Chapter-1

Advanced C#.net

youtube.com/user/kannababubanna/videos

AutoImplemented Properties:-These Properties are used to Automatically set the value and get the value for private variables

```
using System;
namespace ConsoleApplication4
{
    class Employee
    {
        public int Eno { set; get; }
        public string Ename { get; set; }
        public double  Bsal { get; set; }
    }
    class Program
    {
        static void Main()
        {
            Employee e1 = new Employee();
            e1.Eno = 101;
            e1.Ename = "Anilkumar";
            e1.Bsal = 20000;
            Console.WriteLine(e1.Eno);
            Console.WriteLine(e1.Ename);
            Console.WriteLine(e1.Bsal);
        }
    }
}
```

Object Initializers:-Object Initializers ar used to automatically set the values for Properties at the time of creating object

```
using System;
namespace ConsoleApplication6
{
    class Employee
    {
        public int Eno { get; set; }
        public string Ename { get; set; }
        public double Bsal { get; set; }
        public double Da { get; set; }
        public double Hra { get; set; }
        public double Tsal { get; set; }
    }
    class Program
    {
        static void Main()
        {
            Employee e1 = new Employee
            {
                Eno=101,
                Ename="Anil",
                Bsal=20000
            };
            e1.Da = 0.2 * e1.Bsal;
            e1.Hra = 0.4 * e1.Bsal;
            e1.Tsal = e1.Bsal + e1.Da + e1.Hra;
            Console.WriteLine("Da is"+e1.Da);
            Console.WriteLine("Hra is"+e1.Hra);
            Console.WriteLine("Total salary is"+e1.Tsal);
        }   }  }
```

Ex:-

```
using System;
namespace ConsoleApplication6
{
    class Employee
    {
        public int Eno { get; set; }
        public string Ename { get; set; }
        public double Bsal { get; set; }
        public double Da { get; set; }
        public double Hra { get; set; }
        public double Tsal { get; set; }
    }
    class Program
    {
        static void Main()
        {
            Console.WriteLine("Enter eno");
            int eno = int.Parse(Console.ReadLine());
            Console.WriteLine("Enter ename");
            string ename = Console.ReadLine();
            Console.WriteLine("Enter Bsal");
           double bsal= double.Parse(Console.ReadLine());
            Employee e1 = new Employee
            {
                Eno=eno,    Ename=ename,   Bsal=bsal
            };
```

```
            e1.Da = 0.2 * e1.Bsal;
            e1.Hra = 0.4 * e1.Bsal;
            e1.Tsal = e1.Bsal + e1.Da + e1.Hra;
            Console.WriteLine ("Da is"+e1.Da);
            Console.WriteLine ("Hra is"+e1.Hra);
            Console.WriteLine ("Total salary is"+e1.Tsal);
        }
    }
}
```

Collection Initializers:-Collection initializers are group of object Initializers

```
using System;
using System.Collections.Generic;
namespace ConsoleApplication6
{
    class Employee
    {
        public int Eno { get; set; }
        public string Ename { get; set; }
        public double Bsal { get; set; }
        public double Da { get; set; }
        public double Hra { get; set; }
        public double Tsal { get; set; }
    }
```

```
class Program
  {
      static void Main()
      {
          List<Employee> emps = new List<Employee>()
          {
            new Employee{Eno=101,Ename="Anil",Bsal=20000},
            new  Employee{Eno=102,Ename="sunil",Bsal=30000}
          };
          foreach (var item in emps)
          {
              Console.WriteLine("Empno is"+item.Eno);
              Console.WriteLine("Ename is"+item.Ename);
              Console.WriteLine("Salary is"+item.Bsal);
              Console.WriteLine("Da is"+(0.2*item.Bsal));
              Console.WriteLine("Hra is" + (0.4 * item.Bsal));
Console.WriteLine("Tsal is" + item.Bsal+item.Da+item.Hra);
          }
          Console.ReadLine();
      }
  }
}
```

Implicitly typed local variables:-

we can implement These variables by using var keyword

```
var a=10;
var a="sathya";
var a=2.3;
var a=3.4f;
var a='c';
using System;
namespace ConsoleApplication8
{
    class Program
    {
        static void Main(string[] args)
        {
            var a = 10;
            var a1 = "sathya";
            var a2 = 2.3;
            var a3 = 3.4f;
            var a4 = 'c';
            Console.WriteLine(a);
            Console.WriteLine(a1);
            Console.WriteLine(a2);
            Console.WriteLine(a3);
            Console.WriteLine(a4);
        }
    }
}
```

Always implicitly typed variables must be declared with var keyword and must be declared as local variable

The main Advantage of these variables are:-

1. To handle Anonymous types
2. foreach loop
3. To prepare linq queries

K A N N A B A B U

```
using System;
namespace ConsoleApplication8
{
    class Program
    {
        static void Main(string[] args)
        {
            var a= new int[5];
            a[0] = 10;  a[1] = 20;  a[2] = 30; a[3] = 40;
            foreach (object o1 in a)
            {
                Console.WriteLine(o1);
             }
          }
       }
 }
```

```
using System;
namespace ConsoleApplication8
{
    class A
    {
        public void Show()
        {
            Console.WriteLine("i am show");
        }
    }
    class Program
    {
        static void Main()
        {
            new A().Show();
        }
    }
}
```

K
A
N
N
A
B
A
B
U

```
using System;
```

```
namespace ConsoleApplication8
{
    class A
    {
        public void Show(int x)
        {
            Console.WriteLine("i am show");
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
           new A().Show(10);
        }
     }
     }
```

K A N N A B A B U

Extension methods:-

it is a concept of adding new methods to existing class without applying inheritance

while working with extesnionmethods no need to inherit the original class and no need to modify the original class

Rules :-

1. Extesnion methods should be declared in static class
2. method parameter should be class name with this keyword

```
static returntype methodname (this classname arg)
{
}
```

note:-when we compile the program the compiler will add the extesnion methods to the existing class

3. Extension method can be called by using objectname

generally we will use extsnion methods in linq queries

```
using System;
namespace ConsoleApplication8
{
    class A
    {
        public void Show()
        {
            Console.WriteLine("i am show");
        }
        public void Display()
        {
            Console.WriteLine("i am display");
        }
```

```
        public void Print()
        {
            Console.WriteLine("i am print");
        }
    }
    static class XXX
    {
        public static void Newmethod(this A obj)
        {
            Console.WriteLine ("i am newmethod");
        }
    }
==========================================
    class Program
    {
        static void Main(string[] args)
        {
            A a1 = new A();
            a1.Show();
            a1.Display();
            a1.Print();
            a1.Newmethod();
            Console.ReadLine();
        }
```

Delegates:-Delegate is a Type which is used to hold the information of the method

Delegate is a type safe function pointer which is used to invoke methods

Delegate is a reference pointer to method

Rules to declare Delegates:-

1. Delegate can declare inside the class or outside the class
2. The datatype of the delegate and the datatype of the methods must be same
3. The parameters of the delegate and the parameters of the method both must be same
4. we can create object for the delegate and pass methodname as parameter fordelegate object

steps to work with Delegates:-

1. create a delegate

```
public delegate returntype delegatename(parameters);
```

2. create a method for the delegate

```
public void Show()    {    }
```

3. create an object for the delegate and pass methodname as parameter

```
delegatename objectname=new delegaename();
```

4. call the delegateobject

whenever we invoke the delegate then delegate will invoke the methods

```
delegateobjectname();
```

```
using System;
//create the delegate
public delegate void Mydelegate();
class A
{
    //declare method for the delegate
    public static void Show()
    {
       Console.WriteLine("i am show");
     }
    public static void Display()
    {
        Console.WriteLine("i am display");
    }
    public static  void Print()
    {
         Console.WriteLine("i am Print");
     }
    static void Main()
    {    //create an object for the delegate and pass the methodname
as parameter
        Mydelegate obj=new Mydelegate(Show);
        obj += new Mydelegate(Display);
        obj += new Mydelegate(Print);
        //invoke delegateobject
        obj();
    }
```

```
}

using System;
public delegate void MyDelegate(int x,int y);
class A
{
    static void Add(int x, int y)
    {
        Console.WriteLine("sum is"+(x+y));
    }
    static void Sub(int x, int y)
    {
        Console.WriteLine("Diff is"+(x-y));
    }
    static void Mul(int x, int y)
    {
        Console.WriteLine("product is"+(x*y));
    }
    static void Main()
    {
        MyDelegate obj = new MyDelegate(Add);
        obj+=new MyDelegate(Sub);
        obj+=new MyDelegate(Mul);
        obj(7,5);
    }}
```

Q) What is the difference between method overloading and delegates?

class A
{
public void Show(){}
public void Show(int x){}
public void Show(int x,int y){}
}
	public delegate void MyDelegate();
	class A
	{
	public void Show(){ }
	public void Print(){ }
	}

Method overLoading means it is a process of defining multiple methods with same name
but with different parameters
Delegate's means it is a process of defining multiple methods with different method name
but with same parameters
Delegates are used to invoke all the methods at a time
if we want to let the delegate to call a specific method we have to use Events
Events:-Events are the timeperiods which are used to intimate the delegate that which method must gets executed

```
using System;
//create a delegate
public delegate void MyDelegate();
class A
{
    //create an Event
    //public event delegatename eventname;
    public static  event MyDelegate MyEvent1;
    public static  event MyDelegate MyEvent2;
    // create a method for the delegate
    public static void Show()
    {
        Console.WriteLine("i am show");
    }
    public static void Display()
    {
        Console.WriteLine("i am Display");
    }
    static void Main()
    {
//Event will intimate to delegate that which method must gets executed
        MyEvent1+=new MyDelegate(Show);
        MyEvent2+=new MyDelegate(Display);
       //Invoke the Event
       //eventname();
        MyEvent1();     MyEvent2();
    } }
```

Anonymous Types:-

Anonymous Types means unnown Datatypes
Anonymous Types are also called as implicitly typed local variables
Anonymous Types must declare with var keyword

```
object a=10;          var a=10;
object b=20;          var b=20;
object c=a+b;  error       var c=a+b;   valid
                      var d="abc";
```

var is frequently used in Linq Queries and foreach loop
var must always declare as local variable

==========

Anonymous Methods:-

Method without name is called as Anonymous Method
Anonymous Method is used to simplify the code
Anonymous method is used to add a block of code to the delegate reference

```
using System;
public delegate void MyDelegate(int x,int y);
class A
{
   static void Main()
   {
      MyDelegate obj = delegate(int x, int y)
      {     Console.WriteLine(x + y);       };
       obj(6,5);
   }}
```

```
using System;
public delegate void MyDelegate(int x,int y);
class A
{
    static void Main()
    {
       MyDelegate obj=delegate(int x, int y)
       {      Console.WriteLine("sum is"+(x + y));       };
              obj += delegate(int x, int y)
       {     Console.WriteLine("diff is"+(x - y));        };
               obj(6,5);
    }
}
```

```
using System;
public delegate void MyDelegate(int x,int y);
class A
{
    public static event MyDelegate Event1;
    public static event MyDelegate Event2;
    static void Main()
    {
        Event1 += delegate(int x, int y)
       {
           Console.WriteLine("sum is"+(x+y));
       };
```

```
        Event2 += delegate(int x, int y)
        {
            Console.WriteLine("Diff is"+(x-y));
        };
        Event1(7, 5);   Event2(7, 6);    }   }
```

Lambda Expression:-Lambda Expression is used reduce the amount of coding

At compile time Lambda Expression will convert as Anonymous method

syn:- goesto

```
            i/p parameters => methodbody
    public void Show(int x,int y)
    {       C.WL(x+y);     }
    (int x,int y)=>C.WL(x+y);
```

```
using System;
public delegate void MyDelegate(int x,int y);
class A
{
    public static event MyDelegate Event1;
    public static event MyDelegate Event2;
    static void Main()
    {
        Event1 += (x, y) => Console.WriteLine(x + y);
        Event2 += (x, y) => Console.WriteLine(x-y);
        Event1(7, 5);
        Event2(7, 6);
    }
}
```

```
using System;
public delegate int MyDelegate(int x,int y);
class A
{
    public static event MyDelegate Event1;
    public static event MyDelegate Event2;
    static void Main()
    {
        Event1 += (x, y) => x+y;
        Event2 += (x, y) => x - y;
       int sum=Event1(7, 5);
       Console.WriteLine("sum is"+sum);
     int sub=Event2(7, 5);
     Console.WriteLine("Diff is"+(sub));
   }
}
```

Chapter-2

Collections and Generics

youtube.com/user/kannababubanna/videos

Q) What is a Collection?

- A collection is a data structure that holds a set of objects in a specific manner.
- Collections are used to implement data structures in .net
- Data structure is used to store the data and manipulate the data
- Collection is group of objects
- Collection is a container object which is used to store group of objects
- A Collection is a group of individual objects represented as a single unit

Q) What is Collection framework?

The Collection framework represents a unified architecture for storing and manipulating a group of objects.

