Collections in C#

In c sharp collection are of two types.

Non-Generic collections.

Generic collections

Non-Generic collections.

Non generics collections in c sharp comes under namespace called “System.Collections”

Under Non-generic collections we have collection like.

* Array List
* Sorted List
* Hash Table
* Bit Array
* Queue
* Stack

All these collections are ready made classes with inbuilt methods and properties.

Array List:

* To use an ArrayList you need to add a System. Collections namespace.
* ArrayList implements the IList interface using an array whose size is dynamically increased as required.
* ArrayList is an alternate of an Array.
* Array is a fixed size collection whereas an Array List’s size can be dynamically increased or decreased.
* ArrayList is a collection of variables of the same type or multiple types.
* The capacity of an ArrayList is the number of elements the ArrayList can hold. As elements are added to an ArrayList, the capacity is automatically increased.
* The ArrayList collection accepts null as a valid value, an allows duplicate elements.
* default capcity is 4 as values added it will be multiplied 2.
* capacity can be restricted by using overloadded constructor of array list.
* Array list is hetrogenius
* ArrayList accepts duplicate values.
* ArrayList stores values in index order.

Methods and properties of arrayList:

Methods:

* AddRange();
* BinarySearch();
* Clear();
* Clone();
* IndexOf();
* Insert();
* InsertRange();
* LastIndexOf();
* Remove();
* RemoveAt();
* Reverse();
* Sort();
* ToArray();
* TrimToSize();
* SetRange();
* RemoveRange();
* Contains();
* CopyTo();
* Add();

Properties:

* Count;
* IsFixedSize;
* IsReadOnly;
* Capacity;

SortedList:

* A C# SortedList class represents a collection of key/value pairs that are sorted by the keys and are accessible by key and by index. The SortedListv class is included in the System.Collection namespace.
* A SortedList object internally manages two arrays to store the elements of the list; that is, one array for the keys and another array for values.
* In the SortedList index, the sequence is based on the sort sequence. When an element is added, it is inserted into SortedList in the correct sort order,and the indexing adjusts accordingly.
* When an element is removed, the indexing also adjusts accordingly. Therefore, the index of a specific key/value pair might change as elements are added or removed from the SortedList object.
* Elements in this collection can be accessed using an integer index. Indexes in this collection are zero-based.

Methods and properties of sorted list:

Properties

* Keys; Gets the keys contained in the SortedList object.
* Values; Gets the keys contained in the theSortedList object.
* Item[Object]; Gets or sets the value associated with a specific key in a SortedList object.(object=key).
* Capacity; Gets or sets the capacity of a SortedList object.
* Count; Gets the number of elements contained in a SortedList object.
* IsFixedSize; Gets bool value the SortedList is fixed size.
* IsReadOnly; Gets a value indicating whether a SortedList object is read-only

Methods

* Add(Object, Object)
* Clone()
* Contains(Object)
* ContainsKey(Object)
* ContainsValue(Object)
* IndexOfKey(Object)
* IndexOfValue(Object)

HashTable

* The Hashtable in C# is a collection that stores (Keys, Values) pairs.
* A HashTable is immutable and cannot have duplicate entries.
* Here, the Keys are used to find the storage location.
* A hashtable is a general-purpose dictionary collection.
* A key is an object and a value is an object. These are known as Key/Value pairs.
* When items are added to a hash table, a hash code is generated automatically. This code is hidden from the programmer.
* As the items in the collection are sorted according to the hidden hash code, the items should be considered to be randomly ordered.

Methods and properties of sorted list:

Properties:

* Comparer; Gets or Sets the IComparer to use for the Hash Table.
* Count; Gets the number of key/value pairs contained in the hash table.
* IsReadOnly; Get a value indicating whether the hash table is read-only.
* Item; Gets or Sets the value associated with the specified Key.
* Keys; Gets an ICollection containing the keys in the hash table.
* Values; Gets an ICollection containing the values in the hash table.

Methods:

* Add () : Adds an element with the specified key and value in the hash table.
* Clear() :Removes all the elements in the hash table.
* ContainsKey() :Determined whether the hash table contains a specified key or not.
* ContainsValue() :Determined whether the hash table contains a specified value or not.

Generic collections.

Generics collections in c sharp comes under namespace called “System.Collections.Generics”

Under Generic collections we have collection like.

List<>

LinkedList<>

SortedSet<>

SortedDictionary<>

Dictionary<>

HashSet<>

SortedList<>

Stack<>

All these collections are ready made classes with inbuilt methods and properties.

1.List<>:

* C# List class in .NET provides methods and properties to create a list of objects and data types.
* You can create a list of simple data types such as a string or int or you can create a list of objects such as classes and structs.
* C# List class methods allows you to add, remove, and find items in a list object.

