BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT

AVALAHALLI, YELAHANKA, BANGALORE - 64

DEPARTMENT OF MCA

Advanced

Programming Lab

FIFTH SEMESTER

MCA

DOT NET Laboratory
16 MCA 57

Prepared by:
Prof. Drakshaveni.G
Asst Professor

VISION

To develop quality professionals in Computer Applications who can provide sustainable solutions to the societal and industrial needs.

MISSION

Facilitate effective learning environment through quality education and guidance with state-of-the-art facilities and orientation towards research and entrepreneurial skills.

Graduates of MCA Program will be able to

PEO1

Develop innovative IT applications to meet industrial and societal needs

PEO 2

Adapt themselves to changing IT requirements through life-long learning.

PEO 3

Exhibit leadership skill and advance in their chosen career.

Programme Outcomes

PO 1: Apply knowledge of computing fundamentals, computing specialization, mathematics and domain knowledge to provide IT solutions

PO 2: Identify, analyse and solve IT problems using fundamental principles of mathematics and computing sciences

PO 3: Design, Develop and evaluate software solutions to meet societal and environmental concerns

PO 4: Conduct investigations of complex problems using research based knowledge and methods to provide valid conclusions.

PO 5: Select and apply appropriate techniques and modern tools for complex computing activities

PO 6: Practice and follow professional ethics and cyber regulations

PO 7: Involve in life-long learning for continual development as an IT professional.

PO 8: Apply and demonstrate computing and management principles to manage projects in multidisciplinary environments by involving in different roles

PO 9: Comprehend, write effective reports and make quality presentations.

PO 10: Understand the impact of IT solutions on socio-environmental issues

PO 11: Work collaboratively as a member or leader in multidisciplinary teams.

PO 12: Identify potential business opportunities and innovate to create value for the society and seize that opportunity

Table of Contents

1. Lab Instructions

1. **Duration of each laboratory session is 3 hour /week.**

2. **Maximum marks for Internal Assessment is 20.**

3. **Two internal tests will be conducted for the laboratory.**

4. **Award of I.A marks is on the average of two internal tests, Assignment marks continuous evaluation.**

5. **University examination is of 3 hour duration.**

6. **Students will be asked to execute one full question picked on lottery one from PART A and one from PART B.**

7. **Maximum marks for the Examination is 80.**

2. Hardware and Software Requirements

Hardware:

    RAM   : 512 MB
    HDD   : 40 GB

Software:

    OS   : Any Windows

    Compiler   : Visual studio any version

3. Introduction to Dot Net

.Net is a framework, which is a collection of tools, technologies, and languages all these work together in a framework to provide the solutions that are needed to easily build and deploy truly robust enterprise applications. It is not a operating system nor a programming language. It's a layer between the OS and the programming language. It consist of CLR (Common Language Runtime), CTS (Common type System) and CLS (Common Language Specification).

.Net supports many programming languages like VB.NET,C# etc. It provides common set of class libraries to .net based programming languages.

Here programs are executed using C# (pronounced "C sharp") programming language. C# is almost same as Java. No pointers required. It manages the memory automatically. Supports operator overloading and interface-based programming techniques. C# can produce code that can run only on .NET environment.

Visual C# .NET is Microsoft's C# development tool. It includes an interactive development environment, visual designers for building Windows and Web applications, a compiler, and a debugger. Visual C# .NET is part of a suite of products, called Visual Studio .NET, that also includes Visual Basic .NET, Visual C++ .NET, and the JScript scripting language.

Namespaces are the way to organize .NET Framework Class Library into a logical grouping according to their functionality, usability as well as category they should belong to. Namespaces are logical grouping of types for the purpose of identification. The System Namespaces is the root for types in the .NET Framework. In .Net languages every program is created with a default Namespaces. Programmers can also create their own Namespaces in .Net languages

In C# programming the class need not be same as the file name. Here is a basic C# program:

```
using System; -------------------------------------------------------→    Importing a Namespace
//This program illustrates C# basic syntax              → Comments
 public class SimpleProgram                                → Class Wrapper
 {
   static void Main()syntax                                → Main method
    {
        Console.WriteLine("Hello");
    }
 }
```

Using is a C# keyword, which is a reserved word, which have special meaning a in a language and therefore cannot be used by programmers as the names of variable, functions or any of the other C# building blocks

A namespace is a logical grouping of predefined C# programming element. "using System;" statement – is required to compile and run the program properly.

// - to comment just to the end of line i.e. single line comment /* this is a multi line or block comment */

"Class Wrapper" – A class declaration in C# is composed of attributes, modifiers, the class name and a body. The body contains class members that can include constants, variables, methods, properties, events, operators. "Main Method" – it serves as the entry point for a C# program. When the program executable is invoked, the system will call the Main method to launch the application.

Sl no	4.Index of Lab Programs
	Part A
1	Write a Program in C# to demonstrate Command line arguments processing for the following. a) To find the square root of a given number. b) To find the sum & average of three numbers.
2	Write a Program in C# to demonstrate the following a) Boxing and Unboxing b) Invalid Unboxing.
3	Write a program in C# to add Two complex numbers using Operator overloading.
4	Write a Program in C# to find the sum of each row of given jagged array of 3 inner arrays
5	Write a Program in C# to demonstrate Array Out of Bound Exception using Try, Catch and Finally blocks.
6	Write a Program to Demonstrate Use of Virtual and override key words in C# with a simple program.
7	Write a Program in C# to create and implement a Delegate for any two arithmetic operations
8	Write a Program in C# to demonstrate abstract class and abstract methods in C#.
9	Write a program to Set & Get the Name & Age of a person using Properties of C# to illustrate the use of different properties in C#.
10	Write a Program in C# Demonstrate arrays of interface types (for runtime polymorphism).
	PART-B
1	Consider the Database db_EMS (Employee Management System) consisting of the following tables : tbl_Designations (IdDesignation: int, Designation: string) tbl_EmployeeDetails(IdEmployee: int, EmployeeName: string, ContactNumber: string, IdDesignation: int, IdReportingTo: int) Develop a suitable window application using C#.NET having following options. 1. Enter new Employee details with designation & Reporting Manager. 2.　　Display all the Project Leaders (In a Grid) reporting to selected Project Managers (In a Combo box). 3.　　Display all the Engineers (In a Grid) reporting to selected Project Leader (In a Combo box). 4.　　Display all the Employees (In a Grid) with their reporting Manager (No Value for PM). NOTE: tbl_Designation is a static table containing the following Rows in it. <table><tr><td>1</td><td>Project Manager</td></tr><tr><td>2</td><td>Project Leader</td></tr><tr><td>3</td><td>Engineer</td></tr></table>

| 2 | Consider the Database db_LSA (Lecturer Subject Allocation) consisting of the following |

	tables: tbl_Subjects(IdSubject: int, SubjectCode: string, SubjectName: string) tbl_Lecturers(IdLecturer: int, LecturerName: string, ContactNumber: string) tbl_LecturerSubjects(IdSubject: int, SubjectCode: string, IdLecturer: int) Develop a suitable window application using C#.NET having following options. 1. Enter new Subject Details. 2. Enter New Lecturer Details. 3. Subject Allocation with Lecturer Name in a Combo box and subjects to be allocated in Grid with checkbox Column. 4. Display all the subjects allocated (In a Grid) to the selected Lecturer (In a Combo Box).
3	Consider the database db_VSS (Vehicle Service Station) consisting of the following tables: tbl_VehicleTypes(IdVehicleType: int, VehicleType: string, ServiceCharge: int) tbl_ServiceDetails(IdService: int, VehicleNumber: string, ServiceDetails: string, IdVehicleType: int) Develop a suitable window application using C#.NET having following options. 1. Enter new Service Details for the Selected Vehicle Type (In a Combo Box). 2. Update the Existing Service Charges to Database. 3. Total Service Charges Collected for the Selected Vehicle (In a Combo box) with total amount displayed in a text box. NOTE: tbl_VehicleType is a static table containing the following Rows in it. | 1	Two Wheeler	500
---	---	---
2	Four Wheeler	1000
3	Three Wheeler	700	
4	Develop a web application using C#.NET and ASP.NET for the Postal System Management. The master page should contain the hyper links for adding Area Details, Postman details, Letter distributions and View Letters. Consider the database db_PSM (Postal System Management) consisting of the following tables: tbl_AreaDetails(IdArea: int, AreaName: string) tbl_PostmanDetails(IdPostman: int, PostmanName: string, ContactNumber: string, IdArea: int) tbl_AreaLetters(IdLetter: int, LetterAddress: string, IdArea: int) Develop the suitable content pages for the above created 4 hyper links with the following details: 1. Enter New Area Details 2. Enter New Postman Details with the Area he/she is in-charge of (display Area in a Combo box) 3. Enter all the Letters distributed to the selected Area (display Area in a Combo box) 4. Display all the Letter addresses (In a Grid) to be distributed by the selected Postman (In a Combo box)
5	Develop a web application using C#.NET and ASP.NET for the Complaint Management System. The master page should contain the hyper links for Add Engineer, Complaint Registration, Complaint Allocation View Complaints. Consider the database db_CMS (Complaint Management System) consisting of the following tables: tbl_Departments(IdDepartment: int, DepartmentName: string)

tbl_Engineers(IdEngineer: int, EngineerName: string, ContactNumber: string,IdDepartment: int) tbl_RegisteredComplaints(IdComplaint: int, ComplaintDescription: string) tbl_DepartmentComplaints(IdDepartment: int, IdComplaint: int)

Develop the suitable content pages for the above created 4 hyper links with the following details:

· Enter New Engineers belonging to the selected department (displayed in a combo box)

· Register a new Complaint with a submit button.

· View all registered complaints & allocate to the corresponding department (displayed in a combo box)

· Display all the Complaints (In a Grid) to be handled by the selected Engineer (In a Combo box)

NOTE: Consider the table tbl_Departments as a static table containing some pre-entered departments, which are displayed in all the remaining modules.

PART-A

5. PROGRAM- 1

1 .Write a Program in C# to demonstrate Command line arguments processing for the following.
 a) To find the square root of a given number.
 b) To find the sum & average of three numbers.

Basically main method can have two Signatures. static void Main(string[] args) or static int Main(string[] args)
In which string[] args will store whatever is passed through command line to the application. We can process all the elements passed into command Line by simply processing the array string[] args. The parameter of the Main method is a string array that represents the commandline arguments. To check for the existence of the arguments can, it be done by testing the length property. We can access Command Line Arguments globally Using Environment Class's GetCommandLineArguments() method which returns string[] array of all CommandLine Arguments. So one can use command line arguments passed to Main function any were in Program without passing it any were.

a) To find the square root of a given number.

```csharp
// A Program to find the square root of a given number in C# to demonstrate command
line arguments
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Text;

namespace sq
{
    class program1_square
    {
        public static void Main(string[] args)
        {
            //Declare variable
            double argsValue
            = 0; double
            sqrtValue = 0;
            //check the length of command line
            argument if (args.Length == 0)
            {
                Console.WriteLine("************************************");
                Console.WriteLine("There is no command line Argument defined");
```
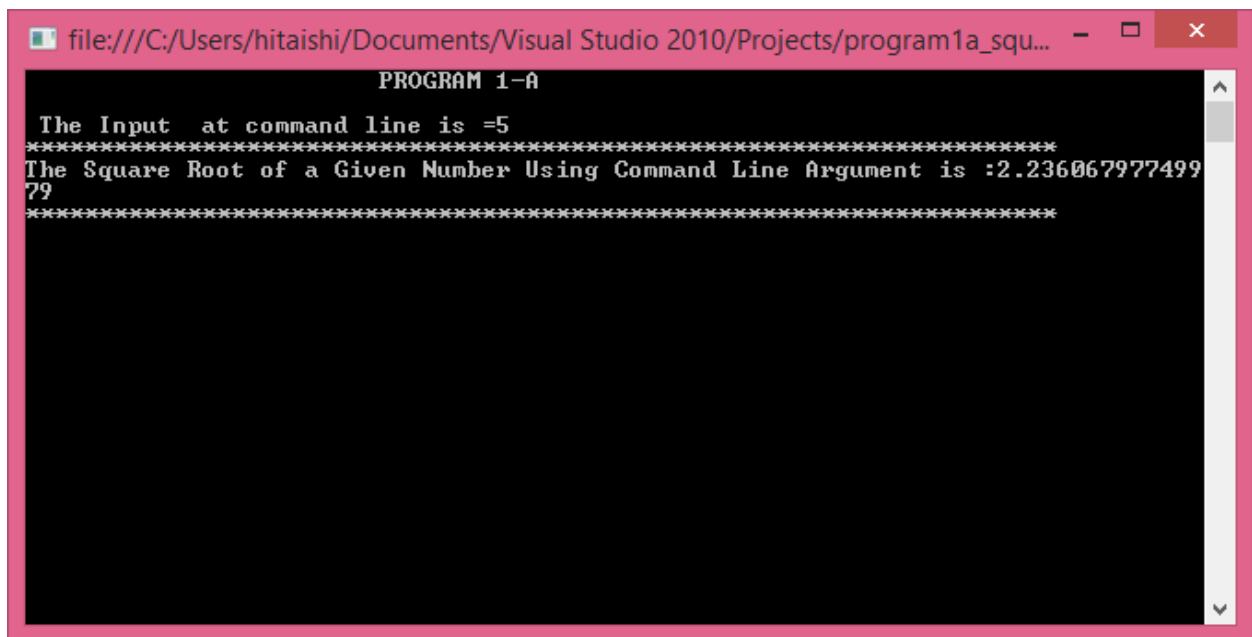
```
            Console.WriteLine("**************************************");
            Console.ReadLine();
            return;
        }
    //Find square root of the number using math
        argsValue =
        double.Parse(args[0].ToString());
        sqrtValue = Math.Sqrt(argsValue);
    //Display sqrt of the number
    Console.WriteLine("\t \t\tPROGRAM
    1-A");
    Console.WriteLine("\n The Input at command line is =" + argsValue);
    Console.WriteLine("**************************************************");
    Console.WriteLine("The Square Root of a Given Number Using Command Line
                       Argument is :{0}", sqrtValue);
    Console.WriteLine("**************************************************");
    Console.ReadLine();


        }
    }

}
```
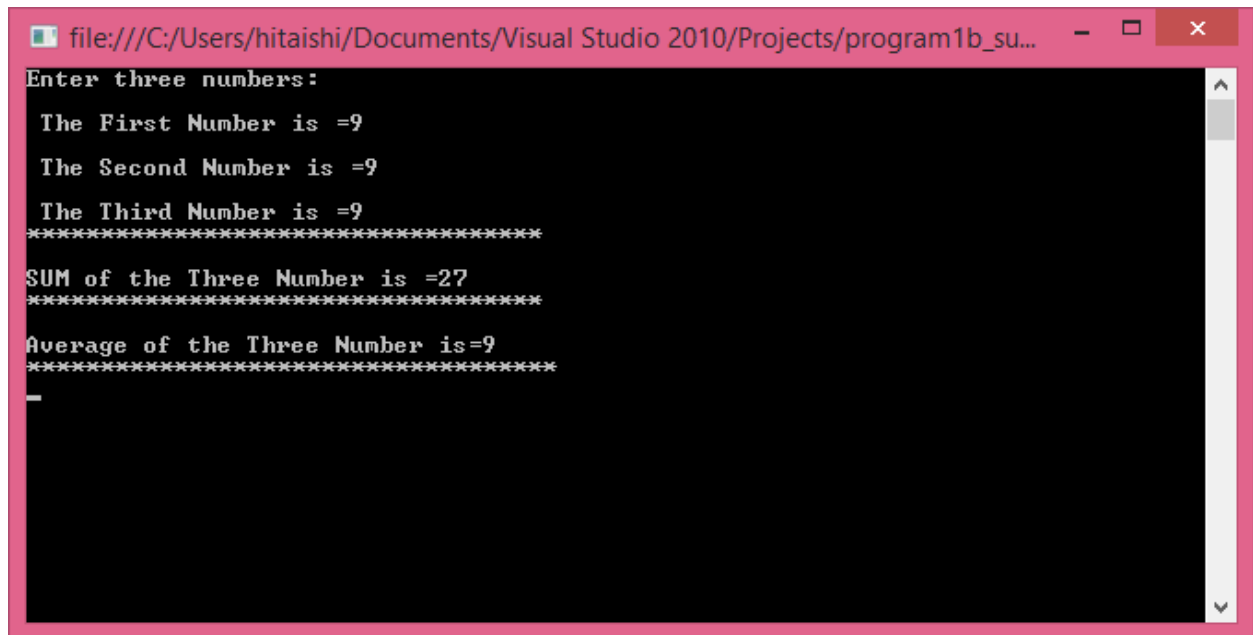
b) To find the sum & average of three numbers.

```csharp
//program to find sum & average of three numbers in c# using command line
argument using System;
class sum_avg
{
    public static void Main(String[] args)
    {
        double a, b, c;
        double sum = 0, av = 0;
        Console.WriteLine("Enter three
        numbers:"); a =
        double.Parse(args[0].ToString());

        b =
        double.Parse(args[1].ToString());
        c =
        double.Parse(args[2].ToString());
        Console.WriteLine("\n The First Number is =" + a);
        Console.WriteLine("\n The Second Number is =" + b);
        Console.WriteLine("\n The Third Number is =" + c);
        //to calculate sum of three
        nos sum = a + b + c;
        //to calculate
        average av = sum
        / 3;
        Console.WriteLine("*********************************");
        Console.WriteLine("\n SUM of the Three Number is =" + sum);
        Console.WriteLine("*********************************");
        Console.WriteLine("\n Average of the Three Number is=" + av);
        Console.WriteLine("*********************************");
        Console.ReadLine();
    }
}
```

```
file:///C:/Users/hitaishi/Documents/Visual Studio 2010/Projects/program1b_su...

Enter three numbers:

 The First Number is =9

 The Second Number is =9

 The Third Number is =9
****************************************

SUM of the Three Number is =27
****************************************

Average of the Three Number is=9
****************************************

_
```

