Windows Programming

with Practical Using C#

BCA Semester-4	Subject CC-213

More than 30 programs with solutions

COMPUTER WORLD®
PUBLICATION
A Division of Live Education System Pvt. Ltd.

BCA | Semester-4

Windows Programming

with Practical Using C#

Subject Code : CC-213

Prof. Kamesh Raval
MCA [Master in Computer Application]
Som-Lalit Institute of Computer Application, Ahmedabad.

COMPUTER WORLD
PUBLICATION
A Division of Live Education System Pvt. Ltd.

COMPUTER WORLD
43, 5th Floor, SANIDHYA Complex, Nr. M. J. Library,
Opp. Sanyas Ashram, Ashram Road, Ahmedabad-09.
Ph. : +91- 79 26580723, 26580823 | Mobile : 9724011150, 9725022917
URL : www.computerworld.ind.in | e-Mail :info@computerworld.ind.in

Publish By :

COMPUTER World

A Division of Live Education System Pvt. Ltd.

An ISO 9001 : 2008 Certified Company

Kalpesh Patel - kalpesh@computerworld.ind.in

Prepare by : Computer World Research Department

Prof. Kamesh Raval, Ravindra Parmar

Price : ₹ 125/-

ISBN : 978-93-88092-22-7

Book Code : CEBCA118

Edition : 1st

Notice of Rights

No part of this publication may be reproduced, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual, or otherwise, without the prior written permission of Computer World except under the terms of a Computer World license agreement.

Notice of Liability

The information in this courseware title is distributed on an 'as is' basis, without warranty. While very precaution has been taken in the preparation of this course, neither the authors nor Computer World shall have any liability to my person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the instruction contained in this book or by the computer software and hardware products described in it.

Disclaimer

We make a sincere effort to ensure the accuracy of the material described here in however, Computer World makes no warranty, expressed or implied, with respect of the quality, correctness, reliability, accuracy, of freedom from error of this document or the products it describes. Data used in examples and sample data files are intended to be fictional. Any resemblance to real persons or companies is entirely coincidental.

Content of Copyright : COMPUTER WORLD PUBLICATION - http://www.computerworld.ind.in

Contact : 43, 5th Floor, Sandhya Complex, Opp. Capital Comm. Center, Ashram Road, Ahmedabad-380009

Phone : +91-79-26580823 : +91-79-26580723. | Mobile : 97240 11150, 97250 22917

URL : www.computerworld.ind.in | e-Mail : info@computerworld.ind.in, info.computerworld@gmail.com

INDEX

Introduction to .NET 07 to 24

Programs of Unit-1 25 to 60

1. Design interface and implement functionalities for Arithmetic calculator with power, square, log, factorial, square root and clear functionalities.
2. Design interface and implement functionalities for Loan calculator. Take Amount, No of instalments and Rate of interest from the user. Also user can choose Early Pay option through a checkbox. Calculate instalment amount using pmt() function. Do proper validation for inputs taken by the user.
3. Design an application which will have 2 radio buttons. One will convert the Celsius to Fahrenheit and another will convert Fahrenheit to Celsius. Show the appropriate output depends on the user's selection. (Use radio button to take user choice and use textbox to enter value).
4. Design a form having two text boxes, combo box and a label. Make the validation so that user can enter only numbers in both textboxes, if user has entered both numerical values then make the combo box visible. The combo box has options like 'ADD', 'SUB', 'MUL' and 'DIV'. According to user's choice from from combo, result will display in label.
5. Create an application with a textbox in which user can enter a sentence then displays
   1) Number of vowels
   2) Number of spaces
   3) Number of digits
   4) Number of special symbols

   When user press "analysis" button.
6. Design and implement a Tic Tack Toe game (Two Player).
7. Write a program to transfer an item from First List box to Second List box and from Second List box to First.
8. Print multiplication table into List box. For multiplication take value using Numeric up down.
9. Take 3 Radio buttons showing the name of 3 Countries. Load the image of the Flag of the country selected by the user from the given Radio buttons in the Picture box.

10. Take a Timer control which will delay to load MainForm by 10 seconds. Show the progress bar in the wait time. Also use Time control to scroll a label having text "Gujarat University", also take two more labels to show date and time on the tick event of the timer.

Programs of Unit-2 61 to 84

1. Take 3 Scrollbars indicating values of Red, Green and Blue colors from 0 to 255. Take 3 labels to show the values of the scroll bars. Depending upon values selected by the user with the help of scrollbars make a color and shows it in the picture box. Place a button having text 'Apply Color', and on the click event of the button apply that color to the form.
2. Implement Textpad application using Rich textbox. Make menus like File (New, Open Save, SaveAs and Exit), Edit (Cut, Copy, Paste, Undo, Redo), Format (Bold, Italic, Underline, Font, Color) etc. Use all common dialog controls and implement functionalities.
3. Take a Rich Textbox and implement Find, FindNext, Replace and ReplaceAll functionalities.
4. Write a program to Read and Write Text file.
5. Write a program to Read and Write Binary file.
6. Accept no from user and perform following operations using user defined sub routines or functions:
   1) Factorial of number
   2) Odd/even
7. Create MDI form. It must have File menu with option open, Close and Exit. It should also have window menu to arrange the child forms like Tile Horizontal, Tile Vertical, Cascade and Arrange Icons.
8. Create MDI form. It must have File menu with option Open, Close and Exit and one picture box. Allow users to open any picture using open dialog box, that picture should be displayed in the picture box.
9. Write function or subroutine to Find maximize, minimize value from an array and also to sort an array.

Programs of Unit-3 85 to 112

1. Create shape class as abstract class having area as must override function. Derive rectangle, triangle, square class based from this class.
2. Write a program to create class Person. Make at least three properties and one method "show detail" of this class. Now inherit class Student and Faculty from class Person and override method "show detail". Create objects of Student and Faculty class and call show detail function for both objects to show details in appropriate text boxes.
3. Write a program to implement the class Employee. Show Constructor Overloading.
4. Write a program to implement the class Book. Show Method Overloading.
5. Create an Invoice application in which user enters the customer name, description, unit price and quantity for the item ordered, then clicks the add item button. The application calculates the order total by multiplying the unit price by the quantity. And calculates a discount based on the order total. The user can then add another item to the order by using all information.
6. Create a user control called myTextBx, having all the properties of the TextBox. Add Four Properties to it called EnterClr as Color, LeaveClr as Color and MandatoryClr as Color and Mandatory as boolean. Use this control on Windows Application. Object of this control will show EnterClr when it is being focused. On Leave if Mandatory property is TRUE and Textbox is Empty then MandatoryClr has to be applied otherwise LeaveClr is applied to the Textbox.
7. Write an application, which works like a window's explorer, using combo box and list box.
   1. The combo box should display the list of drives.
   2. The list box should display the list of files and directories.

Programs of Unit-4 113 to 148

1. Write a Program to develop a Database Application with ADO.NET with Record Navigation and Add, Delete, Save and Update Facility for Books table (BookID,BookName, Qty, UnitPrice,TotalPrice).
2. Create a login form which will validate user name and password from the database. On validation show MDI form. Show appropriate message if user name and password is incorrect.
3. Create two tables State(StateCode, StateName) and City(CityCode, CityName, StateCode). Design a form which will list all the state in the ComboBox. After

selection of the StateName, another ComboBox should display all the Cities which belong to the state selected by the user in the first ComboBox.

4. Write a program to create a table Emp with the following fields:

Field Name Datatype

Eno Integer

Ename Varchar(20)

Salary Single

Birthdate Date

Designation Varchar(20)

Design a form which perform Select, Insert, Update and Delete operations on the table Emp. On Insert

Eno has been generated automatically.

5. Create below mentioned Crystal Reports for the above application.
   - List all the Employees having salary more than 25000
   - List all the Employees who are above the age 35.

