Collections : ArrayList

- ArrayList is a non-generic type of collection in C#.
- It can contain elements of any data types.
- It is similar to an array, except that it grows automatically as you add items in it.
- Unlike an array, we don't need to specify the size of ArrayList.

Property of ArrayList class

Capacity
Gets or sets the number of elements that the ArrayList can contain.

Count
Gets the number of elements actually contained in the ArrayList.

IsFixedSize
Gets a value indicating whether the ArrayList has a fixed size.

IsReadOnly
Gets a value indicating whether the ArrayList is read-only.

Item
Gets or sets the element at the specified index.

Methods of ArrayList

public virtual int Add(object value);
Adds an object to the end of the ArrayList.

public virtual void AddRange(ICollection c);
Adds the elements of an ICollection to the end of the ArrayList.

public virtual void Clear();
Removes all elements from the ArrayList.

public virtual bool Contains(object item);
Determines whether an element is in the ArrayList.

public virtual ArrayList GetRange(int index, int count);
Returns an ArrayList which represents a subset of the elements in the source ArrayList.

public virtual int IndexOf(object);
Returns the zero-based index of the first occurrence of a value in the ArrayList or in a portion of it.

public virtual void Insert(int index, object value);
Inserts an element into the ArrayList at the specified index.

public virtual void InsertRange(int index, ICollection c);
Inserts the elements of a collection into the ArrayList at the specified index.

public virtual void Remove(object obj);
Removes the first occurrence of a specific object from the ArrayList.

public virtual void RemoveAt(int index);
Removes the element at the specified index of the ArrayList.

public virtual void RemoveRange(int index, int count);
Removes a range of elements from the ArrayList.

public virtual void Reverse();
Reverses the order of the elements in the ArrayList.

public virtual void SetRange(int index, ICollection c);
Copies the elements of a collection over a range of elements in the ArrayList.

public virtual void Sort();
Sorts the elements in the ArrayList.

public virtual void TrimToSize();
Sets the capacity to the actual number of elements in the ArrayList.

Application Program which demonstrate use of ArrayList methods and properties.

```
using System;
using System.Collections;

public class Marvellous
{
	public static void Main(string[] args)
	{
		ArrayList al = new ArrayList();

		al.Add(11);
		al.Add(21);
		al.Add(51);

		Console.WriteLine("Capacity: {0} ", al.Capacity);
		Console.WriteLine("Count: {0}", al.Count);

		Console.Write("Content of arraylist: ");
		foreach (int i in al)
		{
			Console.Write(i + " ");
		}

		Console.WriteLine("\nArrayList after Sorting:");

		al.Sort();
```

```
            foreach (int i in al)
            {
                  Console.Write(i + " ");
            }

            al.Add(9.7);
            al.Add("Marvellous");
            al.Add('M');

            Console.WriteLine("\nContent of arraylist: ");
            foreach (Object i in al)
            {
                  Console.Write(i + " ");
            }

            // Access element of arraylist using index
            int j = (int)al[0];

            Console.WriteLine("\nAccessed element is {0}",j);

            Console.WriteLine("Traversal using for loop");
            for(int k = 0 ; k< al.Count; k++)
             {
                  Console.WriteLine(al[k]);
             }

             // Inserting element at specific position
             al.Insert(2, 101);

             // Remove the element by value
             al.Remove(101); //Removes 101 from ArrayList

            // Remove the element by index
            al.RemoveAt(1);  // Remove element at index 1

            Console.WriteLine("Traversal using for loop after update");
            for(int k = 0 ; k< al.Count; k++)
            {
                  Console.WriteLine(al[k]);
            }
      }
}
```