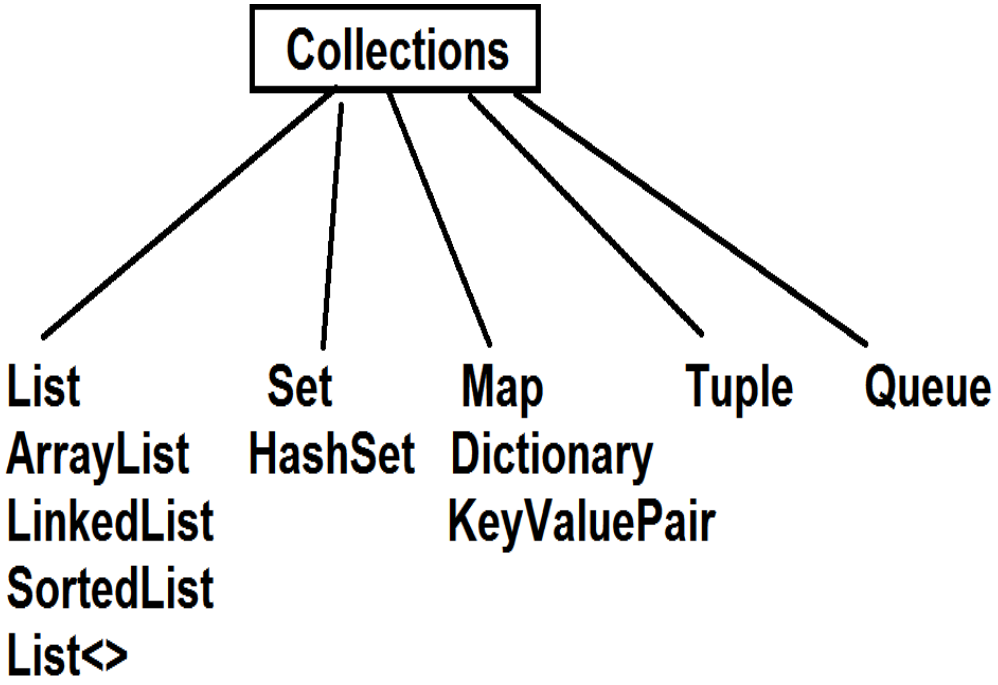
It has Interfaces and its implementation classes

Q) Why to Use Collection Framework?

- Reduce programming effort
- Increase program speed and quality
- Faster software reuse

Q) What is the need of storing group of objects in collection?

1. To perform Different types of operations on group of objects like insertion, Deletion, Updating, searching, sorting etc...
2. if we want to pass group of objects as parameter for a method then use collections
3. Readymade methods are available to perform operations on collections
4. if we want a method to return multiple objects then declare collection as return type of the method



==========

List	Set	Map	Tuple
ordered	ordered insertion	unordered	unordered
Duplicates allowed	Allowed	Not Allowed	Keys--> not Values --> Allowed
null values Allowed	Allowed	Allowed	Key --> not
synchronized	not Syn	Not Syn	Value --> Allowed Not Synchronized

Q) When to use List, Set and Map in .net?

1) If you do not want to have duplicate values in the database then Set should be your first choice as all of its classes do not allow duplicates.

2) If there is a need of frequent search operations based on the index values then List(ArrayList) is a better choice.

3) If there is a need of maintaining the insertion order then also the List is a preferred collection interface.

4) If the requirement is to have the key & value mappings in the database then Map is your best choice

Examples for List:-

1. Store total salary of all employees?
2. Store total marks of all students?
3. Store maths marks of all students?
4. Store 6 subject marks of a specific student?
5. Store list of empnames
6. List of emp salaries

Examples for Set:-

1. store Contact numbers of employees in the company?
2. store Hallticket numbers of students belongs to the college?
3. store product names of a supermarket?
4. Store list of company names of Tvs?
5. store some statenames belongs to india?
6. store citynames of Telangana state?

Examples for Map:-

1. store student results (Hallticketno,percentage)?
2. store Contact numbers along with empname working in the company?
3. store bank accountnos with names?
4. store foodname ,price belongs to a restaurantname?
5. store seat no along with passenger name ?
6. Store branch name with sales amount

7. Store college code with college name

8. Store Token Number with Food Order

9. Store Land line code with area name

10.Store pin code with area name

11. store countrynames along with statenames?

Special Examples:-

1)Store Bus Service numbers with routes information

Ex:- sv1:Hyd, Vij, Raj, Vizag

2)Store country name with state Names

Ex: India :AP, TS, KA, TN,…

3)Store Aadhar Number with person information

Ex:- Aadhar ID : Name, Gender, DOB

4)Store {“India”:{AP: [Vijayawada, Vizag], TS : [Hyd,RR]} }

5)Consider a Customer ordered food items in swiggy like idly, dosa, pongal. Store Order_id along with items

6) A Travel Agency company want to maintain a Customer data like name, phno, email_id, gender,seatno based on TicketNumber

ArrayList:- The ArrayList class is a resizable array

ArrayList will maintain the data in index format

ArrayList will allow duplicate objects

ArrayList will allow null values

Array	**ArrayList**
Array is used to store homogeneous values	ArrayList is used to store heterogeneous values
The size of Array is fixed	The size of ArrayList is not fixed
We cannot increase or decrease the size of Array depending on the requirement	We can increase or decrease the size of Array depending on the requirement
We cannot insert or remove the value from Array at a specific position	We can insert or remove the value from ArrayList at a specific position

Readymade method support is not available in Arrays	Readymade method support is available in Arrays
Searching and sorting operations must done manually	Predefined method support is available for searching and sorting
Array is available under System namespace	ArrayList is available under System.Collections namespace

Q) What is an Interface?

A) Interface is a type which consists of public abstract methods and public abstract properties.

Interface methods must implement in derived class.

We cannot create object for interface but we can create reference for Interface.

Ex:-

```
 interface A
{     void Show();    }
Class X:A
{   public  void Show() {}   }
```

Q) What methods we can access with Interface reference?

A) With Interface reference we can access the implemented methods in derived class.

```
A a1=new X();
a1.Show();
```

Q) **What is Upcasting?**

A) Upcasting means subclass object assigned to super class reference.

Q) **What methods we can access with super class reference?**

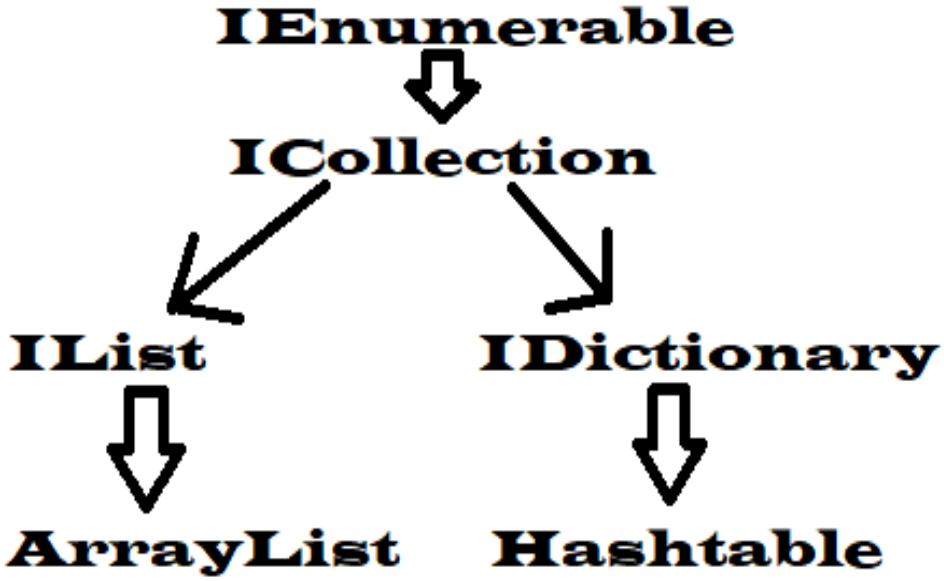
A) With super class reference we can access super class methods and implemented methods in derived class.

With super class reference we cannot access derived class methods.

Q) **What is Downcasting?**

A) Downcasting means super class reference assigned to sub-class reference.

```
B b1=new (B)a1;
b1.Show();
b1.Display();
```




```
Interface IEnumerable
{       IEnumerator GetEnumerator();    }
Interface IEnumerator
{
Bool MoveNext();
void Reset();
object Current { get; }
}
Class ArrayList : IEnumerable
{
 IEnumerator GetEnumerator()
{      return this;}  }
```

K
A
N
N
A
B
A
B
U

GetEnumerator():-

This method will return object of the class which is implementing IEnumerable Interface.

This method will return collection.

MoveNext():-

This method is used to move the pointer to the next position.

Reset():-

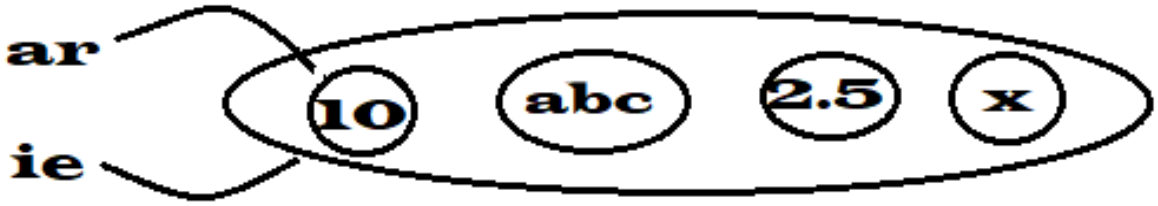
This method is used to reset the pointer to starting position.

Current:-

This property is used to return the object at current position.

```
using System;
usingSystem.Collections;
class A
{
static void Main()
{
ArrayListar = new ArrayList();
ar.Add(10);
ar.Add("abc");
ar.Add(2.5);
ar.Add('x');
IEnumeratorie = ar.GetEnumerator();
```

```
while (ie.MoveNext())
{
object o = ie.Current;
Console.WriteLine(o);
}   }}
```



K
A
N
N
A
B
A
B
U

```
using System;
usingSystem.Collections;
class Employee
{
public int eno;
public string ename;
public Employee(int no, string name)
{
eno = no;
ename = name;
}
}
```

```
class Program
{
static void Main()
{
ArrayListar = new ArrayList();
ar.Add(new Employee(101, "anil"));
ar.Add(new Employee(102, "sunil"));
IEnumeratorie = ar.GetEnumerator();
while (ie.MoveNext())
{
object o = ie.Current;
Employee e1 = (Employee)o;
Console.WriteLine(e1.eno + e1.ename);
}
}
}
class ArrayList : IEnumerable
{
public IEnumerator GetEnumerator() { }
public int Add(object o) { }
public int Remove(object o) { }
public int  Insert(object o) { }
```

```
public int RemoveAt(object o) { }
public int Reverse(object o) { }
public int Sort(object o) { }
public int Clear(object o) { }
}
```

Add() :-

This method is used to add object to the collection.

Remove() :-

This method is used to remove the object from the collection.

Insert() :-

This method is used to insert the object in the collection at a specific position.

RemoveAt() :-

This method is used to remove the object from the collection at a specific position.

Reverse() :-

This method is used to reverse the item from the collection.

Sort() :-

This method is used to sort the item in the collection.

Clear() :-

This method is used clear the items from the collection.

```
using System;
using System.Collections;
class A
{
static void Main()
{
ArrayList ar = new ArrayList();
ar.Add(10);
ar.Add(20);
ar.Add(30);
ar.Add(40);
ar.Add(50);
Console.WriteLine("After Adding");
foreach (var item in ar)
{
Console.WriteLine(item);
}
Console.WriteLine("After Remove");
ar.Remove(30);
foreach (var item in ar)
{   Console.WriteLine(item);  }

Console.WriteLine("After Insert");
ar.Insert(1, 70);
foreach (var item in ar)
{
Console.WriteLine(item);
}
Console.WriteLine("After RemoveAt ");
```

```
ar.RemoveAt(4);
foreach (var item in ar)
{
Console.WriteLine(item);
}
Console.WriteLine("After Reverse");
ar.Reverse();
foreach (var item in ar)
{
Console.WriteLine(item);
}
Console.WriteLine("After Sort");
ar.Sort();
foreach (var item in ar)
{
Console.WriteLine(item);
}
Console.WriteLine("After Clear");
ar.Clear();
foreach (var item in ar)
{
Console.WriteLine(item);
}
Console.ReadLine();
}
}
```

K
A
N
N
A
B
A
B
U

Stack:-

Stack follows LIFO process

LIFO: - Last in First Out

Methods:-

1) **Push() :-**

This method is used to push the items in stack.

2) **Pop() :-**

This method is used to remove the items from stack.

3) **Clear() :-**

This method is used to removes all objects from the stack.

```
using System;
usingSystem.Collections;
class A
{
static void Main()
{
Stack s = new Stack();
s.Push(10);
s.Push(20);
s.Push(30);
s.Push(40);
```

```
Console.WriteLine("After Push");
foreach (var item in s)
{    Console.WriteLine(item);   }
Console.WriteLine("After Pop");
s.Pop();
foreach (var item in s)
{
Console.WriteLine(item);
}
Console.ReadLine();
}}
```

Queue:-

Queue follows FIFO process

1) **Enqueue() :-**

   This method is used to add the items in queue.

2) **Dequeue() :-**

   This method is used to remove the items from queue.

3) **Clear() :-**

   This method is used to removes all objects from queue.

Q) Differences between Structure and Class?

	Structure	Class
1.	Must declare with struct keyword	Must declare with class keyword
2.	Structure is valuetype datatype	Class is Referencetype datatype
3.	Memory is allocated on stack	Memory is allocated on heap
4.	Structure is recomended to store small amount of data	Class is recomended to store large amount of data
5.	Structures are inherited from System.Value Type	Classes are inherited from System.Object Type
6.	new keyword is optional to create object	new keyword is manadatory to create object
7.	Structure doesnot support Default constructor	Class will have Default constructor
8.	Structures doesnot	Supports Destructors

	support Destructors	
9.	Static constructors will not reflect in structures	supports static constructor
10.	Structure cannot be inherited to other type	Class can inherit to other class
11.	Structure cannot be declared as abstract	Must declare as abstract

Hash Table:-

Hash Table will maintain data in key and value pair format.

Key must not be duplicate.

Hash Table will display the output in random manner.

Hash Table will maintain the data DictionaryEntry Format.

