INDEX

Sr no	Practical Title	pageno	Date	Signature
1	basics of c#		25-Apr-2022	
2	To demonstrate garbage collection		26-Apr-2022	
3	Demonstration of cls and cts through vb.net and c#.net using ildasm (intermediate language disassembler)		27-Apr-2022	
4	Comments in c#		27-Apr-2022	
5	Demonstration of namespaces in single project(console application c#) and into different class library (in single executable and different dll's)		27-Apr-2022	
6	Namespaces and Class Library		27-Apr-2022	
7	Typecasting in c#		28-Apr-2022	
8	Methods and types of parameters in c# 1. Call By Value 2. Call By Reference 3. Out parameter 4. Call by Params 5. Demonstration of Goto and break statements in methods		28-Apr-2022	
9	Arrays and jagged array in C#		28-Apr-2022	
10	Demonstration of typesafe and unsafe in C#		29-Apr-2022	
11	Structure,class,fields,properties,constructor in C#		30-Apr-2022	

12	Properties and Inheritance		02-May-2022	
13	Constructors and its type in c#		02-May-2022	
14	Inheritance in c#		02-May-2022	
15	Method hiding in c#		02-May-2022	
16	Polymorphism using method overriding in c#		02-May-2022	
17	Abstraction in c#		04-May-2022	
18	Interfaces		04-May-2022	
19	Partial class and methods in c#		04-May-2022	
20	Extension methods		04-May-2022	
21	Sealed classes and methods in c#		04-May-2022	
22	Method overloading		05-May-2022	
23	Enum in C#		05-May-2022	
24	Delegates in C#		05-May-2022	
25	Multicast Delegate		05-May-2022	
26	Generics and Collections in C#		06-May-2022	
27	Exception Handling in C#		07-May-2022	
28	File Handling in C#		07-May-2022	
29	Serialization and Deserialization		08-May-2022	
30	Create & Connect database in c# with SQL express server Database		09-May-2022	
31	Windows form Application		09-May-2022	
32	Crud operation with window form app		09-May-2022	

33	SQLconnectivity with Webapplication and SQL Injection		09-May-2022	
34	Demonstration of Dataset and DataTable Difference between dataset datatable and DataRelationship		10-May-2022	
35	Using stored procedures for crud operation in Asp.net Web application for crud operation		10-May-2022	
36	Application Life Cycle		10-May-2022	
37	Page Life Cycle		10-May-2022	
38	Request Response and Session object			
39	Asp.net MVC framework and entity models		12-May-2022	
40	Demonstration of WebServices		12-May-2022	

25 Apr 2022

practical 1:

basics of c#

namespace
user input
concatenation and placeholder
data types
value types
nullable operator
null collapsing operator
ternary operator

Namespace
1.without commenting using system
Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    internal class Program
    {
        static void Main(string[] args)//highest priority in stack or single entry point for execution
        {
  //System.Console.WriteLine("Welcome to my first c# program");//writeline() is method of static class Console and
            Console.WriteLine("Welcome to my first c# program");//writeline() is static method of class Console and
```

```
                                        //static method does not need a object
                                        //static will be given highest prefenrence
Console.ReadKey();
}
}
}
```

Output:-

```
D:\visual studio\vs_savefile\ConsoleApp1\ConsoleApp1\bin\Debug\ConsoleApp1.exe
Welcome to my first c# program
```

On commenting namespace:

Error List

Entire Solution | 4 Errors | 0 Warnings | 0 Messages | Build + IntelliSense | Search Er

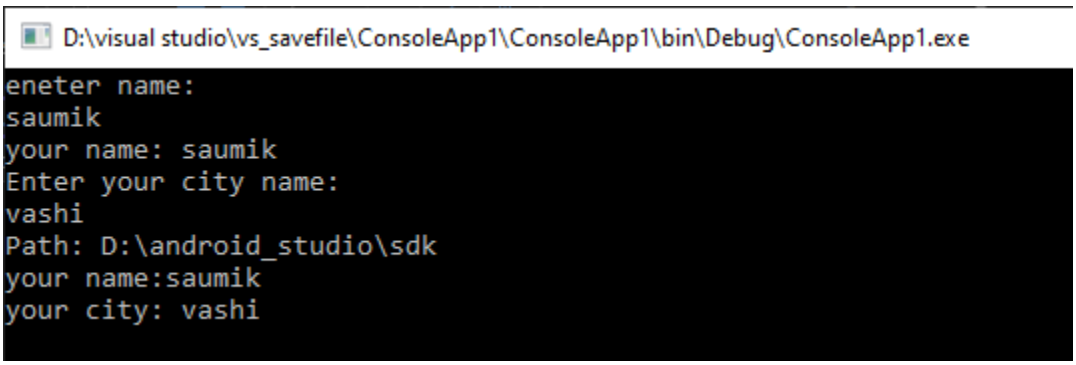
	Code	Description	Project	File	Line
⊗	CS0103	The name 'Console' does not exist in the current context	D:\visual studio\vs_savefil...	Program.cs	29
⊗	CS0103	The name 'Console' does not exist in the current context	D:\visual studio\vs_savefil...	Program.cs	94
⊗	CS0103	The name 'Console' does not exist in the current context	D:\visual studio\vs_savefil...	Program.cs	102
⊗	CS0103	The name 'Console' does not exist in the current context	D:\visual studio\vs_savefil...	Program.cs	103

2.concatenation and placeholder

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    internal class Program
    {
```

```
        static void Main(string[] args)//highest priority in stack or single entry point
for execution
        {
Console.WriteLine("eneter name: ");
                    //var name = Convert.ToInt32(Console.ReadLine());
                    var name = Console.ReadLine();

                    Console.WriteLine("your name: "+name);//concatenation
                    Console.WriteLine("Enter your city name: ");

                    var city = Console.ReadLine();//Readline() return type is string
                    String path = @"D:\android_studio\sdk";// use double backslash to
avoid unrecognised escape sequence
                                                // use @ verbatim literal to avoid
unrecognised escape sequence('\','/','\n'
                    Console.WriteLine("Path: " + path);
                    Console.WriteLine("your name:{0} \nyour city: {1} " , name,
city);//using placeholder

Console.Readkey();

   }
    }
}
```

output:-



3.Datatypes

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    internal class Program
    {
        static void Main(string[] args)//highest priority in stack or single entry point
for execution
        {

    string name = null;
            int?i = null; //use question mark after int ,question mark is called as
nullable operator which
                        //allowprimitive datatypes to hold null value
            Console.WriteLine("value of i {0}",i);

            //? null operator("?")
            //?? null collapsing operator("??") to check value of variable
            // ternary operator ?:

             int a = 10;
            bool availability = a>18?true:false;


            Console.WriteLine("Availability: {0}", availability);
            int totalpresent = 0;
            int? totalabsent = null;
            int? abs = totalabsent ?? 0;
```

```
        Console.WriteLine("total {0} absent students in class", abs);

        if (totalabsent.Equals(null))
        {
            totalabsent = 0;
            Console.WriteLine("{0} absent students in class",totalabsent);
        }

        else
        {
            Console.WriteLine("{0} absent students in class", totalabsent);
        }
    Console.WriteLine("int  min limit is :{0}  and max limit is {1}", int.MinValue,
int.MaxValue);
Console.Readkey();
  }
   }
}
```

D:\visual studio\vs_savefile\ConsoleApp1\ConsoleApp1\bin\Debug\ConsoleApp1.exe

```
value of i
Availability: False
total 0 absent students in class
0 absent students in class
int  min limit is :-2147483648  and max limit is 2147483647
```

26 apr 2022

Practical 2:

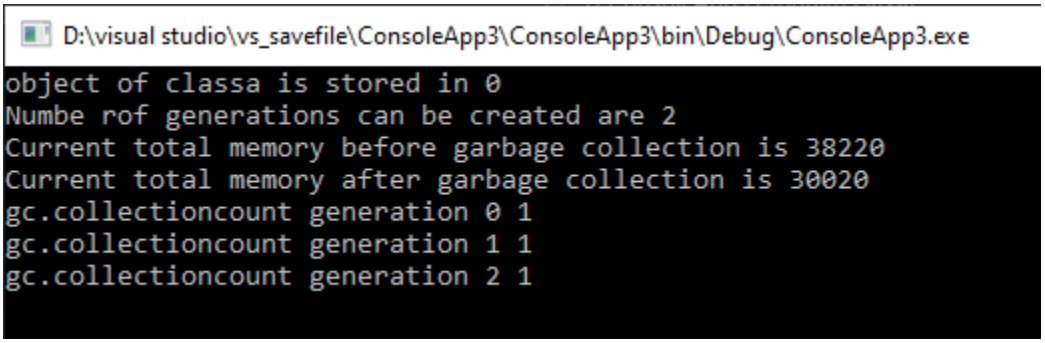
To demonstrate garbage collection

Code:
consoleapp3-> program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp3
{
    internal class Program
    {
        static void Main(string[] args)
        {
            //practical2 to demonstrate garbage collection
            classa a = new classa();
            int i = 2133154135;
            float f = 4353254663.42525425f;
            double d = 4534534.24523d;
  //to check the generation in which current obbject is stored
            Console.WriteLine("object of classa is stored in {0}",
GC.GetGeneration(a));

            //total generations created by system
            Console.WriteLine("Numbe rof generations can be created are
{0}",GC.MaxGeneration);

            Console.WriteLine("Current total memory before garbage collection is
{0}", GC.GetTotalMemory(false));
```

```
            classa a1 = new classa();
            classa a2 = new classa();
            GC.Collect();


            Console.WriteLine("Current total memory after garbage collection is {0}",
GC.GetTotalMemory(false));

            classa a3 = new classa();
            //GC.Collect();
   //collection count by generation number
            Console.WriteLine("gc.collectioncount generation 0 {0}",
GC.CollectionCount(0));
            Console.WriteLine("gc.collectioncount generation 1 {0}",
GC.CollectionCount(1));
            Console.WriteLine("gc.collectioncount generation 2 {0}",
GC.CollectionCount(2));  Console.ReadKey();
            }
  public class classa
        {
        }
    }
}
```

Output:-



Practical 3:

Demonstration of cls and cts through vb.net and c#.net using ildasm (intermediate language disassembler)

Code:
c# code
Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp4
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int i;
            double d;
        }
    }
}
```

Vb.net code
Module1.vb

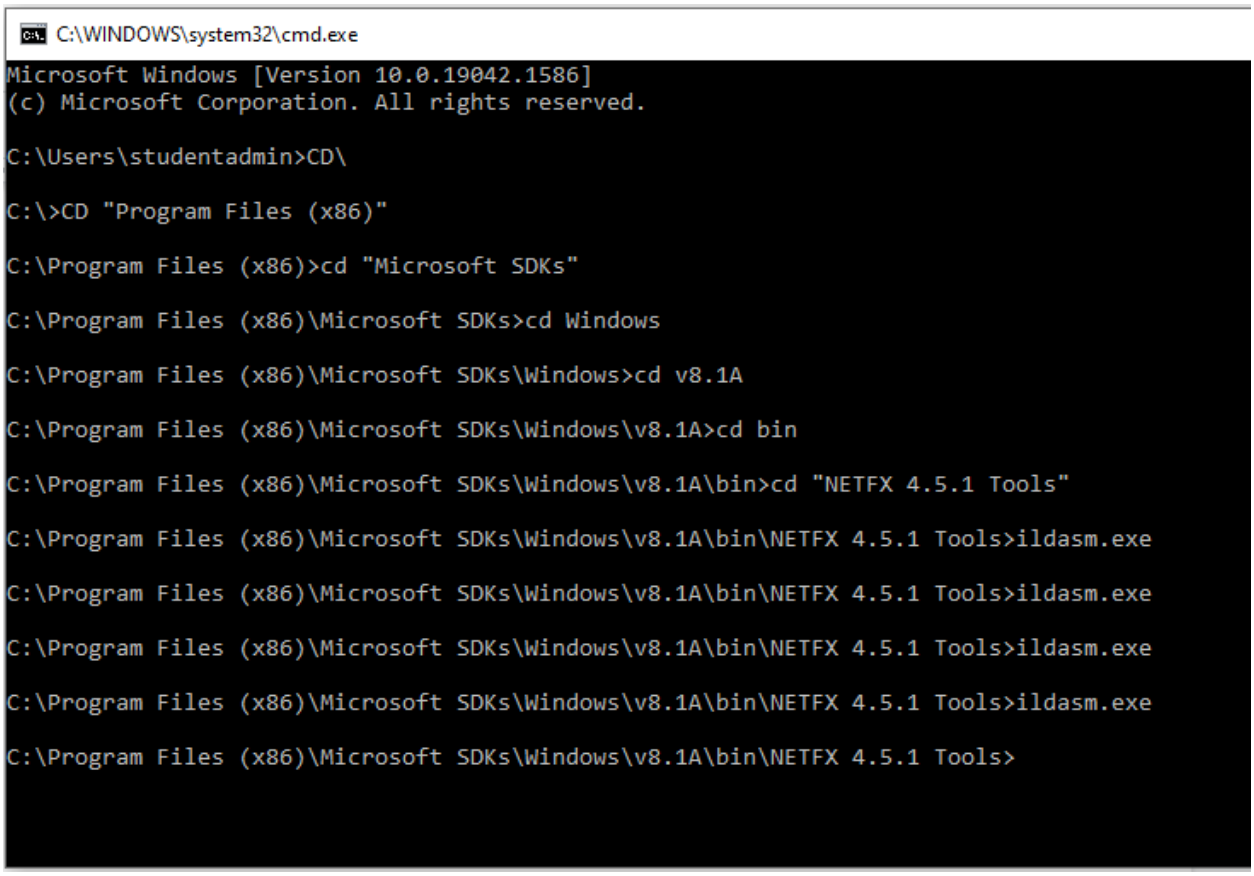
```
Module Module1

    ''' <summary>
    '''
    ''' </summary>
    Sub Main()
```

```
        Dim rollno As Integer
        Dim srno As Double

    End Sub

End Module
```

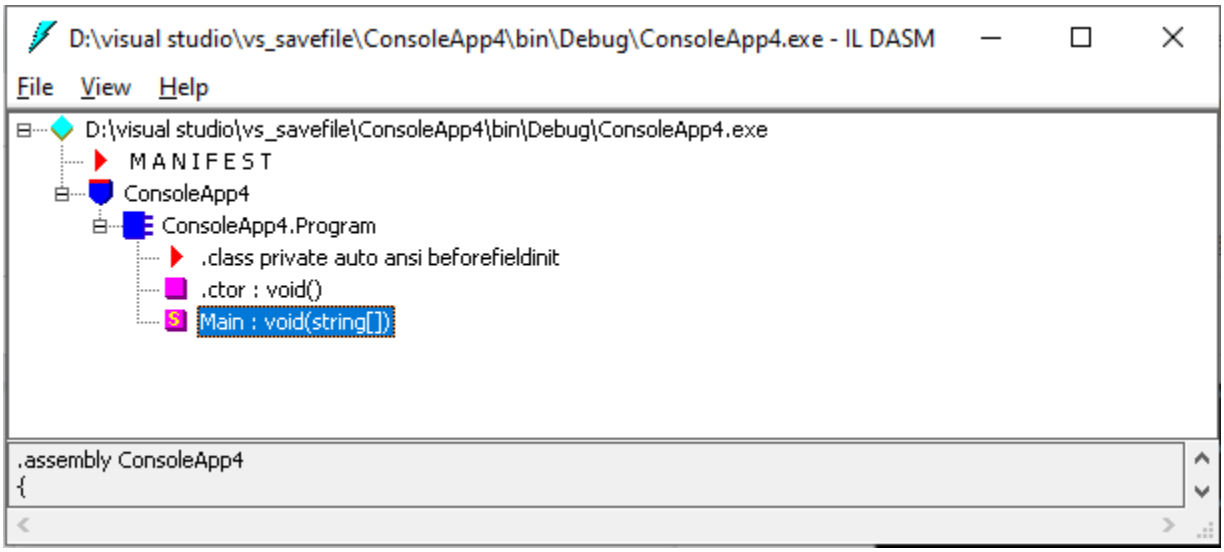
note:-
Irrespective of programming language clr creates similar assembly code for datatypes as per language specification

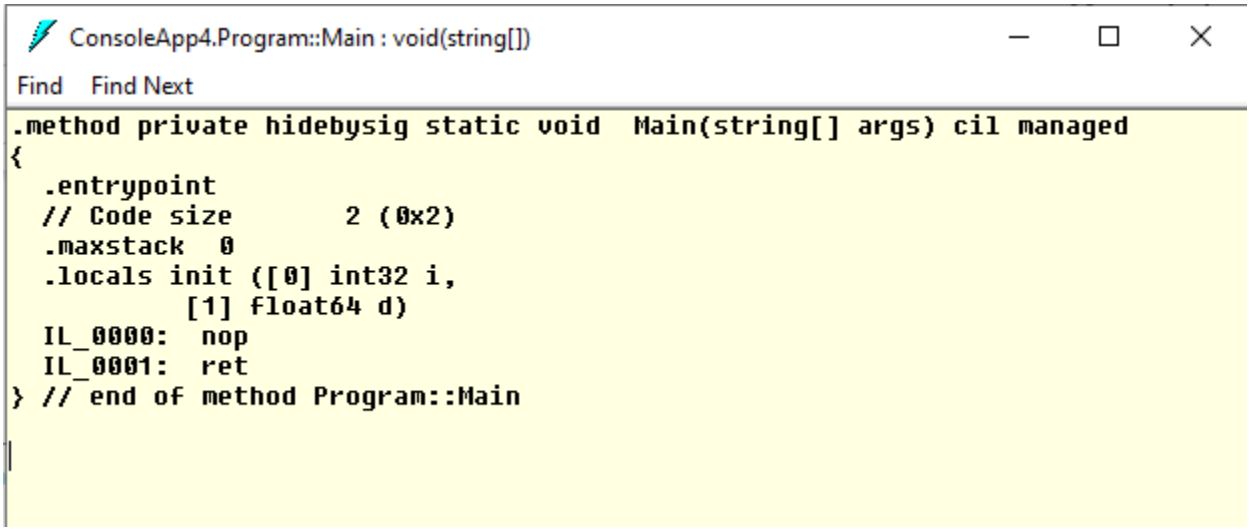
Output:-

Cmd to open ildasm
C:\”Program Files (x86)”\”Microsoft SDKs”\”Windows”\v8.1A\bin\”NETFX 4.5.1 Tools”>

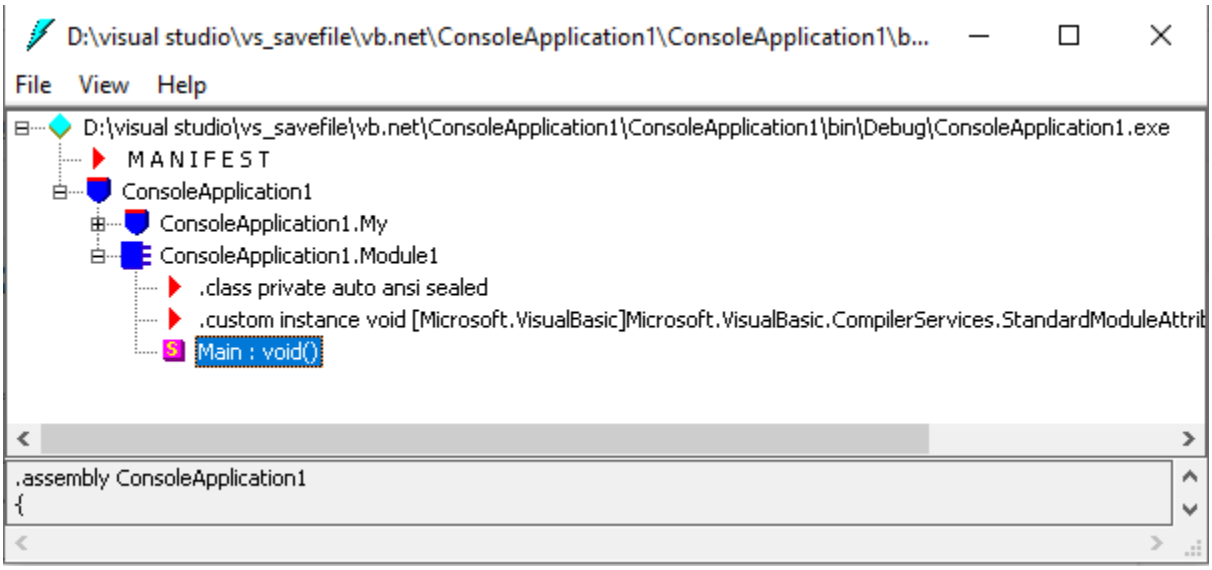


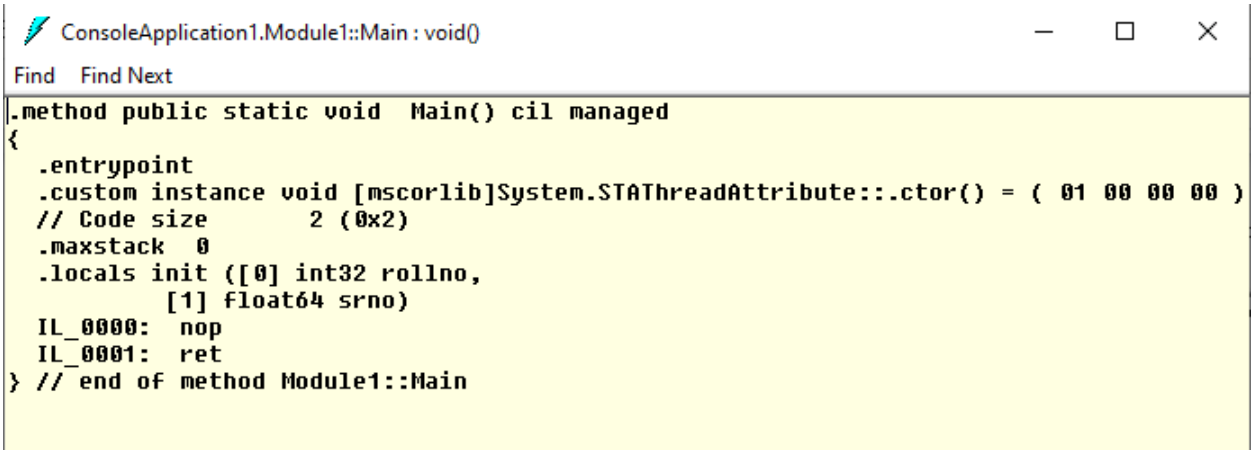
Opening c# exe file in ildasm





Vb.net .exe file open in ildasm


D:\visual studio\vs_savefile\vb.net\ConsoleApplication1\ConsoleApplication1\b...
File View Help
D:\visual studio\vs_savefile\vb.net\ConsoleApplication1\ConsoleApplication1\bin\Debug\ConsoleApplication1.exe
MANIFEST
ConsoleApplication1
ConsoleApplication1.My
ConsoleApplication1.Module1
.class private auto ansi sealed
.custom instance void [Microsoft.VisualBasic]Microsoft.VisualBasic.CompilerServices.StandardModuleAttrib
Main : void()
.assembly ConsoleApplication1
{


ConsoleApplication1.Module1::Main : void()
Find Find Next
.method public static void Main() cil managed
{
.entrypoint
.custom instance void [mscorlib]System.STAThreadAttribute::.ctor() = (01 00 00 00)
// Code size 2 (0x2)
.maxstack 0
.locals init ([0] int32 rollno,
[1] float64 srno)
IL_0000: nop
IL_0001: ret
} // end of method Module1::Main

27 Apr 2022

Practical 4:

Comments in c#

1. Single line
2. Multiline
3. Documentation

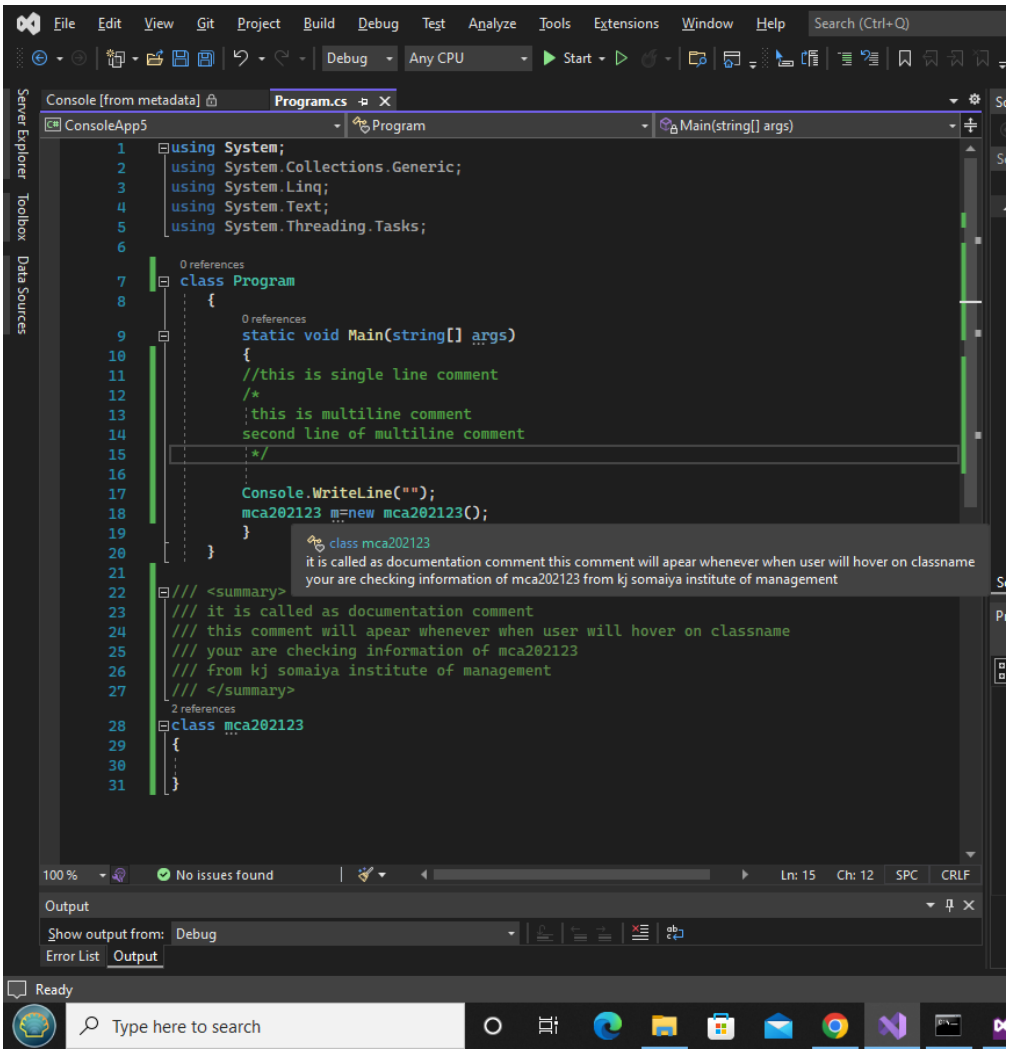
code:-

Consoleapp5

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

 class Program
    {
        static void Main(string[] args)
        {
        //this is single line comment
        /*
         this is multiline comment
        second line of multiline comment
         */
        Console.WriteLine("");
        mca202123 m=new mca202123();
        }
    }
/// <summary>
/// it is called as documentation comment
/// this comment will apear whenever when user will hover on classname
/// your are checking information of mca202123
/// from kj somaiya institute of management
/// </summary>
```

class mca202123
{ }

Output:-



27apr2022

Practical 5 & 6:

Demonstration of namespaces in single project(console application c#) and into different class library (in single executable and different dll's)
Namespaces and Class Library

a.Demonstration of namespaces in single file

Code:-

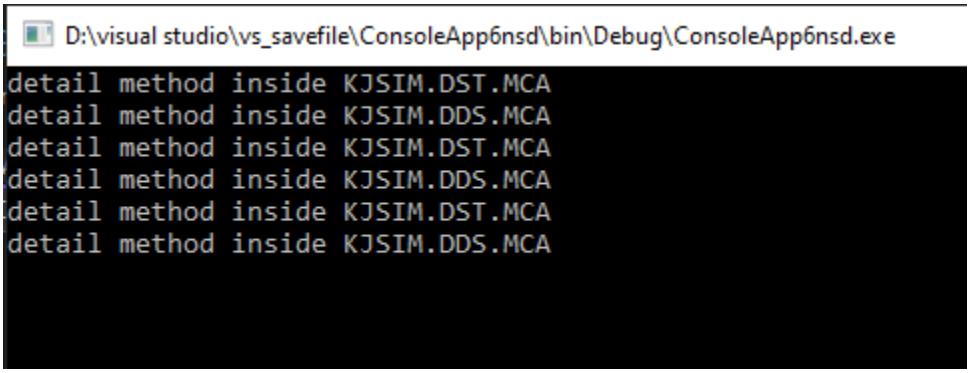
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using kjsimdst = kjsim.DST ;
using kjsimdds = kjsim.DDS;

using kjsimdstmca = kjsim.DST.MCA;
using kjsimddsmca = kjsim.DDS.MCA;

internal class Program
   {
//
       static void Main(string[] args)
       {
       //method1
       //using fullname for accessing class and methodsto
       //avoid ambiguity

       kjsim.DST.MCA.details();
       kjsim.DDS.MCA.details();


       //method 2
       /*
```

using alias for calliaccessing class and methods i.e. (calling & accessing class and methods)

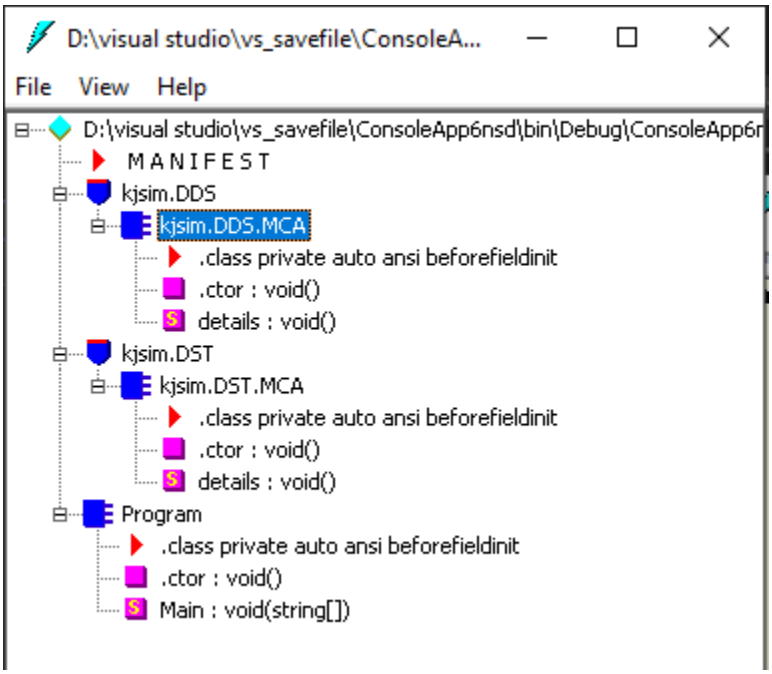
```
        using kjsimdst = kjsim.DST ;
        using kjsimdds = kjsim.DDS;
         */
        kjsimdst.MCA.details();
        kjsimdds.MCA.details();

        kjsimdstmca.details();
        kjsimddsmca.details();



        Console.ReadKey();
    }
    }

/// <summary>
/// kjsim.dst.mca
///this is a namespace
/// </summary>
namespace kjsim
{
    namespace DST
    {
        class MCA
        {
            public static void details()
            {
                Console.WriteLine("detail method inside DST");
            }
        }
    }
}
```

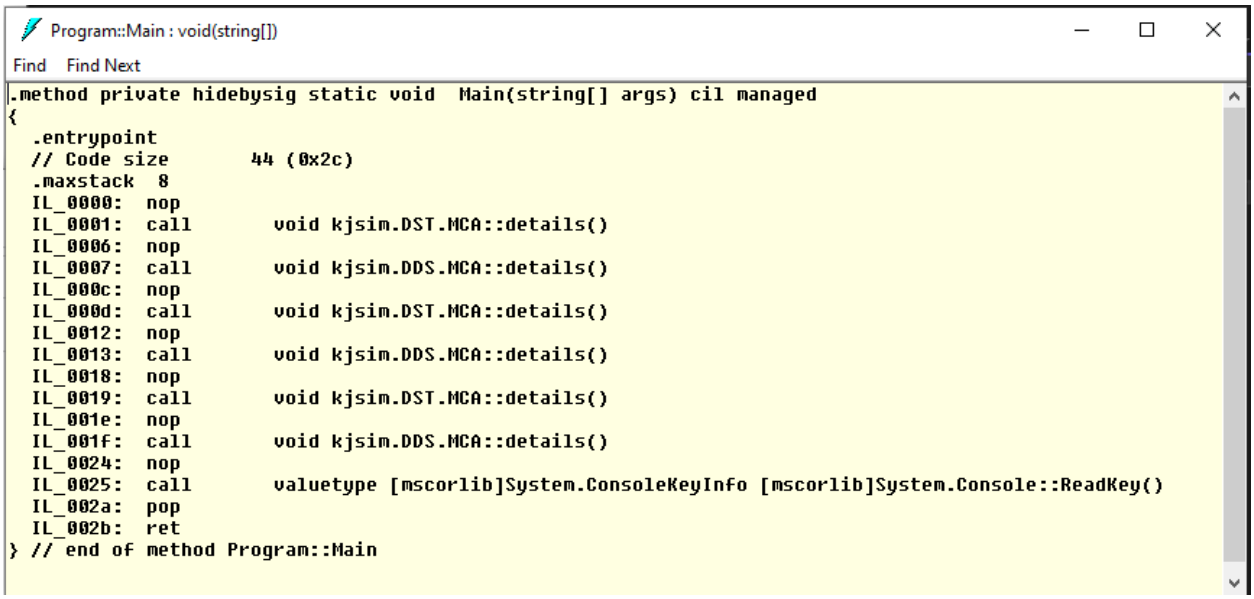
```
/// <summary>
/// kjsim.dds.mca
/// </summary>
namespace kjsim
{
    namespace DDS
    {
        class MCA
        {
            public static void details()
            {
                Console.WriteLine("detail method inside DDS");
            }
        }
    }
}
```

Output:-



In ildasm


D:\visual studio\vs_savefile\ConsoleA...
File View Help
D:\visual studio\vs_savefile\ConsoleApp6nsd\bin\Debug\ConsoleApp6r
MANIFEST
kjsim.DDS
kjsim.DDS.MCA
.class private auto ansi beforefieldinit
.ctor : void()
details : void()
kjsim.DST
kjsim.DST.MCA
.class private auto ansi beforefieldinit
.ctor : void()
details : void()
Program
.class private auto ansi beforefieldinit
.ctor : void()
Main : void(string[])


Program::Main : void(string[])
Find Find Next
.method private hidebysig static void Main(string[] args) cil managed
{
.entrypoint
// Code size 44 (0x2c)
.maxstack 8
IL_0000: nop
IL_0001: call void kjsim.DST.MCA::details()
IL_0006: nop
IL_0007: call void kjsim.DDS.MCA::details()
IL_000c: nop
IL_000d: call void kjsim.DST.MCA::details()
IL_0012: nop
IL_0013: call void kjsim.DDS.MCA::details()
IL_0018: nop
IL_0019: call void kjsim.DST.MCA::details()
IL_001e: nop
IL_001f: call void kjsim.DDS.MCA::details()
IL_0024: nop
IL_0025: call valuetype [mscorlib]System.ConsoleKeyInfo [mscorlib]System.Console::ReadKey()
IL_002a: pop
IL_002b: ret
} // end of method Program::Main

b.Demonstration of namespace class library with different class library

For different dll's

Right Click on solution explorer-> add->new Project-> c# class library-> rename library name to kjsim.mim then proceed and create/same for kjsim.mmm

Rename class1.cs to MCA.cs

Then add code in mca.cs

kjsim.mim->mca

code:

```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace kjsim.mim
{
    public class MCA
    {
        public static void print()
        {
            Console.WriteLine("print method inside kjsim.mim.print");
        }
    }
}
```

ksim.mmm->mca.cs

code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace kjsim.mmm
```

```
{
    public class MCA
    {
        public static void print()
        {
            Console.WriteLine("print method inside kjsim.mmm.print");
        }
    }
}
```

Consoleapp7

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using alias1 = kjsim.mim;
using alias2 = kjsim.mmm;

namespace ConsoleApplication7
{
    class Program
    {
        static void Main(string[] args)
        {
            alias1.MCA.print();
            alias2.MCA.print();
            Console.ReadKey();
        }
    }
}
```

Output:-

file:///D:/visual studio/vs_savefile/ConsoleApplication7/ConsoleApplication7/bin/Debug/ConsoleApplication7.EXE

```
print method inside kjsim.mim.print
print method inside kjsim.mmm.print
```

Check for dll file created or not

kjsim.mim

Debug

File Home Share View

This PC > New Volume (D:) > visual studio > vs_savefile > ConsoleApplication7 > kjsim.mim > bin > Debug

Name	Date modified	Type	Size
kjsim.mim.dll	4/28/2022 9:33 AM	Application exten...	4 KB
kjsim.mim.pdb	4/28/2022 9:33 AM	Program Debug D...	12 KB

Kjsim.mmm

Debug

File Home Share View

This PC > New Volume (D:) > visual studio > vs_savefile > ConsoleApplication7 > kjsim.mmm > bin > Debug

Name	Date modified	Type	Size
kjsim.mmm.dll	4/28/2022 9:33 AM	Application exten...	4 KB
kjsim.mmm.pdb	4/28/2022 9:33 AM	Program Debug D...	12 KB

28apr 2022

Practical 7:

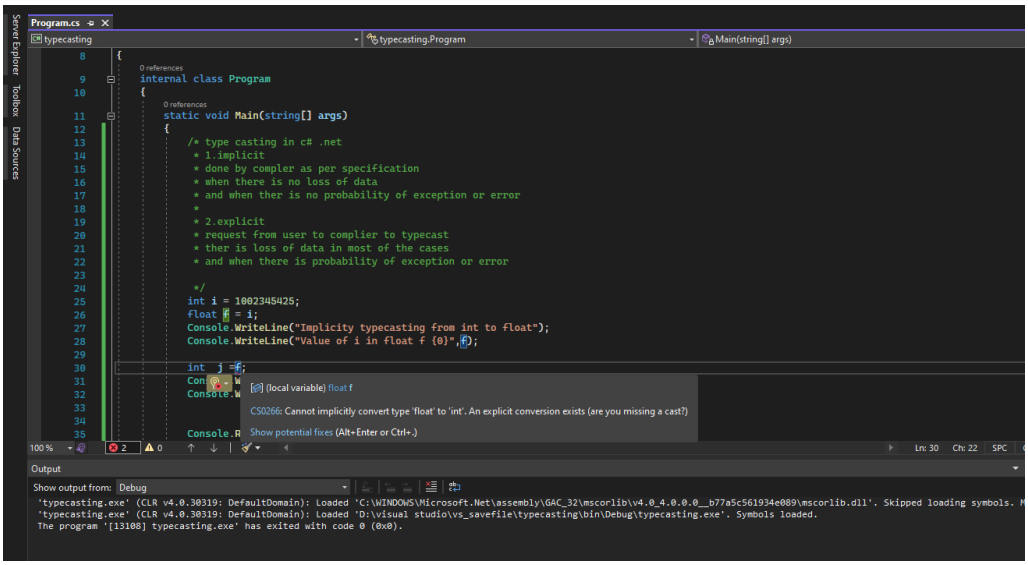
Demonstrate Typecasting in C#

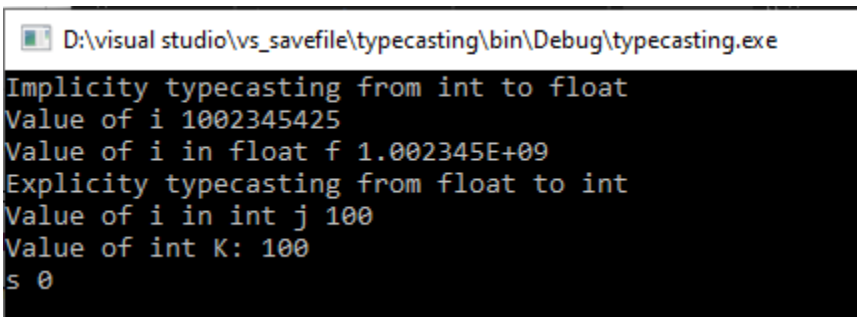
Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace typecasting
{
    internal class Program
    {
        static void Main(string[] args)
        {
            /* type casting in c# .net
             * 1.implicit
             * done by compiler as per specification
             * when there is no loss of data
             * and when there is no probability of exception or error
             *
             * 2.explicit
             * request from user to complier to typecast
             * there is loss of data in most of the cases
             * and when there is probability of exception or error

             */
            int i = 1002345425;
            float f = i;//implicity typecasting from int to float
            Console.WriteLine("Implicity typecasting from int to float");
            Console.WriteLine("Value of i {0}",i);
            Console.WriteLine("Value of i in float f {0}",f);
```

```
            float f2 = 100.243544f;
            int  j =(int)f2; //Explicity typecasting from float to int
            int k=Convert.ToInt32(f2);//by usuing cponvert.toInt32
            Console.WriteLine("Explicity typecasting from float to int");
            Console.WriteLine("Value of i in int j {0}", j);
            Console.WriteLine("Value of int K: {0}", k);


            //
            String s = "100th";
            //int pos=Convert.ToInt32(s);

            bool res=int.TryParse(s,out int i1);
            Console.WriteLine("s {0}",i1);
            Console.ReadKey();
        }
    }
}
```

Output:-


Program.cs
typecasting
typecasting.Program
Main(string[] args)
0 references
internal class Program
0 references
static void Main(string[] args)
/* type casting in c# .net
* 1.implicit
* done by compler as per specification
* when there is no loss of data
* and when ther is no probability of exception or error
*
* 2.explicit
* request from user to complier to typecast
* ther is loss of data in most of the cases
* and when there is probability of exception or error
*/
int i = 1002345425;
float f = i;
Console.WriteLine("Implicity typecasting from int to float");
Console.WriteLine("Value of i in float f {0}",f);
int j = f;
(local variable) float f
CS0266: Cannot implicitly convert type 'float' to 'int'. An explicit conversion exists (are you missing a cast?)
Show potential fixes (Alt+Enter or Ctrl+.)
Output
Show output from: Debug
'typecasting.exe' (CLR v4.0.30319: DefaultDomain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_32\mscorlib\v4.0_4.0.0.0__b77a5c561934e089\mscorlib.dll'. Skipped loading symbols.
'typecasting.exe' (CLR v4.0.30319: DefaultDomain): Loaded 'D:\visual studio\vs_savefile\typecasting\bin\Debug\typecasting.exe'. Symbols loaded.
The program '[13108] typecasting.exe' has exited with code 0 (0x0).


