

Le BiblioBus

On s'occupe de la gestion d'un bibliobus.

La classe Livre

Dans un premier temps, vous devez implémenter une classe `Livre` qui répond au cahier des charges suivant. Un livre est ici à comprendre comme un livre dans un catalogue, pas comme un exemplaire physique d'un livre.

1. Un livre est décrit par cinq informations : son titre, le nom de son auteur, son éditeur, le nombre d'exemplaires présents dans la bibliothèque, et son genre (*littérature française, littérature jeunesse, policier...*);
2. un livre a toujours un titre, un auteur et un éditeur qui ne peuvent pas changer ;
3. le nombre d'exemplaires ne peut pas être négatif. On peut ajouter soit un seul exemplaire à la fois, soit plusieurs exemplaires ;
4. la perte d'un livre étant assez rare, on ne peut signaler que la perte d'un exemplaire à la fois ;
5. on veut savoir si un livre est réellement présent dans le bibliobus, i.e. s'il y en a au moins un exemplaire ;
6. un livre est capable de s'identifier, en affichant les informations qui le caractérisent ;
7. on veut savoir si deux livres sont représentatifs de la même oeuvre (même titre, même auteur, même éditeur) ;
8. lorsqu'un livre change d'éditeur, on doit créer un nouveau livre avec le même titre, le même auteur, le même genre et le nouvel éditeur.

Exercice 1 : Classe Livre, classe DemoLivre

Commencez par créer deux fichiers contenant respectivement une classe `Livre` et une classe `DemoLivre`. Dans la classe `Livre`, écrivez tous les attributs nécessaires. Dans la classe `DemoLivre` implémentez seulement une méthode `main`. **Après chaque écriture d'une nouvelle méthode ou d'un constructeur pour la classe Livre, vous devrez tester votre implémentation dans la classe DemoLivre.**

Exercice 2 : Constructeurs

Écrivez un constructeur pour vérifier la première partie du point 2). Tant que le genre d'un livre n'est pas précisé, il contient la valeur *"Non spécifié"*.

Comment permettre de spécifier le genre d'un livre à sa création ?

Exercice 3 : Accesseurs

Comment être sûr que ni le titre, ni l'auteur ni l'éditeur du livre ne peuvent changer ? Implémentez tous les accesseurs.

Exercice 4 : Achat d'exemplaires, perte d'un livre

Implémentez les méthodes `nouvelExemplaire()`, `nouvelExemplaire(int nb)` pour l'ajout d'un ou de `nb` exemplaires d'un livre, et la méthode `perteExemplaire()`.

Exercice 5 : Donner ses caractéristiques

Implémentez la méthode `toString()` qui retourne une chaîne de caractères. Cette méthode

construit la chaîne de caractères qu'on souhaiterait voir si on veut afficher toutes les caractéristiques d'un livre.

Exercice 6 : Même livre

On dira que deux livres sont égaux s'ils ont le même titre, le même auteur et le même éditeur. Implémentez la méthode `equals(Livre l)` qui retourne un booléen. Cette méthode retourne vrai si le livre passé en paramètre et l'instance courante (celui qui exécute la méthode) sont les mêmes, faux sinon.

Exercice 7 : Changement d'éditeur

Implémentez la méthode `nouvelEditeur(String unEditeur)` qui retourne une nouvelle instance de `Livre` ayant le même titre, le même auteur et le même genre que le livre qui exécute cette méthode.

Le Bibliobus

On désire maintenant gérer le bibliobus. Dans un premier temps, on se limitera aux opérations suivantes :

- retourner les caractéristiques d'un livre dont on connaît l'identifiant ;
- afficher les caractéristiques d'un livre dont on connaît l'identifiant ;
- afficher la liste des livres du catalogue du bibliobus (i.e., les livres connus du bibliobus) ;
- afficher la liste des livres réellement présents dans le bibliobus ;
- afficher la liste des livres d'un genre donné ;
- indiquer si un livre donné est connu du bibliobus ou non ;
- indiquer si un livre donné est réellement présent dans le bibliobus ou non (à partir de son identifiant, ou d'une instance d'un livre) ;
- donner le nombre d'exemplaires d'un livre particulier (à partir de son identifiant, ou d'une instance d'un livre) ;
- faire sortir un livre définitivement du catalogue ;
- ajouter un livre au bibliobus.

Le nombre maximal de livres que peut contenir le bibliobus est fixé. L'ajout d'un livre ne peut se faire que s'il reste de la place. Pour le bibliothécaire, chaque livre est repéré par son indice dans les rayons : on utilisera un tableau pour stocker les livres dans la bibliothèque. Cet indice sera l'identifiant du livre. Lors de la sortie définitive d'un livre, on pensera à *resserrer* les livres, par exemple en rangeant le dernier livre dans le *trou*. Les nouveaux livres seront ainsi ajoutés toujours en fin de tableau.

Comme on peut avoir plusieurs bibliobus sur une commune, chaque bibliobus possède un nom (qui ne peut pas changer).

Exercice 8 : classe *Bibliobus*, classe *DemoBibliobus*

Commencez par créer deux fichiers contenant respectivement une classe `Bibliobus` et une classe `DemoBibliobus`. Dans la classe `Bibliobus`, écrivez tous les attributs nécessaires. Dans la classe `DemoBibliobus` implémentez seulement une méthode `main`. **Après chaque écriture d'une nouvelle méthode ou d'un constructeur pour la classe `Bibliobus`, vous devrez tester votre implémentation dans la classe `DemoBibliobus`.**

Exercice 9 : Constructeurs

Par défaut, un bibliobus peut contenir 100 livres dans son catalogue. Mais il doit être possible de définir à la création du bibliobus la taille du catalogue. Le nom du bibliobus est toujours indiqué à sa création. Implémentez les constructeurs nécessaires.

Exercice 10 : Ajout d'un livre, première version

Implémentez la méthode `ajoutLivre(String titre, String auteur, String editeur)` qui permet d'ajouter un livre au catalogue, dans la seule limite des places disponibles.

Exercice 11 : *Afficher le catalogue*

Implémentez la méthode `afficheCatalogue()` qui affiche sur la console le nom du bibliobus, et toutes les informations sur son catalogue : identifiant de chaque livre et caractéristiques du livre.

Exercice 12 : *Accès aux caractéristiques d'un livre*

Implémentez les méthodes `getTitre(int identifiant)`, `getAuteur(int identifiant)`, `getEditeur(int identifiant)`, `getGenre(int identifiant)` et `getNbExemplaires(int identifiant)` qui retournent les caractéristiques d'un livre indiqué par son identifiant.

Exercice 13 : *Afficher les caractéristiques d'un livre*

Implémentez la méthode `afficheLivre(int identifiant)` qui affiche sur la console toutes les caractéristiques d'un livre indiqué par son identifiant.

Exercice 14 : *À vous !*

Implémentez toutes les autres méthodes du cahier des charges. À vous de déterminer leur signature !