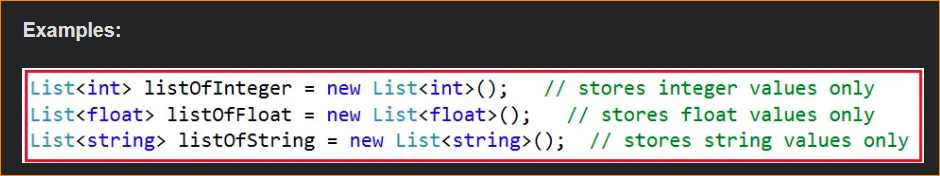
Non-Generic Collection classes such as ArrayList, Hashtable, SortedList, Stack, and Queue., etc. in Generic Collections such as ArrayList<T>, Dictionary<TKey, TValue>, SortedList<TKey, TValue>, Stack<T>, and Queue<T>. Here T is nothing but the type of values that we want to store in the collection. So, the most important point that you need to remember is while creating the objects of Generic Collection Classes, you need to explicitly provide the type of values that you are going to store in the collection.

A Generic Collection is Strongly Type-Safe. Which type of data do you want to store in generic type, this information you have to provide at compile time. It means you can only put one type of data into it. This eliminates type mismatches at runtime.

The Generic Collection Classes are implemented under the System.Collections.Generic namespace. The classes which are present in this namespace are as follows.

* Stack<T>: It represents a variable size last-in-first-out (LIFO) collection of instances of the same specified type.
* Queue<T>: It represents a first-in, first-out collection of objects.
* HashSet<T>: It represents a set of values. It eliminates duplicate elements.
* SortedList<TKey, TValue>: It represents a collection of key/value pairs that are sorted by key based on the associated System.Collections.Generic.IComparer implementation. It automatically adds the elements in ascending order of key by default.
* List<T>: It represents a strongly typed list of objects that can be accessed by index. Provides methods to search, sort, and manipulate lists. It grows automatically as you add elements to it.
* Dictionary<TKey, Tvalue>: It represents a collection of keys and values.
* SortedSet<T>: It represents a collection of objects that are maintained in sorted order.
* SortedDictionary<TKey, TValue>: It represents a collection of key/value pairs that are sorted on the key.
* LinkedList<T>: It represents a doubly linked list.

Note: Here the <T> refers to the type of values we want to store under them.



It is also possible to store a user-defined type like a class type or structure type as shown below

**List<Customer> customerList = new List<Customer> ();**

Assume the Customer is a user-defined class type that represents an entity Customer, Here we can store the customer objects within the customerList collection where each customer object can internally represent different attributes of the customer like Id, Name, City, State, etc.

1. What are Generic Collections in C#?

The Generic Collections in C# are strongly typed. The strongly typed nature allows these collection classes to store only one type of value into it. This not only eliminates the type mismatch at runtime but also will get better performance as they don’t require boxing and unboxing while they store value type data. So, it is always a preferable and a good programming choice to use the Generics Collection Classes in C# rather than using the Non-Generic Collection Classes.

Note: In most cases, it is recommended to use the generic collections because they perform faster than non-generic collections and also minimize exceptions by giving compile-time errors.

1. **List<T> Class in C#**

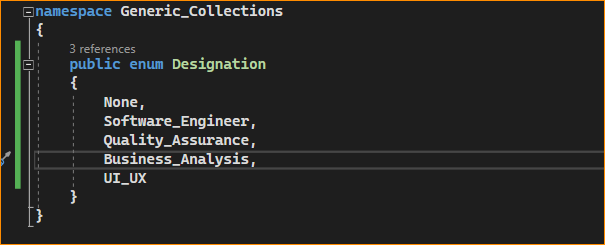
The List<T> Generic Collection Class in C# is used to store and fetch elements. It can have duplicate elements. It belongs to System.Collections.Generic namespace. You can also consider the List<T> collection as the generic version of ArrayList. Here, we need to mention the type of values that we want to store in the collection. Like, ArrayList we are unable to add any types of value into the List<T> collection, which prevents us from runtime exceptions due to type mismatch.