   List all the Employees designation wise.

6. Design a following table in Access. Table name = Book (Bookcode, Booktype, Bookpublisher,

   Bookauther, Bookpage, Bprice). Design Form that Display Above detail. Provide Add, pdate, Delete, Next, first, last, pervious functionalities. Create Crystal Report for above application.

7. Design a crystal report for the table given in the Q:6. Report should be Author wise and summarize it with Book count and Total of the price (Author wise).

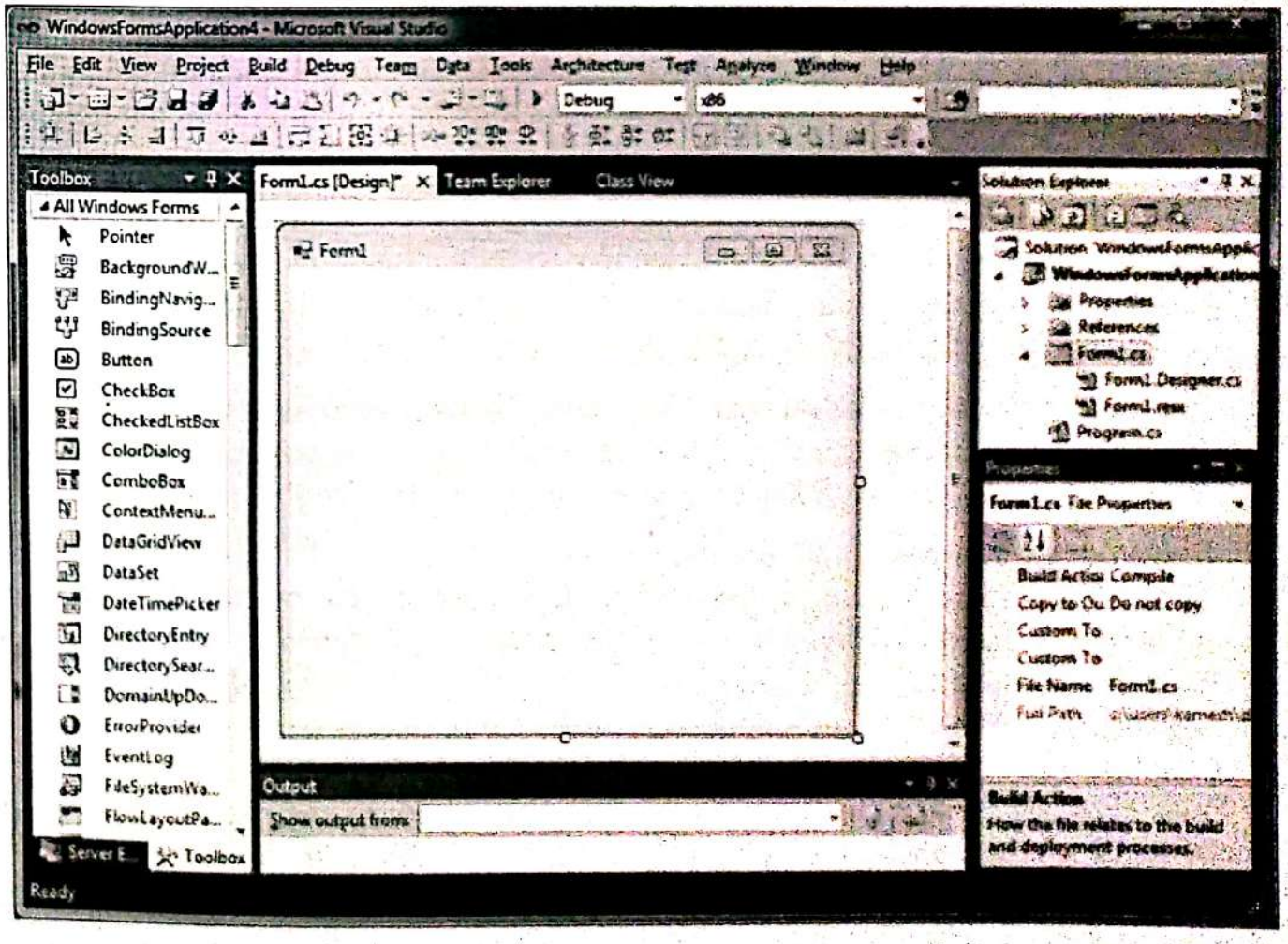
Introduction to .NET

INTRODUCTION:

.NET is a development environment or a platform provided my Microsoft Corporation, by using it developer can develop [1] Desktop based [2] Web-based or [3] Mobile applications. .NET consists of 3 main components.

[1] IDE:

It is called Integrated Development Environment. It is software which facilitates to Create, Design, and Code any type of application discussed above. It provide rapid application development platform to the developer using that developer can easily design the application and do need to write any code to design the form. It provides drag and drop interface. IDE made by many small windows inside it such as, property window, solution explorer, toolbox, design window, code window, output window, command window and many more. The following figure shows the IDE from .NET.



➢ .NET IDE(INTEGRATED DEVELOPMENT ENVIRONMENT) COMPONENTS:

The IDE of Visual Studio.NET contains numerous components that allows programmer to design, debug, test/run and deploy project from single window. The purpose of the IDE is to enable the developer to do as much as possible with visual tools, before writing code.

➢ THE IDE MENU:

The IDE main menu provides many menus, which lead to submenus. Most of the menus can also be displayed as toolbars. Also, not all options are available at all times. The options that cannot possibly apply to the current state of the IDE are either invisible or disabled.

1. File Menu

The File menu contains commands for opening and saving projects, or project items, as well as the commands for adding new or existing items to the current project.

2. Edit Menu

The Edit menu contains the usual editing commands. Among the commands of the Edit menu are the advanced command and the IntelliSense command.

Advanced Submenu

The options of the Edit → advanced submenu are the following. Advanced submenu is visible only in code editor.

View White Space: Space characters (necessary to indent lines of code and make it easy to read) are replaced by periods.

