COLLECTIONS :

LIST :

```
namespace List
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            List<int> numbers= new List<int>() { 1, 2, 3, 4, 5, 6 };
            numbers.Add(7);
            numbers.Add(8);
            Console.WriteLine($"index at 0 :{numbers[0]}");
            Console.WriteLine($"Count : {numbers.Count()}");
            Console.WriteLine("Print ForEach LINQ Method :");
            numbers.ForEach(num => Console.Write(num + ", "));
            Console.Write("\n");
            int[] no = new int[] { 9, 10, 11 };
            numbers.AddRange(no);
            numbers.Insert(0, 0);
            Console.WriteLine($"After adding element using addrange and insert : {numbers.Count()}");
            numbers.Remove(0);
            numbers.RemoveAt(11 - 1);
            Console.WriteLine($"After using remove and removeat : {numbers.Count()}");
            Console.WriteLine($"Element Contains : {numbers.Contains(11)}");
            Console.ReadKey();
        }
    }
}
```

```
Select D:\.NET Framework\ConsoleApp\Collection\List\List\bin\Debug\List.exe
index at 0 :1
Count : 8
Print ForEach LINQ Method :
1, 2, 3, 4, 5, 6, 7, 8,
After adding element using addrange and insert : 12
After using remove and removeat : 10
Element Contains : False
```

STACK :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Stack
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Stack<int> numbers = new Stack<int>();
            numbers.Push(1);
            numbers.Push(2);
            numbers.Push(3);
            numbers.Push(4);
            Console.WriteLine($"No.of .Elements using Count : {numbers.Count()}");
            Console.WriteLine($"Elements using peek : {numbers.Peek()}");
            Console.WriteLine($"Elements using pop : {numbers.Pop()}");
            Console.WriteLine($"Elements 4 exist in stack : {numbers.Contains(4)}");
            numbers.Clear();
            Console.WriteLine($"No.of .Elements after cleared : {numbers.Count()}");
            Console.ReadKey();
        }
    }
}
```

```
D:\.NET Framework\ConsoleApp\Collection\Stack\Stack\Stack\bin\Debug\Stack.exe
No.of .Elements using Count : 4
Elements using peek : 4
Elements using pop : 4
Elements 4 exist in stack : False
No.of .Elements after cleared : 0
```

QUEUE :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Queue
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Queue<int> numbers = new Queue<int>();
            numbers.Enqueue(1);
            numbers.Enqueue(2);
            numbers.Enqueue(3);
            numbers.Enqueue(4);
            Console.WriteLine($"No.of .Elements using Count : {numbers.Count()}");
            Console.WriteLine($"Elements using peek : {numbers.Peek()}");
            Console.WriteLine($"Elements using dequeued : {numbers.Dequeue()}");
            Console.WriteLine($"Elements 4 exist in stack : {numbers.Contains(4)}");
            numbers.Clear();
            Console.WriteLine($"No.of .Elements after cleared : {numbers.Count()}");
            Console.ReadLine();
        }
    }
}
```

```
D:\.NET Framework\ConsoleApp\Collection\Queue\Queue\Queue\bin\Debug\Queue.exe
No.of .Elements using Count : 4
Elements using peek : 1
Elements using dequeued : 1
Elements 4 exist in stack : True
No.of .Elements after cleared : 0
```

SORTEDLIST :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SortedList
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            SortedList<int, string> noNames = new SortedList<int, string>();
            noNames.Add(1, "One");
            noNames.Add(4, "Four");
            noNames.Add(2, "Two");
            noNames.Add(10, null);
            noNames.Add(3, "Three");
            Console.WriteLine("Sorted List : ");
            noNames[5] = "Five";
            foreach (var item in noNames)
                Console.Write("key: {0}, value: {1}", item.Key, item.Value);
            Console.Write("\n");
            Console.WriteLine($"Check Key- 4 Exist : {noNames.ContainsKey(4)}");
            Console.WriteLine("Trygetvalue of key using out : ");
            string result;
            if (noNames.TryGetValue(4, out result))
                Console.WriteLine("Key: {0}, Value: {1}", 4, result);
            Console.WriteLine("Using key and value in for loop :");
            for (int i = 0; i < noNames.Count; i++)
            {
                Console.WriteLine("key: {0}, value: {1}", noNames.Keys[i], noNames.Values[i]);
            }
            Console.WriteLine("Using Remove 10 key and RemoveAt 0 index: ");
            noNames.Remove(10);
            noNames.RemoveAt(0);
            for (int i = 0; i < noNames.Count; i++)
            {
                Console.WriteLine("key: {0}, value: {1}", noNames.Keys[i], noNames.Values[i]);
            }

            Console.ReadLine();
```

```
D:\.NET Framework\ConsoleApp\Collection\SortedList\SortedList\SortedList\bin\Debug\SortedList.exe
Sorted List :
key: 1, value: Onekey: 2, value: Twokey: 3, value: Threekey: 4, value: Fourkey: 5, value: Fivekey: 10, value:
Check Key- 4 Exist : True
Trygetvalue of key using out :
Key: 4, Value: Four
Using key and value in for loop :
key: 1, value: One
key: 2, value: Two
key: 3, value: Three
key: 4, value: Four
key: 5, value: Five
key: 10, value:
Using Remove 10 key and RemoveAt 0 index:
key: 2, value: Two
key: 3, value: Three
key: 4, value: Four
key: 5, value: Five
```

DICTIONARY :

