Collections in C#

- Collections is considered as a library which contains the implementations of Data Structures.
- In case of C++ readymade implementations is available in STL(Standard Template Library).
- In case of Java & C# readymade implementations of major data structures is available in Collections.
- All this implementations are available in System.Collections namespace.
- There are different types of data structures in Collections as ArrayList, Stack, Queue, HashTable, SortedList, BitArray etc.
- Depends on the requirement we can select any specific data structure. As all the necessary methods are already implemented we can directly used in C# application development.
- Use of Collections reduce the overhead of providing the definitions of methods and its safe way to use it.
- .Net Framework provides collection classes, some in the System.Collections namespace and some in the System.Collecctions.Generic namespace.
- The System.Collection classes are not type safe.
- But starting with the .Net Framework 2.0 we were able to use the collections the System.Collections.Generic namespace that is type safe.
- The basic difference is that generic collections are strongly typed and non-generic collections are not, unless they've been specially written to accept just a single type of data.
- In the .NET Framework, the non-generic collections (ArrayList, Hashtable, SortedKist, Queue etc.) store elements internally in 'object' arrays which, store any type of data.
- This means that, in the case of value types (int, double, bool, char etc), they have to be 'boxed' first and then 'unboxed' when you retrieve them which is quite a slow operation.
- We don't have this problem with reference types, you still have to cast them to their actual types before you can use them.
- In contrast, generic collections (List<T>, Dictionary<T, U>, SortedList<T, U>, Queue<T> etc) store elements internally in arrays of their actual types and so no boxing or casting is ever required.
- This means that generic collections are faster than their non-generic counterparts when using value types and more convenient when using reference types.

Classes from System.Collection namespace

1. ArrayList
The ArrayList collection is similar to the Arrays data type in C#. The biggest difference is the dynamic nature of the array list collection.

2. Stack
The stack is a special case collection which represents a last in first out (LIFO) concept

3. Queues
The Queue is a special case collection which represents a first in first out concept

4. Hashtable
A hash table is a special collection that is used to store key-value items

5. SortedList
The SortedList is a collection which stores key-value pairs in the ascending order of key by default.

6. BitArray
A bit array is an array of data structure which stores bits

Classes from System.Collection.Generic namespace

1. List<T>
Generic List<T> contains elements of specified type.
It grows automatically as you add elements in it.

2. Dictionary<TKey,TValue>
Dictionary<TKey,TValue> contains key-value pairs.

3. SortedList<TKey,TValue>
SortedList stores key and value pairs.
It automatically adds the elements in ascending order of key by default.

4. Hashset<T>
Hashset<T> contains non-duplicate elements. It eliminates duplicate elements.

5. Queue<T>
Queue<T> stores the values in FIFO style (First In First Out).
It keeps the order in which the values were added.
It provides an Enqueue() method to add values and a Dequeue() method to retrieve values from the collection.

6. Stack<T>
Stack<T> stores the values as LIFO (Last In First Out).
It provides a Push() method to add a value and Pop() & Peek() methods to retrieve values.