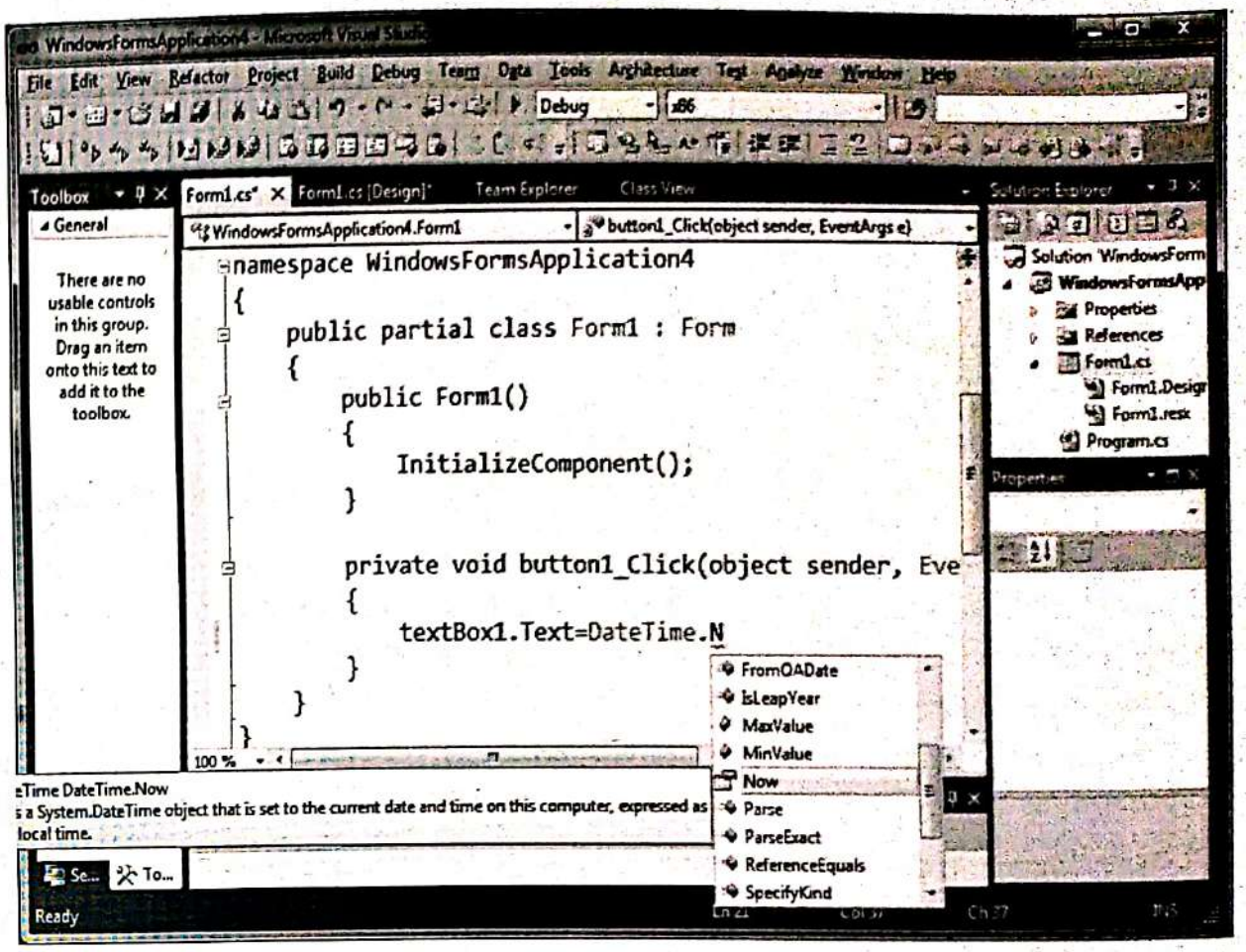
Word Wrap: When a code line's length exceeds the length of the code window, it's automatically wrapped.

Comment Selection/Uncomment Selection: Comments are lines you insert between your code's statements to document your application. This command allows you to comment (or uncomment) large segments of code in a single move.

IntelliSense Submenu:

IntelliSense is a feature of the editor that displays as much information as possible, whenever possible. When you type the name of a function and the opening parenthesis, for example, IntelliSense will display the syntax of the function—its arguments. The IntelliSense submenu includes the following options.

List Members When this option is on, the editor lists all the members (properties, methods, events, and argument list) in a drop-down list. This list will appear when you enter the name of an object or control followed by a period. Then you can select the desired member from the list with the mouse or with the keyboard.



Parameter Info While editing code, you can move the pointer over a variable, method, or property and see its declaration in a yellow ToolTip.

Quick Info This feature displays information about commands and functions. When you type the opening parenthesis following the name of a function, for example, the function's arguments will be displayed in a tooltip box (a yellow horizontal box).

Complete Word The Complete Word feature enables you to complete the current word by pressing Ctrl+spacebar. For example, if you type "TextB" and then press Ctrl+spacebar, you will see a list of words that you're most likely to type (TextBox, TextBox1, and so on).

3. View Menu

This menu contains commands to display any toolbar or window (including output and command window) of the IDE.

4. Project Menu

This menu contains commands for adding items to the current project (an item can be a form, a file, a component, even another project). The last option in this menu is the Set as StartUp Project command, which lets you specify which of the projects in a multiproject solution the startup is.

5. Build Menu

The Build menu contains commands for building (compiling) the project. The two basic commands in this menu are the Build and Rebuild All commands. The Build command compiles (builds the executable) of the entire solution, but it doesn't compile any components of the project that haven't changed since the last build. The Rebuild All command does the same, but it clears any existing files and builds the solution from scratch.

6. Debug Menu

This menu contains commands to start or end an application, as well as the basic debugging tools.

7. Data Menu

This menu contains commands you will use with projects that access data.

8. Format Menu

The Format menu, which is visible only while you design a Windows or Web form, contains commands for aligning the controls on the form.

9. Tools Menu

This menu contains a list of tools, and most of them apply to C++. The Macros command of the Tools menu leads to a submenu with commands for creating macros.

10. Window Menu

This is the typical Window menu of any Windows application. In addition to the list of open windows, it also contains the Hide command, which hides all Toolboxes and devotes the entire window of the IDE to the code editor or the Form Designer.

11. Help Menu

This menu contains the various help options.

➢ THE TOOLBOX WINDOW:

Here you will find all the controls you can use to build your application's interface. This window contains these tabs:

All Windows Forms
Common Controls
Containers
Menus & Toolbars
Data
Components
Printing
Dialogs
Reporting

➢ THE SOLUTION EXPLORER:

This window contains a hierarchical list of the items in the current solution. A solution may contain multiple projects, and each project may contain multiple items. By right clicking the component and selecting property, property of that component can be seen. You can also add items to a project with the Add Item command, or remove a component with the Exclude from Project command. This command removes the selected component from the project, but doesn't affect the component's file on the disk. The Remove command removes the selected component from the project and also deletes the component's file from the disk.

- **THE PROPERTIES WINDOW:**

This window (also known as the Property Browser) displays all the properties of the selected component and their settings. Many properties are set to a single value, like a number or a string. If the possible settings of a property are relatively few, they're displayed as meaningful constants in a drop-down list. Other properties are set through a separate window (ex. Font property). Collections are set in a Collection Editor Dialog box, where you can enter one string for each item of the collection. If Property window is not visible use view → property window command to open it.

- **THE FORM DESIGN WINDOW:**

Form is a basic building block on which various control resides. The form design window is used to design the interface through which user can interact. It is used mainly to create windows based applications.

- **THE CODE WINDOW:**

This is a window used to write code associated to various forms in VB applications. It may contain variable declarations, global items, event procedures and other functions.

- **THE OUTPUT WINDOW:**

The Output window is where many of the tools, including the compiler, send their output. Every time you start an application, a series of messages is displayed on the Output window. You can also send output to this window from within your code with the Console.WriteLine () method.

