Collection Classes - Linear Data Structures - Lists - C#

A **Class** is a description of an object - the data and behavior for an object.

An **Object** is an instance of a class.

A recipe is description of a dish.

A Cake recipe describes how to make a cake; ingredients and the steps of how to use them to make a cake.

The cake you make using the recipe is an instance of the recipe.

Making a a cake using the recipe is instantiation of the cake recipe.

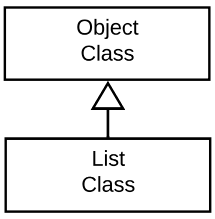
Collection classes are **provided by C#** to allow us to **create groups of objects** that can be **mainipulated/processed as a group.**

List is a Collection class that allows us to easily use objects in an array.

Arrays were **fixed-length, only one type of data, use an index to access an element.**
It is difficult to add or remove elements from an array
we have to know how many elements we need to store when we define the array.

Lists are **variable # of elements, hold more than one type of objects, use methods to access the elements** in the List.
It is easy to add or remove elements from a List
The number elements to be stored does not need to be known when defined.

The Object Class is what all C# classes are types of.



A List is a type of Object

To define an List:

```
List<data-type>   name      = new List<data-type>();
List<String> firstNames    = new List<String>();   // Empty List
```

```
List<data-type>   name      = new List<data-type>() { initial-values };
List<String> firstNames    = new List<String>() { "Worf", "Data", "Jordi" };
```

An List is a Collections Class that treats a Collection of Objects as a flexible array.

Elements are stored in the same order they are added when using `.Add(Object)`

The number of elements in an List can be found using `.Count` *`(this is a property not a method)`*

Elements may be accessed by relative index number using: `listName[index]`   `// Just like an array`

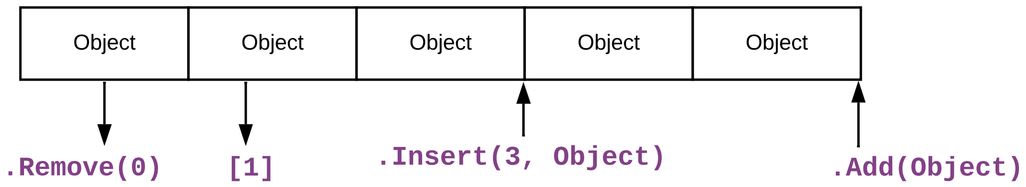
Elements may be added to any place in the List using `. Insert(index, Object)`   `// Insert object after element at index`

Elements may be removed from the List using: **.RemoveAt(index)** // Remove object at index
.Remove(object) // Remove first occurance of object
.RemoveAll(condition) // Remove all objects that satisfy the conditon

There are many methods available to process/access an List.

C# List documentation site:

https://learn.microsoft.com/en-us/dotnet/api/system.collections.generic.list-1?view=net-9.0


Object
Object
Object
Object
Object
.Remove(0)
[1]
.Insert(3, Object)
.Add(Object)