Colecciones en C#

Arrays

```
int[] numeros = { 4, 8, 1, 6, 3 };

Console.WriteLine($"Longitud: {numeros.Length}");

Array.Sort(numeros);
Console.WriteLine("Ordenados:");
foreach (var n in numeros)
     Console.Write($"{n} ");

Array.Reverse(numeros);
Console.WriteLine("\nInvertidos:");
foreach (var n in numeros)
     Console.Write($"{n} ");

int pos = Array.IndexOf(numeros, 6);
Console.WriteLine($"\nPosición del número 6: {pos}");

Array.Clear(numeros);
Console.WriteLine("Después de limpiar:");
foreach (var n in numeros)
     Console.Write($"{n} ");
```

Listas

```
var frutas = new List<string> { "Pera", "Manzana", "Naranja" };

frutas.Add("Plátano");
frutas.Insert(1, "Kiwi");
frutas.Remove("Pera");
frutas.RemoveAt(0);

Console.WriteLine($"¿Contiene 'Plátano'?
{frutas.Contains("Plátano")}");

Console.WriteLine($"Número de frutas: {frutas.Count}");

frutas.Sort();

Console.WriteLine("Lista final ordenada:");
foreach (var f in frutas)
     Console.WriteLine($"- {f}");
```

Diccionarios

```
var edades = new Dictionary<string, int>();

edades.Add("Ana", 20);
edades.Add("Luis", 22);
edades.Add("Marta", 19);

Console.WriteLine($"¿Existe la clave
'Luis'?{edades.ContainsKey("Luis")}");

Console.WriteLine($"¿Existe el valor 20?
{edades.ContainsValue(20)}");

edades.Remove("Luis");

Console.WriteLine($"Total de elementos: {edades.Count}");
Console.WriteLine("Claves:");
foreach (var clave in edades.Keys)
     Console.WriteLine(clave);

Console.WriteLine("Valores:");
foreach (var valor in edades.Values)
     Console.WriteLine(valor);
```

Conjuntos

```
var A = new HashSet<int> { 1, 2, 3, 4 };
var B = new HashSet<int> { 3, 4, 5, 6 };

A.Add(7);
A.Remove(2);

Console.WriteLine($"¿Contiene 3? {A.Contains(3)}");

var union = new HashSet<int>(A);
union.UnionWith(B);
Console.WriteLine("Unión: " + string.Join(", ", union));

var inter = new HashSet<int>(A);
inter.IntersectWith(B);
Console.WriteLine("Intersección: " + string.Join(", ", inter));

var diff = new HashSet<int>(A);
diff.ExceptWith(B);
Console.WriteLine("Diferencia A - B: " + string.Join(", ", diff));
```

Pilas

```
var pila = new Stack<string>();

pila.Push("Inicio");
pila.Push("Tienda");
pila.Push("Carrito");

Console.WriteLine($"Elemento superior: {pila.Peek()}");
Console.WriteLine($"Desapilando:{pila.Pop()}");
Console.WriteLine($"Total restante: {pila.Count}");

Console.WriteLine("Elementos restantes:");
foreach (var e in pila)
     Console.WriteLine(e);
```

Colas

```
var cola = new Queue<string>();

cola.Enqueue("Cliente1");
cola.Enqueue("Cliente2");
cola.Enqueue("Cliente3");

Console.WriteLine($"Primero en la cola: {cola.Peek()}");

Console.WriteLine($"Atendiendo a:
{cola.Dequeue()}");

Console.WriteLine($"Clientes
restantes:{cola.Count}");
Console.WriteLine("Cola actual:");
foreach (var c in cola)
     Console.WriteLine(c);
```

Conversiones entre colecciones sin usar LINQ

De \ A	Array	List	HashSet	Dictionary<TKey,TValue>	Stack	Queue
Array		`var list = new List<T>(array);`	`var set = new HashSet<T>(array);`	`var dict = new Dictionary<int,T>(); for(int i=0;i<array.Length;i++) {dict[i]=array[i]; }`	`var stack = new Stack<T>(array);`	`var queue = new Queue<T>(array);`
List	`var array = list.ToArray();`		`var set = new HashSet<T>(list);`	`var dict = new Dictionary<int,T>(); for(int i=0;i<list.Count;i++) {dict[i]=list[i]; }`	`var stack = new Stack<T>(list);`	`var queue = new Queue<T>(list);`
HashSet	`var array = set.ToArray();`	`var list = new List<T>(set);`		`var dict = new Dictionary<int,T>(); int i=0; foreach(var v in set) {dict[i]=v; i++; }`	`var stack = new Stack<T>(set);`	`var queue = new Queue<T>(set);`
Dictionary <TKey,TValue>	`var array = new KeyValuePair<TKey,TValue> [dict.Count]; dict.CopyTo(array,0);`	`var list = new List<KeyValuePair <TKey,TValue>>(dict);`	`var set = new HashSet<TKey>(dict.Keys);`		`var stack = new Stack<KeyValuePair <TKey,TValue>>(dict);`	`var queue = new Queue<KeyValuePair <TKey,TValue> >(dict);`
Stack	`var array = stack.ToArray();`	`var list = new List<T>(stack);`	`var set = new HashSet<T>(stack);`	`var dict = new Dictionary<int,T>(); int i=0; foreach(var v in stack) {dict[i]=v; i++; }`		`var queue = new Queue<T>(stack.Reverse());`
Queue	`var array = queue.ToArray();`	`var list = new List<T>(queue);`	`var set = new HashSet<T>(queue);`	`var dict = new Dictionary<int,T>(); int i=0; foreach(var v in queue) {dict[i]=v; i++; }`	`var stack = new Stack<T>(queue);`	