Generic Collection

Islam Mohamed Mohamed Zakaria Abd El Rahman El Nagar

A generic collection is strongly typed so that we can eliminate runtime type mismatches, it improves the performance by avoiding boxing and unboxing.

1) List:

- Contains elements of specified type.
- Create a List:
  ```
  List<Part> parts = new List<Part>();
  ```
- Indexes start from zero.
- can be accessed by index
- and having methods for sorting
- We can also add a whole array to a list using the `AddRange` function
  ```
  List < int > numbers = new List < int > ();
  int [] array = new int [] {1, 2, 3};
  numbers AddRange (array);
  ```

---

2) Linked List:

- They are dynamic in nature which allocates the memory when required.
- The LinkedList class implements the ICollection<T> , IEnumerable<T> .
- Insertion and deletion is easy to implement.
- It has faster access time and can be expanded in constant time without memory overhead.
- Insertion and Deletion O (1), Search O (n).
- Create a List:
  ```
  LinkedList<string> sentence = new LinkedList<string>(words);
  ```

---

3) Queue:

- Is a special type of collection that stores the elements in FIFO style
- It keeps the order in which the values were added.
- Doesn't perform boxing-unboxing.
- Insertion and Deletion O (1), Search O (n).
- Create a List:
  ```
  Queue<string> numbers = new Queue<string>();
  ```

---

4) Stack:

- Is a special type of collection that stores the elements in LIFO style

- It provides compile-time type checking and does not perform boxing-unboxing because it is generic
  Stack cannot be sorted.
- It does not support an indexer.
- Insertion and Deletion O (1), Search O (n).
- Create a List:

```
Stack<string> numbers = new Stack<string>();
```

---

5) Dictionary:

- is a generic collection that storeskey-value pairs in no particular order.
- Keys must be unique and cannot be null.
- Values can be null or duplicate.
- Values can be accessed by passing associated key in the indexer
- O (1).
- Create a List:

```
Dictionary<string, string> openWith =    new Dictionary<string, string>();
```

---

6) Sorted List:

- A key must be unique and cannot be null.
- is an array of key-value pairs sorted by keys.
- A key must be unique and cannot be null.
- A value can be accessed by passing associated key in the indexer.
- O(n log n)
- Create a List:

```
SortedList mySL = new SortedList();
```

---

7) Sorted Set:

- The collection of objects in sorted order is called SortedSet
- Maintains a sorted order without affecting performance as elements are inserted and deleted.
- The SortedSet is sorted in the decreasing order with no redundancy of elements in the SortedSet meaning only unique elements are stored in the SortedSet.
- The type of elements to be stored in the SortedSet must be the same.
- The number of elements that can be held by the SortedSet is called the capacity of the SortedSet.
- Create a List:

```
SortedSet<string> aviFiles = mediaFiles1.GetViewBetween("avi", "avj");
```