6. PROGRAM- 2

2. Write a Program in C# to demonstrate the following
a) Boxing and Unboxing
b) Invalid Unboxing.

Boxing

It is defined as the process of explicitly converting a value type into a corresponding reference type by storing the variable in a System.Object.

Consider an example a variable of type short:
short s = 25;

If, during the course of your application, you wish to represent this value type as a reference type, you would "box" the value as follows:

// Box the value into an object

reference. object objShort = s;

When you box a value, the CLR allocates a new object on the heap and copies the value type's value (in this case, 25) into that instance. What is returned to you is a reference to the newly allocated object.

UnBoxing

It is the process of converting the value held in the object reference back into a corresponding value type on the stack. The unboxing operation begins by verifying that the receiving data type is equivalent to the boxed type, and if so, it copies the value back into a local stack-based variable.

The following unboxing operation works successfully, given that the underlying type of the objShort is indeed a short.
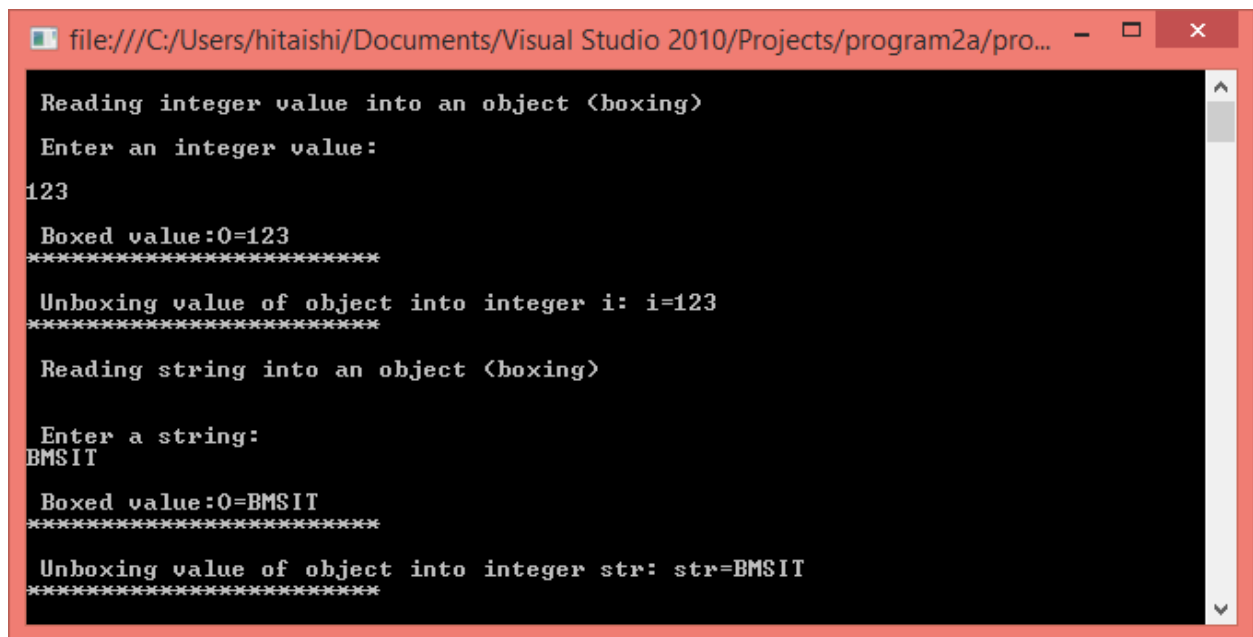
// Unbox the reference back into a corresponding

short. short anotherShort = (short)objShort;

a) Boxing and Unboxing

```csharp
// A Program to demonstrate Boxing and Unboxing

using System;
namespace
labproject
{
    class program2a
    {
        public static void Main()
        {
            Object o;
            Console.WriteLine("\n Reading integer value into an object
            (boxing)"); Console.WriteLine("\n Enter an integer value:\n");
            o=int.Parse(Console.ReadLine());
            int i =(int)o;
            Console.WriteLine(" \n Boxed value:O="+o);
            Console.WriteLine("**********************");
            Console.WriteLine(" \n Unboxing value of object into integer i:
            i="+i); Console.WriteLine("**********************");

            Console.WriteLine(" \n Reading string into an object (boxing)
            \n"); Console.WriteLine("\n Enter a string:");
            o=Console.ReadLine();
            string str =(String)o;
            Console.WriteLine(" \n Boxed value:O="+o);
            Console.WriteLine("**********************");
            Console.WriteLine(" \n Unboxing value of object into integer str:
            str="+str); Console.WriteLine("**********************");
            Console.ReadLine();
        }
    }
}
```

file:///C:/Users/hitaishi/Documents/Visual Studio 2010/Projects/program2a/pro...

```
Reading integer value into an object (boxing)

Enter an integer value:
123

Boxed value:0=123
**************************
Unboxing value of object into integer i: i=123
**************************
Reading string into an object (boxing)


Enter a string:
BMSIT

Boxed value:0=BMSIT
**************************
Unboxing value of object into integer str: str=BMSIT
**************************
```
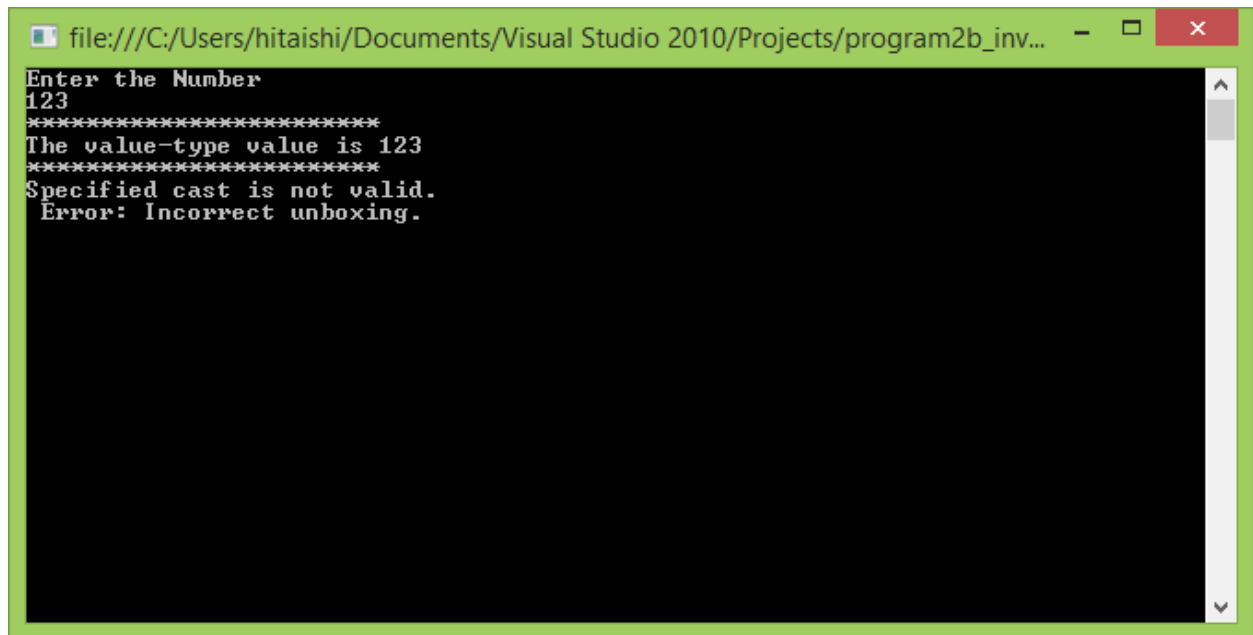
b) Invalid Unboxing.

```csharp
// A Program to demonstrate invaild
Boxing using System;
namespace labproject
{
    class program2a
    {
        public static void Main()
        {
            Console.WriteLine("Enter the
            Number"); int n =
            int.Parse(Console.ReadLine());
            object ob = n;
            Console.WriteLine("*********************"
            ); Console.WriteLine("The value-type value is {0}",
            ob);
            Console.WriteLine("*********************"
            ); try
            {
                int s = (Char)ob;
                Console.WriteLine("*********************");
                Console.WriteLine("The object_type value is ", s);
                Console.WriteLine("*********************");
                Console.WriteLine("Unboxed is OK \n");
            }
            catch (System.InvalidCastException e)
            {
            System.Console.WriteLine("{0} \n Error: Incorrect unboxing.", e.Message);
            }
            Console.ReadLine();
        }
    }
}
```

7. PROGRAM- 3

3. Write a program in C# to add Two complex numbers using Operator overloading .

Operator overloading, also known as overloading, provides a way to define and use operators such as +, -, and / for user-defined classes or structs. It allows us to define/redefine the way operators work with our classes and structs. This allows programmers to make their custom types look and feel like simple types such as int and string. It consists of nothing more than a method declared by the keyword operator and followed by an operator. There are three types of overloadable operators called unary, binary, and conversion. Not all operators of each type can be overloaded.

Overloading Unary Operators: They include +, -, !, ~, ++, --, true, and false.

Overloading Binary Operator : Binary operators are those that require two operands/parameters for the operation. One of the parameters has to be of a type in which the operator is declared.

They include +, -, *, /, %, &, |, ^, <<, >>, ==, !=, >, <, >=, and <=.

Overloading Conversion Operator: Conversion operators are those that involve converting from one data type to another through assignment. There are implicit and explicit conversions. · Implicit conversions are those that involve direct assignment of one type to another. · Explicit conversions are conversions that require one type to be casted as another type in order to perform the conversion. Conversions that may cause exceptions or result in loss of data as the type is converted should be handled as explicit conversions.

```csharp
// Write a program in C# to add Two complex numbers using Operator
overloading . using System;
namespace labproject
{
    class program3
    {
        int X,Y;
        public program3( int x, int y)
        {
            X
            =
            x;
            Y
            =
            y;
        }
        public static program3 operator +(program3 v1,program3 v2)
        {
            return new program3 (v1.X + v2.X, v1.Y + v2.Y);
        }
        public static void Main()
        {
        program3 v1 = new program3 (20, 30);
```
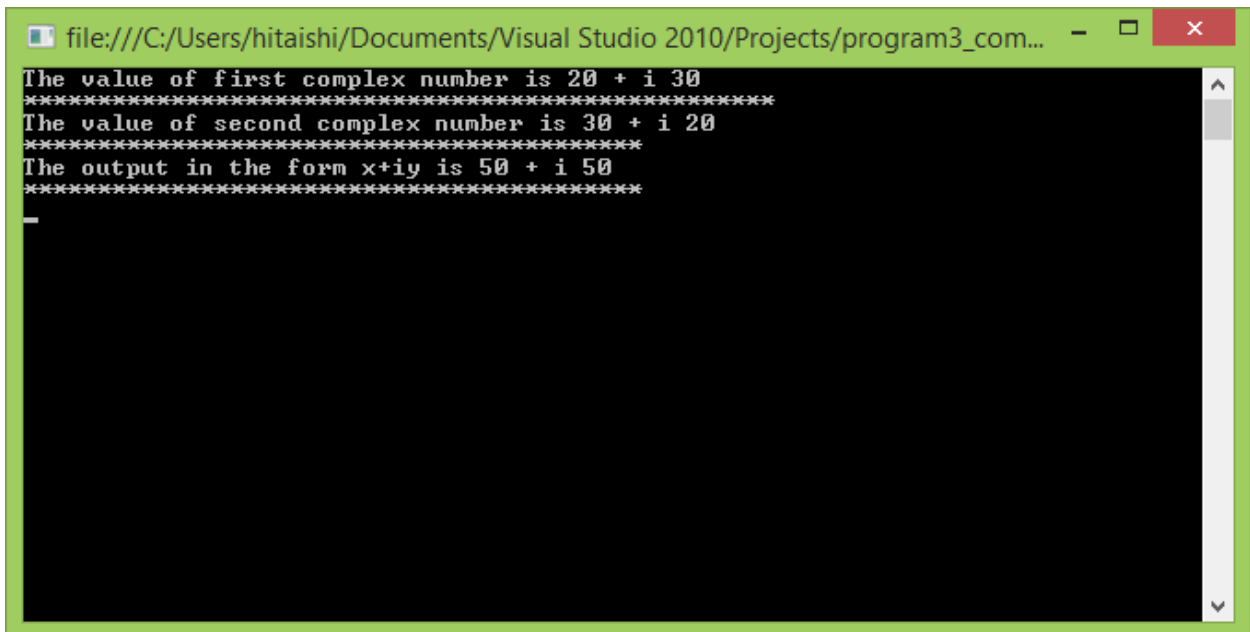
```csharp
Console.WriteLine("The value of first complex number is {0} + i {1}", v1.X,
v1.Y); program3 v2 = new program3 (30, 20);
Console.WriteLine("*********************************************
*"); Console.WriteLine("The value of second complex number is {0} + i {1}",
v2.X, v2.Y); program3 v3 = v1 + v2;
Console.WriteLine("***************************************");
Console.WriteLine ("The output in the form x+iy is {0} + i {1}",v3.X,v3.Y);
Console.WriteLine("***************************************");
Console.ReadLine();

        }
    }
}
```

```
file:///C:/Users/hitaishi/Documents/Visual Studio 2010/Projects/program3_com...

The value of first complex number is 20 + i 30
*********************************************
The value of second complex number is 30 + i 20
***************************************
The output in the form x+iy is 50 + i 50
***************************************
```

8. PROGRAM- 4

4. Write a Program in C# to find the sum of each row of given jagged array of 3 inner arrays.

A jagged array is an array whose elements are arrays. The elements of a jagged array can be of different dimensions and sizes. A jagged array is sometimes called an "array of arrays."

