Le directeur pédagogique de l'ENSA de **Tanger** souhaite réaliser une application .Net de gestion des ressources humaines, s'occupant à la fois des étudiants et du personnel. Les étudiants ont chacun un niveau et une moyenne annuelle. Tout membre du personnel a un bureau. Dans le personnel, on distingue **le directeur de l'école**, le **personnel administratif** et le **personnel enseignant.** Chaque personne a un code, un nom et un prénom. Tout membre du personnel reçoit un **salaire** à la fin du mois, cependant le **directeur,** en plus de son salaire, reçoit des primes de déplacements et les **enseignants** reçoivent, en plus de leur salaire et de leurs primes de déplacement, une somme pour les heures supplémentaires accomplies. La classe **enseignant** possède, hormis des attributs hérités de la classe **Personnel**, un attribut **Grade**, **volume horaire** et une **liste** de ses **groupes** (chaque groupe d'étudiants est stocké dans une liste et les groupes d'un enseignant doivent être stockés dans un **Dictionnaire** dont les clefs sont les noms des groupes et les valeurs sont des listes). Les éléments du **dictionnaire *<clés et valeurs>*** sont des attributs privés de la classe **Groupe**. Les volumes horaires sont exprimés en heures entières et le prix d'une heure sera déterminé en fonction du grade : PA: **300 DH** PH : **350 DH** PES : **400 DH**

Attention, la classe **Personnel** ne correspond en réalité à aucun objet existant, elle ne fait que rassembler les caractéristiques communes à tous les objets réellement manipulés par l'application, qui seront des instances des classes **Directeur**, **Enseignant** et **Administratif** que vous allez définir par la suite. Une interface nommée **IRessourcesHumaines** intégrant les méthodes suivantes :

- **Afficher_Enseignants()** : permet de parcourir la liste ***GRH*** et d'afficher que les **enseignants**.(les groupes doivent être affichés séparément si l'enseignant enseigne plusieurs groupes).
- **Rechercher_Ens(?) :** permet de chercher un enseignant en utilisant son code (elle retourne sa position dans la liste s'il existe ou -1 sinon). Une classe nommée **RessourcesHumaines** et implémente l'interface **IRessourcesHumaines** permettra de gérer les ressources humaines de l'école.

On vous propose d'ajouter aussi les méthodes suivantes à la classe convenable.

- **Ajouter_etudiant(?)** : permet d'affecter un étudiant à un groupe.
- **Ajouter_groupe(?)** : permet d'affecter un groupe à un enseignant.
- **Afficher_grp()** : permet d'afficher les étudiants d'un groupe.
- **Afficher_etd()** : permet d'afficher un étudiant.
- **Afficher_ens** : permet d'afficher un enseignant et ses groupes d'étudiants.

Contraintes :

- ✓ La méthode **Calculer_Salaire(?)** de la classe **Personnel** n'a aucune définition, elle sera définie effectivement dans les sous-classes.
- ✓ La classe **Directeur** est une classe **Singleton** qui ne devra être instanciée qu'une seule fois. *(pour une deuxième tentative d'instanciation un message d'erreur devra s'afficher***).**
- ✓ Utilisez des indexeurs pour accéder aux groupes d'étudiants d'un enseignant.

1) Donnez une conception en classes de l'énoncé précédent.
2) Développez toutes les classes
3) Testez les différentes méthodes.