- **THE COMMAND WINDOW:**

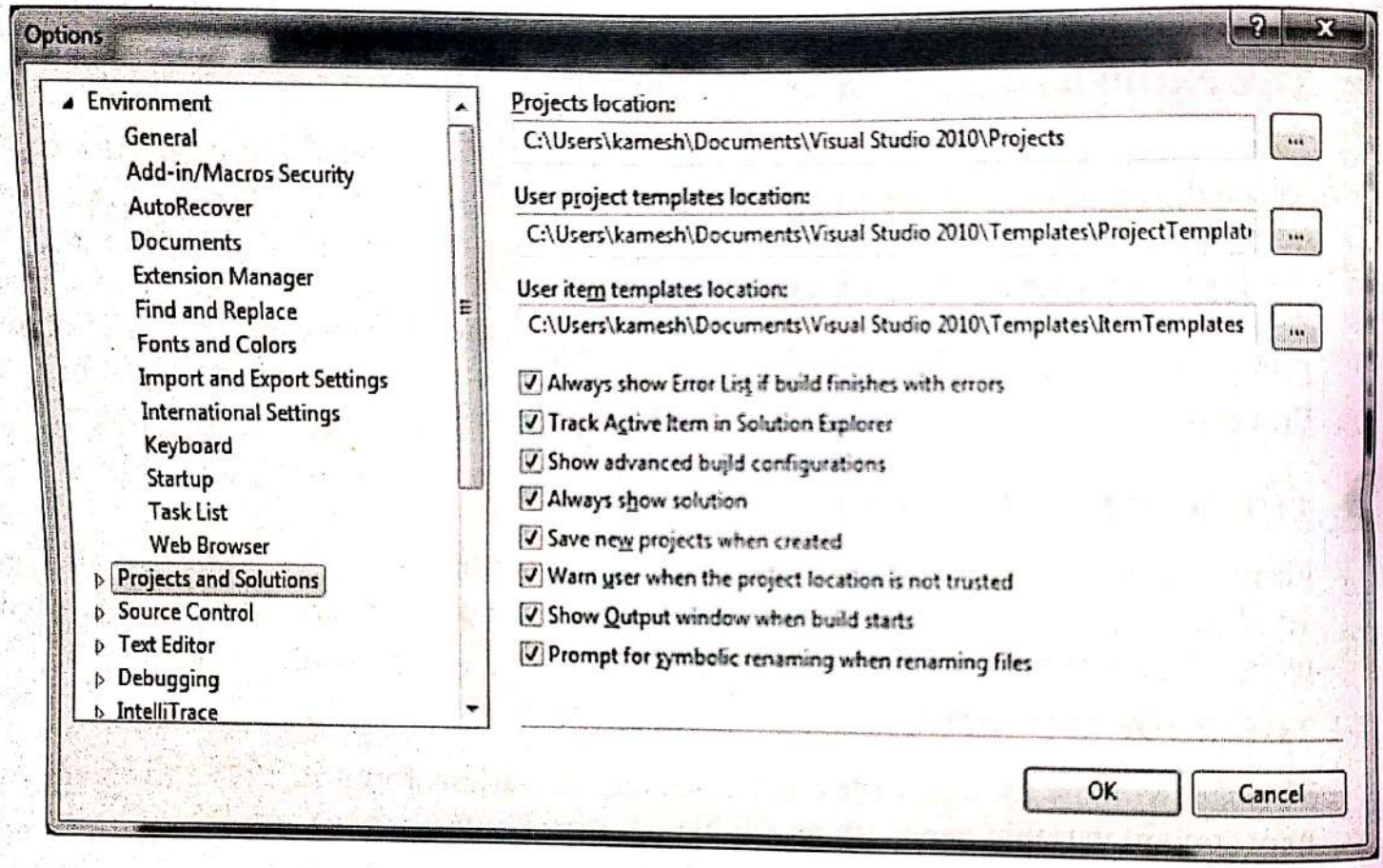
While testing a program, you can interrupt its execution by inserting a *breakpoint.* When the breakpoint is reached, the program's execution is suspended and you can execute a statement in the Command window.

- **THE TASK LIST WINDOW:**

This window is usually populated by the compiler with error messages, if the code can't be successfully compiled. You can double-click an error message in this window, and the IDE will take you to the line with the statement in error.

➢ ENVIRONMENT OPTIONS:

The Visual Studio IDE is highly customizable. Open the Tools menu and select Options where you can set all the options regarding the environment. You can set the font for various categories of items, like the Text Editor, the dialogs and toolboxes, and so on. It also provides Projects and Solutions options like saving changes automatically as soon as project is executed, or explicit saving.



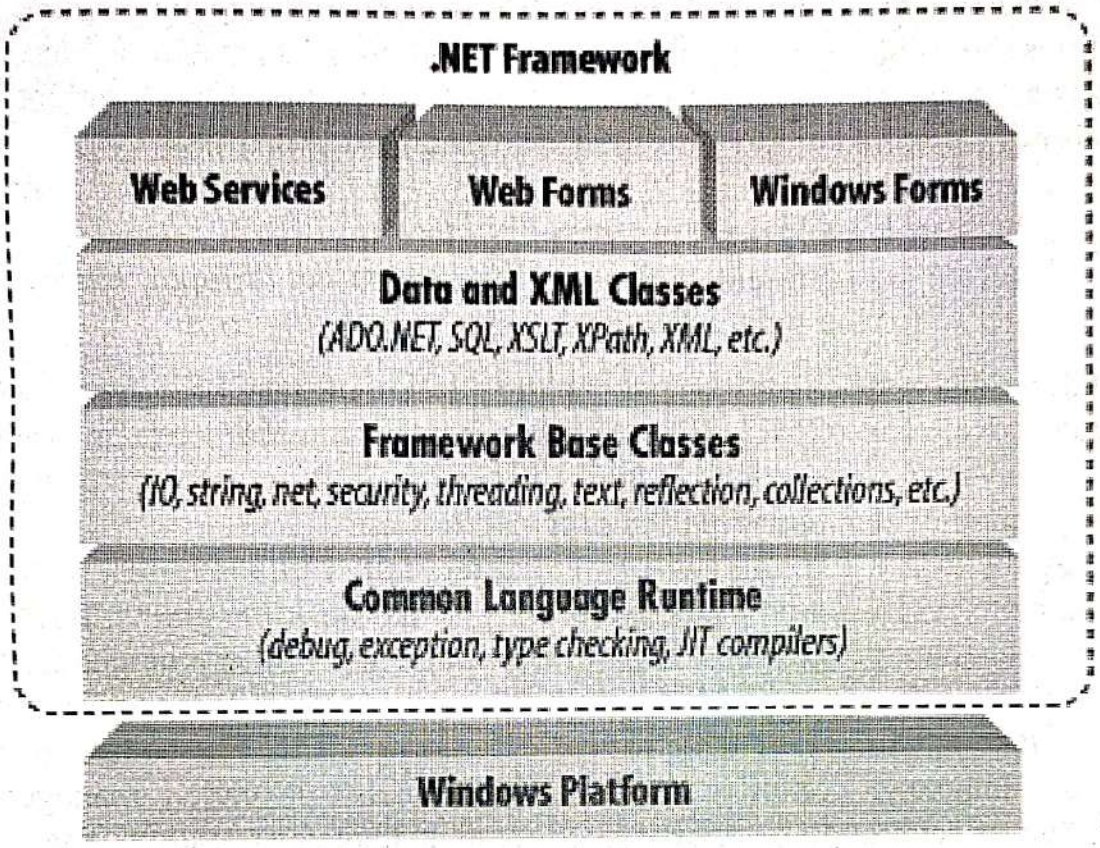
[2] .NET FRAMEWORK:

➢ INTRODUCTION:

Microsoft .NET is a set of Microsoft software technologies for connecting information, people, systems, and devices. It enables a high level of software integration through the use of Web services—small, discrete, building-block applications that connect to each other as well as to other, larger applications over the Internet.

Microsoft describes .Net framework as "A new computing platform designed to simplify application development in the highly distributed environment of the internet" and as "An XML web services platform that will enable developers to create programs that go beyond device boundaries and fully exploit the connectivity of the internet.

Components of the .NET Framework:



.NET framework consists of three layers of components. At the presentation layer it consists of web forms, windows forms and web services. Web forms are used in web-based applications, while windows forms are used in windows-based applications for providing an interactive user interface. Base class library and common language runtime allows creation of such web-based and windows-based applications. Such applications can be created in VB, C++, C# and other .Net languages.

Components and features included in .NET are:

The Common Language Runtime (CLR)
Managed execution
The common type system (CTS)
Cross-language interoperability
The .NET Framework class library
Namespaces
Assemblies
Application domains
Security
Deploying and configuring .NET applications
Web services
Windows forms
ASP.NET

➢ THE COMMON LANGUAGE RUNTIME:

At the heart of the .NET Framework is a new mechanism for loading and running programs and managing their interactions. This mechanism is described in the Common Language Infrastructure (CLI), a specification for a runtime environment that allows software components to:

- Pass data between each other without regard to the programming language in which each component is written.
- Execute on different operating system and on different hardware platforms without having to recompile the high-level source code.

The CLI is just a specification. An implementation of the CLI is known as a Common Language Runtime (CLR). CLI specifies how executable code is loaded, run and managed. Virtual Execution System (VES) of CLR actually performs the task of loading, running and managing the executable code.

