# Arrays

Un Array es un tipo de lista de variables ordenada en un único objeto

```
string[] array = new string[5];
string[] array2 = {"H", "o", "l", "a"};
```

# Collections

Las colecciones proporcionan una manera más flexible de trabajar con grupos de objetos. A diferencia de las matrices, el grupo de objetos con el que trabaja puede aumentar y reducirse de manera dinámica a medida que cambian las necesidades de la aplicación

## Listas:

Una lista es un tipo de colección ordenada(un array)

### Métodos habituales Listas

- `lista.Add(x)` -> Agrega al ultimo elemento de la lista "x"
- `lista.count` -> Devuelve el numero de elementos que tiene la lista
- `lista.IndexOf(x)` -> Devuelve la posición en laque se encuentra la primera x. se pueden poner los parámetros "start", "stop" que indican desde donde hasta donde del array recorrer
- `lista.Insert(x, y)` -> Inserta el objeto "y" en la posición "x"
- `lista.Remove(X)` -> Elimina el primer valor "x" de la lista
- `lista.Reverse()` -> Invierta la lista. Se trabaja sobre la lista real, no sobre copia

```
List<string> array3 = new List<string>();
array3.Add("H");
array3.Add("o");
```

## Tuples:

Una tuple es una lista, pero como una constante, osea que una vez agregados los datos, no se pueden modificar y es mas rápido que las listas, por tanto es recomendable si no tienes que agregar datos nuevos ni nada. Las tuples se identifican mediante paréntesis y comas

```
Tuple<int, string> tupla = new Tuple<int, string>(1, "hola");
```

Declaramos una clase Tuple, entre los `<>` ponemos separado por comas los tipos de variable que van a ser añadidos(int, string, float, bool...) y lo incializamos pasándo al constructor tantos parámetros como tipos hemos puesto entre los `<>`

## Diccionarios:

Un diccionario o tabla de hashes(en otros lenguajes) son colecciones que relacionan una clave y valor. osea que se asocia una especia de significado

### Métodos habituales de diccionario:

- `diccionario.ContainsKey(k)` -> Devuelve el valor de la key
- `diccionario.TryGetValue(k, out null)` -> Comprueba si existe esa key
- `diccionario.Keys` -> Devuelve una lista de claves
- `diccionario.Remove(k)` -> Borra el contenido asociado a la clave k
- `diccionario.Values` -> Devuelve una lista de los valores del diccionario

```
Dictionary<string, string> diccionario = new Dictionary<string, string>();

diccionario.Add("Clave", "resultado");
diccionario.Add("1", "asier");
diccionario.Add("apellido", "garcia");
```