**Real-Time Applications:**

1. **Collections:**

* The collections in C# are classes that represent a group of objects.
* With the help of C# Collections, we can perform different types of operations on objects such as Store, Update, Delete, Retrieve, Search, and Sort objects, etc.
* allocating memory dynamically to elements and accessing a list of items based on an index.

1. **List<T>**

* lists are more flexible and easier to use when you need to add or remove elements dynamically.

1. **SortedList<K, V>**

* an element can be accessed by its key or by its index.
* SortedList object internally maintains two arrays to store the elements of the list i.e. one for keys and one for values.

1. **LinkedList<T>**

* Used when we are not sure about the amount of memory, we may end up using for storing data.

1. **HashSet<T>**

* It is commonly used if we have to access elements randomly. It is because elements in a hash table are accessed using hash codes.
* The hashcode of an element is a unique identity that helps to identify the element in a hash table.
* It cannot contain duplicate elements. Hence, each hash set element has a unique hashcode.
* the HashSet<T> class is faster than the SortedSet<T> classes in search, addition, and data removal processes.

1. **SortedSet<T>**

* Maintains Ascending Order and does not store duplicates.

1. **Stack<T>**

* For LIFO

1. **Queue<T>**

* For FIFO

1. **Dictionary<K, V>**

* Stores in key-value pairs.
* Easily access through keys.
* Dictionary is faster than SortedDictionary because it is implemented as a hash-table, an algorithm that is designed to use excess memory in order to use as few operations as possible.

1. **SortedDictionary<K, V>**

* to store the key/value pairs in the sorted form and the sorting is done on the key.