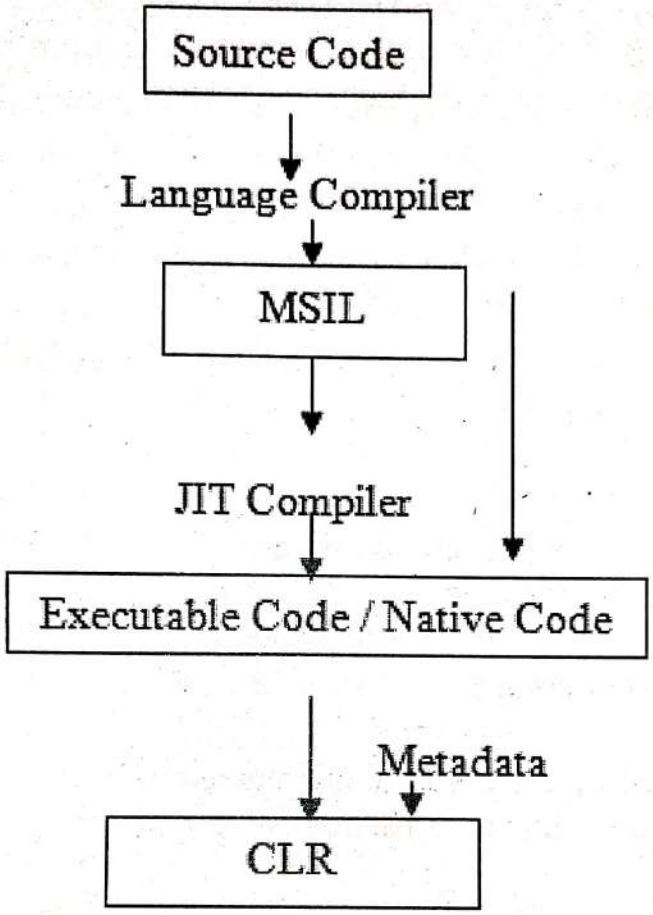
The runtime is just like the any other runtime, but with some differentiating feature. For example, VB runtime can execute only visual basic programs. In contrast to this CLR can execute programs written in any language. Primary languages for CLR are Visual Basic, C#, and C++. The compiler of any of these languages can compile the code to take advantage of the services offered by CLR. The code that is developed with the language compiler that targets the CLR is called managed code, and execution of such code by CLR is known as managed execution. That is because in addition to executing the code the CLR also provides services. For example, the CLR takes care of all memory management and garbage collection (reusing memory occupied by objects that are no longer in use) for .NET applications.

➢ MANAGED EXECUTION:

The entire process of turning source into a running application in the .NET Framework is referred to as managed execution. This process consists of four steps from start to finish:

1. Create your program's source code using a specialized development environment such as Visual Studio .NET or a general-purpose too such as a text editor.
2. Use a .NET compiler to turn the source code into a form known as Microsoft Intermediate Language (MSIL/IL). No matter which language has been used to develop the application, compiler targeting to CLR always translates the code in to IL. MSIL files typically have the file extension .dll or .exe. They look like an executable file to the operating system, although they cannot run without the CLR. The MSIL is a machine language that is not tied to any specific machine. MSIL format in independent of any particular operating system or hardware architecture.
3. To run .NET executable, the .NET Framework uses a just-in-time (JIT) compiler which translates the MSIL instructions into actual hardware-specifications that can be executed by CPU of computer.

4. The CLR passes the compiled code to the CPU, monitoring its execution to perform management services such as memory management, security checking, and versioning support.



Metadata:

As figure shows the MSIL version of application contains information other than the code to be executed. It is known as metadata. Metadata is separate section of the MSIL file that describes the content of the file. For example, the metadata contains description of the datatypes exposed by your application.

Source code consists of some constructs that are procedural in nature and other that are declarative.

Ex. Someobject.somemember = 5 is procedural.
Dim someobject as someclass is declarative.

In the past, declarative information typically was used only by the compiler and did not compile directly into executable code. While CLR uses type and signature information to ensure that memory is always referenced in a safe way. The JIT compiler uses this information to resolve method calls. Compilers that target the CLR therefore store both procedural and declarative information in the resulting *.exe* or *.dll* file. The procedural information is stored as IL, and the declarative information is stored as metadata. Metadata is just CLI's name for declarative information.

➢ THE COMMON TYPE SYSTEM:

The CLR defines the Common Type System (CTS). At the most basic level you can think of the CTS as defining all the datatypes that managed code is allowed to use. It also defines rules for crating, using and binding to types. CTS is an important part of the support provided by CLR for cross-language integration because it provides a type system that is common across all languages. That means the size of integer is the same across all the languages.

CTS also include rules for creating new types. You can define your own values or your own classes such that they will be acceptable to the CTS. The CTS manages many categories of types, including these:

- Built-in value types such as byte or Int32(a 32-bit signed integer)
- User-defined value types (type)
- Enumerations
- Pointers
- Classes from the .Net Framework class library
- User-defined classes
- Arrays
- Delegates (pointers to functions)
- Interfaces

All the types managed by the CTS are guaranteed to be type safe. That means that the CLR and CTS ensures that an instance of a type cannot overwrite memory that doesn't belong to it.

➢ CROSS-LANGUAGE INTEROPERABILITY:

.NET provides cross-language interoperability because the CLR manages all .NET code, regardless of the language in which it is written. That is, code written in one .NET language can be easily used from another .NET language. For example, you can define a class in VB.NET and then call the methods of that class, or even derive a new class from the original class, in C# code.

The key of interoperability is the metadata contained in MSIL files. Because this metadata is standardized across all .NET languages, a component written in one language can use the metadata to figure out the proper way to call a component written in another language.

To enable objects to interact with each other regardless of the language used to develop them, a set of language features has been defined in Common Language Specification (CLS). This specification includes the basic language features required by many applications. If you write components that conform to the CLS, you can be sure that they will be usable by components written in other .NET languages.

➢ THE .NET FRAMEWORK BASE CLASSES / CLASS LIBRARY:

The other major component of the .NET Framework, besides the CLR, is the .NET Framework class library. A class library is a set or predefined classes that can be used to access common functionality. It is built on the object-oriented nature of the runtime. One of the most important features of the .NET Framework class library is that it can be used in a consistent manner across multiple languages. This means that you can use the same set of classes for performing a specific task in Visual Basic as well as in Visual C++. This can simplify learning process a lot.

The .NET Framework class library is exceptionally rich, containing several hundred classes. These classes encapsulate functionality such as the following:

- Defining data with the CLR datatypes
- Defining data structures, including lists, queues, and hash tables
- Installing software
- Debugging applications
- Globalizing software
- Reading and writing data
- Interoperating with unmanaged code
- Managing threads
- Handling security

➢ NAMESPACES:

Classes within the .NET Framework class library are arranged in namespaces, groups of objects that perform similar functions. Namespaces allow you to organize your classes so that they can be easily accesses in other applications. Namespaces can also be used to avoid any naming conflicts between classes that have the same names. One can use two classes with the same name in an application provided that they belong to different namespaces.

Namespaces also contain other .NET entities such as structures, enumerations and interfaces. Namespaces, in turn, are arranges into a hierarchy.

Ex. System.Data.OleDb.OleDbConnetction (class)

└──→ System.Data.OleDb (namespace)

└──→ System.Data (root namespace)

Use statement: Imports System.Data to bring in the namespaces so that its objects can be used in the code.

➢ ASSEMBLIES:

.NET group's code into units called assemblies which contains all the information about the implementation of classes, structures, and interfaces in the code. An assembly can consist of a single file or of components distributed across multiple files. Say, if the application is fully contained in an .exe file, that's a one module assembly. But if the .exe is deployed with two .dll files then the three modules

together form an assembly. In Microsoft's terms, the assembly is the unit of deployment in .NET.

