Apprendre le C# - LINQ

Introduction rapide

LINQ (**L**anguage **In**tegrated **Q**uery) définit une façon propre en C# d'exprimer une requête proche de la façon SQL à partir d'un ensemble.
Cela permet par exemple de récupérer :

- Un sous-ensemble d'éléments
- Une séquence d'éléments
- Une valeur singleton sur les données sources
    - Nombre d'élément
    - Elément le plus grand/petit
    - Premier élément d'une condition

Exercices

Dans votre dépot git créez un nouveau dossier et initialisez votre projet sur Visual Code

L'ensemble des exercices seront directement développés dans notre main.

Exercice 1 : Requêtes simples

A partir de cette liste numérique :

```
List<int> entiers = new List<int> {4,5,2,3,1,1,0,5,8,9,10,15,16,20,21,4,5 };
```

1. Récupérer et afficher tous les nombres supérieurs à 5
2. Récupérer et afficher les nombres supérieurs ou égaux à 15 et inférieurs à 20
3. Récupérer et afficher les nombres supérieurs à 2, qui sont des multiples de 2, inférieurs à 20, différents de 8

Maintenant à partir de cette liste string :

```
List<string> fruits = new List<string> { "Banane", "Ananas", "Cerise", "Framboise", "Groseilles", "Pomme", "Poire", "Tomate", "Kiwi", "Raisin", "Mangue", "Datte"};
```

4. Récupérer et afficher tous les fruits dont le nom contient plus de 5 lettres
5. Récupérer et afficher tous les fruits dont le nom commence par "P", dont la longueur du nom est supérieure à 4, qui contiennent un "o" mais pas de "m"
6. Trier et afficher les fruits selon leur longueur

Exercice 2 : Requêtes sur les objets

A partir de cette liste d'objets :

```
List<Dog> dogs = new List<Dog>
{
    tabnew Dog("Berger Australien", "Banzaï", 1, 28),
    new Dog("Berger Australien", "Letto", 3, 30),
    new Dog("Berger Australien", "Princesse", 8, 32),
    new Dog("Berger Allemand", "Floyd", 10, 32),
    new Dog("Caniche", "Igor", 13, 9),
    new Dog("Labrador", "Swing", 15, 25),
    new Dog("Teckel", "Wonki", 2, 5),
    new Dog("Terre Neuve", "Albator", 1, 50),
    new Dog("Carlin", "Pataud", 13, 10),
    new Dog("Boxer", "Frank", 6, 25),
    new Dog("Lévrier Afghan", "Précieuse", 9, 26),
    new Dog("Yorkshire", "Kakou", 3, 6)
};
```

1. Récupérer et afficher tous les chiens qui sont de la race "Berger Australien
2. Récupérer et afficher tous les chiens qui sont de la race "Berger Australien" et les trier par leur nom
3. Récupérer et afficher tous les chiens âgés de 5 ans et plus, dont la longueur du nom est supérieure à 5 lettres
    - Les trier par leur poids
4. Récupérer et afficher tous les chiens par leur âge (tri décroissant) puis leur poids (tri croissant)
5. Récupérer et afficher tous les noms de chien dont le nom de race tient en un seul mot, leur poids doit être supérieur à 15 kilos, leur nom doit contenir un "i" et les trier par la longueur de leur prénom

Exercice 2 : Requêtes créant de nouveaux objets

A partir de cette liste d'objets :

```
List<Personne> personnes = new List<Personne>
{
    new Personne("Hallyday", "Johnny", false),
    new Personne("Vartan", "Sylvie", false),
    new Personne("Drucker", "Michel", false),
    new Personne("Antoine", "Antoine", true),
    new Personne("Philippe", "Edouard", false),
    new Personne("Demorand", "Patricia", true),
    new Personne("Ulysse", "Margareth", true),
    new Personne("Zenith", "Méryl", true),
    new Personne("Bobo", "Jojo", false)
};
```

