**Travaux pratique : Notions de base pour EF**

Nous voulons implémenter un système d’information pour la gestion des professeurs, ainsi que les étudiants. Pour cette raison nous allons créer 2 classes : Professeur et Etudiant.

Nous considérons qu’un étudiant n’as qu’un seul professeur, et qu’un professeur peut enseigner plusieurs étudiants.

La classe professeur est composée d’un Id, un nom, un prénom, une spécialité et une liste d’étudiants.

La classe Etudiant est composée d’un Id, un nom, un prénom, un genre (Femme/Homme) et un Tel.

1. Créer la classe Etudiant, avec respect des principes d’encapsulation et un constructeur d’initialisation.
2. Redéfinir la fonction **ToString()** pour afficher le nom et le prénom de l’étudiant séparés par un tiret.
3. Créer la classe Professeur, avec respect des principes d’encapsulation et un constructeur d’initialisation dans le quel vous initialiser la liste par une liste vide.
4. Créer la fonction **Ajouter()**, pour ajouter un étudiant à la liste des étudiants du professeur.
5. Créer la fonction **Supprimer(int id)**, pour supprimer l’étudiant dont l’Id est égale à la valeur passée en paramètres.
6. Créer une fonction **FindByGenre(string genre)**, pour retourner la liste des étudiants dont le genre est égale au genre passé en paramètres, utilisez le mot clé yield. Si le genre est vide, retourner la liste complète.
7. Surcharger la fonction **ToString(string genre)** pour afficher le nom, le prénom et la spécialité du professeur séparés par un tiret, ainsi que la liste de tous ces étudiants ayant le même genre passé en paramètre. Nous voulons garantir un couplage faible entre les 2 fonctions.
8. Créer une fonction **Vider()** d’extension de classe **Professor** permettant de vider la liste des étudiants d’un professeur.
9. Créer une fonction d’extension **HaveFix()** de classe **Professor** permettant de retourner la liste des étudiants ayant un Fix.
10. Créer une fonction d’extension **Show()** de classe **List<Student>** permettant d’afficher la liste des étudiants.
11. Dans la fonction Main dans Programs.cs, vous allez :
    1. Déclarer une liste vide de professeurs.
    2. Ajouter 3 professeurs avec 3 étudiants pour chacun.
    3. Afficher les 3 professeurs sans précision du genre.
    4. Afficher les 3 professeurs avec précision du genre.
    5. Supprimer un étudiant de votre choix.
    6. Afficher les 3 professeurs sans précision du genre.
    7. Vider la liste des étudiants pour un professeurs de votre choix.
    8. Afficher les 3 professeurs sans précision du genre.
    9. Tester les fonctions HaveFix() et Show().
12. Le client, à décider de recruter un développeur pour assurer les tâches de maintenance évolutive dans la classe Program (il n’as pas le droit de toucher votre code), il vous demande de donner à son développeur, la possibilité de chercher les étudiants avec n’importe quel critère (rappelez-vous que vous avez déjà créer une fonction **HaveFix()**, mais la condition est constante ). Faites le nécessaire en créant une nouvelle fonction **Where()**, prenant en paramètre un predicate pour assurer la liberté de choix de condition.
13. Testez votre nouvelle fonction dans la classe Program.
14. Faîtes appel à toutes vos fonctions de teste, sans utiliser leurs nom Directe : Utilisez Delegate, Action ou Func.
15. Vous êtes libre de discuter et de proposer des améliorations ou bien des extensions de projet.