D:\visual studio\vs_savefile\typecasting\bin\Debug\typecasting.exe
Implicity typecasting from int to float
Value of i 1002345425
Value of i in float f 1.002345E+09
Explicity typecasting from float to int
Value of i in int j 100
Value of int K: 100
s 0

28apr2022

Practical 8:

Methods and types of parameters in c#
Call By Value
Call By Reference
Out parameter
Call by Params
Demonstration of Goto and break statements in methods

Code

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace methodsandparameters
{
    internal class Program
    {
        static void Main(string[] args)
        {
            //print("abcd","dst");//arguments passed
            //print("Saumik","DST");
            int c = 5;
            int d = 7;
            //call by value
            Console.WriteLine("Before call by value c:{0} d:{1}",c,d);
            printcbv(c, d);
            Console.WriteLine("after call by value c:{0} d:{1}", c, d);

            //call by reference
            Console.WriteLine("\nbefore call by reference c:{0} d: {1}", c, d);
            printcbr(ref c, ref d);
```

```
            Console.WriteLine("after call by reference c:{0} d: {1}",c, d);


        //call by out parameter
            int add,sub,mul,div;
            calc(23, 25, out add, out mul, out sub, out div);
            Console.WriteLine("\nadd:{0} mul:{1} sub:{2} div:{3} \n ", add, sub, mul,
div);

            //calling functions with params
            calculator("Saumik1");
            calculator("Saumik2",10, 20, 30);
            calculator("Saumik3",40, 50, 60, 70);

            //demonstration of goto continue and break statement
            ordercoffee(3);
            Console.ReadKey();
        }

        //type of method parameter
        //call by Value
        //call by reference
        //out parameter
        //
        /* public static void print(String name, String dept)
        {
            Console.WriteLine("Another Static Method of class program");
            // Console.WriteLine("name: {0}  dept:{1}",name,dept);

            Console.WriteLine("name:{0}  Dept:{1}", name, dept);
        } */
        public static void printcbv(int a,int b)
        {
            Console.WriteLine("Another Static Method of class program");
            // Console.WriteLine("name: {0}  dept:{1}",name,dept);
```

```
            a = a * 5;
            b= b * 7;
            Console.WriteLine("Value A:{0}  B:{1}",a,b);
        }

        public static void printcbr(ref int a, ref int  b)
        {
            Console.WriteLine("Another Static Method of class program");
            // Console.WriteLine("name: {0}  dept:{1}",name,dept);

            a = a * 5;
            b = b * 7;
            Console.WriteLine("reference A:{0}  B:{1}", a, b);
        }

        //function with out paramters to return multiple values
        public static void calc(int num1,int num2,out int add, out int sub,out int
mul,out int div)
        {
            //out is used to return values //out is essntial keyword if returning values
through function
           add = num1 + num2;
           sub = num1 - num2;
           mul = num1 * num2;
           div = num1 / num2;
        }

        /*function with param
        //should be always last parameter of your function
         */
        public static void calculator(String name,params int[] intArray)
        {
            Console.WriteLine("Function called by {0} with :{1} parameters",
name,intArray.Length);
```

```
    foreach(int i in intArray)
    {
        Console.WriteLine((int)i);
    }
}

//Demonstration of goto,continue and break statement
public static void ordercoffee(int i)
{
    start: //flag to check if yes or no
    switch(i)
    {
        case 0:
        case 2:
        case 3:
            Console.WriteLine("You ordered {0} cups of Coffee",i);
            break;
        default:
            Console.WriteLine("Your order number is invalid");
            break;
    }

    for (int j = 0; j < 10; j++)
    {
        if(j%2 !=0)
           continue;
        Console.WriteLine("{0}", j);
    }
    Console.WriteLine("Do you want to repeat again ");
    string flag=Console.ReadLine();
    if(flag=="yes")
    {
        goto start;
    }
    if (flag == "no")
```

```
        {
            Console.WriteLine("Thank you!! ");
        }
    }

  }
}
```

Output:-

```
D:\visual studio\vs_savefile\methodsandparameters\bin\Debug\methodsandparameters.exe
Before call by value c:5 d:7
Another Static Method of class program
Value A:25  B:49
after call by value c:5 d:7

before call by reference c:5 d: 7
Another Static Method of class program
reference A:25  B:49
after call by reference c:25 d: 49

add:48 mul:575 sub:-2 div:0

Function called by Saumik1 with :0 parameters
Function called by Saumik2 with :3 parameters
10
20
30
Function called by Saumik3 with :4 parameters
40
50
60
70
```

Goto break and continue statement

```
You ordered 3 cups of Coffee
0
2
4
6
8
Do you want to repeat again
no
Thank you!!
```

```
//function with out paramters to return multiple values
public static void calc(int num1,int num2,out int add, out int sub,out int mul,out int div)
{
    //out is used to return values //out is essntial keyword if returning values through function
   add = num1 + num2;
   sub = num1 - num2;
   mul = num1 * num2;
   div = num1 / num2;

}

/*function with param
//should be alwats last parameter of your function


 */
public static void calculator(params int[] intArray,String name)
{
    Console.WriteLine("Function call with :{0} pa
    foreach(int i in intArray)
    {
        Console.WriteLine((int)i);
    }
}
```

```
'methodsandparameters.exe' (CLR v4.0.30319: DefaultDomain): Loaded 'C:\WINDOWS\Microsoft.Net\assembly\GAC_32\mscorlib\v4.0_4.0.0.0__b77a5c561934e08
'methodsandparameters.exe' (CLR v4.0.30319: DefaultDomain): Loaded 'D:\visual studio\vs_savefile\methodsandparameters\bin\Debug\methodsandparameter
The program '[17020] methodsandparameters.exe' has exited with code 0 (0x0).
```

28apr2022

Practical 9:

Arrays and jagged array in C#

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace arrayjaggedarray
{
    internal class Program
    {
        static void Main(string[] args)
        {
            //simple array declaration
            int[] rollno = new int[5];
            rollno[0] = 1;
            rollno[1] = 2;
            rollno[2] = 3;
            rollno[3] = 4;
            rollno[4] = 5;

            foreach (int i in rollno)
            {
                Console.WriteLine("rollno{0}:",i);
            }

            Console.WriteLine();
            //skip the size of array while defining
            //size will be dynamic
            int[] arr1 = new int[] { 10,20,30,40,50};
```

```
            foreach (int j in arr1)
            {
                Console.WriteLine("Value:{0}",j);
            }

            Console.WriteLine();
            //you can skip new operator
            String[] subjects = { "web tech-2", "software engineering", "web tech-1"
,"Mobile development"};
            foreach (String k in subjects)
            {
                Console.WriteLine("Subject:{0}",k);
            }

            //multidimensional array
            // Console.WriteLine("Demonstration of Multidimensional array");
            int[,] months = new int[3, 3];// { {10,20, 30},{ 40,50 , 60 },{ 70,80,90  } };
            months[0, 0] = 20;
            months[0, 2] = 40;
            months[1, 2] = 30;
            months[2, 2] = 50;

            //multidimension with new operator
            int[,] md1 = new int[,] {
                { 1,2,3 },
                { 4,5,6 },
                { 7,8,9 } };

            //without new operator
            int[,] md2 = { {11,22,33 },{ 44,55,66} };

           //  Console.WriteLine(months.Length);

            for (int r = 0; r < months.GetLength(0); r++)
            {
```

```
    for (int s = 0; s <months.GetLength(1); s++)
    {
        Console.WriteLine("a[{0},{1}] : {2} ",r,s,months[r,s]);
    }
   // Console.WriteLine() ;
}

//md1
Console.WriteLine("md1");
for (int t = 0; t < md1.GetLength(0); t++)
{
    for (int u = 0; u < md1.GetLength(1); u++)
    {
        Console.Write("md1[{0},{1}] : {2} ", t, u, md1[t, u]);
    }
     Console.WriteLine() ;
}

//md2
Console.WriteLine("md2");
for (int v = 0; v < md2.GetLength(0); v++)
{
    for (int w = 0; w < md2.GetLength(1); w++)
    {
        Console.Write("md2[{0},{1}] : {2} ", v, w, md2[v, w]);
    }
    Console.WriteLine();
}



//jaggedarray with normal declaration
Console.WriteLine("Jagged Array");
int[][] vs = new int[3][]
{
```

```
    new int[]{0,1,2,3},
    new int[]{4,5,6,7,8},
    new int[]{9,10,11,12,13,14,15}
};

for(int x=0;x<vs.Length; x++)
{
    for (int y=0;y<vs[x].Length;y++)
    {
        //Console.Write("vs[{0}][{1}] : {2} ", t, u, vs[t][u]);
        Console.Write(" {0} ",vs[x][y]);
    }
    Console.WriteLine();
}

//without specifying size
Console.WriteLine("vs2");
int[][] vs2 = new int[][]
{
    new int[]{0,1,2,3},
    new int[]{4,5,6,7,8},
    new int[]{16,17}

};

for (int x = 0; x < vs2.Length; x++)
{
    for (int y = 0; y < vs2[x].Length; y++)
    {
        //Console.Write("vs[{0}][{1}] : {2} ", t, u, vs[t][u]);
        Console.Write(" {0} ", vs2[x][y]);
    }
    Console.WriteLine();
}
```

```
            //without new operator for row size
            Console.WriteLine("vs3");
            int[][] vs3={
                new int[]{01,11,22,33},
                new int[]{44,55,66,77,88},
                new int[]{99}

            };

            for (int x = 0; x < vs3.Length; x++)
            {
                for (int y = 0; y < vs3[x].Length; y++)
                {
                    //Console.Write("vs[{0}][{1}] : {2} ", t, u, vs[t][u]);
                    Console.Write(" {0} ", vs3[x][y]);
                }
                Console.WriteLine();
            }

            Console.ReadKey();
        }
    }
}
```

Output:-

D:\visual studio\vs_savefile\arrayjaggedarray\bin\Debug\arrayjaggedarray.exe

```
Subject:Mobile development
a[0,0] : 20
a[0,1] : 0
a[0,2] : 40
a[1,0] : 0
a[1,1] : 0
a[1,2] : 30
a[2,0] : 0
a[2,1] : 0
a[2,2] : 50
md1
md1[0,0] : 1 md1[0,1] : 2 md1[0,2] : 3
md1[1,0] : 4 md1[1,1] : 5 md1[1,2] : 6
md1[2,0] : 7 md1[2,1] : 8 md1[2,2] : 9
md2
md2[0,0] : 11 md2[0,1] : 22 md2[0,2] : 33
md2[1,0] : 44 md2[1,1] : 55 md2[1,2] : 66
Jagged Array
 0  1  2  3
 4  5  6  7  8
 9  10  11  12  13  14  15
vs2
 0  1  2  3
 4  5  6  7  8
 16  17
vs3
 1  11  22  33
 44  55  66  77  88
 99
```

29apr2022

Practical 10:

Demonstration of type safe and unsafe in C# using pointers
Demonstration of type safe and unsafe in C#
Code:-

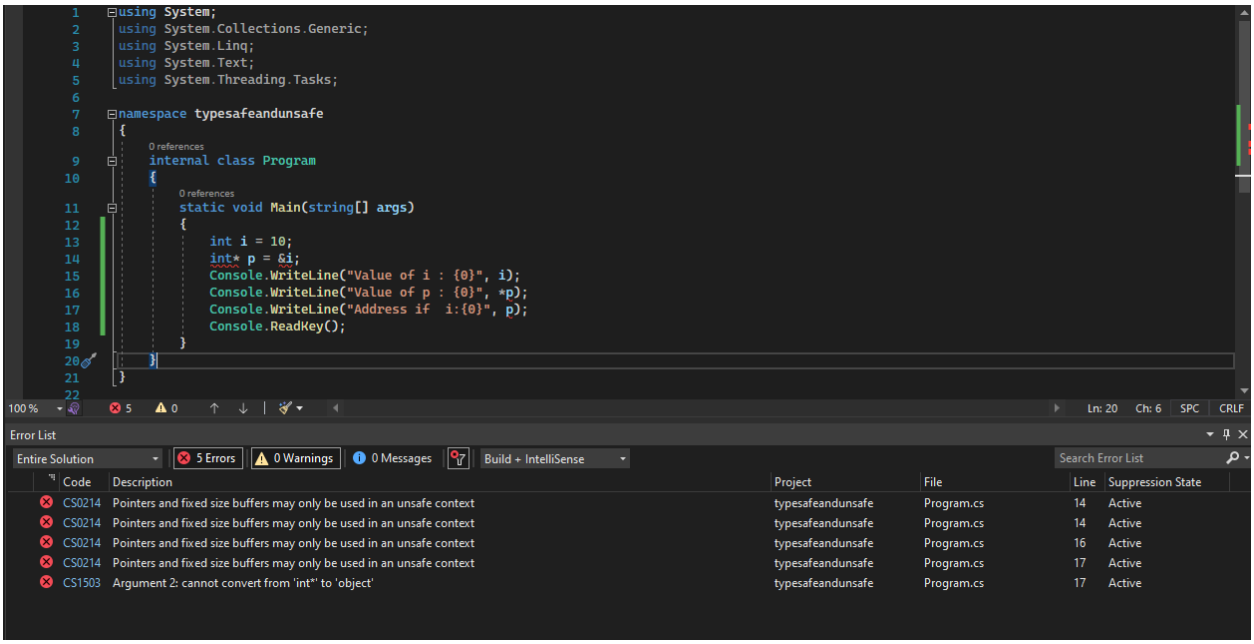
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace typesafeandunsafe
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int i = 10;
            int* p = &i;
            Console.WriteLine("Value of i : {0}", i);
            Console.WriteLine("Value of p : {0}", *p);
            Console.WriteLine("Address if  i:{0}", p);
            Console.ReadKey();
        }
    }
}
```

Using unsafe keyword

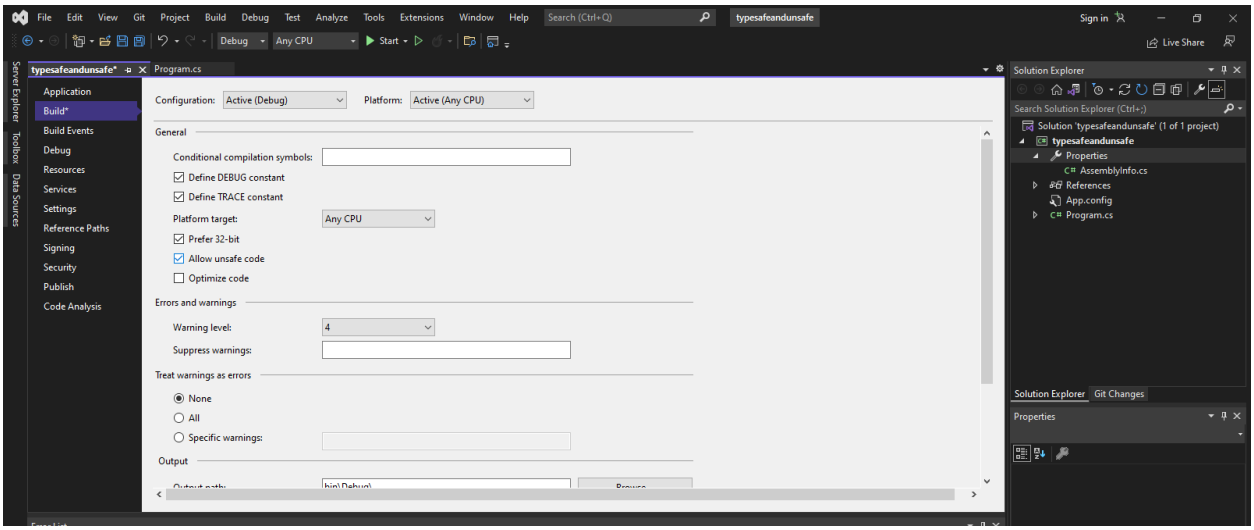
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;

namespace typesafeandunsafe
{
    internal class Program
    {
        unsafe static void Main(string[] args)
        {
            int i = 10;
            int* p = &i;
            Console.WriteLine("Value of i : {0}", i);
            Console.WriteLine("Value of p : {0}", *p);
            Console.WriteLine("Address if  i:{0}", (int)p);
            Console.ReadKey();
        }
    }
}
```

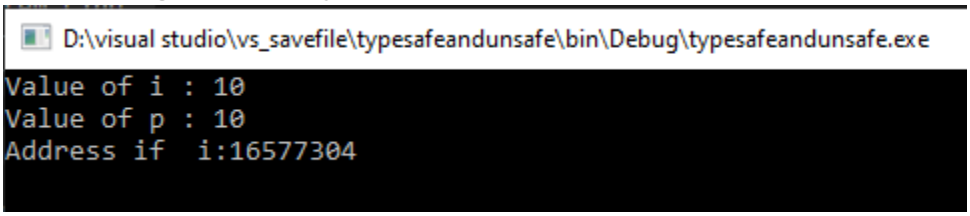
Output:



Double click on properties and goto build option and check on allow unsafe code



After using unsafe keyword



30 apr 2022

Practical 11:

Structure,class,fields,properties,constructor in C#

Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//structures in c#

class A
{
    int x = 5;
    int y = 10;
}

public struct MCA
{
   // public static int X=10;
   // public static int Y = 10;
    public  int X;
    public  int Y;


    public MCA(int i,int j)
    {
        X = i;
        Y = j;
    }
    public void show()
    {
```

```
        Console.WriteLine("Method of Struct {0}  {1}",X,Y);
    }
}
namespace ConsoleApp9
{

    internal class Program
    {
        static void Main(string[] args)
        {
            MCA mca = new MCA(10,20);
           // MCA mca = new MCA(sn);
            mca.show();
            Console.ReadKey();
        }
    }
}
```

//#2 code:- getter setter method

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//structures in c#


class A
{
    int x = 5;
    int y = 10;
}

public struct MCA
```

```
{
  // public static int X=10;
  // public static int Y = 10;
   public  int X;
   public  int Y;


   public MCA(int i,int j)
   {
       X = i;
       Y = j;
   }
   public void show()
   {
       Console.WriteLine("Method of Struct {0}  {1}",X,Y);
   }
}


public class employee
{
   //fields
   /*  public int age;
     public int empid;
     public string empname;

     public employee(int empid,string empname,int age)
     {
         this.age = age;
         this.empid = empid;
             this.empname = empname;

     }
    public int age=22;
   public int empid=101;
```

```
    public string empname="Saumik";

    */
    private int deptid; //cannot be negative or 0 and should be readonly
    private int empid; //cannot be negative or 0
    private string empname; //cannot be empty or 0


    // using properties instead of getter and setter methods directly
    //use value keyword when using this type of get and set method
    public int Empid
    { set { this.empid = empid <= 0 ? throw new Exception("id cannot be negative
or 0") : this.empid = value; } get { return empid; }  }

   public string Empname
    { set { this.empname = string.IsNullOrEmpty(empname) ? throw new
Exception(empname) : this.empname = value; } get { return empname; } }

    public int Deptid
    { get { return 1701; }  }


    //using getter and setter methods to access values

    public void setEmpname(string empname)
    {
        this.empname = string.IsNullOrEmpty(empname)||
string.IsNullOrWhiteSpace(empname) ? throw new Exception("Name cannot be
empty or null") :this.empname=empname;

      /*  if (string.IsNullOrEmpty(empname))
        {
            throw new Exception("name cannot be empty or null");
        }
```

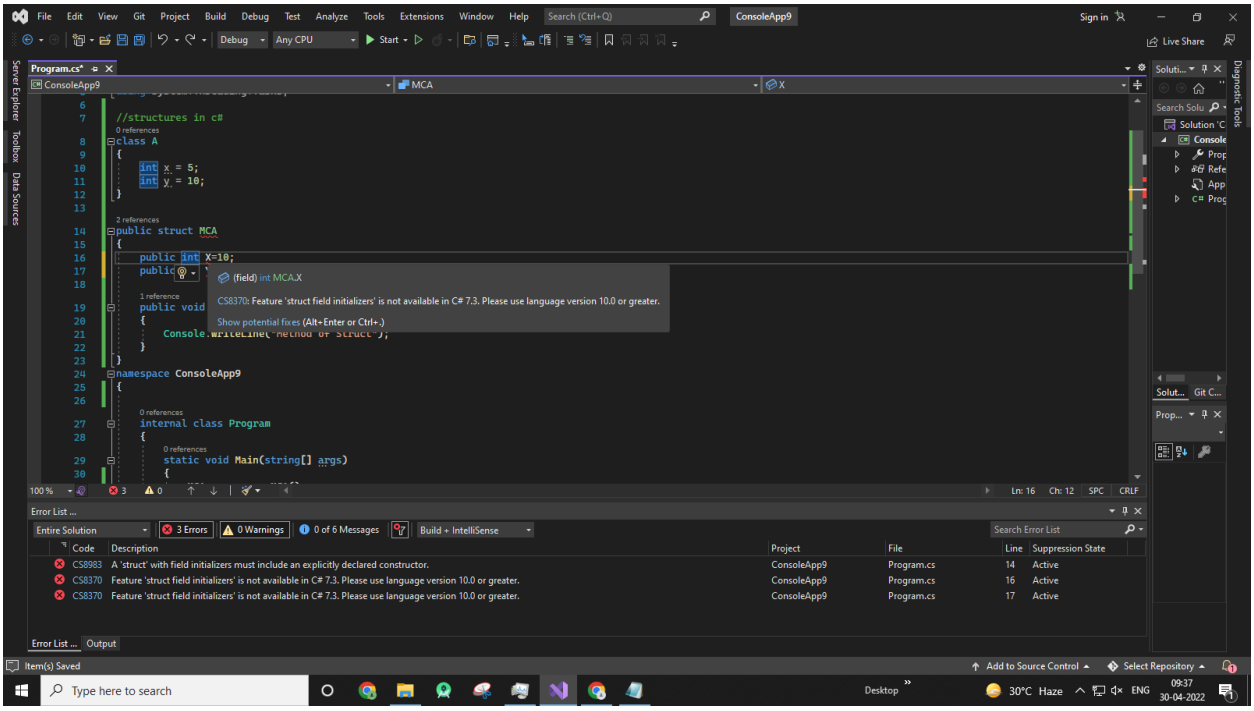
```
        else
        {
            this.empname = empname;
        }*/
    }


    public void setEmpid(int empid)
    {
        this.empid = empid<=0 ? throw new Exception("id cannot be negative or 0") :
this.empid = empid;
      /*  if (empid <= 0)
        {  throw new Exception("id cannot be negative or 0");   }
        else
        {  this.empid = empid;  } */
    }

    /*   public void setEmpdeptid(int deptid)
    {
        if (deptid <= 0)
        {    throw new Exception("deptid cannot be negative or 0 and should be
readonly");        }
        else
        {  this.deptid = deptid;  }
    }  */


    public int getEmpid()
    {
        return this.empid;
    }


    //when no setter method is provifded the field becomes readonly
    //when no getter method is provided the field becomes writeonly
```

```
    public int getEmpdeptid()
    {   return 1701;  }


    public string getEmpname()
    {   return this.empname;  }

    public void show()
   {
   // Console.WriteLine("Name:{0}deptid:{1}empid:{2}",empname,deptid,empid
);
      Console.WriteLine("Show method inside emp->show()\nName:{0}  deptid:{1}
empid:{2}\n", getEmpname(),getEmpdeptid(),getEmpid() );
      Console.WriteLine("Show method inside emp->show()\nName:{0}  deptid:{1}
empid:{2}\n", Empname,Deptid,Empid );

   }
}
namespace ConsoleApp9
{
  internal class Program
  {
      static void Main(string[] args)
      {
          // MCA mca = new MCA(10,20);
          // MCA mca = new MCA(sn);
          // mca.show();


          //parameterised constrtuctor
         // employee emp = new employee(101, "saumik", 22);
          employee emp = new employee();
           // emp.setEmpdeptid(1);
            emp.setEmpname("Saumik ");
            emp.setEmpid(101);
```

```
        emp.show();
        Console.WriteLine("outside show method in main method\nName:{0} deptid:{1}  empid:{2}\n", emp.getEmpname(), emp.getEmpdeptid(), emp.getEmpid());
        emp.Empid = 102;
        emp.Empname = "Saumik1";

        Console.WriteLine("outside show method in main method\nName:{0} deptid:{1}  empid:{2}\n", emp.Empname, emp.Deptid, emp.Empid);
        Console.ReadKey();
    }
  }
}
```

Output:-

Structure fields cannot be initialised in structure without constructor



Static fields of structure can be initialized without constructor

```
D:\visual studio\ConsoleApp9\ConsoleApp9\bin\Debug\ConsoleApp9.exe
Method of Struct
```

```
D:\visual studio\ConsoleApp9\ConsoleApp9\bin\Debug\ConsoleApp9.exe
Method of Struct 10  20
```

```
D:\visual studio\ConsoleApp9\ConsoleApp9\bin\Debug\ConsoleApp9.exe
Name:saumik  age:22  empid:101
```

//setter getter method

```
D:\visual studio\ConsoleApp9\ConsoleApp9\bin\Debug\ConsoleApp9.exe
Show method inside emp->show()
Name:Saumik   deptid:1701  empid:101

outside show method in main method
Name:Saumik   deptid:1701  empid:101
```

```
public string empname="Saumik";

*/
private int deptid; //cannot be negative or 0 and should be readonly
private int empid; //cannot be nagatiuve or 0
private string empname; //cannot be empty or 0

//using getter and setter methods to access values

1 reference
public void setEmpname(string empname)
{
    if (String.IsNullOrEmpty(empname) )
    {
        throw new Exception("name cannot be empty or null");
    }

    else
    {
        this.empname = empname;
    }
}


1 reference
public void setEmpid(int empid)
{
    if (empid <= 0)
    {
```

Exception Unhandled

System.Exception: 'name cannot be empty or null'

View Details | Copy Details | Start Live Share session...

Exception Settings

Name	Value	Type
empname	""	string
this	{employee}	employee

Call Stack

ConsoleApp9.exe!employee.setEmpname(string empname) Line 65

ConsoleApp9.exe!ConsoleApp9.Program.Main(string[] args) Line 147

```
    }

    else
    {
        this.empname = empname;
    }
}


1 reference
public void setEmpid(int empid)
{
    if (empid <= 0)
    {
        throw new Exception("id cannot be negative or 0");
    }

    else
    {
        this.empid = empid;
    }
}

/*
public void setEmpdeptid(int deptid)
{
    if (deptid <= 0)
    {
```

Exception Unhandled

System.Exception: 'id cannot be negative or 0'

View Details | Copy Details | Start Live Share session...

Exception Settings

Name	Value	Type
empid	-101	int
this	{employee}	employee

Call Stack

ConsoleApp9.exe!employee.setEmpid(int empid) Line 79

ConsoleApp9.exe!ConsoleApp9.Program.Main(string[] args) Line 148

02may2022/06May2022

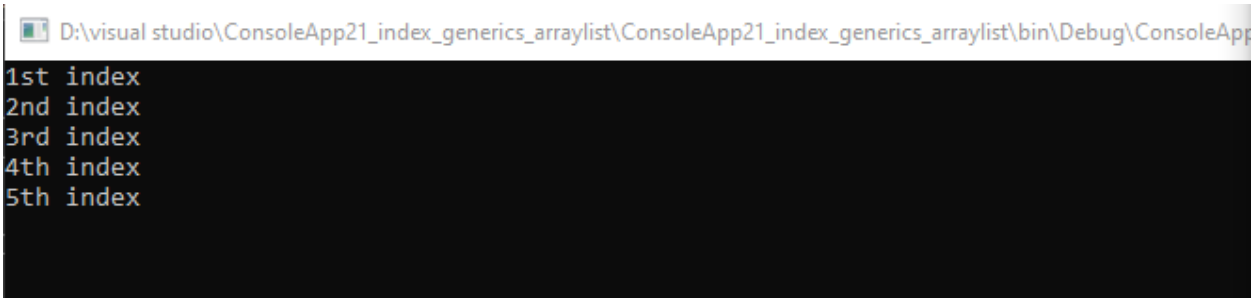
Practical 12:

Properties and Inheritance
Indexer property

Code:-

```
//indexer property
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ConsoleApp21_index_generics_arraylist
{
    internal class Program
    {
        static void Main(string[] args)
        {
            A a = new A();
            a[0] = "1st index";
            a[1] = "2nd index";
            a[2] = "3rd index";
            a[3] = "4th index";
            a[4] = "5th index";

            Console.WriteLine(a[0]);
            Console.WriteLine(a[1]);
            Console.WriteLine(a[2]);
            Console.WriteLine(a[3]);
            Console.WriteLine(a[4]);


            Console.ReadKey();
        }
```

```
    }

    class A
    {
        private /*static*/ string[] name=new string[10];

        public string this[int i]
        {
            get { return name[i]; }
            set { name[i] = value;}

        }



        public  void show()
        {
            Console.WriteLine("show function called");
            foreach(string name1 in name)
                Console.WriteLine(name1);
        }
    }


}
```



Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//properties and inheritance
//PropertiesIn_inheritance_polymorphism_static_abstract

class Base
{
    string name;
    public string Name
    {
        get {
            Console.WriteLine("Base-get");
            return name; }
        set {
            Console.WriteLine("Base-set");
            name = value; }
    }
}

class Derived : Base
{

}
namespace ConsoleApp10
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Derived derived = new Derived();
            derived.Name = "MCA202123";
```

```
            Console.WriteLine("name: {0}",derived.Name);

            Console.ReadKey();
        }
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp10\ConsoleApp10\bin\Debug\ConsoleApp10.exe
Base-set
Base-get
name: MCA202123
```

Accessing static method(use classname to access not object name)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;

//properties and inheritance
//PropertiesIn_inheritance_polymorphism_static_abstract

class Base
{
    string name;
    //static string dept;
    static string inst;

    //static properties accessd with class name not object
    public static string Inst
    {
        get { return inst; }
        set { inst = value; }
    }
    public string Name
    {
        get {
            Console.WriteLine("Base-get");
            return name; }
        set {
            Console.WriteLine("Base-set");
            name = value; }
    }
}

class Derived : Base
{

}
namespace ConsoleApp10
{
    internal class Program
```

```
    {
        static void Main(string[] args)
        {
            Derived derived = new Derived();
            derived.Name = "MCA202123";
            Derived.Inst = "KJ somaiya institute of management";
            Console.WriteLine("name: {0}\nInstitute:
{1}",derived.Name,Derived.Inst);

            Console.ReadKey();
        }
    }
}
```

D:\visual studio\ConsoleApp10\ConsoleApp10\bin\Debug\ConsoleApp10.exe

```
Base-set
Base-get
name: MCA202123
Institute: KJ somaiya institute of management
```

Creating readonly property
In class derive:base ie in base
//creating readonly property with private set accessor

```
    public string Dept
    {
```

```
        get;
        private set;
    } = "deparment of data science and technology";
```

In main

```
 Console.WriteLine("name: {0}\nInstitute:
{1}\ndept:{2}",derived.Name,Derived.Inst,derived.Dept);
```

```
D:\visual studio\ConsoleApp10\ConsoleApp10\bin\Debug\ConsoleApp10.exe

Base-set
Base-get
name: MCA202123
Institute: KJ somaiya institute of management
dept:deparment of data science and technology
```

02may2022

Practical 13:

Constructors and its type in c#

Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//constructor
/*
 1.called when object is created
2.having same name as classname
3.does not have return type ie void or int or string

4.comiler bydefault provides definition
if no user defined constructor available

5.in case of derived class constructor is the
first line of code is super()

6.constructor chaining is followed while creating
and deleteing child objects in inheritance

types
1.default(compiler provided)
2.default(user defined)
3.parameterised(user defined)
4.static constructor
5.copy constructor (compiler provided )
6.copy constructor (user defined)
7.private constructor
```

8.default parameterised/argument constructor
9.
*/

```
namespace ConsoleApp11_constructor
{
    class A
    {
        int id;
        string name;

        //user defined default constructor
        //public A()
        //{
        //    Console.WriteLine("A- user defined Default Constructor");
        //}
        //default parameterised constructor
        public A():this(0,"no name provided")
        {
            Console.WriteLine("A- user Default parameterised Constructor");
        }

        //parameterised constructor
        public A(int id,string name)
        {
            this.id = id;
            this.name = name;
        }

        public A(A b)
        {
            id = b.id;
            name = b.name;
        }
```

```
    public void show()
    {
        Console.WriteLine("A-show()");
        Console.WriteLine("id:{0}  name:{1}",id,name);
    }

    ~A()
    {
        Console.WriteLine("Destructor called");
    }

    //variable argiument
    public A(params int[] p){
        foreach (int i in p)
        {
            Console.WriteLine(i);
        }
    }
}


internal class Program
{
    static void Main(string[] args)
    {
        A a = new A(); a.show();
        A a1 = new A(1,"hello"); a1.show();


        A b=new A();
        b = a1; //calling default copy constructor
        b.show();

        A a2 = new A(2, "def");
```

```
            A a3 = new A(a2); //calling userdefined copy constructor

            a3.show();

            A a4 = new A(1,2,3,4,5);


            Console.ReadKey();
        }
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp11_constructor\ConsoleApp11_constructor\bin\Debug\ConsoleApp11_constructor.exe
A- user Default parameterised Constructor
A-show()
id:0  name:no name provided
A-show()
id:1  name:hello
A- user Default parameterised Constructor
A-show()
id:1  name:hello
A-show()
id:2  name:def
```

<u>Static constructor in c#</u>

```
//Static constructor in c#
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//constructor
/*
 1.called when object is created
2.having same name as classname
3.does not have return type ie void or int or string

4.comiler bydefault provides definition
if no user defined constructor available

5.in case of derived class constructor is the
first line of code is super()

6.constructor chaining is followed while creating
and deleteing child objects in inheritance

types
1.default(compiler provided)
2.default(user defined)
3.parameterised(user defined)
4.static constructor
5.copy constructor (compiler provided )
6.copy constructor (user defined)
7.private constructor
8.default parameterised/argument constructor
9.
 */
```

```
namespace ConsoleApp11_constructor
{
    class A
    {
        int id;
        string name;
        static string batch;
        static string subject;

        //user defined default constructor
        //public A()
        //{
        //    Console.WriteLine("A- user defined Default Constructor");
        //}



        //static constructor
        //called before all instance constructors
        //called only once
        //intialized the static DataMisalignedException member
        static A()
        {
            batch = "MCA2123";
            subject = "Web technology-2";

            Console.WriteLine("static called");

        }

        //there can be only one static constructor in any class
        static A()
        {
```

```
}

//default parameterised constructor
public A():this(0,"no name provided")
{
    Console.WriteLine("A- user Default parameterised Constructor");
}

//parameterised constructor
public A(int id,string name)
{
    this.id = id;
    this.name = name;
}

public A(A b)
{
    id = b.id;
    name = b.name;
}

public void show()
{
    Console.WriteLine("A-show()");
    Console.WriteLine("id:{0}  name:{1}",id,name);
}

~A()
{
    Console.WriteLine("Destructor called");
}

//variable argiument
public A(params int[] p){
    foreach (int i in p)
```

```
            {
                Console.WriteLine(i);
            }
        }
    }

    internal class Program
    {
        static void Main(string[] args)
        {
            A a = new A(); a.show();
            A a1 = new A(1,"hello"); a1.show();


            A b=new A();
            b = a1; //calling default copy constructor
            b.show();

            A a2 = new A(2, "def");
            A a3 = new A(a2); //calling userdefined copy constructor

            a3.show();

            A a4 = new A(1,2,3,4,5);


            Console.ReadKey();
        }
    }
}
```

Output:-

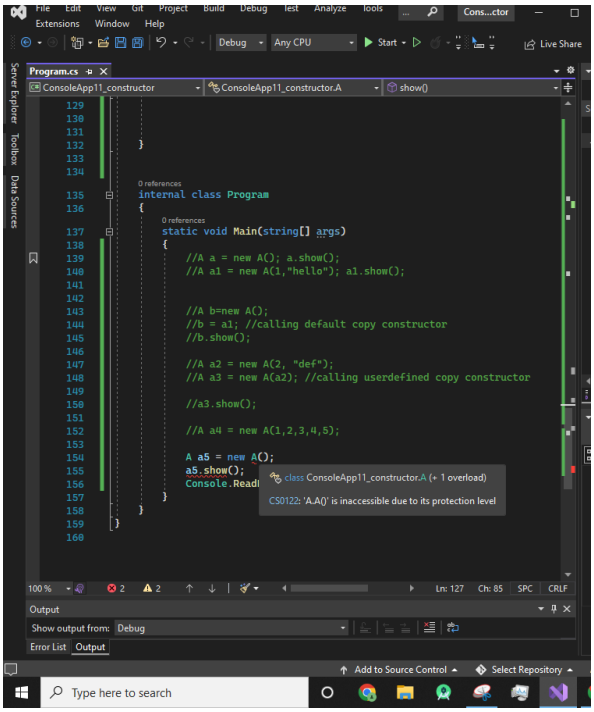
D:\visual studio\ConsoleApp11_constructor\ConsoleApp11_constructor\bin\Debug\ConsoleApp11_constructor.exe

```
static called
A- user Default parameterised Constructor
A-show()
id:0  name:no name provided
A-show()
id:1  name:hello
A- user Default parameterised Constructor
A-show()
id:1  name:hello
A-show()
id:2  name:def
1
2
3
4
5
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//constructor
/*
 1.called when object is created
2.having same name as classname
3.does not have return type ie void or int or string

4.comiler bydefault provides definition
if no user defined constructor available

5.in case of derived class constructor is the
first line of code is super()

6.constructor chaining is followed while creating
and deleteing child objects in inheritance

types
1.default(compiler provided)
2.default(user defined)
3.parameterised(user defined)
4.static constructor
5.copy constructor (compiler provided )
6.copy constructor (user defined)
7.private constructor
8.default parameterised/argument constructor
9.
 */


namespace ConsoleApp11_constructor
```

```
{
    class A
    {


        /**********************************************/
        static int inst_code;
        static string inst_name;

        static A()
        {
            inst_code = 365;
            inst_name = "KJSIM";
        }
        private A()
        {

        }

        public static void show()
        {
            Console.WriteLine("inst_code:{0}  inst_name:{1}", inst_code, inst_name);
        }



    }


    internal class Program
    {
        static void Main(string[] args)
        {
```

```
            A a5 = new A();
            a5.show();
            Console.ReadKey();
        }
    }
}
```



—----------

Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp11_constructor
{
    class A
    {


        /*************************************************/
        static int inst_code;
        static string inst_name;
        private A()
        {
            Console.WriteLine("calling private constructor");
        }
        static A()
        {
            inst_code = 365;
            inst_name = "KJSIM";
        }


        public static void show()
        {
            Console.WriteLine("inst_code:{0}  inst_name:{1}", inst_code, inst_name);
        }
```

```
    }


    internal class Program
    {
        static void Main(string[] args)
        {

            A.show();
            Console.ReadKey();
        }
    }
}
```

D:\visual studio\ConsoleApp11_constructor\ConsoleApp11_constructor\bin\Debug\ConsoleApp11_constructor.exe

```
inst_code:365  inst_name:KJSIM
```

constructor chaining

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//constructor
/*
 1.called when object is created
2.having same name as classname
3.does not have return type ie void or int or string
4.compiler by default provides definition if no user defined constructor available
5.in case of derived class constructor is the first line of code is super()
6.constructor chaining is followed while creating and deleteing child objects in
inheritance

types
1.default(compiler provided)
2.default(user defined)
3.parameterised(user defined)
4.static constructor
5.copy constructor (compiler provided )
6.copy constructor (user defined)
7.private constructor
8.default parameterised/argument constructor
9.
 */


namespace ConsoleApp11_constructor
{
    class Base
    {
        public Base()
```

```
        {           Console.WriteLine("Base() called");        }
    }

    class Derived:Base
    {
        public Derived()
        {           Console.WriteLine("Derived() called");        }
    }

    class Derivedtocall : Derived
    {
        public Derivedtocall()
        {           Console.WriteLine("Derivedtocall() called");        }
    }

    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Constructor chaining");
            Derivedtocall derived = new Derivedtocall();
            Console.ReadKey();
        }
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp11_constructor\ConsoleApp11_constructor\bin\Debug\ConsoleApp11_constructor.exe
Constructor chaining
Base() called
Derived() called
Derivedtocall() called
```

02may2022

Practical 14:

Inheritance in c#

Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp11_constructor
{
      class Base
    {
        protected string name;

        public Base()
        {
            Console.WriteLine("Base() called");
        }
    }

    class Derived:Base
    {
        protected string city;

        public Derived()
        {
            Console.WriteLine("Derived() called");
        }
    }

    class Derivedtocall : Derived
```

```
    {
        string state;
        public Derivedtocall(string name,string city,string state)
        {
            this.name = name;
            this.city = city;
                this.state = state;
            Console.WriteLine("Derivedtocall() called");
        }

        public void show()
        {
            Console.WriteLine("name:{0} city:{1} state:{2}", this.name, this.city,
this.state);
        }
    }

    internal class Program
    {
        static void Main(string[] args)
        {
           Console.WriteLine("Inheritance");
            Derivedtocall derived = new
Derivedtocall("saumik","vashi","maharashtra");
            derived.show();

            Console.ReadKey();
        }
    }
}
```

Output:-

D:\visual studio\ConsoleApp11_constructor\ConsoleApp11_constructor\bin\Debug\ConsoleApp11_constructor.exe

```
Inheritance
Base() called
Derived() called
Derivedtocall() called
name:saumik city:vashi state:maharashtra
```

02may2022

Practical 15:

Method hiding in c#

Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
class Base
{
    public void show()
    {
        Console.WriteLine("base class -show()");
    }
}

class Derived : Base
{
    //method hiding
    //using new keyword is optional but usage is standard
    public new void show()
    {
       // base.show();//calling default behavious of base
                    //class show method
        Console.WriteLine("Derived class -show()");
    }
}

namespace ConsoleApp12_method_hiding
{
    internal class Program
    {
        static void Main(string[] args)
```

```
        {
            Derived d=new Derived();
            d.show();

            //base class refernece variable to refer to  base class
            Base b =new Derived();
            b.show();

            ((Base)b).show();//typecasting methods to access b to base type
            Console.ReadKey();
        }
    }
}
```

Output:-

Using base.show();

```
D:\visual studio\ConsoleApp12_method_hiding\ConsoleApp12_method_hiding\bin\Debug\ConsoleApp12_method_hiding.exe
base class -show()
Derived class -show()
```

Using base class object to access child class object

Base b =new Derived();

b.show();

```
D:\visual studio\ConsoleApp12_method_hiding\ConsoleApp12_method_hiding\bin\Debug\ConsoleApp12_method_hiding.exe
base class -show()
```

Using typecasting method

((Base)b).show();

D:\visual studio\ConsoleApp12_method_hiding\ConsoleApp12_method_hiding\bin\Debug\ConsoleApp12_method_hiding.exe

```
base class -show()
```

```
9       {
10          Console.WriteLine("base class -show()");
11      }
12  }
13
    0 references
14  class Derived : Base
15  {
16
        0 references
17      public void show()
18      {
19          Console.Wr
20      }
21
22
23  }
24
25  namespace ConsoleApp12_method_hiding
26  {
        0 references
27      internal class Program
28      {
            0 references
29          static void Main(string[] args)
30          {
31          }
32      }
33  }
34
```

void Derived.show()

CS0108: 'Derived.show()' hides inherited member 'Base.show()'. Use the new keyword if hiding was intended.

IDE1006: Naming rule violation: These words must begin with upper case characters: show

Show potential fixes (Alt+Enter or Ctrl+.)

02may2022

Practical 16:

Polymorphism using method overriding in c#

Code;-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
class Base
{
    public virtual void show()
    {
        Console.WriteLine("base class -show()");
    }
}

class Derived1 : Base
{
    //public  void show()
    //{
    //    //base.show();
    //    Console.WriteLine("Derived1 -show()");
    //}
    public override void show()
    {
        //base.show();
        Console.WriteLine("derived1 -show()");
    }

}

class Derived2 : Base
```

```
{
    //public void show()
    //{
    //    base.show();
    //    Console.WriteLine("Derived2 -show()");
    //}

    public override void show()
    {
        //base.show();
        Console.WriteLine("derived2 -show()");
    }
}

class Derived3 : Base
{
    //public  void show()
    //{
    //    //base.show();
    //    Console.WriteLine("Derived3 -show()");
    //}

    public override void show()
    {
        //base.show();
        Console.WriteLine("Derived3 -show()");
    }
}
namespace ConsoleApp13_polymorphism
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Base[] bases = new Base[4];
```

```
            bases[0]=new Base();
            bases[1]=new Derived1();
            bases[2]=new Derived2();
            bases[3]=new Derived3();

            foreach(Base b in bases)
            {
                b.show();
            }


            Console.ReadKey();
        }
    }
}
```

Output:-

Without using override keyword

```
D:\visual studio\ConsoleApp13_polymorphism\ConsoleApp13_polymorphism\bin\Debug\ConsoleApp13_polymorphism.exe
base class -show()
base class -show()
base class -show()
base class -show()
```

Using override keyword

```
D:\visual studio\ConsoleApp13_polymorphism\ConsoleApp13_polymorphism\bin\Debug\ConsoleApp13_polymorphism.exe
base class -show()
derived1 -show()
derived2 -show()
Derived3 -show()
```

04may2022

Practical 17:

Abstract class in c#

Code:-

```
/*
 *abstraction
 *definition-hiding implementation details from user
 *0% abstraction concrete class
 *1-99% abstraction-abstract class
 *100% abstraction - interfaces
 */


/*
 Abstract class
1. use abstract keyword to declare/define abstract class or methods
2.abstract class may or may not have abstract methods in c#
3.can have concrete methods
4.can have constants
5.can have constructors and destructors
6.normally used for creating base classes
7.we can create reference var of abstract class
8.we can't instantiate(can't create object) of abstract class
9.abstract classes or method cannot be sealed
10.abstract class cant be static
11.abstract methods of abstract class are virtual by defult
12.in order to inherit from abstract class derived class has to
implement abstract method or declare itself as abstract
13.destructor cannot have public or private modifier explicitly
14.abstract class can have abstract property
15.if only get accessor is declared in abstract property
   its readonly field
  set cannot be inherited since not declared
```

```
     */

—----------------------------------------------------------------------------------------------------
1.cannot create object of abstract class
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp14_abstract
{
    /*
     *abstraction
     *definition-hiding implementation details from user
     *0% abstraction concrete class
     *1-99% abstraction-abstract class
     *100% abstraction - interfaces
     */


    /*
     Abstract class
    1. use abstract keyworkd to declare/define abstract class or methods
    2.abstract class may or may not have abstract methods in c#
    3.can have concrete methods
    4.can have constants
    5.can have constructors and destructors
    6.normally used for creating base classes
    7.we can create reference var of abstract class
    8.we can't instantiate(can't create object) of abstract calss
    9.abstract classess or method cannot be sealed
    10.abstract class cant be static
    11.abstract methods of abstract class are virtual by defult
```

```
 */


abstract class abs

{
    public abs()
    {
        Console.WriteLine("Abstract() - constructor called");
    }

    ~abs()
    {
        Console.WriteLine("abs()- destructor callled");
    }

    public void show()
    {
        Console.WriteLine("show method called");
    }

    public abstract void run();
}



class derived : abs
{
    public override void run()
    {
        Console.WriteLine("derived-run() called");

    }
}
internal class Program
```

```
    {
        static void Main(string[] args)
        {

            abs a=new abs();

            Console.ReadKey();
        }
    }
}
```

2.cannot be sealed or static

```
 using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp14_abstract
{
    /*
     *abstraction
     *definition-hiding implementation details from user
     *0% abstraction concrete class
     *1-99% abstraction-abstract class
     *100% abstraction - interfaces
     */



    /*
     Abstract class
    1. use abstract keyworkd to declare/define abstract class or methods
    2.abstract class may or may not have abstract methods in c#
    3.can have concrete methods
    4.can have constants
```

5.can have constructors and destructors
6.normally used for creating base classes
7.we can create reference var of abstract class
8.we can't instantiate(can't create object) of abstract class
9.abstract classes or method cannot be sealed
10.abstract class cant be static
11.abstract methods of abstract class are virtual by default

```
 */


sealed abstract class abs

{
    public abs()
    {
        Console.WriteLine("Abstract() - constructor called");
    }

    ~abs()
    {
        Console.WriteLine("abs()- destructor callled");
    }

    public const string dept = "dept of data science and technology";

    public void show()
    {
        Console.WriteLine("abs-show() method called\n" +
            "concrete method");
    }

    public abstract void run();
}
```

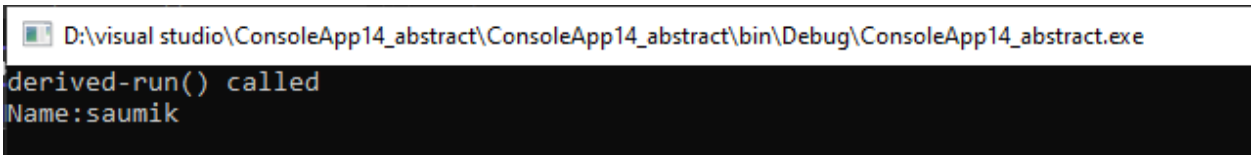
```
    class derived : abs
    {
        public override void run()
        {
            Console.WriteLine("derived-run() called");

        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {

           // abs a=new abs();
            abs a=new derived();//cre3ating reference type of abstract calss
            //
            a.run();

            Console.ReadKey();
        }
    }
}
```

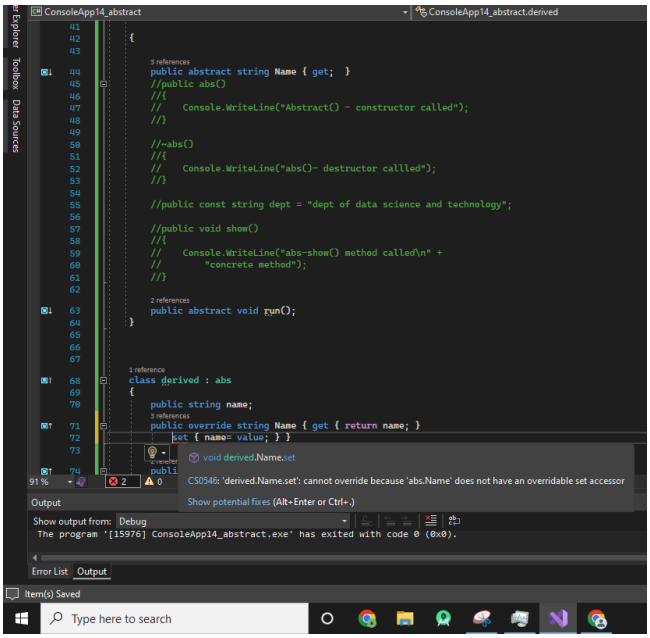
4.abstract property

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp14_abstract
{
    /*
     *abstraction
     *definition-hiding implementation details from user
     *0% abstraction concrete class
     *1-99% abstraction-abstract class
     *100% abstraction - interfaces
     */




   abstract class abs

    {

        public abstract string Name { get; set; }
        public abs()
        {
            Console.WriteLine("Abstract() - constructor called");
        }

        ~abs()
        {
            Console.WriteLine("abs()- destructor callled");
        }
```

```
    public const string dept = "dept of data science and technology";

    public void show()
    {
        Console.WriteLine("abs-show() method called\n" +
            "concrete method");
    }

    public abstract void run();
}



class derived : abs
{
    public string name;
    public override string Name { get { return name; } set { name= value; } }

    public override void run()
    {
        Console.WriteLine("derived-run() called");
        Console.WriteLine("Name:{0}", this.Name);

    }
}

//class derived1:abs
//{

//}
internal class Program
{
    static void Main(string[] args)
    {
```

```
        // abs a=new abs();
         abs a=new derived();//creating reference type of abstract class
         //
         a.Name = "saumik";
         a.run();

         Console.ReadKey();
      }
   }
}
```





4.

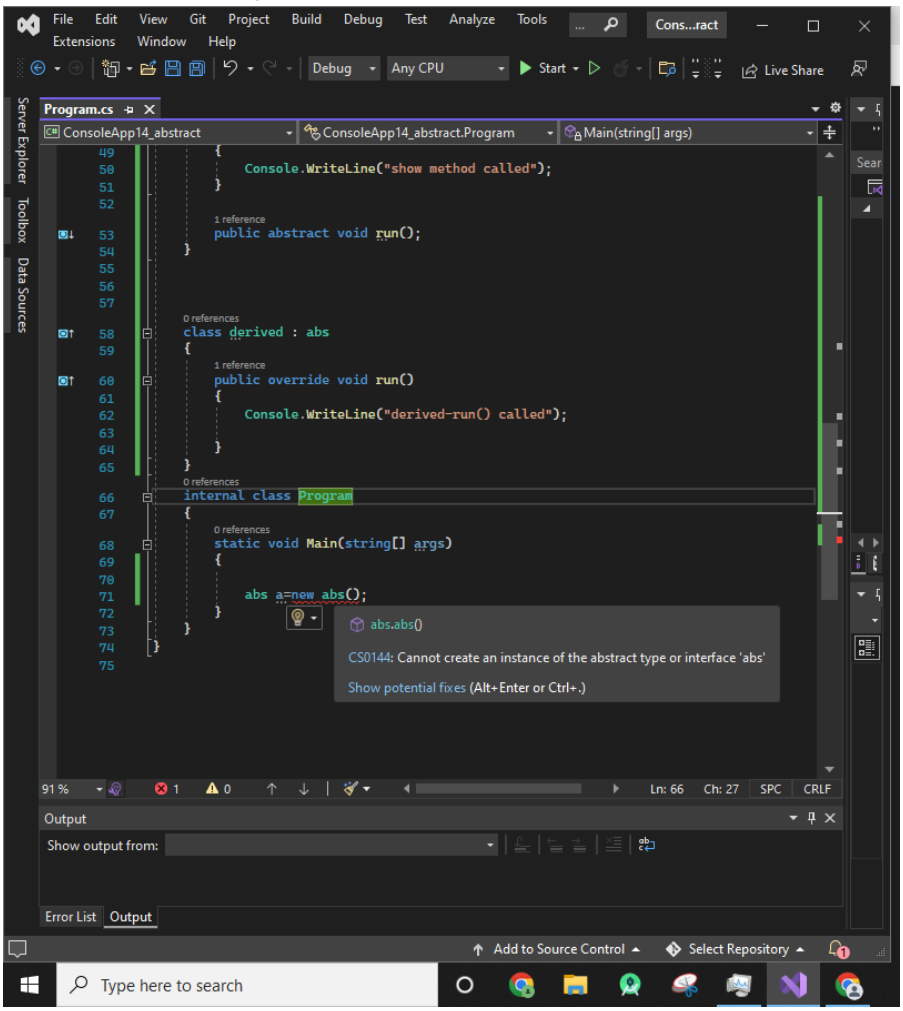
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp14_abstract
{
  abstract class abs
    {
        public abstract string Name { get;/* set;*/ }
       public abstract void run();
    }



    class derived : abs
    {
        public string name;
        public override string Name { get { return "Saumik Dutta"; }
            /*set { name= value; }*/ }

        public override void run()
        {
            Console.WriteLine("derived-run() called");
            Console.WriteLine("Name:{0}", this.Name);

        }
    }

    //class derived1:abs
    //{

    //}
```

```
    internal class Program
    {
        static void Main(string[] args)
        {

            // abs a=new abs();
             abs a=new derived();//cre3ating reference type of abstract class
             //
            // a.Name = "saumik";
             a.run();

             Console.ReadKey();
        }
    }
}
```