DictionaryEntry is predefined structure.

```
struct  DictionaryEntry
{
public object Key{set;get;}
public object Value{set;get;}
}
```

```
using System;
using System.Collections;
class A
{
static void Main()
{
Hashtableht=new Hashtable();
ht.Add(7,"C");
ht.Add(1,"C++");
ht.Add(2,".Net");
ht.Add(3,"Java");
foreach (DictionaryEntry item in ht)
{
Console.WriteLine(item.Key+" "+item.Value);
}
}
}
```


SortedList() :-

SortedList will arrange the data in sorted order.

Generics:- The Generics **are used to introduce to deal with type-safe objects. It makes the code stable by detecting the bugs at compile time.**

Before generics, we can store any type of objects in the collection, i.e., non-generic. Now generics force the C# programmer to store a specific type of objects.

Advantage of Generics

There are mainly 3 advantages of generics. They are as follows:

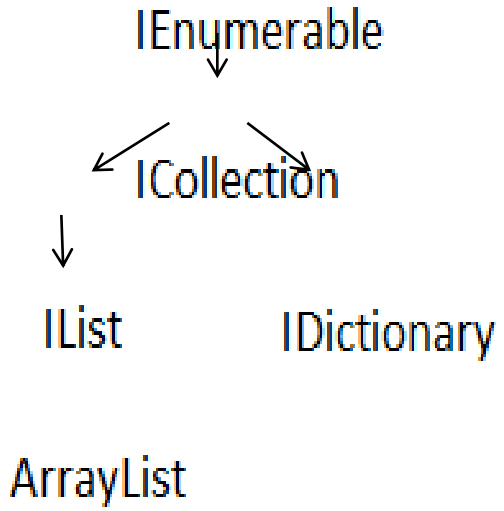
- Generics are called as general datatype.
- Generics are used to avoid unnecessary typecasting like Boxing, UnBoxing, Widening, Narrowing,Upcasting,DownCasting, etc..,
- Generics are used to avoid overloading.
- In order to work with generics MicroSoft has given a predefined namespace “System.Collections.Generic”.
- Generics can be declared by using Placeholder and Typeparameater,Placeholder<>,Typeparameater ()

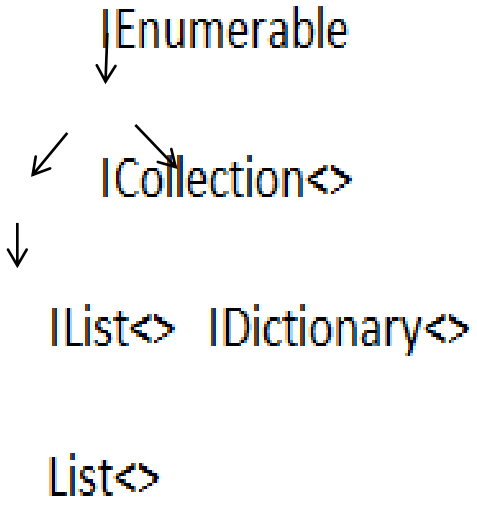
```
using System;
using System.Collections.Generic;
class A
{
static void Display<Tp>(Tp x)
{
Console.WriteLine(x);
}
static void Main()
{
Display<int>(10);
Display<string>("sathya");
Display<double>(10.5);
}
}
```

Generics do not support Arithmetic operators.

Q) Difference between Collections and Generics?

Collections	**Generics**
Whenever we want to perform operations on group of objects of any type we use Collections	Whenever we want to perform operations on group of objects of same type we use Generics
Collections support Typecasting	Generics does not support Typecasting
Collections support Arthimetic Operations by using Typecasting	Doesnot support Arthimetic Operations by using Typecasting






```
using System;
using System.Collections.Generic;
class Employee
{
public int eno;
public string ename;
public Employee(int no, string name)
{
eno = no;
ename = name;
}
}
```

K
A
N
N
A
B
A
B
U

```
class Program
{
static void Main()
{
List<Employee> li = new List<Employee>();
li.Add(new Employee(101, "anil"));
li.Add(new Employee(102, "sunil"));
li.Add(new Employee(103, "raju"));
IEnumerator<Employee>ie = li.GetEnumerator();
while(ie.MoveNext())
{
Employee e1 = ie.Current;
Console.WriteLine(e1.eno + e1.ename);
}
}
}
```

K A N N A B A B U

HashSet:-

HashSet class is used to create a collection that uses a hash table for storage. It inherits the AbstractSet class and implements Set interface.

The important points about C#.net HashSet class are:

- HashSet stores the elements by using a mechanism called hashing.
- HashSet contains unique elements only.
- HashSet allows null value.
- HashSet class is non synchronized.
- HashSet doesn't maintain the insertion order. Here, elements are inserted on the basis of their hashcode.
- HashSet is the best approach for search operations.
- The initial default capacity of HashSet is 16, and the load factor is 0.75.

```
using System;
using System.Collections.Generic;
 class GFG {
    // Driver code
    public static void Main()
    {       // Creating a HashSet of odd numbers
        HashSet<int> odd = new HashSet<int>();

        // Inserting elements in HashSet
        for (int i = 0; i < 5; i++) {
            odd.Add(2 * i + 1);
        }
        // Displaying the elements in the HashSet
        foreach(int i in odd)
        {
            Console.WriteLine(i);
        }
    }
}
```

Ex:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace HashSetDemo {
    class Program {
        static void Main(string[] args) {
 HashSet < string > names = new HashSet < string > {
                "Rajeev",
                "Akash",
                "Amit"
            };
            foreach(var name in names) {
                Console.WriteLine(name);
            }
            Console.ReadKey();
        }
    }
}
```

K A N N A B A B U

Dictionary:- Dictionary is used to store Group of objects in the form of key and value pair formats where cannot be duplicated and value can be duplicated.

Dictionary<TKey, TValue> collection in C# is same as English dictionary. English dictionary is a collection of words and their definitions, often listed alphabetically in one or more specific languages. In the same way, the Dictionary in C# is a collection of Keys and Values, where key is like word and value is like definition.

The Dictionary<TKey, TValue> class is a generic collection class in the System.Collection.Generics namespace. TKey denotes the type of key and TValue is the type of TValue.

```
using System;
using System.Collections.Generic;
public class Demo {
  public static void Main() {
 IDictionary<int, int> d = new Dictionary<int, int>();
    d.Add(1,97);
    d.Add(2,89);
    d.Add(3,77);
    d.Add(4,88);
```

```
    // Dictionary elements
Console.WriteLine("Dictionaly elements:”+d.Count);
  }
}
```

Example: Access Elements using for Loop

```
Dictionary<int, string> dict = new Dictionary<int, string>()
{
  {1,"One"},
  {2, "Two"},
  {3,"Three"}
};
for(int i = 0; i < dict.Count; i++)
{
    Console.WriteLine("Key: {0}, Value: {1}",
    dict.Keys.ElementAt(i),
    dict[ dict.Keys.ElementAt(i)]);
}
```

Ex:-

```
using System;
using System.Collections.Generic;
namespace DictionaryDemo
{
  class Example
  {
     static void Main(string[] args)
     {
       Dictionary d = new Dictionary();
         d.Add(1,"Harry");
         d.Add(2,"Sally");
         d.Add(3,"Clarke");
         d.Add(4,"James");
         d.Add(5,"Emma");
         d.Add(6,"Susan");
       Console.WriteLine("Original dictionary elements:");
       foreach (KeyValuePair i in d)
       {    Console.WriteLine("Key: {0}     Value: {1}", i.Key, i.Val        }
       d.Remove(3);
       d.Remove(6);
       Console.WriteLine("Dictionary elements after deletion:");
```

```
        foreach (KeyValuePair i in d)
        {
            Console.WriteLine("Key: {0}     Value: {1}", i.Key, i.Value);
        }
    }
  }
}
```

1. create a collection to store list of mobile companynames?
2. create a collection to store list of mobile networktypes?
3. create a collection to store mobilemodelno,price?
4. create a collection to store features of mobile along with modelno?
5. waq to display mobile companynames?
6. waq to display mobile companynames starts with s?
7. waq to display the no of mobile companys ?
8. waq to display the no of mobile network types ?
9. waq to display mobilemodelno,price?
10. waq to display mobilemodelno whose price>10000?

K
A
N
N
A
B
A
B
U

11.waq to display mobilemodelno whose price bewteen 10000 and 20000?

12.waq to display highest price mobilemodelno ?

13.waq to display least pricemobilemodelno ?

14.waq to display mobile models having HdRecording feature?

15.create a collection to store mobile modelno,screensize,ramsize,brand,price?

Q)waq to display mobile details based on Brand and ramsize?

16.waq to display mobile modelnos with screensize,camera?

17. waq to display mobile modelnos which is having highest screensize?

18. waq to display mobile modelnos which is having lowest screensize?

19. waq to display mobile modelno which is having 21MP?

20. waq to display mobilemodelno,camera,screensize?

K
A
N
N
A
B
A
B
U

Chapter-3

LINQ

youtube.com/user/kannababubanna/videos

ORM: - Object Relation mapping

Generally in Realtime applications we can transfer the data in 2 formats

1. Text format
2. object format

Text format means data is stored in variable

object format means data is stored in object

ADo.net will interact with Database server in plain text format

ADo.net cannot transfer the data in object format

Transferring the data in the form of object format is a Design pattern DTO (Data Transfer Object)

Q) what is Design Pattern?

Design pattern is a readymade solution for already Existing problem

Q)what is ORM?

ORM is a tool which is used to Transfer the data in the form of objects

ORM means Object Relation mapping

ADO.net	**ORM**
Activex Database object . network enable Technology	Object relation Mapping Language Integrated query
ADo.net will transfer the data in plain text format	ORM will transfer the data in object format
ADO.net will interact with database server through sql query	ORM will interact with database with LINQ Query

ADO.net will interact with databaseserver everytime to perform operations	ORm will reducethe no of trips to interact withdatabase server
if we develop an appn using ADO.net to connect with Sqlserver database later if the client wants to change the database from sqlserver to oracle we need to change the ADo.net code	LINQ query can communicate withany type of database
in ADO.net after developing the application if the structure of of the table is modified then ADO.net code will not work again the developer must modify the code	ORM will directlygenerate the codewhen the structureof table ismodified lly when the code is modified then ORM willautomatically add the columns
ADO.net will Directly display database errormessages to Frontend appn	ORM will not display

in ADO.net we have to write seperate code for connection automatcially	ORM will handleconnectionpooling pooling
Opening and closing of connection is required	ORm will take care
Performance is poor	Good
Runtime syntax checking of sql queries	compiletime syntax checking
Not Type safe	Type safe
No intelligent Support	Intellisense support is available
Debugging of sql statements is not possible	Debugging of Linq query is poosible
code is combination of object oreinetd and Relation	complete object orineted code

Different Types of ORM Tools are:-

1. LINQ (Language integrated query)
   + Linq To Objects
   +Linq To SQL
   +Linq To XML
2. ADO.net Entity Framework
3. Hibernate(Java)
4. NHibernate

Q)what is Query?

query is an Expression which is used to retrieve the data from data source

Steps to work with LINQ:-

1. prepare the datasource
2. write a query to fetch the data from Datasource
3. Execute the query

syn to write Linq query:-

```
from variablename in datasource
select variablename;
```

Linq Query consists of 2 parts

1. from clause
2. select command

Here from clause is used to specify the datasource

select command is used to fetch the data from datasource

Linq To Objects:- it is used to querying the data from a Datasource like Arrays,Collections

Q)what is a Query?

A query is an Expression that retrieve the datya from a datasource

generally Querys are represented in Query Language
in order to work with LINQ we need to follow 3 steps:-

1. Prepare Datasource
2. Prepare the Linq Query
3. Execute the query

in order to write the Linq Query we have to use 2 clauses

1. from clause
2. select clause

syn to write Linq Query:-

```
var query= from  variablename in datasource
                 select variablename;
```

we can execute Linq Query by using foreach loop

```
using System;
using System.Linq;
class A
{
    static void Main(string[] args)
    {
        //prepare datasource
        int[] Ar = new int[6] { 2, 3, 4, 5 ,6,10};
        //prepare the linq query
        var q1 = from x in Ar
                 select x;        //Execute the query
        Console.WriteLine("Q1. waq to Display the values");
        foreach (var item in q1)
        {
            Console.WriteLine(item);
        }
        Console.WriteLine("Q2. waq to display Even nos");
```

```
var q2 = from x in Ar
            where x % 2 == 0
             select x;
foreach (var item in q2)
{           Console.WriteLine(item);       }

Console.WriteLine("Q3. Waq to Display Odd nos");
var q3 = from x in Ar
             where x % 2 != 0
             select x;
foreach (var item in q3)
{
    Console.WriteLine(item);
}
 Console.WriteLine("Q4.  waq to dispaly the value whose value is
5");
var q4 = from x in Ar
            where x == 5
            select x;
foreach (var item in q4)
{      Console.WriteLine(item);       }
Console.WriteLine("Q5. waq to display Array elements in
Ascending order");
var q5=from x in Ar
           orderby x
            select x;
```

```
            foreach (var item in q5)
            {       Console.WriteLine(item);        }
            Console.WriteLine("Q6. waq to display Array elements in
            Descending order");
            var q6 = from x in Ar
                    orderby x descending
                    select x;
            foreach (var item in q6)
            {
                Console.WriteLine(item);
            }
        }
    }
```

Ex:-

```
using System;
using System.Collections.Generic;
using System.Linq;
class Employee
{
    public int Eno { get; set; }
    public string Ename { get; set; }
    public int Dno { get; set; }
    public string Dname { get; set; }
    public string Designation { get; set; }
    public double Salary { get; set; }
}
```