The following is a declaration of a single-dimensional array that has three elements, each of which is a single-dimensional array of integers:

```
int[][] jaggedArray = new int[3][];
```

Before using jaggedArray, its elements must be
initialized. jaggedArray[0] = new int[5];
jaggedArray[1] = new
int[4]; jaggedArray[2] =
new int[2];

Each of the elements is a single-dimensional array of integers. The first element is an array of 5 integers, the second is an array of 4 integers, and the third is an array of 2 integers. It is also possible to use initializers to fill the array elements with values, in which case the array size is not needed.

For example:
jaggedArray[0] = new int[] { 1, 3, 5, 7, 9 };
jaggedArray[1] = new int[] { 0, 2, 4,
6 }; jaggedArray[2] = new int[] { 11,
22 };
This way also the array can be

initialized: int[][] jaggedArray2 =

new int[][]

```
{
  new int[] {1,3,5,7,9},
  new int[] {0,2,4,6},
  new int[] {11,22}
};
```

```csharp
//Write a Program in C# to find the sum of each row of given jagged array of 3 inner
arrays. using System;
namespace labproject
{    class program4
    {
        public static void Main()
        { int[][] jag;
            int i, j, var, sum = 0;
            Console.WriteLine("Enter the number of
            rows"); int row =
            int.Parse(Console.ReadLine());
            jag = new
            int[row][]; for (i =
            0; i < row; i++)
            {
            Console.WriteLine("Enter the number of elements in row {0} :", i +
            1); var = int.Parse(Console.ReadLine());
             jag[i] = new int[var];
              Console.WriteLine(" Enter the {0} Values ",
              var); for (j = 0; j < var; j++)
              {
                  jag[i][j] = int.Parse(Console.ReadLine());
              }
              Console.WriteLine();
            }
              Console.WriteLine("jag[{0}][]:\t",
              row); Console.WriteLine(" \n");
            for (i = 0; i < row; i++)
            {
                System.Console.Write("Element({0}):
                ", i); sum = 0;
                for (j = 0; j < jag[i].Length; j++)
              {
                  Console.Write(" " +
                  jag[i][j]); sum = sum +
                  jag[i][j];
              }
              Console.WriteLine(" \n");
              Console.WriteLine("sum is " +
              sum); Console.WriteLine(" \n");
            }
              Console.ReadLine();
        }
    }
}
```
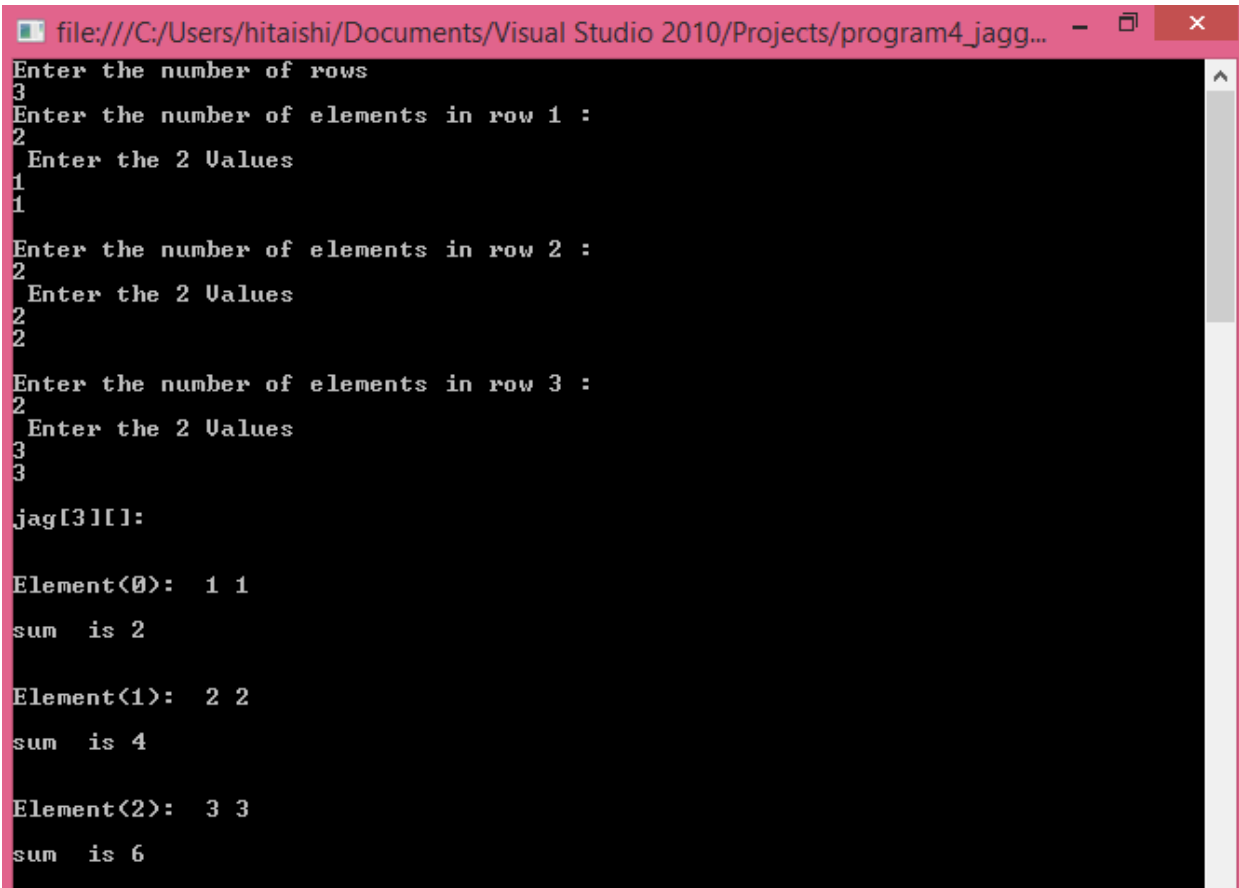
file:///C:/Users/hitaishi/Documents/Visual Studio 2010/Projects/program4_jagg...

```
Enter the number of rows
3
Enter the number of elements in row 1 :
2
 Enter the 2 Values
1
1

Enter the number of elements in row 2 :
2
 Enter the 2 Values
2
2

Enter the number of elements in row 3 :
2
 Enter the 2 Values
3
3

jag[3][]:

Element(0):  1 1

sum  is 2

Element(1):  2 2

sum  is 4

Element(2):  3 3

sum  is 6
```

9. PROGRAM- 5

5. **Write a Program in C# to demonstrate Array Out of Bound Exception using Try , Catch and Finally blocks.**

The .NET platform provides a standard technique to send and trap runtime errors: structured exception handling (SEH). Developers now have a unified approach to error handling, which is common to all languages targeting the .NET universe. The syntax used to throw and catch exceptions across assemblies and machine boundaries is identical. Another bonus of .NET exceptions is the fact that rather than receiving a cryptic numerical value that identifies the problem at hand, exceptions are objects that contain a human-readable description of the problem, as well as a detailed snapshot of the call stack that triggered the exception in the first place.

This involves the use of four interrelated entities:
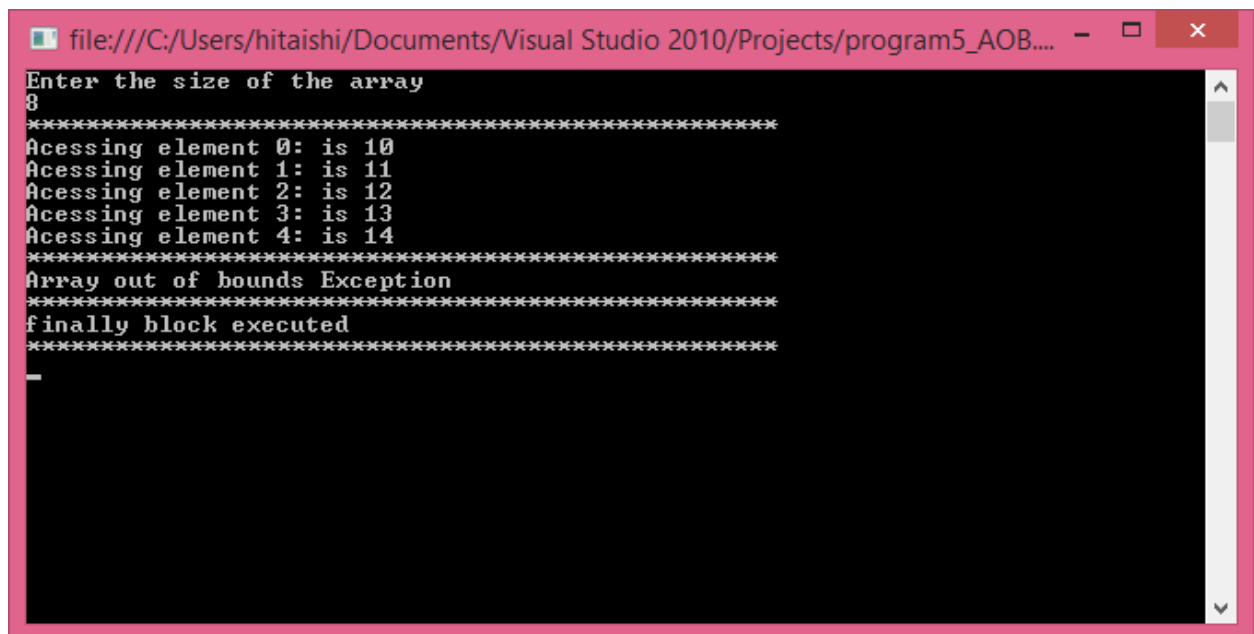
 • A class type that represents the details of the exception that occurred
 • A member that throws an instance of the exception class to the caller
 • A block of code on the caller's side that invokes the exception-prone member
 • A block of code on the caller's side that will process (or catch) the exception should it occur

```csharp
//Write a Program in C# to demonstrate Array Out of Bound Exception using Try , Catch
and Finally blocks.

using System;
namespace
labproject
{    class program5
   {
   public static void Main()
     { int[] array = new
       int[5]{10,11,12,13,14}; int i;
         Console.WriteLine("Enter the size of the
         array"); int row =
         int.Parse(Console.ReadLine());
         Console.WriteLine("***********************************************
         "); try
         { for (i = 0; i <= row; i++)
           {
           Console.WriteLine("Acessing element {0}: is {1}", i,
           array[i]); array[i] = i;
           }
         }
         catch (IndexOutOfRangeException e)
         {
             //runs this instead of crashing the program
```

```csharp
            Console.WriteLine("********************************************")
; Console.WriteLine("Array out of bounds Exception");
Console.WriteLine("*********************************************");
}
 finally
 {
     Console.WriteLine("finally block executed");
 }
Console.WriteLine("*********************************************");
 Console.ReadLine();
 }


 }
}
```

10.   PROGRAM- 6

6.   Write a Program to Demonstrate Use of Virtual and override key words in C# with a simple program.

Overriding method is basically run time polymorphism in C#. When a subclass contains a method with the same name and signature as in the supper class then it is called as method overriding.

Virtual method is a method in a base class which the child class can reuse it or redefine and customize its behavior to be appropriate to its functionality. The virtual keyword is used to modify a method, property, indexer or event declaration, and allow it to be overridden in a derived class. For example, this method can be overridden by any class that inherits it:
public virtual double Area() { return x * y; }

The implementation of a virtual member can be changed by an overriding member in a derived class. · When a virtual method is invoked, the run-time type of the object is checked for an overriding member. · The overriding member in the most derived class is called, which might be the original member, if no derived class has overridden the member. · By default, methods are non- virtual. One cannot override a non-virtual method. · You cannot use the virtual modifier with the static, abstract, private or override modifiers.
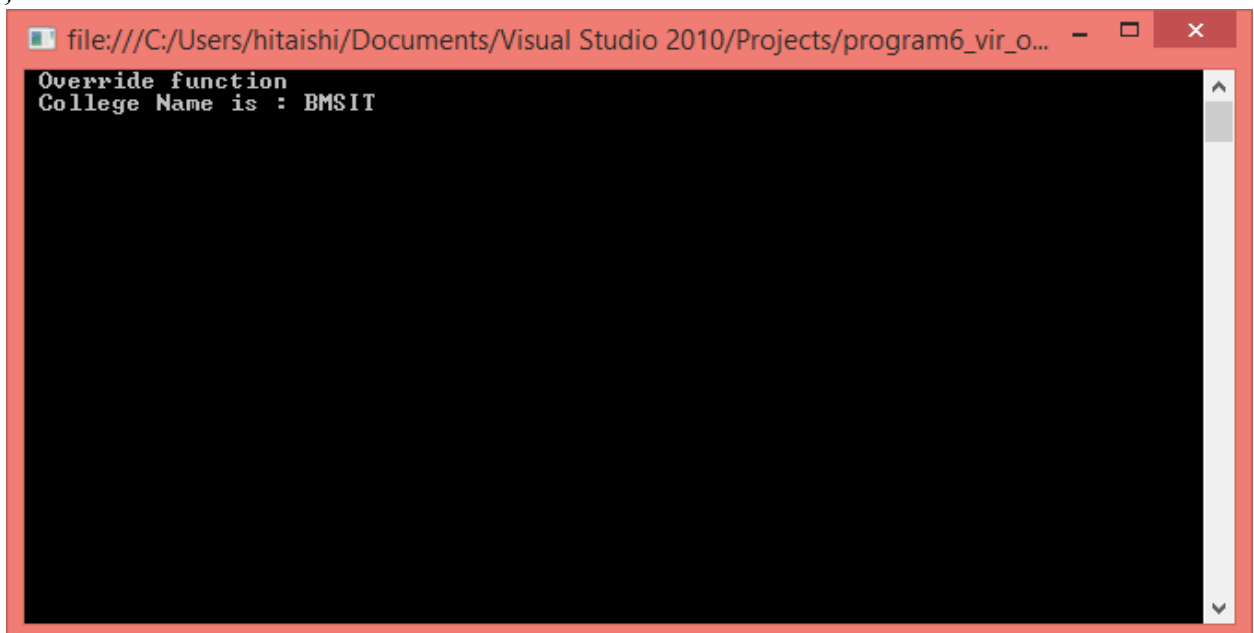
A virtual inherited property can be overridden in a derived class by including a property declaration that uses the override modifier.