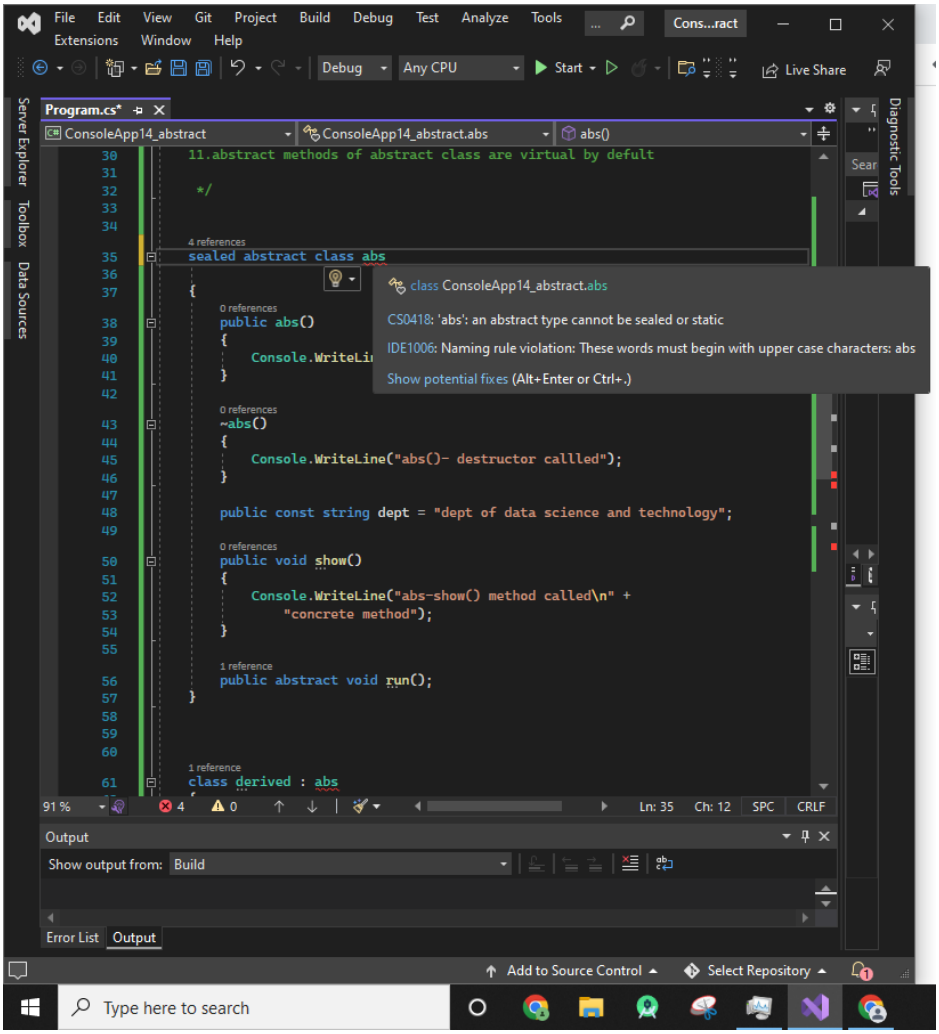
```
D:\visual studio\ConsoleApp14_abstract\ConsoleApp14_abstract\bin\Debug\ConsoleApp14_abstract.exe
derived-run() called
Name:Saumik Dutta
```

Output:-

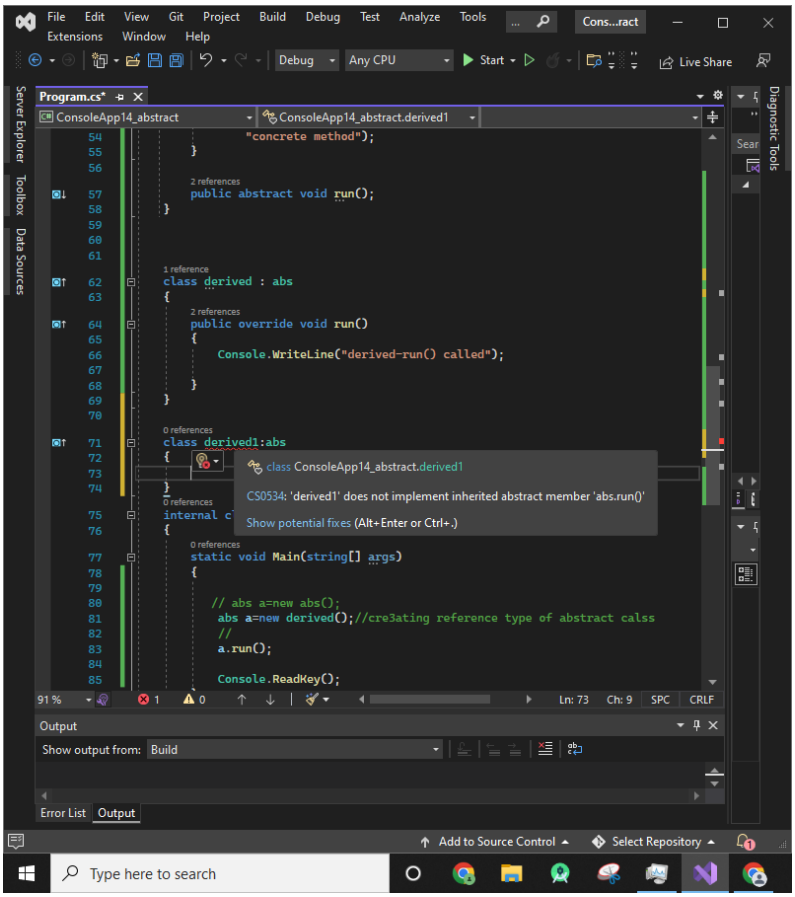
Cannot create object of abstract class



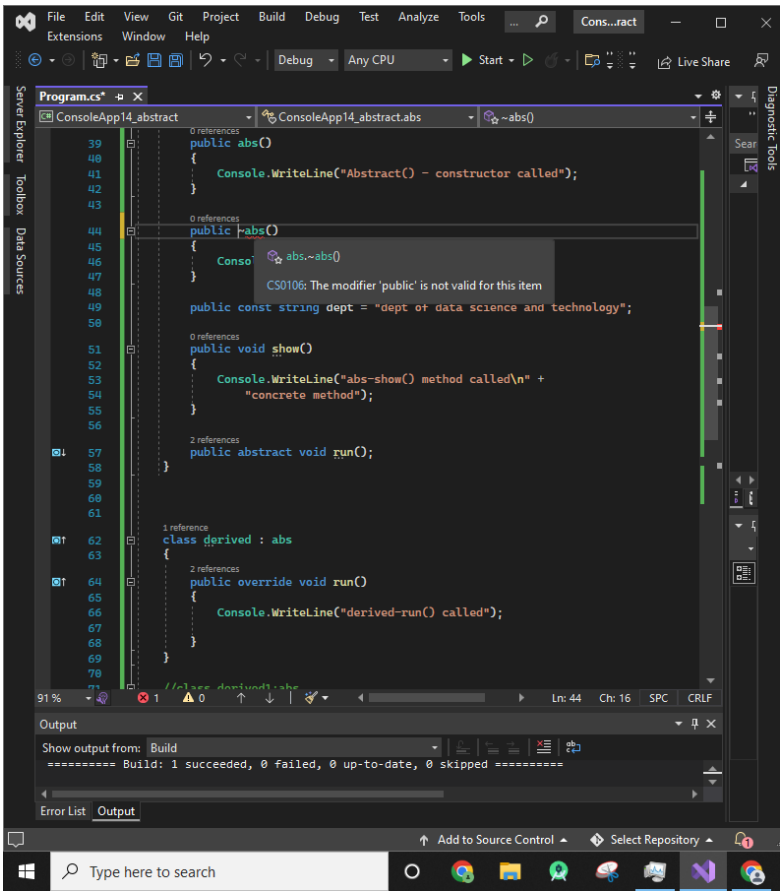
2.cannot be sealed or static



Implementation rule



Destructor cannot have modifier explicitly


File Edit View Git Project Build Debug Test Analyze Tools
Extensions Window Help
Cons...ract
Debug
Any CPU
Start
Live Share
Program.cs*
ConsoleApp14_abstract
ConsoleApp14_abstract.abs
~abs()
0 references
public abs()
{
Console.WriteLine("Abstract() - constructor called");
}
0 references
public ~abs()
abs.~abs()
CS0106: The modifier 'public' is not valid for this item
public const string dept = "dept of data science and technology";
0 references
public void show()
{
Console.WriteLine("abs-show() method called\n" +
"concrete method");
}
2 references
public abstract void run();
}
1 reference
class derived : abs
{
2 references
public override void run()
{
Console.WriteLine("derived-run() called");
}
}
Ln: 44
Ch: 16
SPC
CRLF
Output
Show output from: Build
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
Error List
Output
Add to Source Control
Select Repository
Type here to search

02may2022

Practical 18:

Interface:-

```
/*
 interface
0 supports multiple inheritance
1.100% abstraction
2.cant have method definition
3.by default methods inside interface are abstract in nature
4.explicit use of abstract modifier is not allowed
5.instance data member are not allowed
 6.interface can inherit from another interface without defining member function
7.cannot instantiate or create object or interface
8.reference type of interface can be created
9.explicit implementation of interface methods with smae name in case of explicit
 */
```

Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/*
 interface
0 supports multiple inheritance
1.100% abstraction
2.cant have method definition
3.by default methods inside interface are abstract in nature
4.explicit use of abstract modifier is not allowed
5.instance data member are not allowed
 6.interface can inherit from another interface without defining member function
7.cannot instantiate or create object or interface
8.reference type of interface can be created
```

9.explicit implementation of interface methods with smae name in case of explicit
10.to call explicitly defined method typecasting is neccessary
11.default explicit implementation is possible for only one interface
and can be called without typecasting of derved class object
*/

```
namespace ConsoleApp15_interface
{

    interface calc
    {
       // public const double pi=3.145;
        void add();
    }

    interface sci_calc:calc
    {
        void sci_add();
    }

    interface vedik_calc:sci_calc
    {
        void vedic_add();
    }

    class calcuser: /*vedik_calc*/ calc, sci_calc, vedik_calc
    {
        public void add()
        {
            Console.WriteLine("Method overriden from calc interface");
        }

        public void sci_add()
        {
            Console.WriteLine("Method overriden from sci_calc interface");
        }
```

```
        public void vedic_add()
        {
            Console.WriteLine("Method overriden from vedic_calc interface");
        }
    }

    internal class Program
    {
        static void Main(string[] args)
        {
            calcuser cal = new calcuser();
            cal.add();
            cal.sci_add();
            cal.vedic_add();

            //calc calc = new calc();

            //can create interface reference type
            vedik_calc calc = new calcuser();
            calc.add();
            calc.sci_add();
            calc.vedic_add();
            Console.ReadKey();
        }
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp15_interface\ConsoleApp15_interface\bin\Debug\ConsoleApp15_interface.exe
Method overriden from calc interface
Method overriden from sci_calc interface
Method overriden from vedic_calc interface
Method overriden from calc interface
Method overriden from sci_calc interface
Method overriden from vedic_calc interface
```

Default implementation method

Code;-

```
using System;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace ConsoleApp15_interface
{



    interface calc
    {
        // public const double pi=3.145;
        void add();
    }

    interface sci_calc
    {
        void add();
    }

    interface vedik_calc
```

```
    {
        void add();
    }

    class calcuser :  calc, sci_calc, vedik_calc
    {
        //default implementation
        public void add()
        {
            Console.WriteLine("Explicit implementation normal vedik_calc-add()
method called");
        }

        void calc.add()
        {
            Console.WriteLine("Explicit implementation calc Add() method called");
        }

        void sci_calc.add()
        {
            Console.WriteLine("Explicit implementation sci_calc Add() method
called");
        }

        //void vedik_calc.add()
        //{
        //    Console.WriteLine("Explicit implementation vedik_calc Add() method
called");
        //}


    }


    internal class Program
```

```
    {
        static void Main(string[] args)
        {
            //calcuser cal = new calcuser();
            //cal.add();
            //cal.sci_add();
            //cal.vedic_add();

            ////calc calc = new calc();

            ////can create interface reference type
            //vedik_calc calc = new calcuser();
            //calc.add();
            //calc.sci_add();
            //calc.vedic_add();

            //using type casting of derived class obejct
            // for calling explicitly defined interface methods
            calcuser cal = new calcuser();
            cal.add();
           ((calc)cal).add();
           ((sci_calc)cal).add();
          // ((vedik_calc)cal).add();//


            Console.ReadKey();
        }
    }
}
```

Output:-

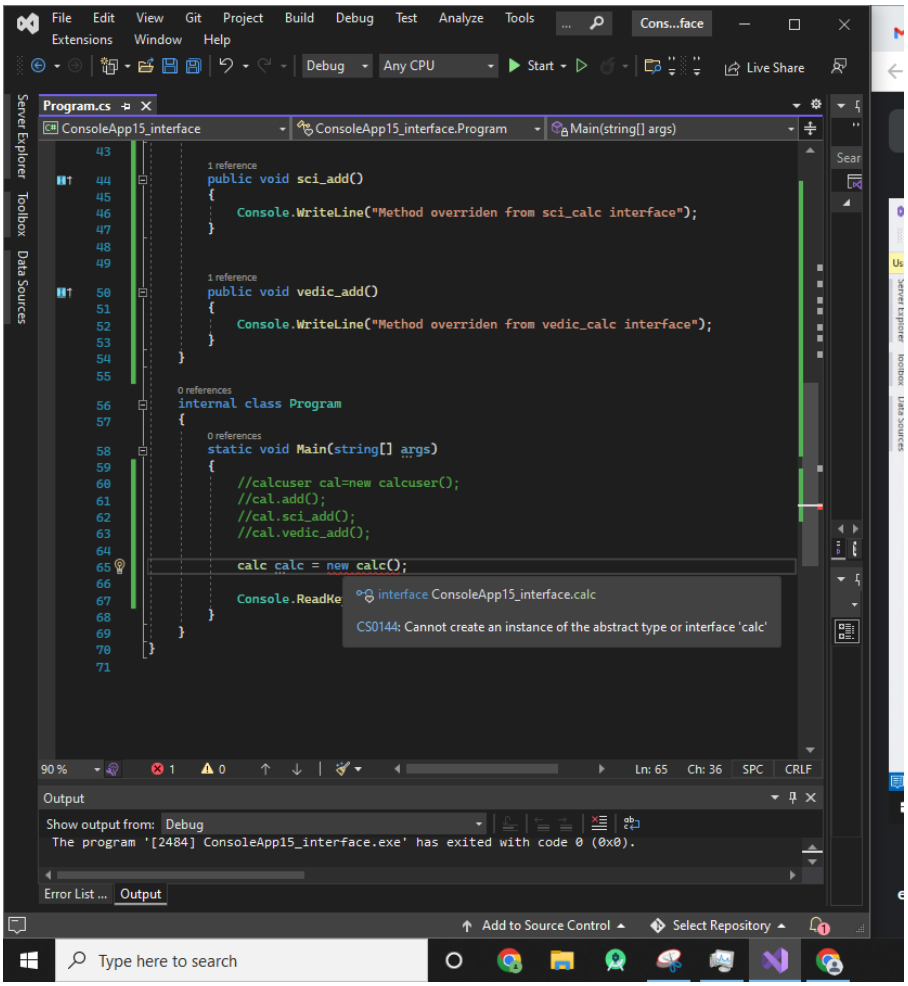
D:\visual studio\ConsoleApp15_interface\ConsoleApp15_interface\bin\Debug\ConsoleApp15_interface.exe

```
Explicit implementation normal add() method called
Explicit implementation calc Add() method called
Explicit implementation sci_calc Add() method called
Explicit implementation vedik_calc Add() method called
```

By commenting vedik_calc.add()

```
D:\visual studio\ConsoleApp15_interface\ConsoleApp15_interface\bin\Debug\ConsoleApp15_interface.exe
Explicit implementation normal vedik_calc-add() method called
Explicit implementation calc Add() method called
Explicit implementation sci_calc Add() method called
```

Cannot instantiate interface

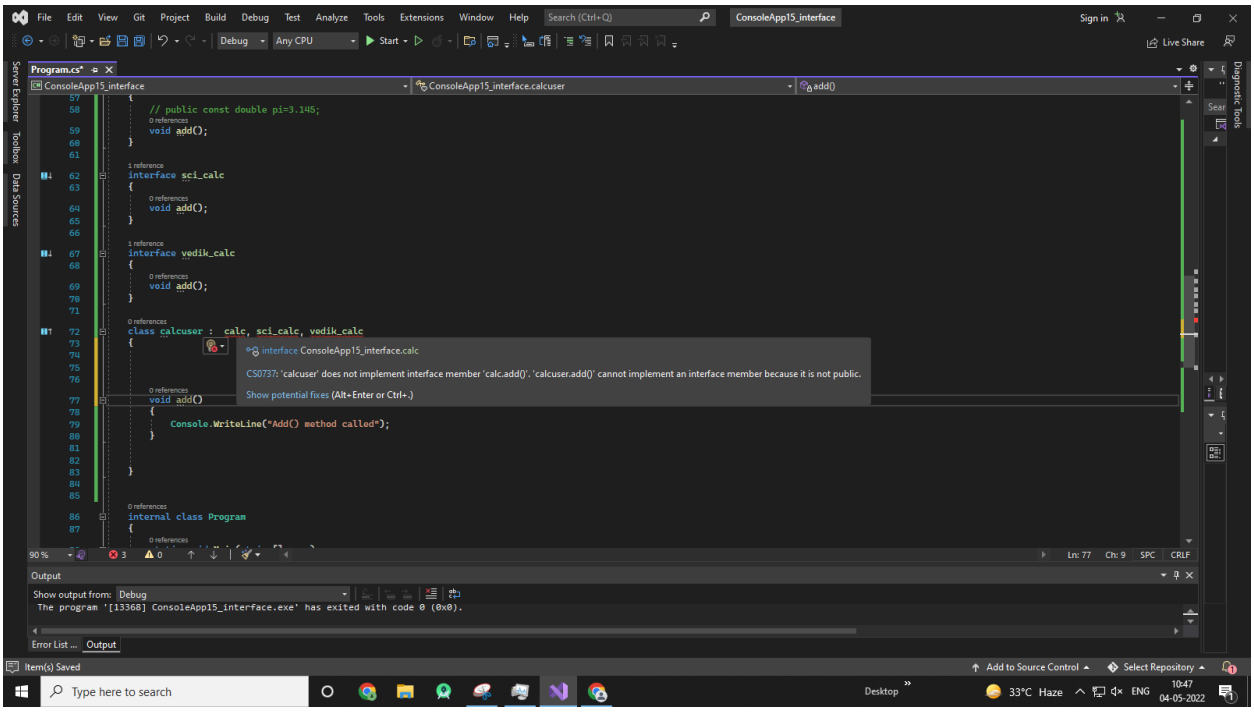


\

Cannot implement method with same method name in different interfaces

```
interface calc
  {
     // public const double pi=3.145;
     void add();
  }

  interface sci_calc
  {
     void add();
  }

  interface vedik_calc
  {
     void add();
  }

  class calcuser :  calc, sci_calc, vedik_calc
  {



     void add()
     {
        Console.WriteLine("Add() method called");
     }



  }


  internal class Program
  {
     static void Main(string[] args)
     {
        //calcuser cal = new calcuser();
```

```
        //cal.add();
        //cal.sci_add();
        //cal.vedic_add();

        ////calc calc = new calc();

        ////can create interface reference type
        //vedik_calc calc = new calcuser();
        //calc.add();
        //calc.sci_add();
        //calc.vedic_add();
        Console.ReadKey();
    }
}
```



Explicit implementation
Condition multiple inheritance and all interface to have smae method name

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/*
 interface
0 supports multiple inheritance
1.100% abstraction
2.cant have method definition
3.by default methods inside interface are abstract in nature
4.explicit use of abstract modifier is not allowed
5.instance data member are not allowed
 6.interface can inherit from another interface without defining member function
7.cannot instantiate or create object or interface
8.refernece type of interface can be created
9.explicit implementation of interface methods with same name in case of explicit
 */
namespace ConsoleApp15_interface
{



    interface calc
    {
        // public const double pi=3.145;
        void add();
    }

    interface sci_calc
    {
        void add();
```

```
    }

    interface vedik_calc
    {
        void add();
    }

    class calcuser :  calc, sci_calc, vedik_calc
    {



        void calc.add()
        {
            Console.WriteLine("Explicit implementation calc Add() method called");
        }

        void sci_calc.add()
        {
            Console.WriteLine("Explicit implementation sci_calc Add() method
called");
        }

        void vedik_calc.add()
        {
            Console.WriteLine("Explicit implementation vedik_calc Add() method
called");
        }


    }



 internal class Program
```

```
    {
        static void Main(string[] args)
        {

            //using type casting of derived class obejct
            // for calling explicitly defined interface methods
            calcuser cal = new calcuser();
           ((calc)cal).add();
           ((sci_calc)cal).add();
           ((vedik_calc)cal).add();


            Console.ReadKey();
        }
    }
}
```

D:\visual studio\ConsoleApp15_interface\ConsoleApp15_interface\bin\Debug\ConsoleApp15_interface.exe

```
Explicit implementation calc Add() method called
Explicit implementation sci_calc Add() method called
Explicit implementation vedik_calc Add() method called
```

04May2022

Practical 19:

Partial class and methods in c#

Code:-

a.partial class

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp16_partialclass
{
    internal class Program
    {
        static void Main(string[] args)
        {
            exam e = new exam(1, "kjsim", "maths");
            e.show();
            Console.ReadKey();
        }
    }
}
```

Class1.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp16_partialclass
{
```

```
    partial class exam
    {
        private int center_id;
        private string center_name;
    }
}
```

Class2.cs

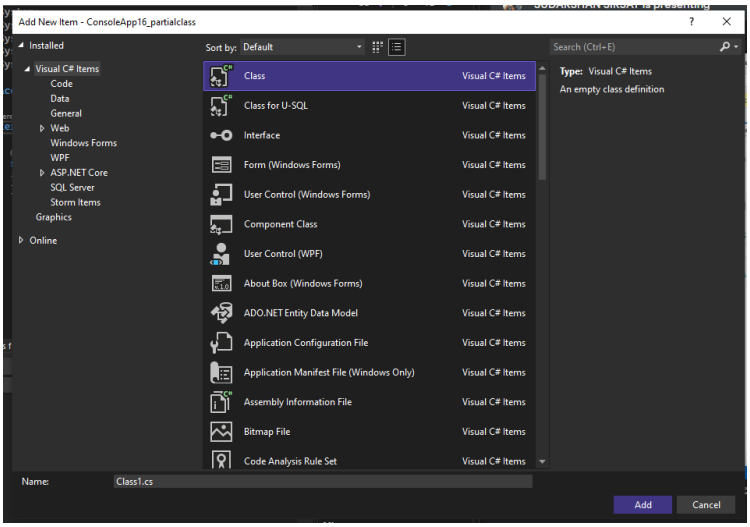
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp16_partialclass
{
     partial class exam
    {
        private string subject;

        public exam(int cid,string cname,string sub)
        {
            this.center_id = cid;
            this.center_name= cname;
            this.subject = sub;
        }
    }
}
```

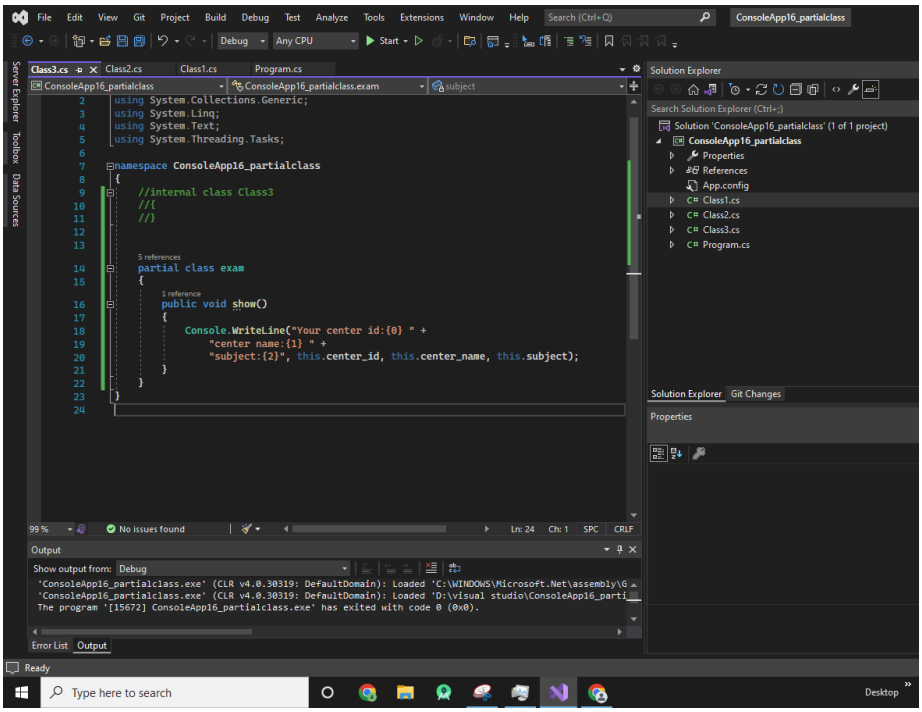
Class3.cs

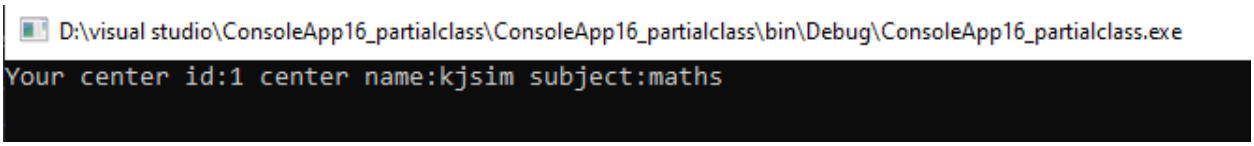
```
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp16_partialclass
{
    partial class exam
    {
        public void show()
        {
            Console.WriteLine("Your center id:{0} " +
                "center name:{1} " +
                "subject:{2}", this.center_id, this.center_name, this.subject);
        }
    }
}
```

Output:-







b.Partial methods

/*

```
            * partial methods
            * 1.implicitly private
            * 2.can have only void as return type
            * 3.can be static or generic
            * 4.declaration and definition can be in
            * same or different files
   */
```

—----------

Class2.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp16_partialclass
{
    //internal class Class2
    //{
    //}

   partial class exam
    {
        private string subject;

        public exam(int cid,string cname,string sub)
        {
            this.center_id = cid;
            this.center_name= cname;
            this.subject = sub;
        }
        partial void print()
        {
            Console.WriteLine("defining the partial method");
        }
```

```
    }
}
```

Class3.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp16_partialclass
{
    //internal class Class3
    //{
    //}


    partial class exam
    {

        partial void print();

        public void show()
        {
            Console.WriteLine("Your center id:{0} " +
                "center name:{1} " +
                "subject:{2}", this.center_id, this.center_name, this.subject);
            print();
            //partial methods are by default private and can be static but cannot be
public
                ////defiault return type is void
        }

    }
```

}

```
{
    //internal class Class3
    //{
    //}


    5 references
    partial class exam
    {

        2 references
        partial int print();

        1 reference
        public void show
        {
            Console.Writeline("your center id:{0}, "
                "center name:{1} " +
                "subject:{2}", this.center_id, this.center_name, this.subject);
            print();
            //partial methods are by default private and can be static but cannot be public
                    ////defiault return type is void
        }

    }
}
```

int exam.print()

CS8796: Partial method 'exam.print()' must have accessibility modifiers because it has a non-void return type.

Show potential fixes (Alt+Enter or Ctrl+.)

```
D:\visual studio\ConsoleApp16_partialclass\ConsoleApp16_partialclass\bin\Debug\ConsoleApp16_partialclass.exe
Your center id:1 center name:kjsim subject:maths
defining the partial method
```

04may2022

Practical 20:

Extension methods

Code:-

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

class MCA
{
    public void method1()
    {       Console.WriteLine("Method1() call");        }

    public void method2()
    {       Console.WriteLine("Method2() call");        }
    public void method3()
    {       Console.WriteLine("Method3() call");        }

    //if same method signature exists in both files(original and created class)
    //then method from original class definition will be called
    //example of runtime polymorphism
    public void method5(string subj, string prof)
    {
        Console.WriteLine("method5 in mca called {0} {1}",subj,prof);
    }
}
```

```
namespace ConsoleApp16_extensionmethod
{
    internal class Program
    {
        static void Main(string[] args)
        {
            MCA m=new MCA();
            m.method1();
            m.method2();
            m.method3();
            m.method4();
            m.method5("saumik","dept of data science");
            Console.ReadKey();
        }
    }
}
```

Extensionmethods.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp16_extensionmethod
{
    static class extensionmethods
    {  //enstionsion method of mca class
        //this mca m - binding parameter
        //binding parameter should be always
        //at first place in formal parameter of list order
//extension class should be top level static class

        public static void method4(this MCA m)
        {  Console.WriteLine("extension method-Method4() call");    }
```

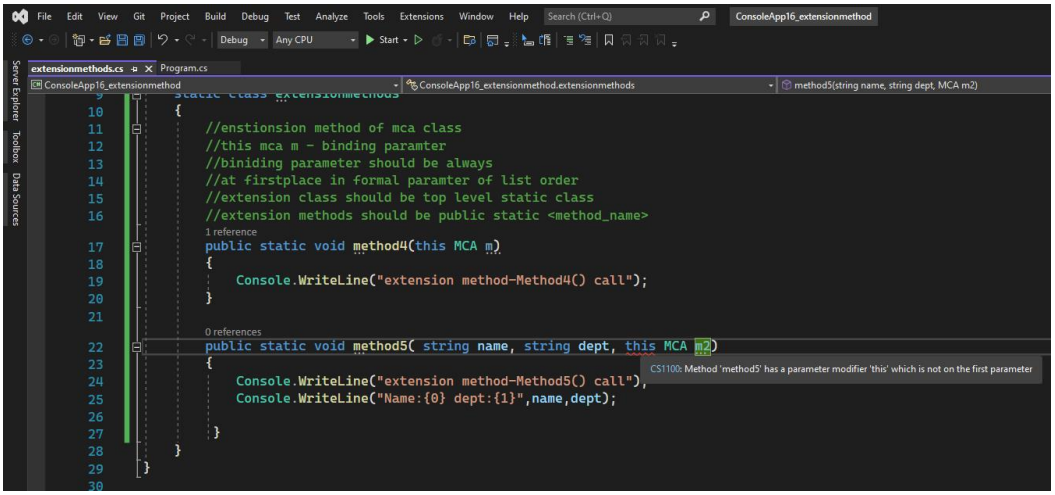
```
        public static void method5(this MCA m2, string name, string dept)
        {
            Console.WriteLine("extension method-Method5() call");
            Console.WriteLine("Name:{0} dept:{1}",name,dept);
        }
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp16_extensionmethod\ConsoleApp16_extensionmethod\bin\Debug\ConsoleApp16_extensionmethod.exe
Method1() call
Method2() call
Method3() call
extension method-Method4() call
extension method-Method5() call
```

Having same method5() in extensionmethods

```
D:\visual studio\ConsoleApp16_extensionmethod\ConsoleApp16_extensionmethod\bin\Debug\ConsoleApp16_extensionmethod.exe
Method1() call
Method2() call
Method3() call
extension method-Method4() call
method5 in mca called saumik dept of data science
```



04may2022

Practical 21:

Sealed classes and methods in c#

Code:-

Sealed class

program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//structs are sealed implicitly (Can't be inherited)
sealed class Base
{
    public void show()
    {
        Console.WriteLine("this is sealed class and hence cant be inherited");
    }
}

//class derived : Base
//{

//}

namespace ConsoleApp17_sealedclass
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Base b=new Base();
```

```
            b.show();
            Console.ReadKey();
        }
    }
}
```

By creating object of base class we can access its methods

```
D:\visual studio\ConsoleApp17_sealedclass\ConsoleApp17_sealedclass\bin\Debug\ConsoleApp17_sealedclass.exe
this is sealed class and hence cant be inherited
```

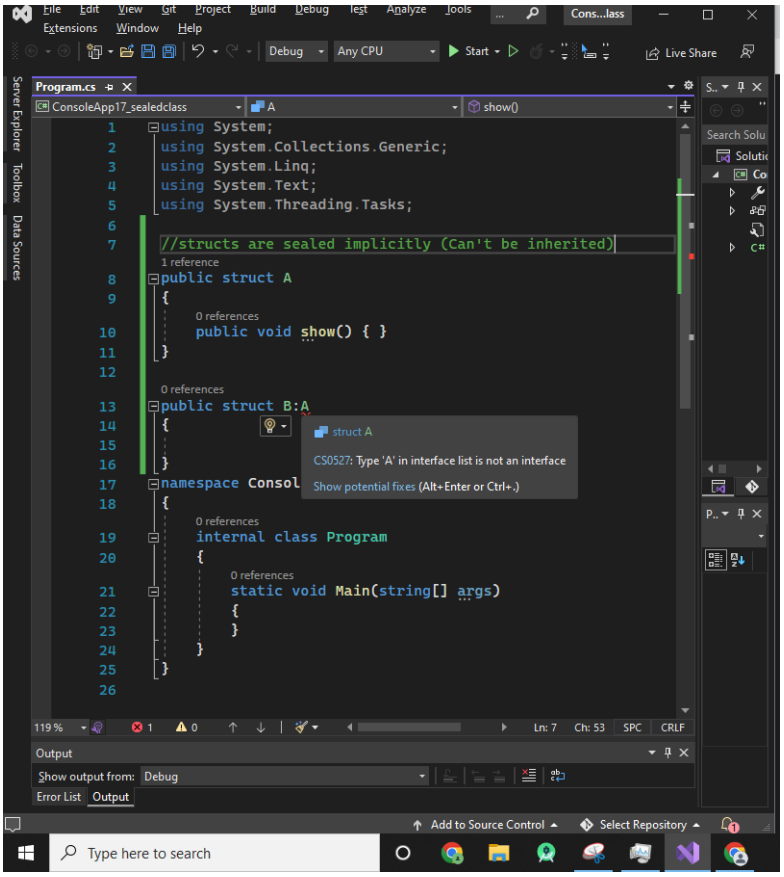
For Methods:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//structs are sealed implicitly (Can't be inherited)
/*sealed*/ class Base
{
    public virtual void show()
    {
        Console.WriteLine("this is sealed class and hence cant be inherited");
    }


}

class derived : Base
{
    public sealed override  void show()
    {
```

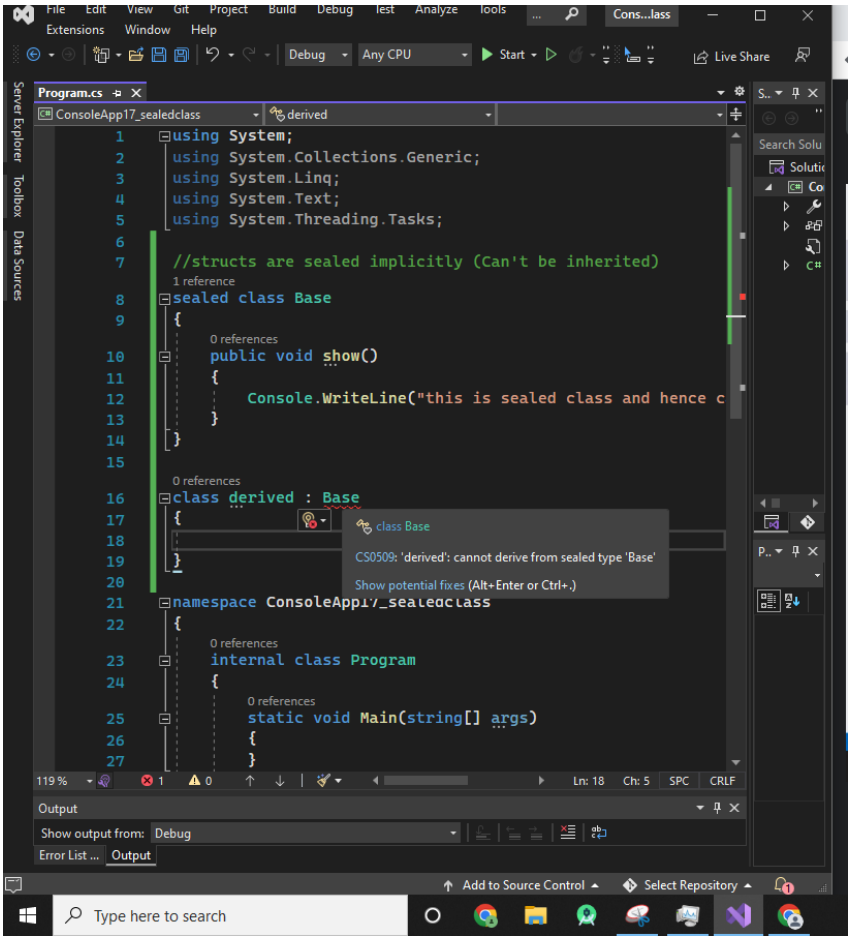
```
        Console.WriteLine("this is final overriden behaviour of show and henceforth
will be called ");
    }
}

//class derived1:derived
//{
//    public sealed override void show()
//    {

//    }
//}


namespace ConsoleApp17_sealedclass
{
    internal class Program
    {
        static void Main(string[] args)
        {
            //Base b=new Base();
            //b.show();

            derived d = new derived();
            d.show();
            Console.ReadKey();
        }
    }
}
```

Outputs:

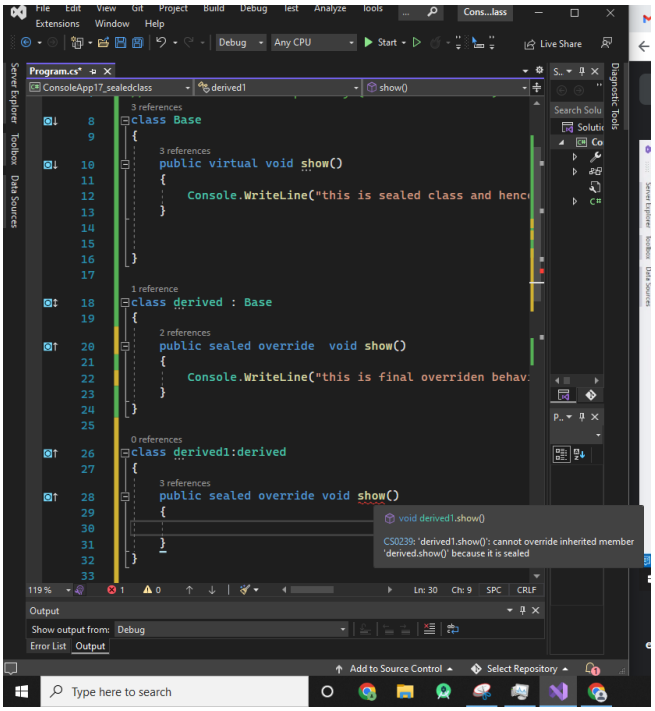
Structure cannot be inherited i.e by default sealed



Cannot be inherited by class as well



In multilevel inheritance it is not possible to override the same method but in single method it is allowed


Program.cs*
ConsoleApp17_sealedclass
derived1
show()
3 references
class Base
{
3 references
public virtual void show()
{
Console.WriteLine("this is sealed class and henc
}
}
1 reference
class derived : Base
{
2 references
public sealed override void show()
{
Console.WriteLine("this is final overriden behav
}
}
0 references
class derived1:derived
{
3 references
public sealed override void show()
{
}
}
void derived1.show()
CS0239: 'derived1.show()': cannot override inherited member 'derived.show()' because it is sealed
Output
Show output from: Debug
Error List
Output
Add to Source Control
Select Repository
Type here to search

05May2022

Practical 22:

Method overloading

Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
public class A
{ }
namespace ConsoleApp17_methodoverloading
{
    internal class Program
    {
        static void Main(string[] args)
        {
            float a, b, c,x;
            a = 12.3f;
            b = 1.1f;

            add(2, 3);
            add(2.3f, 3);
            add(2, 3.2f);
            add(3.2f, 5.6f, 5.6f);
            add(3.2f, 5.6f,7.8f,8.9f,1.1f );
            add(ref a, ref b,out c);
            Console.WriteLine("reference c{0} ", c);
            Console.ReadKey();
        }



        //overloading with type of method parameters
```

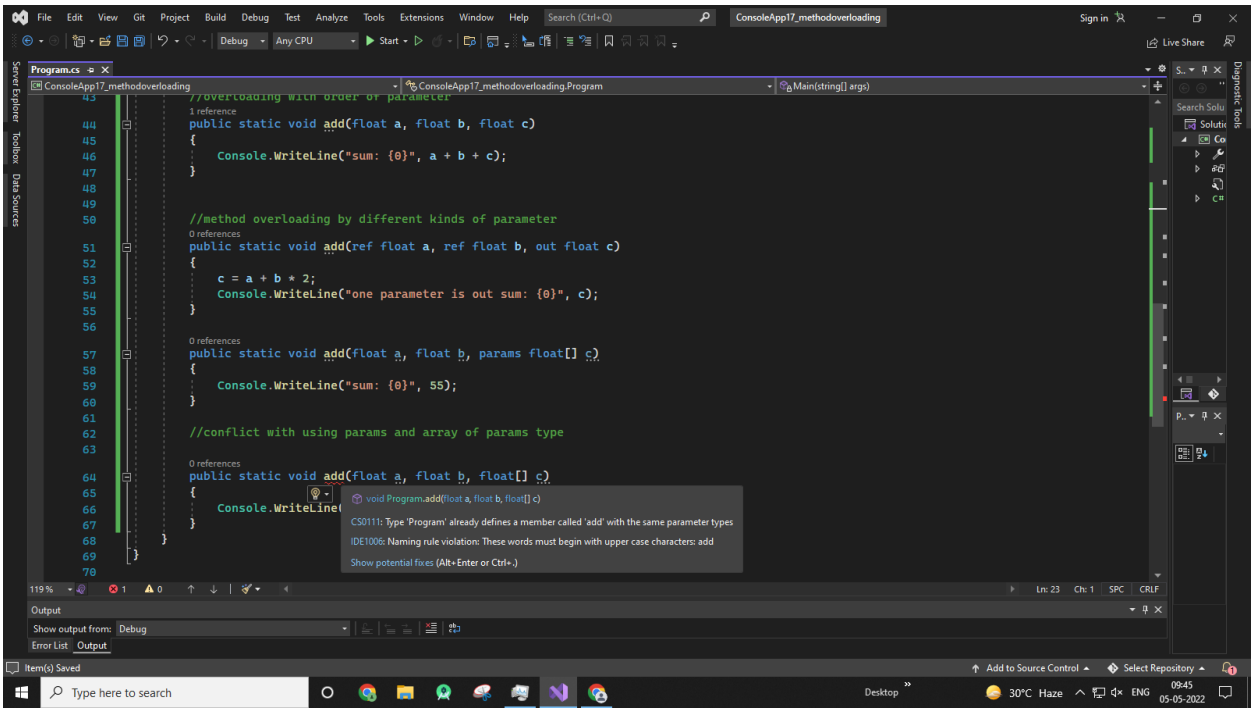
```
public static void add(int a, int b)
{
    Console.WriteLine("sum of int int: {0}", a + b);
}
public static void add(float a, float b)
{
    Console.WriteLine("sum of float float: {0}", a + b);
}


public static void add(float a, int b)
{
    Console.WriteLine("sum of float int: {0}", a + b);
}
//overloading with order of parameter
public static void add(float a, float b, float c)
{
    Console.WriteLine("sum of float float float: {0}", a + b + c);
}


//method overloading by different kinds of parameter
public static void add(ref float a, ref float b, out float c)
{
    c = a + b * 2;
    Console.WriteLine("one parameter is out sum: {0}", c);
}

public static void add(float a, float b, params float[] c)
{
    Console.WriteLine("sum: {0}", 55);
}

//conflict with using params and array of params type
```

```
        //public static void add(float a, float b, float[] c)
        //{
        //    Console.WriteLine("sum: {0}", 55);
        //}
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp17_methodoverloading\ConsoleApp17_methodoverloading\bin\Debug\ConsoleApp17_methodoverloading.exe
sum of int int: 5
sum of float int: 5.3
sum of float float: 5.2
sum of float float float: 14.4
sum: 55
one parameter is out sum: 14.5
reference c14.5
```

Conflict with params type



05may2022

Practical 23:

Enum in C#

Enum is primitive data type

Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
//enum are strongly typed constants
public enum days
{
    monday,
    tuesday,
    wednesday,
    thursday,
    friday,
    saturday,
    sunday
}

public enum button:long //explicity specify data types
{
    dst=7304053115,
    eco,
    fin,
    ops,
    marks,
    //ON,
    //OFF,
}
```

```
namespace ConsoleApp18_enum
{
    internal class Program
    {
        static void Main(string[] args)
        {
            foreach (var arg in Enum.GetValues(typeof(days)))
                Console.WriteLine(arg);
            int[] enumval = (int[])Enum.GetValues(typeof(days));

            Console.WriteLine("printing int value of string");
            foreach (int val in enumval)
            {
                Console.WriteLine(val);
            }

            Console.ReadKey();
        }
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp18_enum\ConsoleApp18_enum\bin\Debug\ConsoleApp18_enum.exe
monday
tuesday
wednesday
thursday
friday
saturday
sunday
```

```
D:\visual studio\ConsoleApp18_enum\ConsoleApp18_enum\bin\Debug\ConsoleApp18_enum.exe
monday
tuesday
wednesday
thursday
friday
saturday
sunday
printing int value of string
0
1
2
3
4
5
6
```

Types required by enumeration

```
public enum b1:Boolean
{
}

public enum button:
{
    dst=7304053115,
    eco,
    fin,
    ops,
    marks,
    //ON,
    //OFF,
}

namespace ConsoleApp18_enum
{
    internal class Program
    {
        static void Main(string[] args)
        {
            foreach (var arg in Enum.GetValues(typeof(days)))
                Console.WriteLine(arg);

            int[] enumval = (int[])Enum.GetValues(typeof(days));
```

<u>Using enum in class</u>

Code and output

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
//enum are strongly typed constants
public enum days
{
    monday,
    tuesday,
    wednesday,
    thursday,
    friday,
    saturday,
    sunday
}
//public enum b1:Boolean//{//}
public enum button:long //explicity specify data types
{
    dst=7304053115,
    eco,
    fin,
    ops,
    marks,
    //ON,
    //OFF,
}

public enum Electivesubject
{
    webTech,
    RPogramming,
    AdvancedJava,
```

```
    python,
    AIMl
}


class student
{
    public string Name { get; set; }
    public Electivesubject subject { get; set; }
}

namespace ConsoleApp18_enum
{
    internal class Program
    {
        static void Main(string[] args)
        {
            foreach (var arg in Enum.GetValues(typeof(days)))
                Console.WriteLine(arg);

            int[] enumval = (int[])Enum.GetValues(typeof(days));


            Console.WriteLine("printing int value of string");
            foreach (int val in enumval)
            {               Console.WriteLine(val);           }


            student[] stu = new student[3];
            //using enum value as a constant to pass as a value
      stu[0] = new student { Name = "saumik", subject = Electivesubject.python };
      stu[1] = new student { Name = "defg", subject = Electivesubject.AIMl };
     stu[2] = new student { Name = "mnop", subject = Electivesubject.webTech };
//stu[2] = new student{Name ="wxyz",subject=Electivesubject.AdvancedJava };
```

```
            foreach(student s in stu)
            {
                Console.WriteLine("name:{0} opted subject{1}",s.Name,s.subject);
            }
            //Console.WriteLine(button.ON);
            Console.ReadKey();
        }
    }
}
```