Assembly also stores all the information about itself in a file called *assembly manifest*. Manifest contains metadata about the assembly as a whole, including the list of modules contained in the assembly, the name and version of the assembly, security information, and information about the dependencies. Namespaces are also stored in assembly. Assembly and metadata provide the CLR with the information required for executing an application. Say, if any component is required to be loaded and executed, assembly will provide the information about that component.

- **APPLICATION DOMAINS:**

Application domains provide a second level of code grouping in .NET. It is composed of a group of assemblies loaded together. The CLR enforces isolation between application domains, such that code running in one application domain cannot directly manipulate objects in another application domain.

- **SECURITY:**

Security concerns have become more critical as more applications are connected to the internet, where a wide variety of people with bad intensions can attempt to exploit any holes. The .NET Framework implements both code access security (CAS) and role-based security (RBS). Code access security is designed to protect the operating system from malicious code by granting permissions to resources based on the source code and the operations that the code is attempting to perform.
Role-based security allows you to grant or deny access to resources based on identity of the user. Here, user's identity determines the roles to which the user belongs, and permissions are granted to roles.

- **DEPLOYING AND CONFIGURING .NET APPLICATIONS:**

The .NET Framework is designed to make deploying applications simple. .NET applications can be deployed simply by copying files to target computers. The CLR in conjunction with the metadata will take care of the rest.

With the Component Object Model (COM), every component needs to be registered before it can be used in an application. And two versions of the component cannot run side-by-side. That means if you install new application having new version of component in it, a new application will work properly but an existing application that uses previous version of the component may no function properly.

While in a .NET Framework installation of new application or components does not have adverse effect on the existing application. In .NET Framework applications are employed in assemblies. An assembly stores metadata. Therefore, registry entries are not required to store information about components and applications. Assemblies also

store information about the versions of components. So the problem of versioning is solved.

- **WEB SERVICES:**

They are a means for interacting with objects over the Internet. All communication between web services clients and servers is over HTTP and XML by default. For this to work there has to be a way to translate objects (as well as their methods and properties) into XML. That way is called SOAP, the simple Object Access Protocol. Because SOAP is XML-based, it can be interpreted by a wide variety of software on many operating systems.

- **WINDOWS FORMS:**

.NET includes a new visual programming model called Windows Forms. It is based on the MFC (Microsoft Foundation Class) and visual basic forms model, and provides a powerful way to design the user interface for .NET applications. It includes easy connection to ADO.NET data as well as support for CLR features such as versioning, licensing, and security.

- **ASP.NET:**

ASP.NET is Microsoft's server-based framework for running .NET code on web servers. ASP.NET still uses the CLR to execute and manage code, but it's tightly integrated into the web development process.It works just like existing ASP processor. The user sends, via a web browser, a request to open a page. The ASP.NET process on the server executes the .NET code that is contained within the requested page and sends only the results of the execution back to the client.

- **MEMORY MANAGEMENT AND GARBAGE COLLECTION:**

Memory management is done by memory manager of CLR, which keeps pointer to the address at which free memory (known as the heap) starts. To satisfy any memory request it just increments the pointer by the size of the request, leaving it in a position to satisfy the next memory request. This makes memory allocation very fast. As long as the heap doesn't run out, memory is not freed until the application exits.

If the heap runs out of memory, there is more work to do. To satisfy a memory request when the heap is exhausted, the memory manager looks for any previously allocated memory that can be freed. This can be done by invoking the garbage collection process. The garbage collection process runs through all objects and terminates objects that are no longer used by any application.

Features of CLR:

- Automatic memory management
- Standard type system
- Language interoperability

- Platform independence
- Security Management
- Types safety

Features of .NET framework:

- Consistent programming model (object-oriented programming model)
- Multi-platform applications
- Multi-language integration
- Automatic resource management
- Ease of deployment

[3] Languages:

.NET supports variety of languages. It gives sport to 14+ languages. Visual Basic .NET and C#.NET are most popular languages. Apart from the developer can develop the application by using visual C++, Visual J# and many other languages. At the time of creating the new project, IDE will provide the option to choose desire language to the developer.

❖ VISUAL PROGRAMMING:

In visual programming you "*paint*" your interface as opposed to using lines and lines of code to describe the interface.
The main 3 steps for creating application under VB .NET are:

1. **Design the user interface:** It includes designing forms and placing controls over it. Interface is basically a display that provides basic user interactivity.
2. **Set the properties for control:** Set the properties of the objects, say give each object a name and define such attributes as the contents of a label, the size of the text, and the words that appear on top of a button and in the form's title bar. You might think of this step as describing each object.
3. **Write the code for events:** Here you will define the actions of your program by writing basic programming statements.

❖ VISUAL STUDIO .NET PROJECT TYPES / APPLICATION TYPES:

All the project types supported by Visual Studio are displayed on the New Project dialog box, and they're the following:

CLASS LIBRARY:

A class library is a basic code-building component, which has no visible interface and adds specific functionality to your project. A class is a collection of functions that will be used in other projects beyond the current one. With classes, however, you don't

have to distribute source code. Class libraries are equivalent to ActiveX DLL and ActiveX EXE project types of VB6.

WINDOWS CONTROL LIBRARY:

A Windows control (or simply *control*), such as a TextBox or Button, is a basic element of the user interface. If the controls that come with Visual Basic don't provide the functionality you need, you can build your own custom controls. Custom controls can be created for very specific operations to simplify the development of large applications in a team environment.

CONSOLE APPLICATION:

A Console application is an application with a very limited user interface. Console applications are DOS-like applications. This type of application displays its output on a Command Prompt window and receives input from the same window.

WINDOWS SERVICE:

A Windows service is a new name for the old NT services, and they're long running applications that don't have a visible interface. These services can be started automatically when the computer is turned on, paused, and restarted. An application that monitors and reacts to changes in the file system is a prime candidate for implementing as a Windows service. When users upload files to a specific folder, the Windows service might initiate some processing (copy the file, read its contents and update a database, and so on).

ASP.NET WEB APPLICATION:

Web applications are among the most exciting new features of Visual Studio. A Web application is an app that resides on a Web server and services requests made through a browser. An online bookstore, for example, is a Web application. The application that runs on the Web server must accept requests made by a client (a remote computer with a browser) and return its responses to the requests in the form of HTML pages.

WEB CONTROL LIBRARY:

Just as you can build custom Windows controls to use with your Windows forms, you can create custom Web controls to use with your Web pages. Such web controls can be utilized in different ASP applications.

ASP.NET WEB SERVICE:

A Web service is not the equivalent of a Windows service. A Web service is a program that resides on a Web server and services requests, just like a Web application, but it doesn't return an HTML page. Instead, it returns the result of a calculation or a database lookup. Requests to Web services are usually made by another server, which is responsible for processing the data. A *Web application* that

accepts a query for all VB books published by Sybex will return a HTML page with the results. A *Web service* that accepts the same query will return an XML (Extensible Markup Language) file with the results. The file will be used by the application that made the request to prepare a new page and send it to the client, or to populate a Windows form.

❖ USER INTERFACE CONTROLS:

The user interface is what appears in the application's window when it runs. VB.Net provides rich set of built in interface controls. These controls are available in Tool Box of IDE. Some standard controls are:

Label , Text Box , Command Button , Option Button , Check Box , Frame , Picture Box, List Box , Combo Box, Group Box, List View, Tree View, Data Grid, Main Menu, Panel. There are 3 main things associated with each control.

1. Properties:

A property is named attribute of a control that defines characteristic behavior of particular control such as Size, Color etc. Some common properties are Name, Height, Top, Width, Left etc.

2. Events:

Events are various things that can happen in a program. Basically Events determine the control's reaction to some external conditions. For e.g. Click event : Once you specify the subroutine for the control's Click event, this subroutine is executes each time that control is clicked. Subroutines associated with different events are also known as event handler routines.