The override modifier is required to extend or modify the abstract or virtual implementation of an inherited method, property, indexer, or event. · An override method provides a new implementation of a member inherited from a base class. · The method overridden by an override declaration is known as the overridden base method. · The overridden base method must have the same signature as the override method.

```csharp
//Write a Program to Demonstrate Use of Virtual and override key words in C# with a
simple program.

using System;
namespace
labproject
{
    class BaseClass
    {   public virtual string city()
        { Console.WriteLine(" Inside Virtual
            function"); return "BMSCE";
        }
    }
    class DerivedClass : BaseClass
    { public override string city()
        { Console.WriteLine(" Override function");
```

```csharp
            return "BMSIT";
        }
    }
    class Program6
    {
        public static void Main()
        {
            DerivedClass o = new
            DerivedClass(); string city =
            o.city();
            Console.WriteLine(" College Name is : {0}",
            city); Console.ReadLine();
        }
    }
}
```



```
Override function
College Name is : BMSIT
```

11. PROGRAM- 7

7. Write a Program in C# to create and implement a Delegate for any two arithmetic operations

Delegates:- C# delegates are similar to pointers to functions, in C or
C++. A delegate is a reference type variable that holds the reference to a
method. The reference can be changed at runtime.
*   Delegates are especially used for implementing events and the call-back methods.
*   All delegates are implicitly derived from the **System.Delegate**class.
*   Declaring Delegates:- Delegate declaration determines the methods that can be
    referenced by the delegate.
*   A delegate can refer to a method, which has the same signature as that of the
    delegate. For example, consider a delegate −
        public delegate int MyDelegate (string s);
*   The preceding delegate can be used to reference any method that
    has a single *string* parameter and returns an *int* type variable.
Syntax for delegate declaration is −
*   delegate <return type> <delegate-name>

```csharp
//C# program to create & implement a delegates for any two arithmetic
operations. using System;
namespace labproject
{
        public delegate int deli(int n,int
        m); public class pgm9
        {
                public int sum(int a,int b)
                {
                        return a + b;
                }
                public int diff(int a,int b)
                {
    return a - b;
 }
 public int mul(int a, int b)
 {
    return a * b;
 }
 public int div(int a, int b)
 {
    return a / b;
 }
    }
```
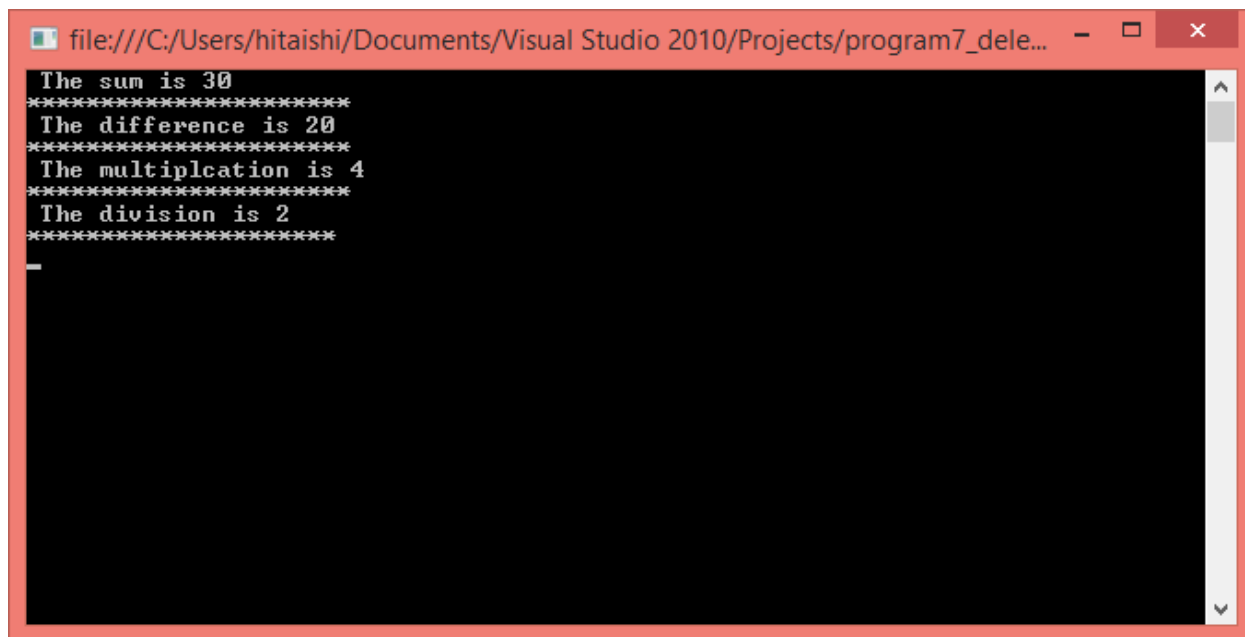
```csharp
public class program9
{
        public static void Main()
        {
        pgm9 p1 = new
        pgm9 (); deli d =
        p1.sum;
int i=d(10,20);

Console.Write(" The sum is " + i );
Console.WriteLine("\n********************");
        pgm9 p2=new
        pgm9(); deli
        f=p2.diff;
        int j=f(40,20);
Console.Write(" The difference is "
+ j);
Console.WriteLine("\n********************");
pgm9 p3 = new pgm9();

deli g =
p3.mul; int
k = g(2, 2);

Console.Write(" The multiplcation is " + k);
Console.WriteLine("\n********************");
pgm9 p4 = new pgm9();
deli h =
p4.div; int l
= h(4,2);

Console.Write(" The division is " + l);
Console.WriteLine("\n********************");
Console.ReadLine();
        }
    }
}
```

```
file:///C:/Users/hitaishi/Documents/Visual Studio 2010/Projects/program7_dele...

 The sum is 30
***********************
 The difference is 20
***********************
 The multiplcation is 4
***********************
 The division is 2
***********************

_
```

12. PROGRAM- 8

8.   Write a Program in C# to demonstrate abstract class and abstract methods in C#.
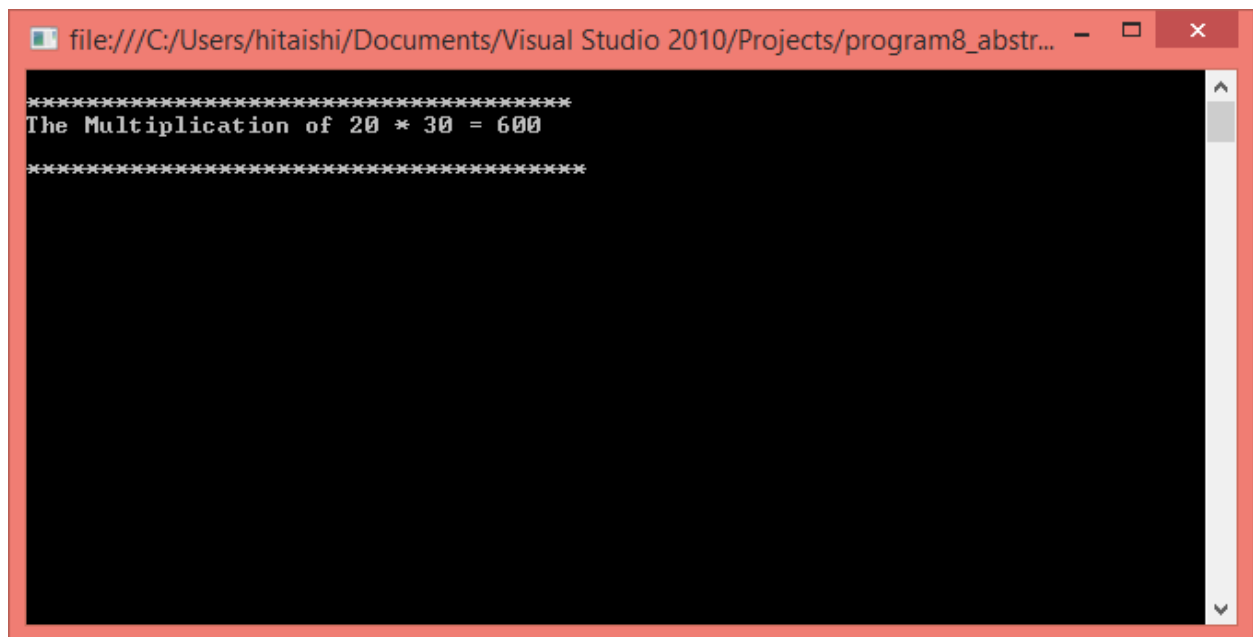
Abstract classes are one of the essential behaviors provided by .NET. Commonly, you would like to make classes that only represent base classes, and don't want anyone to create objects of these class types. Here we make use of abstract classes to implement such functionality in C# using the modifier 'abstract'. An abstract class means that, no object of this class can be instantiated, but can make derivations of this. An abstract class can contain either abstract or non-abstract methods. Abstract members do not have any implementation in the abstract class, but the same has to be provided in its derived class.

```csharp
//A program to demonstrate abstract class and abstract methods in
C#. using System;
namespace labproject
{
    public abstract class program
    {
        public abstract int mul(int a, int b);
    }
    public class demo : program
    {
        public override int mul(int a, int b)
        {
            return a * b;
        }
    }
    public class main
    {
        public static void Main()
        {
            demo d = new
            demo(); int j =
            d.mul(20, 30);
            Console.WriteLine("\n**********************************");
            Console.WriteLine("The Multiplication of {0} * {1} = {2}", 20, 30, j);
            Console.WriteLine("\n**********************************");
            Console.ReadLine();
        }
    }
}
```

```
file:///C:/Users/hitaishi/Documents/Visual Studio 2010/Projects/program8_abstr...

************************************************
The Multiplication of 20 * 30 = 600

************************************************
```

13. PROGRAM- 9

9.   Write a program to Set & Get the Name & Age of a person using Properties of C# to illustrate the use of different properties in C#.

In contrast to traditional accessor and mutator methods, .NET languages prefer to enforce encapsulation using properties, which simulate publicly accessible points of data. Rather than requiring the user to call two different methods to get and set the state data, the user is able to call what appears to be a public field. A C# property is composed using a get block (accessor) and set block (mutator).

Properties enable a class to expose a public way of getting and setting values, while hiding implementation or verification code. The get property accessor is used to return the property value, and a set accessor is used to assign a new value. These accessors can have different access levels. The value keyword is used to define the value being assigned by the set indexer. · Properties that do not implement a set method are read only. · The code block for the get accessor is executed when the property is read; the code block for the set accessor is executed when the property is assigned a new value. · A property without a set accessor is considered read-only. A property without a get accessor is considered write-only. A property with both accessors is read-write.

```csharp
//Write a program to Set & Get the Name & Age of a person using Properties of C# to illustrate the use of different
properties in C#.
using System;
namespace
labprogram
{
    class person
    {
        private string code = "N.A";
        private string name = "not
known"; private int age = 0;

        // Declare a Code property of type
        string: public string Code
        {
            get
            {
                return code;
            }
            set
            {
                code = value;
            }
        }
    }
```

```csharp
// Declare a Name property of type
string: public string Name
{
    get
    {
        return name;
    }
    set
    {
        name = value;
    }
}

// Declare a Age property of type
int: public int Age
{
    get
    {
        return age;
    }
    set
    {
        age = value;
    }
}
public override string ToString()
{
    return "Code = " + Code + ", Name = " + Name + ", Age = " + Age;
}
}

class ExampleDemo
{
    public static void Main()
    {

        // Create a new person object:
        person s = new person();
        // Setting code, name and the age of the
        person s.Code = "001";
        s.Name =
        "geetha"; s.Age
        = 9;
        Console.WriteLine("*************************************************")
        ;
```
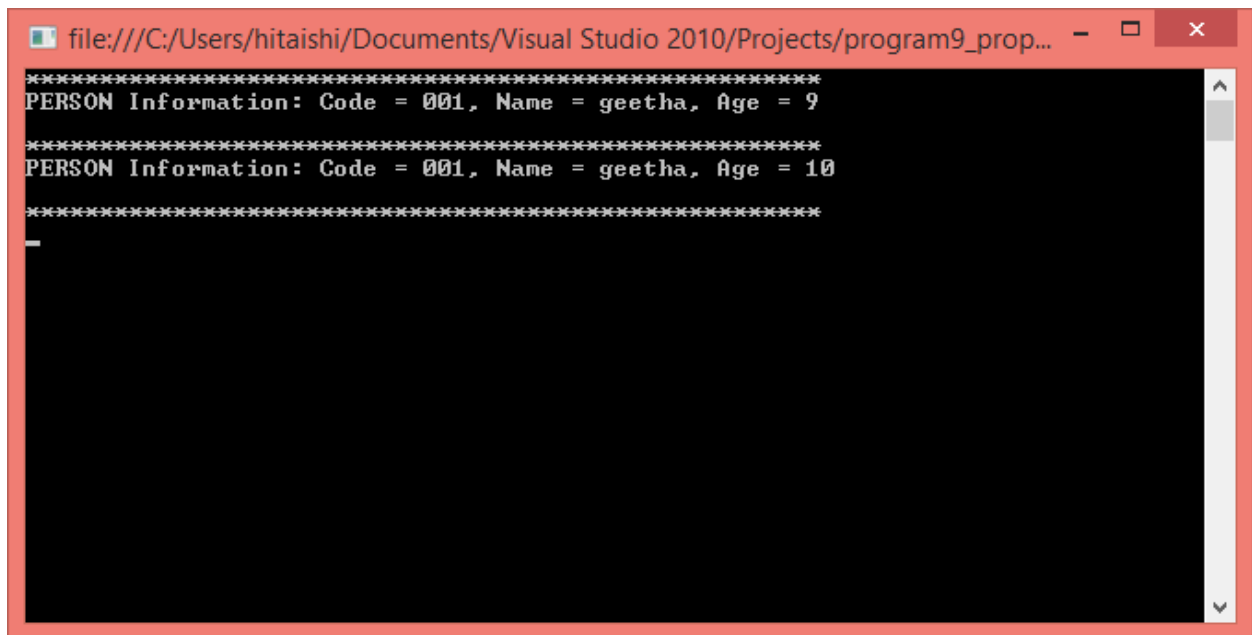
```
Console.WriteLine("PERSON Information: {0} \n", s);

//let us increase
age s.Age += 1;
Console.WriteLine("*********************************************************")
; Console.WriteLine("PERSON Information: {0}\n", s);
Console.WriteLine("*********************************************************")
; Console.ReadKey();
        }
    }
}
```

```
file:///C:/Users/hitaishi/Documents/Visual Studio 2010/Projects/program9_prop...

********************************************************
PERSON Information: Code = 001, Name = geetha, Age = 9

********************************************************
PERSON Information: Code = 001, Name = geetha, Age = 10

********************************************************
_
```