D:\visual studio\ConsoleApp18_enum\ConsoleApp18_enum\bin\Debug\ConsoleApp18_enum.exe

```
monday
tuesday
wednesday
thursday
friday
saturday
sunday
printing int value of string
0
1
2
3
4
5
6
name:saumik opted subjectpython
name:defg opted subjectAIMl
name:mnop opted subjectwebTech
```

05may2022

Practical 24:

Delegates in C#

Code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//delegates are typesafe function pointers
/* */

public delegate void fp();

namespace ConsoleApp19_delegates
{
    internal class Program
    {
        static void Main(string[] args)
        {
            fp f1 = new fp(show);
            fp f2 = new fp(print);
            f1();
            f2();

            Console.ReadKey();
        }

        public static void show()
        {
            Console.WriteLine("Show method for a delegate demo ");
        }
```

```
        public static void print()
        {
            Console.WriteLine("print method for a delegate demo ");
        }
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp19_delegates\ConsoleApp19_delegates\bin\Debug\ConsoleApp19_delegates.exe
Show method for a delegate demo
print method for a delegate demo
```

05may2022

Practical 25:

Multicast delegate

code:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/* Multicast delegate points to more than one methods
 * += operator used to add method
 * -= operator is used to remove method
 * works in FIFO fashion i.e it calls the methods in order they
 * are added or removed
 */
public delegate void MDelegate();
namespace ConsoleApp20_multicastdelegate
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("using multicast delegate");
            MDelegate mdelegate1 = new MDelegate(method1);

            mdelegate1 += method2;//add reference operator ("+=")
            mdelegate1 += method3;
            mdelegate1 += method4;
            mdelegate1();

            Console.WriteLine("Update multicast delegate calling ...");
            mdelegate1 -= method1;//remove reference operator("-=")
            mdelegate1();
```

```
            Console.ReadKey();
        }


        public static void method1()
        {
            Console.WriteLine("Method1() called");
        }


        public static void method2()
        {
            Console.WriteLine("Method2() called");
        }


        public static void method3()
        {
            Console.WriteLine("Method3() called");
        }


        public static void method4()
        {
            Console.WriteLine("Method4() called");
        }

    }
}
```

```
D:\visual studio\ConsoleApp20_multicastdelegate\ConsoleApp20_multicastdelegate\bin\Debug\ConsoleApp20_multicastdelegate.exe
using multicast delegate
Method1() called
Method2() called
Method3() called
Method4() called
Update multicast delegate calling ...
Method2() called
Method3() called
Method4() called
```

2.checking specific order wise execution

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/* Multicast delegate points to more than one methods
 * += operator used to add method
 * -= operator is used to remove method
 * works in FIFO fashion i.e it calls the methods in order they
 * are added or removed
 */
public delegate void MDelegate();
namespace ConsoleApp20_multicastdelegate
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("using multicast delegate");
            MDelegate mdelegate1 = new MDelegate(method3);

            mdelegate1 += method1;//add reference operator ("+=")
            mdelegate1 += method3;
            mdelegate1 += method4;
```

```
            mdelegate1();

            Console.WriteLine("Update multicast delegate calling ...");
            mdelegate1 -= method1;//remove reference operator("-=")
            mdelegate1();
            Console.ReadKey();
        }


        public static void method1()
        {
            Console.WriteLine("Method1() called");
        }


        public static void method2()
        {
            Console.WriteLine("Method2() called");
        }


        public static void method3()
        {
            Console.WriteLine("Method3() called");
        }


        public static void method4()
        {
            Console.WriteLine("Method4() called");
        }

    }
}
```

```
D:\visual studio\ConsoleApp20_multicastdelegate\ConsoleApp20_multicastdelegate\bin\Debug\ConsoleApp20_multicastdelegate.exe
using multicast delegate
Method3() called
Method1() called
Method3() called
Method4() called
Update multicast delegate calling ...
Method3() called
Method3() called
Method4() called
```

By using return type

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/* Multicast delegate points to more than one methods
 * += operator used to add method
 * -= operator is used to remove method
 * works in FIFO fashion i.e it calls the methods in order they
 * are added or removed
 * if multicast delegate methods return value then it will be
 * the value returnned by last method in the order
 */
public delegate int MDelegate();
namespace ConsoleApp20_multicastdelegate
{
    internal class Program
    {
        static void Main(string[] args)
        {
```

```
    int retvalue=0;
    Console.WriteLine("using multicast delegate");
    MDelegate mdelegate1 = new MDelegate(method1);

    mdelegate1 += method2;//add reference operator ("+=")
    mdelegate1 += method3;
    //mdelegate1 += method4;
    retvalue = mdelegate1();
    Console.WriteLine("Multicast delegate returned {0}",retvalue);

    Console.WriteLine("Update multicast delegate calling ...");
    mdelegate1 -= method1;//remove reference operator("-=")
    mdelegate1 += method2;

    retvalue = mdelegate1();
    Console.WriteLine("Multicast delegate returned {0}", retvalue);
    Console.ReadKey();
}


public static int method1()
{
    Console.WriteLine("Method1() called");
    return 11;
}


public static int method2()
{
    Console.WriteLine("Method2() called");
    return 22;
}


public static int method3()
```

```
        {
            Console.WriteLine("Method3() called");
            return 33;
        }


        //public static int method4()
        //{
        //    Console.WriteLine("Method4() called");
        //    return 44;
        //}

    }
}
```

```
D:\visual studio\ConsoleApp20_multicastdelegate\ConsoleApp20_multicastdelegate\bin\Debug\ConsoleApp20_multicastdelegate.exe
using multicast delegate
Method1() called
Method2() called
Method3() called
Multicast delegate returned 33
Update multicast delegate calling ...
Method2() called
Method3() called
Method2() called
Multicast delegate returned 22
```

By using out parameter

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp2
{
    public delegate void MDelegate(out int a);
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("using multicast delegate ");
            int outValue = 0;
            MDelegate mDelegate = new MDelegate(md3);
            mDelegate += md1; //add ref operator
            mDelegate += md2;
            mDelegate(out outValue);
            Console.WriteLine("Multicast delegate returned: {0}", outValue);
            //mDelegate();

            Console.WriteLine("updated multicast delegate calling....");
            mDelegate -= md2;
            mDelegate(out outValue);

            Console.ReadKey();

        }

        public static void md1(out int a)
        {
            a = 17;
            Console.WriteLine("md1() called and stored {0} in out", a);
```

```
        }

        public static void md2(out int a)
        {
            a = 36;
            Console.WriteLine("md2() called and stored {0} in out", a);
        }

        public static void md3(out int a)
        {
            a = 40;
            Console.WriteLine("md3() called and stored {0} in out", a);
        }
    }
}
```

```
D:\visual studio\vs_savefile\ConsoleApp2\ConsoleApp2\bin\Debug\ConsoleApp2.exe
using multicast delegate
md3() called and stored 40 in out
md1() called and stored 17 in out
md2() called and stored 36 in out
Multicast delegate returned: 36
updated multicast delegate calling....
md3() called and stored 40 in out
md1() called and stored 17 in out
```

Delegate use in student application

Using delegate in student app to pass with grace mark

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


public delegate bool IsPassedWithGraceMarks(Students students);
public class Students
{
    public int Id { get; set; }
    public string Name { get; set; }

    public int Marks { get; set; }

    public static void PasswithGraceMarks(List<Students> studlist,
IsPassedWithGraceMarks isgracable1)
    {



        foreach (Students student in studlist)
        {
            if (isgracable1(student))
            {
                student.Marks += 6;
                Console.WriteLine("ID:{0}  Name:{1}  Student with {2} marks passed
with 6 grace mark range", student.Id, student.Name, student.Marks);
            }
        }
    }
}
namespace ConsoleApp19b
{
```

```
    internal class Program
    {
        static void Main(string[] args)
        {
            List<Students> students1 = new List<Students>();
            students1.Add(new Students() { Id = 101, Name = "Saumik Dutta", Marks
= 77 });
            students1.Add(new Students() { Id = 102, Name = "Karishma tanna",
Marks = 37 });
            students1.Add(new Students() { Id = 103, Name = "sanjay sharma", Marks
= 29 });
            students1.Add(new Students() { Id = 104, Name = "shreya mhatre", Marks
= 39 });
            students1.Add(new Students() { Id = 105, Name = "gaurav sethi", Marks =
19 });


            IsPassedWithGraceMarks IsGracable = new
IsPassedWithGraceMarks(isgracable);
            Students.PasswithGraceMarks(students1, IsGracable);
            Console.ReadKey();
        }


        public static bool isgracable(Students students)
        {
            if (students.Marks >= 29 && students.Marks < 35)
                return true;
            else
                return false;
        }

    }
}
```

```
D:\visual studio\ConsoleApp19b\ConsoleApp19b\bin\Debug\ConsoleApp19b.exe
ID:103  Name:sanjay sharma  Student with 35 marks passed with 6 grace mark range
```

Using lambda expression

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


public delegate bool IsPassedWithGraceMarks(Students students);
public class Students
{
    public int Id { get; set; }
    public string Name { get; set; }

    public int Marks { get; set; }

    public static void PasswithGraceMarks(List<Students> studlist,
IsPassedWithGraceMarks isgracable1)
    {



        foreach (Students student in studlist)
        {
            if (isgracable1(student))
            {
                student.Marks += 6;
                Console.WriteLine("ID:{0}  Name:{1}  Student with {2} marks passed
with 6 grace mark range", student.Id, student.Name, student.Marks);
            }
```

```
        }
    }
}
namespace ConsoleApp19b
{
    internal class Program
    {
        static void Main(string[] args)
        {
 List<Students> students1 = new List<Students>();
students1.Add(new Students() { Id = 101, Name = "Saumik Dutta", Marks = 77 });
students1.Add(new Students() { Id = 102, Name = "Karishma tanna", Marks = 37
});
students1.Add(new Students() { Id = 103, Name = "sanjay sharma", Marks = 29 });
students1.Add(new Students() { Id = 104, Name = "shreya mhatre", Marks = 39 });
students1.Add(new Students() { Id = 105, Name = "gaurav sethi", Marks = 31 });
IsPassedWithGraceMarks IsGracable = new IsPassedWithGraceMarks(isgracable);
Students.PasswithGraceMarks(students1, IsGracable);
///

List<Students> students2 = new List<Students>();
students2.Add(new Students() { Id = 301, Name = "Saumik ", Marks = 21 });
students2.Add(new Students() { Id = 302, Name = "Karishma ", Marks = 23 });
students2.Add(new Students() { Id = 303, Name = "sanjay ", Marks = 29 });
students2.Add(new Students() { Id = 304, Name = "shreya J", Marks = 33 });
students2.Add(new Students() { Id = 305, Name = "gaurav bane", Marks = 22 });
Console.WriteLine("Gracemark range from 21-24");
            //replacing delegate call with lambda expression
            //=> is lamba operator
            //(input-obj)=>expression
//IsPassedWithGraceMarks IsGracable1 = new
IsPassedWithGraceMarks(isgracable);
Students.PasswithGraceMarks(students2,Students=> Students.Marks>=21 &&
Students.Marks < 25);
```

```
            Console.ReadKey();
        }
    public static bool isgracable(Students students)
        {
            if (students.Marks >= 29 && students.Marks < 35)
                return true;
            else
                return false;
        }

    }
}
```

```
D:\visual studio\ConsoleApp19b\ConsoleApp19b\bin\Debug\ConsoleApp19b.exe
ID:103  Name:sanjay sharma  Student with 35 marks passed with 6 grace mark range
ID:105  Name:gaurav sethi  Student with 37 marks passed with 6 grace mark range
Gracemark range from 21-24
ID:301  Name:Saumik Dutta  Student with 27 marks passed with 6 grace mark range
ID:302  Name:Karishma tanna  Student with 29 marks passed with 6 grace mark range
ID:305  Name:gaurav sethi  Student with 28 marks passed with 6 grace mark range
```

```
D:\visual studio\ConsoleApp19b\ConsoleApp19b\bin\Debug\ConsoleApp19b.exe
ID:103  Name:sanjay sharma  Student with 35 marks passed with 6 grace mark range
ID:105  Name:gaurav sethi  Student with 37 marks passed with 6 grace mark range
Gracemark range from 21-24
ID:301  Name:Saumik   Student with 27 marks passed with 6 grace mark range
ID:302  Name:Karishma   Student with 29 marks passed with 6 grace mark range
ID:305  Name:gaurav bane  Student with 28 marks passed with 6 grace mark range
```

06May2022

Practical 26:

Generics and Collections in C#

Code/Output:-

1.Generic class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

public class GenericClass<T>
{
    public T Name { get; set; }
    public T Id { get; set; }

    //public GenericClass(T msg)
    //{
    //    Console.WriteLine("Generic class handling type: {0} and value:
{1}",msg.GetType(),msg.);
    //}



    public void show()
    {
        Console.WriteLine("id:{0}  name:{1}",this.Id,this.Name);
    }

}

namespace ConsoleApp21a_generics
{
    internal class Program
```

```
    {
        static void Main(string[] args)
        {
            GenericClass<int> instance =new GenericClass<int>();
            instance.Name = 17019;
            instance.Id = 1;
            instance.show();

            GenericClass<string> instance2 = new GenericClass<string>();
            instance2.Name = "saumik";
            instance2.Id = "2";
            instance2.show();


            Console.ReadKey();

        }
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp21a_generics\ConsoleApp21a_generics\bin\Debug\ConsoleApp21a_generics.exe
id:1  name:17019
id:2  name:saumik
```

Passing value in generic class constructor

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

public class GenericClass<T>
{
    public T Name { get; set; }
    public T Id { get; set; }

    public GenericClass(T msg)
    {
        Console.WriteLine("Generic class handling type: {0} and value: {1}",
msg.GetType().Name, msg);
    }



    public void show()
    {
        Console.WriteLine("id:{0}  name:{1}",this.Id,this.Name);
    }

}

namespace ConsoleApp21a_generics
{
    internal class Program
    {
        static void Main(string[] args)
        {
```

```
            GenericClass<int> instance = new GenericClass<int>(12345);
            GenericClass<long> instance1 = new GenericClass<long>(17236123);
            GenericClass<string> instance2 = new GenericClass<string>("saumik");

            Console.ReadKey();

        }
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp21a_generics\ConsoleApp21a_generics\bin\Debug\ConsoleApp21a_generics.exe
Generic class handling type: Int32 and value: 12345
Generic class handling type: Int64 and value: 17236123
Generic class handling type: String and value: saumik
```

Generic methods

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


public class A
{
   public void show<T>(T a,T b)
   {
      Console.WriteLine("calling generic method a- {0} b-{1} parameter-{2}",
         a,b,a.GetType().Name);

   }
}
public class GenericClass<T>
{
   public T Name { get; set; }
   public T Id { get; set; }

   public GenericClass(T msg)
   {
Console.WriteLine("Generic class " +"handling type: {0} and value: {1}",
         msg.GetType().Name, msg);    }

public void show()
   {
      Console.WriteLine("id:{0}  name:{1}",this.Id,this.Name);
   }

}
```

```
namespace ConsoleApp21a_generics
{
    internal class Program
    {
        static void Main(string[] args)
        {
//for generic method
            A a = new A();
            a.show<int>(10, 20);
            a.show<string>("saumik", "dutta");
            Console.ReadKey();

        }
    }
}
```

Output:-

```
D:\visual studio\ConsoleApp21a_generics\ConsoleApp21a_generics\bin\Debug\ConsoleApp21a_generics.exe
calling generic method a- 10 b-20 parameter-Int32
calling generic method a- saumik b-dutta parameter-String
```

2.arraylist

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Collections;
//using system.collections to access arraylist
namespace ConsoleApp21_index_generics_arraylist
{
    internal class Program
    {
        static void Main(string[] args)
        {

            ArrayList arrayList = new ArrayList();
            arrayList.Add(111);
            arrayList.Add("this is string value");
            arrayList.Add(4.32f);
            arrayList.Add(123.45);
            arrayList.Add(true);
            arrayList.Add(null);
            //even null is allowed as a value
            arrayList.Add("previous value is null");

            foreach(var arrayListItem in arrayList)
            {
                Console.WriteLine(arrayListItem);
            }
            Console.ReadKey();
        }
    }

 }
```

Output:-

```
D:\visual studio\ConsoleApp21_index_generics_arraylist\ConsoleApp21_index_generics_arraylist\bin\Debug\ConsoleApp21_index_generics_arraylist.exe
111
this is string value
4.32
123.45
True

previous value is null
```

<u>arrayList with all operations</u>
<u>remove ,RemoveAt,RemoveRange,Insert,Add,Reverse</u>

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Collections;

namespace ConsoleApp21_index_generics_arraylist
{

    /*
     * arraylist
     * dynamic array with increasing and decreasing capacity (length)
     * can hold any primimtive or user defined type of value
     * */
    internal class Program
    {
        static void Main(string[] args)
        {


            ArrayList arrayList = new ArrayList();
```

```
arrayList.Add(111);
arrayList.Add("this is string value");
arrayList.Add(4.32f);
arrayList.Add(123.45);
arrayList.Add(true);
arrayList.Add(null);
//even null is allowed as a value
arrayList.Add("previous value is null");


foreach(var arrayListItem in arrayList)
{
    Console.WriteLine(arrayListItem);
}

Console.WriteLine("Capacity of arraylist now is {0}",arrayList.Capacity);

arrayList.Add(112);
arrayList.Add("this is string value1");
arrayList.Add(41.32f);
arrayList.Add(1231.45);
arrayList.Add(true);
arrayList.Add(null);
//even null is allowed as a value
arrayList.Add("previous value is null  1");
arrayList.Insert(13,"this is 13th value in arraylist");

foreach (var arrayListItem1 in arrayList)
{
    Console.WriteLine(arrayListItem1);
}

Console.WriteLine("Capacity of arraylist now is1 {0}",arrayList.Capacity);
```

```
        //removing specific value

        arrayList.Remove("previous value is null");

        //removing specific index
        Console.WriteLine("\nafter using remove and RemoveAt function");
        arrayList.RemoveAt(2);
        foreach (var arrayListItem5 in arrayList)
        {
            Console.WriteLine(arrayListItem5);
        }

        //removes values from 7 to 12
        Console.WriteLine("\nafter using removerange function");
        arrayList.RemoveRange(0, 10);
        foreach (var arrayListItem6 in arrayList)
        {
            Console.WriteLine(arrayListItem6);
        }

        Console.ReadKey();
    }
}

class A
{
    private /*static*/ string[] name=new string[10];

    public string this[int i]
    {
        get { return name[i]; }
        set { name[i] = value;}
```

```
        }
    public  void show()
        {
            Console.WriteLine("show function called");
            foreach(string name1 in name)
                Console.WriteLine(name1);
        }
    }
}
```

D:\visual studio\ConsoleApp21_index_generics_arraylist\ConsoleApp21_index_generics_arraylist\bin\Debug\ConsoleApp21_index_generics_arraylist.exe

```
111
this is string value
4.32
123.45
True

previous value is null
Capacity of arraylist now is 8
111
this is string value
4.32
123.45
True

previous value is null
112
this is string value1
41.32
1231.45
True

this is 13th value in arraylist
previous value is null  1
Capacity of arraylist now is1 16

after using remove and RemoveAt function
111
this is string value
123.45
True

112
this is string value1
41.32
1231.45
True

this is 13th value in arraylist
previous value is null  1

after using removerange function

this is 13th value in arraylist
previous value is null  1
```

3.List

using System;

```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp22_list
{
    internal class Program
    {
        static void Main(string[] args)
        {
            List<string> subjects = new List<string>();
            subjects.Add("Webtech2");
            subjects.Add("python");
            subjects.Add("AIML");
            subjects.Insert(0,"MCA Subject List:");


            List<int> sub_code = new List<int>();
            sub_code.Add(101);
            sub_code.Add(102);
            sub_code.Add(103);


            Console.WriteLine("Capacity of subject list  is {0}",subjects.Capacity);
            Console.WriteLine("Capacity of sub_code list  is {0}",sub_code.Capacity);


            /*traversing through list
             * method 1: using for loop
             *
             * */
            for (int i = 0; i < subjects.Count; i++)
            {
                Console.WriteLine("List item:{0} : {1}", i, subjects[i]);
```

```
            }
            /*for(int i = 0; i < subjects.Count;i++)
            {
                Console.WriteLine("List item:{0} : {1}", i, subjects[i]);
            }*/
            //for loop not standard due to probability of out of range exception


            //method 2: using foreach loop - standard
            Console.WriteLine("\nSubjects code are");
            foreach(var v in sub_code)
            {
                Console.WriteLine("sub_code :{0}",v);
            }


            //method 3: ForEach Mthod of List class
            Console.WriteLine("\nmethod 3: ForEach Mthod of List class");
            subjects.ForEach(subject => Console.WriteLine("subjects_a:
{0}",subject));


            //Method 4: compacting ForEach method of list class
            Console.WriteLine("\nMethod 4: compacting ForEach method of list
class");
            subjects.ForEach(Console.WriteLine);


            //Method 5: using string.join()
            Console.WriteLine("\nMethod 5: using string.join()");
            Console.WriteLine(String.Join("\n", subjects));
            Console.ReadKey();

        }
    }
```

}

```
D:\visual studio\ConsoleApp22_list\ConsoleApp22_list\bin\Debug\ConsoleApp22_list.exe
Capacity of subject list  is 4
Capacity of sub_code list  is 4
List item:0 : MCA Subject List:
List item:1 : Webtech2
List item:2 : python
List item:3 : AIML

Subjects code are
sub_code :101
sub_code :102
sub_code :103

method 3: ForEach Mthod of List class
subjects_a: MCA Subject List:
subjects_a: Webtech2
subjects_a: python
subjects_a: AIML

Method 4: compacting ForEach method of list class
MCA Subject List:
Webtech2
python
AIML

Method 5: using string.join()
MCA Subject List:
Webtech2
python
AIML
```

<u>Reason and why generics and collections are used for high performance</u>

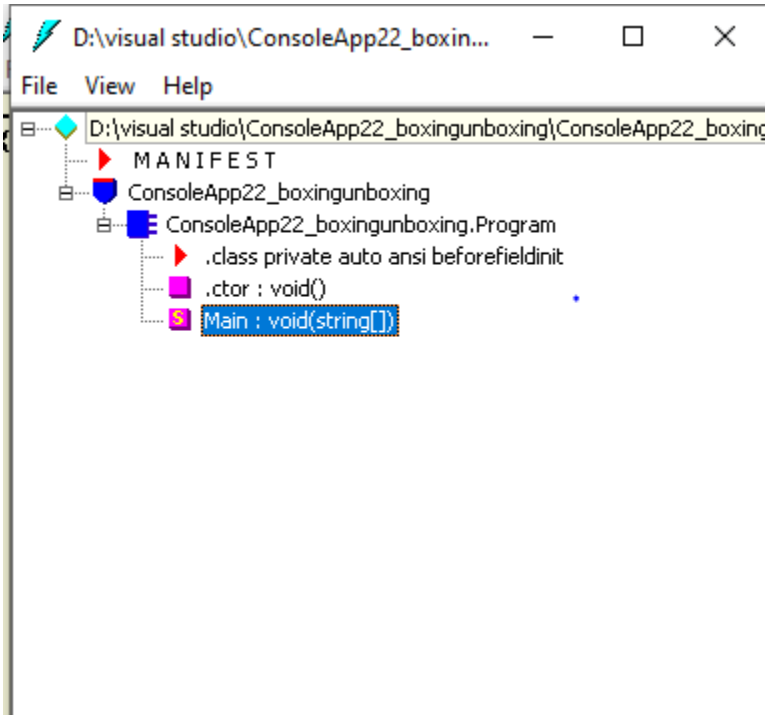
<u>Boxing and unboxing</u>

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp22_boxingunboxing
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int a = 101;
            Object obj = new Object();

            //boxing a primitvie value to ibject type
            obj = a;
            Console.WriteLine(obj.ToString());


            //unboxing object type to primitive type
            int b = (int)obj;
            Console.WriteLine(b);

            Console.ReadKey();
        }
    }
}
```

Opening .exe file through ildasm

C:\WINDOWS\system32\cmd.exe

```
The syntax of the command is incorrect.

C:\>cd Program Files (x86)\Microsoft SDKs\Windows\v8.1A\bin\NETFX 4.5.1 Tools>
The syntax of the command is incorrect.

C:\>cd "Program Files (x86)"

C:\Program Files (x86)>cd "Microsoft SDKs"

C:\Program Files (x86)\Microsoft SDKs>cd v8.1A
The system cannot find the path specified.

C:\Program Files (x86)\Microsoft SDKs>cd "Windows"

C:\Program Files (x86)\Microsoft SDKs\Windows>cd v8.1A
The system cannot find the path specified.

C:\Program Files (x86)\Microsoft SDKs\Windows>cd v10.0A

C:\Program Files (x86)\Microsoft SDKs\Windows\v10.0A>bin
'bin' is not recognized as an internal or external command,
operable program or batch file.

C:\Program Files (x86)\Microsoft SDKs\Windows\v10.0A>cd bin

C:\Program Files (x86)\Microsoft SDKs\Windows\v10.0A\bin>cd "NETFX 4.8 Tools"

C:\Program Files (x86)\Microsoft SDKs\Windows\v10.0A\bin\NETFX 4.8 Tools>ildasm.exe

C:\Program Files (x86)\Microsoft SDKs\Windows\v10.0A\bin\NETFX 4.8 Tools>
```



ConsoleApp22_boxingunboxing.Program::Main : void(string[])

Find Find Next

```
.method private hidebysig static void  Main(string[] args) cil managed
{
  .entrypoint
  // Code size       50 (0x32)
  .maxstack  1
  .locals init ([0] int32 a,
           [1] object obj,
           [2] int32 b)
  IL_0000:  nop
  IL_0001:  ldc.i4.s   101
  IL_0003:  stloc.0
  IL_0004:  newobj     instance void [mscorlib]System.Object::.ctor()
  IL_0009:  stloc.1
  IL_000a:  ldloc.0
  IL_000b:  box        [mscorlib]System.Int32
  IL_0010:  stloc.1
  IL_0011:  ldloc.1
  IL_0012:  callvirt   instance string [mscorlib]System.Object::ToString()
  IL_0017:  call       void [mscorlib]System.Console::WriteLine(string)
  IL_001c:  nop
  IL_001d:  ldloc.1
  IL_001e:  unbox.any  [mscorlib]System.Int32
  IL_0023:  stloc.2
  IL_0024:  ldloc.2
  IL_0025:  call       void [mscorlib]System.Console::WriteLine(int32)
  IL_002a:  nop
  IL_002b:  call       valuetype [mscorlib]System.ConsoleKeyInfo [mscorlib]System.Console::ReadKey()
  IL_0030:  pop
  IL_0031:  ret
} // end of method Program::Main
```

ConsoleApp22_boxingunboxing.Program::Main : void(string[])

Find Find Next

```
.method private hidebysig static void  Main(string[] args) cil managed
{
  .entrypoint
  // Code size       50 (0x32)
  .maxstack  1
  .locals init ([0] int32 a,
           [1] object obj,
           [2] int32 b)
  IL_0000:  nop
  IL_0001:  ldc.i4.s   101
  IL_0003:  stloc.0
  IL_0004:  newobj     instance void [mscorlib]System.Object::.ctor()
  IL_0009:  stloc.1
  IL_000a:  ldloc.0
  IL_000b:  box        [mscorlib]System.Int32
  IL_0010:  stloc.1
  IL_0011:  ldloc.1
  IL_0012:  callvirt   instance string [mscorlib]System.Object::ToString()
  IL_0017:  call       void [mscorlib]System.Console::WriteLine(string)
  IL_001c:  nop
  IL_001d:  ldloc.1
  IL_001e:  unbox.any  [mscorlib]System.Int32
  IL_0023:  stloc.2
  IL_0024:  ldloc.2
  IL_0025:  call       void [mscorlib]System.Console::WriteLine(int32)
  IL_002a:  nop
  IL_002b:  call       valuetype [mscorlib]System.ConsoleKeyInfo [mscorlib]System.Console::ReadKey()
  IL_0030:  pop
  IL_0031:  ret
} // end of method Program::Main
```

4.Stack

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp22_boxingunboxing
{
    internal class Program
    {
        static void Main(string[] args)
        {
           Stack<int> stack = new Stack<int>();
            stack.Push(11);
            stack.Push(12);
            stack.Push(13);
            stack.Push(14);
            stack.Push(15);


            foreach (var v in stack)
                Console.WriteLine(v);
            Console.WriteLine("Peek position is:{0}", stack.Peek());

            stack.Pop();
            Console.WriteLine("Peek position is:{0}",stack.Peek());

            stack.Pop();
            Console.WriteLine("Peek position is:{0}",stack.Peek());

            int[] arr = stack.ToArray();
            foreach(var v in arr)
                Console.WriteLine(v);
```

```
            Console.WriteLine("check if item 15 exist:{0}", stack.Contains(15));
            Console.WriteLine("check if item 15 exist:{0}", stack.Contains(1));
            Console.ReadKey();
        }
    }
}
```

```
D:\visual studio\ConsoleApp22_boxingunboxing\ConsoleApp22_boxingunboxing\bin\Debug\ConsoleApp22_boxingunboxing.exe
15
14
13
12
11
Peek position is:15
Peek position is:14
Peek position is:13
13
12
11
check if item 15 exist:False
check if item 15 exist:False
```

5.Queue

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp23_queue_stack
{
    internal class Program
    {
        static void Main(string[] args)
        {
            //queue
```

```
            Queue<int> q = new Queue<int>();
            q.Enqueue(111);
            q.Enqueue(112);
            q.Enqueue(113);
            q.Enqueue(114);
            q.Enqueue(115);
            q.Enqueue(116);

            Console.WriteLine("queue contains ");
            foreach (var v in q)
                Console.WriteLine(v);

            Console.WriteLine("Peek position is:{0}", q.Peek());
            q.Dequeue();

            Console.WriteLine("Peek position is:{0}", q.Peek());
            q.Dequeue();

            foreach (var v in q)
                Console.WriteLine(v);

            Console.WriteLine("After ToArray()");
           int[] s = q.ToArray();

            Console.WriteLine(s);

            Console.ReadKey();
        }
    }
}
```

```
D:\visual studio\ConsoleApp23_queue_stack\ConsoleApp23_queue_stack\bin\Debug\ConsoleApp23_queue_stack.exe
queue contains
111
112
113
114
115
116
Peek position is:111
Peek position is:112
113
114
115
116
After ToArray()
System.Int32[]
```

Dictionary

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp23_queue_stack
{
    internal class Program
    {
        static void Main(string[] args)
```

```
{
  //dictionary
   Dictionary<string,long> phoneDirectory=new Dictionary<string,long>();

   phoneDirectory.Add("A",5646554411);
   phoneDirectory.Add("B",5646554422);
   phoneDirectory.Add("C",5646554433);
   phoneDirectory.Add("D",5646554444);
   phoneDirectory.Add("E",5646554455);
   phoneDirectory.Add("F",5646554466);

   foreach(KeyValuePair<string,long> entry in phoneDirectory)
   {
      Console.WriteLine("Key:{0}  value:{1}",entry.Key,entry.Value);
      Console.WriteLine("----------------------------------------");
   }

   Console.WriteLine("Enter name to search phone number: ");
   var name=Console.ReadLine();
   foreach (KeyValuePair<string, long> entry in phoneDirectory)
   {
      //if(phoneDirectory.ContainsKey(name))
      //{
      //    Console.WriteLine("Search result key:{0} phone
no:{1}",name,phoneDirectory[name] );
      //    break;
      //}
      long res = 0;
      if (phoneDirectory.TryGetValue(name, out res))
      {
         //phoneDirectory.TryGetValue(name, out res);
         Console.WriteLine("Search result key:{0} phone no:{1}", name, res);
         break;
      }
      else
```

```
            {
                Console.WriteLine("no record found");
            }
        }

        Console.ReadKey();
    }
  }
}
```

D:\visual studio\ConsoleApp23_queue_stack\ConsoleApp23_queue_stack\bin\Debug\ConsoleApp23_queue_stack.exe

```
Key:A  value:5646554411
---------------------------------------
Key:B  value:5646554422
---------------------------------------
Key:C  value:5646554433
---------------------------------------
Key:D  value:5646554444
---------------------------------------
Key:E  value:5646554455
---------------------------------------
Key:F  value:5646554466
---------------------------------------
Enter name to search phone number:
A
Search result key:A phone no:5646554411
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp23_dictionary
{
    class Program
    {
        static void Main(string[] args)
        {

            //Stack<int> stack = new Stack<int>();
            //stack.Push(11);
            //stack.Push(12);
            //stack.Push(13);
            //stack.Push(14);
            //stack.Push(15);


            //foreach (var v in stack)
            //    Console.WriteLine(v);
            //Console.WriteLine("Peek position is:{0}", stack.Peek());

            //stack.Pop();
            //Console.WriteLine("Peek position is:{0}", stack.Peek());

            //stack.Pop();
            //Console.WriteLine("Peek position is:{0}", stack.Peek());

            //int[] arr = stack.ToArray();
            //foreach (var v in arr)
            //    Console.WriteLine(v);
```

```
//Console.WriteLine("check if item 15 exist:{0}", stack.Contains(15));
//Console.WriteLine("check if item 15 exist:{0}", stack.Contains(1));
//Console.ReadKey();




//queue
// Queue<int> q = new Queue<int>();
// q.Enqueue(111);
// q.Enqueue(112);
// q.Enqueue(113);
// q.Enqueue(114);
// q.Enqueue(115);
// q.Enqueue(116);

// Console.WriteLine("queue contains ");
// foreach (var v in q)
//     Console.WriteLine(v);

// Console.WriteLine("Peek position is:{0}", q.Peek());
// q.Dequeue();

// Console.WriteLine("Peek position is:{0}", q.Peek());
// q.Dequeue();

// foreach (var v in q)
//     Console.WriteLine(v);

// Console.WriteLine("After ToArray()");
//int[] s = q.ToArray();


// Console.WriteLine(s);
```

```
        //dictionary
        //Dictionary<string, long> phoneDirectory = new Dictionary<string,
long>();

        //phoneDirectory.Add("A", 5646554411);
        //phoneDirectory.Add("B", 5646554422);
        //phoneDirectory.Add("C", 5646554433);
        //phoneDirectory.Add("D", 5646554444);
        //phoneDirectory.Add("E", 5646554455);
        //phoneDirectory.Add("F", 5646554466);

        //foreach (KeyValuePair<string, long> entry in phoneDirectory)
        //{
        //    Console.WriteLine("Key:{0}  value:{1}", entry.Key, entry.Value);
        //    Console.WriteLine("----------------------------------------");
        //}

        //Console.WriteLine("Enter name to search phone number: ");
        //var name = Console.ReadLine();
        //foreach (KeyValuePair<string, long> entry in phoneDirectory)
        //{
        //    //if(phoneDirectory.ContainsKey(name))
        //    //{
        //    //    Console.WriteLine("Search result key:{0} phone
no:{1}",name,phoneDirectory[name] );
        //    //    break;
        //    //}
        //    long res = 0;
        //    if (phoneDirectory.TryGetValue(name, out res))
        //    {
        //        //phoneDirectory.TryGetValue(name, out res);
        //        Console.WriteLine("Search result key:{0} phone no:{1}", name,
res);
```

```
            //        break;
            //    }
            //    else
            //    {
            //        Console.WriteLine("no record found");
            //    }
            //}


            //creating dictionary with class objects
            Students[] students = new Students[5];
            students[0] = new Students() { Id = 1, Name = "Aniket" };
            students[1] = new Students() { Id = 2, Name = "Gaurav" };
            students[2] = new Students() { Id = 3, Name = "pranavi" };
            students[3] = new Students() { Id = 4, Name = "kushal" };
            students[4] = new Students() { Id = 5, Name = "Shafiya" };

            Dictionary<int, Students> studdictionary = new Dictionary<int,
Students>();
    studdictionary.Add(students[0].Id, students[0]);
            studdictionary.Add(students[1].Id, students[1]);
            studdictionary.Add(students[2].Id, students[2]);

            foreach (KeyValuePair<int,Students> entry in studdictionary)
            {
                Students s1 = (Students)entry.Value;
                Console.WriteLine("Key:{0}  value:{1}", entry.Key, s1.Name);
                Console.WriteLine("----------------------------------------");
            }

            Console.ReadKey();
        }
    }
   public class Students
    {
```

```
        public int Id { get; set; }
        public string Name { get; set; } //public int Marks { get; set; }

    }
  }
```

```
C:\Users\studentadmin\source\repos\ConsoleApp23_dictionary\ConsoleApp23_dictionary\bin\Debug\ConsoleApp23_dictionary.exe
Key:1  value:Aniket
----------------------------------------
Key:2  value:Gaurav
----------------------------------------
Key:3  value:pranavi
----------------------------------------
```

07May2022

Practical 27:

Exception Handling in C#

Code/Output:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
/// <summary>
/// Excepotion handling
/// exception abuse- when exception handling is used for business logic control
flow
/// </summary>
namespace ConsoleApp24_exception_handling
{
    class Program
    {
        static void Main(string[] args)
        {

            try
             {
                Console.WriteLine("Enter Numerator: ");
                int numerator,denominator,res /*=
Convert.ToInt32(Console.ReadLine())*/;

                if (int.TryParse(Console.ReadLine(), out numerator)||
(numerator<int.MinValue && numerator>int.MaxValue) )
                {
                    Console.WriteLine("Enter Denominator: ");
                    //
```

```
                //int denominator;
                if (int.TryParse(Console.ReadLine(), out denominator))
                {
                    if (denominator > 0)
                    {
                        res = numerator / denominator;
                        Console.WriteLine("res:{0}", res);
                    }
                    else
                    {
                        Console.WriteLine("Denominator cannot be zero");
                    }
                 }
            }
            else
            {
                throw new OverflowException("Overflow Exception");

                //Console.WriteLine("Enter Integer Value");
            }


            //int res = numerator / denominator;
            //Console.WriteLine("res:{0}", res);
        }

        catch(Exception e)
        {
            Console.WriteLine(e.Message);
        }

        Console.ReadKey();
    }
  }
}
```

```
D:\39kunalvb\ConsoleApp24_exception_handling\ConsoleApp24_exception_handling\bin\Debug\ConsoleApp24_exception_handling.exe
Enter Numerator:
10
Enter Denominator:
2
res:5
```

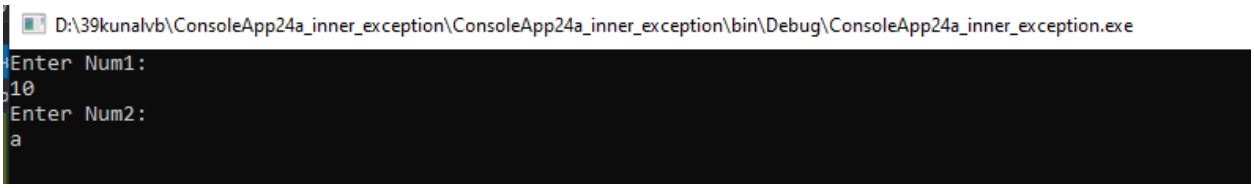
```
D:\39kunalvb\ConsoleApp24_exception_handling\ConsoleApp24_exception_handling\bin\Debug\ConsoleApp24_exception_handling.exe
Enter Numerator:
10
Enter Denominator:
0
Denominator cannot be zero
```

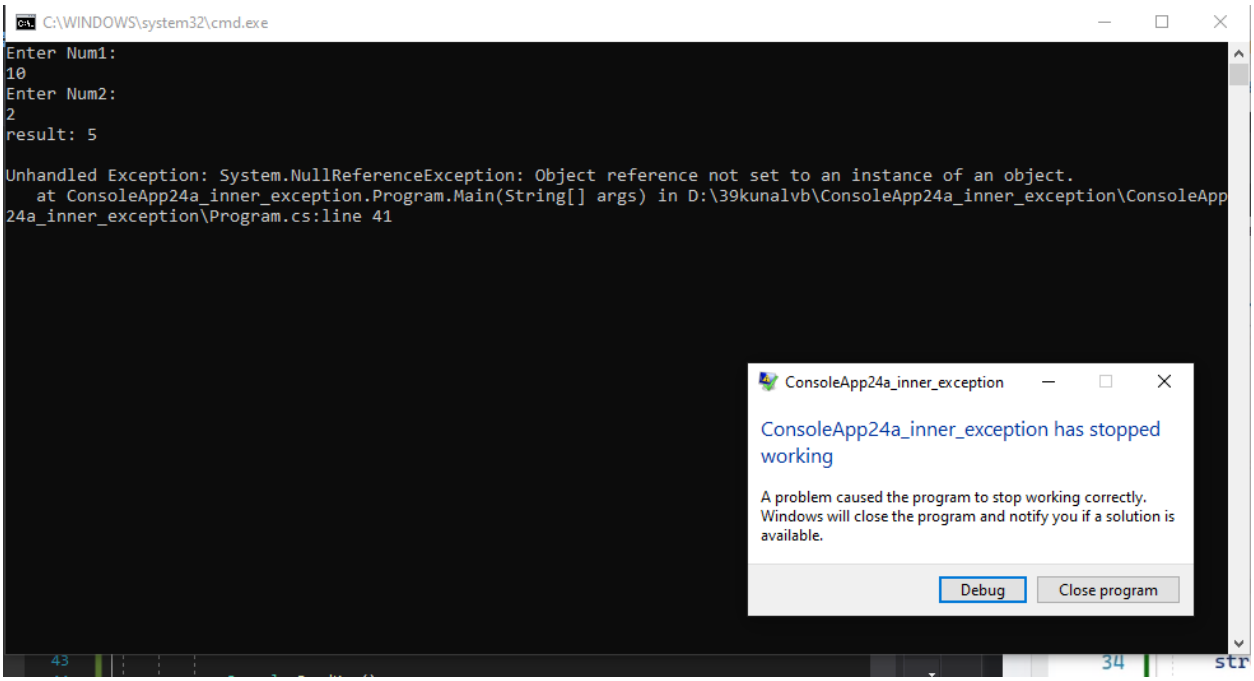
```
D:\39kunalvb\ConsoleApp24_exception_handling\ConsoleApp24_exception_handling\bin\Debug\ConsoleApp24_exception_handling.exe
Enter Numerator:
3214758690765436789076 5
Overflow Exception
```

<u>Inner exception with writing in text file</u>

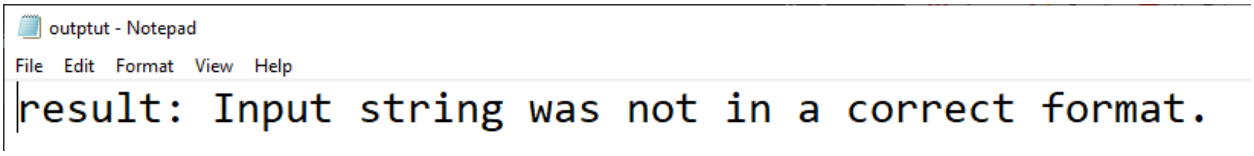
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace ConsoleApp24a_inner_exception
{
    class Program
    {
        static void Main(string[] args)
        {
            StreamWriter streamWriter = null;
            try
            {

                Console.WriteLine("Enter Num1: ");
                int num1 = Int32.Parse(Console.ReadLine());
```

```
            Console.WriteLine("Enter Num2: ");
            int num2 = Int32.Parse(Console.ReadLine());
            int res;
            res = num1 / num2;
            Console.WriteLine("result: {0}", res);
        }
        catch(Exception e)
        {
            streamWriter = new StreamWriter(@"D:\outptut.log");
            streamWriter.WriteLine("result: {0}",e.Message);
        }

        finally
        {
            streamWriter.Close();
        }

        Console.ReadKey();
    }
  }
}
```





Output.log

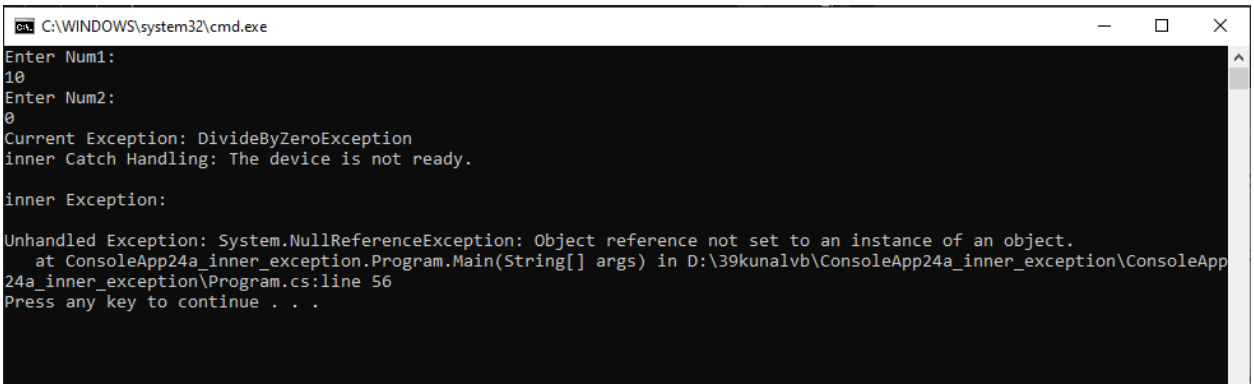


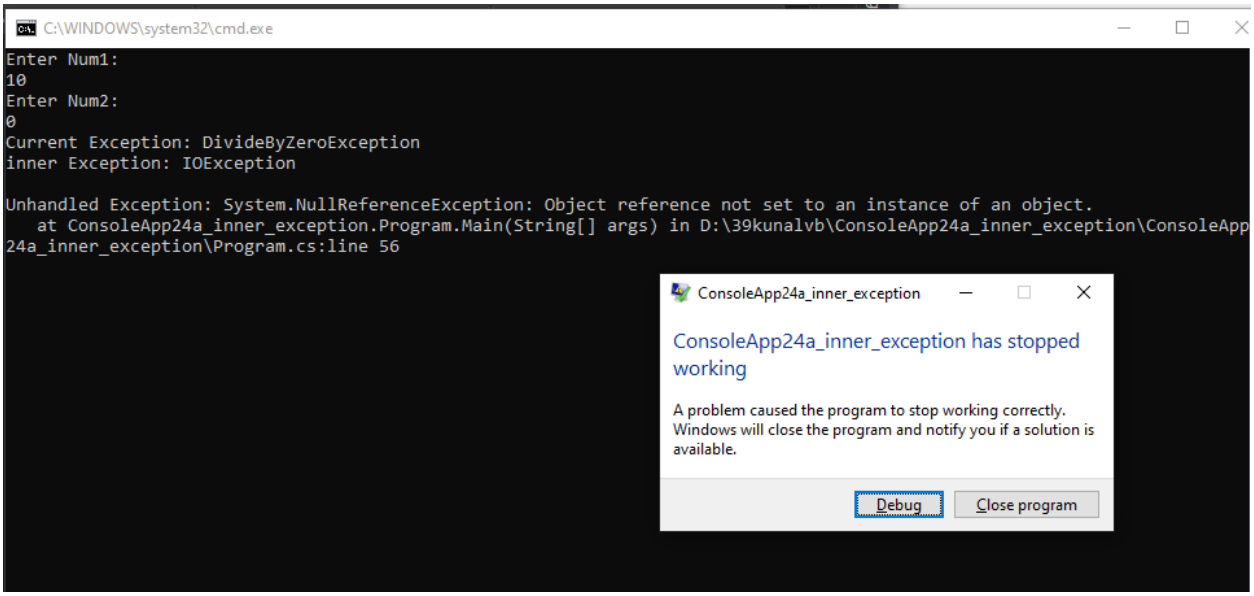
2.Inner Exception

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace ConsoleApp24a_inner_exception
{
    class Program
    {
        static void Main(string[] args)
        {
            StreamWriter streamWriter = null;
            try
```

```
        {


            Console.WriteLine("Enter Num1: ");
            int num1 = Int32.Parse(Console.ReadLine());

            Console.WriteLine("Enter Num2: ");
            int num2 = Int32.Parse(Console.ReadLine());
            int res;

             res = num1 / num2;
            Console.WriteLine("result: {0}", res);



