 Quelle est la différence entre les classes `java.io.FileFilter` et `java.io.FilenameFilter`

4 Affichez les fichiers d'un répertoire dont le nom commence par une chaîne prise en paramètre en utilisant un filtre.

5 Faire la même chose en utilisant une classe anonyme à la place d'une classe "top-level".

6 Affichez récursivement l'ensemble des répertoire à partir d'un repertoire donné en utilisant un filtre.
Attention à ne pas allouer trop d'objets inutilement.

7 Écrire une méthode `listFile(File,FileFilter)` en utilisant la méthode `File.listFile(FileNameFilter)` de `java.io.File`.
On rappelle qu'il existe un constructeur `File(File directory, String name)`.

Exercice 3 - Avec style

On souhaite pouvoir écrire des textes de façon marrante sur la console. On définit pour cela la méthode :

```
String stylify(String text,int flags)
```

L'entier `flags` indique la façon dont le texte doit s'afficher. Avec le texte `toto`

1 STAR affiche `t*o*t*o`
2 REVERSE affiche `otot`
3 REVERSE | STAR affiche `o*t*o*t`

1 Déclarer STAR et REVERSE comme des constantes entières.
2 Ecrire la méthode `stylify`.
3 Quel est l'intérêt de remplacer les constantes par une énumération ?
  Transformer les constantes en énumération et modifier la méthode `stylify` en conséquence.
  Note : Vous pouvez regarder la documentation de la classe `EnumSet`.
4 Ajouter une méthode `applyStyle` dans l'énumération.
  Quel est l'intérêt de ce design ?
  Changer le code de `stylify` en conséquence.
5 Ajouter un flag BANG qui affiche le texte `toto` comme ceci `t!o!t!o`. Partagez le code au maximum !
  En cas de BANG/STAR, le STAR devra être traité d'abord (le résultat est alors `t!*!o!*!t!*!o`).
6 Que doit-on modifier pour pouvoir choisir si l'on veut faire un STAR avant ou après un BANG dans le cas d'un STAR/BANG ?
  L'astuce consiste à utiliser un ensemble qui conserve l'ordre d'insertion de ses élements en argument de `stylify`.