14. PROGRAM- 10

10. Write a Program in C# Demonstrate arrays of interface types (for runtime polymorphism).

Interface is a reference type and it contains only abstract members. Has only signature. Can implement any number of interfaces in a single derived class. An interface can inherit multiple inheritances. Few existing interfaces are ICloneable, IComparable, etc

Interfaces describe a group of related functionalities that can belong to any class or struct. · When a class or struct is said to inherit an interface, it means that the class or struct provides an mplementation for all of the members defined by the interface. Classes and structs can inherit from interfaces in a manner similar to how classes can inherit a base class or struct, with two exceptions:

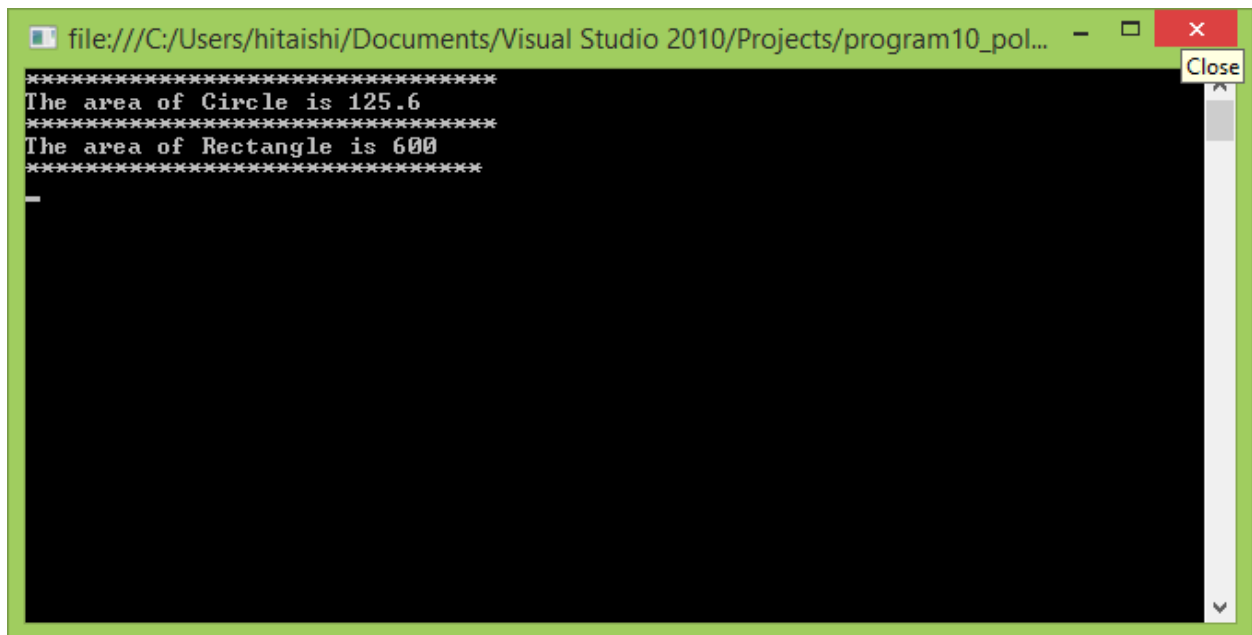1. A class or struct can inherit more than one interface.

2. When a class or struct inherits an interface, it inherits only the method names and signatures, because the interface itself contains no implementations. · An interface contains only the signatures of methods, delegates or events. The implementation of the methods is done in the class that implements the interface. An interface cannot contain fields. · Interfaces members are automatically public.

```csharp
/ Write a Program in C# Demonstrate arrays of interface types (for runtime
polymorphism). using System;
namespace labproject
{
    public interface shape
    {
        void area();
    }
    public class circle : shape
    {
        public void area()
        {
            Console.WriteLine("*****************************");
            Console.WriteLine("The area of Circle is " + 2 * 3.14 * 20);
            Console.WriteLine("*****************************");
        }
    }
    public class rect : shape
    {
        public void area()
        {
            Console.WriteLine("The area of Rectangle is " + 20 * 30);
            Console.WriteLine("****************************");
        }
    }
```

```csharp
public class program10
{
    public static void Main()
    {
        shape[] s = new
        shape[2]; s[0] = new
        circle();
        s[1] = new rect();
        s[0].area();
        s[1].area();
        Console.ReadLi
        ne();
    }
}
}
```

file:///C:/Users/hitaishi/Documents/Visual Studio 2010/Projects/program10_pol...

```
*********************************
The area of Circle is 125.6
*********************************
The area of Rectangle is 600
*********************************
```

PART-B

15. PROGRAM- 1

1. Consider the Database db_EMS (Employee Management System) consisting of the following tables
   :tbl_Designations (IdDesignation: int, Designation: string)
   tbl_EmployeeDetails(IdEmployee: int, EmployeeName: string, ContactNumber: string, IdDesignation: int, IdReportingTo: int)

   Develop a suitable window application using C#.NET having following options.
   1. Enter new Employee details with designation & Reporting Manager.
   2.   Display all the Project Leaders (In a Grid) reporting to selected Project Managers (In a Combo box).
   3. Display all the Engineers (In a Grid) reporting to selected Project Leader (In a Combo box).
   4.   Display all the Employees (In a Grid) with their reporting Manager (No Value

   for PM). NOTE: tbl_Designation is a static table containing the following Rows in it.

1	Project Manager
2	Project Leader
3	Engineer

```csharp
//Form1
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Threading.Tasks;
using System.Data.SqlClient;

namespace EmployeeManagementSystem
{
    public partial class Form1 : Form
    {
        SqlConnection con = new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\Documents\EMPDB.mdf;Integrat
```

```
ed Security=True;Connect Timeout=30;User Instance=True");
    public Form1()
    {
        InitializeComponent();
```

```csharp
        }

        private void button2_Click(object sender, EventArgs e)
        {
            eid.Clear(); ename.Clear(); cno.Clear(); idd.Clear(); idr.Clear();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            con.Open();
            SqlDataAdapter sda = new SqlDataAdapter("insert into empdetails values (" +
eid.Text + ",'" + ename.Text + "','" + cno.Text + "','" + idd.Text + "','" + idr.Text + "')", con);

            sda.SelectCommand.ExecuteNonQuery()
            ; con.Close();
            MessageBox.Show("Details were added to the database.", "",
MessageBoxButtons.OK, MessageBoxIcon.Information);

        }

        private void button4_Click(object sender, EventArgs e)
        {
            con.Open();
            SqlDataAdapter sda = new SqlDataAdapter("select * from empdetails",
            con); DataTable dt = new DataTable();
            sda.Fill(dt);
            dataGridView1.DataSource =
            dt; con.Close();

        }

        private void button3_Click(object sender, EventArgs e)
        {
            empdes form = new
            empdes(); form.Show();
        }

        }
}


//Empdeta
ils using
System;
using System.Collections.Generic;
using
System.ComponentModel;
using System.Data;
using System.Drawing;
```

```csharp
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Threading.Tasks;

namespace EmployeeManagementSystem
{
    public partial class empdes : Form
    {
        SqlConnection con = new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\Documents\EMPDB.mdf;Integrat
ed Security=True;Connect Timeout=30;User Instance=True");
        public empdes()
        {
            InitializeComponent();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Form1 form = new
            Form1(); form.Show();
        }

        private void empdes_Load(object sender, EventArgs e)
        {

        }

        private void textBox1_TextChanged(object sender, EventArgs e)
        {
            try
            {
                if (comboBox1.Text == "Project Manager")
                {
                    con.Open();
                    SqlDataAdapter sda = new SqlDataAdapter("select empid,empname,cno,idd,idr
from empdetails where idd like'" + textBox1.Text + "%'", con);
                    DataTable dt = new
                    DataTable(); sda.Fill(dt);
                    dataGridView1.DataSource =
                    dt; con.Close();
                }
                else if (comboBox1.Text == "Project Leader")
                {
                    con.Open();
```

```
        SqlDataAdapter sda = new SqlDataAdapter("select empid,empname,cno,idd,idr
from empdetails where idd like'" + textBox1.Text + "%'", con);
        DataTable dt = new
        DataTable(); sda.Fill(dt);
        dataGridView1.DataSource =
        dt; con.Close();
    }

}
catch (Exception ex)
{
    MessageBox.Show("error" + ex);
}

}
```

IDDesignation   Project Manager ▼   2   Empdetails

	empid	empname	cno	idd	idr
▶	3	raj	7777777	2	1
	5	sss	1111111	2	2
*					

IDDesignation   Project Leader ▼   3   Empdetails

	empid	empname	cno	idd	idr
▶	4	sam	666666	3	1
	6	qqqq	7777777	3	2
	7	wwa	666666666	3	1
	30	qqq	99999999	3	1
*					

Employee Designation

Employee Database

16. PROGRAM- 2

2. Consider the Database db_LSA (Lecturer Subject Allocation) consisting of the following tables: tbl_Subjects(IdSubject: int, SubjectCode: string, SubjectName: string) tbl_Lecturers(IdLecturer: int, LecturerName: string, ContactNumber: string) tbl_LecturerSubjects(IdSubject: int, SubjectCode: string, IdLecturer: int) Develop a suitable window application using C#.NET having following options.
   1. Enter new Subject Details.
   2. Enter New Lecturer Details.
   3. Subject Allocation with Lecturer Name in a Combo box and subjects to be allocated in Grid with checkbox Column.
   4. Display all the subjects allocated (In a Grid) to the selected Lecturer (In a Combo Box).

```csharp
//Subject
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Threading.Tasks;
using System.Data.SqlClient;

namespace LecturerSubjectAllocation
{
    public partial class Form1 : Form
    {
        SqlConnection con = new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\documents\visual studio
2010\Projects\LecturerSubjectAllocation\LecturerSubjectAllocation\LSADB.mdf;Integrated
Security=True;Connect Timeout=30;User Instance=True");
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            con.Open();
```

```
        SqlDataAdapter sda = new SqlDataAdapter("insert into SubjectDetails values (" +
        ids.Text
+ ",'" + sc.Text + "','" + sn.Text + "')", con);
        sda.SelectCommand.ExecuteNonQuery()
        ;
```

```csharp
        con.Close();
        MessageBox.Show("Details were added to the database.", "",
MessageBoxButtons.OK, MessageBoxIcon.Information);

    }

    private void button2_Click(object sender, EventArgs e)
    {
        ids.Clear();
        sc.Clear();
        sn.Clear();
    }

    private void button3_Click(object sender, EventArgs e)
    {
        con.Open();
        SqlDataAdapter sda = new SqlDataAdapter("select * from SubjectDetails",
        con); DataTable dt = new DataTable();
        sda.Fill(dt);
        dataGridView1.DataSource =
        dt; con.Close();

    }

    private void button4_Click(object sender, EventArgs e)
    {
        lecturer form1 = new
        lecturer(); form1.Show();
    }

    private void button5_Click(object sender, EventArgs e)
    {
        LecSub form1 = new
        LecSub(); form1.Show();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        // TODO: This line of code loads data into the 'lSADBDataSet1.SubjectDetails' table.
You can move, or remove it, as needed.
        this.subjectDetailsTableAdapter.Fill(this.lSADBDataSet1.SubjectDetails);

    }

    private void button6_Click(object sender, EventArgs e)
    {
```

```csharp
            sub_allocation form = new
            sub_allocation(); form.Show();
        }


    }
}

// Lecturer
form using
System;
using System.Collections.Generic;
using
System.ComponentModel;
using System.Data;
using
System.Drawing;
using System.Linq;
using System.Text;
using
System.Windows.Forms;
using
System.Threading.Tasks;
using
System.Data.SqlClient;

namespace LecturerSubjectAllocation
{

    public partial class lecturer : Form
    {
        SqlConnection con = new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\documents\visual studio
2010\Projects\LecturerSubjectAllocation\LecturerSubjectAllocation\LSADB.mdf;Integrated
Security=True;Connect Timeout=30;User Instance=True");
        public lecturer()
        {
            InitializeComponent();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            idl.Clear();
            ln.Clear();
            cn.Clear();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            con.Open();
            SqlDataAdapter sda = new SqlDataAdapter("select * from Lecturer",
```

```
con); DataTable dt = new DataTable();
sda.Fill(dt);
```

```csharp
        dataGridView1.DataSource =
        dt; con.Close();

    }

    private void button3_Click(object sender, EventArgs e)
    {
        Form1 form = new
        Form1(); form.Show();

    }

    private void button1_Click(object sender, EventArgs e)
    {
        con.Open();
        SqlDataAdapter sda = new SqlDataAdapter("insert into Lecturer values ("+ idl.Text +
",'"+ ln.Text +"','" +cn.Text + "')", con);
        sda.SelectCommand.ExecuteNonQuery()
        ; con.Close();
        MessageBox.Show("Details were added to the database.", "",
MessageBoxButtons.OK, MessageBoxIcon.Information);


    }

    private void button5_Click(object sender, EventArgs e)
    {
        LecSub form1 = new
        LecSub(); form1.Show();
    }

    private void lecturer_Load(object sender, EventArgs e)
    {
        // TODO: This line of code loads data into the 'lSADBDataSet14.Lecturer' table. You
can move, or remove it, as needed.
        this.lecturerTableAdapter.Fill(this.lSADBDataSet14.Lecturer);


    }

    private void button6_Click(object sender, EventArgs e)
    {
        sub_allocation form = new
        sub_allocation(); form.Show();
    }
```

```
    }
}

//Subject
allocation using
System;
using System.Collections.Generic;
using
System.ComponentModel;
using System.Data;
using
System.Drawing;
using System.Linq;
using System.Text;
using
System.Windows.Forms;
using
System.Threading.Tasks;
using
System.Data.SqlClient;
using System.Data.Sql;

namespace LecturerSubjectAllocation
{
    public partial class LecSub : Form
    {
        SqlConnection con = new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\documents\visual studio
2010\Projects\LecturerSubjectAllocation\LecturerSubjectAllocation\LSADB.mdf;Integrated
Security=True;Connect Timeout=30;User Instance=True");
        public LecSub()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            lecturer form1 = new
            lecturer(); form1.Show();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Form1 form = new
            Form1(); form.Show();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            LecSub form = new
            LecSub(); form.Show();
```