        }
        catch(Exception ex)
        {
            try
            {
                streamWriter = new StreamWriter(@"E:\outptut.log");
                streamWriter.WriteLine("exception occured: {0}", ex.Message);
                //throw new Exception(ex.Message, ex);
            }

            catch(Exception ex2)
            {
Console.WriteLine("Current Exception: {0} ", ex.GetType().Name);
 // Console.WriteLine("inner Catch Handling: {0} ", ex2.Message);
Console.WriteLine("inner Exception: {0} ", ex2.GetType().Name);
                //throw new Exception()
            }
            //Console.WriteLine("Current Exception: {0} ", ex.Message);
            //Console.WriteLine("inner Exception: {0} ", ex.InnerException);
```

```
        }

        finally
        {
            streamWriter.Close();
        }

        Console.ReadKey();
    }
  }
}
```

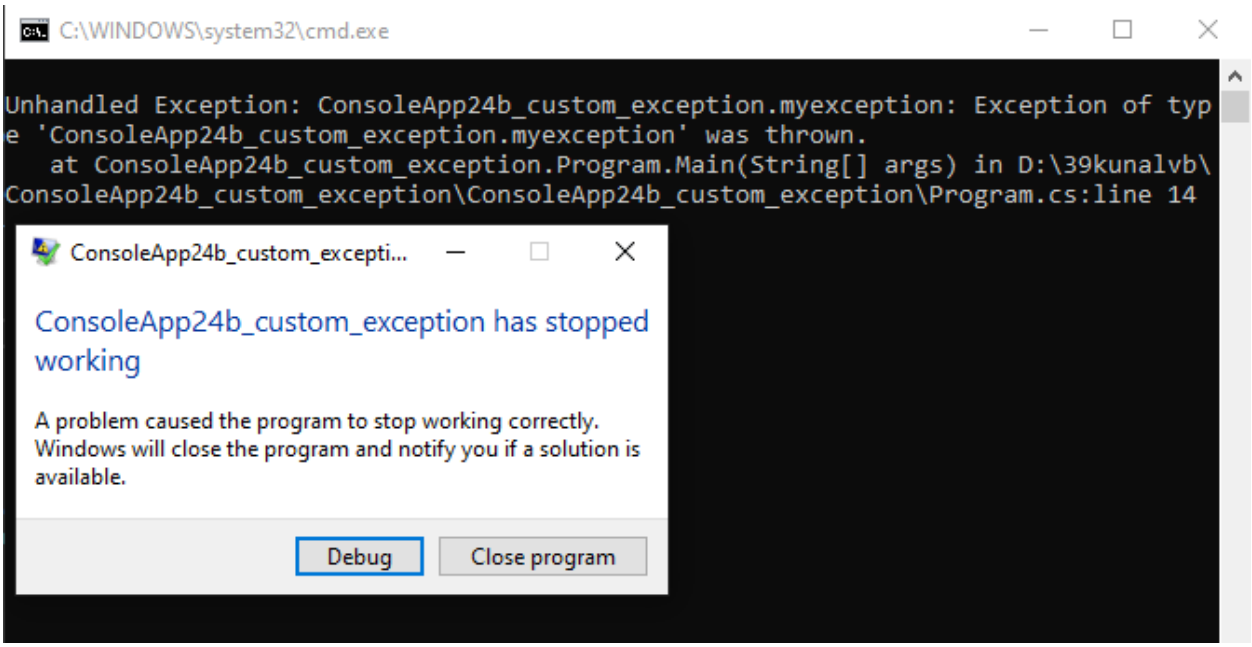




3.Custom Exception

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp24b_custom_exception
{
    class Program
    {
        static void Main(string[] args)
        {

            throw new myexception();
        }
    }

    public class myexception:Exception
    {

    }
```

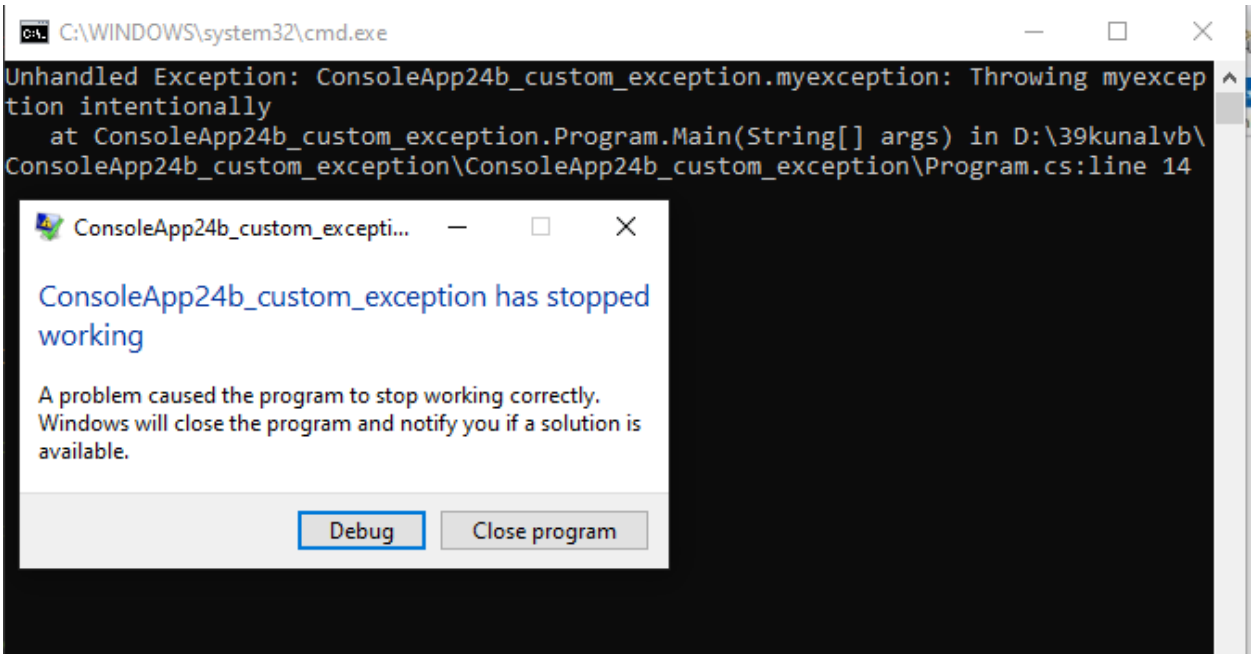
}



<u>By calling Exception(msg) constructor in custom exception(going to definition) and using base keyword to call</u> customException()

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp24b_custom_exception
{
    class Program
    {
        static void Main(string[] args)
        {

            throw new myexception("Throwing myexception intentionally");
```

```
            Console.ReadKey();
        }
    }
    //to create custom exception create class inheriting
    //the exception base class
    public class myexception:Exception
    {
        //base(msg) will call Exception(String msg ) Constructor
        public myexception(String msg) : base(msg)
        {

        }
    }

}
```



Program.cs | Exception [from metadata]

mscorlib | System.Exception | Exception(string message)

```
1   Assembly mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
4
5   using ...
10
11  namespace System
12  {
13      public class Exception : ISerializable, _Exception
21      {
22          public Exception();
26          public Exception(string message);
35          public Exception(string message, Exception innerException);
48          protected Exception(SerializationInfo info, StreamingContext context);
69
```

C:\WINDOWS\system32\cmd.exe

```
   at ConsoleApp24b_custom_exception.Program.Main(String[] args) in D:\
39kunalvb\ConsoleApp24b_custom_exception\ConsoleApp24b_custom_exception
\Program.cs:line 20

Unhandled Exception: ConsoleApp24b_custom_exception.myexception: Throwi
ng myexception intentionally
   at ConsoleApp24b_custom_exception.Program.Main(String[] args) in D:\
39kunalvb\ConsoleApp24b_custom_exception\ConsoleApp24b_custom_exception
\Program.cs:line 34
```

ConsoleApp24b_custom_excepti...

ConsoleApp24b_custom_exception has stopped working

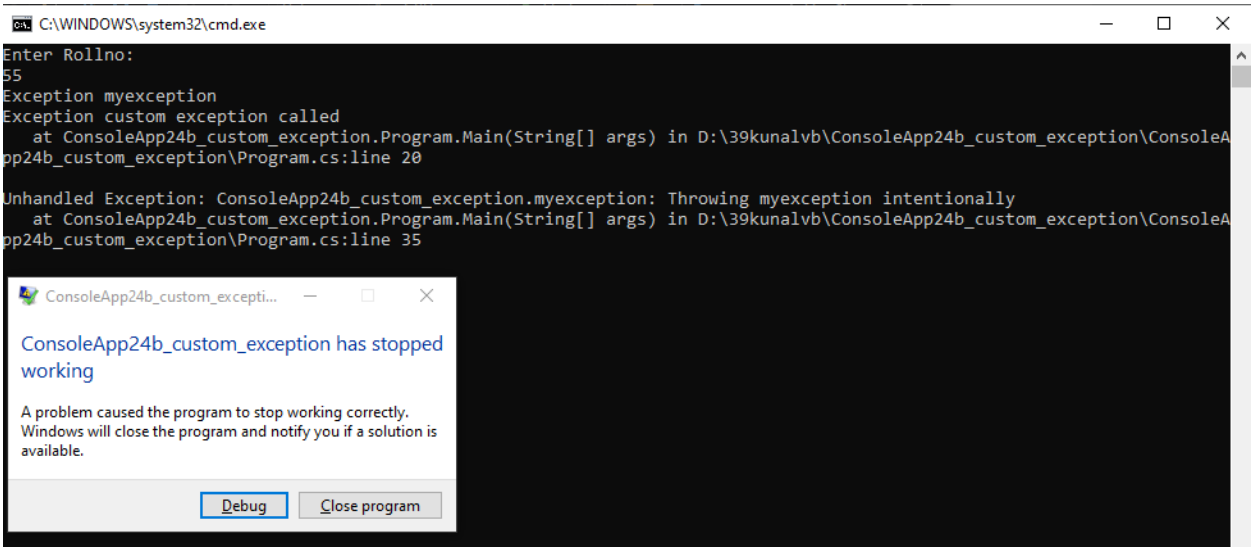
A problem caused the program to stop working correctly. Windows will close the program and notify you if a solution is available.

Debug | Close program

Methods in exception handling

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp24b_custom_exception
{
    class Program
    {
        static void Main(string[] args)
        {

            //
            Console.WriteLine("Enter Rollno: ");
            int rollno = Int32.Parse(Console.ReadLine());
            try {

                if (rollno < 0 || rollno > 47)
                    throw new myexception("custom exception called");

                else
                    Console.WriteLine("Rollno: {0}", rollno);
            }

            catch(myexception ex)
            {
                Console.WriteLine("Exception {0}", ex.GetType().Name);
                Console.WriteLine("Exception {0}", ex.Message);
                Console.WriteLine(ex.StackTrace);
            }
```

```
            throw new myexception("Throwing myexception intentionally");
            Console.ReadKey();
        }
    }
    //to create custom exception create class inheriting
    //the exception base class
    public class myexception:Exception
    {
        //base(msg) will call Exception(String msg ) Constructor
        public myexception(String msg) : base(msg)
        {

        }

        public myexception(String msg,Exception innerException) :
base(msg,innerException)
        {

        }
    }

}
```



07may2022

Practical 28:

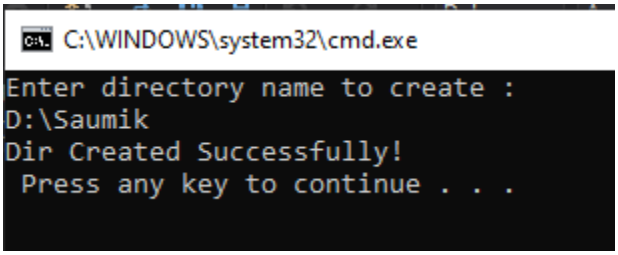
File Handling in C#

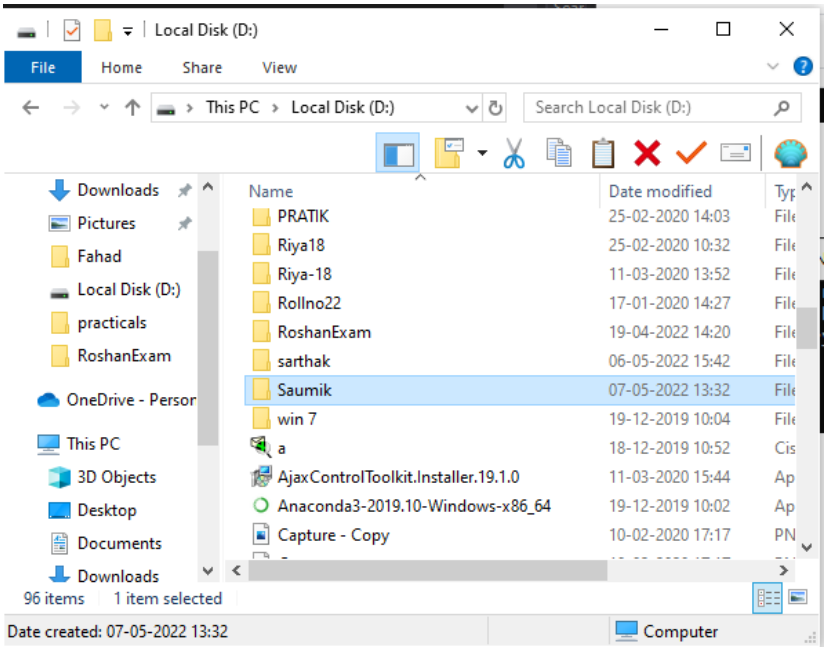
Code/Output:-

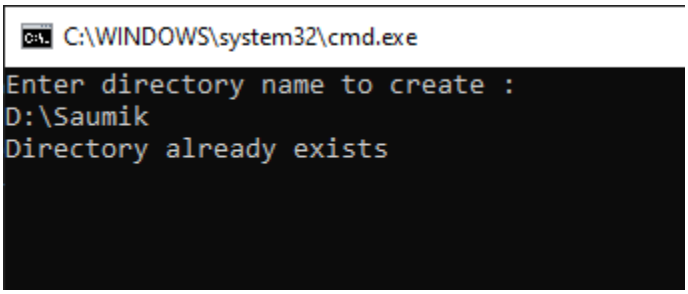
1.creating Directory

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;

/// <summary>
/// File retrieving information about drive directory and files
/// system.io provides 3 information classsess
/// Driveinfo
/// </summary>
```

```
namespace ConsoleApp25_filehandling
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter directory name to create : ");
            var dir = Console.ReadLine();

            if (!Directory.Exists(dir))
            {
                Directory.CreateDirectory(dir);
                Console.WriteLine("Dir Created Successfully! ");
            }

            else
            {
                Console.WriteLine("Directory already exists");
            }
          Console.ReadKey();
        }
    }
}
```







2.creating file

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace ConsoleApp25_filehandling
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter directory name to create : ");
            var dir = Console.ReadLine();
```

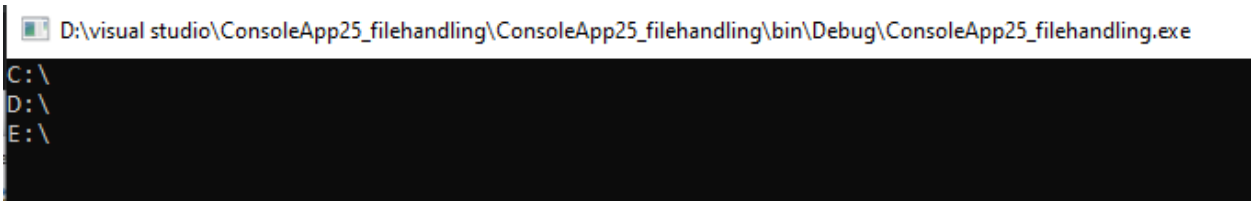
```
            if(!Directory.Exists(dir))
            {
                Directory.CreateDirectory(dir);
                Console.WriteLine("Dir Created Successfully! ");
            }

            else
            {
                Console.WriteLine("Directory already exists");
                Console.WriteLine("Enter Filename to create");
                String filename = Console.ReadLine();

                var path = Path.Combine(dir,filename);
                if(!File.Exists(path))
                {
                    File.Create(path);
                    Console.WriteLine("File Created:{0}", path);
                }
                else
                {
                    Console.WriteLine("File Already Exits");
                }

            }

            Console.ReadKey();
        }
    }
}
```

```
D:\39kunalvb\ConsoleApp25_filehandling\ConsoleApp25_filehandling\bin\Debug\ConsoleApp25_filehandling.exe
Enter directory name to create :
D:\Saumik
Directory already exists
Enter Filename to create
hello.txt
File Created:D:\Saumik\hello.txt
```

3.DriveInfo

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace ConsoleApp25_filehandling
{
    internal class Program
    {
        static void Main(string[] args)
        {
```

```
            DriveInfo[] driveInfo = DriveInfo.GetDrives();
            foreach(DriveInfo drive in driveInfo)
            {
                Console.WriteLine(drive.Name);
            }
            Console.ReadKey();
        }
    }
}
```



3a.Driveinfo properties

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;

/// <summary>
/// File retrieving information about drive directory and files
/// system.io provides 3 information classes
/// Driveinfo
/// </summary>
namespace ConsoleApp25_filehandling
{
```

```
internal class Program
{
    static void Main(string[] args)
    {


        DriveInfo[] driveInfo = DriveInfo.GetDrives();
        foreach(DriveInfo drive in driveInfo)
        {
            Console.WriteLine(drive.Name);
        }



        Console.WriteLine("Enter the drive name to know more: ");
        string drive1=Console.ReadLine();

        DriveInfo driveinfo1 = new DriveInfo(drive1);
        Console.WriteLine("Name:{0} ",driveinfo1.Name);
        Console.WriteLine("DriveType:{0} ",driveinfo1.DriveType);
        Console.WriteLine("DriveFormat:{0} ",driveinfo1.DriveFormat);
        Console.WriteLine("Volume Label:{0} ",driveinfo1.VolumeLabel);
        Console.WriteLine("TotalSize:{0}
",driveinfo1.TotalSize/1073741824+"GB");
        Console.WriteLine("TotalFreeSpace:{0}
",driveinfo1.AvailableFreeSpace/1073741824+ "GB");
        Console.WriteLine("driveinfo1.IsReady:{0} ",driveinfo1.IsReady);
        Console.WriteLine("RootDirectory:{0} ",driveinfo1.RootDirectory);
        Console.WriteLine("driveinfo1.GetType().FullName:{0} ",
driveinfo1.GetType().FullName);



        Console.ReadKey();

    }
}
```

}

```
D:\visual studio\ConsoleApp25_filehandling\ConsoleApp25_filehandling\bin\Debug\ConsoleApp25_filehandling.exe
C:\
D:\
E:\
Enter the drive name to know more:
D:\
Name:D:\
DriveType:Fixed
DriveFormat:NTFS
Volume Label:
TotalSize:260GB
TotalFreeSpace:238GB
driveinfo1.IsReady:True
RootDirectory:D:\
driveinfo1.GetType().FullName:System.IO.DriveInfo
```

Filesystem and directoryinfo

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace ConsoleApp25_filehandling
{
    internal class Program
    {
        static void Main(string[] args)
        {
            DriveInfo[] driveInfo = DriveInfo.GetDrives();
             foreach (DriveInfo drive in driveInfo)
             {
```

```
            Console.WriteLine(drive.Name);
        }


        Console.WriteLine("Enter the drive name to know more: ");
        string drive2 = Console.ReadLine();
        DirectoryInfo directory1 = new DirectoryInfo(drive2);
        Console.WriteLine("Name:{0}", directory1.Name);
        Console.WriteLine("FullName:{0}", directory1.FullName);
        Console.WriteLine("Parent:{0}", directory1.Parent);
        Console.WriteLine("Attributes:{0}", directory1.Attributes);
        Console.WriteLine("CreationTime:{0}", directory1.CreationTime);
        Console.WriteLine("Extension:{0}", directory1.Extension);


        DirectoryInfo[] subdirectories = directory1.GetDirectories();
        Console.WriteLine("\nSubDirectory: ");

        foreach(DirectoryInfo subdirectory in subdirectories)
        {
            Console.WriteLine("{0}",subdirectory.Name);
        }

        FileSystemInfo[] directory2 = directory1.GetFileSystemInfos();
        foreach(FileSystemInfo fsi in directory2)
        {
            Console.WriteLine("{0}",fsi.Name);
        }

        Console.WriteLine("Last Accessed Time:{0}",
directory1.LastAccessTime);
        //Console.WriteLine("Name:{0}", directory1.Name);
        Console.ReadKey();

    }
```

```
    }
}
```

D:\visual studio\ConsoleApp25_filehandling\ConsoleApp25_filehandling\bin\Debug\ConsoleApp25_filehandling.exe

```
C:\
D:\
E:\
Enter the drive name to know more:
D:\saumik_vs
Name:saumik_vs
FullName:D:\saumik_vs
Parent:
Attributes:Directory
CreationTime:08-05-2022 09:59:00
Extension:

SubDirectory:
ConsoleApp1
ConsoleApp10
ConsoleApp11_constructor
ConsoleApp12_method_hiding
ConsoleApp13_polymorphism
ConsoleApp14_abstract
ConsoleApp15_interface
ConsoleApp16_extensionmethod
ConsoleApp16_partialclass
ConsoleApp17_methodoverloading
ConsoleApp17_sealedclass
ConsoleApp18_enum
ConsoleApp19a
ConsoleApp19b
ConsoleApp19_delegates
ConsoleApp20_multicastdelegate
ConsoleApp21a_generics
```

For checking extension type

D:\visual studio\ConsoleApp25_filehandling\ConsoleApp25_filehandling\bin\Debug\ConsoleApp25_filehandling.exe

```
C:\
D:\
E:\
Enter the drive name to know more:
D:\saumik06may.rar
Name:saumik06may.rar
FullName:D:\saumik06may.rar
Parent:
Attributes:Archive
CreationTime:06-05-2022 17:09:14
Extension:.rar
```

FileInfo

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace ConsoleApp25_filehandling
{
    public class DriveA
    {
        public static DriveInfo[] driveInfo = DriveInfo.GetDrives();
        public static string file;


        public static void Createfoldr()
        {    }
        public static void infodir()
        {
           // DriveInfo[] driveInfo = DriveInfo.GetDrives();
            foreach (DriveInfo drive in driveInfo)
            {
                Console.WriteLine(drive.Name);
            }
         }

        public static void driveinfoa()
        {
            Console.WriteLine("Enter the drive name to know more: ");
            string drive2 = Console.ReadLine();
            DirectoryInfo directory1 = new DirectoryInfo(drive2);
```

```
            Console.WriteLine("Name:{0}", directory1.Name);
            Console.WriteLine("FullName:{0}", directory1.FullName);
            Console.WriteLine("Parent:{0}", directory1.Parent);
            Console.WriteLine("Attributes:{0}", directory1.Attributes);
            Console.WriteLine("CreationTime:{0}", directory1.CreationTime);
            Console.WriteLine("Extension:{0}", directory1.Extension);
            Console.WriteLine("Last Accessed Time:{0}",
directory1.LastAccessTime);

            DirectoryInfo[] subdirectories = directory1.GetDirectories();
            Console.WriteLine("\nSubDirectory: ");

            foreach (DirectoryInfo subdirectory in subdirectories)
            {
                Console.WriteLine("{0}", subdirectory.Name);
            }

            FileSystemInfo[] directory2 = directory1.GetFileSystemInfos();
            foreach (FileSystemInfo fsi in directory2)
            {
                Console.WriteLine("{0}", fsi.Name);
            }
            Console.WriteLine("Last Accessed Time:{0}",
directory1.LastAccessTime);
        }

        public static void getFileInfo()
        {
            Console.WriteLine("Enter the filepath to know more about");
            file=Console.ReadLine();
            FileInfo fileInfo = new FileInfo(file);
            Console.WriteLine("Name:{0}",fileInfo.Name);
            Console.WriteLine("FullName:{0}", fileInfo.FullName);
            Console.WriteLine("Length:{0}", fileInfo.Length);
            Console.WriteLine("CreationTime:{0}", fileInfo.CreationTime);
```

```
            Console.WriteLine("DirectoryName:{0}", fileInfo.DirectoryName);
            Console.WriteLine("Attributes:{0}", fileInfo.Attributes);
            Console.WriteLine("Extension:{0}", fileInfo.Extension);
            Console.WriteLine("fileInfo.GetType().Name:{0}",
fileInfo.GetType().Name);
            Console.WriteLine("LastAccessTime:{0}", fileInfo.LastAccessTime);
                }

    }
    internal class Program
    {
        static void Main(string[] args)
        {
            DriveInfo[] driveInfo = DriveInfo.GetDrives();
            foreach (DriveInfo drive in driveInfo)
            {
                Console.WriteLine(drive.Name);
            }
            DriveA.getFileInfo();
            Console.ReadKey();
        }
    }
}
```

```
D:\visual studio\ConsoleApp25_filehandling\ConsoleApp25_filehandling\bin\Debug\ConsoleApp25_filehandling.exe
C:\
D:\
E:\
Enter the filepath to know more about
D:\saumik_vs\Score.xlsx
Name:Score.xlsx
FullName:D:\saumik_vs\Score.xlsx
Length:11224
CreationTime:08-05-2022 10:18:19
DirectoryName:D:\saumik_vs
Attributes:Archive
Extension:.xlsx
fileInfo.GetType().Name:FileInfo
LastAccessTime:08-05-2022 10:18:19
```

4.FileStream Read/Write

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace ConsoleApp26_filereadwrite
{
    internal class Program
    {
        static void Main(string[] args)
        {
            //file read write using various classes
            filestreamwr();
            filestreamread(); //uncomment to perform read function

        }
        //Writing and reading file with File Stream class

        static void filestreamwr()
        {
            string filename = @"D:\visual studio\file_handling\filehandle.txt";
            //var text = "hello";
```

```
            FileStream filestream = new FileStream(filename,
FileMode.OpenOrCreate);

            for (int i = 65; i <= 90; i++)
            {
                filestream.WriteByte((byte)i);
                filestream.WriteByte((byte)i);
            }
            //byte[] bytes1 = Encoding.UTF8.GetBytes(text);
            //filestream.Write(bytes1, 0, bytes1.Length);
            filestream.Flush();
            filestream.Close();
            Console.WriteLine("File is created and Data is written");
        }

        static void filestreamread()
        {
            string filename = @"D:\visual studio\file_handling\filehandle.txt";
            var text = "Hello";
       FileStream filestream = new FileStream(filename, FileMode.OpenOrCreate);
            int i;
            while((i=filestream.ReadByte())!=-1)
            { //Console.Write((char)i);
                Console.Write("{0} ",i);
            }

            Console.WriteLine("\nEnd of file");
        }
    }
}
```

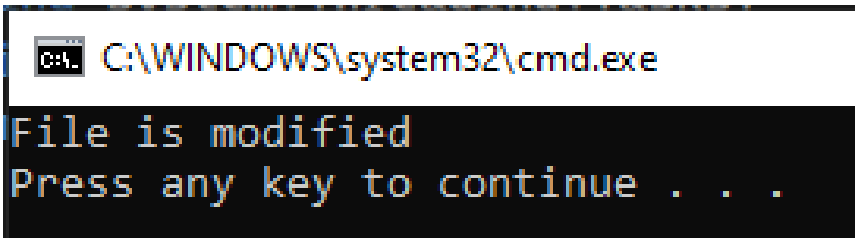
```
C:\WINDOWS\system32\cmd.exe
File is created and Data is written
Press any key to continue . . .
```

```
C:\WINDOWS\system32\cmd.exe
65 65 66 66 67 67 68 68 69 69 70 70 71 71 72 72 73 73 74 74 75 75 76 76 77 77 78 78 79 79 80 80 81 81 82 82 83 83 84 84
85 85 86 86 87 87 88 88 89 89 90 90
End of file
Press any key to continue . . .
```

<u>Using Streamwriter read and write</u>

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace ConsoleApp26_filereadwrite
{
    internal class Program
    {
        static void Main(string[] args)
        {
            streamwriterwr();
            streamreaderread();
            Console.ReadKey();
        }


        //write and read with stream writer
        static void streamwriterwr()
        {
```

```
            string filename = @"D:\visual studio\file_handling\filehandle_stream.txt";
            FileStream filestream = new FileStream(filename,
FileMode.OpenOrCreate);

            //accepts the obj of filestream to read and write
            StreamWriter streamWriter = new StreamWriter(filestream);
            streamWriter.WriteLine("Hello and we are using streamwriter111");
// filestream.Flush();
            streamWriter.Flush();
            streamWriter.Close();
            filestream.Close();
            Console.WriteLine("File is modified");
        }

        static void streamreaderread()
        {
            string filename = @"D:\visual studio\file_handling\filehandle_stream.txt";
            FileStream filestream = new FileStream(filename,
FileMode.OpenOrCreate);

            StreamReader streamReader = new StreamReader(filestream);
            string data=streamReader.ReadToEnd();
            Console.WriteLine(data);
            filestream.Flush();
            filestream.Close();
            Console.WriteLine("File Contents");
        }
    }
}
```



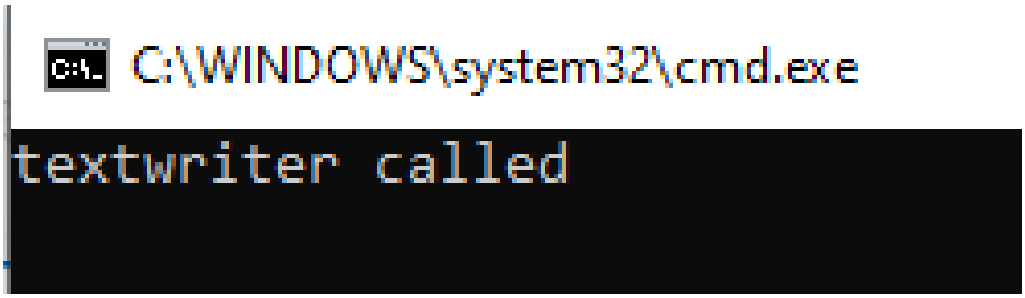
C:\WINDOWS\system32\cmd.exe

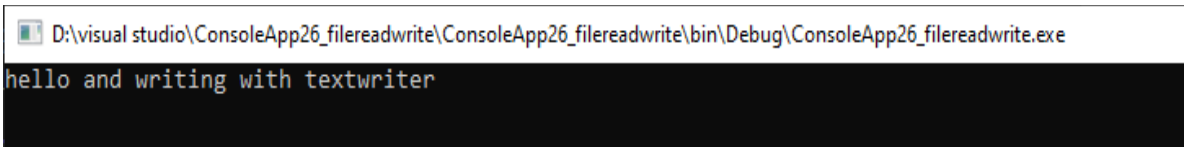
```
Hello and we are using streamwriter111

File Contents
Press any key to continue . . .
```

Textwriter and TextReader

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace ConsoleApp26_filereadwrite
{
    internal class Program
    {
        static void Main(string[] args)
        {
textwriterwr();
txread();
    Console.ReadKey();
       }

        //Read/Writer with textwriter
        static void textwriterwr()
        {
            string filename = @"D:\visual studio\file_handling\filehandle_textwr.txt";

           TextWriter teamwriter = File.CreateText(filename);

            teamwriter.WriteLine("hello and writing with textwriter");
            teamwriter.Close();
            Console.WriteLine("textwriter called");

        }

        static void txread()
        {
            string filename = @"D:\visual studio\file_handling\filehandle_textwr.txt";
```

```
        TextReader txtreader = File.OpenText(filename);

        Console.WriteLine(txtreader.ReadToEnd());
        txtreader.Close();

    }
  }
}
```

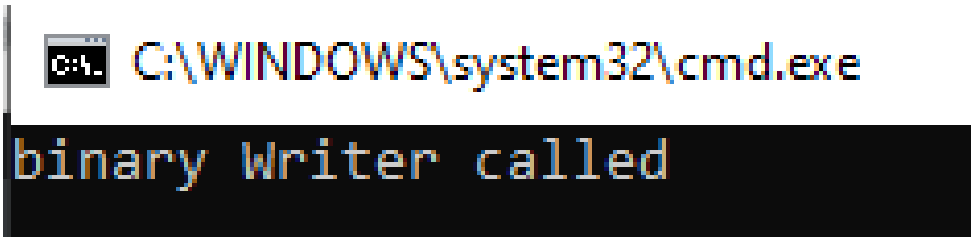


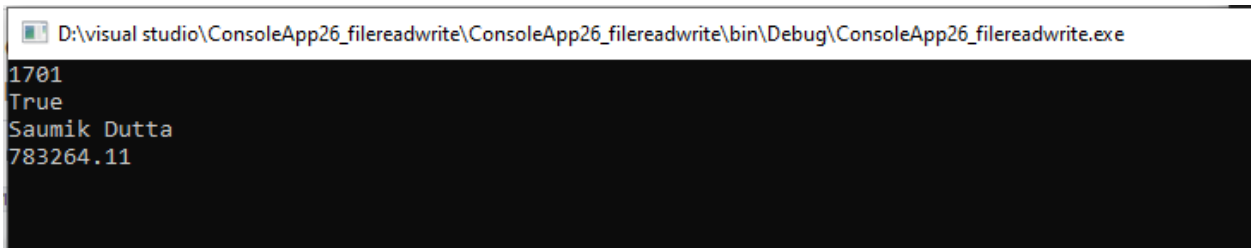


<u>BinaryWriter and BinaryReader</u>

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace ConsoleApp26_filereadwrite
{
    internal class Program
    {
        static void Main(string[] args)
        {
            binarywriter();
            binaryreader();
            Console.ReadKey();
        }


        //binarywriter and binaryreader
        static void binarywriter()
        {
            string filename = @"D:\visual studio\file_handling\filehandle_bin.dat";
            FileStream filestream = new FileStream(filename,
FileMode.OpenOrCreate);
            BinaryWriter bw=new BinaryWriter(filestream);
            bw.Write(1701);
            bw.Write(true);
            bw.Write("Saumik Dutta");
            bw.Write(783264.11);

            bw.Flush();
            bw.Close();
            filestream.Close();
            Console.WriteLine("binary Writer called");
        }
```

```
        static void binaryreader()
        {
            string filename = @"D:\visual studio\file_handling\filehandle_bin.dat";
            FileStream filestream = new FileStream(filename,
FileMode.OpenOrCreate);
            BinaryReader br = new BinaryReader(filestream);

            Console.WriteLine(br.ReadInt32());
            Console.WriteLine(br.ReadBoolean());
            Console.WriteLine(br.ReadString());
            Console.WriteLine(br.ReadDouble());


            filestream.Close();
        }
    }
}
```





08May2022

Practical 29:

Serialization and Deserialization

Code/Output:-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Binary;
using System.IO;
//required to serialize and deserialize objects

namespace ConsoleApp27_serializationdeserialization
{
    internal class Program
    {
        //serialization and deserialization
        static void Main(string[] args)
        {
            /*
             * step1:- using System.Runtime.Serialization
             * using System.Runtime.Serialization.Formatters.Binary;
             * create serializable class student
             * create objects of serializable class
             * serialize (write into file) and deserialize(read from file_
             * students objects
             * */

            Student[] students = new Student[]
            {
 new Student(){Id = 101, Name = "Saumik Dutta", email="saumik@gmail.com" },
 new Student(){Id = 102,Name ="Akshay Shetty", email="akshetty@gmail.com"},
```

```
new Student(){Id = 103,Name="sanjay dewan",email="dewansanjay@gmail.com"
}  };

            serializestudent(students);
            deserializestudent();
            Console.ReadKey();
        }

        static void serializestudent(Student[] students)
        {
            string file = @"D:\visual studio\file_handling\studentdata.txt";
            FileStream fs=new FileStream(file, FileMode.Create);

            BinaryFormatter binaryformatter = new BinaryFormatter();
            foreach (Student student in students)
            {    binaryformatter.Serialize(fs, student);        }
            fs.Close();
            Console.WriteLine("Serialization completed!!!");
        }

        static void deserializestudent()
        {
            string file = @"D:\visual studio\file_handling\studentdata.txt";
            FileStream fs = new FileStream(file, FileMode.Open);

            BinaryFormatter binaryformatter = new BinaryFormatter();
            List<Student> students = new List<Student>();
            while(fs.Position!=fs.Length)
            {
                students.Add((Student)binaryformatter.Deserialize(fs));
            }
            foreach (Student student in students)
            {
                Console.WriteLine("Id:{0} Name:{1} Email:{2}",
student.Id,student.Name,student.email);
```

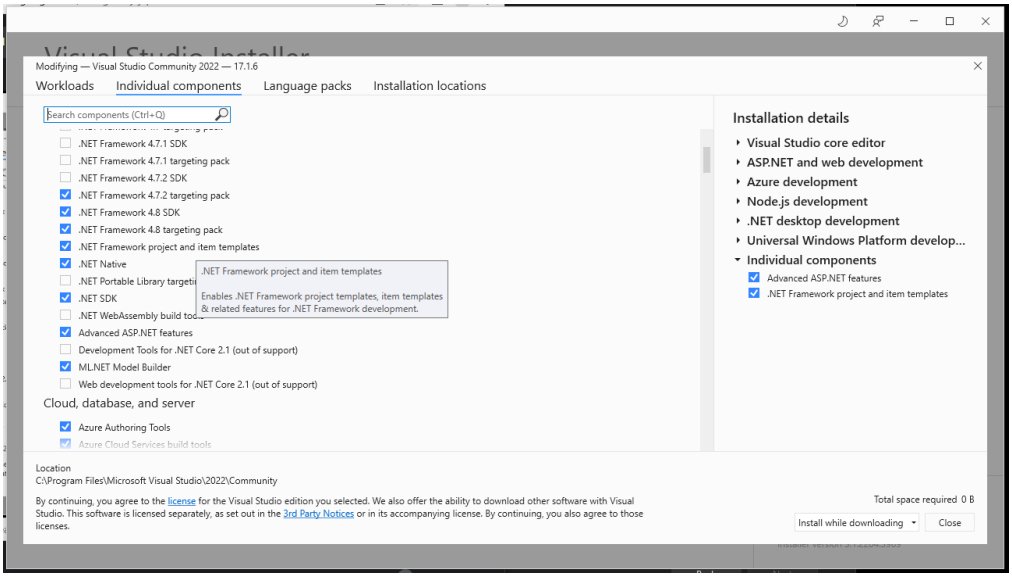
```
            }
            fs.Close();
            Console.WriteLine("DeSerialization completed!!!");
        }
    }

    [Serializable]
    public class Student
    {
        public int Id { get; set; }
        public string Name { get; set; }
    public string email { get; set; }
    }
  }
```

```
D:\visual studio\ConsoleApp27_serializationdeserialization\ConsoleApp27_serializationdeserialization\bin\Debug\ConsoleApp27_serializationdeserializat...
Serialization completed!!!
Id:101 Name:Saumik Dutta Email:saumik@gmail.com
Id:102 Name:Akshay Shetty Email:akshetty@gmail.com
Id:103 Name:sanjay dewan Email:dewansanjay@gmail.com
DeSerialization completed!!!
```

08May2022

Necessary updates for ADO.Net

Add .net project items and templates from visual studio installer and download and install sql express edition 2019


Modifying — Visual Studio Community 2022 — 17.1.6
Workloads
Individual components
Language packs
Installation locations
Search components (Ctrl+Q)
.NET Framework 4.7.1 SDK
.NET Framework 4.7.1 targeting pack
.NET Framework 4.7.2 SDK
.NET Framework 4.7.2 targeting pack
.NET Framework 4.8 SDK
.NET Framework 4.8 targeting pack
.NET Framework project and item templates
.NET Native
.NET Portable Library targeti
.NET SDK
.NET WebAssembly build too
Advanced ASP.NET features
Development Tools for .NET Core 2.1 (out of support)
ML.NET Model Builder
Web development tools for .NET Core 2.1 (out of support)
Cloud, database, and server
Azure Authoring Tools
Azure Cloud Services build tools
.NET Framework project and item templates
Enables .NET Framework project templates, item templates & related features for .NET Framework development.
Installation details
Visual Studio core editor
ASP.NET and web development
Azure development
Node.js development
.NET desktop development
Universal Windows Platform develop...
Individual components
Advanced ASP.NET features
.NET Framework project and item templates
Location
C:\Program Files\Microsoft Visual Studio\2022\Community
By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.
Total space required 0 B
Install while downloading
Close

09may2022

Practical 30:

Create & Connect database in c# with SQL express server Database

Database connectivity in C# console application

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;
namespace ConsoleApp_dbconnectivity
{
    class Program
    {
        static void Main(string[] args)
        {
string connect = @"Data Source=.\sqlexp03;Initial Catalog=mca;Integrated
Security=True;Pooling=False";
            SqlConnection conn = new SqlConnection(connect);
            conn.Open();
 SqlCommand sqlCommand = new SqlCommand("select * from emp", conn);
SqlDataReader sqlDataReader = sqlCommand.ExecuteReader();
while (sqlDataReader.Read())
            {
Console.WriteLine("ID : {0}, Name :{1}, Gender: {2}, Email: {3}, city: {4}",
sqlDataReader.GetInt32(0), sqlDataReader.GetString(1),
sqlDataReader.GetString(2),
sqlDataReader.GetString(3), sqlDataReader.GetString(4));
            }
            sqlDataReader.Close();
            conn.Close();
            Console.ReadKey();
        }
    }
```

}

```
D:\visual studio\vs_savefile\ConsoleApp_dbconnectivity\ConsoleApp_dbconnectivity\bin\Debug\Consol...
ID : 1, Name :saumik, Gender: male, Email: saumik@gmail.com, city: vashi
ID : 2, Name :karishma, Gender: female, Email: kt@gmail.com, city: tilaknagar
ID : 3, Name :sanjay dewan, Gender: male, Email: dewansanjay@gmail.com, city: kanjun marg
ID : 4, Name :vidushi, Gender: female, Email: vidushiv@gmail.com, city: andheri
ID : 5, Name :jayraj, Gender: male, Email: jayrajg@gmail.com, city: borivali
```

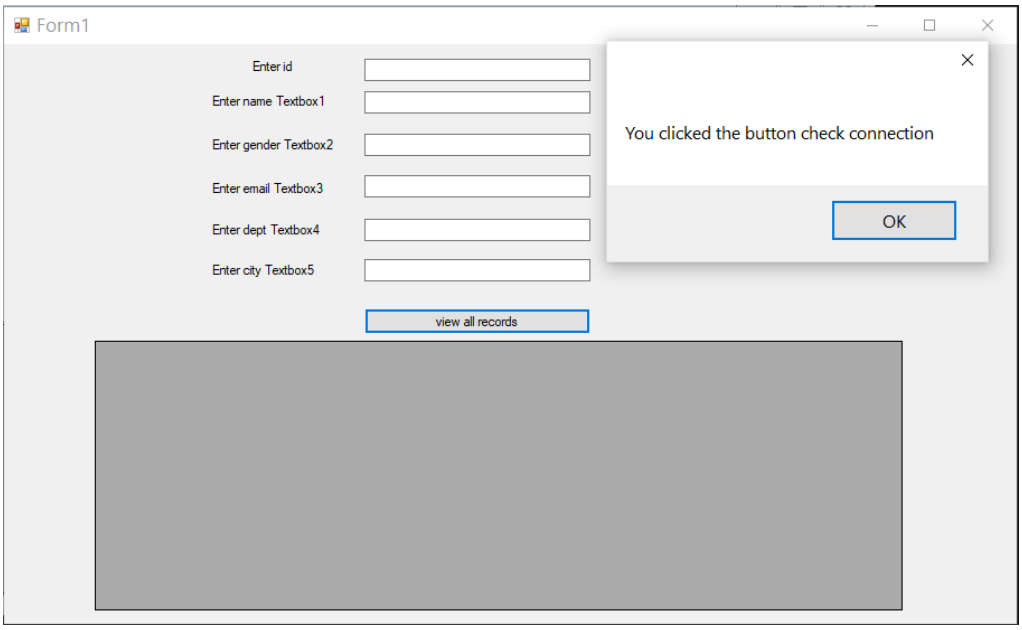
Practical 31:

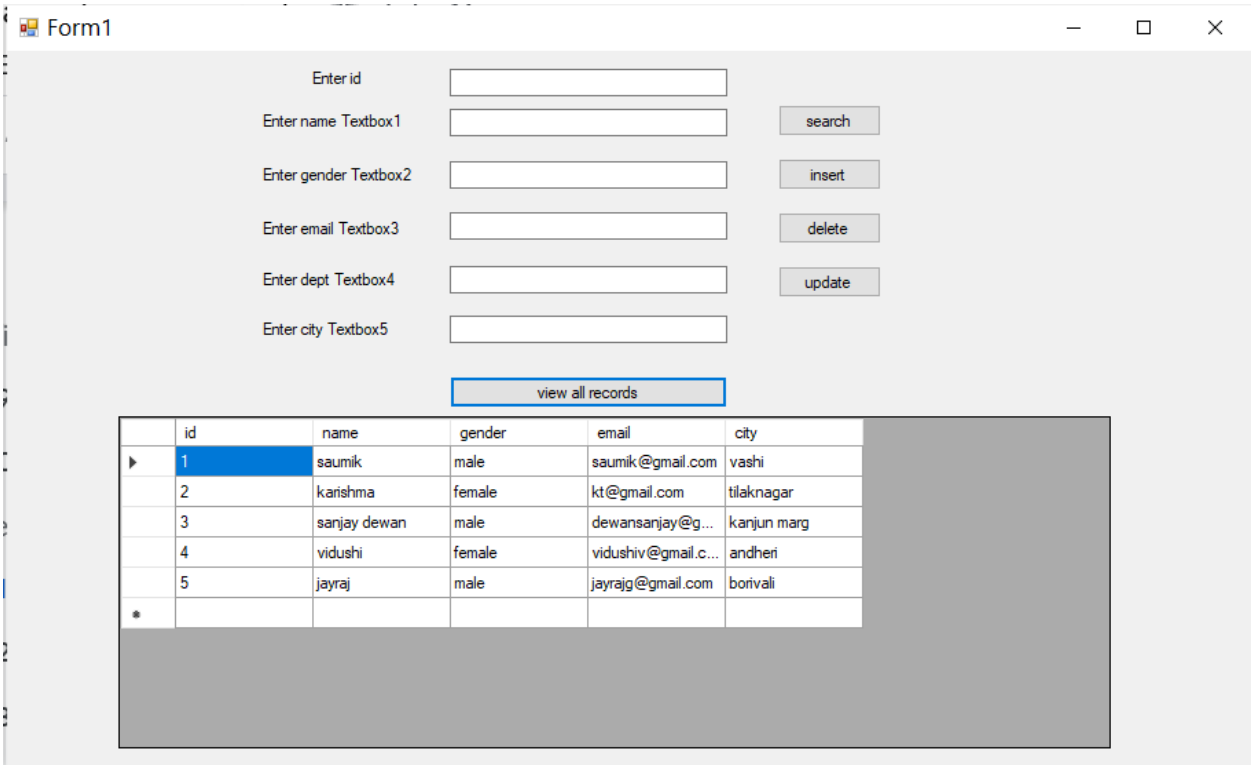
Windows form application

Code:-

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
namespace WindowsFormsApp1
{
    public partial class Form1 : Form
    {
        public static string datasource = @"Data Source=.\sqlexp03;Initial
Catalog=mca;Integrated Security=True";
        public static SqlConnection sqlconnection = new SqlConnection(datasource);


        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            MessageBox.Show("You clicked the button check connection");
            using(SqlConnection sqlconnection=new SqlConnection(datasource))
            {
                sqlconnection.Open();
```

```
            //cmd.ExecuteReader();
           // SqlCommand cmd = new SqlCommand("select * from
employee",sqlconnection);
            SqlCommand cmd = new SqlCommand("select * from
emp",sqlconnection);
            SqlDataReader sqldatareader = cmd.ExecuteReader();
            DataTable dt = new DataTable();
            dt.Load(sqldatareader);
            dataGridView1.DataSource = dt;



        }
        sqlconnection.Close();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
    }