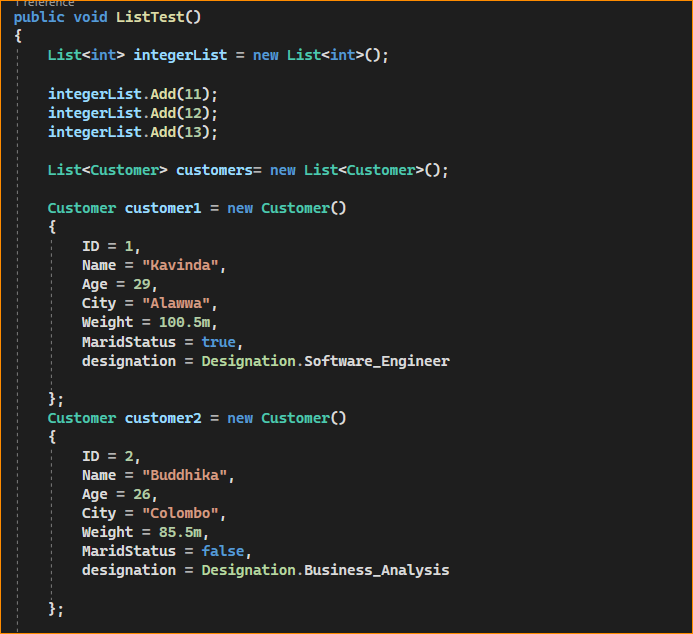
Ex-

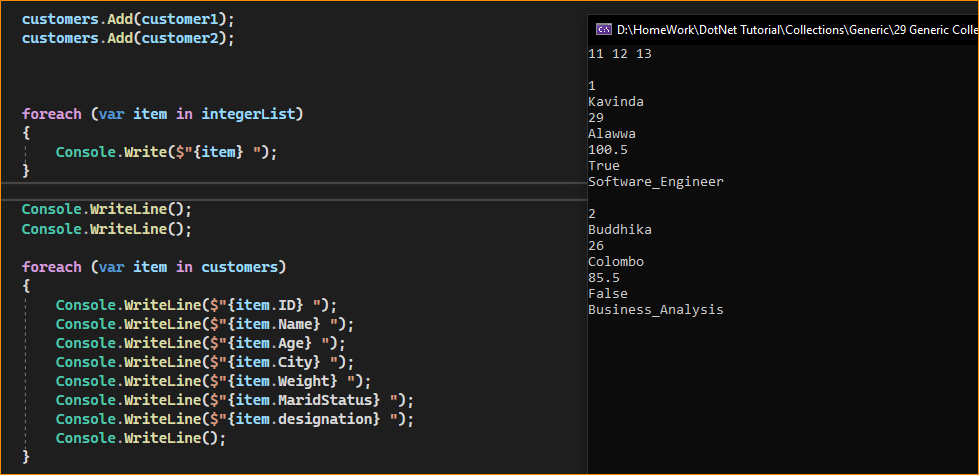
Customer Entity-



Enum – Designation

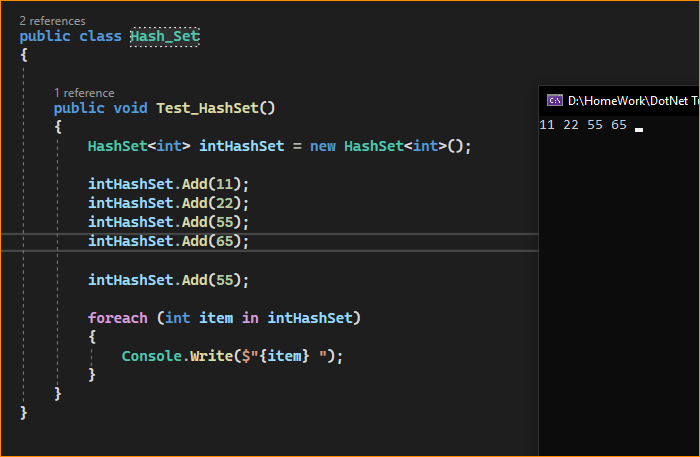






1. **HashSet<T> Class in C#**

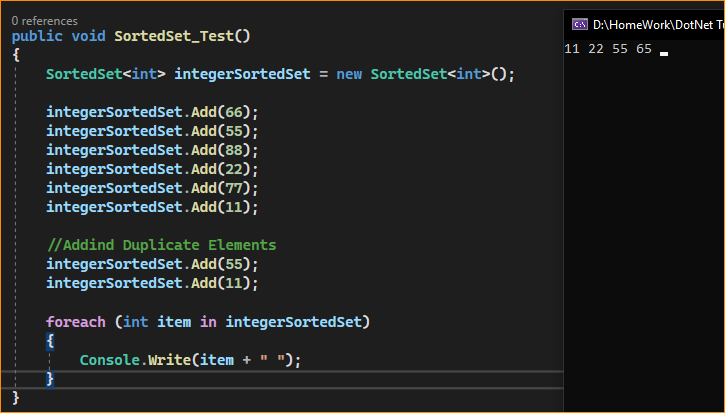
The Generic HashSet<T> Collection Class in C# can be used to store, remove or view elements. It does not allow the addition of duplicate elements. It is suggested to use the HashSet class if you have to store only unique elements. It belongs to System.Collections.Generic namespace. Let’s see an example of a generic HashSet<T> collection class that stores elements using Add() method and iterates elements using for-each loop.