1. Créer un itérable d'ingénieurs, trier par nom, et ensuite par prénom

2. Récupérer et afficher la liste des personnes qui ne sont pas ingénieures
3. Créer une liste d'objets anonymes (Ingénieurs + techniciens)

Exercice 4 : Requêtes et variables

A partir de cette liste d'objets :

```
List<Personne> personnes = new List<Personne>
{
    new Personne("Beauvoir", "Simon", 16, "M"),
    new Personne("Beauvoir", "Simone", 25, "F"),
    new Personne("De Caunes", "Richard", 41, "M"),
    new Personne("Sullivan", "Sullivan", 31, "M"),
    new Personne("Rémy", "Camille", 22, "F"),
    new Personne("Manchon", "Camille", 19, "M"),
    new Personne("Thiebaud", "Marie", 61, "F"),
    new Personne("Crouchon", "Mélanie", 55, "F"),
    new Personne("Baline", "Mélodie", 74, "F"),
    new Personne("Karine", "Pascal", 31, "M"),
    new Personne("Katherine", "Pascale", 36, "F"),
    new Personne("Zoula", "Charles", 20, "M"),
    new Personne("Romain", "Collin", 34, "M"),
    new Personne("Fouchard", "Aïcha", 48, "F"),
    new Personne("Blandine", "Maëva", 18, "F")
};
```

1. Créer une variable nom_complet = Nom +" "+ Prenom et la mettre comme seule attribut de l'objet créé dans le select et les afficher
2. Pour les personnes majeures ayant moins de 50 ans :
    - Créer une variable "initiale" qui contient seulement les initiales du nom et du prénom : `p.Nom[0]+"."+p.Prenom[0]`
    - Créer une variable taille_nom_complet = longueur du prénom + longueur du nom
    - Créer un objet anonyme avec les attributs : Nom, prénom, initiale, taille_nom_complet, age
    - Et les afficher

Exercice 5 : Requêtes et listes à 2 dimensions

A partir de cette liste à 2 dimensions d'objets :

```
List<List<Personne>> personnes = new List<List<Personne>>
{
    new List() {new Personne("Drucker", "Michel"),
                            new Personne("Bedia", "Ramzy"),
                            new Personne("Judor", "Eric")},

    new List() {new Personne("Diaz", "Cameron"),
```

```
                            new Personne("Depardieu", "Gerard"),
                            new Personne("Stallone", "Sylvester"),
                            new Personne("Macron", "Emmanuel")},

    new List() {new Personne("Benzema", "Karim"),
                            new Personne("Antoine", "Eric"),
                            new Personne("Ruiz", "Olivia"),
                            new Personne("Clavier", "Christian"),
                            new Personne("Einstein", "Albert")}
};
```

1. Récupérer et afficher tous les noms dont la longueur est supérieure à 5
2. Récupérer toutes personnes dont le nom contient un "e" et dont le prénom contient un "a"
    - Trier les personnes par leur nom (décroissant)
    - Leur réer un objet anonyme avec un attribut identite = prénom+" "+nom
    - Et afficher tous les "identite"
3. Récupérer toutes les listes qui contiennent plus de 4 personnes
    - En extraire les personnes dont le nom commence par "A", "B", ou "C"
    - Trier les personnes par leur prénom (croissant)
    - Leur créer un objet anonyme avec l'attribut "initiale" = 1ère lettre du prénom+"."+1ère lettre du nom
    - Et afficher tous les "initiale"
4. Récupérer toutes les listes qui contiennent moins de 5 personnes et afficher toutes les personnes comme ceci : Nom+" "+Prenom

