Labo #1

Structures de données et analyse de complexité

On veut modéliser la gestion d'une bibliothèque : on définira un certain nombre de classes : GestionBibliotheque, Ouvrage, BiblioTab, Bibliotheque, Periodique, CD, Livre. Les livres auront comme propriétés : auteur, titre, éditeur ; les périodiques : nom, numéro, périodicité ; les CDs : titre, auteur. De plus tous les ouvrages auront une date d'emprunt (potentiellement nulle), une cote (le numéro (1, 2, 3 …) par ordre de création). On implémentera également sur chaque objet une méthode toString() renvoyant toutes les informations sur l'ouvrage sous forme d'une chaîne de caractères.

La classe **BiblioTab** permettra de stocker dans une structure les livres (ajout, recherche suppression, la suppression et recherche prenant en argument la cote de l'ouvrage). Elle aura également une méthode toString() affichant le nombre d'ouvrages, puis chaque ouvrage successivement. La classe Bibliotheque sera simplement une version abstraite déclarant les mêmes méthodes que BiblioTab mais sans les implémenter. BiblioTab héritera de Bibliotheque.

La classe GestionBibliotheque ne contiendra que la méthode main et testera la bibliothèque en y insérant et supprimant quelques ouvrages, quelques recherches, puis en affichant le contenu de la bibliothèque. Vous pouvez créer un menu pour la gestion et même un fichier texte pour les données.

À faire :

On mettra en évidence pour chaque classe les méthodes et les champs qu'elle définit, redéfinit ou hérite. On souhaite que tous les champs soient déclarés privés et que l'on puisse y accéder de l'extérieur que par des méthodes.

Partie 1

Implémentez les classes ci-dessus. Pour la classe **BiblioTab** on utilisera un tableau de longueur suffisante (exemple 20. On vérifiera quand même à chaque opération d'insertion que l'on ne dépasse pas les bornes du tableau).

Partie 2

Dans ce qui suit, on implémente une deuxième version de la bibliothèque, que l'on appellera **BiblioList** et qui héritera également de Bibliotheque. Cette nouvelle implémentation utilisera la classe **LinkedList** définie dans l'API Java standard.

Vous devrez ajouter au début du fichier BiblioList.java la commande import java.util.* ;

Partie 3

Dans ce qui suit, on implémente une troisième version de la bibliothèque, que l'on appellera **BiblioListPerso** et qui héritera également de Bibliotheque. Cette nouvelle implémentation utilisera la classe **ListeChainee** que vous allez créer pour implémenter une liste chaînée personnalisée.

Vous devez faire les tests pour les opérations insertion, recherche et suppression et aussi faire une analyse asymptotique (big O) des 3 parties concernant les opération insertion, recherche et suppression. Voir notes de cours comment faire.

REMISE

Le projet déposé dans LEA et l'analyse asymptotique en format **pdf** placée dans le dossier «documents» de votre projet. Lors de la remise placez l'url de votre vidéo des tests ainsi que le nom des membres de votre équipe.

Toute consigne non respectée vous fera perdre 10% de la note finale.