## 9) Les collections en java

Une collection peut s'apparenter à un tableau qui contient des références d'objet. Sa taille n'est pas figée et elle possède des méthodes la manipuler.

Une des classes techniques Collection en java est la classe ***ArrayList<E>***

**E étant un élément de type Classe**

**Documentation complète :**
http://docs.oracle.com/javase/6/docs/api/java/util/ArrayList.html

**Documentation sommaire :**

**Method Summary**

| | |
|---|---|
| boolean | add(E e)<br>Appends the specified element to the end of this list. |
| void | clear()<br>Removes all of the elements from this list. |
| boolean | contains(Object o)<br>Returns true if this list contains the specified element. |
| E | get(int index)<br>Returns the element at the specified position in this list. |
| int | indexOf(Object o)<br>Returns the index of the first occurrence of the specified element in this list, or -1 if this list does not contain the element. |
| boolean | isEmpty()<br>Returns true if this list contains no elements. |
| int | lastIndexOf(Object o)<br>Returns the index of the last occurrence of the specified element in this list, or -1 if this list does not contain the element. |
| E | remove(int index)<br>Removes the element at the specified position in this list. |
| boolean | remove(Object o)<br>Removes the first occurrence of the specified element from this list, if it is present. |
| int | size()<br>Returns the number of elements in this list. |

Exemple d'utilisation en java

```java
import java.util.ArrayList; //A importer pour utiliser ArrayList

public class TestCollection{
    public static void main(String[] args){
        //Déclaration de la collection
        ArrayList<String> lesFruits;

        //Instanciation de la collection
        lesFruits = new ArrayList<String>();

        //Ajout d'objet String dans la collection
        lesFruits.add("Pomme");
        lesFruits.add("Orange");
        lesFruits.add("Banane");
        lesFruits.add("Noix");
        lesFruits.add("Pomme");
        lesFruits.add("Kiwi");
        lesFruits.add("Kaki");
        lesFruits.add("Mandarine");

        //Parcours de la collection du début à la fin
        for(String unFruit : lesFruits){
            System.out.print(unFruit + "-");
        }
    }
}
```

A l'exécution nous obtenons :

```
Pomme-Orange-Banane-Noix-Pomme-Kiwi-Kaki-Mandarine-
```

**Remarque** : Une structure itérative a été spécialement conçue pour parcourir une collection du début à la fin :

```
for(Object unObjet : uneCollection)
```

Par exemple : `for(String unFruit : lesFruits)`

qui peut se traduire par "Pour chaque fruit dans lesFruits" permet de parcourir la collection **lesFruits** du début à la fin. La variable **unFruit** sera affecté de l'objet en cours.

## Exercice 10 :

Dans un programme de test et en utilisant la classe Client déjà existante, coder le scénario suivant :

- déclarer une collection lesClients,
- instancier la collection,
- créer 5 objets Client,
- ajouter ces 5 objets Clients à la collection,
- afficher le nom de tous les clients en parcourant la collection **en utilisant une boucle for pour chaque**,
- afficher le nombre de client présents dans la collection,
- afficher le nom de tous les clients en partant de la fin en parcourant la collection **avec un for classique** (rechercher dans la documentation sommaire les méthodes nécessaires),
- Dire si Oui ou Non il y a un client nommé "Wanda" dans la collection,
- enlever le client qui se trouve à l'indice 2 (réafficher les noms des clients pour vérifier),
- vider la collection (réafficher les noms des clients pour vérifier).