  }
}
```


Form1
Enter id
Enter name Textbox1
Enter gender Textbox2
Enter email Textbox3
Enter dept Textbox4
Enter city Textbox5
view all records
You clicked the button check connection
OK


Form1
Enter id
Enter name Textbox1
Enter gender Textbox2
Enter email Textbox3
Enter dept Textbox4
Enter city Textbox5
search
insert
delete
update
view all records
id name gender email city
1 saumik male saumik@gmail.com vashi
2 karishma female kt@gmail.com tilaknagar
3 sanjay dewan male dewansanjay@g... kanjun marg
4 vidushi female vidushiv@gmail.c... andheri
5 jayraj male jayrajg@gmail.com borivali

Practical 32:

Crud operation with window form application

Code:-

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
namespace WindowsFormsApp1
{
    public partial class Form1 : Form
    {
        public static string datasoruce = @"Data Source=.\sqlexpress03;Initial
Catalog=students;Integrated Security=True";
        public static SqlConnection sqlconnection = new SqlConnection(datasoruce);


        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            MessageBox.Show("You clicked the button check connection");
            using(SqlConnection sqlconnection=new SqlConnection(datasoruce))
            {
                sqlconnection.Open();
```

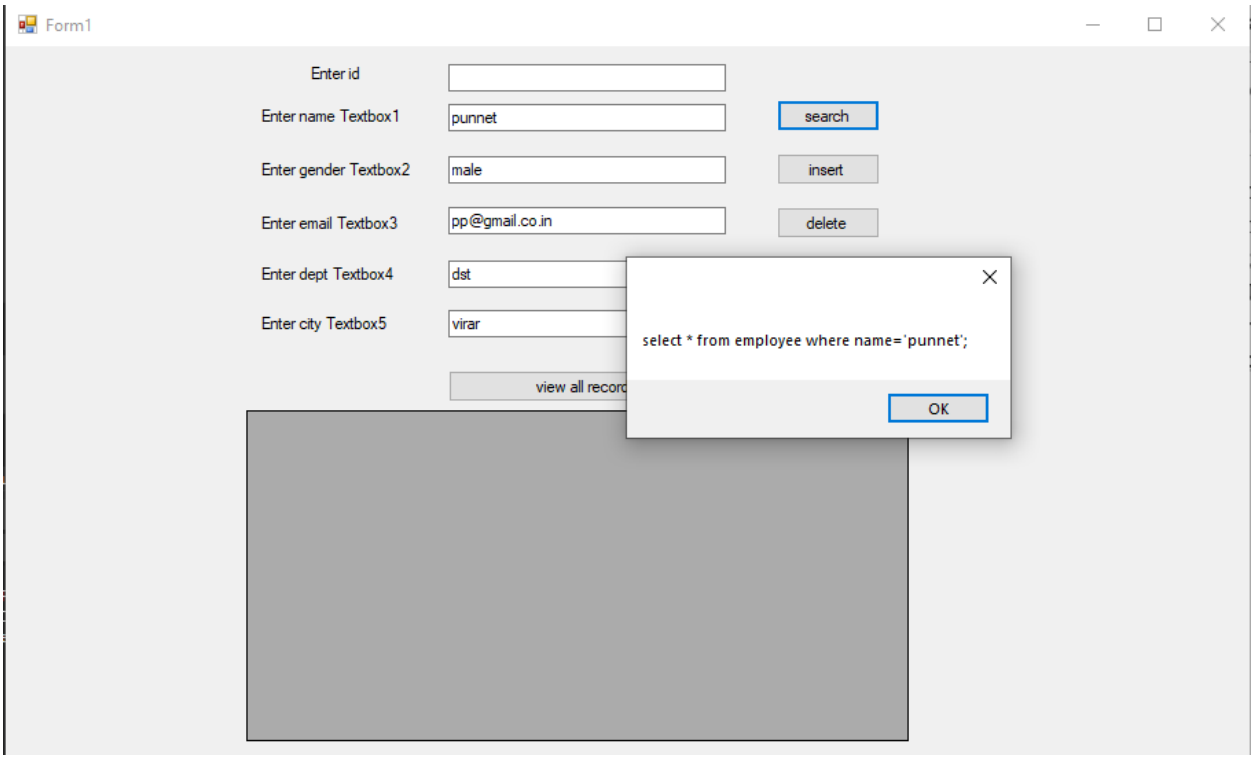
```
            //cmd.ExecuteReader();
            SqlCommand cmd = new SqlCommand("select * from
employee",sqlconnection);
            SqlDataReader sqldatareader = cmd.ExecuteReader();
            DataTable dt = new DataTable();
            dt.Load(sqldatareader);
            dataGridView1.DataSource = dt;



        }
        sqlconnection.Close();
    }

    private void Form1_Load(object sender, EventArgs e)
    {

    }

    private void button3_Click(object sender, EventArgs e)
    {
        //for insert

        using (SqlConnection sqlconnection = new SqlConnection(datasoruce))
        {
            string insquery = "insert into employee" +
            "(name,gender,email,dept,city)" +
            "values('" + textBox1.Text + "','" + textBox2.Text + "','" + textBox3.Text
+ "'" +
            "','" + textBox4.Text + "','" + textBox5.Text + "')";
            MessageBox.Show(insquery);
            SqlCommand cmd = new SqlCommand(insquery, sqlconnection);
            sqlconnection.Open();
```

```
            MessageBox.Show("record inserted");
            cmd.ExecuteNonQuery();


        }
        sqlconnection.Close();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        using (SqlConnection sqlconnection = new SqlConnection(datasoruce))
        {
            sqlconnection.Open();
            string search = "select * from employee where
name='"+textBox1.Text+"';";
           // string search = "select * from employee where
name='"+textBoxid.Text+"';";
            MessageBox.Show(search);
            //cmd.ExecuteReader();
            SqlCommand cmd = new SqlCommand(search, sqlconnection);
            SqlDataReader sqldatareader = cmd.ExecuteReader();
            while(sqldatareader.Read())
            {
                textBox1.Text = sqldatareader["name"].ToString();
                textBox2.Text = sqldatareader["gender"].ToString();
                textBox3.Text = sqldatareader["email"].ToString();
                textBox4.Text = sqldatareader["dept"].ToString();
                textBox5.Text = sqldatareader["city"].ToString();
            }
        }
        sqlconnection.Close();
    }

    private void button4_Click(object sender, EventArgs e)
    {
```

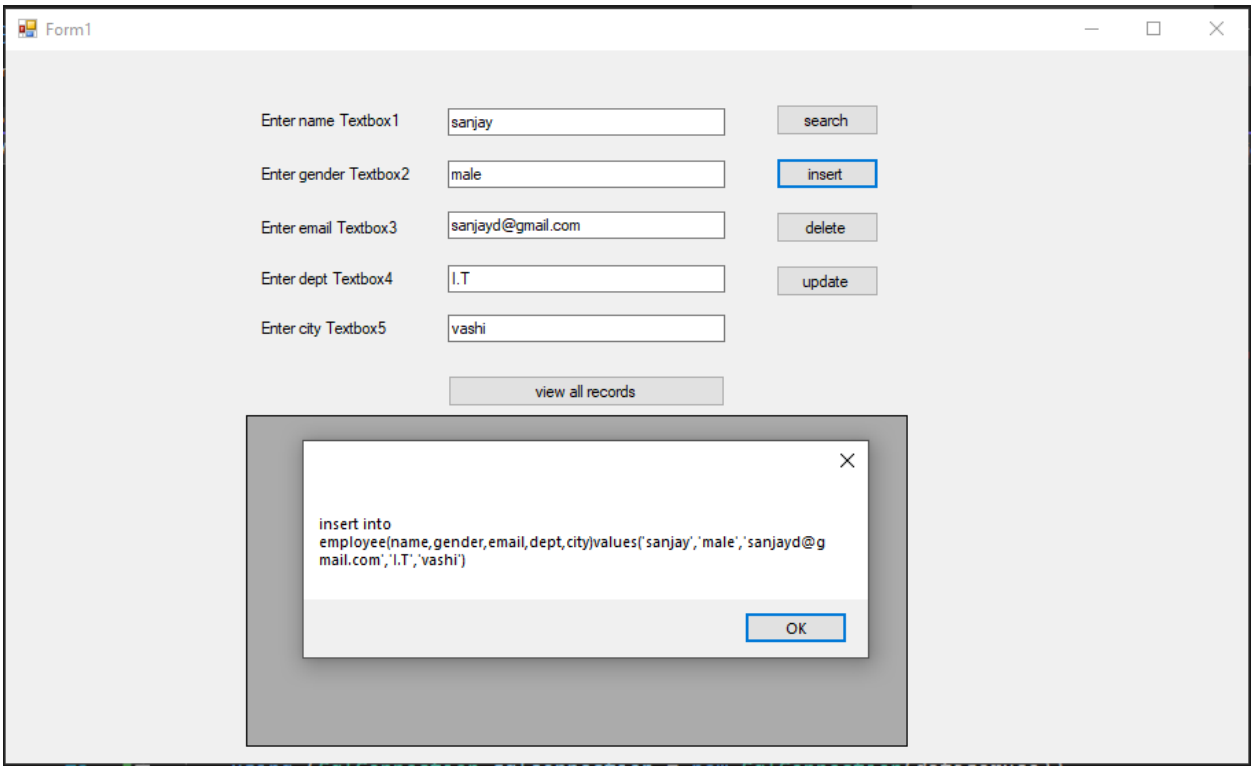
```
            using (SqlConnection sqlconnection = new SqlConnection(datasoruce))
            {
                if (textBox1.Text != "")
                {
                    string delquery = "delete from employee where name='" +
textBox1.Text + "'";
                    //string delquery = "delete from employee where name='" +
textBoxid.Text + "'";
                    MessageBox.Show(delquery);
                    SqlCommand cmd = new SqlCommand(delquery, sqlconnection);
                    sqlconnection.Open();

                    //cmd.ExecuteNonQuery();
                    int i = cmd.ExecuteNonQuery();
                    string res = i + "records affected";
                    MessageBox.Show(res);
                }

                else
                {
                    MessageBox.Show("please enter name");
                }

            }
            sqlconnection.Close();
        }

        private void button5_Click(object sender, EventArgs e)
        {
            if ((textBox1.Text == "") && (textBox2.Text == "")
               && (textBox3.Text == "") && (textBox4.Text == "")
               && (textBox5.Text == "")/*&&(textBoxid.Text=="")*/)
            {
                MessageBox.Show("please fill all fields");
            }
```

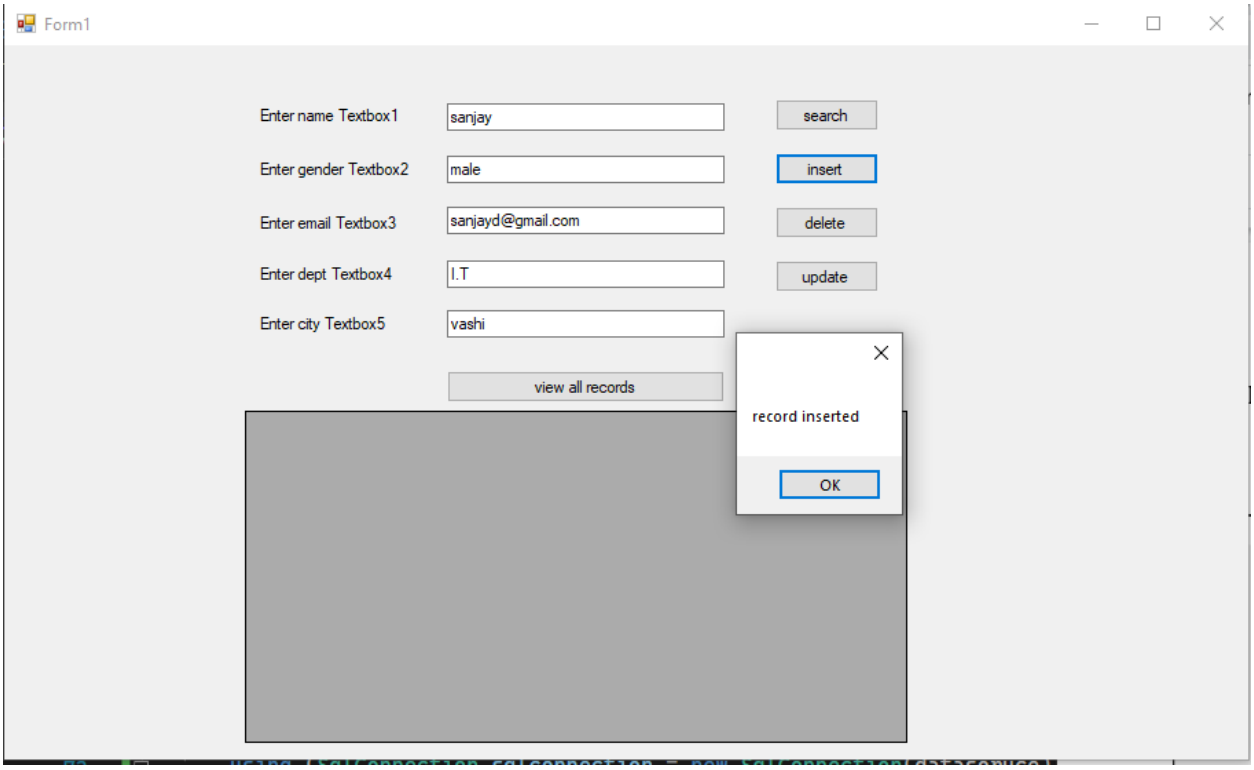
```
            else
            {
                //string updquery="update employee set
gender='"+textBox2.Text+"',email='" +textBox3.Text+ "',dept='" +textBox4.Text+
"',city='"+textBox5.Text+"' where name='"+textBox1.Text+"';";
                string updquery = "update employee set gender='" + textBox2.Text +
"',email='" + textBox3.Text + "',dept='" + textBox4.Text + "',city='" +
textBox5.Text + "' where id=" + textBoxid.Text + ";";
                MessageBox.Show(updquery);
                SqlCommand cmd = new SqlCommand(updquery, sqlconnection);
                sqlconnection.Open();
                int res=cmd.ExecuteNonQuery();
                string r = res + " record updated";
                MessageBox.Show(r);

            }
            sqlconnection.Close();


        }
    }
}
```

Search

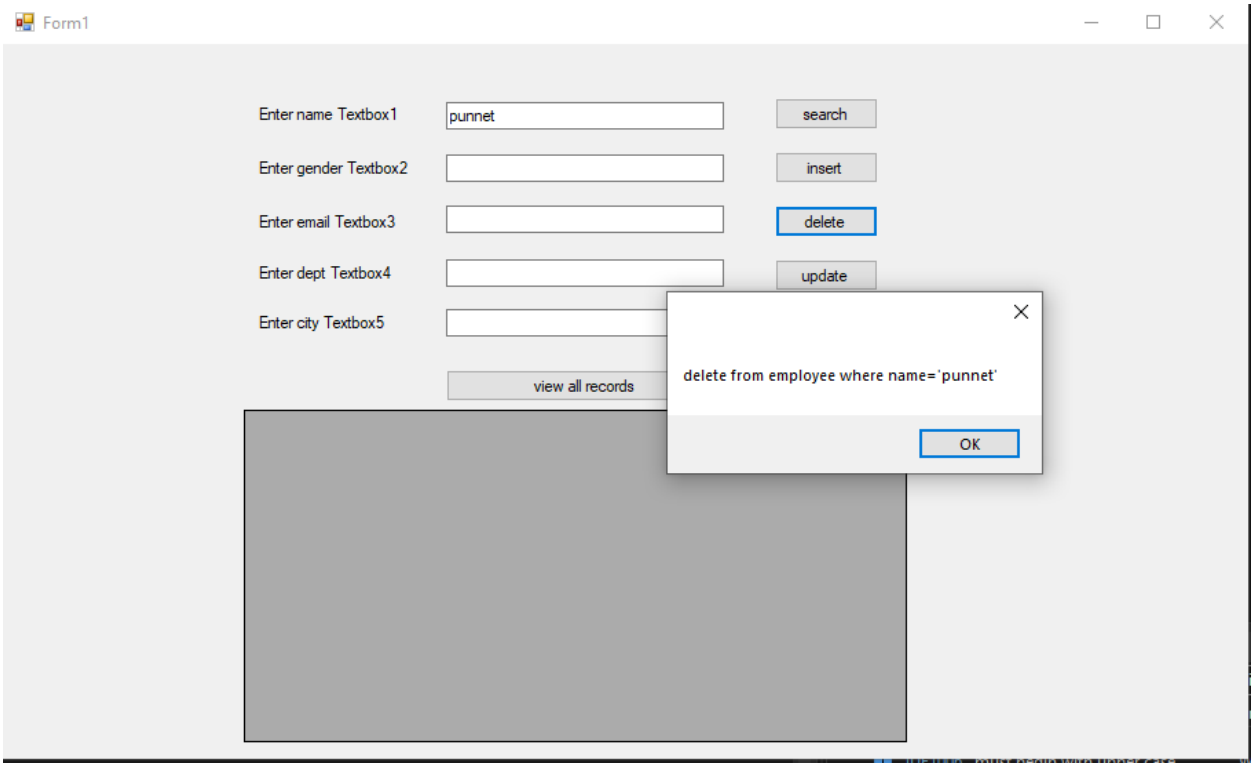

Form1
Enter id
Enter name Textbox1
punnet
search
Enter gender Textbox2
male
insert
Enter email Textbox3
pp@gmail.co.in
delete
Enter dept Textbox4
dst
Enter city Textbox5
virar
view all record
select * from employee where name='punnet';
OK

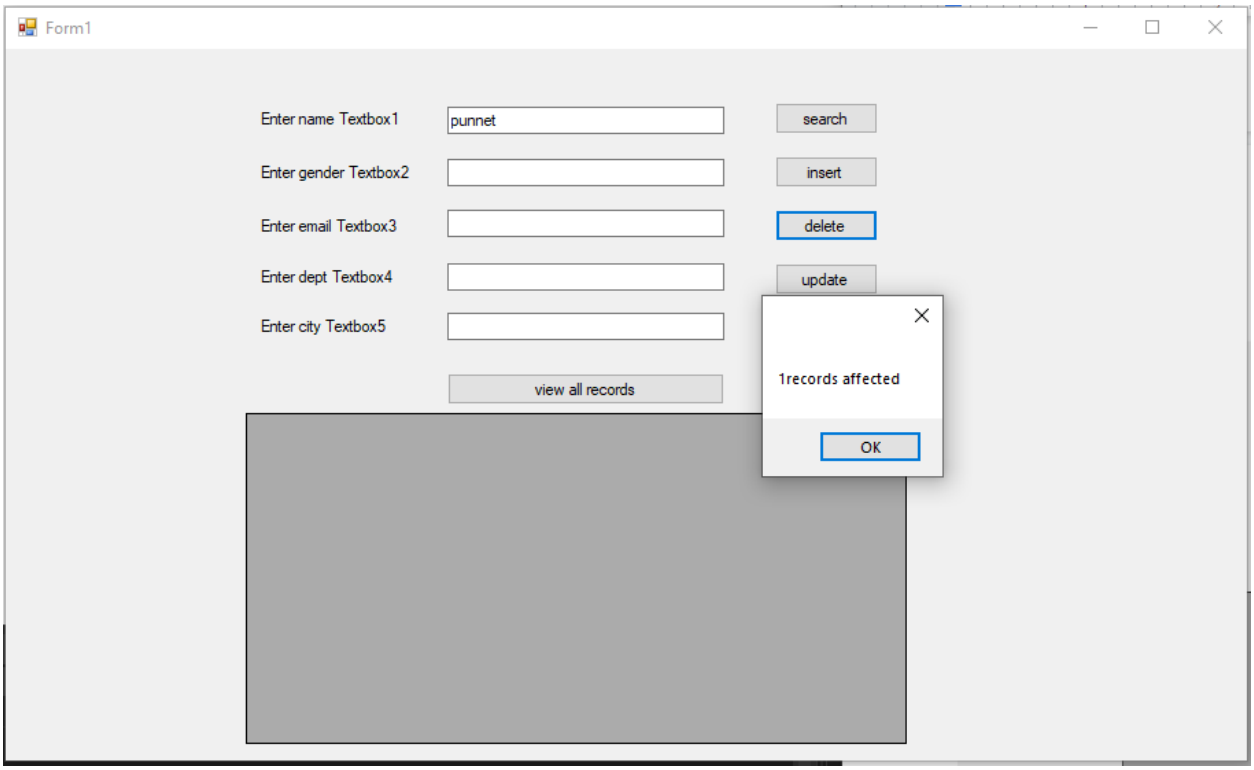
Insert



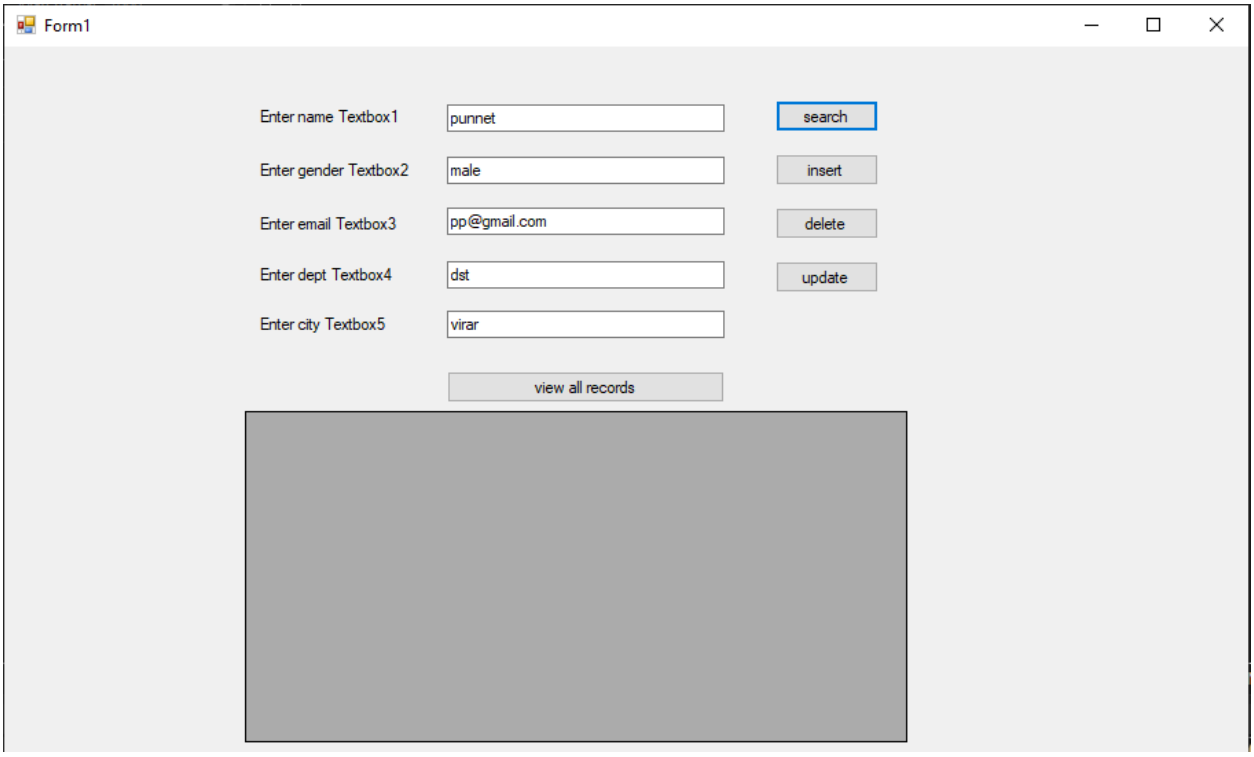


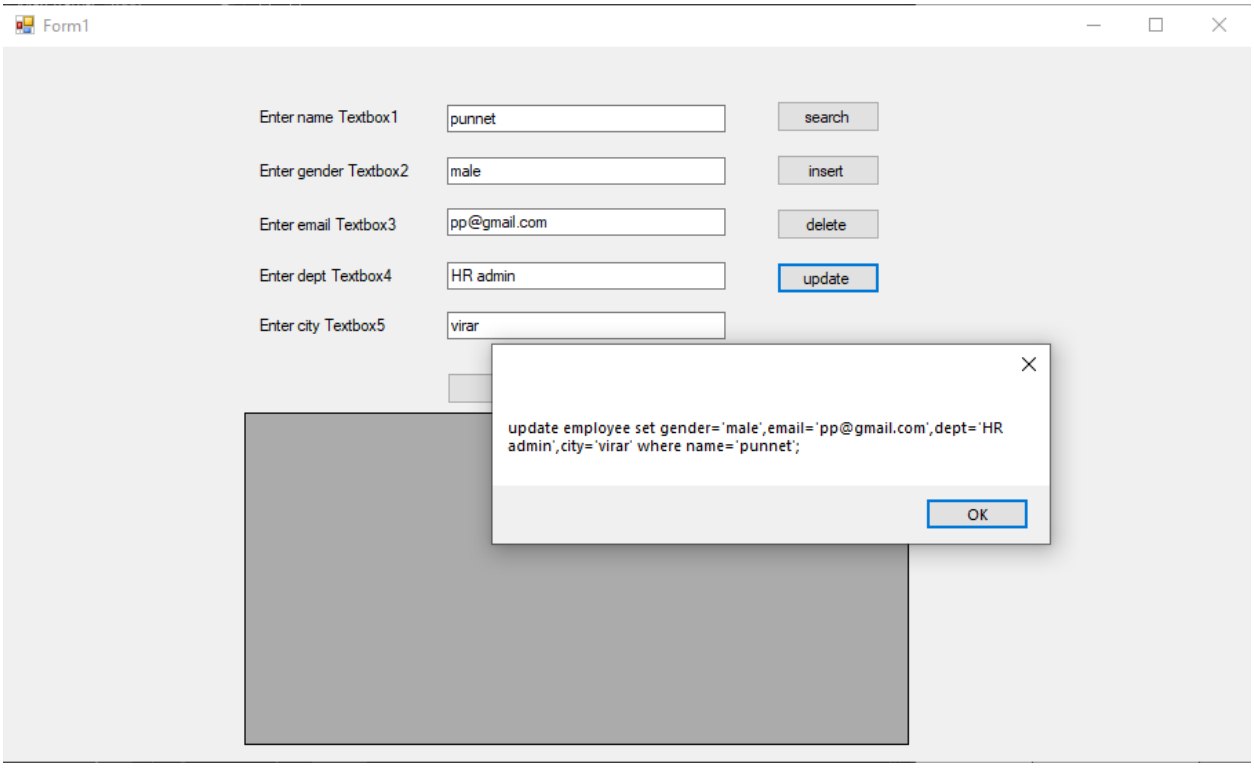
Delete

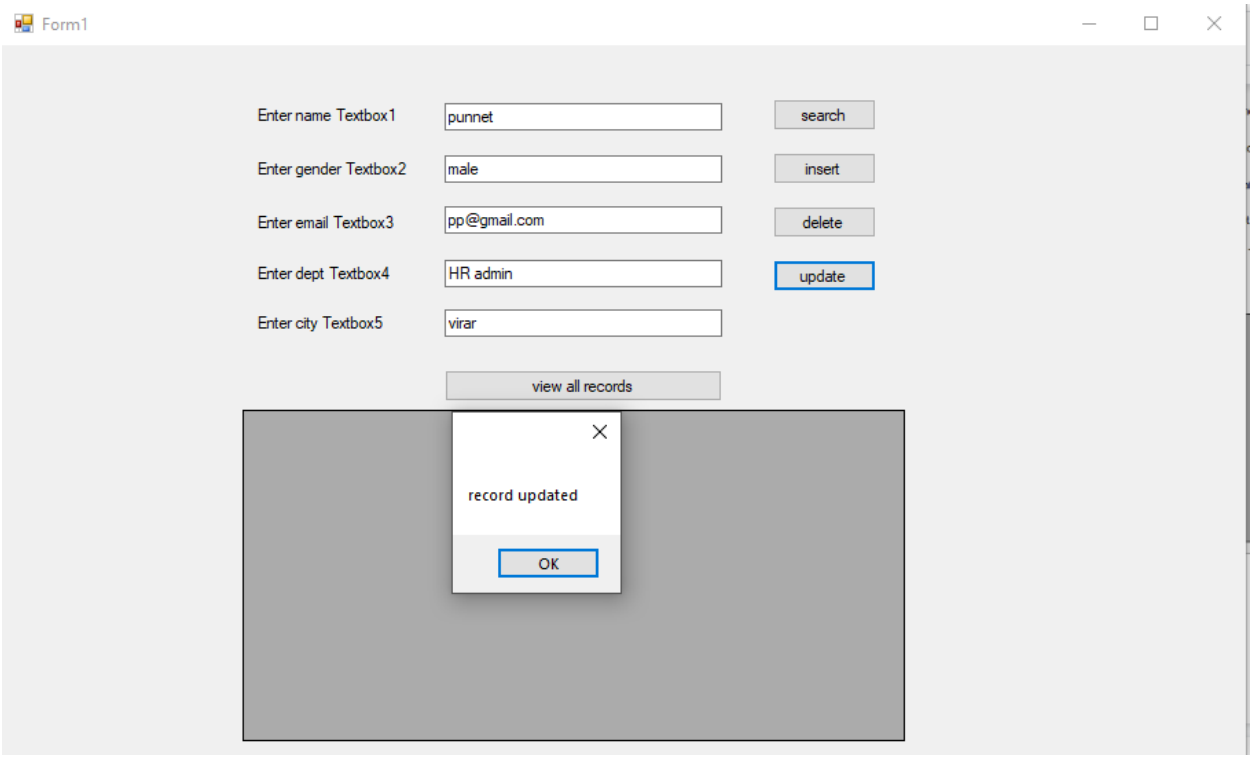

Form1
Enter name Textbox1
punnet
search
Enter gender Textbox2
insert
Enter email Textbox3
delete
Enter dept Textbox4
update
Enter city Textbox5
view all records
delete from employee where name='punnet'
OK


Form1
Enter name Textbox1
punnet
search
Enter gender Textbox2
insert
Enter email Textbox3
delete
Enter dept Textbox4
update
Enter city Textbox5
view all records
1records affected
OK

Update


Form1
Enter name Textbox1
punnet
search
Enter gender Textbox2
male
insert
Enter email Textbox3
pp@gmail.com
delete
Enter dept Textbox4
dst
update
Enter city Textbox5
virar
view all records

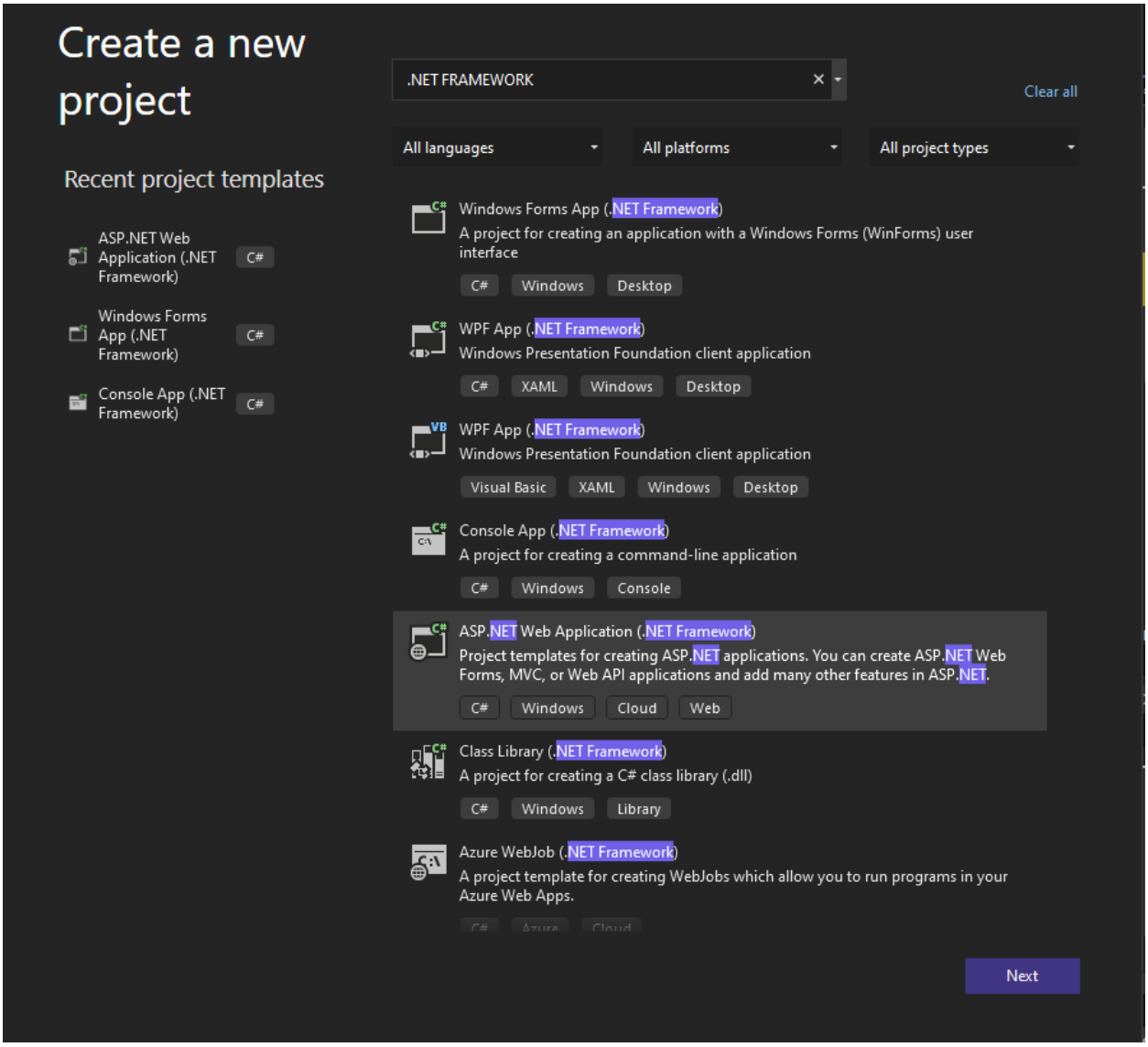

Form1
Enter name Textbox1
punnet
search
Enter gender Textbox2
male
insert
Enter email Textbox3
pp@gmail.com
delete
Enter dept Textbox4
HR admin
update
Enter city Textbox5
virar
update employee set gender='male',email='pp@gmail.com',dept='HR admin',city='virar' where name='punnet';
OK



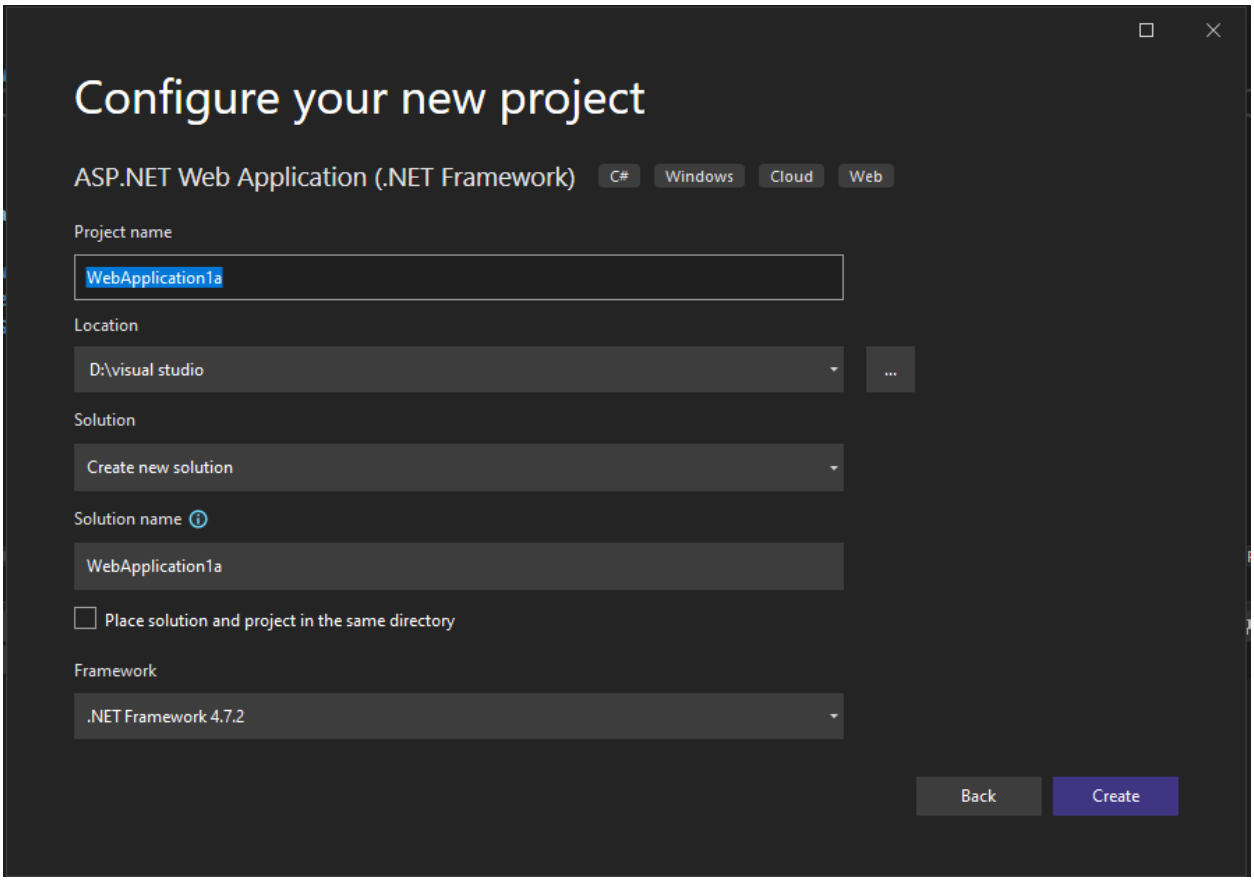
Practical 33:

SQLconnectivity with Webapplication and SQL Injection Webapplication

New project->select ASP.Net Web Application (.Net Framework)



Assign name->select empty and uncheck configure for https


Configure your new project
ASP.NET Web Application (.NET Framework)
C#
Windows
Cloud
Web
Project name
WebApplication1a
Location
D:\visual studio
...
Solution
Create new solution
Solution name
WebApplication1a
Place solution and project in the same directory
Framework
.NET Framework 4.7.2
Back
Create

Create a new ASP.NET Web Application

Empty

An empty project template for creating ASP.NET applications. This template does not have any content in it.

Web Forms

A project template for creating ASP.NET Web Forms applications. ASP.NET Web Forms lets you build dynamic websites using a familiar drag-and-drop, event-driven model. A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.

MVC

A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.

Web API

A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.

Single Page Application

A project template for creating rich client side JavaScript driven HTML5 applications using ASP.NET Web API. Single Page Applications provide a rich user experience which includes client-side interactions using HTML5, CSS3, and JavaScript.

Authentication

None

Add folders & core references

- [ ] Web Forms
- [ ] MVC
- [ ] Web API

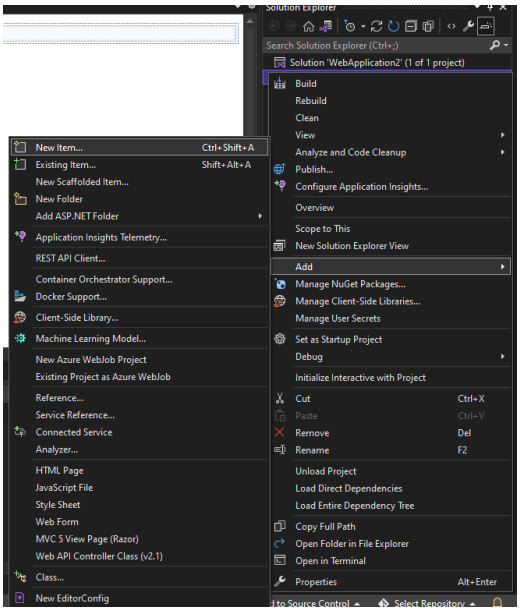
Advanced

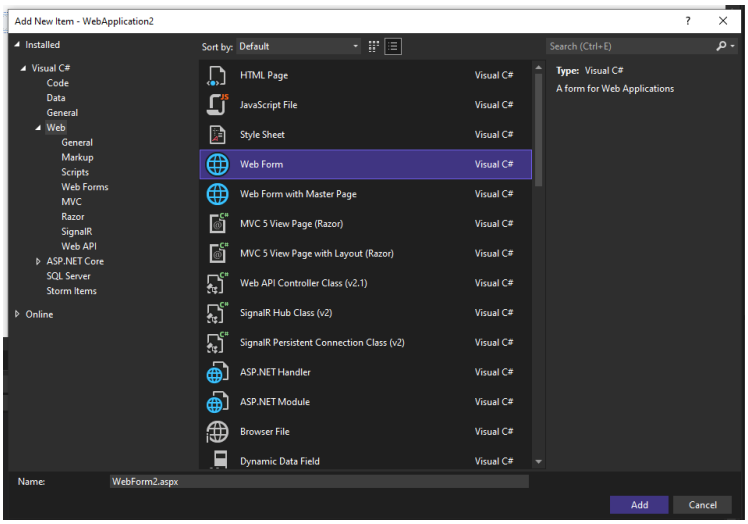
- [ ] Configure for HTTPS
- [ ] Docker support
  (Requires Docker Desktop)
- [ ] Also create a project for unit tests

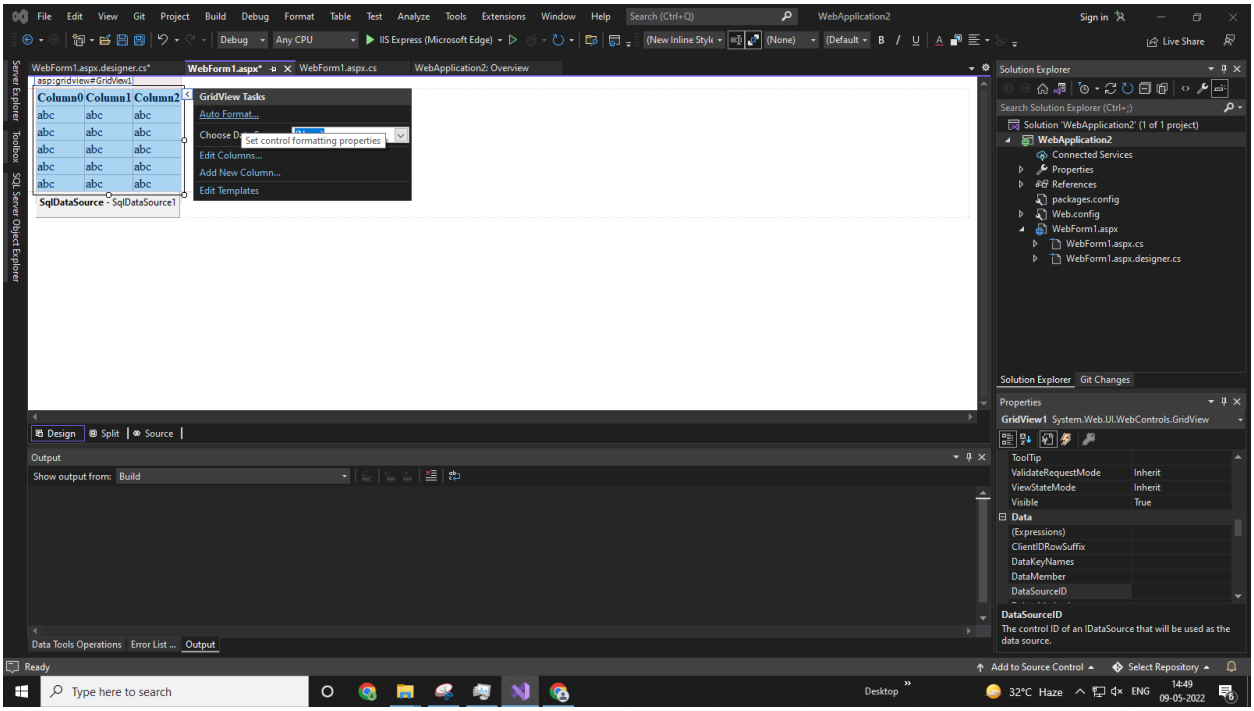
WebApplication1a.Tests

Back | Create

Right click on web app in soln explorer -> add-> new item-> webform







Showing data using from database

Webform1.apsx

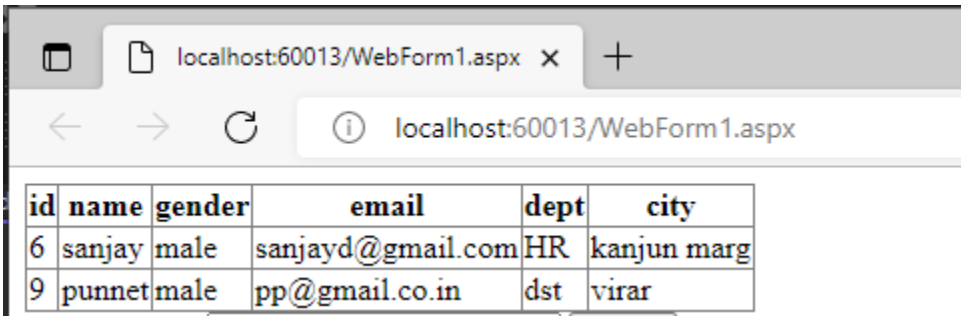
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs" Inherits="WebApplication1a.WebForm1" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
      <div>
      </div>
      <asp:GridView ID="GridView1" runat="server" >
      </asp:GridView>
      <asp:Label ID="Label1" runat="server" Text="Enter Name:-"></asp:Label>
      <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
```

```
        <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click" />
      <%--  <asp:GridView ID="GridView2" runat="server" >
        </asp:GridView>--%>
    </form>
</body>
</html>
```

webform1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
namespace WebApplication1a
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            using(SqlConnection sqlconnection=new SqlConnection(@"Data
Source=.\sqlexpress03;Initial Catalog=students;Integrated Security=True"))
            {
                sqlconnection.Open();
                SqlCommand cmd = new SqlCommand("Select * from
employee",sqlconnection);

                GridView1.DataSource = cmd.ExecuteReader();
                GridView1.DataBind();
                sqlconnection.Close();
            }
        }
```

```
        protected void Button1_Click(object sender, EventArgs e)
        {
            //using (SqlConnection sqlconnection = new SqlConnection(@"Data
Source=.\sqlexpress03;Initial Catalog=students;Integrated Security=True"))
            //{
            //    sqlconnection.Open();
            //    string search= "Select * from employee where
name='%"+TextBox1.Text+"%'";
            //    Response.Write(search);
            //    SqlCommand cmd = new SqlCommand(search, sqlconnection);

            //    GridView1.DataSource = cmd.ExecuteReader();
            //    GridView1.DataBind();
            //    sqlconnection.Close();

            //    //  Response.Write("hello");
            //}
        }
    }
}
```



id	name	gender	email	dept	city
6	sanjay	male	sanjayd@gmail.com	HR	kanjun marg
9	punnet	male	pp@gmail.co.in	dst	virar

Sql injection

Webform1.aspx

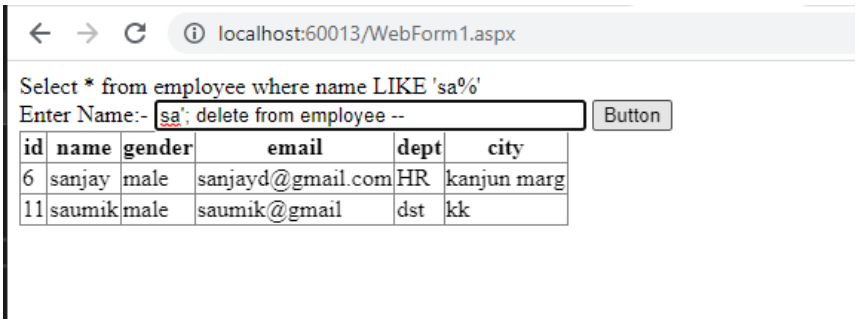
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs" Inherits="WebApplication1a.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
      <div>
      </div>
      <asp:Label ID="Label1" runat="server" Text="Enter Name:-"></asp:Label>
      <asp:TextBox ID="TextBox1" runat="server"
Width="291px"></asp:TextBox>
      <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click" />
      <asp:GridView ID="GridView1" runat="server" >
      </asp:GridView>


