TP JAVA – Gestion d'une bibliothèque

Création des classes *Adherent* & *Livre*

On s'intéresse à réaliser une application de gestion d'une bibliothèque et particulièrement :

- La gestion des adhérents
- La gestion des livres
- La gestion des emprunts
- Et la gestion des retours

Un adhérent est caractérisé par son nom, prénom et son âge.
Un adhérent est identifié par un matricule de type String avec les initiales de son prénom et un numéro séquentiel à 3 chiffres qui commence par 100. (Ex : NP101).

Un livre est caractérisé par un titre, un auteur et l'année d'édition.
Un livre est identifié par sa cote, qui est une combinaison des deux premières lettres du nom d'auteur, suivi des deux derniers chiffres de l'année d'édition suivi d'un numéro séquentiel à deux chiffres, qui commence par 10. (Ex, AU17-10)

Question 1

1. Définir les deux classes **Adherent** et **Livre** avec un constructeur ave paramètres chaqu'une.
2. Ecrire une fonction **generateMatricule()**, qui permet à chaque appel du constructeur de générer automatiquement le matricule de la personne
3. Ecrire une fonction **generateCote()**, qui permet à chaque appel du constructeur de générer automatiquement la cote.
4. Ecrire un programme main qui permet de créer deux instances un livre et un adhérent, puis les afficher sur la console.
5. Redéfinir les méthodes **toString** () de chaque classe, ré-exécuter le programme, que constatez-vous ?

Initialisation des données de la base

On s'intéresse de sauvegarder les livres et les adhérents en mémoire, pour ce faire, nous allons utiliser des map

Rappel sur les Maps :

```
//déclaration
Map<String, Object> adherentsMap = new HashMap<String, Object>();

//Inserer un élement dans une map
String key = "Clé";
Object value = new Object();
adherentsMap.put(key, value);

//Récupérer un element d'une map
adherentsMap.get(key);
```

Question 2

1. Initialiser les maps ***adherentsMap*** et ***livresMap*** avec des données tests
   Pour tester le résultat de l'initialisation, il faut créer deux méthodes :
   - ***public List<Adherent> getAllAdherents ()***
   - ***public List<Livre> getAllLivres ()***

L'emprunt :

Un adhérent se présente à l'accueil de la bibliothèque, présente sa carte d'adhérent (Matricule) et demander d'emprunter un livre (cote), l'action d'emprunter consiste à faire une relation entre cette personne et ce livre, de telle sorte qu'on puisse savoir à tout instant :

- Pour chaque personne, la liste de livre qu'il a emprunté.
- Pour chaque livre, s'il est disponible ou pas, si non, quel personne l'a emprunté.
- Le nombre de livre que peut encore emprunter une personne.

NB : Un adhérent ne peut pas emprunter plus de 3 livres au même temps.

Question 3

1. Implémenter la méthode ***emprunter (String matricule, String cote).***
   Pour ce faite il y a besoins de créer deux méthodes
   - ***public Adherent findAdherentByMatricule (String matricule)***
   - ***public Livre findLivreByCote (String cote)***
2. Modifier le programme main comme suite : choisir un adhérent (matricule) et faites lui faire 4 emprunts successives (choisir 4 cotes).
3. Redéfinir les méthodes **toString** () de la classe ***Adherent*** de telle sorte à visualiser pour chaque instance, la liste des livres qu'il a emprunté, ré-exécuter le programme, que constatez-vous ?
4. Redéfinir les méthodes **toString** () de la classe ***Livre*** de telle sorte à visualiser pour chaque instance, la disponibilité ou pas du livre, et éventuellement, la personne qu'il a emprunté, ré-exécuter le programme, que constatez-vous ?

La restitution :

L'adhérent se présente à l'accueil de la bibliothèque, présente sa carte d'adhérent et le livre qu'il souhaite restituer, l'action de restituer consiste à couper cette relation entre la personne et le livre de telle sorte que :

- Que le livre devient à nouveau disponible
- Augmenter la capacité d'emprunt de la personne de 1.

NB : Un livre n'est disponible qu'on un seul exemplaire, s'il est emprunté une fois, une autre personne ne peut l'emprunter que si le livre est restitué.

Question 4

1. Implémenter la méthode ***restituer (String matricule, String cote).***
2. Modifier le programme main comme suite : Choisir un adhérent (matricule) et faites lui faire 4 emprunts successives (choisir 4 cotes), puis une restitution d'une cote de votre choix, puis retenter le dernier emprunt.

Question 5

1. Ecrire une méthode qui permet à tout instant de retourner la liste des livres disponibles.
   - ***public List<Livre> listerLivresDisponibles () ;***
2. Ecrire une méthode qui permet à tout instant de retourner la liste des livres empruntés.
   - ***public List<Livre> listerLivresEmpruntes () ;***