Methods and properties of sorted list:

Methods:

* AddRange();
* Clear();
* Contains("a");
* Contains();
* CopyTo();
* EnsureCapacity(1);
* Exists();
* Find();
* FindAll();
* FindLast();
* IndexOf("a");
* Insert(0, "a");
* Remove();
* RemoveAll();
* RemoveAt();
* Reverse();
* Sort();
* Add();

Properties

* Capacity;
* Count;

2. LinkedList<>:

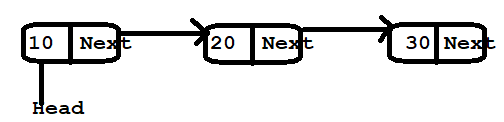
* Linked List is a linear data structure which consists of a group of nodes in a sequence. Each node contains two parts.
* Data− Each node of a linked list can store a data.
* Address − Each node of a linked list contains an address to the next node, called "Next".

Advantages of Linked List:

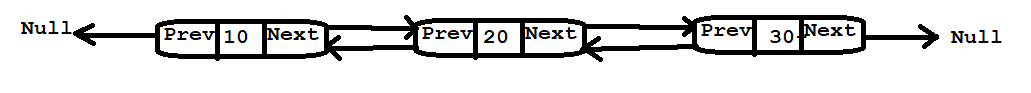
* They are dynamic in nature and allocate memory as and when required.
* Insertion and deletion is easy to implement.
* It has faster access time and can be expanded in constant time without memory overhead.
* Since there is no need to define an initial size for a linked list, hence memory utilization is effective.

Typed Linked List

Singly Linked List: Singly linked lists contain nodes which have a data part and an address part, i.e., Next, which points to the next node in the sequence of nodes. The next pointer of the last node will point to null.

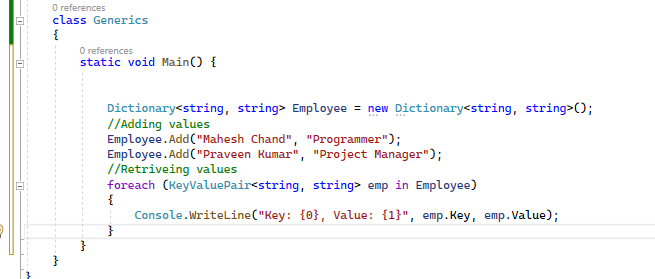


Doubly Linked List: In a doubly linked list, each node contains two links - the first link points to the previous node and the next link points to the next node in the sequence. The previous pointer of the first node and next pointer of the last node will point to null.



3.Dictionary:

C# Dictionary class is a generic collection of keys and values pair of data. The Dictionary class is defined in the System.Collections.A generic namespace is a generic class and can store any data type in a form of keys and values. Each key must be unique in the collection.



4.HashSet:

What is hash set?

HashSet is Unorder Collection which contains unique values to get High performance in C#.net.

* It can store only values not keys like other collections in C#.net.
* HashSet is unorder collection.
* Contain unique values.
* It can allow a single null value.
* It’s part of the System.Collections.Generic namespace.
* Give you high performance as it has unique values.

How to Handle duplicate Element in collections?

Capacity of all Types collection classes (Generic and Non-Generic)

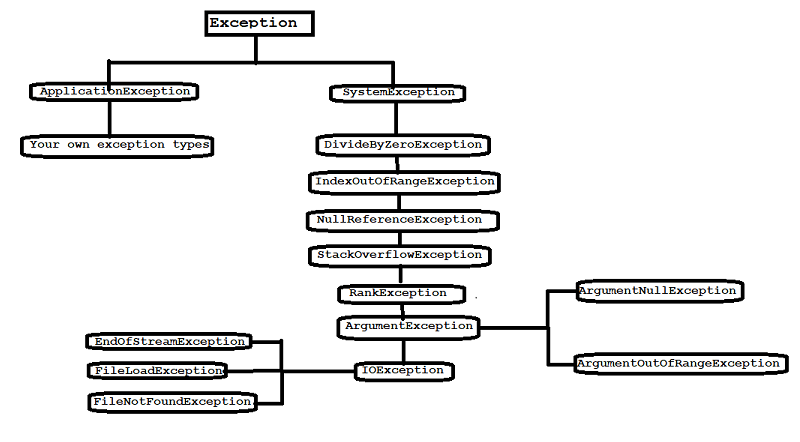
What will happen if we try to add a duplicate element in collections?

Will it throw a runtime exception?

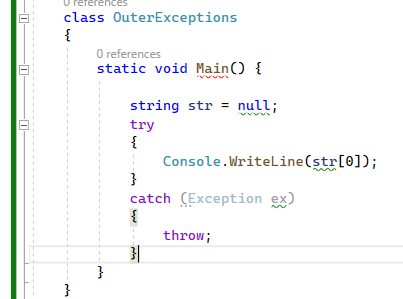
Will it ignore duplicate elements without throwing any exception?