```
class Program
{
    static void Main()
    {
        //prepare datasource
        List<Employee> emps = new List<Employee>()
        {
            new
Employee{Eno=101,Ename="anil",Dno=10,Dname="ECE",Designati
on="Asstprof",Salary=20000},
            new
Employee{Eno=102,Ename="sunil",Dno=10,Dname="ECE",Designa
tion="Asstprof",Salary=23000},
            new
Employee{Eno=103,Ename="ajay",Dno=20,Dname="EEE",Designati
on="Asstprof",Salary=24000},
            new
Employee{Eno=104,Ename="john",Dno=20,Dname="EEE",Designat
ion="Asstprof",Salary=20000},
            new
Employee{Eno=105,Ename="james",Dno=10,Dname="ECE",Design
ation="Prof",Salary=21000},
            new
Employee{Eno=106,Ename="ram",Dno=10,Dname="ECE",Designati
on="Hod",Salary=2000},
        };
        Console.WriteLine("Q1. waq to display emp details");
        var q1 = from x in emps
```

```
                    select x;
        foreach (var item in q1)
        {
Console.WriteLine("Eno"+"\t"+"Ename"+"\t"+"Dno"+"\t"+"Salary"+"\t"
+"Doj"+"\t"+"Desig"+"\t"+"Dname");
            Console.Write(item.Eno + "\t");
            Console.Write(item.Ename + "\t");
            Console.Write(item.Dno + "\t");
            Console.Write(item.Salary + "\t");
            Console.Write(item.Designation + "\t");
            Console.Write(item.Dname + "\t");
        }
        Console.WriteLine("Q waq to display eno,ename,salary who
are woking in dno 10");
        var q2 = from x in emps
                where x.Dno == 10
                select x;
        foreach (var item in q2)
        {

Console.WriteLine(item.Eno+"\t"+item.Ename+"\t"+item.Salary);
        }
        Console.WriteLine("waq tro Display Emp Details whose
        sal>20000");
        var q3 = from x in emps
                where x.Salary > 20000
                select x;
        foreach (var item in q3)
        {
```

```
            Console.WriteLine(item.Eno + "\t" + item.Ename + "\t" +
          item.Salary);
          }
          Console.WriteLine("Q. display emp details who are woreking
      in dno 10 and salary>20000");
          var q4 = from x in emps
                 where x.Salary > 20000 && x.Dno == 10
                 select x;

          foreach (var item in q4)
          {
        Console.WriteLine(item.Eno+"\t"+item.Ename+"\t"+item.Salary);
          }
        }
      }
```

//order by clause

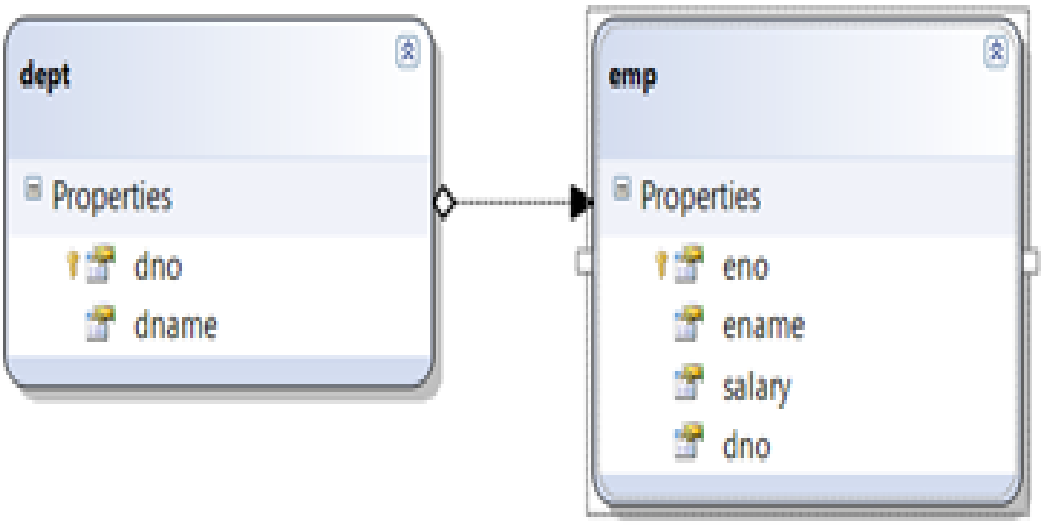
```
      //waq to sort the data based on salary in ascending order
      Console.WriteLine("==========================");
      Console.WriteLine("/waq to sort the data based on salary in
ascending order");
        var r6 = from x in emplist
               orderby x.Salary
               select x;
      foreach (var item in r6)
      {
          Console.WriteLine(item.Empid + "\t" + item.Ename + "\t" +
item.Designation +        "\t" + item.Salary + "\t" + item.Doj);
      }
      Console.WriteLine("==========================");
```

```
        Console.WriteLine("/waq to sort the data based on salary in
descending order");
        var r7= from x in emplist
                orderby x.Salary descending
                select x;
        foreach (var item in r7)
        {
            Console.WriteLine(item.Empid + "\t" + item.Ename + "\t" +
item.Designation +        "\t" + item.Salary + "\t" + item.Doj);        }
      Console.WriteLine("======");
        Console.WriteLine("/waq to display employees detils who are
working in dno 10          order by salary in asc order");
        var r8 = from x in emplist where x.Dno==10
                orderby x.Salary
                select x;
        foreach (var item in r8)
        {
            Console.WriteLine(item.Empid + "\t" + item.Ename + "\t" +
item.Designation +        "\t" + item.Salary + "\t" + item.Doj);
        }
        Console.WriteLine("============================");
        Console.WriteLine("Q)waq to display empid,ename,salary");
        var r9 = from x in emplist
                select new { x.Empid, x.Ename, x.Salary };
        foreach (var item in r9)
        {
            Console.WriteLine(item.Empid + "\t" + item.Ename +
"\t"+item.Salary);
        }
```

```
        Console.WriteLine("========================");
        Console.WriteLine("Q)waq to display the top 3 salaries of
employees");
        var r10 = (from x in emplist orderby
                    x.Salary descending select x).Take(3);
        foreach (var item in r10)
        {
            Console.WriteLine(item.Empid + "\t" + item.Ename + "\t" +
item.Designation +        "\t" + item.Salary + "\t" + item.Doj);
        }
       Console.WriteLine("========================");
        Console.WriteLine("Q)waq to display the top 2 salaries of
employees");
        var r11 = from x in emplist
                  orderby x.Salary descending
                  select x;
        var r12 = r11.Take(2);
        foreach (var item in r12)
        {
            Console.WriteLine(item.Empid + "\t" + item.Ename + "\t" +
item.Designation +        "\t" + item.Salary + "\t" + item.Doj);
        }
        Console.WriteLine("==================================");
        Console.WriteLine("Q)waq to display the top employee details
except top 2 salaries     of employees");
        var r13 = from x in emplist
                  orderby x.Salary descending
                  select x;
        var r14 = r13.Skip(2);
```

```
        foreach (var item in r14)
        {
            Console.WriteLine(item.Empid + "\t" + item.Ename + "\t" +
item.Designation +        "\t" + item.Salary + "\t" + item.Doj);
        }
       Console.WriteLine("========================");
        Console.WriteLine("Q)waq to display 4th max salary");
        var r15 = (from x in emplist
                   orderby
                       x.Salary descending
                   select x).Take(4).Skip(3);
        foreach (var item in r15)
        {
            Console.WriteLine(item.Empid + "\t" + item.Ename + "\t" +
item.Designation +        "\t" + item.Salary + "\t" + item.Doj);
        }
        //waq to display first maxsalary of an employee working in dno 10
        var r16 = (from x in emplist
                   orderby x.Salary descending
                   select x).First();
        foreach (var item in r16)
        {
            Console.WriteLine(item.Empid + "\t" + item.Ename + "\t" +
item.Designation +        "\t" + item.Salary + "\t" + item.Doj);
        }
```

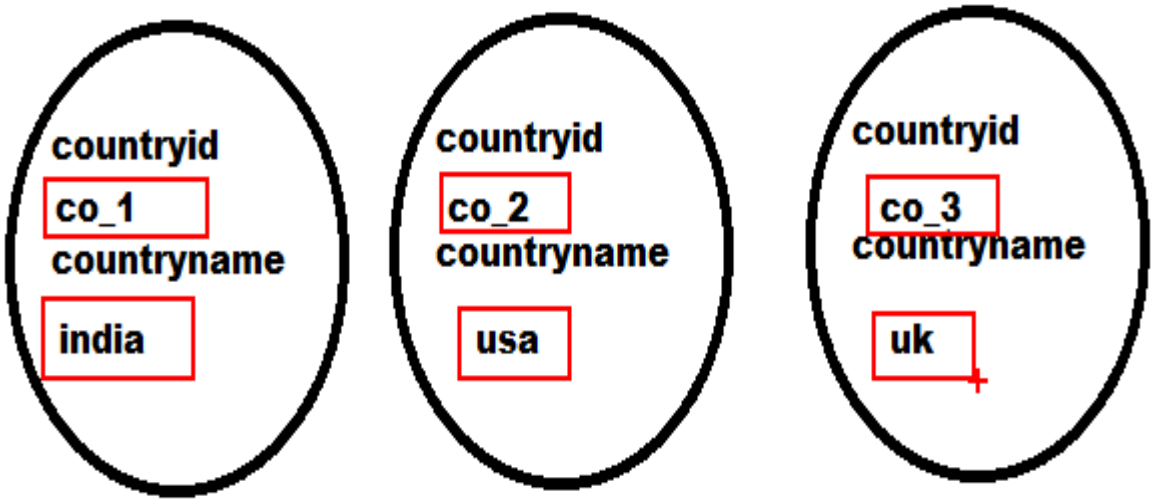
```
using System;
using System.Collections.Generic;
using System.Linq;
namespace WebApplication14
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      DemoDataContext obj = new DemoDataContext();
      protected void Page_Load(object sender, EventArgs e)
      {
         var q1=from emps in obj.emps
                 join depts in obj.depts
                 on emps.dno equals depts.dno
                 select new{
                              emps.eno,
                              emps.ename,
                              emps.salary,
                              depts.dname
                            };
         GridView1.DataSource = q1;
         GridView1.DataBind();
      }
   }}
```



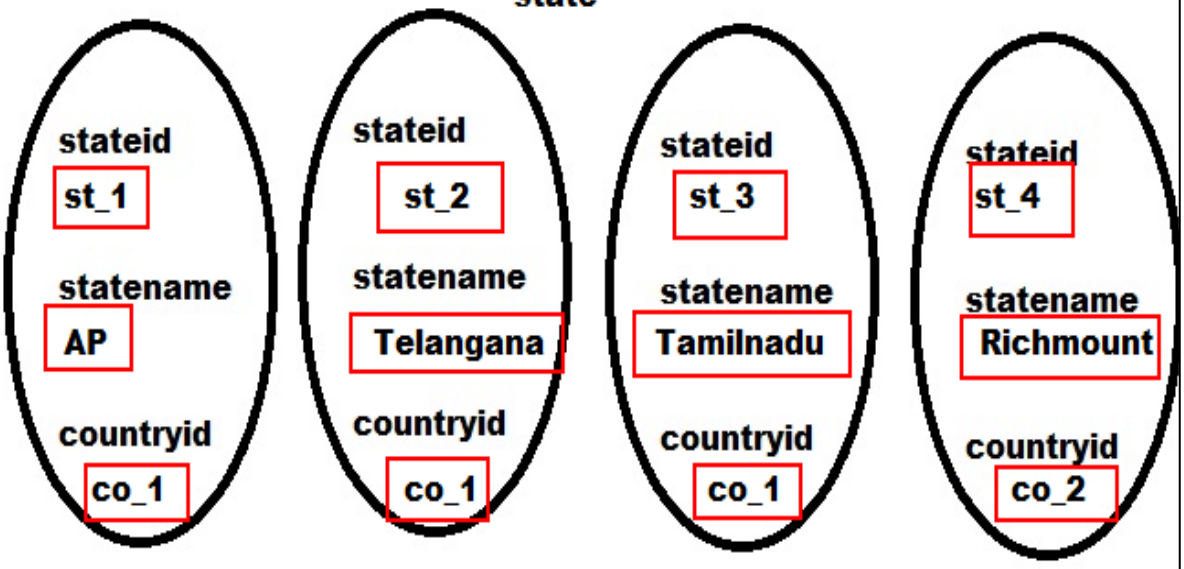
Autogenerateid in LINQ:-