     <%--  <asp:GridView ID="GridView2" runat="server" >
      </asp:GridView>--%>

   </form>


</body>
</html>
```

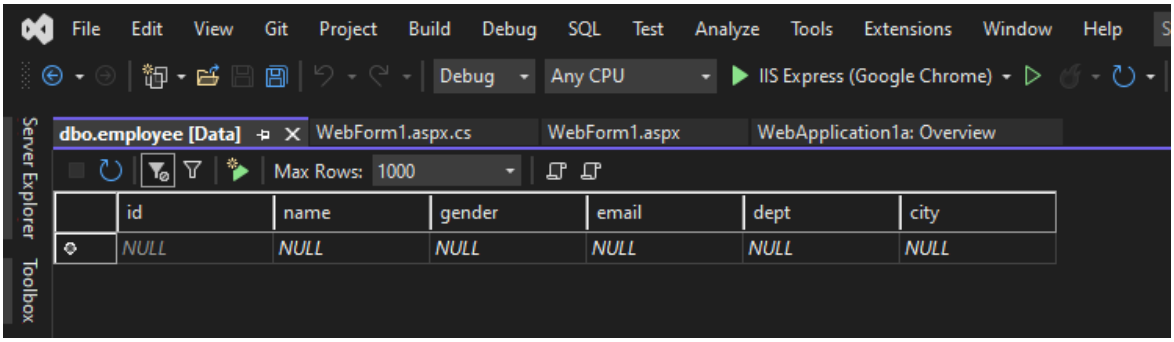
Webform1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
namespace WebApplication1a
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            using(SqlConnection sqlconnection=new SqlConnection(@"Data
Source=.\sqlexpress03;Initial Catalog=students;Integrated Security=True"))
            {
                //sqlconnection.Open();
                //SqlCommand cmd = new SqlCommand("Select * from
employee",sqlconnection);

                //GridView1.DataSource = cmd.ExecuteReader();
                //GridView1.DataBind();
                //sqlconnection.Close();

              //  Response.Write("hello");
            }
        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            using (SqlConnection sqlconnection = new SqlConnection(@"Data
Source=.\sqlexpress03;Initial Catalog=students;Integrated Security=True"))
            {
                sqlconnection.Open();
```

```
            string search = "Select * from employee where name LIKE '" +
TextBox1.Text + "%'";
            Response.Write(search);
            SqlCommand cmd = new SqlCommand(search, sqlconnection);

            GridView1.DataSource = cmd.ExecuteReader();
            GridView1.DataBind();
            sqlconnection.Close();

            //  Response.Write("hello");
        }
    }
  }
}
```

Commmand:-
sa'; delete from employee –
Text'; delete from <table_name>

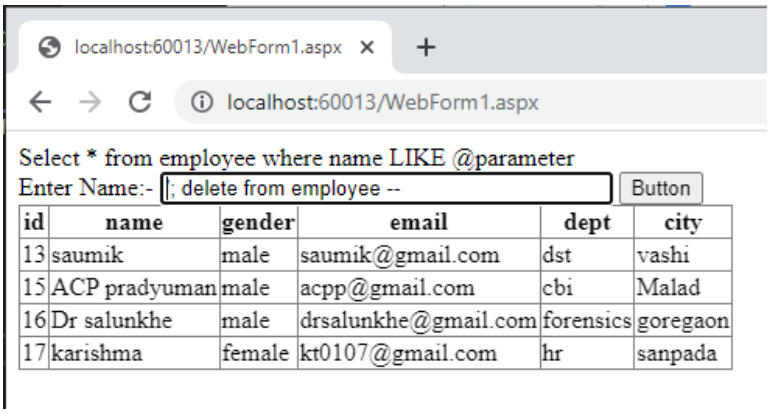




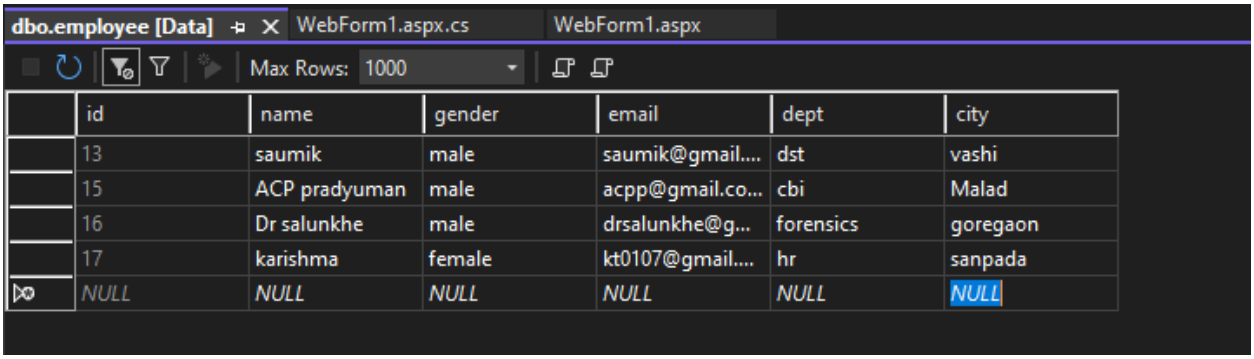
Avoiding sql injection
Webform1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
namespace WebApplication1a
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {  }
        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            using (SqlConnection sqlconnection = new SqlConnection(@"Data
Source=.\sqlexpress03;Initial Catalog=students;Integrated Security=True"))
            {
                //to avoid sql injection attack there are two approaches
                //1. using paramtereized query
                sqlconnection.Open();
                //string search = "Select * from employee where name LIKE '" +
TextBox1.Text + "%'";
                string search = "Select * from employee where name LIKE
@parameter";
                Response.Write(search);
                SqlCommand cmd = new SqlCommand(search, sqlconnection);

                cmd.Parameters.AddWithValue("@parameter", TextBox1.Text + "%");
                GridView1.DataSource = cmd.ExecuteReader();
```

```
            GridView1.DataBind();
            sqlconnection.Close();

            //  Response.Write("hello");
        }
    }
  }
}
```



id	name	gender	email	dept	city
13	saumik	male	saumik@gmail.com	dst	vashi
15	ACP pradyuman	male	acpp@gmail.com	cbi	Malad
16	Dr salunkhe	male	drsalunkhe@gmail.com	forensics	goregaon
17	karishma	female	kt0107@gmail.com	hr	sanpada



id	name	gender	email	dept	city
13	saumik	male	saumik@gmail....	dst	vashi
15	ACP pradyuman	male	acpp@gmail.co...	cbi	Malad
16	Dr salunkhe	male	drsalunkhe@g...	forensics	goregaon
17	karishma	female	kt0107@gmail....	hr	sanpada
NULL	NULL	NULL	NULL	NULL	NULL

10May2022

Practical 34:

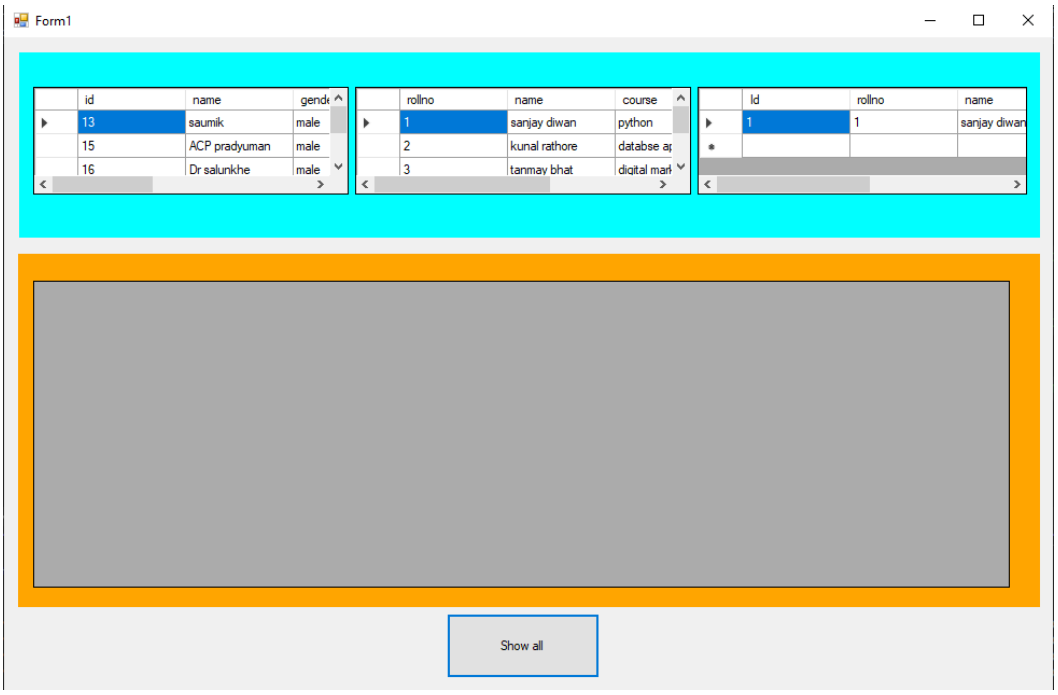
Demonstration of Dataset and DataTable
Difference between dataset datatable and DataRelationship

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
namespace WindowsFormsApp2_dataset
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            using(SqlConnection sqlconnection=new SqlConnection(@"Data
Source=.\sqlexpress03;Initial Catalog=students;Integrated Security=True"))
            {
                sqlconnection.Open();
                DataSet dataset = new DataSet();
                DataSet dataset1 = new DataSet();
                DataTable datatable = new DataTable();
```

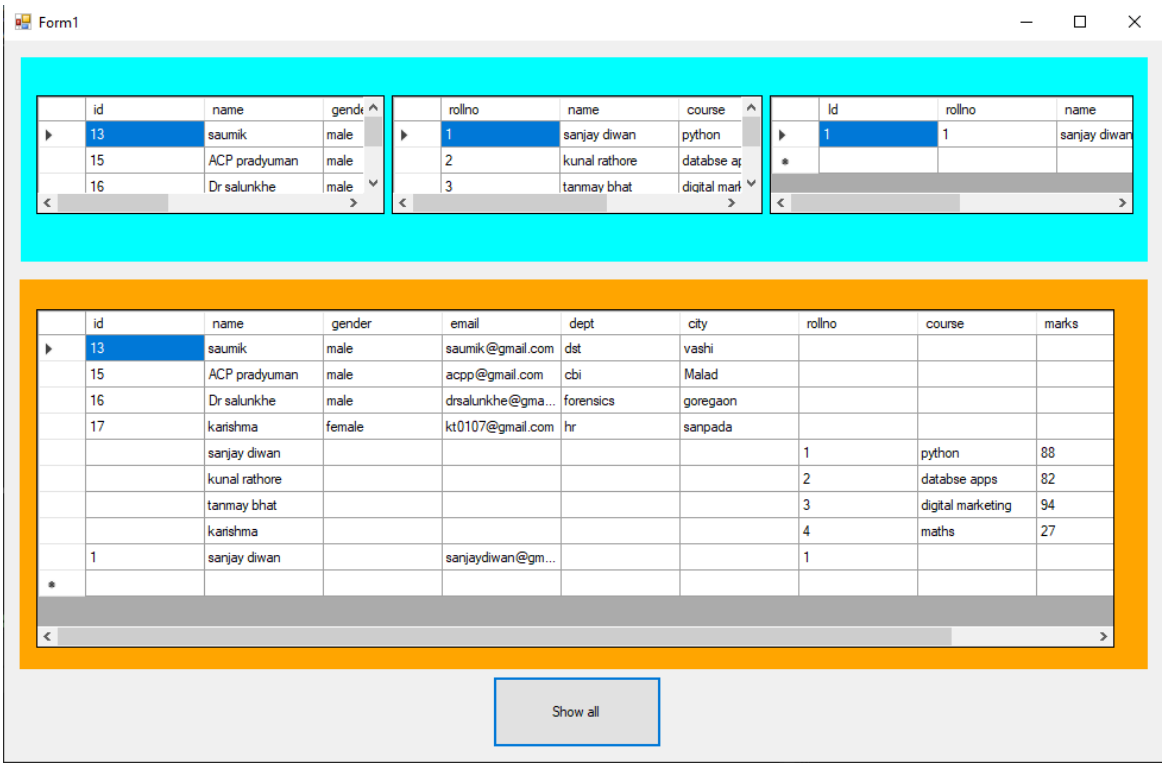
```
SqlDataAdapter adapter = new SqlDataAdapter("select * from
employee",sqlconnection);
adapter.Fill(dataset,"employee"); //Tables[0]

SqlDataAdapter adapter1 = new SqlDataAdapter("select * from
student1", sqlconnection);
adapter1.Fill(dataset, "student1");//Tables[1]  also reffered as
dataset.tables[1]
adapter1.Fill(dataset1, "student1");//Tables[1]  also reffered as
dataset.tables[1]

SqlDataAdapter adapter2 = new SqlDataAdapter("select * from
stud_details", sqlconnection);
adapter2.Fill(dataset, "stud_details");
adapter2.Fill(dataset1, "stud_details");

dataGridView1.DataSource = dataset.Tables["employee"];
dataGridView2.DataSource = dataset.Tables[1];
//dataGridView3.DataSource = dataset.Tables["stud_details"];
dataGridView3.DataSource = dataset.Tables[2];


//filing datatable with sqldataadapter
SqlDataAdapter adapter3 = new SqlDataAdapter("select * from
employee", sqlconnection);
adapter3.Fill(datatable);

SqlDataAdapter adapter4 = new SqlDataAdapter("select * from
student1", sqlconnection);
adapter4.Fill(datatable);

SqlDataAdapter adapter5 = new SqlDataAdapter("select * from
stud_details", sqlconnection);
adapter5.Fill(datatable);
```

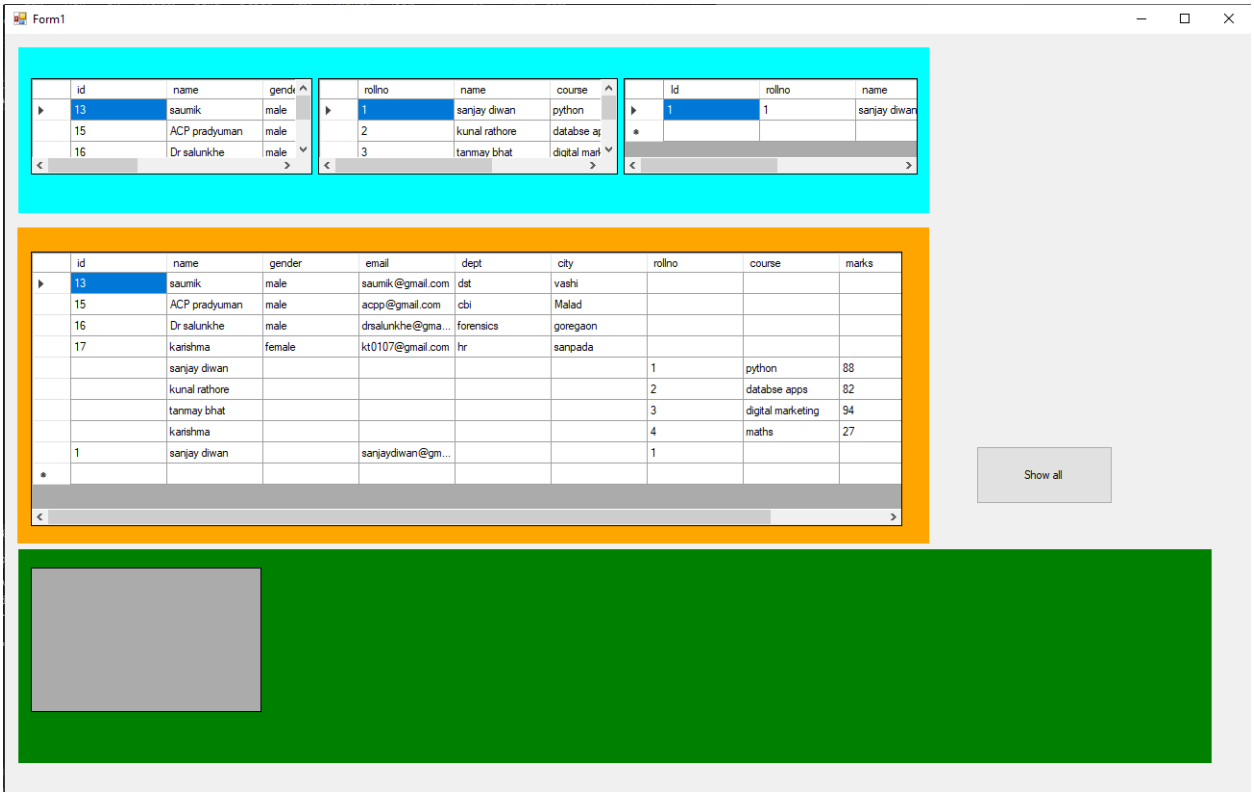
```
            dataGridView4.DataSource = datatable;
 //creating parent column

            DataColumn parentcolumn=dataset1.Tables["student1"].Columns[0];
            DataColumn childcolumn=dataset1.Tables["stud_details"].Columns[1];

            DataRelation datarelation = new DataRelation("rel",
parentcolumn,childcolumn);
            //DataRelation datarelation = new DataRelation("rel",
dataset1.Tables["student1"].Columns[0],
dataset1.Tables["stud_details"].Columns[1]);

            dataset1.Relations.Add(datarelation);
            dataGridView5.DataSource = dataset1;
        }
    }

    private void panel3_Paint(object sender, PaintEventArgs e)
    {

    }
  }
}
```

Datagridview with dataset



With data table

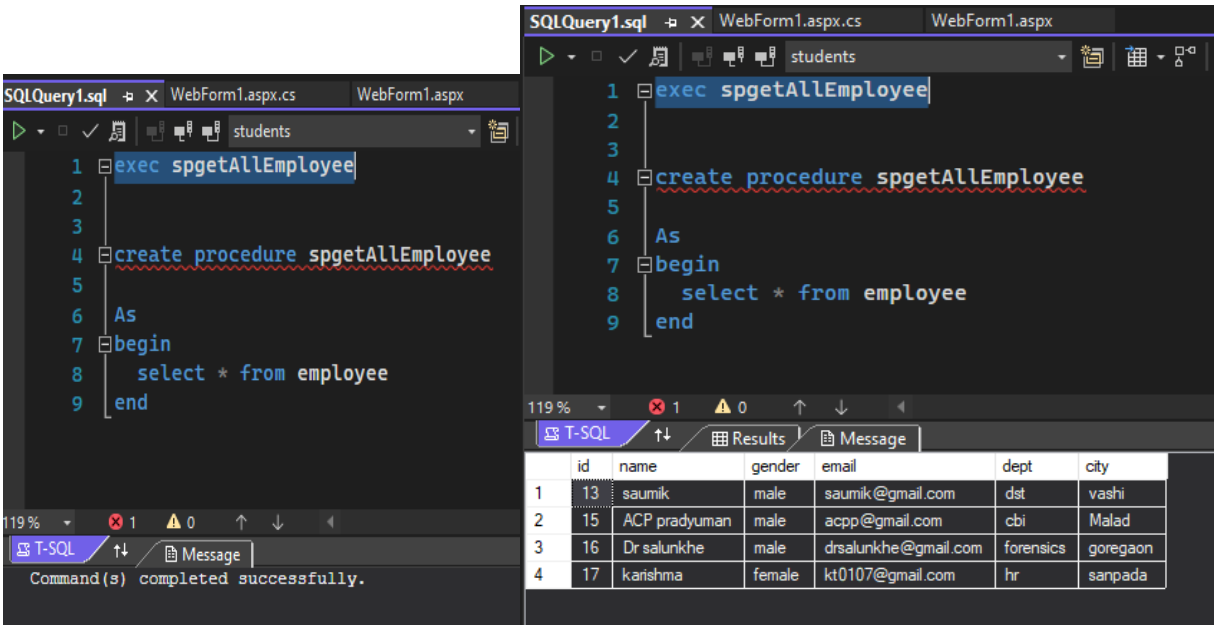

Form1
id name gende
13 saumik male
15 ACP pradyuman male
16 Dr salunkhe male
rollno name course
1 sanjay diwan python
2 kunal rathore databse ap
3 tanmay bhat digital mark
Id rollno name
1 1 sanjay diwan
id name gender email dept city rollno course marks
13 saumik male saumik@gmail.com dst vashi
15 ACP pradyuman male acpp@gmail.com cbi Malad
16 Dr salunkhe male drsalunkhe@gma... forensics goregaon
17 karishma female kt0107@gmail.com hr sanpada
sanjay diwan 1 python 88
kunal rathore 2 databse apps 82
tanmay bhat 3 digital marketing 94
karishma 4 maths 27
1 sanjay diwan sanjaydiwan@gm... 1
Show all


Form1
id name gende
13 saumik male
15 ACP pradyuman male
16 Dr salunkhe male
rollno name course
1 sanjay diwan python
2 kunal rathore databse ap
3 tanmay bhat digital mark
Id rollno name
1 1 sanjay diwan
id name gender email dept city rollno course marks
13 saumik male saumik@gmail.com dst vashi
15 ACP pradyuman male acpp@gmail.com cbi Malad
16 Dr salunkhe male drsalunkhe@gma... forensics goregaon
17 karishma female kt0107@gmail.com hr sanpada
sanjay diwan 1 python 88
kunal rathore 2 databse apps 82
tanmay bhat 3 digital marketing 94
karishma 4 maths 27
1 sanjay diwan sanjaydiwan@gm... 1
Show all

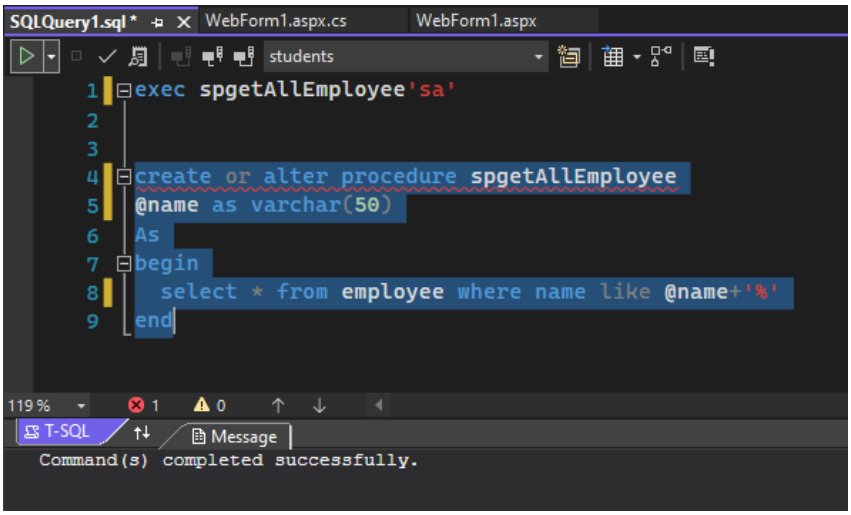
10May2022

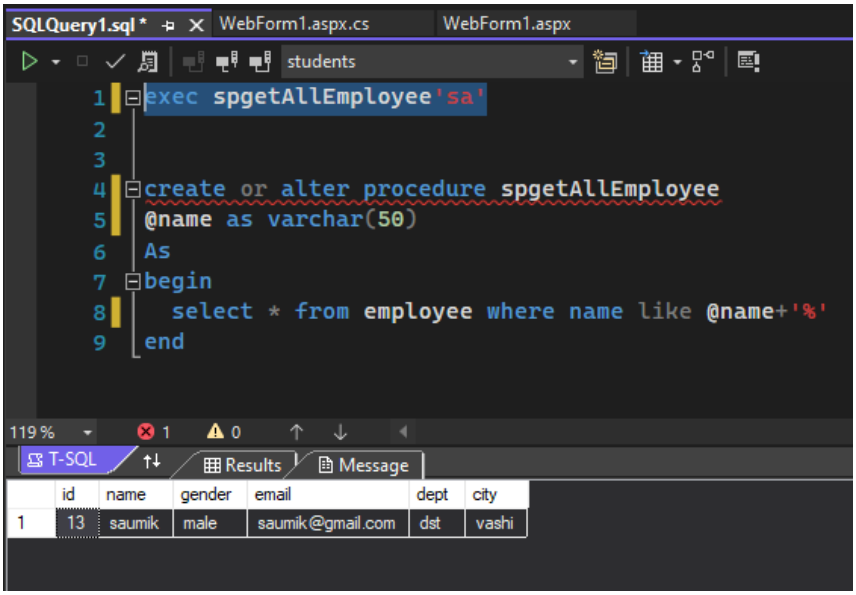
Practical 35:

Using stored procedures for crud operation in Asp.net Web application for crud operation



Parameterised stored procedure





Crud operation using stored procedure
Sql_crud.sql

```
create or alter procedure spallemp
As
begin
  select * from employee
end


create or alter procedure spdelemp
@id as varchar(50)
As
begin
  delete from employee where id = @id
end


create or alter  procedure spupdemp
@id as Int,
@name as varchar(50),
@gender as varchar(50),
@email as varchar(50),
@dept as varchar(50),
@city as Varchar(50)
As
begin
  update employee set
name=@name,gender=@gender,email=@email,dept=@dept,city=@city where
id=@id;
end

create or alter procedure spsrchemp
@id as INT
As
Begin
Select * from employee where id=@id;
end
```

```
create or alter procedure spinsemp
@name as varchar(50),
@gender as varchar(50),
@email as varchar(50),
@dept as varchar(50),
@city as Varchar(50)
As
begin
  insert into employee(name,gender,email,dept,city)
values(@name,@gender,@email,@dept,@city);
end
```

Webform1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs"
Inherits="WebApplication2_crudsql.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
         <asp:Panel ID="Panel1" runat="server" style="/*text-
align:center;*/margin:5px;">
            <asp:TextBox ID="TextBox2" runat="server"
placeholder="id"></asp:TextBox><br />
            <asp:TextBox ID="TextBox3" runat="server"
placeholder="name"></asp:TextBox><br />
```

```
            <asp:TextBox ID="TextBox4" runat="server" placeholder="gender"
></asp:TextBox><br />
            <asp:TextBox ID="TextBox5" runat="server"
placeholder="email"></asp:TextBox><br />
            <asp:TextBox ID="TextBox6" runat="server" placeholder="dept"
></asp:TextBox><br />
            <asp:TextBox ID="TextBox7" runat="server"
placeholder="city"></asp:TextBox><br />

            <asp:Button ID="Button2" runat="server" Text="Insert"
OnClick="Button2_Click" />
            <asp:Button ID="Button3" runat="server" Text="Delete"
OnClick="Button3_Click" />
            <asp:Button ID="Button4" runat="server" Text="Update"
OnClick="Button4_Click" style="height: 26px" />
            <asp:Button ID="Button5" runat="server" Text="show all employees"
OnClick="Button5_Click" />
             <asp:Button ID="Button6" runat="server" Text="search  record"
OnClick="Button6_Click" />

            </asp:Panel>

        </div>
        <div>
            <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
            <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
            <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click" />

            <asp:GridView ID="GridView1" runat="server" CellPadding="4"
ForeColor="#333333" GridLines="None">
                <AlternatingRowStyle BackColor="White" />
                <EditRowStyle BackColor="#7C6F57" />
                <FooterStyle BackColor="#1C5E55" Font-Bold="True"
ForeColor="White" />
```

```
            <HeaderStyle BackColor="#1C5E55" Font-Bold="True"
ForeColor="White" />
            <PagerStyle BackColor="#666666" ForeColor="White"
HorizontalAlign="Center" />
            <RowStyle BackColor="#E3EAEB" />
            <SelectedRowStyle BackColor="#C5BBAF" Font-Bold="True"
ForeColor="#333333" />
            <SortedAscendingCellStyle BackColor="#F8FAFA" />
            <SortedAscendingHeaderStyle BackColor="#246B61" />
            <SortedDescendingCellStyle BackColor="#D4DFE1" />
            <SortedDescendingHeaderStyle BackColor="#15524A" />
        </asp:GridView>
    </div>
  </form>
</body>
</html>
```

webform1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
using System.Configuration;
namespace WebApplication2_crudsql
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      public  string connection =
ConfigurationManager.ConnectionStrings["dbconnect"].ConnectionString;
      protected void Page_Load(object sender, EventArgs e)
```

```
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            using (SqlConnection sqlconnection = new SqlConnection(connection))
            {
                //to avoid sql injection attack there are two approaches
                //1. using paramtereized query
                sqlconnection.Open();
                //string search = "Select * from employee where name LIKE '" +
TextBox1.Text + "%'";
                string search = "exec spGetAllEmployee @name";
                Response.Write(search);
                //SqlCommand cmd = new SqlCommand(search, sqlconnection);
                SqlCommand cmd = new SqlCommand("spGetAllEmployee",
sqlconnection);
                cmd.CommandType = System.Data.CommandType.StoredProcedure;
                //cmd.Parameters.AddWithValue("@parameter", TextBox1.Text + "%");
                cmd.Parameters.AddWithValue("@name", TextBox1.Text);
                GridView1.DataSource = cmd.ExecuteReader();
                GridView1.DataBind();
                sqlconnection.Close();

                //  Response.Write("hello");
            }
        }

        protected void Button2_Click(object sender, EventArgs e)
        {
            //insert
            using (SqlConnection sqlconnection = new SqlConnection(connection))
            {
                sqlconnection.Open();
```

```
        SqlCommand cmd = new SqlCommand("spinsemp", sqlconnection);
        cmd.CommandType = System.Data.CommandType.StoredProcedure;

        cmd.Parameters.AddWithValue("@name", TextBox3.Text);
        cmd.Parameters.AddWithValue("@gender", TextBox4.Text);
        cmd.Parameters.AddWithValue("@email", TextBox5.Text);
        cmd.Parameters.AddWithValue("@dept", TextBox6.Text);
        cmd.Parameters.AddWithValue("@city", TextBox7.Text);
        int i = cmd.ExecuteNonQuery();
        Response.Write(i + " record inserted");
        sqlconnection.Close();

    }

}

protected void Button3_Click(object sender, EventArgs e)
{
    //delete
    using (SqlConnection sqlconnection = new SqlConnection(connection))
    {
        sqlconnection.Open();
        SqlCommand cmd = new SqlCommand("spdelemp", sqlconnection);
        cmd.CommandType = System.Data.CommandType.StoredProcedure;

        cmd.Parameters.AddWithValue("@id", Int32.Parse(TextBox2.Text));
        int i= cmd.ExecuteNonQuery();
        Response.Write(i+" record deleted");
        sqlconnection.Close();

    }


}
```

```
protected void Button4_Click(object sender, EventArgs e)
{
    //update
    using (SqlConnection sqlconnection = new SqlConnection(connection))
    {
        sqlconnection.Open();

        SqlCommand cmd = new SqlCommand("spupdemp", sqlconnection);
        cmd.CommandType = System.Data.CommandType.StoredProcedure;

        cmd.Parameters.AddWithValue("@id", TextBox2.Text);
        cmd.Parameters.AddWithValue("@name", TextBox3.Text);
        cmd.Parameters.AddWithValue("@gender", TextBox4.Text);
        cmd.Parameters.AddWithValue("@email", TextBox5.Text);
        cmd.Parameters.AddWithValue("@dept", TextBox6.Text);
        cmd.Parameters.AddWithValue("@city", TextBox7.Text);
        int i = cmd.ExecuteNonQuery();
        Response.Write(i + " record updated");
        sqlconnection.Close();
    }
}

protected void Button5_Click(object sender, EventArgs e)
{

    //select all records
    using (SqlConnection sqlconnection = new SqlConnection(connection))
    {
        sqlconnection.Open();

        SqlCommand cmd = new SqlCommand("spallemp", sqlconnection);
        cmd.CommandType = System.Data.CommandType.StoredProcedure;

        GridView1.DataSource = cmd.ExecuteReader();
```

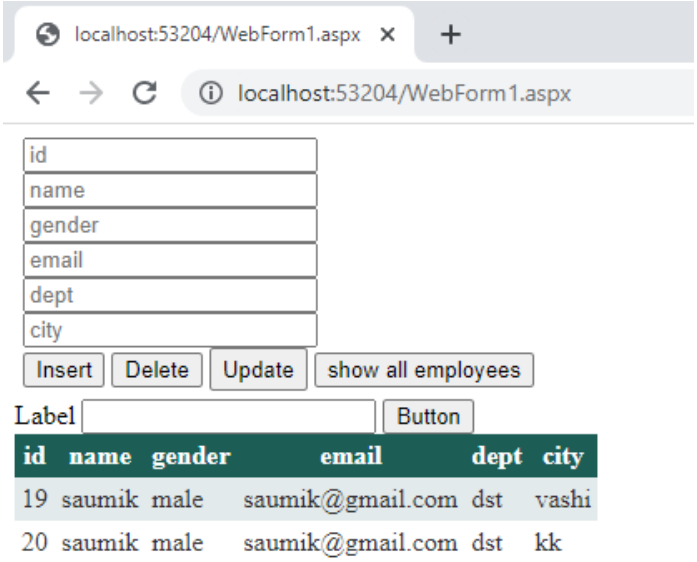
```
            GridView1.DataBind();
            sqlconnection.Close();
        }
    }

    protected void Button6_Click(object sender, EventArgs e)
    {
        using (SqlConnection sqlconnection = new SqlConnection(connection))
        {
            sqlconnection.Open();

            SqlCommand cmd = new SqlCommand("spsrchemp", sqlconnection);
            cmd.CommandType = System.Data.CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("@id", Int32.Parse(TextBox2.Text));

            //GridView1.DataSource = sqlread;
            //GridView1.DataBind();
            SqlDataReader sqlread=cmd.ExecuteReader();
            while (sqlread.Read())
            {
                TextBox3.Text = sqlread.GetString(1);
                TextBox4.Text = sqlread.GetString(2);
                TextBox5.Text = sqlread.GetString(3);
                TextBox6.Text = sqlread.GetString(4);
                TextBox7.Text = sqlread.GetString(5);

            }
            sqlconnection.Close();
        }
    }
  }
}
```

Web.config (Connection string)

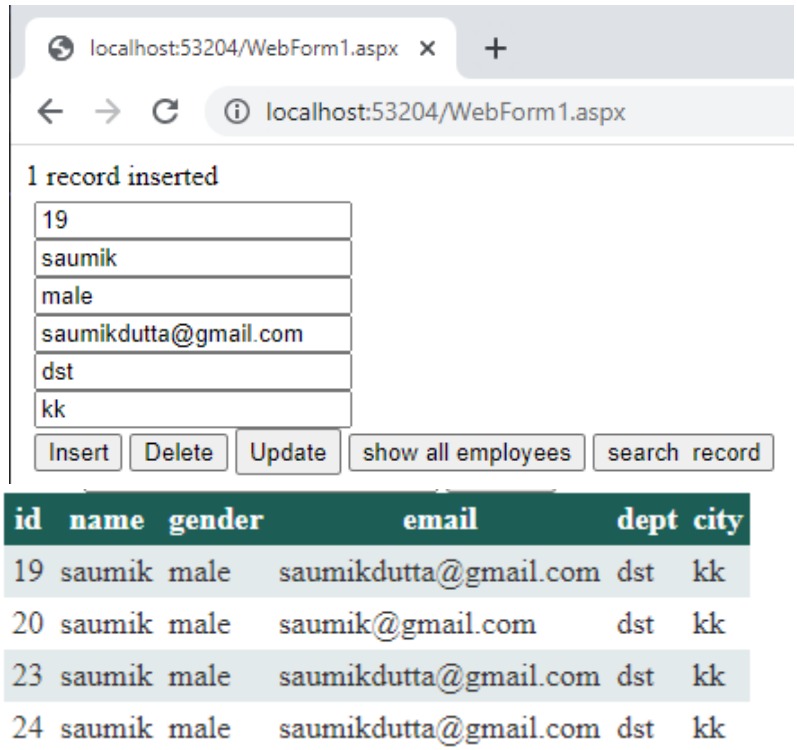
```
<?xml version="1.0" encoding="utf-8"?>
<!--
```

```
  For more information on how to configure your ASP.NET application, please visit
  https://go.microsoft.com/fwlink/?LinkId=169433
  -->
<configuration>
        <connectionStrings>
                <add name="dbconnect" connectionString="Data Source=.\sqlexpress03;Initial Catalog=students;Integrated Security=True"
                        providerName="System.Data.SqlClient"/>
        </connectionStrings>
        <system.web>
    <compilation debug="true" targetFramework="4.7.2" />
    <httpRuntime targetFramework="4.7.2" />
  </system.web>
  <system.codedom>
    <compilers>
      <compiler language="c#;cs;csharp" extension=".cs" type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default /nowarn:1659;1699;1701" />
      <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb" type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default /nowarn:41008 /define:_MYTYPE=\&quot;Web\&quot; /optionInfer+" />
    </compilers>
  </system.codedom>
</configuration>
```

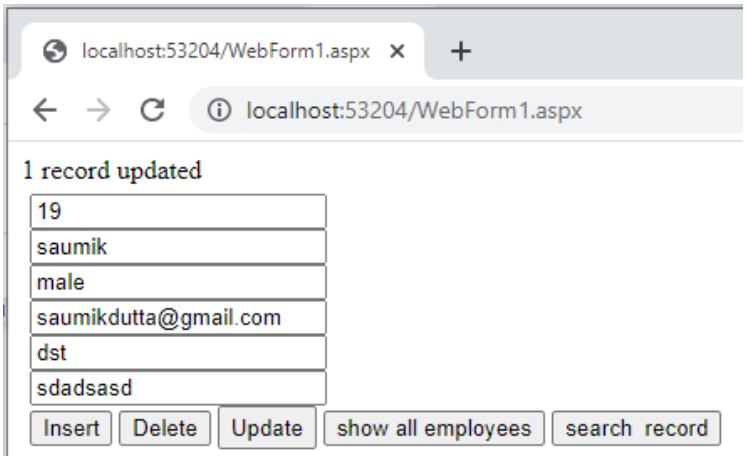
Showall employee



Insert emp records

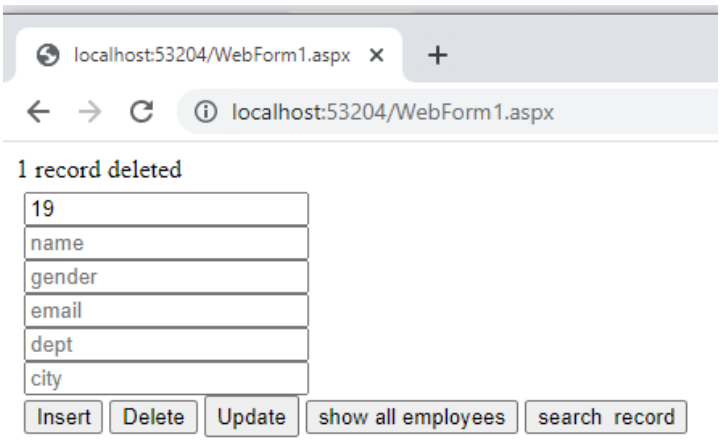


UPDATE Employee record



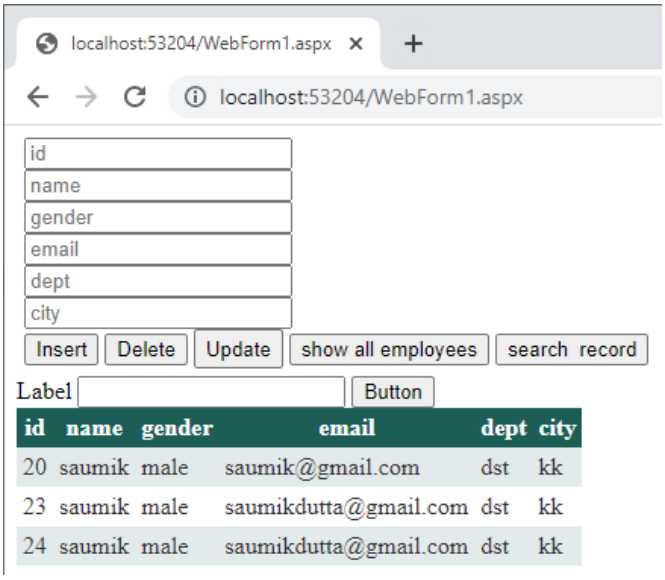
id	name	gender	email	dept	city
19	saumik	male	saumikdutta@gmail.com	dst	sdadsasd
20	saumik	male	saumik@gmail.com	dst	kk
23	saumik	male	saumikdutta@gmail.com	dst	kk
24	saumik	male	saumikdutta@gmail.com	dst	kk

Delete employee record

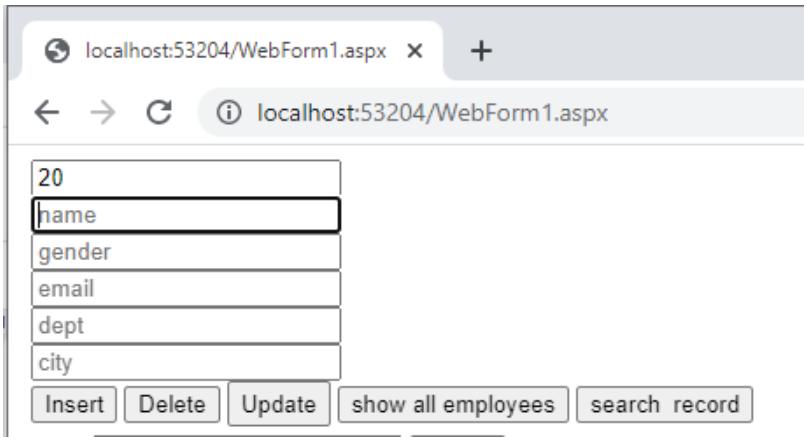


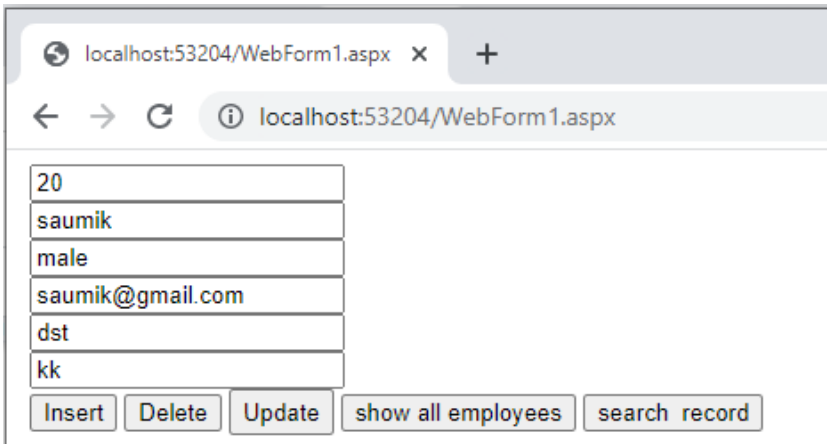
id	name	gender	email	dept	city
20	saumik	male	saumik@gmail.com	dst	kk
23	saumik	male	saumikdutta@gmail.com	dst	kk
24	saumik	male	saumikdutta@gmail.com	dst	kk

Show all record



Search record





10May2022

Practical 36:

Application Life Cycle

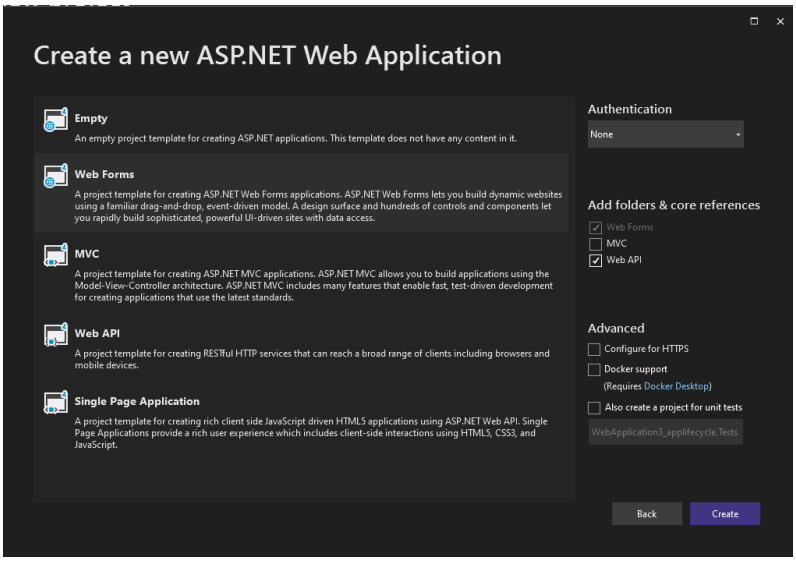
Global.asax.cs

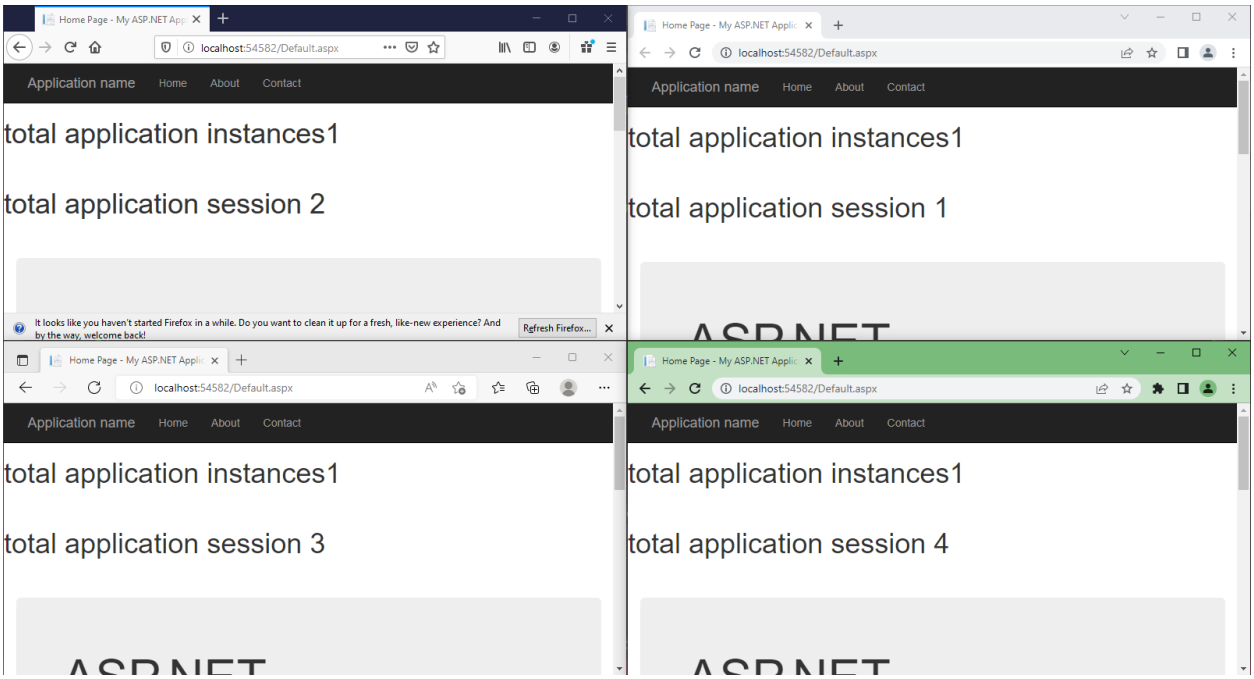
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Optimization;
using System.Web.Routing;
using System.Web.Security;
using System.Web.SessionState;
using System.Web.Http;

namespace WebApplication3_applifecycle
{
    public class Global : HttpApplication
    {
        void Application_Start(object sender, EventArgs e)
        {
            // Code that runs on application startup
/*          GlobalConfiguration.Configure(WebApiConfig.Register);
            RouteConfig.RegisterRoutes(RouteTable.Routes);
            BundleConfig.RegisterBundles(BundleTable.Bundles);*/

            Application["TotalApplications"] = 0;
            Application["TotalSessions"] = 0;
            Application["TotalApplications"] =(int)Application["TotalSessions"]+1;


        }

        void Session_Start(object sender, EventArgs e)
```

```
        {
            // Code that runs on application startup
            /*          GlobalConfiguration.Configure(WebApiConfig.Register);
                        RouteConfig.RegisterRoutes(RouteTable.Routes);
                        BundleConfig.RegisterBundles(BundleTable.Bundles);*/
            Application["TotalSessions"] = (int)Application["TotalSessions"]+1;

        }

        void Session_End(object sender, EventArgs e)
        {
            // Code that runs on application startup
            /*          GlobalConfiguration.Configure(WebApiConfig.Register);
                        RouteConfig.RegisterRoutes(RouteTable.Routes);
                        BundleConfig.RegisterBundles(BundleTable.Bundles);*/
            Application["TotalSessions"] = (int)Application["TotalSessions"] - 1;

        }

        void Application_End(object sender, EventArgs e)
        {
            // Code that runs on application startup
            /*          GlobalConfiguration.Configure(WebApiConfig.Register);
                        RouteConfig.RegisterRoutes(RouteTable.Routes);
                        BundleConfig.RegisterBundles(BundleTable.Bundles);*/

            Response.Write("Application end");

        }
    }
}
```

Default.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication3_applifecycle
{
    public partial class _Default : Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
          Response.Write("<h1>total application instances "  +
Application["TotalApplications"]+ "</h1><br>");
            Response.Write("<h1>total application session " +
Application["TotalSessions"]+ "</h1><br>");

        }
    }
}
```





