Dictionnaire

Exercice 1

Créer un dictionnaire qui contient les jours de la semaine en français et en anglais.

Fait un programme qui demande a l'utilisateur de saisir un jour de la semaine en français et qui affiche le jour en anglais.
Si l'utilisateur saisit un jour qui n'existe pas, affiche un message d'erreur.

Exercice 2

Créer une liste de 10 élèves (avec un dictionnaire) avec leur nom et leur note.
Fait la moyenne de la classe.
Puis afficher le prénom et la note de l'élève qui a la meilleure note.
Fait de meme pour l'élève qui a la pire note.

Exercice 3

Recreer une classe Contacts qui contient un dictionnaire de contacts.
Contact sera un dictionnaire qui contient des numeros de telephone et des noms.
Le numeros de telephone est une string de 10 chiffres.
Le nom est une string.

Faite une methode `addContact` qui permet d'ajouter un contact avec son nom et son numero de telephone.

Exercice 3

Faite une methode `deleteContact` qui permet de supprimer un contact.
Elle peut prendre en parametre le nom ou le numero de telephone ou les deux.

Faite une methode `changeContact` qui permet de modifier un contact.
Elle prendra en parametre un nom, et son nouveau numero de telephone. (verifier que le contact existe)

Exercice 3

Faite une methode `GetContact` qui permet de rechercher un contact par son nom.
Elle retournera le numero de telephone.

Faite une methode `FindContact` qui permet de rechercher un contact par son numero de telephone.
Elle retournera le nom du contact.

Exercice 3.1

Faite une methode `GetAllContacts` qui retourne tous les contacts dans un tableau de string. Sous la forme :

```
string[] contacts = GetAllContacts();
foreach(string contact in contacts)
{
    Console.WriteLine(contact); // Nom : " + nom + " , numero : " + numero
}
```

Exercice 3.2

Faire un main qui permet de tester toutes les methodes.

Exercice 4 Liste de Cours

Ecrire une classe Cours.
Elle aura comme attributs :

- 📌 un dictionnaire priceProduct qui contient les produits et leurs prix.
- 📌 une lists lstProducts qui contient les produits achetés.

Exercice 4 Liste de Cours

Ecrire un constructeur par default qui initialise les deux attributs a vide.

Ecrire une methode `PriceProductAleatory` qui prendra en parametre un string et qui creera un prix aleatoire entre 1 et 18.99 et ajoutera ce produit a priceProduct s'il n'existe pas.

Exercice 4.1

Ecrire une methode `addProduit` qui permet d'ajouter un produit a la liste des produits achetés.

Ecrire une methode `addMultipleProduct` qui permet d'ajouter plusieurs produits a la liste des produits achetés. (Elle prendra un tableau de string en parametre).

Exercice 4.2

Ecrire une methode `removeProduct` qui permet de supprimer un produit de la liste des produits achetés.

Ecrire une methode `removeMultipleProduct` qui permet de supprimer plusieurs produits de la liste des produits achetés. (Elle prendra un tableau de string en parametre).

Exercice 4.3

Creer une methode qui permet de definir le prix d'un produit.
Ainsi que la methode qui permet de definir le prix de plusieurs produits. (Elle prendra un tableau de string en parametre, un tableau de int).
(attention au doublons des produits, si un produit a un prix definie, il ne doit pas etre modifie dans la methode)

Exercice 4.4

Ecrire une methode `GetPriceLstProducts` qui retourne le prix total de la liste des produits achetés.
Si un produit n'a pas de prix, il en aura un aleatoire par votre premiere methode.