Exercice 1 : Gestion des clients

1. Créer un projet et Ajouter la classe Client au projet :

- **Nom de la classe :** Client
- **Liste des attributs :**
    - Code client
    - Nom de client
    - Adresse

2. Ajouter des accesseurs aux champs de la classe client.
3. Ajouter le constructeurs suivant :
    - Constructeur d'initialisation pour initialiser tous les champs de la classe.
4. Ajouter les méthodes suivantes :

- **ToString** : retourne une chaine qui contient tous les informations sur le produit.
- **Equals** : méthode booléenne, accepte en paramètre un objet de client. Deux clients sont égaux s'il possède le même code et le même nom

5. Dans le main

- Déclarer une liste des clients.
- Proposer aux utilisateurs le menu suivant :
    - Ajouter Client
    - Afficher tous les clients
    - Supprimer Client par son nom
    - Rechercher un client par son nom
    - Modifier l'adresse d'un client par son nom.
    - Quitter le programme.

Exercice 2 : Gestion des produits

1. Ajouter la classe Fournisseur au projet :
    a. **Nom de la classe** : Fournisseur
    b. **Liste des attributs :**
        i. Code Fournisseur
        ii. Nom de fournisseur
        iii. Adresse
    c. Ajouter les accesseurs aux attributs de la classe
    d. Ajouter le constructeurs suivant :
        i. Constructeur d'initialisation pour initialiser tous les champs de la classe.
    e. Ajouter les méthodes suivantes :
        i. **ToString** : retourne une chaine qui contient tous les informations sur le produit.
        ii. **Equals** : méthode booléenne, accepte en paramètre un objet de fournisseur. Deux fournisseur sont égaux s'il possède le même code et le même nom.
2. Ajouter la classe Produit au projet :
    a. **Nom de la classe :** Produit
    b. **Liste des attributs ou champs :**
        i. Désignation de type chaine de caractères
        ii. Prix de type double
        iii. Fournisseur
    c. **Liste des accesseurs et des modificateurs (Les propriétés) :** Pour chaque attribut de la classe Produit, créer des propriétés de lecture et d'écriture.
    d. **Ajouter les constructeurs :**

i. Constructeur qui permet l'initialisation de tous les champs de la classe Prodtuit.

3. **Ajouter les méthodes :**
   a. **ToString** : retourne une chaine qui contient tous les informations sur le produit.
   b. **Equals** : méthode booléenne, accepte en paramètre un objet de produit. Deux produit sont égaux s'il possède la même désignation.
4. Dans le programme principal, gérer un ensemble de produit en proposant le menu suivant :
   - Liste des produits
   - Ajouter produit
   - Supprimer Produit
   - Trouver un Produit par son nom
   - Quitter le programme

Exercice 3 : Gestion de bibliothèque

1. Ajouter la classe Auteur au projet :

a. **Nom de la classe :** Auteur

b. Liste des attributs ou champs :

i. Nom de type chaine de caractères

ii. Nationalité de type chaine de caractères

c. **Liste des accesseurs et des modificateurs (Les propriétés) :** Pour chaque attribut de la classe Auteur créer des propriétés de lecture et d'écriture.

d. **Constructeurs :**

i. Constructeur qui permet l'initialisation de tous les champs de la classe Auteur.

e. **Méthodes :**

   i. AfficheAuteur() : Méthode qui ne retourne aucune valeur et qui permet d'afficher les informations d'un auteur.

2. Ajouter la classe Livre au projet :

a. **Nom de la classe :** Livre

b. Liste des attributs ou champs :

   i. Nom de type chaine de caractères

   ii. Editor de type chaine de caractères

   iii. Nombre de pages de type entier

   iv. Liste des auteurs

c. **Liste des accesseurs et des modificateurs (Les propriétés) :** Pour chaque attribut de la classe Livre créer des propriétés de lecture et d'écriture.

d. **Constructeurs :**

   i. Constructeur qui permet l'initialisation de tous les champs de la classe Livre.

e. **Méthodes :**

   i. **AfficheAuteursLivres()** : Méthode qui ne retourne aucune valeur et qui permet d'afficher les auteurs d'un livre.

   ii. **AjouterAuteur(Auteur A)** : Méthode qui ne retourne aucune valeur et qui permet d'ajouter l'auteur A à la liste des auteurs.

   iii. **AjouterAuteur()** : Méthode qui ne retourne aucune valeur et qui permet d'ajouter un auteur à la liste des auteurs.

   iv. **Supprimer(Auteur A)** : Méthode qui ne retourne aucune valeur et qui permet de supprimer l'auteur A de la liste des auteurs.

   v. **Supprimer(string NomAuteur)** : méthode qui ne retourne aucune valeur et qui permet de supprimer l'auteur avec le nom passer en paramétre.

vi. **AfficheLivre()** : Méthode qui ne retourne aucune valeur et qui permet d'afficher tous les informations d'un livre (même les auteurs).

3. Ajouter la classe Biblio au projet :

a. **Nom de la classe :** Biblio

b. Liste des attributs ou champs :

i. Nom de type chaine de caractères.

ii. adresse de type chaine de caractères.

iii. Liste des Livres.

c. **Liste des accesseurs et des modificateurs (Les propriétés) :** Pour chaque attribut de la classe Livre créer des propriétés de lecture et d'écriture.

d. **Constructeurs :**

i. Constructeur qui permet l'initialisation de tous les champs de la classe Biblio.

e. **Méthodes :**

i. **AfficheLivreBiblio()** : Méthode qui ne retourne aucune valeur et qui permet d'afficher les livres d'une bibliothèque.

ii. **AjouterLivre(Livre A)** : Méthode qui ne retourne aucune valeur et qui permet d'ajouter le livre A à la liste des livres.

iii. **AjouterLivre()** : Méthode qui ne retourne aucune valeur et qui permet d'ajouter un livre à la liste des livres.

iv. **SupprimerLivre(Livre A)** : Méthode qui ne retourne aucune valeur et qui permet de supprimer le livre A de la liste des livres.

v. **SupprimerLivre(string nomLivre)** : Méthode qui ne retourne aucune valeur, et qui permet de supprimer le nom de livre passer en paramètre.

vi. **AfficheBiblio()** : Méthode qui ne retourne aucune valeur et qui permet d'afficher tous les informations d'une bibliothèque (même les livres).

5- Tester dans le main tester les classes crées