I here Skiped 55.

1. **SortedSet<T> Class in C#:**

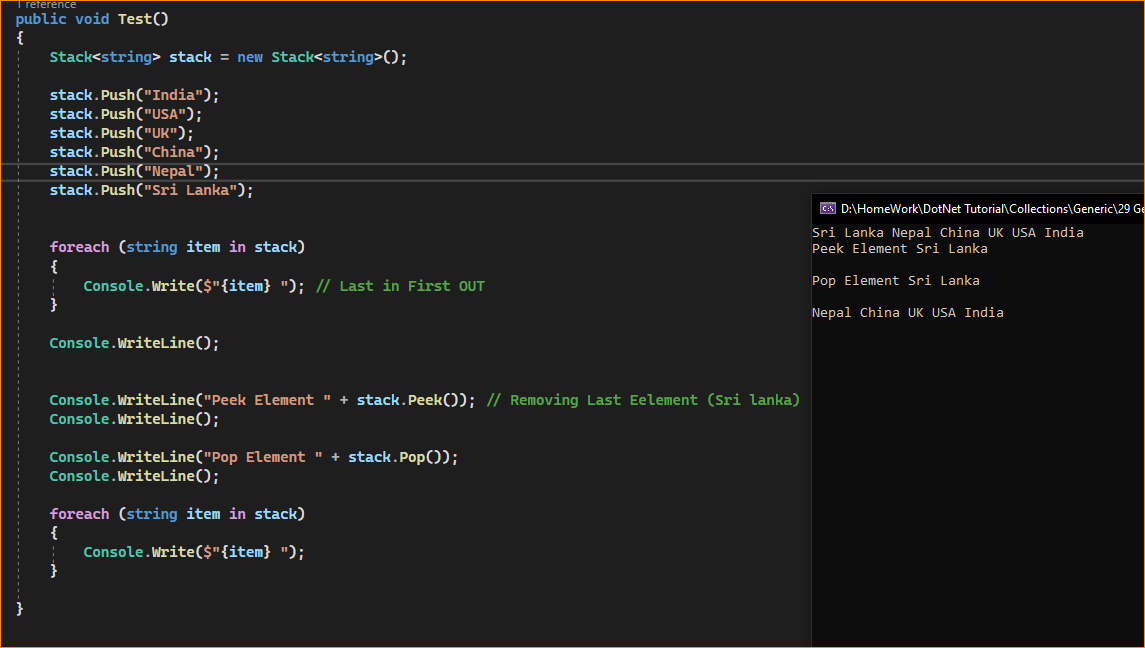
The Generic SortedSet<T> Collection Class in C# is used to store, remove or view elements. By default, it stores the elements in ascending order and does not store duplicate elements. It is recommended to use if you have to store unique elements as well as if you want to maintain ascending order. The SortedSet<T> class belongs to System.Collections.Generic namespace. Let’s see an example of a generic SortedSet<T> collection class in C# that stores elements using Add() method and iterates elements using for-each loop.