Exception Handling

Exception are nothing but runtime interruption which will stop flow of execution of program.

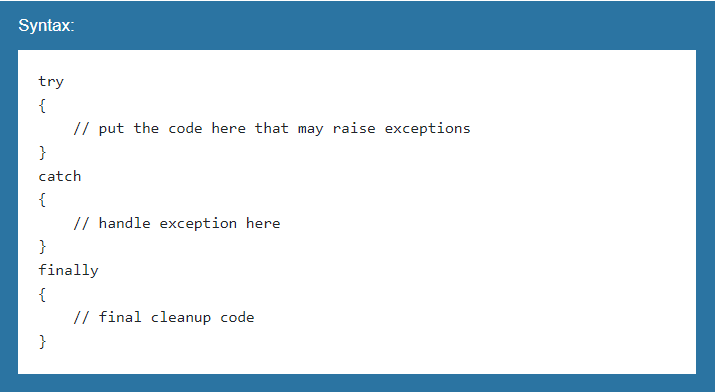


In C Sharp we can also propagate exception by using Throw keyword.



In C Sharp we can handle exception by using try, catch and finally blocks.

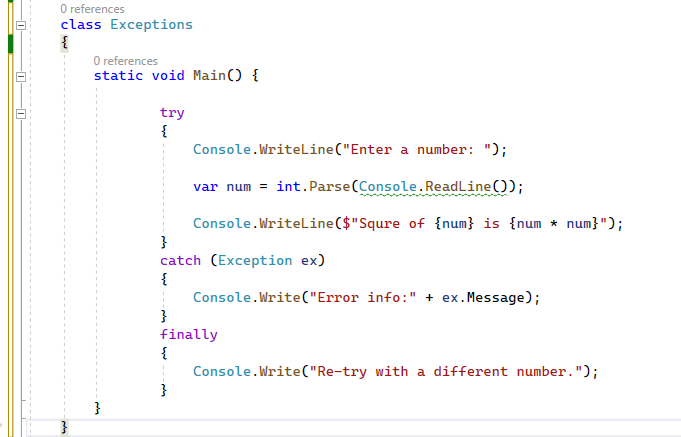
Syntax



Try Block:Any suspected code that may raise exceptions should be put inside a try{ } block.During the execution, if an exception occurs, the flow of the control jumps to the first matching catch block.

Catch Block:The catch block is an exception handler block where you can perform some action such as logging and auditing an exception. The catch block takes a parameter of an exception type using which you can get the details of an exception.

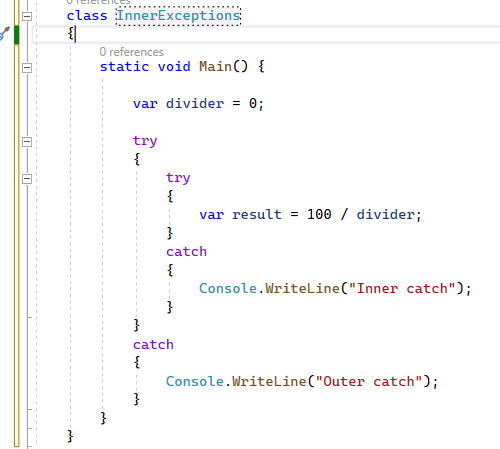
Finally Block:The finally block will always be executed whether an exception raised or not. Usually, a finally block should be used to release resources, e.g., to close any stream or file objects that were opened in the try block.



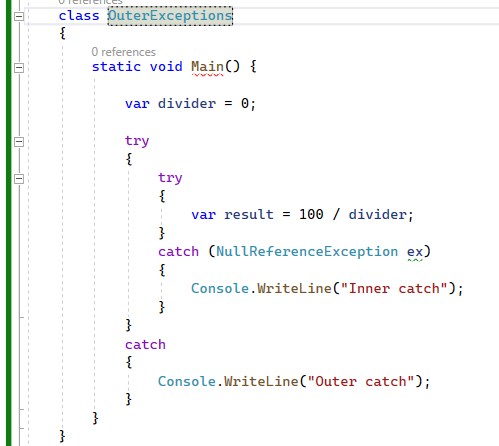
Nested Try-Catch Block.

C# allows nested try-catch blocks. When using nested try-catch blocks, an exception will be caught in the first matching catch block that follows the try block where an exception occurred.

Inner exception .



Outer exception



Custom Exception

C# includes the built-in exception types such NullReferenceException, MemoryOverflowException, etc. However, you often like to raise an exception when the business rule of your application gets violated. So, for this, you can create a custom exception class by deriving the ApplicationException class.

The .Net framework includes ApplicationException class since .Net v1.0. However, Microsoft now recommends Exception class to create a custom exception class.

For example, create InvalidStudentNameException class in a school application, which does not allow any special character or numeric value in a name of any of the students.