}

```csharp
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    /*
    try
    {
        string str = "SELECT S.SubName,S.SubCode from Lecturer L, SubjectDetails S,LecSub LS ";
        //where LecName like'" + textBox1.Text + "%'";
        con.Open();
        SqlCommand cmdsql = new SqlCommand(str, con);
        SqlDataReader rs;
        rs = cmdsql.ExecuteRea
```

```
der();while                                        (rs.Read()){

                comboBox1.Items.Add(rs[0]);
                comboBox1.SelectedItem =
                rs[0];


             }
             con.Close();
          }
          catch (Exception ex)
          {
             MessageBox.Show(ex.Message);
          }

          */
       }


       private void button4_Click(object sender, EventArgs e)
       {
          sub_allocation form = new
          sub_allocation(); form.Show();
       }

       private void Display_Click(object sender, EventArgs e)
       {
         con.Open();
          SqlDataAdapter sda = new SqlDataAdapter("SELECT SubID,SubName,SubCode
from SubjectDetails",con);
          DataTable dt = new
          DataTable(); sda.Fill(dt);
        // for
          dataGridView1.DataSource =
          dt; con.Close();
```

```
        }

        private void LecSub_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'lSADBDataSet16.SubjectDetails' table.
You can move, or remove it, as needed.
            this.subjectDetailsTableAdapter.Fill(this.lSADBDataSet16.SubjectDetails);
            // TODO: This line of code loads data into the 'lSADBDataSet15.Lecturer' table. You
can move, or remove it, as needed.
            this.lecturerTableAdapter.Fill(this.lSADBDataSet15.Lecturer);
        }
}

using System;
using
System.Collections.Generic;
using
System.ComponentModel;
using System.Data;
using
System.Drawing;
using System.Linq;
using System.Text;
using
System.Windows.Forms;
using
System.Data.SqlClient;

namespace LecturerSubjectAllocation
{
    public partial class sub_allocation : Form
    {
        SqlConnection con = new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\documents\visual studio
2010\Projects\LecturerSubjectAllocation\LecturerSubjectAllocation\LSADB.mdf;Integrated
Security=True;Connect Timeout=30;User Instance=True");
        public sub_allocation()
        {
            InitializeComponent();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Form1 form = new
            Form1(); form.Show();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            lecturer form = new
            lecturer(); form.Show();
```

```csharp
        }

    private void button4_Click(object sender, EventArgs e)
    {
       LecSub form = new
       LecSub(); form.Show();
    }
     DataTable dt;
    private void sub_allocation_Load(object sender, EventArgs e)
    {
       // TODO: This line of code loads data into the 'lSADBDataSet17.SubjectDetails' table.
You can move, or remove it, as needed.
       this.subjectDetailsTableAdapter.Fill(this.lSADBDataSet17.SubjectDetails);
       // TODO: This line of code loads data into the 'lSADBDataSet10.Lecturer' table. You
can move, or remove it, as needed.
       this.lecturerTableAdapter.Fill(this.lSADBDataSet10.Lecturer);

    }

  /*    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {
      /*
      con.Open(
      ); try
      {
        if (comboBox1.SelectedItem == "LecName")
        {
           con.Open();
           SqlDataAdapter sda = new SqlDataAdapter("SELECT S.SubName,S.SubCode from
Lecturer L,SubjectDetails S", con);
           DataTable dt = new
           DataTable(); sda.Fill(dt);
           dataGridView1.DataSource =
           dt; con.Close();
        }
      }
      catch (Exception ex)
      {
        MessageBox.Show("error" + ex);
      }
      con.Close();

      /*string query = "select s.SubCode,s.SubName,l.LecName from SubjectDetails s
,Lecturer l where LecName ='" + comboBox1.Text + "'";
      SqlCommand cmd =new
      SqlCommand(query,con); con.Open();
      cmd.ExecuteNonQuery();
```

```
            DataTable dt = new DataTable();
            SqlDataAdapter da = new
            SqlDataAdapter(cmd); da.Fill(dt);
            dataGridView1.DataSource =
            dt; con.Close();
              } */


    /*private void button1_Click(object sender, EventArgs e)
    {
        SqlCommand
        sc1;
        SqlDataReade
        r rd;
        string sql = "select s.SubID,s.SubCode,s.SubName,l.LecName from
SubjectDetails s,Lecturer l where LecName ='" + comboBox1.Text + "';";
        {
            try
            {
                con.Open();
                sc1 = new SqlCommand(sql,
                con); rd =
                sc1.ExecuteReader();
                // if (rd.Read())
                // while(rd.Read())
                //
                while(rd.HasRo
                ws)
                {
                    while(rd.Read())
                    {
                        textBox1.Text = rd.GetValue(2).ToString();

                    }
                    rd.NextResult();
                }
                sc1.Dispos
                e();
                con.Close(
                );
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString());
            }
        }


    }
```

```
/* SqlCommand sc1;
```

```csharp
        SqlDataReader rd;
        string sql = "select s.SubID,s.SubCode,s.SubName,l.LecName from
SubjectDetails s,Lecturer l where LecName ='" + comboBox1.Text + "';";
        {
          try
          {
            con.Open();
            sc1 = new SqlCommand(sql,
            con); rd =
            sc1.ExecuteReader();
            while (rd.Read())
            {
              // dataGridView1.DataSource = rd.GetValue(4).ToString();
            }
            sc1.Dispos
            e();
            con.Close(
            );
          }
          catch (Exception ex)
          {
            MessageBox.Show(ex.ToString());
          }
        }
*/

        //con.Open();
        //try
        //{
          //    if (comboBox1.SelectedItem == "LecName")
          //{
        //    con.Open();
        //    SqlDataAdapter sda = new SqlDataAdapter("select
s.SubName,l.LecName,l1.SubID from SubjectDetails s,Lecturer l,LecSub l1 where
s.SubID=l1.SUbID and LecName ='" + comboBox1.Text + "'", con);
        // SqlDataAdapter sda = new SqlDataAdapter("select
s.SubID,s.SubCode,s.SubName,l.LecName from SubjectDetails s,Lecturer l where LecName
='" + comboBox1.Text + "'", con);
          //  if (sda != null &&
          // foreach(dt.Rows
          // {
        // dt = new DataTable();
         // sda.Fill(dt);
        // dataGridView1.DataSource = dt;
          //}
          // else

          // {
        //    MessageBox.Show("no data");
```

```
//}


//}
//con.Close();
//}
//catch (Exception ex)
//{
// MessageBox.Show("error" + ex);
//}


// SqlDataAdapter sda = new SqlDataAdapter("insert into LecSub
(SubID,SubCode,LecID) select s.SubID,s.SubCode,l.LecID from SubjectDetails s,lecturer l",
con);


// SqlDataAdapter sda = new SqlDataAdapter("select SubjectDetails.SubName as
'SubjectDetails',Lecturer.LecName as 'Lecturer' from SubjectDetails inner join Lecturer,LecSub
where SubjectDetails.SubID=LecSub.SubID and Lecturer.LecID=LecSub.LecID and LecName
='"
+ comboBox1.Text + "'", con);
// SqlDataAdapter sda = new SqlDataAdapter("select
s.SubName,l.LecName,l1.SubID from SubjectDetails s,Lecturer l,LecSub l1 where
s.SubID=l1.SUbID and LecName ='" + comboBox1.Text + "'", con);
//      SqlDataAdapter sda = new SqlDataAdapter("select
SubjectDetails.SubName as 'SubjectDetails',Lecturer.LecName as 'Lecturer' from
SubjectDetails ,Lecturer,LecSub where SubjectDetails.SubID=LecSub.SubID and
Lecturer.LecID=LecSub.LecID and LecName ='" + comboBox1.Text + "'", con);
//      DataTable dt = new DataTable();
//    sda.Fill(dt);
// dataGridView1.DataSource = dt;
// con.Close();


// if (comboBox1.Text == "LecName")
// string var;
// var = comboBox1.Text;
/*
   con.Open();
   SqlDataAdapter sda = new SqlDataAdapter("select s.SubName,l.LecName
from SubjectDetails s,Lecturer l where LecName ='" + comboBox1.Text + "'", con);
   // SqlDataAdapter sda = new SqlDataAdapter("select s.SubName,l.LecName
from SubjectDetails s,Lecturer l ,LecSub ls where LecName ='" + comboBox1.Text +
"'", con);
   // SqlDataAdapter sda = new SqlDataAdapter("select
SubjectDetails.SubName,Lecturer.LecName from Lecturer inner join LecSub on LecSub.LecID =
Lecture.LecID where LecName ='" + comboBox1.Text + "'", con);
       DataTable dt = new
       DataTable(); sda.Fill(dt);
//    if (comboBox1.SelectedItem == "LecName")
// {
```

```csharp
//        while (sda.)
/* for (int i = 0; i > 5; i++ )
  {
    dataGridView1.DataSource = dt;
  }
     */


// }

//else
// {
// MessageBox.Show("plesea ");
//}
//     con.Close();

// }
// }

    private void button5_Click(object sender, EventArgs e)
    {
       con.Open();
    // SqlDataAdapter sda = new SqlDataAdapter("select distinct
s.SubName,l1.SubID,l.LecName from SubjectDetails s,Lecturer l,LecSub l1 where
s.SubID=l1.SUbID and LecName ='" + comboBox1.Text + "' and SubName = '" +
textBox1.Text + "'", con);
        SqlDataAdapter sda = new SqlDataAdapter("select distinct
s.SubName,l1.SubID,l.LecName from SubjectDetails s,Lecturer l,LecSub l1 where
s.SubID=l1.SUbID and LecName ='" + comboBox1.Text + "' and SubName = '" +
comboBox2.Text + "'", con);
        DataTable dt = new
        DataTable(); sda.Fill(dt);
        dataGridView1.DataSource =
        dt; con.Close();
    }


 /*    private void textBox1_TextChanged(object sender, EventArgs e)
    {
       DataView dv = new DataView(dt);
       dv.RowFilter = "" + comboBox1.Text + " like '%"+ textBox1.Text
       +"%'"; dataGridView1.DataSource = dv;
    }
    */

}
}
```
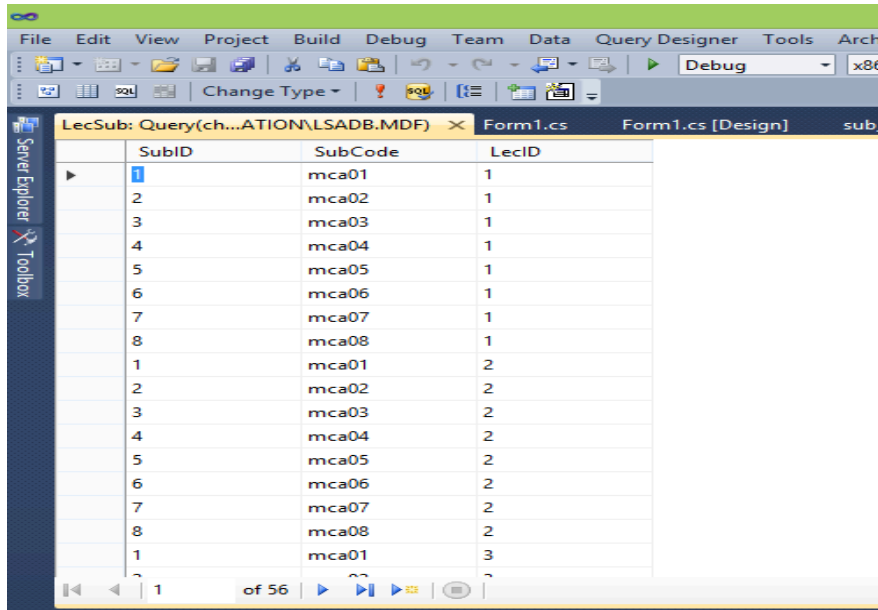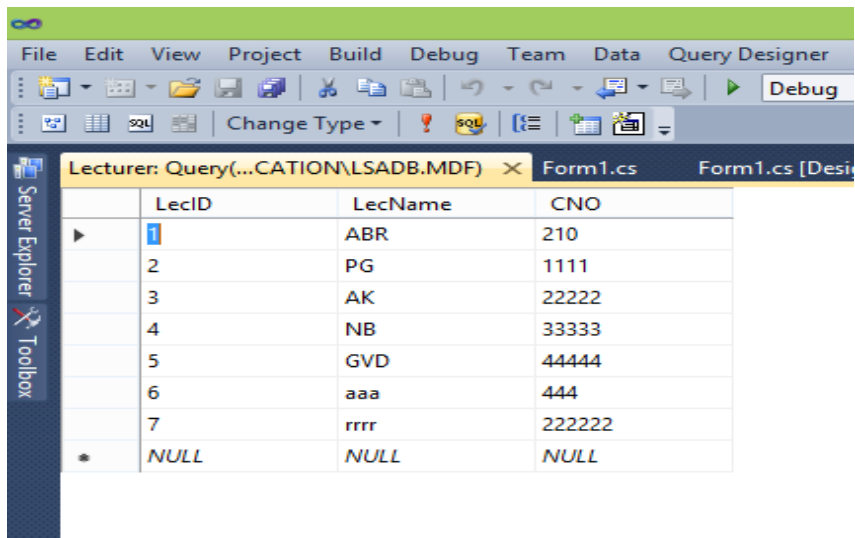
17. PROGRAM- 3

3.   Consider the database db_VSS (Vehicle Service Station) consisting of the following tables: tbl_VehicleTypes(IdVehicleType: int, VehicleType: string, ServiceCharge: int) tbl_ServiceDetails(IdService: int, VehicleNumber: string, ServiceDetails: string, IdVehicleType: int)

Develop a suitable window application using C#.NET having following options.

1. Enter new Service Details for the Selected Vehicle Type (In a Combo Box).

2. Update the Existing Service Charges to Database.

3.   Total Service Charges Collected for the Selected Vehicle (In a Combo box) with total amount displayed in a text box.

NOTE: tbl_VehicleType is a static table containing the following Rows in it.