3. Methods:

A method is built-in procedure that can be invoked to perform an action that can be performed on controls. For e.g. Clear method clears the content of particular control.

To manipulate a control you use its properties, either on the Property Browser at design time, or though your code at runtime. To program a control, supply a handler for the appropriate events. Controls expose methods, too, which act on the control. The Hide method, for example, makes the control invisible. Properties, methods, and events constitute the programmatic interface of the control and are collectively known as the control's members.

➢ A Few Common Properties:

Different controls expose different properties, but here are some that are common to most:

Name: The control's name. This name appears at the top of the Properties window when a control is selected on the form and is also used in programming the control.

Font: A Font objects that determines how the text of the control will be rendered at both design and runtime.

Enabled: By default, all controls are enabled. To disable a control, set its Enabled property to False. When a control is disabled, it appears in gray color and users can't interact with it.Ex: If the user hasn't entered a value in all required fields on the form, clicking the Process button isn't going to do anything. After all fields have been set to a valid value, you can enable the control, indicating to the user that the button can now be clicked.

Size: Sets, or returns, the control's size. The Size property is a Size object, which exposes two properties, the Width and Height properties. You can set the Size property to a string like "320, 80" or expand the Size property in the Properties window and set the Width and Height properties individually.

Tag: Holds some data you want to associate with a specific control. For example, you can set the Tag property to the control's default value, so that you can restore the control's default value if the user supplies invalid data (a string in a TextBox control that expects a numeric value or a date).

Text: The text (a string) that appears on the control. The Label control's caption can be set (or read) through the Text property. A control that displays multiple items, like the ListBox or the ComboBox control returns the currently selected item in the Text property.

TabStop: Only one control at a time can have the focus on a form. To move the focus from one control to the other on the same form, you press the Tab key (and Shift+Tab to move the focus in reverse order). The TabStop property determines whether the control belongs to the so-called tab order. If True (which is the default value), you can move the focus to the control with the Tab key. If False, the control will be skipped in the tab order.

TabIndex: A numeric value that determines the position of the control in the Tab order. The control with the smallest TabIndex value is the one that has the focus when the form is first loaded. If you press Tab once, the focus will be moved to the control with the next larger TabIndex value.

Visible: Sometimes we want to make a control invisible. We do so by setting its visible property to False (the default value of the property is True).

- **A Few Common Events:**

The events that are most frequently used in programming Windows applications are shown next.

Click: This is the most common event in Windows programming, and it's fired when a control is clicked.

DoubleClick: Fired when the control is double-clicked.

Enter: Fired when the control received the focus.

Leave: Fired when the control loses the focus.

MouseEnter: Fired when the mouse pointer enters the area of the control. This event is fired once. If you want to monitor the movement of the mouse over a control, use the MouseMove event.

MouseLeave: This is the counterpart of the MouseEnter event, and it's fired when the mouse pointer moves out of the area of the control.

XXXChanged: Some events are fired when a property changes value. These events include BackColorChanged, FontChanged, VisibleChanged, and many more. The control's properties can also change through code, and it's very common to do so. You may wish to change the background color of a TextBox control if the numeric value displayed on it is negative:

```
If Val(TextBox.Text) < 0 Then
        TextBox1.BackColor = Color.Red
End If
```

- **A Few Common Methods:**

The following methods apply to most of the Windows controls.

Focus(): This method moves the focus to the control to which the method applies, regardless of the control that has the focus at the time. Your validation routine could move the focus to the control with an erroneous entry with the following statement: **TextBox1.Focus()**

Clear(): Many controls provide a method to clear their contents, and this is the Clear method. When you call the Clear method on a TextBox control, the control's Text property is set to an empty string.

Hide()/Show(): The Hide and Show methods reveal or conceal the control. The two methods are equivalent to setting the Visible property to True and False respectively.

PerformClick(): It's rather common to invoke the Click event of a button from within our code. To do so, call the PerformClick() method of the Button control. There are no equivalent methods for other events.

Scale(): This method scales the control by a value specified as argument. The following statement scales the TextBox1 control down to 75 percent of its current size: TextBox1.Scale(0.75)

☺ ☺ ☺ ☺ ☺

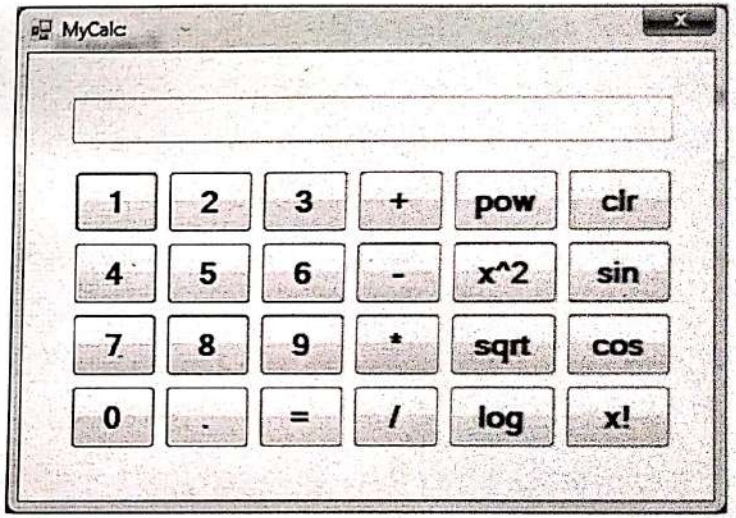
Programs of Unit-1

In this chapter we will do practical of standard controls of Visual Basic .NET that is Textbox, Label, Radiobutton, Checkbox, Combobox, Listbox and Timer control.

Program: 1

Design interface and implement functionalities for Arithmetic calculator with power, square, log, factorial, square root and clear functionalities.

- Design the following interface by taking a Textbox and 24 Buttons from the toolbox.



- **Set the following properties:**

Object	Property	Value	Explanation
Form	Text	MyCalc:	Shows MyCalc in the caption of the form
	MinimizeBox	False	User can not minimize the form
	MaximizeBox	False	User can not maximize the form
	FormBorderStyle	FixedSingle	User can not resize the form
	StartPosition	CenterScreen	Form opens in the center of the screen
	Font -> Size	14	
	Font -> Bold	True	
Textbox	Name	lblDisplay	In the coding we will use this name to refer our textbox
	Enabled	False	User can not type anything in the textbox

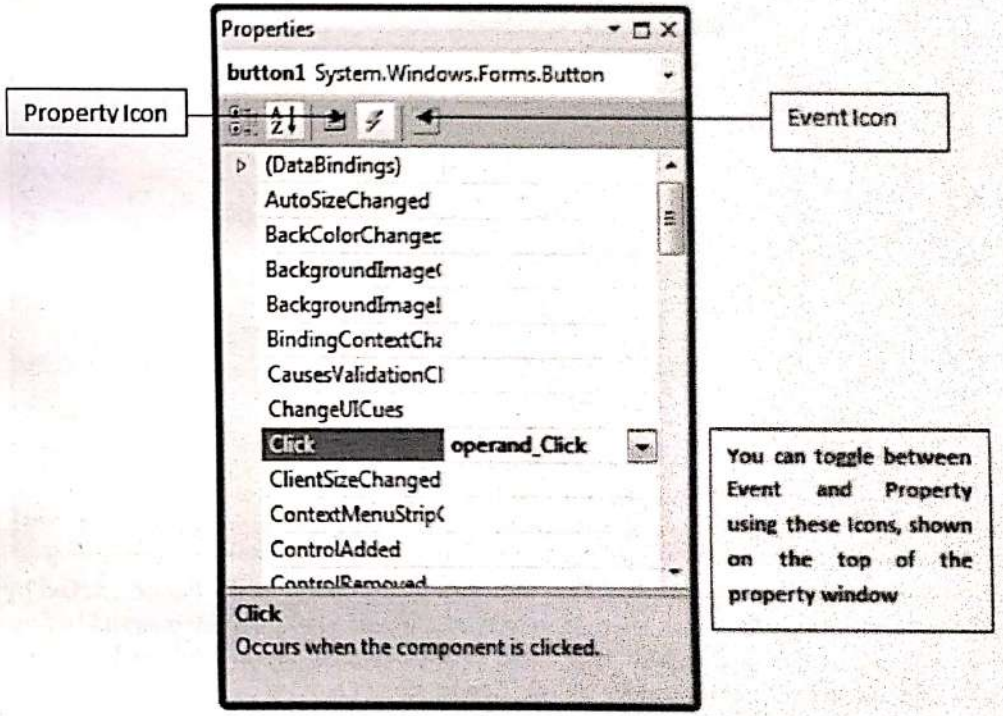
	TextAlign	Right	Textbox will show right aligned text
Button	Text	1	Set the caption for button1
Button	Text	2	
Button	Text	3	
Button	Text	4	
Button	Text	5	