```
protected void Page_Load()
{
    if(IsPostBack==false)
    {       int y=0;
    string eno="eno_";
    EmployeeDataContext obj=new EmployeeDataContext();
    employee e1=new employee();
    var i=(from x in obj.employees
            orderby x.id descending
            select x).Take(1);
    foreach(var item in i)
    {     y=item.id;     }
    y++;
    eno=eno+y;
    e1.eno=eno;
    TextBox1.Text =e1.eno;          }       }
```

country



state



K A N N A B A B U

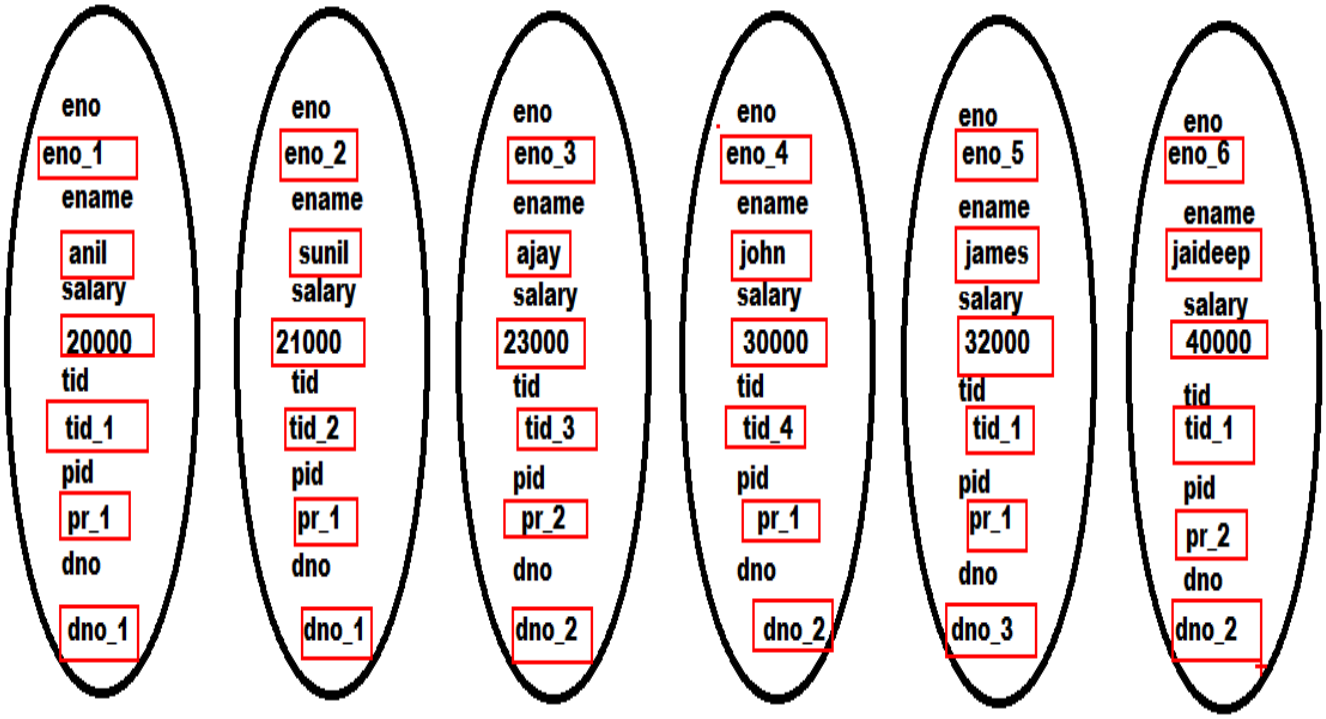
city

cityid	cityname	stateid
ci_1	Hyderabad	st_2
ci_2	Vizag	st_1
ci_3	vijayawada	st_1
ci_4	chennai	st_3
ci_5	releign	st_4

dept

dno	dname
dno_1	Technical
dno_2	Hr
dno_3	marketing

employee



1. Create a Collection with name country with countryid,countryname?
2. Create a Collection with name state with stateid,statename, countryid?
3. Create a Collection with name city with cityid,cityname, stateid?
4. Create a Collection with name dept with dno,dname?
5. Create a Collection with name emp with eno,ename,salary,tid,pid,dno?
6. Waq to diplay Countries?
7. Waq to display countrynames?
8. Waq to display statenames starts with H ?
9. Waq to display statenames belongs to AP?

10. Waq to display statenames belongs to Telangana and whose cityname starts with H?
11. Waq to display Dept details?
12. Waq to display Emp details?
13. Waq to display enames starts with a?
14. Waq to display emp detyails whose salary>210000?
15. Waq to display emp details whose dno=dno_1?
16. Waq to display Emp namesa whose salary>20000 and who is working in Technical?
17. Waq to display emps working in eno_2?
18. Waq to display emps whp are working in dno_1?
19. Waq to display Dept details?
20. waq to display Enames along with Dname?

Joins in LINQ

```
using System;
using System.Linq;
using System.Collections.Generic;
class Dept
{
    public int Dno { get; set; }
    public string Dname { get; set; }
}
class Emp
{
    public int Eno { get; set; }
    public string Ename { get; set; }
    public double Salary { get; set; }
    public int Dno { get; set; }
}
```

```
class Program
{
    static void Main()
    {
        List<Dept> depts = new List<Dept>()
        {
        new Dept(){Dno=10,Dname="ECE"},
        new Dept(){Dno=20,Dname="CSE"}
        };
        List<Emp> emps = new List<Emp>()
        {
        new Emp(){Eno=101,Ename="Anil",Salary=20000,Dno=10},
        new Emp(){Eno=102,Ename="Sunil",Salary=2000,Dno=10},
        new Emp(){Eno=103,Ename="Ajay",Salary=20000,Dno=20},
        };
        Console.WriteLine("Q1 waq to display Emp details along with
Dname");
        Console.WriteLine("Eno"+"\t"+"Ename"+"\t"+"Salary"+"\t"+"Dname");
        var q = (from d in depts
                 join e in emps
                 on d.Dno equals e.Dno
                 select new
                 {
                     e.Eno,
                     e.Ename,
                     e.Salary,
                     d.Dname
                 });
        foreach (var item in q)
        {
            Console.WriteLine(item.Eno + "\t" + item.Ename + "\t" +
item.Salary + "\t" + item.Dname);
        }
```

```
        Console.WriteLine();
        Console.WriteLine("Q2 waq to display Emp detailswho are working
in ECE Dept");
        Console.WriteLine("Eno" + "\t" + "Ename" + "\t" + "Salary" + "\t" +
"Dname");
        var q1 = (from d in depts
                  join e in emps
                  on d.Dno equals e.Dno
                  where d.Dname=="ECE"
                  select new
                  {
                     e.Eno,   e.Ename,  e.Salary, d.Dname
                  });

        foreach (var item in q1)
        {
            Console.WriteLine(item.Eno + "\t" + item.Ename + "\t" +
item.Salary + "\t" + item.Dname);
        }
        Console.WriteLine("Q3 waq to display Emp detailswho are working
in ECE Dept and whose sal>20000");
        Console.WriteLine("Eno" + "\t" + "Ename" + "\t" + "Salary" + "\t" +
"Dname");
        var q3 = (from d in depts
                   join e in emps
                   on d.Dno equals e.Dno
                   where d.Dname == "ECE" && e.Salary>=20000
                   select new
                   {
                      e.Eno, e.Ename, e.Salary, d.Dname              });
```

```
        foreach (var item in q3)
        {
            Console.WriteLine(item.Eno + "\t" + item.Ename + "\t" +
item.Salary + "\t" + item.Dname);
        }
        Console.WriteLine("Q4 waq to display Emp details whose are
working in ECE Dept and whose name starts with a");
        Console.WriteLine("Eno" + "\t" + "Ename" + "\t" + "Salary" + "\t" +
"Dname");
        var q4 = (from d in depts
                  join e in emps
                  on d.Dno equals e.Dno
                  where d.Dname == "ECE" && e.Salary >= 20000
                 && e.Ename.StartsWith("A")
                  select new
                  {
                      e.Eno,
                      e.Ename,
                      e.Salary,
                      d.Dname
                  });
        foreach (var item in q4)
        {
            Console.WriteLine(item.Eno + "\t" + item.Ename + "\t" +
item.Salary + "\t" + item.Dname);
        }
    }
}
```

Chapter-4

Element Operators in LINQ

youtube.com/user/kannababubanna/videos

Element Operators in LINQ:-

1. First or FirstOrDefault

2. Last or LastOrDefault

3. Single or SingleOrDefault

Element Operators are used to return a single Element from the sequence based on Index or based on condition

1**. First():-** This method is used to return the first element from the given sequence

```
using System;

using System.Linq;

class A
{
    static void Main()
    {
        int[] Ar=new int[4]{3,2,1,5};
        int r = Ar.First();
        Console.WriteLine(r);
    }
}
```

note:- if the sequence doesnot consist of any Element then First() will raise an Exception

```
using System;
```

```
using System.Linq;
class A
{
    static void Main()
    {
        int[] Ar=new int[5]{3,7,4,6,9};
        int r = Ar.First(x => x % 2 == 0);
        Console.WriteLine(r);
    }
}
```

FirstOrDefault():- This method is used to return the first element in the given sequence if element doesnot exist it will return default value

Ex:-

```
int[] Ar=new int[]{};

int r = Ar.FirstOrDefault();

Console.WriteLine(r);     o/p:- 0
```

Last():- This method is used to return the last element from the given sequence

Last(lambaexpression)

Ex:- int r = Ar.Last(x=>x%2!=0);

Single():- This method is used to get the single element or single instance from the given sequence

it will not support duplicate values

```
int[] Ar=new int[4]{2,7,3,5};
```

Single() will raise error if no elements in the sequence

```
int r = Ar.SingleOrDefault(x=>x%2==0);
Console.WriteLine(r);
```

Chapter-5

Assemblies and Access Modifiers

youtube.com/user/kannababubanna/videos

Assembly:-

Assembly is the compiled format of any .Net program which may be .dll or .exe

.exe:-

Direct Executable File

.dll:-

Dynamic Link Library

C#.net→.cs→csc→.exe or .dll

Q)what is the difference between Assembly and Namespace ?

Sol: Assembly is collection of namespaces and namespace is collection of classes.

Assembly→.dll or .exe→namespaces→classes→methods→code

System.dll→system→Console→Write() & WriteLine()

System.Data.SqlClient.dll → dll file

System.Data.SqlClient → Namespace

SqlConnectionSqlCommandSqlDataReader → Class

Open() Close() ExecuteNonQuery()

ExecuteScalar() methods

ExecuteReader()

Assemblies are of 3 types:-

1.Private Assembly

2.Public Assembly or Shared Assembly

3.Satellite Assembly

Private Assembly:-The assembly that was specific for a single appn or a specific folder is called as private assembly.

Private assembly is also called as folder specific assembly i.e. a separate copy of dll file was copied into each and query folder.

We can create dll files by using classlibrary template.

Steps to work with private assembly:-

1.goto→D Drive and create a folder with name Sample Assemblies.

2.goto→Start→run→devenv→ok

3. File→new project→select language=visual c#
Template=ClassLibrary→name=privatedll

Location=D:/SampleAssemblies→Add

4.
```
using System;
namespace privatedll
{
publicclassA
   {
publicvoid Show()
      {
Console.WriteLine("i am show");
      }
   }
}
```

5. whenever we compile classlibrary then the compiler will generate .dll file

We can't directly execute .dll file

If we want to execute we have to consume the dll file in any of the executable project.

Go and check in

D:\SampleAssemblies\privatedll/privatedll\bin\debug\Privatedll.dll

6. Consuming privatedll.dll file in Console project

Goto→start→run→devenv→ok

Select language=visual c#-->

Template=Console.Application

Name=consume privatedll

Location=D:\SampleAssemblies→Add

7. Inorder to consume dll file in any of the Executable project, we have to add the dll.File under References.

Goto→SolnExplorer→rc on References→Add Reference→Browse→

D:\SampleAssemblies\privatedll\privatedll\bin\Debug\privatedll.dll

```
8. using System;
using privatedll;

namespace consumeprivatedll
{
classprogram
   {
staticvoidMain()
      {
         A a1 = new A();
         a1.Show();
      }
   }
}
```

Press F5 and check the o/p

Goand check in D:\SampleAssemblies\Consumeprivatedll\Consumeprivatedll\bin\Debug\privatedll.dll

Note:- Whenever we consume privatedll.dll in Console Project a separate copy of dll file was copied into console project . So private Assembly is also called as Folder Specific Assembly.

Public Assembly or Shared Assembly:

Public assembly means the assembly that was registered in GAC location is called as Public Assembly

Q)What is GAC ?

Sol:- Global Assembly Cache

If we place the dll file in GAC and whenever we consume dll file then a separate copy of dll file was not copied in consumed folder.

Note:- All the dll files that was given by MS are placed under GAC

Q) Where is GAC Location ?

Sol:- c:\windows\Microsoft.Net\assembly\GAC_MSIL

Note:- Inorder to Register the dll file in GAC we have to create strongname and public key token.

Ex:-

1. Goto→start→run→devenv→ok
2. File→new project→select language=visual c# Template=classlibrary→Location=D:\SampleAssemblies→name=publicdll

```
using System;
namespace publicdll
{
publicclassA
   {
publicvoid Show()
      {
Console.WriteLine("I am show");
      }
   }
}
```

Goto project on menu bar→publicdll properties→signing→ Sign the assembly→choose a strongname key file →select new name=mykey.dll→ok

3. dll file is eligible to register under GAC
4. build→build soln(compile)
5. goto→start→allprograms→visual studio 2013→ run as administrator→ok
6. inorder to register dll file under GAC

gacutil –i path of the dll file

type command in command prompt

gacutil –i D:\SampleAssemblies\publicdll\publicdll\bin\Debug\publicdll.dll

7. go and check in GAC Location publicdll.dll if registered.
8. Consume publicdll in console appn
9. Goto→start→run→devenv→ok
10. File→new project→select Language=

visual c#-->Template=ConsoleApplication-->name=Consumepublicdll location=D:\SampleAssemblies

11. Goto→solnExplorer→rc on References→add reference→

C:\Windows\Microsoft.Net\assembly\GAC-MSIL\publicdll\v4.0-1.0.00-F3efcc96ca2c4efe\publicdll.dll

```
using System;
using publicdll
namespace consumepublicdll
{
classprogram
{         static void Main()
         {
           A a1=new A();
           a1.Show();
         }  }}
```

12. Press F5 and check o/p
13. Goto→D:\SampleAssemblies\publicdll\publicdll\bin\Debug and check publicdll.dll is available under consumed folder.

Satellite Assembly:-

The assembly which is used to create a dll file in other languages.

Access Modifiers:-

Access Modifiers are used to prepare accessability permissions for the type and its members i.e. who can access and who cannot access the type and its members.

Different types of accessmodifiers are:-

1. Private

2. Protected

3. Internal

4. Protected Internal

5. Public

1. Private:-The scope of private is with in the class if we declare the variable as private we can't access the variable outside the class
->if we declare the method as private we can't access the method outside the class.
->if we declare the constructor as private we can't create object outside the class.
->by default the members of class are private.

2. Protected:- The scope of protected is within the class or in the immediate derived class

->Rules to be followed while working with protected Access Modifiers:-

1. Inheritance is mandatory

2. object and reference for derived class