Skip Duplicate values and sort int values

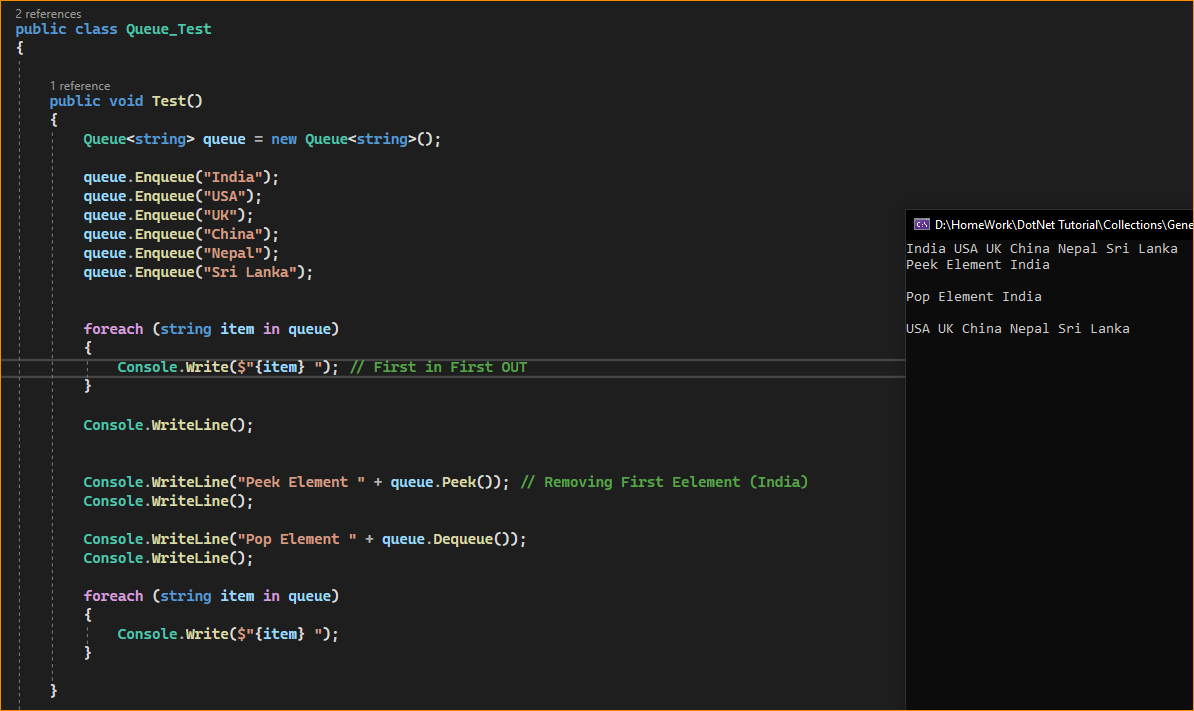
1. **Stack<T> Class in C#**

The Generic Stack<T> Collection Class in C# is used to push and pop elements in LIFO (Last in First Out) order. The push operation adds an element to a collection, whereas the pop operation is used to remove the most recently added element from a collection. It can have duplicate elements. The Stack<T> class belongs to System.Collections.Generic namespace. Let’s see an example of a generic Stack<T> collection class in C# that stores elements using the Push() method, removes elements using the Pop() method and iterates elements using for-each loop.



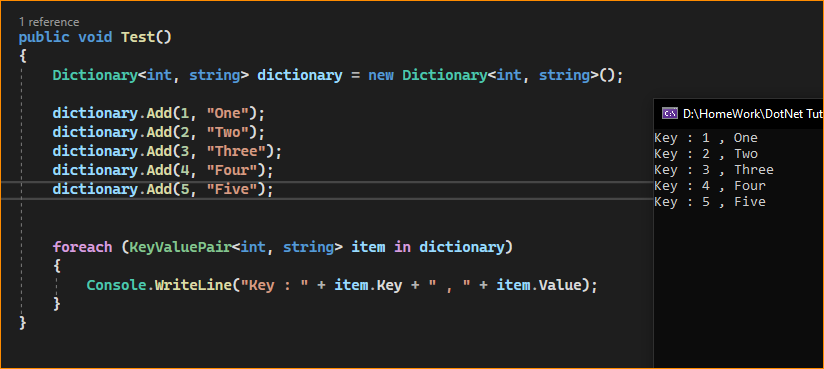
1. **Queue<T> Class in C#:**

The Generic Queue<T> Collection Class in C# is used to Enqueue and Dequeue elements in FIFO (First In First Out) order. The Enqueue operation adds an element in a collection, whereas the Dequeue operation is used to remove the firstly added element from the queue collection. It can have duplicate elements. The Queue<T> Collection Class belongs to System.Collections.Generic namespace. Let’s see an example of a generic Queue<T> collection class in C# that add elements using Enqueue() method, removes elements using Dequeue() method and iterates elements using the for-each loop.



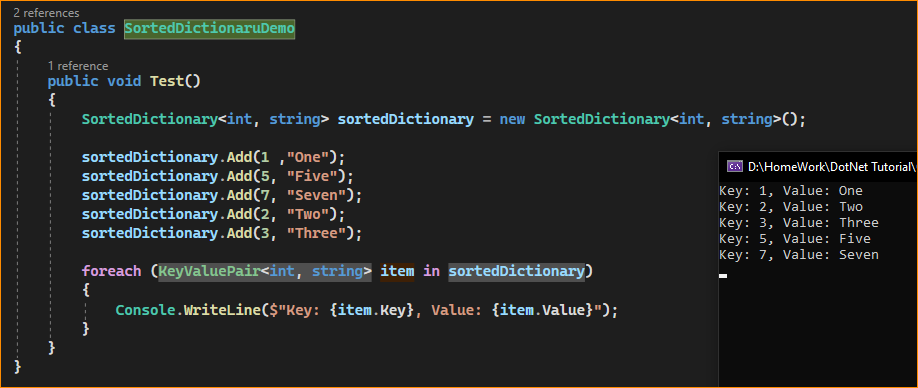
1. **Dictionary<TKey, TValue> Class in C#:**