1	Two Wheeler	500
2	Four Wheeler	1000
3	Three Wheeler	700

```csharp
//Service
Details using
System;
using
System.Collections.Generic;
using
System.ComponentModel;
using System.Data;
using
System.Drawing;
using System.Linq;
using System.Text;
using
System.Windows.Forms;
using
System.Data.SqlClient;
namespace vss
{
    public partial class Form1 : Form
    {
        SqlConnection con = new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=c:\users\hitaishi\documents\visual
studio 2010\Projects\vss\vss\VSSDB.mdf;Integrated Security=True;Connect
Timeout=30;User Instance=True");
        public Form1()
        {
            InitializeComponent();
        }
        private void button2_Click(object sender, EventArgs e)
        {
            sid.Clear();
```

```
vno.Clear();
sd.Clear();
vt.Clear();
```

```csharp
        }
    private void button1_Click(object sender, EventArgs e)
    {
        con.Open();
        SqlDataAdapter sda = new SqlDataAdapter("insert into serviceDetails
(ServiceID,VehicleNumber,ServiceDetails,VehicleID)values (" + sid.Text + ",'" + vno.Text +
"','" + sd.Text + "','" + vt.Text + "')", con);
        sda.SelectCommand.ExecuteNonQuery()
        ; con.Close();
        MessageBox.Show("Details were added to the database.", "",
MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    private void Display_Click(object sender, EventArgs e)
    {
        con.Open();
        SqlDataAdapter sda = new SqlDataAdapter("select * from serviceDetails",
        con); DataTable dt = new DataTable();
        sda.Fill(dt);
        dataGridView1.DataSource =
        dt; con.Close();
    }

    private void button3_Click(object sender, EventArgs e)
    {
        UPDate_Form form = new
        UPDate_Form(); form.Show();
    }

    private void button4_Click(object sender, EventArgs e)
    {
        Total_service form = new
        Total_service(); form.Show();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        // TODO: This line of code loads data into the 'vSSDBDataSet1.serviceDetails' table.
You can move, or remove it, as needed.
        this.serviceDetailsTableAdapter.Fill(this.vSSDBDataSet1.serviceDetails);
        // TODO: This line of code loads data into the 'vSSDBDataSet.vehicletypes' table. You
can move, or remove it, as needed.
        this.vehicletypesTableAdapter.Fill(this.vSSDBDataSet.vehicletypes);


    }
```

```csharp
    }
}

// Update Form
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace vss
{
    public partial class UPDate_Form : Form
    {
        SqlConnection con = new SqlConnection(@"Data Source=.\SQLEXPRESS;AttachDbFilename=c:\users\hitaishi\documents\visual studio 2010\Projects\vss\vss\VSSDB.mdf;Integrated Security=True;Connect Timeout=30;User Instance=True");
        private SqlCommand cmd;
        public UPDate_Form()
        {
            InitializeComponent();
        }


        private void button3_Click(object sender, EventArgs e)
        {
            sid.Clear();
            sc.Clear();

        }


        private void button2_Click(object sender, EventArgs e)
        {

            con.Open();
            cmd = new SqlCommand("UPDATE serviceDetails SET servicecharges=@a1 where ServiceID=@a2",con);
            cmd.Parameters.Add("a1",sc.Text);
            cmd.Parameters.Add("a2",sid.Text);
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("Details were added to the database.","",MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
```

```csharp
        private void button1_Click(object sender, EventArgs e)
        {
            con.Open();
            SqlDataAdapter sda = new SqlDataAdapter("select * from serviceDetails",
            con); DataTable dt = new DataTable();
            sda.Fill(dt);
            dataGridView1.DataSource =
            dt; con.Close();
        }

        private void UPDate_Form_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'vSSDBDataSet3.serviceDetails' table.
You can move, or remove it, as needed.
            this.serviceDetailsTableAdapter.Fill(this.vSSDBDataSet3.serviceDetails);
            // TODO: This line of code loads data into the 'vSSDBDataSet2.vehicletypes' table. You
can move, or remove it, as needed.
            this.vehicletypesTableAdapter.Fill(this.vSSDBDataSet2.vehicletypes);

        }

        private void button4_Click(object sender, EventArgs e)
        {
            Form1 form = new
            Form1(); form.Show();
        }

        private void button5_Click(object sender, EventArgs e)
        {
            Total_service form = new
            Total_service(); form.Show();
        }
    }
}

//Total Service
using System;
using
System.Collections.Generic;
using
System.ComponentModel;
using System.Data;
using
System.Drawing;
using System.Linq;
using System.Text;
using
System.Windows.Forms;
using
System.Data.SqlClient;
```

```
namespace vss
```

```csharp
{
    public partial class Total_service : Form
    {
        SqlConnection con = new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=c:\users\hitaishi\documents\visual
studio 2010\Projects\vss\vss\VSSDB.mdf;Integrated Security=True;Connect
Timeout=30;User Instance=True");

        public Total_service()
        {
            InitializeComponent();
        }


        private void Total_service_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'vSSDBDataSet4.serviceDetails' table.
You can move, or remove it, as needed.
            this.serviceDetailsTableAdapter.Fill(this.vSSDBDataSet4.serviceDetails);

        }

        private void button2_Click(object sender, EventArgs e)
        {
            Form1 form = new
            Form1(); form.Show();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            UPDate_Form form = new
            UPDate_Form(); form.Show();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            SqlCommandscl;S
```

"'.".
; ;

```
q                                        e
l                                        r
D                                        r
a                                        d
t                                        ;
a                                        string sql = "select * from serviceDetails where
R                                        VehicleNumber ='" + comboBox1.Text +
e
a                                        {
d                                            try
                                             {
             con.Open();
             sc1 = new SqlCommand(sql,
             con); rd =
             sc1.ExecuteReader();
```

```csharp
                while (rd.Read())
                 {
                   textBox1.Text = rd.GetValue(4).ToString();
                 }
               sc1.Dispos
               e();
               con.Close(
               );
            }catch (Exception ex)
            {

               MessageBox.Show(ex.ToString());
            }
          }
       }
     }
}
```


	ServiceID	VehicleNumber	ServiceDetails	VehicleID	servicecharges
▶	1	ka1	break	1	1000
	2	ka2	handel	2	1000
	3	ka3	wheel	3	700
	4	ka4	break	1	500
	5	ka5	wheel	2	1000
	6	ka6	cluch	3	700
*	NULL	NULL	NULL	NULL	NULL

	VehicleID	VehicleTYPE	servicecharges
▶	1	Two Wheeler	500
	2	Four Wheeler	1000
	3	Three Wheeler	700
*	NULL	NULL	NULL

18. PROGRAM- 4

4.   Develop a web application using C#.NET and ASP.NET for the Postal System Management. The master page should contain the hyper links for adding Area Details, Postman details, Letter distributions and View Letters.

Consider the database db_PSM (Postal System Management) consisting of the following tables: tbl_AreaDetails(IdArea: int, AreaName: string) tbl_PostmanDetails(IdPostman: int, PostmanName: string, ContactNumber: string, IdArea: int) tbl_AreaLetters(IdLetter: int, LetterAddress: string, IdArea: int)

Develop the suitable content pages for the above created 4 hyper links with the following details:

1. Enter New Area Details
2. Enter New Postman Details with the Area he/she is in-charge of (display Area in a Combo box)
3. Enter all the Letters distributed to the selected Area (display Area in a Combo box)
4.   Display all the Letter addresses (In a Grid) to be distributed by the selected Postman (In a Combo box)

Site master form

```aspx
<%@ Master Language="C#" AutoEventWireup="true"
CodeBehind="Site.master.cs" Inherits="postalSystemManagement.SiteMaster" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head runat="server">
    <title></title>
    <link href="~/Styles/Site.css" rel="stylesheet" type="text/css" />
    <asp:ContentPlaceHolder ID="HeadContent" runat="server">
    </asp:ContentPlaceHolder>
    <style type="text/css">
        .menu
        {
            background-color: #99FF33;
        }
        .menu
        {
            color: #33CCFF;
        }
        .style3
        {
            text-align: justify;
        }
```

```
    </style>
  </head>
<body bgcolor="#ccffff">
    <form runat="server">
    <div class="page" align="center">
```

```
<div class="header">
    <div class="clear hideSkiplink">
    <div class="loginDisplay"

        style="font-family: 'Times New Roman', Times, serif; font-size: xx-large;
font-weight: bold; font-style: normal; font-variant: normal; text-transform: capitalize; color:
#FF0000; width: 707px; margin-left: 0px;">
            Postal System Management</div>
        <asp:Menu ID="NavigationMenu" runat="server" CssClass="menu"
            EnableViewState="false" IncludeStyleBlock="false"
            Orientation="Horizontal"
            onmenuitemclick="NavigationMenu_MenuItemClick">
            <Items>
                <asp:MenuItem NavigateUrl="~/home.aspx" Text="Home"/>
                <asp:MenuItem NavigateUrl="~/area details.aspx" Text="Area_Details"/>
                <asp:MenuItem NavigateUrl="~/postman details.aspx" Text="Postman_Details"/>
                <asp:MenuItem NavigateUrl="~/letter
distributions.aspx" Text="Letter_Distribution"/>
                <asp:MenuItem NavigateUrl="~/View_letters.aspx" Text="View_Letters"/>

            </Items>
        </asp:Menu>
    </div>
    </div>
    <div class="main">
        <asp:ContentPlaceHolder ID="MainContent" runat="server">
        </asp:ContentPlaceHolder>
    <p>
        <img alt="Image result for postal system
        management" src="https://encrypted-
tbn0.gstatic.com/images?q=tbn:ANd9GcReAyQPUIghrsBZ5k4GAE3DZ_eDuBpozFiHaMMQ
LsQ Cqw513OYY" /></p>
        <div class="style3">
    Postal system Management   is the software that conatin the Hyper link for adding sub
deatils of the information about the area details,Postman deatils,letters distribution and view
letters. It is possible for any person to send a letter to any address in the same country or abroad
,in the expectation that it will be conveyed according to certain established standards of
regularity,speed and security.


    </div>
        </div>
    </form>
</body>
</html>
```

Home page

```aspx
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="home.aspx.cs" Inherits="postalSystemManagement.home" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <style type="text/css">
        .style1
        {
            text-align: justify;
        }
    </style>
</head>
<body bgcolor="#66ffcc" style="width: 705px">
    <form id="form1" runat="server">
    <div style="font-family: 'Times New Roman', Times, serif; font-size: xx-large; font-weight:
100; font-style: normal; font-variant: small-caps; text-transform: capitalize; color: #000000;
width: 721px;">

        WEL COME TO POSTAL SYSTEM MANAGEMENT</div>
    </form>
    <p style="height: 184px; width: 347px">
     <img
src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcROvKeNR_6moB-
tCEYZyd-puDqCBA8ub-JvjWSjWpL-aVVa5PkJ" alt="Image result for postal system"/></p>
        <div class="style1">
        The Department of Posts (DoP), trading as India Post, is a government-operated postal
system in India under the Department of Posts, which is part of the Ministry of
Communications of the Government of India. Generally called "the post office" in India, it is
the most widely distributed postal system in the world.[2]

It is involved in delivering mails, remitting money by money orders, accepting deposits under
Small Savings Schemes, providing life insurance cover under Postal Life Insurance (PLI) and
Rural Postal Life Insurance (RPLI) and providing retail services like bill collection, sale of
forms, etc.
The DoP also acts as an agent for Government of India in discharging other services for
citizens such as old age pension payments and Mahatma Gandhi National Rural
Employment Guarantee Scheme (MGNREGS) wage disbursement. With 155,015 post
offices, India Post has the most widely distributed postal network in the world.
    </div>
</body>
    </html>
```

Area Details

```csharp
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.Data;

namespace postalSystemManagement
{
    public partial class area_details : System.Web.UI.Page
    {
        SqlConnection con = new SqlConnection(@" Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\Documents\PSMDB.mdf;Integrat
ed Security=True;Connect Timeout=30;User Instance=True ");
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button2_Click(object sender, EventArgs e)
        {
            ida.Text="";
            an.Text="";
        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            con.Open();
            SqlDataAdapter sda = new SqlDataAdapter("insert into
AreaDetails(IDArea,AreaName) values('"+ida.Text+"','"+an.Text+"')",con);
            sda.SelectCommand.ExecuteNonQuery()
            ; con.Close();
            Response.Write("<script>alert('Inserted into the AreaDetails
Database.Thank You')</script>");
        }
```

```csharp
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.Data;

namespace postalSystemManagement
{
    public partial class postman_details : System.Web.UI.Page
    {
        SqlConnection con = new SqlConnection(@" Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\Documents\PSMDB.mdf;Integrat
ed Security=True;Connect Timeout=30;User Instance=True ");
        protected void Page_Load(object sender, EventArgs e)
        {

        }
        protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {

          // if (DropDownList1.Text == "AreaName")
            {
                con.Open();
                SqlDataAdapter sda = new SqlDataAdapter("insert into
PostmanDetails(IDPostman,PostmanName,ContactNumber,IDArea) values('" + PID.Text +
"','" + PN.Text + "','" + CN.Text + "','" + AID.Text + "')", con);
                sda.SelectCommand.ExecuteNonQuer
                y(); con.Close();
                Response.Write("<script>alert('Inserted into the PostmanDetails
Database.Thank You')</script>");
                // Response.Write("Inserted Successfully...")

            }
        }


        protected void Button2_Click(object sender, EventArgs e)
        {
            PID.Text = "";
```

```
        PN.Text = "";
        CN.Text = "";
        AID.Text = "";
    }


    protected void Button3_Click1(object sender, EventArgs e)
    {

        con.Open();
        SqlDataAdapter sda = new SqlDataAdapter("select * from PostmanDetails",
        con); DataTable dt = new DataTable();
        sda.Fill(dt);
        // GridView1.DataSource
        = dt;
        GridView1.DataBind();
        con.Close();
    }
  }
}
```

Site master page
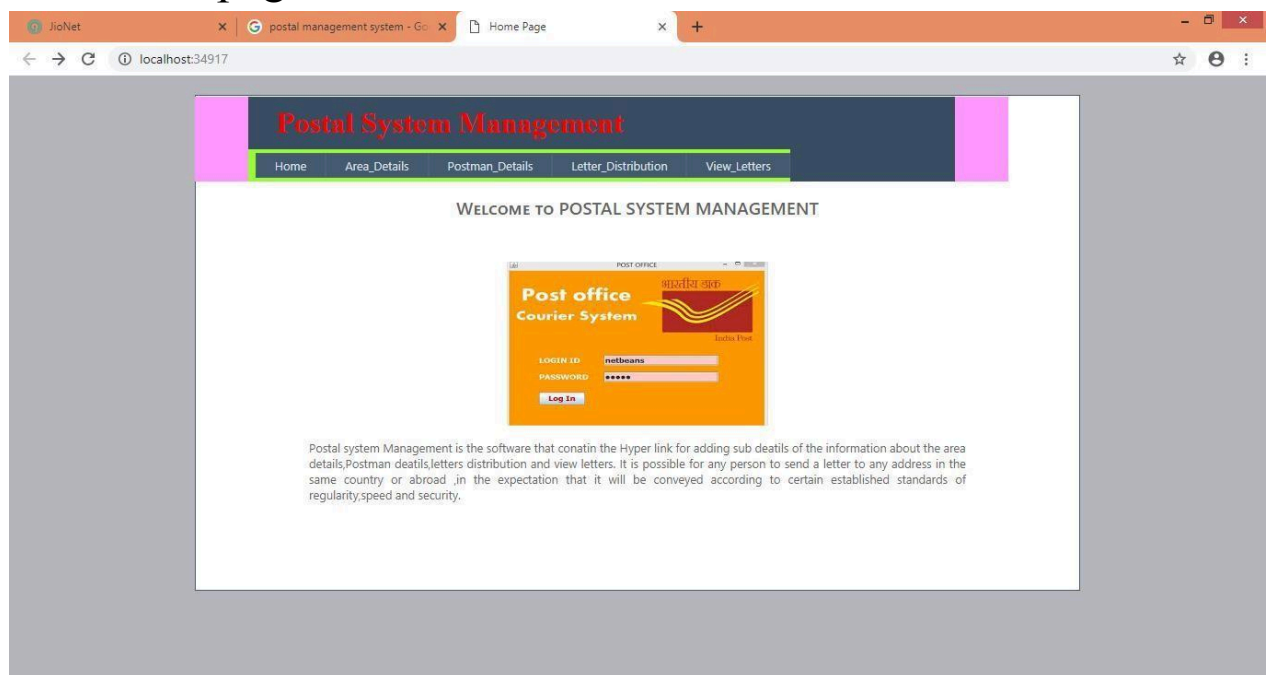
Home page
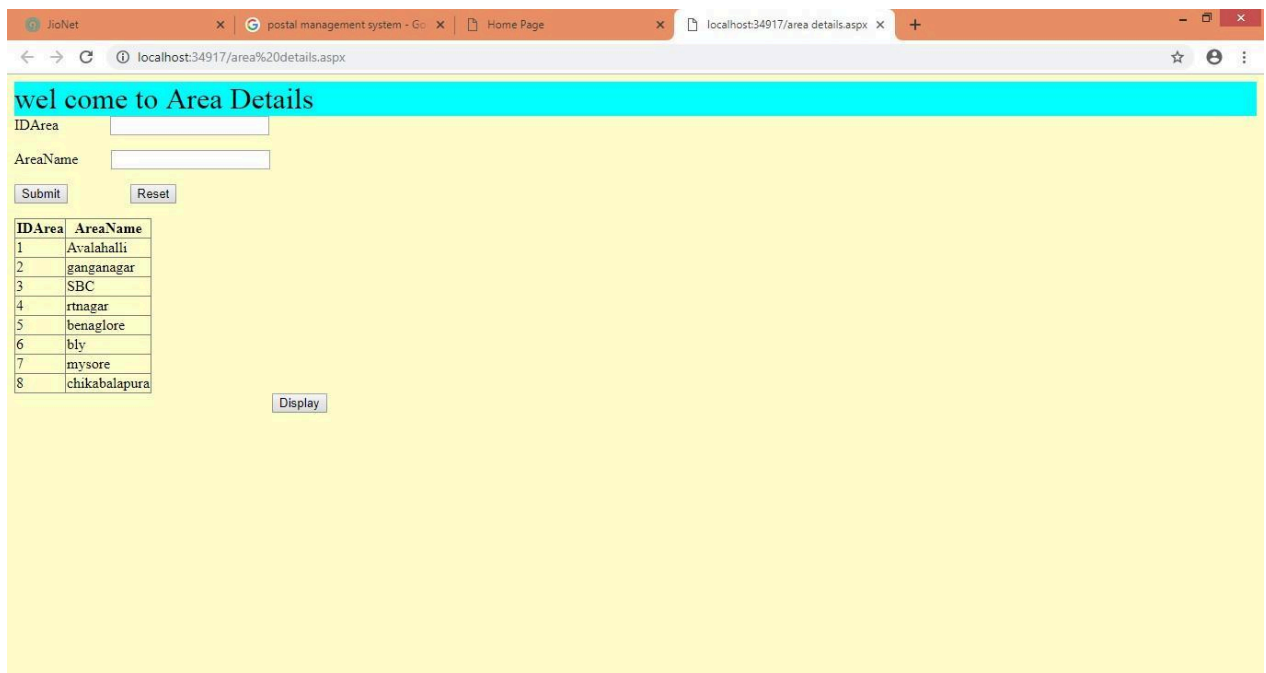
wel come to postman details