Exercice 6 : Requêtes et group by

A partir de la liste d'objets Personne() :

```
List<Personne> personnes = new List<Personne>()
{
    new Personne("Garett", "Ramzy", 45, "M"),
    new Personne("Caire", "Joe", 35, "M"),
    new Personne("Clay", "Alicia", 18, "F"),
    new Personne("Bavette", "Simone", 68, "F"),
    new Personne("Henry", "Thierry", 44, "M"),
    new Personne("Jacquesonne", "Janett", 25, "F"),
    new Personne("Buveur", "Joe", 25, "M"),
    new Personne("Louet", "Karim", 31, "M"),
    new Personne("Louette", "Karima", 31, "F"),
    new Personne("Caire", "Paul", 19, "M"),
    new Personne("Mille", "Camille", 20, "F"),
    new Personne("Cent", "Camille", 40, "F"),
    new Personne("Million", "Camille", 60, "M"),
    new Personne("Gold", "Roger", 17, "M"),
```

```
    new Personne("Lion", "Sandra", 8, "F"),
    new Personne("René", "Jean", 6, "M")
};
```

1. Faire un group by sur le genre (sexe) des personnes présentes dans la liste d'objets Personne()
2. Faire un group by sur l'âge des personnes les trier par âge croissant
3. Faire un group by sur le prénom des personnes, et afficher les noms de famille par prénom.
    - Récupérer les personnes majeures (18 ans et plus)
    - Les trier par leur prénom en ordre décroissant
    - Et les afficher

A partir de la liste d'objets Personne() :

`List<int> nombres = new List<int>() { 1, 2, 3, 4, 5, 6, 7, 8, 9, 20, 11, 13, 12, 14, 18, 17, 16, 14, 14 };`

4. Grouper les éléments d'une liste de nombres. D'un côté les chiffres/nombres pairs, de l'autre ceux impairs
5. Grouper les individus par la première lettre de leur nom et faire un tri croissant sur l'attribut Nom de la classe Personne

Exercice 7 : Requêtes et group by multiple

```
List<Chien> chiens = new List<Chien>()
{
    new Chien("Gnocci", "Gnoc Gnoc", "Labrador", "Sable", "M", 1, 20),
    new Chien("Vagabond", "Gros Loup", "Labrador", "Noir", "M", 8, 25),
    new Chien("Milou", "Milos", "Labrador", "Sable", "M", 10, 24),
    new Chien("Sirène", "Sissy", "Labrador", "Sable","F", 4, 19),
    new Chien("Félicia", "Felicci", "Labrador", "Sable", "F", 6, 20),
    new Chien("Kratos", "Mon tueur", "Chihuahua", "Fauve", "M", 1, 2),
    new Chien("Jack", "Jaja", "Chihuahua", "Fauve", "M", 1, 2),
    new Chien("Mojave", "Mojojo", "Chihuahua", "Fauve", "M", 1, 2),
    new Chien("Hercule", "Herc", "Chihuahua", "Beige", "M", 35, 2),
    new Chien("Médusa", "Med", "Terre-Neuve", "Noire", "F", 6, 40),
    new Chien("Mélusine", "Mel", "Terre-Neuve", "Noire", "F", 7, 41),
    new Chien("Venus", "Violette", "Terre-Neuve", "Noire", "F", 8, 38),
    new Chien("Letto", "Lele", "Berger Australien", "Bleu Merle", "M", 3, 30),
    new Chien("Cabron", "Dum dum", "Berger Australien", "Bleu Merle", "M", 9, 31),
    new Chien("Banzaï", "Zaïzaï", "Berger Australien", "Noir et blanc", "M", 1, 28 ),
    new Chien("Haricot", "Harry", "Berger Australien", "Noir et blanc", "M", 2, 27),
    new Chien("Gédéon", "Gégé", "Berger Allemand", "Noir et feu", "M", 9, 31),
    new Chien("Bella", "Belbel", "Berger Allemand", "Noir et feu", "F", 5, 28),
    new Chien("Oui-oui", "oui", "Berger Allemand", "Sable", "M", 7, 35),
    new Chien("Pataud", "Patoche", "Carlin", "Sable", "M", 16, 8),
    new Chien("Killer", "Kiki", "Carlin", "Sable", "M", 10, 8),
```

```
    new Chien("Frank", "Colonel", "Carlin", "Noir", "M", 9, 9)
};
```

