Exercice1 : - Ecrire une classe Personne contenant les champs nom, prenom et âge et qui implémente l'interface IComparable

-Ecrire une classe TestPersonne qui permet de définir un list de personne,Trier ce list par ordre alphabétique de nom ,afficher les éléments de ce list ,rechercher un élément dans ce list .

Exercice2 : - Ecrire une classe Personne contenant les champs nom, prenom et âge et qui implémente l'interface IComparer

-Ecrire une classe TestPersonne qui permet de définir un list de personne,Trier ce list par ordre alphabétique de nom et prénom, ou à partir de personne plus jeune ,le choix de critère de tri est choisi par l'utilisateur. afficher les éléments de ce vecteur,rechercher un élément dans ce list et extraire la personne la plus jeune.

Exercice3 : - Ecrire une classe Personne contenant les champs nom, prenom et âge et qui implémente l'interface Comparable. -Ecrire une classe TestPersonne3 qui permet de définir un tableau de personne puis effectuer les opérations de tri et de recherche en utilisant la classe Arrays.

Exercice4 : On part de deux ensembles d'entiers différents. Le but est d'obtenir un ensemble dont les éléments ne font partie que de l'un des ensembles de départ, mais pas des deux : Exemple :Si au départ on a : {1,5,7,8,9,3,6} et {2,0,3,8,9,7} On devra obtenir pour résultat l'ensemble :{1,5,6,0,2}

1.Ecrire une classe Ensemble permettant de gérer un ensemble d'entiers avec un constructeur qui prend en paramètre une collection de type list et les méthodes suivantes :

-**void ajouter(int i) :** pour ajouter un entier i à l'ensemble.

-**boolean exist(int i,Ensemble e):** pour tester si un élément existe dans un ensemble ou non.

-**void supprimer(int i) :** pour supprimer de l'ensemble l'élément situé à l'index i.si l'index n'existe pas ,cette méthode traite une exception de type ArrayIndexOutOfBoundsException

2. Ajouter à la classe Ensemble la méthode

-**Ensemble getOr(Ensemble e1,Ensemble e2)** retournant le ou exclusif de deux ensembles.

3. Ecrire la classe TestEnsemble permettant de tester l'ensemble.

Exercice5 :

L'objectif de cet exercice est d'écrire une classe **Annuaire** pour mémoriser des numéros de téléphone et d'adresses.Chaque entrée est représentée par une fiche à plusieurs champs de type String :un nom,un numéro de téléphone et une adresse.La structure des fiches est décrite par une classe **Fiche**,de plus la classe Annuaire à utiliser comporte une table associative dictionnary < String, Fiche> qui sera faites d'association (un nom,une fiche).

1. Construire la classe Fiche,faites simple,ce n'est qu'une classe auxiliaire « constructeur,les accesseurs,toString).

2. Construire la classe Annuaire en ajoutant son propre champ(dictionnary),son constructeur ainsi que les fonctions suivantes :

-**void rechercherFicheParNom(String nom)** : cette fonction permet de rechercher et d'afficher la fiche concernant le nom indiqué s'il existe sinon afficher un message montrant que cette fiche n'existe pas.

-**void ajouterFicheParNom(String nom)** :cette fonction permet d'ajouter un nouveau enregistrement dans l'annuaire si la clé donnée comme argument à la fonction n'existe pas.Si la clé existe, on demande à l'utilisateur s'il veut modifier ou non les valeurs de la fiche de la clé correspondante.

-**String toString( ) :**retourne une chaine représentant tous les fiches de l'annuaire.