```
namespace Dictionary
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Dictionary<int, string> noDict = new Dictionary<int, string>();
            noDict.Add(1, "One");
            noDict.Add(4, "Four");
            noDict.Add(2, "Two");
            noDict.Add(3, "Three");
            foreach (var item in noDict)
                Console.WriteLine("Key: {0}, Value: {1}", item.Key, item.Value);
            Console.WriteLine($"Key 1 value : {noDict[1]}");
            Console.WriteLine($"Key 2 Exist using containkey : { noDict.ContainsKey(2)}");
            Console.WriteLine($"Key 3 Exist using Trygetvalue : ");

            string result;

            if (noDict.TryGetValue(3, out result))
            {
                Console.WriteLine(result);
            }
            Console.WriteLine($"Using ElementAt  :");
            for (int i = 0; i < noDict.Count; i++)
            {
                Console.WriteLine("Key: {0}, Value: {1}",noDict.ElementAt(i).Key,noDict.ElementAt(i).Value);
            }
            Console.WriteLine("Remove element : ");
            noDict.Remove(1);
            Console.WriteLine("Elements after removal of key 1 :");
            for (int i = 0; i < noDict.Count; i++)
            {
                Console.WriteLine("Key: {0}, Value: {1}", noDict.ElementAt(i).Key, noDict.ElementAt(i).Value);
            }
            noDict.Clear();
            Console.Write($"Elements after removal of all elements :{noDict.Count()}");
            Console.ReadLine();

        }
    }
}
```

```
D:\.NET Framework\ConsoleApp\Collection\Dictionary\Dictionary\Dictionary\bin\Debug\Dictionary.exe
Key: 1, Value: One
Key: 4, Value: Four
Key: 2, Value: Two
Key: 3, Value: Three
Key 1 value : One
Key 2 Exist using containkey : True
Key 3 Exist using Trygetvalue :
Three
Using ElementAt  :
Key: 1, Value: One
Key: 4, Value: Four
Key: 2, Value: Two
Key: 3, Value: Three
Remove element :
Elements after removal of key 1 :
Key: 4, Value: Four
Key: 2, Value: Two
Key: 3, Value: Three
Elements after removal of all elements :0
```

ARRAYLIST :

```
{
    0 references
    static void Main(string[] args)
    {
        var ListAll = new ArrayList(){2, "Steve", " ", true, 4.5, null};
        Console.WriteLine("\nPrint Array List :");
        foreach (var item in ListAll)
            Console.Write(item + ", ");
        int[] arr = { 100, 200, 300, 400 };
        ListAll.AddRange(arr);
        Console.WriteLine("\nElements after using addrange : ");
        foreach (var item in ListAll)
            Console.Write(item + ", ");
        Console.WriteLine("\nElement after Insert at index 1 : ");
        ListAll.Insert(1, "Jaga");
        foreach (var item in ListAll)
            Console.Write(item + ", ");

        Console.WriteLine("\nRemove & Removeat index :");
        ListAll.Remove("Jaga");
        ListAll.RemoveAt(5);
        foreach (var item in ListAll)
            Console.Write(item + ", ");
        Console.WriteLine($"\nElement 2 exist in list : {ListAll.Contains(2)}");
        Console.ReadLine();


    }
}
```

```
D:\.NET Framework\ConsoleApp\Collection\ArrayList\ArrayList\ArrayList\bin\Debug\ArrayList.exe

Print Array List :
2, Steve,  , True, 4.5, ,
Elements after using addrange :
2, Steve,  , True, 4.5, , 100, 200, 300, 400,
Element after Insert at index 1 :
2, Jaga, Steve,  , True, 4.5, , 100, 200, 300, 400,
Remove & Removeat index :
2, Steve,  , True, 4.5, 100, 200, 300, 400,
Element 2 exist in list : True
```

HASHTABLE :

```
0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        Hashtable numberNames = new Hashtable();
        numberNames.Add(1, "One");
        numberNames.Add(2, "Two");
        numberNames.Add(3, "Three");
        Console.WriteLine("Print HashTable :");
        foreach (DictionaryEntry de in numberNames)
            Console.WriteLine("Key: {0}, Value: {1}", de.Key, de.Value);
        numberNames[4] = "four";
        numberNames.Remove(1);
        Console.WriteLine("After update and remove :");
        foreach (DictionaryEntry de in numberNames)
            Console.WriteLine("Key: {0}, Value: {1}", de.Key, de.Value);
        numberNames.Clear();
        Console.WriteLine("After Clearing :");
        foreach (DictionaryEntry de in numberNames)
            Console.WriteLine("Key: {0}, Value: {1}", de.Key, de.Value);
        Console.ReadLine();

    }
}
```

```
D:\.NET Framework\ConsoleApp\Collection\Hashtable\Hashtable\Hashtable\bin\Debug\Hashtable.exe
Print HashTable :
Key: 3, Value: Three
Key: 2, Value: Two
Key: 1, Value: One
After update and remove :
Key: 4, Value: four
Key: 3, Value: Three
Key: 2, Value: Two
After Clearing :
```