📗**ArrayList**

ArrayList is non generic collection, like an array but add more functionalities.

Such as dynamic resizing, adding, and deleting elements from the middle of collection.

ArrayList can be used for any unknown data.

## **⚒️How to Create**

* ArrayList() : Initialize new instance (empty and default initial capacity)
* ArrayList(ICollection c) : Initialize that element copied from the specified collection.
* ArrayList(int capacity) : Initialize (empty but specified capacity)

## **➕Add Element**

Add() for add element to array list.

| ArrayList myArrayList = new ArrayList(); myArrayList.Add(101); myArrayList.Add("JAMES"); |
| --- |

## **👓How to Access**

As it implements **IList**, so we can access the element using the indexer (start from 0).

While adding element to ArrayList, the element will be boxed (casted) and stored in collection.

| **int** firstElement = (**int**)arrayList1[0]; *//returns 101* string secondElement = (string)arrayList1[1]; *//returns "James"* |
| --- |

## **🏃🏻How to Iterate**

As it implements **ICollection**, it supports iteration.

| foreach(var item **in** arrayList1) { Console.Write(item); } |
| --- |

## **🔌How to Insert Element (into specified position)**

Add() will be add element to end of collection.

Insert() will be insert element into a specified index position.

| //**Insert** "First Element" **at** **First** **Position** i.e. **Index** 0 arrayList.Insert(0, "First Element"); |
| --- |

## **❌How to Remove Elements**

* Remove(object? obj) : Remove first occurence of specific object
* RemoveAt(int index) : Remove element at specific index
* RemoveRange(int index, int count) : Remove a range of elements from ArrayList.
* Clear() : Remove all elements.

| *//Insert "First Element" at First Position i.e. Index 0*  **arrayList**.Remove("HongKong"); *//Remove element at index*  **arrayList**.RemoveAt(3); *//Removes two elements starting from 1st item (0 index)*  **arrayList**.RemoveRange(0, 2); *//Remove all items from the Array list*   **arrayList**.Clear(); |
| --- |

## **🔍Check Element**

bool Contains(object? item) : Locate the object, return true if found.

| Console.WriteLine($"\n\nIs ArrayList Contains India: {arrayList.Contains("India")}"); |
| --- |

## **📚Sort ArrayList**

* Sort() : Sort elements
* Sort(IComparer? comparer) : Sort element using specified comparer
* Sort(int index, int count, IComparer? comparer) : Sort element in a range of element using specified comparer

| arrayList.Sort()*;* |
| --- |

## **🖖🏻Conclusion**

| Array | ArrayList |
| --- | --- |
| Fixed Length | Variable Length |
| Cannot insert it into the middle | Can insert an element into the middle of the collection |
| Cannot delete from middle | Can delete elements from the middle of the collection |
| It is type-safe, so we can store only similar types of data based on the data type. | It is not type-safe, so we can store any type of data. |
| Boxing and Unboxing are not required. | Boxing and Unboxing are required as it is operated on the object data type. |