The Generic Dictionary<TKey, TValue> Collection Class in C# is the generic version of Hashtable. It works as same as the Hashtable except that it operates on a type object, and this is one of the most useful collections based on key and value pairs. It stores values on the basis of keys. It contains unique keys only. With the help of the key, we can easily search or remove elements. The Dictionary<TKey, TValue> Collection Class belongs to System.Collections.Generic namespace. Let’s see an example of a generic Dictionary<TKey, TValue> collection class in C# that stores elements using Add() method and iterates elements using for-each loop. Here, we are using KeyValuePair class to get keys and values.



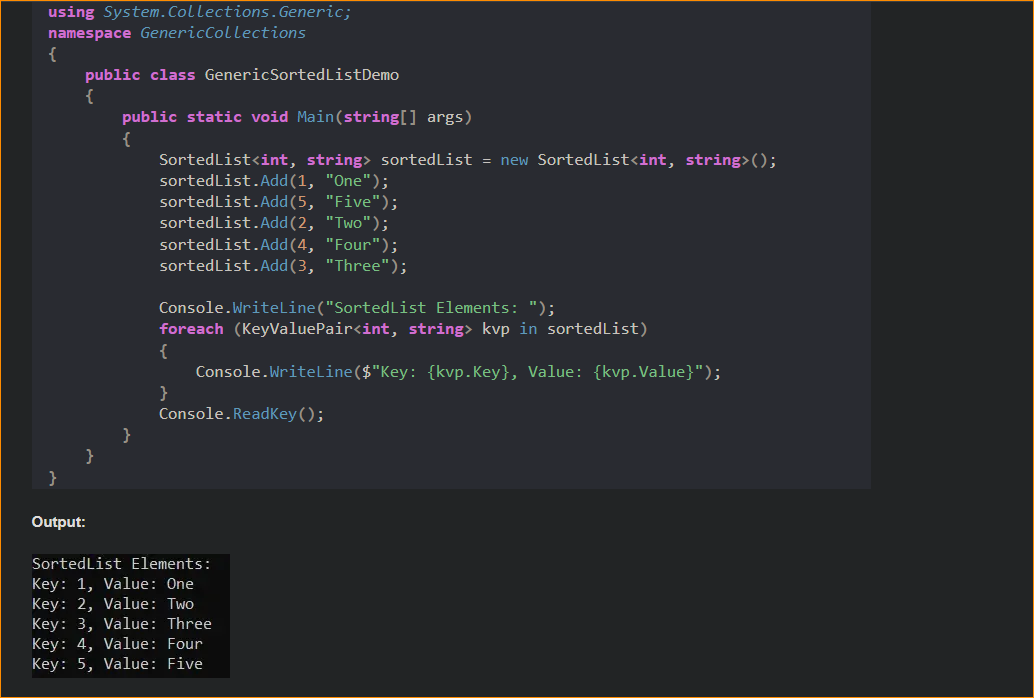
1. **SortedDictionary<TKey, TValue> Class in C#**

The Generic SortedDictionary<TKey, TValue> Collection Class in C# works similar to Dictionary<TKey, TValue> collection class. It stores values on the basis of keys. It contains unique keys and the most important thing is it stored the elements in ascending order on the key. With the help of a key, we can easily search or remove elements. The SortedDictionary<TKey, TValue> Collection Class belongs to System.Collections.Generic namespace. Let’s see an example of a generic SortedDictionary<TKey, TValue> collection class in C# that stores elements using Add() method and iterates elements using the for-each loop. Here, we are using KeyValuePair class to get keys and values.



1. **SortedList<TKey, TValue> Class in C#**

The Generic SortedList<TKey, TValue> Collection Class in C# is a collection of key/value pairs that are sorted according to keys. By default, this collection sort the key/value pairs in ascending order. With the help of a key, we can easily search, or remove elements. The SortedList<TKey, TValue> class belongs to System.Collections.Generic namespace.



1. **LinkedList<T> Class in C#:**

The Generic LinkedList<T> Collection Class in C# uses the concept of a linked list. It allows us to insert and delete elements firstly. It can have duplicate elements. The LinkedList<T> Collection Class belongs to System.Collections.Generic namespace. It allows us to add and remove elements before or after the last index. Let’s see an example of a generic LinkedList<T> collection class in C# that stores elements using AddLast() and AddFirst() methods and iterates elements using the for-each loop.