2 profiles from chrome,1-internet explorer,1-Mozilla firefox

10May2022

Practical 37:

PageLifeCycle
Global.asax

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Optimization;
using System.Web.Routing;
using System.Web.Security;
using System.Web.SessionState;
using System.Web.Http;

namespace WebApplication3_applifecycle
{
    public class Global : HttpApplication
    {
        void Application_Start(object sender, EventArgs e)
        {
            // Code that runs on application startup
/*          GlobalConfiguration.Configure(WebApiConfig.Register);
            RouteConfig.RegisterRoutes(RouteTable.Routes);
            BundleConfig.RegisterBundles(BundleTable.Bundles);*/

            Application["TotalApplications"] = 0;
            Application["TotalSessions"] = 0;
            Application["TotalApplications"] =(int)Application["TotalSessions"]+1;


        }

        void Session_Start(object sender, EventArgs e)
```

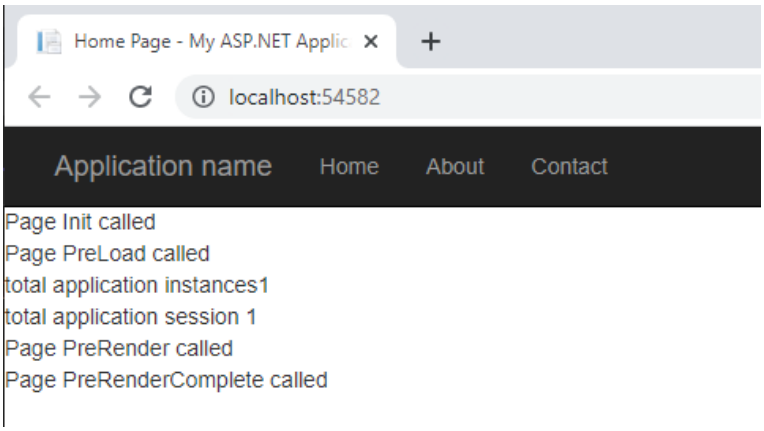
```
        {
            // Code that runs on application startup
            /*          GlobalConfiguration.Configure(WebApiConfig.Register);
                        RouteConfig.RegisterRoutes(RouteTable.Routes);
                        BundleConfig.RegisterBundles(BundleTable.Bundles);*/
            Application["TotalSessions"] = (int)Application["TotalSessions"]+1;

        }

        void Session_End(object sender, EventArgs e)
        {
            // Code that runs on application startup
            /*          GlobalConfiguration.Configure(WebApiConfig.Register);
                        RouteConfig.RegisterRoutes(RouteTable.Routes);
                        BundleConfig.RegisterBundles(BundleTable.Bundles);*/
            Application["TotalSessions"] = (int)Application["TotalSessions"] - 1;

        }

        void Application_End(object sender, EventArgs e)
        {
            // Code that runs on application startup
            /*          GlobalConfiguration.Configure(WebApiConfig.Register);
                        RouteConfig.RegisterRoutes(RouteTable.Routes);
                        BundleConfig.RegisterBundles(BundleTable.Bundles);*/

            Response.Write("Application end");
        }
    }
}
```

Default.aspx.cs

//cache,load,postback,controlleve events

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication3_applifecycle
{
    public partial class _Default : Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
 Response.Write("total application instances"+Application["TotalApplications"]+
"<br>");
  Response.Write("total application session "+Application["TotalSessions"]+
"<br>");
        }
        //protected void Page_UnLoad(object sender, EventArgs e)   //{
        //    Response.Write("Page Unload called"); //}

   protected void Page_PreRender(object sender, EventArgs e)
        {
            Response.Write("Page PreRender called<br>");

        }
        protected void Page_PreRenderComplete(object sender, EventArgs e)
        {
            Response.Write("Page PreRenderComplete called<br>");

        }
        protected void Page_Init(object sender, EventArgs e)
        {
```

```
        Response.Write("Page Init called<br>");

    }

    protected void Page_PreLoad(object sender, EventArgs e)
    {
        Response.Write("Page PreLoad called<br>");
    }

    protected void Page_PreLoadComplete(object sender, EventArgs e)
    {
        Response.Write("Page PreLoadComplete called<br>");
    }

  }
}
```



11May2022

Practical 38:

Request Response and Session object

1.Redirection using Response.Redirect("<webform_name.aspx>")

WebForm1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs"
Inherits="WebApplication3_applifecycle.WebForm1" %>

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Button ID="Button1" runat="server" Text="Home"
OnClick="Button1_Click" />
        </div>
    </form>
</body>
</html>
```

WebForm1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
namespace WebApplication3_applifecycle
{
    public partial class WebForm1 : System.Web.UI.Page
```

```
    {
        protected void Page_Load(object sender, EventArgs e)
        {
        }
        protected void Button1_Click(object sender, EventArgs e)
        {
            Response.Redirect("WebForm2.aspx");
        }
    }
}
```

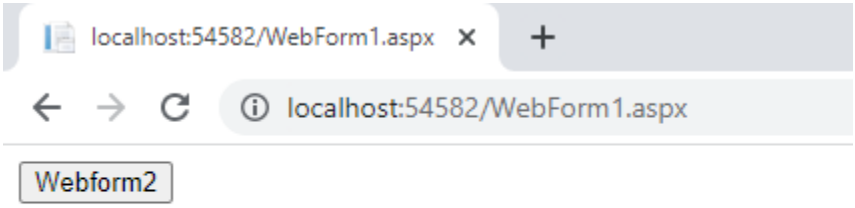
WebForm2.aspx

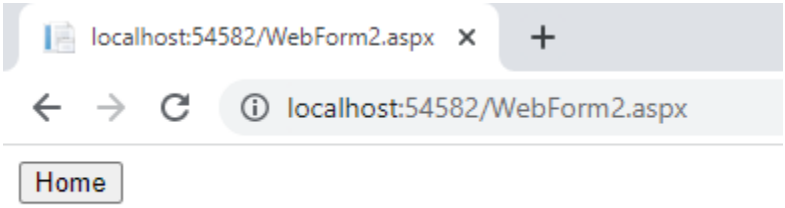
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm2.aspx.cs"
Inherits="WebApplication3_applifecycle.WebForm2" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Button ID="Button1" runat="server" Text="Home"
OnClick="Button1_Click" />
        </div>
    </form>
</body>
</html>
```

WebForm2.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication3_applifecycle
{
    public partial class WebForm2 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {        }
        protected void Button1_Click(object sender, EventArgs e)
        {
            Response.Redirect("WebForm1.aspx");
        }
    }
}
```





Validators

1. Regular field and compare field validator for username and password,confirm password respectively

WebForm1.aspx

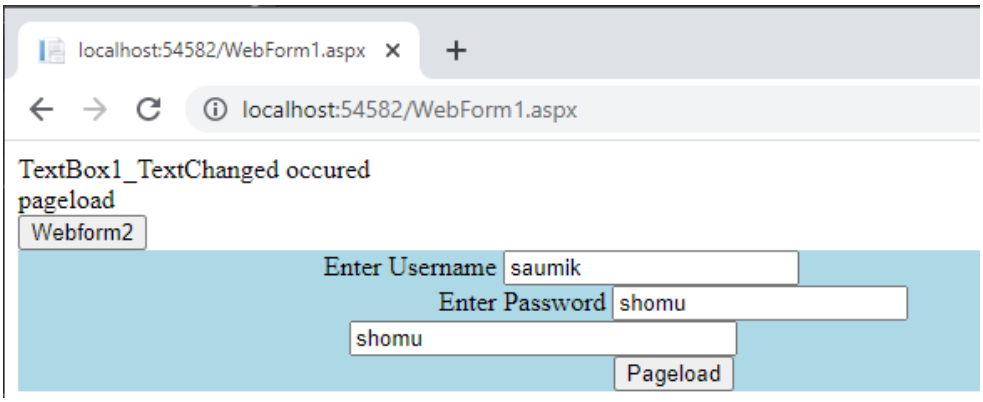
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs"
Inherits="WebApplication3_applifecycle.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server"> <title></title></head>
<body>
    <form id="form1" runat="server">
        <div>  <asp:Button ID="Button1" runat="server" Text="Webform2"
OnClick="Button1_Click" />   </div>
   <div  style="background-color:lightblue; text-align:center;">
            <asp:Label ID="Label1" runat="server" Text="Enter
Username"></asp:Label>
            <asp:TextBox ID="TextBox1" runat="server"
OnTextChanged="TextBox1_TextChanged"></asp:TextBox>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
runat="server" ErrorMessage="*Username required"
ControlToValidate="TextBox1"></asp:RequiredFieldValidator> <br />
            <asp:Label ID="Label2" runat="server" Text="Enter
Password"></asp:Label>
            <asp:TextBox ID="TextBox2"  runat="server"></asp:TextBox><br />
             <asp:TextBox ID="TextBox3"  runat="server"
Width="232px"></asp:TextBox>
            <asp:CompareValidator ID="CompareValidator1" runat="server"
ErrorMessage="*password donot match" ControlToValidate="TextBox3"
ControlToCompare="TextBox2"
ValueToCompare="TextBox2"></asp:CompareValidator> <br />
 <asp:Button ID="Button2" runat="server" Text="Pageload"
OnClick="Button2_Click" />
```

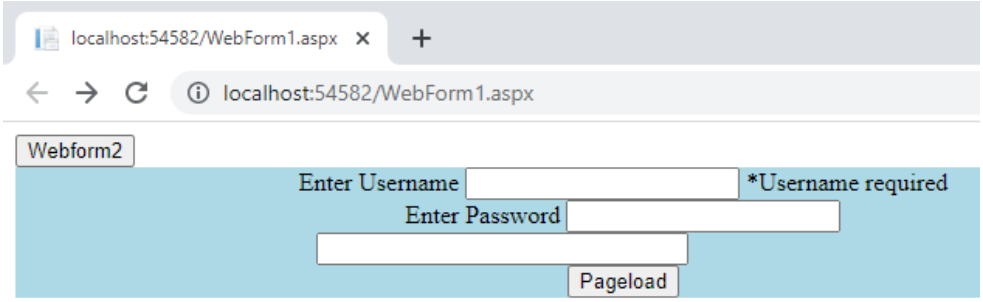
```
 </div> </form>
</body>
</html>
```

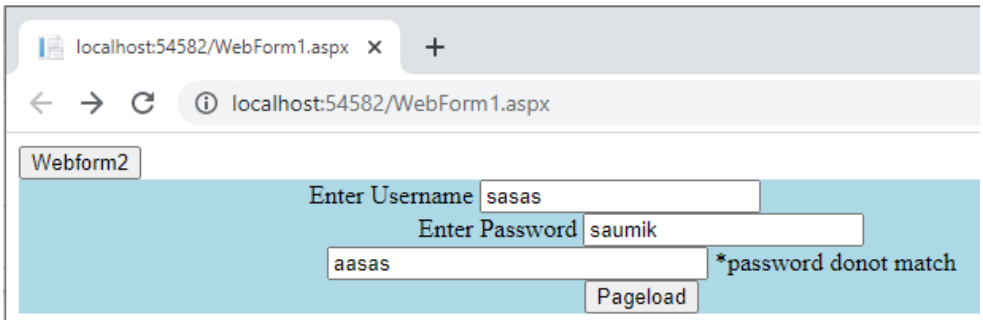
WebForm1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication3_applifecycle
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {  }

        protected void Button1_Click(object sender, EventArgs e)
        {
            Response.Redirect("WebForm2.aspx");
        }

        protected void TextBox1_TextChanged(object sender, EventArgs e)
        {
            Response.Write("TextBox1_TextChanged occured<br/>");
        }
        protected void Button2_Click(object sender, EventArgs e)
        {
            Response.Write("pageload");
        }
    }
```

}







2.retrieveing value using Request and session
WebForm1.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inherits="WebApplication3_applifecycle.WebForm1" %>

<!DOCTYPE html>

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title> </head>
<body>
   <form id="form1" runat="server">
      <div></div>

       <div  style="background-color:lightblue; text-align:center;">
         <asp:Label ID="Label1" runat="server" Text="Enter
Username"></asp:Label>
         <asp:TextBox ID="TextBox1" runat="server"
OnTextChanged="TextBox1_TextChanged"></asp:TextBox>
         <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
runat="server" ErrorMessage="*Username required"
ControlToValidate="TextBox1"></asp:RequiredFieldValidator>  <br />
         <asp:Label ID="Label2" runat="server" Text="Enter
Password"></asp:Label>
         <asp:TextBox ID="TextBox2"  runat="server"></asp:TextBox><br />
          <asp:TextBox ID="TextBox3"  runat="server"
Width="151px"></asp:TextBox>
         <asp:CompareValidator ID="CompareValidator1" runat="server"
ErrorMessage="*password donot match" ControlToValidate="TextBox3"
ControlToCompare="TextBox2"
ValueToCompare="TextBox2"></asp:CompareValidator>  <br />
 <asp:Button ID="Button2" runat="server" Text="Pageload"
OnClick="Button2_Click" />   <br />  <br />
         <asp:Button ID="Button1" runat="server" Text="Webform2"
OnClick="Button1_Click" style="margin-left: 17px" />

      </div>
   </form>
</body>
</html>
```

Webform1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication3_applifecycle
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {    }

        protected void Button1_Click(object sender, EventArgs e)
        {

Response.Redirect("WebForm2.aspx?email="+TextBox1.Text+"&password="+Te
xtBox2.Text);
        }

        protected void TextBox1_TextChanged(object sender, EventArgs e)
        {
            Session["email"] = TextBox1.Text;
            Response.Write("TextBox1_TextChanged occured<br/>");
        }
        protected void Button2_Click(object sender, EventArgs e)
        {  Response.Write("pageload");   }
    }
}
```

WebForm2.aspx

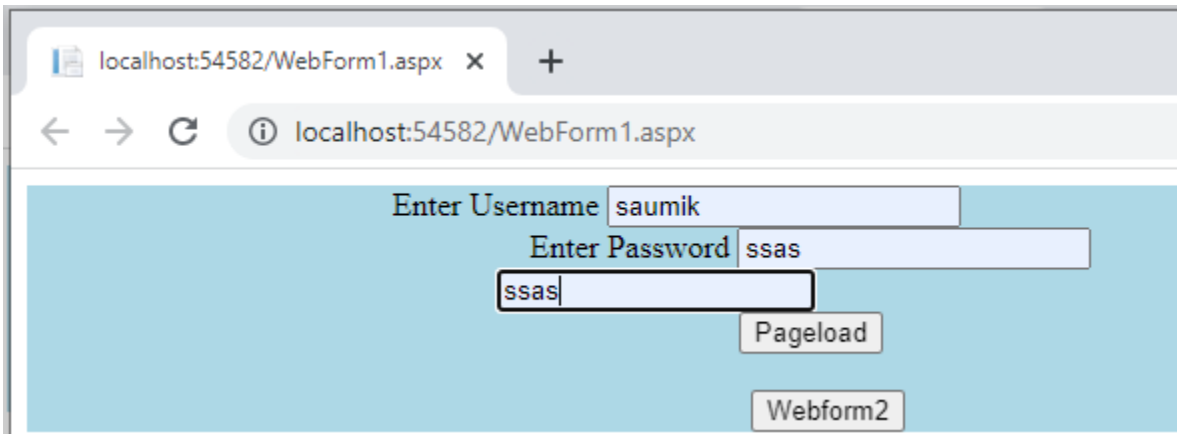
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm2.aspx.cs"
Inherits="WebApplication3_applifecycle.WebForm2" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body style="background-color:lightblue;">
   <form id="form1" runat="server">
      <div>
         <asp:Button ID="Button1" runat="server" Text="Home"
OnClick="Button1_Click" />
      </div>


      <%--<div align="center">
         <asp:Label ID="Label1" runat="server" Text="Enter
Username"></asp:Label>
         <asp:TextBox ID="TextBox1" runat="server"
OnTextChanged="TextBox1_TextChanged"></asp:TextBox>
         <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
runat="server" ErrorMessage="Username required"
ControlToValidate="TextBox1"></asp:RequiredFieldValidator>

         <br />
         <asp:Label ID="Label2" runat="server" Text="Enter
Password"></asp:Label>\
         <asp:TextBox ID="TextBox2"  runat="server"></asp:TextBox>
          <asp:TextBox ID="TextBox3"  runat="server"></asp:TextBox>
         <asp:CompareValidator ID="CompareValidator1" runat="server"
ErrorMessage="password donot match" ControlToValidate="TextBox2"
```

```
ControlToCompare="TextBox3"
ValueToCompare="TextBox2"></asp:CompareValidator>

            <br />
            <asp:Button ID="Button2" runat="server" Text="Pageload"
OnClick="Button2_Click" />

        </div>--%>
    </form>
</body>
</html>
```

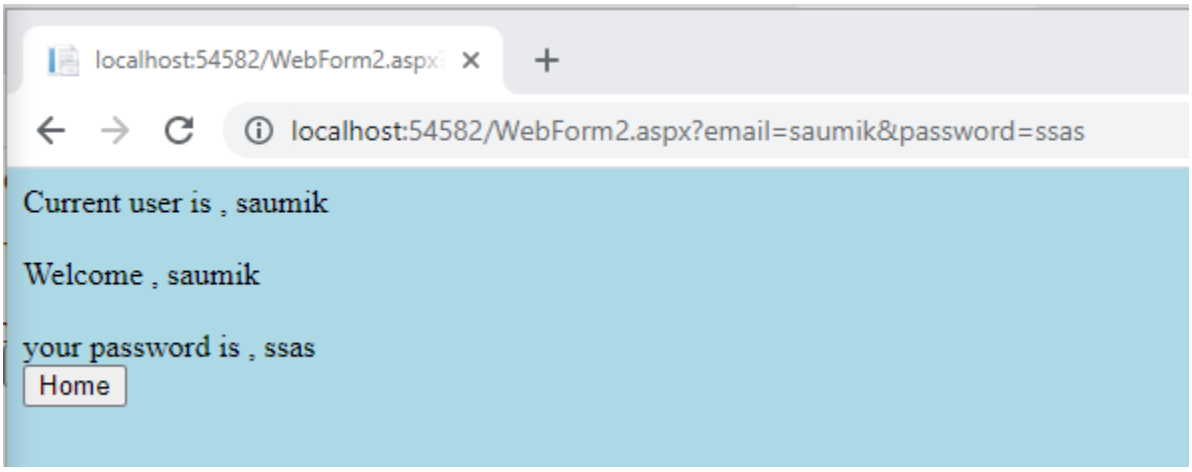
WebForm2.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication3_applifecycle
{
    public partial class WebForm2 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            Response.Write("Current user is , " + Session["email"] + "<br />");
            Response.Write("<br />Welcome , " + Request["email"] + "<br />");
            Response.Write("<br />your password is , " + Request["password"] + "");
        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            Response.Redirect("WebForm1.aspx");
        }
```

```
        protected void Button2_Click(object sender, EventArgs e)
        {
            Response.Write("pageload");
        }

        protected void TextBox1_TextChanged(object sender, EventArgs e)
        {
            Response.Write("TextBox1_TextChanged occured");
        }
    }
}
```



Url is query string and using request value is retrieved

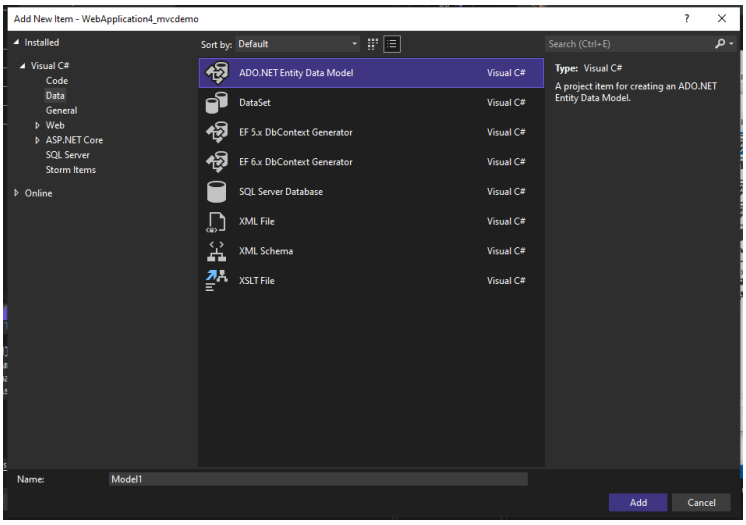


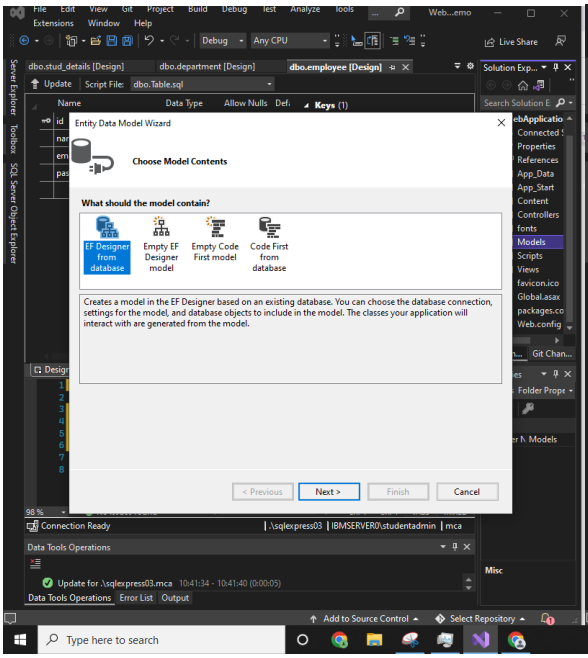
10may2022

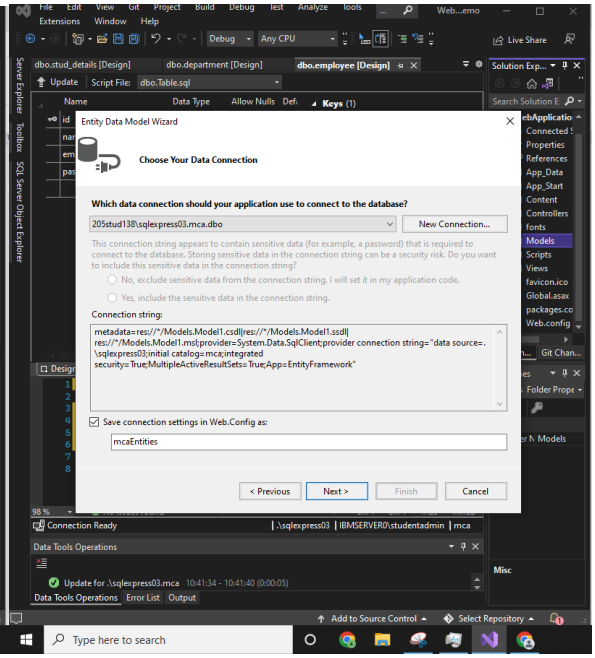
Practical 39:

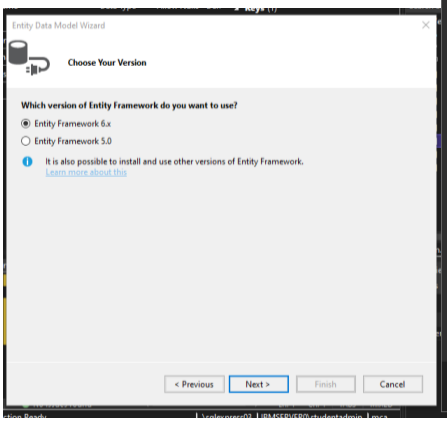
Asp.net MVC framework and entity models
Before creating controller build project once
1.right click on model-> add-> new item -> ado.net entity data model
2.select type and data connection->assign name-> click on add

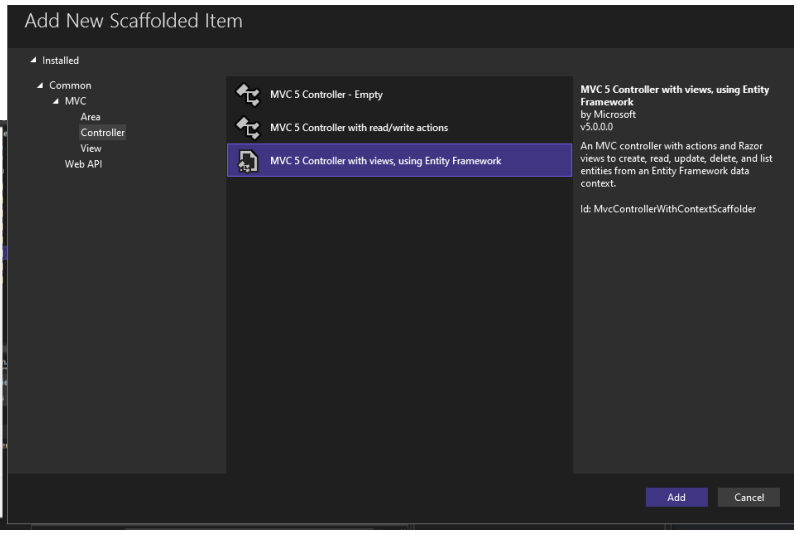
Creating controller
1.build project once
2.right click on controller add controller select model class and datacontext

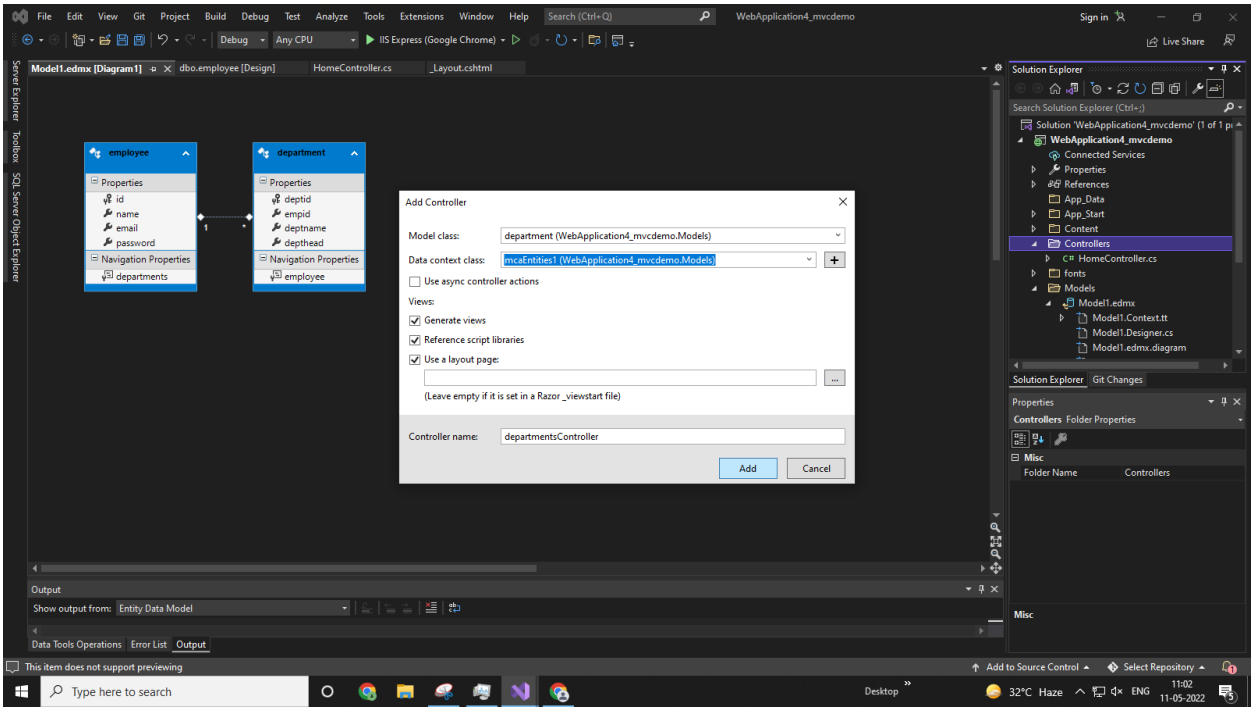


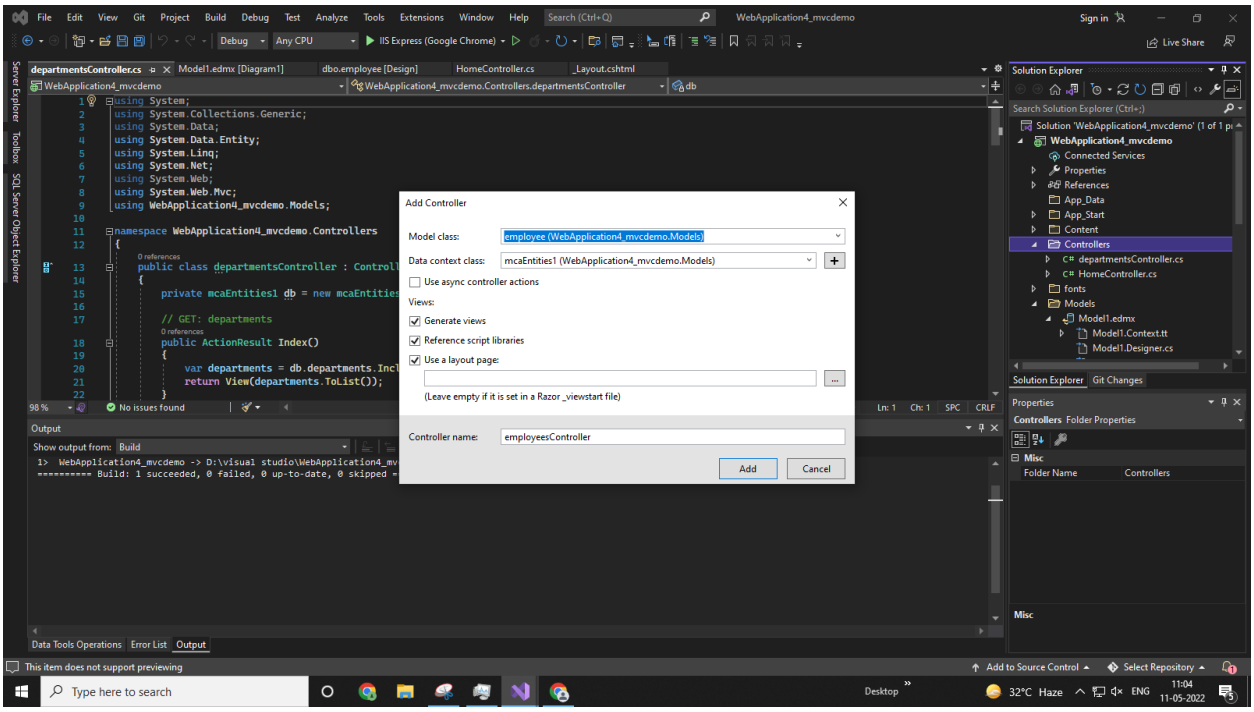

Entity Data Model Wizard
Choose Model Contents
What should the model contain?
EF Designer from database
Empty EF Designer model
Empty Code First model
Code First from database
Creates a model in the EF Designer based on an existing database. You can choose the database connection, settings for the model, and database objects to include in the model. The classes your application will interact with are generated from the model.
< Previous
Next >
Finish
Cancel


Entity Data Model Wizard
Choose Your Data Connection
Which data connection should your application use to connect to the database?
205stud138\sqlexpress03.mca.dbo
New Connection...
Connection string:
Save connection settings in Web.Config as:
mcaEntities
< Previous
Next >
Finish
Cancel


Entity Data Model Wizard
Choose Your Version
Which version of Entity Framework do you want to use?
Entity Framework 6.x
Entity Framework 5.0
It is also possible to install and use other versions of Entity Framework.
Learn more about this
< Previous
Next >
Finish
Cancel


Add New Scaffolded Item
Installed
Common
MVC
Area
Controller
View
Web API
MVC 5 Controller - Empty
MVC 5 Controller with read/write actions
MVC 5 Controller with views, using Entity Framework
MVC 5 Controller with views, using Entity Framework
by Microsoft
v5.0.0.0
An MVC controller with actions and Razor views to create, read, update, delete, and list entities from an Entity Framework data context.
Id: MvcControllerWithContextScaffolder
Add
Cancel



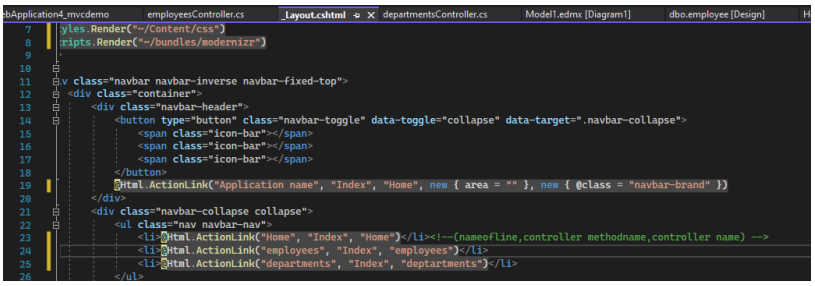


Layout.cshtml

<li>@Html.ActionLink("Home", "Index", "Home")</li><!-- (nameofline,controller methodname,controller name) -->

<li>@Html.ActionLink("employees", "Index", "employees")</li>

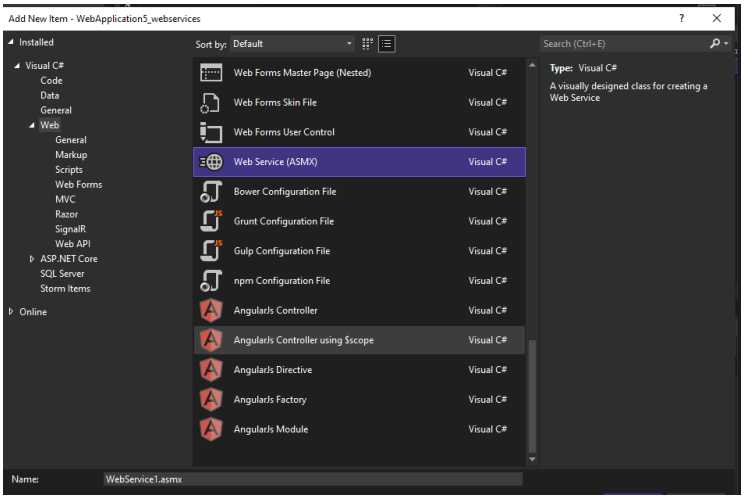
<li>@Html.ActionLink("departments", "Index", "deptartments")</li>

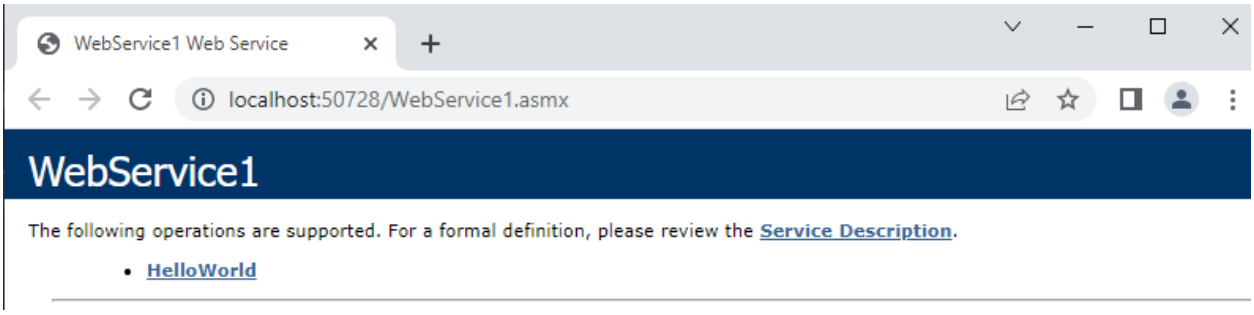


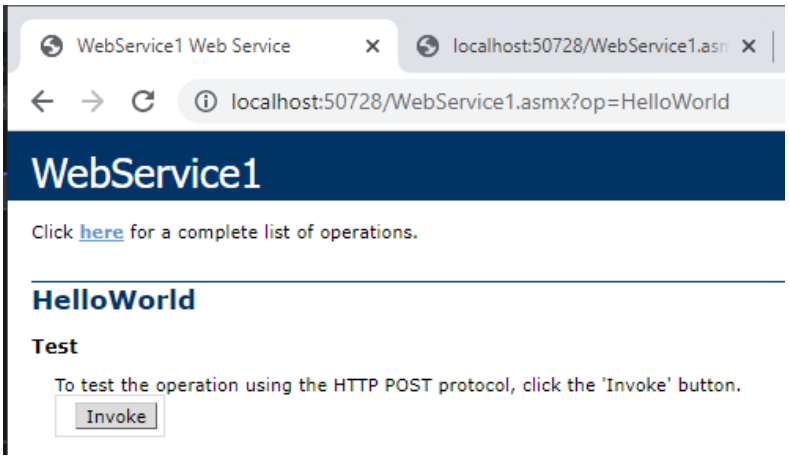
12 MAY 2022

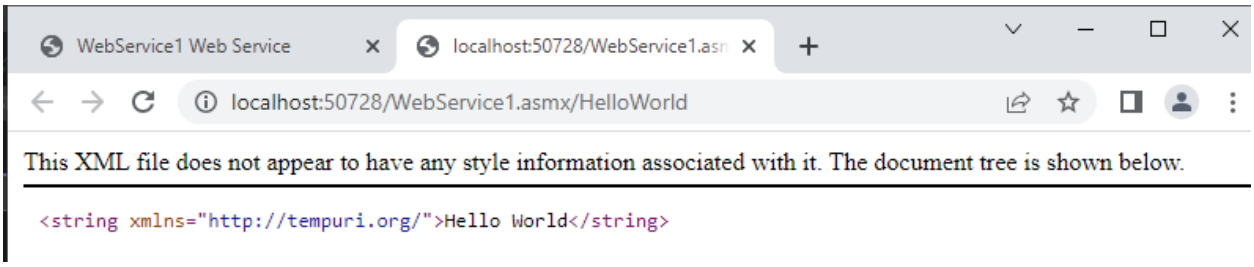
Practical 40:

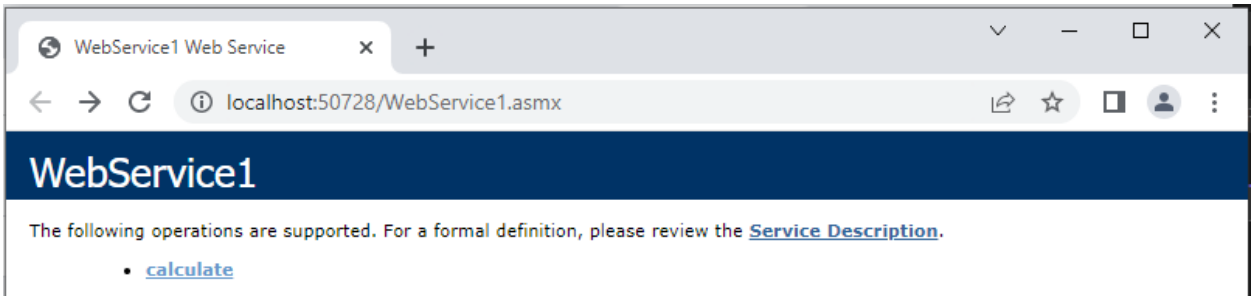
Demonstration of WebServices

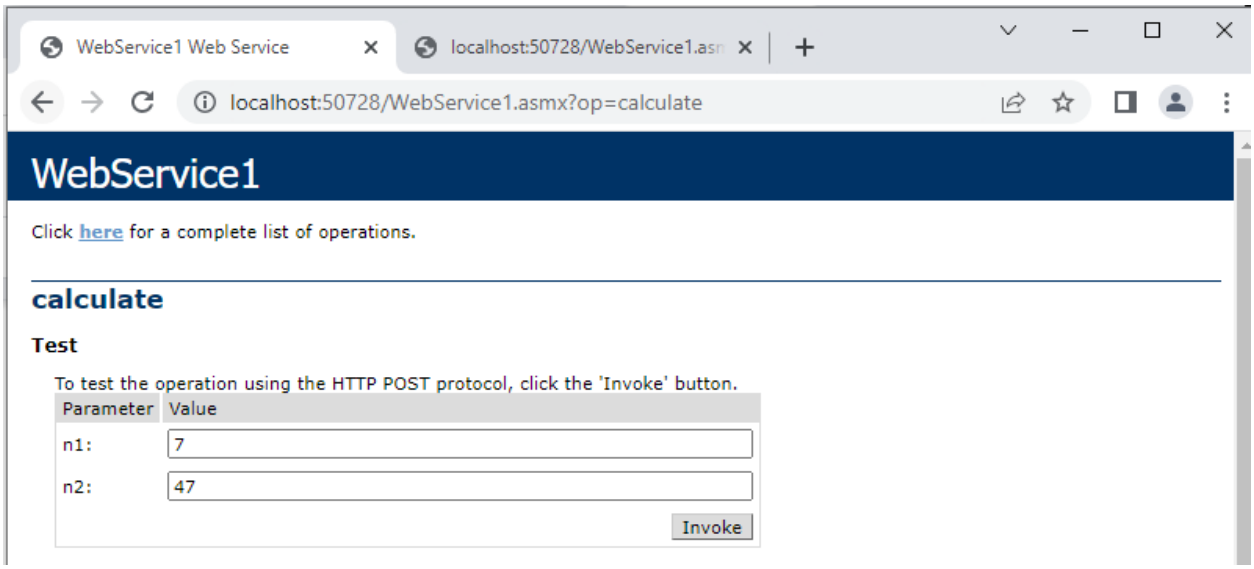


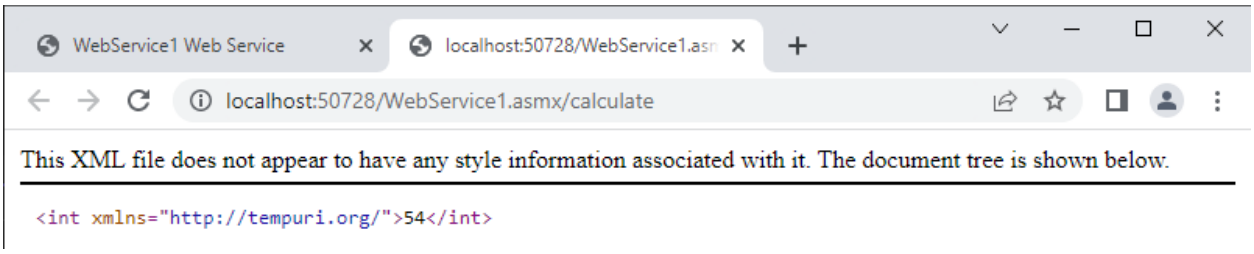



WebService1 Web Service
localhost:50728/WebService1.asn
localhost:50728/WebService1.asmx?op=HelloWorld
WebService1
Click here for a complete list of operations.
HelloWorld
Test
To test the operation using the HTTP POST protocol, click the 'Invoke' button.
Invoke

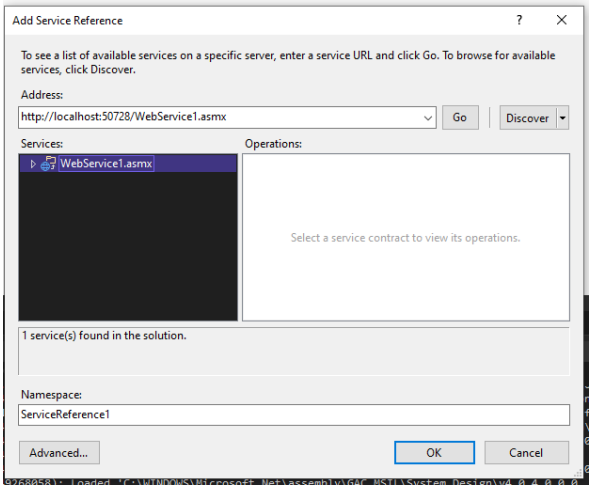

WebService1 Web Service
localhost:50728/WebService1.asn
localhost:50728/WebService1.asmx/HelloWorld
This XML file does not appear to have any style information associated with it. The document tree is shown below.
<string xmlns="http://tempuri.org/">Hello World</string>


WebService1 Web Service
localhost:50728/WebService1.asmx
WebService1
The following operations are supported. For a formal definition, please review the Service Description.
calculate





RIGHT CLICK ON 2ND PROJECT IN SAME SOLUTION ADD SERVICE REFERENCE



Using webservice in a webform(session object)
WebApplication5_webservices
Webservice1.asmx

```
    [WebMethod(EnableSession = true, Description = "This method add two
numbers")]
public int calculate(int num1, int num2)
    {
        List<string> history;
        if (Session["calcHistory"] == null)
        {
            history = new List<string>();
            //history.Add("There are no recent calculatios");
        }
        else
        {
            history = (List<string>)Session["calcHistory"];
        }
        string resString = num1.ToString() + " + " + num2.ToString()
        + " = " + (num1 + num2).ToString();
        history.Add(resString);
        Session["calcHistory"] = history;
        return num1 + num2;
    }
    [WebMethod(EnableSession = true)]
    public List<string> getRecentCalculations()
    {

        if (Session["calcHistory"] == null)
        {
            List<string> history = new List<string>();
            history.Add("There are no recent calculatios");
```

```
            //return history;
           //  return (List<string>)Session["calcHistory"];
            return history;
        }
        else
        {
            return (List<string>)Session["calcHistory"];
        }
    }
```

webservice_client(set as startup project after adding service reference)
WebForm1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs" Inherits="webservice_client.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:TextBox ID="TextBox1" runat="server" placeholder="Enter
Num1"></asp:TextBox>
            <br />
            <asp:TextBox ID="TextBox2" runat="server" placeholder="Enter
Num2"></asp:TextBox>
            <br />
            <asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
Text="Button" />
```

```
            <br />
             <asp:GridView ID="GridView1" runat="server" CellPadding="4"
ForeColor="#333333" GridLines="None">
                <AlternatingRowStyle BackColor="White" />
                <EditRowStyle BackColor="#7C6F57" />
                <FooterStyle BackColor="#1C5E55" Font-Bold="True"
ForeColor="White" />
                <HeaderStyle BackColor="#1C5E55" Font-Bold="True"
ForeColor="White" />
                <PagerStyle BackColor="#666666" ForeColor="White"
HorizontalAlign="Center" />
                <RowStyle BackColor="#E3EAEB" />
                <SelectedRowStyle BackColor="#C5BBAF" Font-Bold="True"
ForeColor="#333333" />
                <SortedAscendingCellStyle BackColor="#F8FAFA" />
                <SortedAscendingHeaderStyle BackColor="#246B61" />
                <SortedDescendingCellStyle BackColor="#D4DFE1" />
                <SortedDescendingHeaderStyle BackColor="#15524A" />
            </asp:GridView>
    </div>
    </form>
</body>
</html>
```

Web.config

changes added in system.service.model->biindings

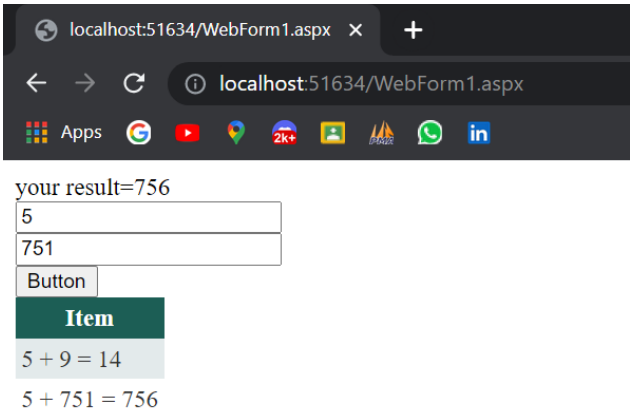
```
 <basicHttpBinding>
        <binding name="WebService1Soap" allowCookies="true"/>
        <binding name="WebService1Soap1" allowCookies="true" />
         <binding allowCookies="true" name="Historycalc1" />
      </basicHttpBinding>
```

WebForm1.aspx.cs

```
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace webservice_client
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
        }
        protected void Button1_Click(object sender, EventArgs e)
        {
            webservice_client.calcweb.WebService1SoapClient soap = new
calcweb.WebService1SoapClient();

            int result = soap.calculate(Int32. Parse(TextBox1.Text),
Int32.Parse(TextBox2.Text));

            GridView1.DataSource = soap.getRecentCalculations();
            GridView1.DataBind();
              Response.Write("your result=" + result);



        }



    }
}
```

Output:-


localhost:51634/WebForm1.aspx
localhost:51634/WebForm1.aspx
Apps
your result=756
5
751
Button
Item
5 + 9 = 14
5 + 751 = 756