

GESTION D'UN ÉTABLISSEMENT SCOLAIRE

Dans notre établissement scolaire, nous souhaitons enregistrer les données des enseignants, des étudiants et des cours dans des listes qui respectent les consignes suivantes :

- Dans la liste des cours, il faut s'assurer qu'il n'existe pas de doublons (deux cours ayant le même code cours);
- Chaque enseignant sera définie par une liste de cours (au lieu d'un tableau de cours);
- La liste des enseignants doit être triée par leurs matricules;
- Dans la liste des étudiants, il faut s'assurer qu'il n'existe pas de doublons (deux étudiants ayant le même code étudiant);

Pour créer ces listes, nous allons s'appuyer sur les classes [SortedList](#) et [Hashtable](#).

Une classe appelée [ListeCours](#) qui est une [Hashtable](#), contient :

- 1 méthode [void](#) AjouterCours([Cours](#) cours) qui ajoute un nouveau cours à la liste des cours;
- 1 méthode [void](#) AjouterCours([Cours](#)[] cours) qui ajoute un tableau de nouveaux cours à la liste des cours;
- 1 méthode [Cours](#) RechercherParCode([string](#) code) qui retourne le cours dont le code est passé en argument de la méthode;
- 1 méthode [Cours](#) RechercherParIntitulé([string](#) intitulé) qui retourne le cours dont l'intitulé est passé en argument de la méthode;
- 1 méthode [void](#) SupprimerParCode([string](#) code) qui supprime le cours dont le code est passé en argument de la méthode;
- 1 méthode [void](#) SupprimerParIntitulé([string](#) intitulé) qui supprime le cours dont l'intitulé est passé en argument de la méthode;
- 1 méthode [int](#) getNombreCours() qui retourne le nombre de cours dans la liste;
- 1 méthode ToString().

Modifier la classe [Enseignant](#), et ajouter les méthodes :

- 1 méthode `void AffecterCours(Cours cours)` qui affecte un nouveau cours à un enseignant;
- 1 méthode `void AffecterCours(Cours[] cours)` qui affecte un ensemble de nouveaux cours à un enseignant;
- 1 méthode `void EnleverCours(string code)` qui annule l'affectation d'un cours à un enseignant;

Une classe appelée `ListeEnseignants`, qui est une liste triée, contient :

- 1 méthode `void AjouterEnseignant(Enseignant enseignant)` qui ajoute un nouveau enseignant à la liste des enseignants;
- 1 méthode `void AjouterEnseignants(Enseignant[] enseignants)` qui ajoute un tableau de nouveaux enseignants à la liste des enseignants;
- 1 méthode `Enseignant RechercherParMatricule(string matricule)` qui retourne l'enseignant dont le matricule est passé en argument de la méthode;
- 1 méthode `ListeEnseignants RechercherParCours(string nomCours)` qui retourne une liste triée des enseignants dont le nom du cours qu'ils enseignent est passé en argument de la méthode;
- 1 méthode `ListeEnseignants RechercherParCours(Cours cours)` qui retourne une liste triée des enseignants dont le cours qu'ils enseignent est passé en argument de la méthode;
- 1 méthode `void SupprimerParMatricule(string matricule)` qui supprime l'enseignant dont le matricule est passé en argument de la méthode;
- 1 méthode `int getNombreEnseignants()` qui retourne le nombre d'enseignants dans la liste;
- 1 méthode `SortedList<DateTime, Enseignant> Retraite()` qui retourne une liste triée par date de naissance, des enseignants qui vont avoir leur retraite cette année;
- 1 méthode `ToString()`.

Une classe appelée `ListeEtudiants`, qui est une `Hashtable`, contient :

- 1 méthode `void AjouterEtudiant(Etudiant etudiant)` qui ajoute un nouveau étudiant à la liste des étudiants;

- 1 méthode `void` `AjouterEtudiants(Etudiant[] étudiants)` qui ajoute un tableau de nouveaux étudiants à la liste des étudiants;
- 1 méthode `Etudiant` `RechercherParCode(string code)` qui retourne l'étudiant dont le code est passé en argument de la méthode;
- 1 méthode `ListeEtudiants` `RechercherParCours(string nomCours)` qui retourne une liste triée des étudiants dont le nom du cours qu'ils suivent est passé en argument de la méthode;
- 1 méthode `ListeEnseignants` `RechercherParCours(Cours cours)` qui retourne une liste triée des étudiants dont le nom du cours qu'ils suivent est passé en argument de la méthode;
- 1 méthode `void` `SupprimerParCode(string code)` qui supprime l'étudiant dont le code est passé en argument de la méthode;
- 1 méthode `int` `getNombreEtudiants()` qui retourne le nombre d'étudiants dans la liste;
- 1 méthode `Etudiant` `getPremier()` qui retourne l'étudiant le mieux classé;
- 1 méthode `List<Etudiant>` `Réussi()` qui retourne la liste des étudiants qui sont au-dessus de la moyenne;
- 1 méthode `List<Etudiant>` `Classement()` qui retourne une liste triée des étudiants par moyenne dans l'ordre décroissant (on impose le tri par une `SortedList<double, Etudiant>` dans un premier lieu, et puis une inversion);
- 1 méthode `ToString()`.

Une classe appelée `Classe`, contient :

- 4 variables d'instance privées : `nomClasse`, `listeCours`, `listeEnseignantsClasse` et `listeEtudiantsClasse`;
- 4 accesseurs.
- 1 méthode `void` `AjouterEtudiant(Etudiant étudiant)` qui ajoute un nouveau étudiant à la liste des étudiants et lui affecte la liste de cours de la classe;
- 1 méthode `void` `AjouterEtudiants(Etudiant[] étudiants)` qui ajoute un tableau de nouveaux étudiants à la liste des étudiants et leur affecte la liste des cours de la classe;

- 1 méthode `void AjouterEnseignant(Enseignant enseignant)` qui ajoute un nouveau enseignant à la liste des enseignants, si l'un des cours qu'il enseigne existe dans la liste des cours de la classe, et qu'aucun enseignant ne prend en charge ce cours;
- 1 méthode `void AjouterEnseignants(Enseignant[] enseignants)` qui ajoute un tableau de nouveaux enseignants à la liste des enseignants, si l'un des cours qu'ils enseignent existe dans la liste des cours de la classe, et qu'aucun enseignant ne prend en charge ce cours;

Une classe appelée `Institut`, qui est une liste de classes contient :

- 1 variable d'instance privée : `nomInstitut` et `listeClasses`;
- 2 accesseurs;
- 1 méthode `ListeEtudiants EtudiantsParClasse(string nomClasse)` qui retourne la liste des étudiants d'une classe;
- 1 méthode `ListeEnseignants EnseignantsParClasse(string nomClasse)` qui retourne la liste triée des enseignants d'une classe;
- 1 méthode `ListeEnseignants EnseignantsParClasse(string nomClasse)` qui retourne la liste triée des enseignants d'une classe;
- 1 méthode `ListeEnseignants Enseignants()` qui retourne la liste des enseignants de l'institut;
- 1 méthode `int getNombreEtudiants()` qui retourne le nombre d'étudiants dans l'institut;
- 1 méthode `double TauxRéussite()` qui retourne le taux de réussite des étudiants de l'institut;
- 1 méthode `List<Etudiant> getPremiers()` qui retourne les 3 premiers étudiants de l'institut;

Travail à réaliser :

- Écrire les quatre classes `ListeCours`, `ListeEnseignants`, `ListeEtudiants`, `Classe` et `Institut`.
- Créer un jeu de tests.