Area   Avalahalli ▼

PostmanID

Postman Name

ContactNumber

AreaID

Submit   Reset

IDPostman	PostmanName	ContactNumber	IDArea
1	chinni	999999	1
2	qqq	2222	1
3	abc	11111	3
4	zzzz	44444	4

Display

wel come to letter distributions

Area   Avalahalli ▼

LetterID

Letter Address

AreaID

Submit   Reset

IDletter	LetterAddress	IDArea
1	sbc	1
2	ynk	1
3	rtnagar	1
4	avahalli	4
5	mysore	1

Display
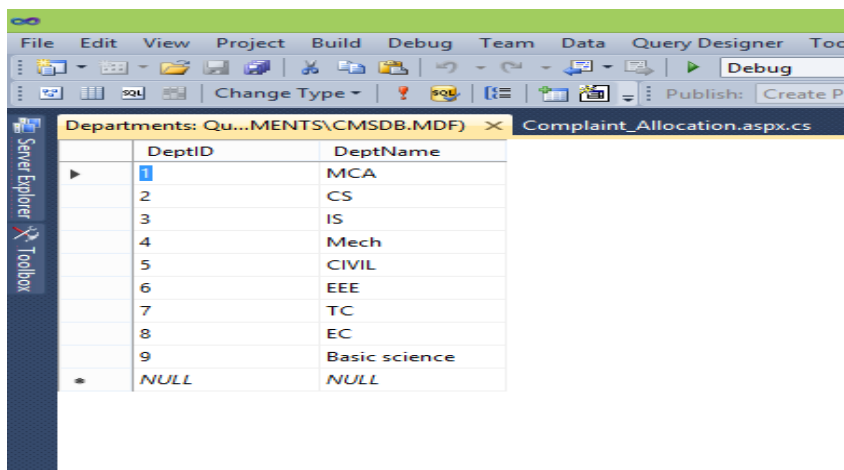
19. PROGRAM- 4

5. Develop a web application using C#.NET and ASP.NET for the Complaint Management System. The master page should contain the hyperlinks for Add Engineer, Complaint Registration, and Complaint Allocation View Complaints.

    Consider the database db_CMS (Complaint Management System) consisting of the following tables: tbl_Departments(IdDepartment: int, DepartmentName: string), tbl_Engineers(IdEngineer: int, EngineerName: string, ContactNumber: string,IdDepartment: int) tbl_RegisteredComplaints(IdComplaint: int, ComplaintDescription: string) tbl_DepartmentComplaints(IdDepartment: int, IdComplaint: int)
    Develop the suitable content pages for the above created 4 hyper links with the following details:
    1. Enter New Engineers belonging to the selected department (displayed in a combo box)
    2. Register a new Complaint with a submit button.
    3.   View all registered complaints & allocate to the corresponding department (displayed in a combo box)
    4.   Display all the Complaints (In a Grid) to be handled by the selected Engineer (In a Combo box) NOTE: Consider the table tbl_Departments as a static table containing some pre-entered departments, which are displayed in all the remaining modules.

Site master

```
<%@ Master Language="C#" AutoEventWireup="true"
CodeBehind="Site.master.cs" Inherits="CMS.SiteMaster" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head runat="server">
```

```
<title></title>
<link href="~/Styles/Site.css" rel="stylesheet" type="text/css" />
<asp:ContentPlaceHolder ID="HeadContent" runat="server">
</asp:ContentPlaceHolder>
<style type="text/css">
    .style1
    {
        clear: both;
        text-align: center;
    }
</style>
</head>
<body>
    <form runat="server">
    <div class="page">
    <div class="header">
        <div class="title">
            <h1>
                WEL COME TO COMPLAINT MANAGEMENT SYSTEM
            </h1>
        </div>
        <div class="loginDisplay">
            <asp:LoginView ID="HeadLoginView" runat="server" EnableViewState="false">
                <AnonymousTemplate>
                    [ <a href="~/Account/Login.aspx" ID="HeadLoginStatus" runat="server">Log
In</a> ]                                         </AnonymousTemplate>
                                                <LoggedInTemplate>
                                                    Welcome <span
                                                    class="bold"><asp:LoginName
                                                    ID="HeadLoginName"
runat="server" /></span>!
                    [ <asp:LoginStatus ID="HeadLoginStatus" runat="server"
LogoutAction="Redirect" LogoutText="Log Out" LogoutPageUrl="~/"/> ]
                </LoggedInTemplate>
            </asp:LoginView>
        </div>
        <div class="clear hideSkiplink">
            <asp:Menu ID="NavigationMenu" runat="server" CssClass="menu"
                EnableViewState="false" IncludeStyleBlock="false"
                Orientation="Horizontal"
                onmenuitemclick="NavigationMenu_MenuItemClick">
                <Items>
                    <asp:MenuItem NavigateUrl="~/Default.aspx" Text="Home"/>
                    <asp:MenuItem NavigateUrl="~/ADD_Engineer.aspx" Text="Engineer"/>
                    <asp:MenuItem
NavigateUrl="~/Complaint_Registration.aspx"
Text="Complaint_Registration"/>
                    <asp:MenuItem
NavigateUrl="~/Complaint_Allocation.aspx"
Text="Complaint_Allocation"/>
```

```
                <asp:MenuItem
NavigateUrl="~/View_Complaints.aspx"
Text="View_Complaints"/>
            </Items>
        </asp:Menu>
      </div>
    </div>
    <div class="main" style="background-color: #00FFFF">
        <asp:ContentPlaceHolder ID="MainContent" runat="server"/>

        <br />
        <img alt="Image result for complaint management system"
            src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTsn5
            pO-
YPowqTOhJNfbezyv7TH0FWxhj0iGXJSueta4TwLUfcg" /><br />
        <br />
        <br />

Managing complaints is favorable business practice for ensuring excellent customer service in
a company. By having complaint management systems in place, businesses can use the
information that is captured to make process improvements.
        </div>
        <div class="style1">
        </div>
    </div>
    <div class="footer">

    </div>
    </form>
    <p>
         </p>
</body>
</html>
```

Default :-

```
<%@ Page Title="Home Page" Language="C#"
MasterPageFile="~/Site.master" AutoEventWireup="true"
    CodeBehind="Default.aspx.cs" Inherits="CMS._Default" %>

<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent">
</asp:Content>
<asp:Content ID="BodyContent" runat="server" ContentPlaceHolderID="MainContent">
    <h2>
         </h2>

    <p>
```

According to Metric Stream, complaint management is the process of how organizations handle, manage, respond to and report customer complaints. Systems are put into place to track and trend the data that is captured by complaint management processes.
 </p>
</asp:Content>

//Engineers

```csharp
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.Data;

namespace CMS
{
    public partial class ADD_Engineer : System.Web.UI.Page
    {
        SqlConnection con =new SqlConnection(@"Data Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\Documents\CMSDB.mdf;Integrated Security=True;Connect Timeout=30;User Instance=True");
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button2_Click(object sender, EventArgs e)
        {
            EngID.Text = "";
            EngName.Text =
            "";
            ContactNo.Text =
            ""; DeptID.Text =
            "";
        }

        protected void Button1_Click(object sender, EventArgs e)
        {

            con.Open();
            SqlDataAdapter sda = new SqlDataAdapter("insert into Engineers(EngID,EngName,ContactNo,DeptID) values('" + EngID.Text + "','" + EngName.Text + "','" + ContactNo.Text + "','" + DeptID.Text + "')", con);
            sda.SelectCommand.ExecuteNonQuery()
            ; con.Close();
```

```
        Response.Write("<script>alert('Inserted into the Engineers
Database.Thank You')</script>");


    }

    protected void Button3_Click(object sender, EventArgs e)
    {
        con.Open();
        SqlDataAdapter sda = new SqlDataAdapter("select * from Engineers",
        con); DataTable dt = new DataTable();
        sda.Fill(dt);
        // GridView1.DataSource
        = dt;
        GridView1.DataBind();
        con.Close();
    }

  }
}
```

Complaint Registration

```
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.Data;

namespace CMS
{
    public partial class Complaint_Registration : System.Web.UI.Page
    {
        SqlConnection con = new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\Documents\CMSDB.mdf;Integrat
ed Security=True;Connect Timeout=30;User Instance=True");
        protected void Page_Load(object sender, EventArgs e)
        {

        }
    protected void Button1_Click(object sender, EventArgs e)
    {
        con.Open();
        SqlDataAdapter sda = new SqlDataAdapter("insert into
RegCom(ComID,ComDesp) values('" + CID.Text + "','" + CD.Text + "')",con);
        sda.SelectCommand.ExecuteNonQuery()
        ; con.Close();
```

```csharp
            Response.Write("<script>alert('Inserted into the Registred Complaints
Database.Thank You')</script>");
        }
        protected void Button2_Click(object sender, EventArgs e)
        {
            CID.Text = "";
            CD.Text = "";
        }
        protected void Button3_Click(object sender, EventArgs e)
        {
            con.Open();
            SqlDataAdapter sda = new SqlDataAdapter("select * from RegCom",
            con); DataTable dt = new DataTable();
            sda.Fill(dt);
            // GridView1.DataSource
            = dt;
            GridView1.DataBind();
            con.Close();
        }
    }
}
```

Complaint allocation

```csharp
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.Data;

namespace CMS
{
    public partial class Complaint_Allocation : System.Web.UI.Page
    {
        SqlConnection con =new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\Documents\CMSDB.mdf;Integrat
ed Security=True;Connect Timeout=30;User Instance=True");
        protected void Page_Load(object sender, EventArgs e)
        {

        }


        protected void Button1_Click(object sender, EventArgs e)
        {
```

```csharp
        con.Open();
        SqlDataAdapter sda = new SqlDataAdapter("INSERT INTO
DeptCom(DeptID,ComID) SELECT d.DeptID, r.ComID from Departments d,RegCom
r,Engineers e where d.DeptID=e.DeptID;", con);
        sda.SelectCommand.ExecuteNonQuery()
        ; con.Close();
        Response.Write("<script>alert('Inserted into the Engineers
Database.Thank You')</script>");
    }

    protected void Button2_Click(object sender, EventArgs e)
    {
        con.Open();
        SqlDataAdapter sda = new SqlDataAdapter("select * from DeptCom ",
        con); DataTable dt = new DataTable();
        sda.Fill(dt);
        GridView1.DataBind();
        con.Close();
    }
    }
    }
View
complaints
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.Data;

namespace CMS
{
    public partial class View_Complaints : System.Web.UI.Page
    {
        SqlConnection con = new SqlConnection(@"Data
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\hitaishi\Documents\CMSDB.mdf;Integrat
ed Security=True;Connect Timeout=30;User Instance=True");
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            con.Open();
```

```csharp
            SqlDataAdapter sda = new SqlDataAdapter("select * from RegCom",
            con); DataTable dt = new DataTable();
            sda.Fill(dt);
            // GridView1.DataSource
            = dt;
            GridView1.DataBind();
            con.Close();
        }
    }
}
```

```csharp
            SqlDataAdapter sda = new SqlDataAdapter("select * from RegCom",
            con); DataTable dt = new DataTable();
            sda.Fill(dt);
            // GridView1.DataSource
            = dt;
            GridView1.DataBind();
```

20. VIVA- QUESTIONS

1 What is boxing and Unboxing feature in C#?

2   State the difference between value types and reference types 3 What is an object?

4 What is a class?

5 What is a namespace?

6   Explain some of the important features of Visual studio 7 Explain the anatomy of a basic C# class

8 What is a constructor overloading

9   What are the basic input and output console class? 10 What is reference type? Give examples

11 List the different iteration constructs used in C#

12   List the different control flow constructs used in C# 13 Explain a jagged array

14 Mention the difference between the rectangular array and jagged array 15 What is the difference between for loop and while loop?

16 What is enumeration? Give Example 17 What is a value type? Give examples

18 What are the different functions of array object?

19   What the different function and properties of string object? 20 What is encapsulation

21 What is ReadOnly Field 22 Explain casting

23 What is polymorphism? Explain with an example 24 Which are the 3 pillars of OOPs

25 What is an exception 26 What is debugging? 27 What is an error?

28 What are the different types of Exceptions thrown in .Net 29 Explain Garbage collection in .Net

30 Explain the base class System.Exception

31   What is the difference between Final and Finally 32 What is interfaces?

33 What is the difference between and interface and abstract class 34 Explain IEnumeration

35 Explain IConvertible

36 Explain ICloneable

37 Explain Icomparable
38 What are the built in interfaces

39 What is a metadata?

40   What is call backs, give
example 41 Describe
overloading

42 What is
Delegate? 43
What is events?

44 What are the different types of
Delegates? 45 What is indexers?

46 Explain checked and unchecked keyword in
C# 47 What is operator overloading?

48 What is an array?

49   Give few String methods and
explain 50 Explain 'foreach' loop