```
classA
{
    protected void Show()
    {
    }
}

classB:A
{
static void Main(string[] args)
    {
       B b1 = newB();
        b1.Show();
    }
}
```

3. Internal:-

The Scope of Internal is with in the assembly.

->Internal will work like public with in the assembly and private outside the assembly.

->the default access modifier of class is internal.

x.dll

```
namespace n1
{
classA
   {

   }
classB
   {

   }
}
namespace n2
{
classc
{
  void Show()
  {
    A a1 = new A();
  }
 }
}
```

y.exe

```
  namespace n
{
    class y
    {
  static void Main()
        {
          A a1 = new A();
        }
}
   }
```

K
A
N
N
A
B
A
B
U

4. Protected Internal:- It will behave like Internal with in the assembly and protected outside the assembly.

Ex:-

x.dll

```
namespace n1
{
publicclassA
  {
protectedinternalvoid Show()
    {

   }
  }
}
```

y.exe

```
namespace n
{
classy:A
{
static void Main()
{
y y1 = newy();
y1.Show();
}
}

}
```

5. Public:- The scope of public is not restriction i.e within the assembly or outside the assembly.

	Private	Protected	Internal	PI	Public
1.with in the same class	✓	✓	✓	✓	✓
2.with in the separate class	✗	✗	✓	✓	✓
3.In derived class with super class reference	✗	✗	✓	✓	✓
4.In derived class with subclass reference	✗	✓	✓	✓	✓
5.With in assembly	✗	✗	✓	✓	✓
6.outside assembly+ inheritan-ce + DC object and reference	✗	✗	✗	✓	✓
7.outside assembly	✗	✗	✗	✗	✓

Chapter-6

Exception Handling

youtube.com/user/kannababubanna/videos

Exception Handling:-

At the time of developing the appn Different types of Errors will occur
Errors are of 2 Types:-
1. Compile time Error
2. Runtime Error

Compile time Error:-The Error that will occur at the time of Compilation of the program are called as Compile time Errors
Compiletime Errors are also called as Syntax Errors
C#.net->.cs->csc->.exe->CLR--> nativecode->O.S->H/W
MSIL jitcompiler

Q) what is Runtime Error?
The Error that will occur at the time of Execution of the program is called as Runtime Error

Q) what is Exception?
Exception is Runtime Error

Q) why Exception will occur?
Exception will occur because of the wrong i/p given by Enduser or because of the invalid logic written by developer

Q) When Exception will occur?
Exception will occur at the time of Execution of the Program

Q) what will happen when Exception occurs?
Abnormal Termination of Program

Q) Can we rectify Runtime Errors?
no we cannot rectify Runtime Errors but we can handle them

Q) How to Handle Exceptions?

we can handle Exceptions in 3 ways:-

1. By using Logical Implementation
2. By using try-catch implementation
3. By using Application Exception

```
using System;
class A
{
  static void Main()
  {
      Console.WriteLine ("Enter First no");
      int x=int.Parse(Console.ReadLine());
      Console.WriteLine("Enter Second no");
      int y=int.Parse(Console.ReadLine());
      int z=x/y;
      Console.WriteLine(z);
  }
}
```


Observation:- in the above program if y is 0 then an Exception will occur

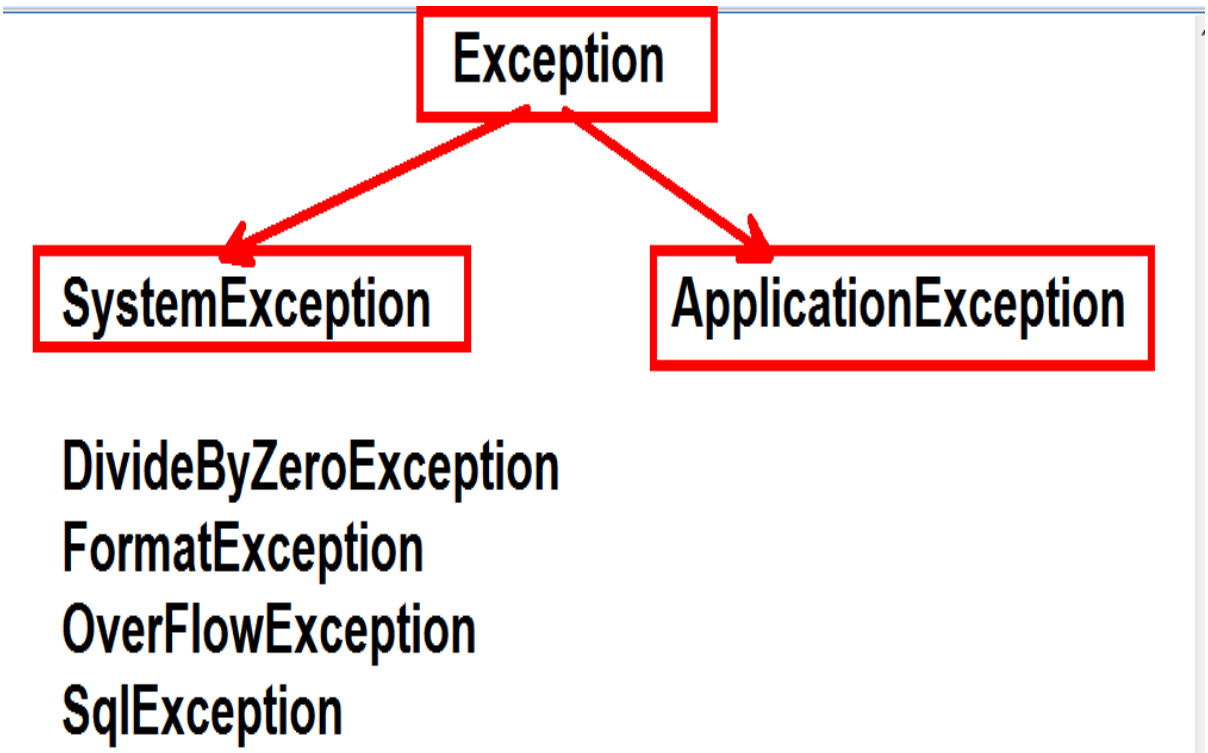
in order to handle the Exception developer must write Logic i.e Logical Implementation

```
using System;
class A
{
    static void Main()
    {
        Console.WriteLine("Enter First no");
        int x = int.Parse(Console.ReadLine());
        Console.WriteLine("Enter Second no");
        int y = int.Parse(Console.ReadLine());
        if (y == 0)
        {   Console.WriteLine("y must not be 0");    }
        else
        {
            int z = x / y;
            Console.WriteLine(z);
        }
    }
}
```

it is difficult for the developer to analyse that what type of Exception will occur and write logic to handle Runtime Errors

so Microsoft has given a mechanism to handle Runtime Errors i.e. try catch implementation

```
try
{
  place the code here
}
catch()
{
   display Error
}
finally
{
  The code that was written inside finally will execute
  even if Exception occurs
}
```



```
class Exception
{
   public virtual string Message
   {      get;    }
}
class DivideByZeroException:Exception
{
   public override string Message
   {     get{ return "Attempting to divide by 0"; }   }
}
class OverFlowException:Exception
{
   public override string Message
   {     get{ return "value is too lareg or too small"; }   }
}

class FormatException:Exception
{
   public override string Message
   {     get{ return "i/p string was not in correct format"; }   }
}
```

Ex:-

```
using System;
class A
{
    static void Main()
    {
        try
        {
            Console.WriteLine("Enter First no");
            int a = int.Parse(Console.ReadLine());
            Console.WriteLine("Enter Second no");
            int b = int.Parse(Console.ReadLine());
            int c = a / b;
            Console.WriteLine(c);
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
        finally
        {
          Console.WriteLine("i will execute even Exception occurs");
        }
    }
}
```

K
A
N
N
A
B
A
B
U

only try	invalid
only catch	invalid
only finally	invalid
try-catch	valid
try - finally	valid
try-catch-finally	valid
try-catch inside try	valid

whenever an Exception occur CLR will create object for corressponding Exception class and CLR will throw the object to catch block
catch will display Error Message
the code that was written inside finally block will gets
executed even if Exception occurs
UserFriendly Exception:-it is difficult for the enduser to understand predefined Error Messages,so programmer must display user friendly Error Messages by declaring multiple Catch blocks

```
using System;
class A
{
    static void Main()
    {
        try
        {
            Console.WriteLine("Enter a no");
            int a = int.Parse(Console.ReadLine());
            Console.WriteLine("Enter b no");
            int b = int.Parse(Console.ReadLine());
            int c = a / b;
            Console.WriteLine(c);
        }
        catch (DivideByZeroException)
        {
            Console.WriteLine("b must not be 0");
        }
        catch (OverflowException)
        {
            Console.WriteLine("plz enter less value");
        }
        catch (FormatException)
        {
            Console.WriteLine("Plz Enter only no");
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
}
```

Userdefined Exceptions or Application Exception:-MS has given a flexibility to developers to create userdefined Exception classes and Override the Message property in Derived class depending on the user requirement

```
using System;
class  OddNoException:Exception
{
    public override string Message
    {  get{  return "Plz Enter Even no";  }    }
}
class A
{
    static void Main()
    {
        try
        {
            Console.WriteLine("Enter a no");
            int no = int.Parse(Console.ReadLine());
            if(no % 2 == 0)
            {
                Console.WriteLine("Even no");
            }
            else
            {
                throw new OddNoException();
            }
        }
        catch(Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
}
```

Chapter-7

C#.net Faq's

youtube.com/user/kannababubanna/videos

1. What is .net?

.net Framework is a runtime Environment which is used for building, deploying, and running applications that use .**NET** languages and Technologies

2. What is Microsoft visual studio Editor?

It is a integrated development Environment which is used to develop Different Types of Applications like Console, Windows, Webapplications, WCF, MVC etc..

3. What are Desktop Applications?

The Applications that was installed on users Desktop are called as Desktop Applications Desktop Applications can developed by using Programming Languages like C,C++,C#.net,vb.net,java etc..

4. What are WebApplications?

A Web application (Web app) is an application program that is stored on a remote server and delivered over the Internet through a browser

interface.Webapplications can develop by using Technologies like Asp.net,JSP,Php etc..

5. What is CLR?

CLR is common language Runtime which provides the execution environment for all .NET Framework code

6. What is CTS?

CTS is Common Type System which provides Common datatypes for all .net supportable languages .At compieltime Language datatype will convert into CTS Types

7. What is CLS?

Common Language Specification (CLS) defines a subset of Common Type System (CTS) . Common Type System (CTS) describes a set of types that can use different .Net languages have in common , which ensure that objects written in different languages can interact with each other.

8. What is CAS?

Code Access security is a security model which grants or denies permission to your assembly depending on evidences like from where the code has emerged, who the publisher is? , strong names etc.

9. What is Assembly?

Assembly is the compiled format of any .net program which may be .dll or .exe,Assembly is collection of namespaces

10. What is MSIL?

a. Microsoft intermediate Language also called as Assembly

b. Whenever we compile any .net supportive language program(C#.net/vb.net) then the language compilers will generate .exe file or .dll file which internally consists of MSIL Code

11. What is managed code?

a. Managed code is the code that is executed directly by the CLR instead of the operating system. The code compiler first compiles the managed code to intermediate language (IL) code, also called

as MSIL code. This code doesn't depend on machine configurations and can be executed on different machines.

12. What is unmanaged code?

a. The **code**, which is developed outside .NET, Framework is known as **unmanaged code**

b. Unmanaged code compiles straight to machine code and directly executed by the Operating System

13. What is Garbage Collector?

a. Garbage Collector is integral part of .net Framework which will take care about Automatic memory management

14. What is JITCompiler?

Just in time Compiler will convert MSIL (Microsoft Intermediate Language) code to Native code because operating system can understand only native code or machine code.

15. What are the Different Types of jitcompiler?

Pre-JitCompiler:- Pre-JIT compiles complete source code into native code in a single compilation cycle. This is done at the time of deployment of the application.

Econo-JitCompiler:- Econo-JIT compiles only those methods that are called at runtime. However, these compiled methods are removed when they are not required.

Normal Jit or standard Jit:- Normal-JIT compiles only those methods that are called at runtime. These methods are compiled the first time they are called, and then they are stored in cache. When the same methods are called again, the compiled code from cache is used for execution.

16. What is The Difference between valuetype and Reference type Datatype?

Value Type	Reference Type
They are stored on stack	They are stored on heap
Contains actual value	Contains reference to a value
Cannot contain null values. However this can be achieved by nullable types	Can contain null values.
Value type memory is automatically destroyed on its own from stack when they go out of scope.	Required garbage collector to free memory.
Memory is allocated at compile time	Memory is allocated at run time
Ex:-Structure and Enumerators	Ex:- class,interface,Array,Delegate

17. What is Boxing?

a. Boxing is a process of converting Valuetype datatype to Reference type datatype

18. What is UnBoxing?

a. UnBoxing is a process of converting Reference type datatype to Valuetype datatype

19. What is the Difference between GetType() and typeof()?

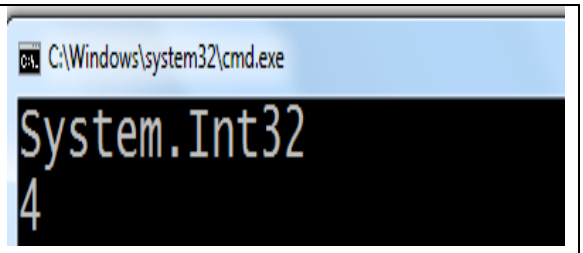
typeof()	**GetType()**
It will return the CTS type of the given datatype	It will return the System.Type of the current instance
It is an operator	It is method

20. What is the difference between typeof() and sizeof()?

a. typeof() is used to get the CTS type of Value type or Reference type datatype

b. sizeof() is used to get the size of valuetype datatype

```
class Program
{
static void Main()
{
Console.WriteLine(typeof(int));
Console.WriteLine(sizeof(int));
}
}
```



21. What are Nullable Datatypes?

a. Nullable Datatypes are used to assign null values to valuetype datatypes

b. Ex:- int? i=null;

22. What is the Difference between int.Parse() and Convert.ToInt32() and int.TryParse()?

	int.Parse()	Convert. ToInt32()	Int.TryParse()
1.	It is used to convert string to int	It is used to convert any datatype to int	It is used to convert string to int
2.	int.Parse() cannot handle null values	Convert.ToInt32() can handle null values	Int.TryParse() can handle null values
3.	Int.Parse() will not check whether the type of parsing is valid or not	Convert.ToInt32() will not check whether the type of parsing is valid or not	Int.TryParse() will check whether the type of parsing is valid or not
4.	**string s = "10"; int i= int.Parse(s);**	string s="20"; int i=Convert.ToInt32(s);	string s = null; int i; bool b = int.TryParse(s, out i); Console.WriteLine(b);

23. How to find the range of byte?