1. Faire un group by multiple sur la race et la couleur et trier par ordre croissant la race, puis la couleur
2. Faire un group by multiple sur la couleur et le sexe et trier par ordre croissant sur le sexe
3. Faire un group by par sexe, age, couleur
    - Pour les chiens âgés entre 2 et 15 ans (inclus)
    - Sélectionner les chiens dont le surnom n'est pas un nom composé (on cherche les surnoms sans espace)
    - Trier par sexe (croissant), par âge (décroissant), par race (décroissant), par couleur (croissant)

Exercice 8 : Requêtes, group by et into

```
List<Chien> chiens = new List<Chien>()
{
    new Chien("Gnocci", "Gnoc Gnoc", "Labrador", "Sable", "M", 1, 20),
    new Chien("Vagabond", "Gros Loup", "Labrador", "Noir", "M", 8, 25),
    new Chien("Milou", "Milos", "Labrador", "Sable", "M", 10, 24),
    new Chien("Sirène", "Sissy", "Labrador", "Sable","F", 4, 19),
    new Chien("Félicia", "Felicci", "Labrador", "Sable", "F", 6, 20),
    new Chien("Kratos", "Mon tueur", "Chihuahua", "Fauve", "M", 1, 2),
    new Chien("Jack", "Jaja", "Chihuahua", "Fauve", "M", 1, 2),
    new Chien("Mojave", "Mojojo", "Chihuahua", "Fauve", "M", 1, 2),
    new Chien("Hercule", "Herc", "Chihuahua", "Beige", "M", 35, 2),
    new Chien("Médusa", "Med", "Terre-Neuve", "Noire", "F", 6, 40),
    new Chien("Mélusine", "Mel", "Terre-Neuve", "Noire", "F", 7, 41),
    new Chien("Venus", "Violette", "Terre-Neuve", "Noire", "F", 8, 38),
    new Chien("Letto", "Lele", "Berger Australien", "Bleu Merle", "M", 3, 30),
    new Chien("Cabron", "Dum dum", "Berger Australien", "Bleu Merle", "M", 9, 31),
    new Chien("Banzaï", "Zaïzaï", "Berger Australien", "Noir et blanc", "M", 1, 28 ),
    new Chien("Haricot", "Harry", "Berger Australien", "Noir et blanc", "M", 2, 27),
    new Chien("Gédéon", "Gégé", "Berger Allemand", "Noir et feu", "M", 9, 31),
    new Chien("Bella", "Belbel", "Berger Allemand", "Noir et feu", "F", 5, 28),
    new Chien("Oui-oui", "oui", "Berger Allemand", "Sable", "M", 7, 35),
    new Chien("Pataud", "Patoche", "Carlin", "Sable", "M", 16, 8),
    new Chien("Killer", "Kiki", "Carlin", "Sable", "M", 10, 8),
    new Chien("Frank", "Colonel", "Carlin", "Noir", "M", 9, 9)
};
```

1. Récupérer et afficher les chiens après avoir les avoir regrouper par leur race et les avoir trié par âge croissant **sans utiliser orderby**
2. Regrouper les chiens par âge dans ageChiens

- Trier les chiens par âge croissant
- Regrouper depuis ageChiens les chiens ayant un âge pair et ceux impair
- Afficher le nom des chiens impairs, puis pairs classé par age
  Exemple de présentation :

```
Pair : x
  Age : y
    Nom 1
    Nom 2
    ...
  Age : z
    Nom 3
    ...
```

3. Faire un group by sur la race et la couleur des chiens et créer chiensRaceCouleur avec into, Trier les chiens par race et par couleur (ordre croissant)