Non Generic Collections

|  |  |  |
| --- | --- | --- |
|  | Hashtable | ArrayLists |
|  | Same as dictionary  Stores key- value pairs  Non generic collection | Same as array but its size increases dynamically. It can be used to add data whose type and size is not known. It has Add(), Count(), Capacity() |
| Create | Hashtable numberNames = new Hashtable(); | using System.Collections;  ArrayList arlist = new ArrayList(); |
| Read | foreach(DictionaryEntry de in numberNames)  Console.WriteLine("Key: {0}, Value: {1}", de.Key, de.Value); | //Access individual item using indexer  int firstElement = (int) arList[0]; |
| Update | numberNames.Add(1,"One");  cities["UK"] = "Liverpool, Bristol"; // update value of UK key | To update we can use Add()  arList.Add(“ A”);  So A will be added to arList  To add a range of values:  int[] arr = { 100, 200, 300, 400 };  arList.AddRange(arr); //adding array in arraylist |
| Delete | cities.Remove("UK"); // removes UK  cities.Clear(); //removes all elements | To remove elements from arList:  arList.Remove(null); //Removes first occurance of null  arList.RemoveAt(4); //Removes element at index 4  arList.RemoveRange(0, 2) ;  //Removes two elements starting from 1st item (0 index) |

Generic Collections : Comes under System.Collections.Generic namespace

|  |  |  |  |
| --- | --- | --- | --- |
|  | Dictionary | LIST | Queue |
|  | Dictionary<TKey, TValue> stores key-value pairs.  Keys : should be unique and not null  Values: can be null and duplicate | Generic version of ArrayList | Queue<T> is FIFO (First In First Out) collection. |
| Create | IDictionary<int, string> numberNames = new Dictionary<int, string>();  numberNames.Add(1,"One"); //adding a key/value using the Add() method  numberNames.Add(2,"Two"); | List<int> primeNumbers = new List<int>();  primeNumbers.Add(1); | Enqueue() method is used  Queue<int> callerIds = new Queue<int>();  callerIds.Enqueue(1); |
| Read | //print value of key  Console.WriteLine(cities["UK"]);  //ElementAt() method to retrieve key-value pair using index  for (int i = 0; i < cities.Count; i++){  Console.WriteLine("Key: {0}, Value: {1}", cities.ElementAt(i).Key, cities.ElementAt(i).Value);} | List<int> numbers = new List<int>() { 1, 2, 5, 7, 8, 10 };  Console.WriteLine(numbers[0]); // prints 1  // using for loop  for(int i = 0; i < numbers.Count; i++)  Console.WriteLine(numbers[i]); | Dequeue() : retrieve elements from queue  Peek(): retrieve first element from queue  callerIds.Dequeue(); //prints all elements in the queue  callerIds.Peek(); // prints only the first element in the  queue |
| Update | //update value  cities["UK"] = "Liverpool";  //updating using ContainsKey() method  if (cities.ContainsKey("India"))  {  cities["India"] = "Goa";  } | numbers.Insert(1, 11);// inserts 11 at 1st index: after 1 |  |
| Delete | cities.Remove("UK");//remove key UK  cities.clear(); //deletes the entire dictionary | numbers.Remove(10); // removes the first 10 from a list | Remove() can be used to delete elements in the queue  Clear() to delete the entire queue |