```
Console.WriteLine(byte.MinValue+" to "+byte.MaxValue)
```

24. What is the Difference ToString() and Convert.ToString()?

a. Convert.ToString() handles NULL values even if variable value become NULL.

```
string abc = "";
abc = null;
Console.Write(Convert.ToString(abc));
    Console.ReadKey();
```

b. Tostring() will not handles NULL values it will throw a NULL reference exception error.

Ex:-

```
string abc = "" ;
 abc = null;

Console.Write(abc.ToString());
 Console.ReadKey();
```

25. **what is the Difference between Console.Write() and Console.WriteLine()?**

Console.Write() will write a line on Console and place the cursor on the sameline

Console.WriteLine() will write a line on Console and place the cursor on the next **line**

26. what is the Difference between Console.Read() and Console.ReadLine() and Console.ReadKey()?

Console.ReadLine() will read the input from console in the form of string and returns string

Console.Read() will read the input from console in the form of string and returns int

Console.ReadKey()method accept the Character and return ASCII value of that character

27. what is implicitly type variable?

implicitly type variable must declare as local variable

implicitly type variable must declare with var keyword

implicitly type variables are also called as Anonymous Types

The datatype will be decided based on the value that we store

var a=10; at compiletime based on the value that we store\

28. what is Difference between var and Dynamic datatype?

var type of variables are required to be initialized at the time of declaration. Once assigned a value into var type then it act as that data type like int,string,etc. So after that we can't able to change variable datatype.

- Dynamically typed - Dynamically we can change the variable type like int,string,etc. That means the variable type declared at runtime.

Ex:-

```
using System;

class Program

{

  static void Main(string[] args)

  {

      var a = 10;

      a = "abc";       ----------------------(invalid)Error

      dynamic b = 20;

      b = "abc";--------------------------(valid)

      Console.WriteLine(b);

  }

}
```

29. Can we perform Arthimetic operations on object type?

no we cannot perform Arthimetic operators on object type
object a=10;
object b=20;
object c=a+b; Error we cannot perform Arthimetic operations on object type

30. what is implicit Typecasting?

It is not required for the programmer to write I code while converting from one datatype to another datatype

Ex:-

int i=10;

long l=I; valid

31. what is Explicit Typecasting?

The programmer has to write I code while converting from one datatype to another datatype

Ex:- Boxing,UnBoxing

int i=10;

string s=i.ToString();

32. What is OOPS?

OOPS is a concept which is used to write computer programs by using classes and objects

33. What are the Principles of oops?

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism

34. What are the Advantages of OOPS?

- Dataorganization,
- DataSecurity
- Reusability
- Extensability
- Reimplementation

35. What is Abstraction?

Abstraction is used for hiding the unwanted data and giving only relevant data

36. What is Encapsulation?

It is a process of Wrapping or Binding or Grouping of state and Behavior in a single container

37. What is class?

Class is user defined Referencetype data type which consists of variable and methods

38. What is object?

Object is instance of class (instance means allocating sufficient memory spaces for instance variables)

39. what is the difference between static variable and instance variable?

Static variable	**Instance variable**
Must declare with static keyword	No special keyword is required to declare instance variable

Memory is allocated on stack	Memory is allocated on heap
Memory is allocated at the time of class loading	Memory is allocated when we create object for a class
Memory is allocated only one time at the time of loading class	Memory is allocated everytime when we create a new object
Static variable can be accessed by using classname	Instance variable can be accessed by using objectname

40. What are the characteristics of object?

State,Behaviour and Identity

41. What is the Difference between Structure and Class?

	Structure	**Class**
1.	Must declare with struct keyword	Must declare with class keyword
2.	Structure is valuetype datatype	Class is Referencetype datatype

3.	Memory is allocated on stack	Memory is allocated on heap
4.	Structure is recomended to store small amount of data	Class is recomended to store large amount of data
5.	Structures are inherited from System.Value Type	Classes are inherited from System.Object Type
6.	new keyword is optional to create object	new keyword is manadatory to create object
7.	Structure doesnot support Default constructor	Class will have Default constructor
8.	Structures doesnot support Destructors	Supports Destructors

9.	Static constructors will not reflect in structures	supports static constructor
10.	Structure cannot be inherited to other type	Class can inherit to other class
11.	Structure cannot be declared as abstract	Must declare as abstract

42. what is Enum?

- An enum is a value type datatype with a set of related named constants
- An enum is used to create numeric constants in .NET framework
- Enums are not for end-user, they are meant for developers.
- Enums are strongly typed constants. i.e. an enum of one type may not be implicitly assigned to an enum of another type even though the underlying value of their members are the same.
- Enumerations (enums) make your code much more readable and understandable.

syn:- enum enumname { }

Ex:- enum Days{ Sun,Mon,Tue,Wed,Thu,Fri,Sat}

43. what is the Difference between Method and Constructor?

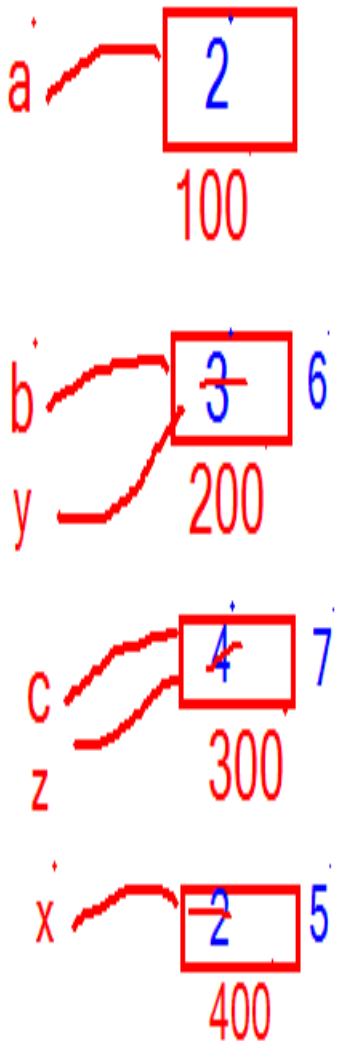
	Method	**Constructor**
1.	Method is a subprogram which is used to perform some operation	Constructor is used to initialize the values for instance variables
2.	Methodname and classname must not be same	Constructorname and classname both must be same
3.	Method must have returntype atleast void	Constructor doesnot have returntype atleast void
4.	Method will gets executed when we call it	Constructor will gets executed when we create object

44. what is the Difference between Call by value and Call by Reference?

Call by value	**Call by Reference**
It is a process of calling the method by passing the value as parameter	It is a process of calling a method by passing Reference as parameter
The modifications of Formal Parameters will not affect actual parameters	The modifications of Formal parameters will affect actual paarameters
At the time of calling the method we pass value	We can achieve callby reference by using ref and out keyword

Ex:-

```
using System;
class Program
{
static void Show(int x, ref int y, out int z)
{
Console.WriteLine("Formal parameters");
        x = 5;    y = 6;      z = 7;
        Console.WriteLine(x);
        Console.WriteLine(y);
        Console.WriteLine(z);
    }
    static void Main()
    {
      int a = 2;     int b=3;     int c=4;
      Show(a,ref b,out c);
      Console.WriteLine("Actual parameters");
      Console.WriteLine(a);
      Console.WriteLine(b);
      Console.WriteLine(c);
    }
}
```



45. what is the Difference between ref and out keyword?

Ref	Out
ref must declare with ref keyword	out must declare with out keyword
Initialization of actual parameters is manadatory	Initialization of actual parameters is optional
Initialization of formal parameters is optional	Initialization of formal parameters is manadatory

46. what is the difference between constant,Readonly?

Constant	Readonly
Constant must declare with const keyword	Readonly must declare with readonly keyword
Const is compiletime constant	Runtime constant but only through non static constructor

Const value must be initialized at the time of declaring	Readonluy can be initialized at the time of declaring or through non static constructor
Const value cannot be modified	Readonly cannot be modified through constructor only
Cannot be declared as static	Can be declared as static

47. what is this?

this is a keyword which represents current class object

48. How to call current class constructors?

by using this()

49. How to call super class constructors?

base()

50. what is the difference between this,base?

this is used to access current class instance variables,instance methods and base is used to access super class instance variable and methods

51. if we declare static constructor and static void Main() which one will execute first?

static constructor

52. what is inheritance?

inheritance is a mechanism of creating a new class by already existing class

inheritance is used to establish the relationship between 2 or more classes

53. what are different Types of inheritance?

- Singlelevel inheritance
- Multilevel inheritance
- Multiple inheritance
- Hybrid inheritance
- Heirarcheal inheritance

54. Can we inherit constructors?

no we cannot inherit

55. Does C#.net supports Multiple inheritance?

C#.net doesnot supports multiple inheritance directly we can achieve multiple inheritance by using interfaces

56. what is the difference between private constructor and static constructor?

Private constructor	**Static constructor**
The private constructor will be executed each time it is called.	The static constructor will only be executed once.
The private Constructor may have parameters	The static constructor cannot have parameters.
. Private constructor is called after the instance of the class is created.	A static constructor is called before the first instance is created
A class can have only multiple private constructor	A class can have only one static constructor
We cannot create object for private constructor outside the class	NA

Private constructor must declare as private	Static constructor doenot have accessmodifiers
A private constructor is a special instance constructor. It is generally used in classes that contain static members only	NA

57. How to stop inheritance?

By declaring the class as sealed

58. what is partial class?

Declaring multiple classes with same name is called partial class at compiletime all the partial classes will become as single class

59. what is Polymorphism?

Polymorphism came from 2 greek words poly and morphos which means many forms

simply Polymorphism is an ability to take morethan one form

60. How can we achieve Polymorphism?

we can achieve polymorphism by using Overloading and Overriding

61. what is Overloading or Method Overloading?

it is a process of defyning multiple methods with same method name but with different parameters in same class or in derived class

in overloading which method will execute was decided based on the number of values,order of values and type of values that we pass at the time of calling method

```
class A
{
public void Show()
{
Console.WriteLine("i am without parameters");
}
public void Show(int x)
{
Console.WriteLine("i am with single parameters");
}
public void Show(int x,int y)
{
Console.WriteLine("i am with double parameters");
}
static void Main()
{
A a1=new A();
a1.Show();
a1.Show(10);
a1.Show(10,20);
} }
```

62. what is Overriding?

it is a process of defyning multiple methods with same method Signature in base class and derived class

Overriding is a process of Reimplementing the baseclass method in Derived class

Ex:-

```
class A
{
 public virtual void Show()
{
Console.WriteLine("i am A show");
}
}
class B:A
{
 public override void Show()
{
Console.WriteLine("i am B show");
}
}
```

```
class program
{
        static void Main()
        {
           A a1=new B();
              a1.Show();
        }
 }
```

o/p:- i am B Show

63. what is the Difference between Overloading and Overriding?

Overloading	**Overriding**
possible in single class	Not possible in single class
Possible with inheritance	Inheritance is manadatory
Can overload static methods	We cannot override static constructors
Can overload constructors	Cannot override constructors

64. what is the Difference between Compiletime polymorphism and runtime polymorphism?

Compiletime Polymorphism	**Runtime Polymorphism**
Method call will bind with method behaviour at compiletime	Method call will bind with method behaviour at runtime
Which method must gets executed was decided at compile time	Which method must gets executed was decided at runtime

Also called as static polymorphism	Also called as Dynamic polymorphism
Also called as compiletime polymorphism	Also called as runtime polymorphism
Ex:- Overloading	Ex:- Overriding

65. How to stop Overriding?

we can stop overriding by declaring overriden method as sealed

66. what is the difference between virtual method and sealed method?

virtual method can override in derived class but we cannot override sealed method

67. **what is Method Hiding?**

it is the concept of hiding the base class method from the derived class by using new **keyword**

68. Can we Override properties?

yes

69. what is the super class for all classes?

System.Object class

70. What are the methods that are available in Object class?

- Equals()
- GetHashCode()
- ToString()
- GetType()

71. what is the default accesmodifier of Default constructor?

public

72. **How can you call a base class constructor from a child class ?**

by using base()

73. what is abstract class?

An abstract class is a class that cannot be instantiated. It can only be used as a base class

74. Why do we need an Abstract Class?

75. Why we cannot create object for abstract class?

Abstract class is not fully implemented class so we cannot create object for abstract class but we can create reference

76. what is interface?

- interface is a type which consists of public abstract methods and public abstract properties
- interface is used to acheive multiple inheritance
- interface is a contract or an aggrement between itself and its implemented class

77. Does C#.net supports multiple inheritance?

C#.net deosnot support multiple inheritance directly we can achieve multiple inheritance by using interfaces

78. What are the differences between abstract class,interface,class ,partial class,sealed class,static class?

Untitled - Notepad

File Edit Format View Help

		class	sealed	static	partial	abstract	interface
1	static variables	✓	✓	✓	✓	✓	✗
2	instance variables	✓	✓	✗	✓	✓	✗
3	static Methods	✓	✓	✓	✓	✓	✗
4	instance methods	✓	✓	✗	✓	✓	✗
5	abstract method	✗	✗	✗	✗	✓	✓
6	static constructor	✓	✓	✓	✓	✓	✗
7	instance constructor	✓	✓	✗	✓	✓	✗
8	static property	✓	✓	✓	✓	✓	✗
9	instance property	✓	✓	✗	✓	✓	✗
10	abstract property	✗	✗	✗	✗	✓	✓

09:17
03-09-2019

		class	sealed	static	partial	abstract	interface
11	object	✓	✓	✗	✓	✗	✗
12	Reference	✓	✓	✗	✓	✓	✓
13	inheritance	✓	✗	✓	✓	✓	✓
14	multiple inheritance	✗	✗	✗	✗	✗	✓
15	Overriding	✓	✗	✗	✓	✓	✗

79. what is Assembly?

Assembly is the compiled format of any .net program which may be .dll or .exe

whenever we compile any C#.net program or Vb.net program then the compiler will generate .exe or .dll file which is called as Assembly

Assembly is also called as IL,MSIL,CIL

80. What is the difference between .dll and .exe?

.dll	**.exe**
Dynamic Link Library	Directly Executable program
.dll file is Reusable file but not Executbale	.exe file is Executable file but not Reusable
.An EXE file can be run independently	DLL is used by other applications.
Is used for multiusers	Is used for single user
Doesnot have Entrypoint static void Main(){}	Must have Entrypoint static void Main(){}

81. what is ILDASM?

ILDASM.exe is a tool which is used to view the information of Assembly

ILDASM:- intermediate Language Disassembler

82. what is Reflection?

Reflection is api which is used to get the information of the assembly programatically by writing some code

83. what are Different Types of Assemblies?

- Private Assembly
- Public Assembly or Shared Assembly
- Satellite Assembly

84. what is the Difference between private assembly and public assembly?

Private Assembly	Public Assembly
The Assembly which is specific to a single application or specific folder is called as Private Assembly	The Assembly which was regietsred under GAC is called as Public Assembly
Private assembly is also called as Folder Specific Assembly because a seperate copy of dll file is copied into each and every folder	Public assembly is also called as Shared assembly because a single copy of dll file is shared from GAC location

85. what is GAC?

GAC is Global assembly Cache which is used to register dll files

86. where is GAC Location?

C:\Windows\Microsoft.NET\assembly\GAC_MSIL

87. How to install and uninstall dll file in GAC?

gacutil -i dllname.dll

gacutil -u dllname.dll

88. what is Satellite Assembly?

An assembly which can used to develop multi lingual applications in .net

89. what is Delegate?

A delegate is a type safe function pointer like function pointers in C,

delegates are object-oriented, type safe, and secure

90. what is the Difference between Method overloading and Delegates?

Method overloading means Defyning multiple methods with same name but with different parameters

Delegates means defyning multiple methods with Different methodname and same parameters

91. what is multicast delegate?

A delegate that is subscribed by more than one method is called as multicast delegate

92. what is an Exception?

Exception is Runtime Error

93. what is the base class for all Exception classes?

System.Exception

94. what is Exception Handling?

Exception Handling is a mechansim of handling runtime Errors

95. what is try-catch-finally?

A try block:- is the block of code in which exceptions occur.

catch block:- is used to display Errormessages

finally block:- the code that we write inside finally block will gets executed even if Exception occurs

96. observe below Table?

Only try	Invalid
Only catch	Invalid
try-catch	Valid
try-finally	valid
try-catch-finally	Valid
Catch-finally	Invalid
try catch inside try	valid
Only finally	Invalid

97. Can we write multiple catch blocks?

yes if we want to display multiple user friendly Exceptions in a single program then we can write multiple catch blocks

```
using System;
class A
{
    static void Main()
    {  try
       {
           Console.WriteLine("Enter first no");
           int a = int.Parse(Console.ReadLine());
           Console.WriteLine("Enter second no");
           int b = int.Parse(Console.ReadLine());
           int c = a / b;
           Console.WriteLine(c);
       }
```

```
        catch (DivideByZeroException)
        {    Console.WriteLine("Denominator must not be 0");      }
        catch (FormatException)
        {    Console.WriteLine("please enter only no");       }
        catch (OverflowException)
        {     Console.WriteLine("please enter less value");    }
        catch (Exception e1)
        {     Console.WriteLine(e1.Message);      }
    }
}
```

98. what is the Difference between string and stringBuilder?

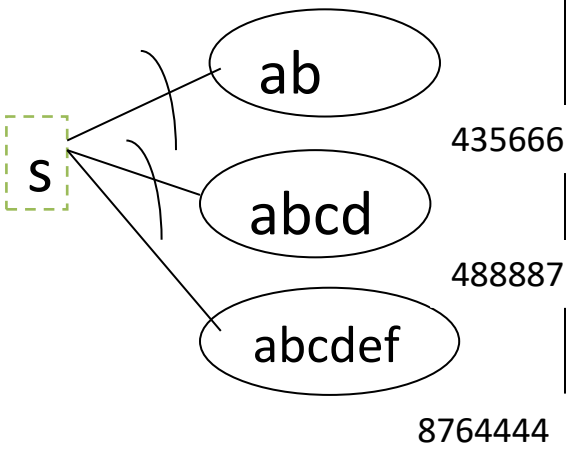
String	StringBuilder
string is immutable which means value will not change	Is mutable which means value will change
string is a class available under System namespace	StringBuilder is available in System.Text namespace
string will always allocate a new memory whenevrer we perform concadination operation	StringBuilder will modify the existing memory whenever we append a string
Performance wise string is slow because every time it will create new instance	Performance wise stringbuilder is high because it will use same instance of object to perform any action

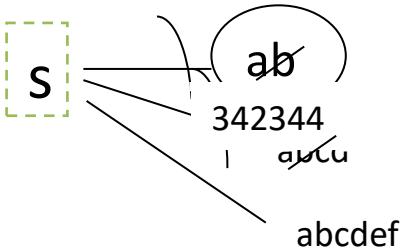
```
using System;
using System.Text;
class A
{
    static void Main()
    {
        Console.WriteLine("string will allocate seperate memory for every concadination operation");
        string s = "ab";
Console.WriteLine(s.GetHashCode());
        s = s + "cd";
Console.WriteLine(s.GetHashCode());
        s = s + "ef";
Console.WriteLine(s.GetHashCode());
        Console.WriteLine("stringbuilder will modify the existing memory");
        StringBuilder sb = new StringBuilder("ab");

Console.WriteLine(sb.GetHashCode());
        sb.Append("cd");

Console.WriteLine(sb.GetHashCode());
        sb.Append("ef");

Console.WriteLine(sb.GetHashCode());
    }
}
```





note:- string will allocate seperate memory for every concadination operation so every time a new object is created and a seperate hashcode is generated

stringbuilder will modify same memory so every time a new hashcode is not generated

99. what is Serialization?

Serialization is a process of converting object into stream of bytes and store the stream of bytes in file or database or network

100. What is DeSerialization?

DeSerialization is a process of converting stream of bytes in to object

101. How many ways we can acheive Serialization?

1. Binary Serialization
2. Soap Serialization
3. XML Serialization

102. what is collection?

Collections are used to implement Datastructures in .net

Collections are used to perform operations on group of objects

103. what is the Difference between Array and ArrayList?

Array	**ArrayList**
The size of Array is fixed	The size of ArrayList is not fixed
Array is used to storeHomogeneous values i.e multiple values of same datatype	ArrayList is used to storeHeterogeneous values i.e multiple values of different datatype
We cannot insert the value in Array at a specific position	We can insert the value in ArrayList at a specific position
We cannot remove the value from Array at a specific position	We can remove the value from ArrayList at a specific position
We cannot increase or decrease the size of Array depending on the requirement	We can increase or decrease the size of ArrayList depending on the requirement
Predefined methods are not available to perform searching and sorting operations	Predefined methods are available to perform searching, sorting,inserting operations

Array is available in System.Array namespace	ArrayList is available in System.Collections namespace

104. what is IEnumerable?

IEnumerable is an interface which is used to iterate over a collection

When you want to get all the results from your query and apply any further filtering once your collection has been loaded in memory.

105. What is the difference between IEnumerable and IQueryable methods in c#?

IEnumerable	**IQueryable**
IEnumerable exists in System.Collections Namespace	IQueryable exists in System.Linq Namespace.
IEnumerable can move forward only over a collection, it can't move backward and between the items.	Queryable can move forward only over a collection, it can't move backward and between the items.
IEnumerable is best to query	IQueryable is best to query data

data from in-memory collections like List, Array etc.	from out-memory (like remote database, service) collections.
While query data from database, IEnumerable execute select query on server side, load data in-memory on client side and then filter data.	While query data from database, IQueryable execute select query on server side with all filters.
IEnumerable is suitable for LINQ to Object and LINQ to XML queries	IQueryable is suitable for LINQ to SQL queries.
IEnumerable doesn't supports custom query.	IQueryable supports custom query using CreateQuery and Execute methods.
IEnumerable doesn't support lazy loading. Hence not suitable for paging like scenarios.	IQueryable support lazy loading. Hence it is suitable for paging like scenarios.
IEnumerable will get all the records at once	IQueryable only get the records that you want

note:-

Let's assume that we have a table with name emp with 1000 records If you use IEnumerable to get the first 5 emps we need to load all 1000 records and then select the first 5. With IQueryable you only select the first 5 (saving a lot of resources !)

i.e if we want o retrieve specific records from collections then use IQueryable

```
static void Main()
{
    using (var db = new MyDbContext())
    {
        IEnumerable<Employee> emps = db.Clients.Take(5).ToList();
        // ToList() executes the query straight away
        // All the list of clients is loaded into memory (1000 employees)
        // After that, only the first 5 are selected
        IQueryable< Employee > employees = db.Clients.Take(5).ToList();
        // ToList() executes the query straight away
        // The first 5 emps are loaded into memory
    }
```

106. **What is the difference between dispose and finalize methods in c#?**

Finalize:

- Finalize used to free unmanaged resources those are not in use like files, database connections in application domain and more, held by an object before that object is destroyed.
- In the Internal process it is called by Garbage Collector and can't called manual by user code or any service.
- Finalize belongs to System.Object class.
- Implement it when you have unmanaged resources in your code, and make sure that these resources are freed when the Garbage collection happens.

Dispose:

- Dispose is also used to free unmanaged resources those are not in use like files, database connections in Application domain at any time.
- Dispose explicitly it is called by manual user code.
- If we need to dispose method so must implement that class by IDisposable interface.
- It belongs to IDisposable interface.
- Implement this when you are writing a custom class that will be used by other users.

107. what are Accessmodifiers?

Access modifiers are keywords used to specify the declared accessibility of a member or a type

108. what is Garbage collector?

Garbage collector is integral component of CLR which is resposible for Memory Management

109. what are the Different Accessmodifiers that are available?

Modifier	Description
Public	There are no restrictions on accessing public

	members
Private	The scope of private is within the class By default the members of the class or private
Protected	The scope of protected is within the class or the class that inherits from that class
Internal	The scope of internal is within the Assembly
Protected internal	Access is limit within the Assembly or outside the assembly within the class that inherits from that class

110. what are Destructors?

D estructors are used to destruct the object. A class can only have one destructor.Destructors cannot be inherited or overloaded. · Destructors cannot be called. They are invoked automatically.

The destructor implicitly calls Finalie() method to free unmanaged resources

111. what are Generics?

Generics provide the ability to create type-safe collections in .NET.

Generics are available in System.Collections.Generics namespace

Generics are used to avoid Function Overloading

112. Can we perform Arthimetic Operations on generics? or What are limitations of Generics?

Generics doesnot support Arthimetic operations

113. Difference between Generics and Array List?

Array List is not type safe because it faces problems of boxing and UN boxing.

List generics are type safe and do not require boxing and UN boxing situations.

In terms of performance of application List generics is better than array list.

Array List can save different type data types.

List generics we can save only specific data type.

Array List consumes lots of memory compare to list generics

114. what are Extension Methods?

without inheritance if we want to add some extra methods to existing class then we have to use Extension Methods

An extension method is a static method of a static class, where the "this" modifier is applied to the first parameter. The type of the first parameter will be the type that is extended.

```
using System;
public class A
{
public void Show()
{      Console.WriteLine("i am show");    }
}
public static class X
{
public static void NewMethod(this A obj)
{     Console.WriteLine("This is extended method");       }
}
class Program
{
static void Main()
{
A a1= new A();
a1.Show();
a1.NewMethod(); } }
```

115. What is the difference between Static class and Singleton instance?

In c# a static class cannot implement an interface. When a single instance class needs to implement an interface for some business reason or IoC purposes, you can use the Singleton pattern without a static class.

You can clone the object of Singleton but, you can not clone the static class object

Singleton object stores in Heap but, static object stores in stack

A singleton can be initialized lazily or asynchronously while a static class is generally initialized when it is first loaded

116. what is GC.Collect?

it forces an immediate garbage collection of all generations.

117. what is dllhell in .net?

in COM technology there is a problem called DLL Hell creating more than one dll with same name will override previous dll and that dll hell problem was overcomed in .net

Dll hell, is kind of conflict that occurred previously, due to the lack of version supportability of dll for (within) an application

.NET Framework provides operating system with a global assembly cache. This cache is a repository for all the .net components that are shared globally on a particular machine. When a .net component installed onto the machine, the global assembly cache looks at its version, its public key and its language information and creates a strong name for the component. The component is then registered in the repository and indexed by its strong name, so there is no confusion between the different versions of same component, or DLL