- In the same way, set the captions for all the buttons as shown in the figure, using Text property of the buttons. Now double click on the button 1.
- It will open the following code in the code window.

```
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;

namespace Unit1_Q1
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
}

private void button1_Click(object sender, EventArgs e)
{

}
}
}
```

- In the namespace 'Unit1_Q1' is written. It is nothing but the name of the project I have given at the time of creation of new project. If you have given any other name then do not worry. But try to avoid name which can be a keyword (use naming rules which already we have learnt.

- Here you can get the subroutine 'button1_click', which is a subroutine gets executed every time when button1 is clicked. Change the name of the 'button1_click' to 'operand_Click'. The main idea here to write same code for all operands like 1, 2, 3, 4, 5, 6, 7, 8, 9, 0 and Dot(.), as they perform similar type of functionalities.
- Now go to the Design view, select button1. Click on the event icon in the property windows as shown below. Select 'Click' event and select 'operand_Click' event. This means, when user Click on the button subroutine 'operand_Click' will be executed.
- The same process has to be repeated for button2, button3 ... button9, button0 and that button on which Text is Dot(.). Here the same code will be executed on every button from 0 to 9 and Dot.



- Now declare three variables as shown below in the general declaration section, and write the following code in the subroutine 'operand_Click'.

```
namespace Unit1_Q1
{
    public partial class Form1 : Form
    {
        Boolean cleardisplay;
        String op;

        float operand;

        public Form1()
        {
            InitializeComponent();
        }

    private void operand_Click(object sender, EventArgs e)
        {
        if (((Button)sender).Text == "." &&
lblDisplay.Text.IndexOf(".") >= 0)
            {
                return;
            }
            if (cleardisplay == true)
            {
                lblDisplay.Clear();
                cleardisplay = false;
            }
            lblDisplay.Text = lblDisplay.Text +
((Button)sender).Text ;
        }
```

- In the code ***((Button)sender).Text*** indicates text property of the button clicked by the user. See here the same function is executed if user clicks on any operand button like (1,2,3 … etc.) . Then how will come to know which button is clicked by the user? We can get it through sender object. Sender object is type casted by (button) and we are fetching its Text property.
- The code you need to type is shown in the bold face font. Three variables are cleardisplay, op and operand. Variable cleardisplay is of type Boolean, which is able to store on two values True or False. This variable is used to control when to clear lblDisplay. Second variable is 'op' which is of type string and it will store which operator is selected by the user. Third variable is operand which of type float and it will store the first number entered by the user. Once the first number is copied into

the operand variable we can clear the lblDisplay so that user can enter second number.

- To understand this let us consider an example: suppose user want to do addition of 12 and 23. Obviously in this case user will click 12+23=. Let's understand what will be values in these variables after pressing each button.

State	Variable	Value
Initially	cleardisplay lblDisplay op operand	False Empty Empty 0.0
After pressing 1	clearDisplay lblDisplay op operand	False 1 Empty 0.0
After pressing 2	clearDisplay lblDisplay op operand	False 12 [As cleardisplay is False it will do concatenation of 1 and 2] Empty 0.0
After pressing +	clearDisplay lblDisplay op operand	True 12 + [When operator is clicked then it will be stored in the op var] 12 [When operator is clicked then the value in the lblDisplay will be copied in the operand variable]
After pressing 2	clearDisplay lblDisplay op operand	False [If clearDisplay is Then is will clear lblDisplay and it becomes false] 2 [It removes previously entered value 12 and takes recently entered value that is 2] + 12
After pressing 3	clearDisplay lblDisplay op operand	False 23 [As cleardisplay is False it will do concatenation of 2 and 3] + 12
After pressing =	clearDisplay lblDisplay op operand	False 35 [As on the click event of equal button will check op variable is having + so we will do operand + lblDisplay (12+23) and placed it the lblDisplay. + 12

- I think from the above discussion you can easily understand the code we have written in the subroutine 'operand_click'. In this subroutine we will check if the button pressed by the user is Dot(.) and if already Dot(.) exists in the lblDisplay then we will return from the subroutine. As already 1 Dot(.) is there we will not allow user to enter another Dot(.).
- Now click on the button '+'. It will open another subroutine in the code view. Change its name to 'operator_Click'. Type following code in it and set this subroutine to the Click event for buttons +, -, * and /. Obviously we will execute the same code for all the binary operators.

```
private void operator_Click(object sender, EventArgs e)
        {
            if (op != "")
            {
                equal_Click(sender, e);
                op = "";
            }
            op = ((Button)sender).Text;
            operand = Convert.ToSingle(lblDisplay.Text);
            cleardisplay = true;
        }
```

- If user do not click on = button and does continues operation like 2+2+2 then, automatically code written on equal button will be executed when user presses + button second time in the above operation. When user is clicking on any operator buttons like +, -, * and /, then operator gets stored in the op variable, first number which in the lblDisplay get stored in the operand variable and cleardisplay variable will becomes true so that now if user presses any number button then lblDisplay gets clear. In the same way create another subroutine called 'equal_Click', type the following code and assign this subroutine to the click event of the equal button using property window.

```
private void equal_Click(object sender, EventArgs e)
        {
            if (op == "+")
                lblDisplay.Text = Convert.ToString(operand +
Convert.ToSingle(lblDisplay.Text));
            if (op == "-")
                lblDisplay.Text = Convert.ToString(operand -
Convert.ToSingle(lblDisplay.Text));
            if (op == "*")
```



```
                lblDisplay.Text = Convert.ToString(operand *
Convert.ToSingle(lblDisplay.Text));
            if (op == "/")
                lblDisplay.Text = Convert.ToString(operand /
Convert.ToSingle(lblDisplay.Text));
        }
```

- The code is very simple; on the click event of the equal button we will check the op variable. If it has '+' sign then we will do operand (first number) + lblDisplay.Text (second number) and put it into the Text property of the lblDisplay. If op is having '-' then we will do subtraction between operator and lblDisplay and so on.
- **Now change the Name property for the following buttons:**

Pow	bpow
X^2	bsquare
Sqrt	bsqrt
Log	blog
Clr	bclr
Sin	bsin
Cos	bcos
X!	bfact

- Now pow is binary operation (it needs two operands), where as x^2 (square), sqrt (square root), x! (factorial), log, sin, cos are unary operators (they need on one opreand). Operation on binary operations are same as operation done for the operator +. -, * and division. You just have to call the procedure 'operator_Click' on the Click event of the button pow, and you need to add following 2 lines of code in the 'equal_Click' subroutine as given below:

```
if(op=="pow")
  lblDisplay.Text =
Convert.ToString(Math.Pow(operand,Convert.ToDouble(lblDisplay.Text
)));
```

- Now, go to the design mode and double click on the x^2 button. It will open a new subroutine in the code view. Write the following code in that subroutine.

```
private void bsqure_Click(object sender, EventArgs e)
        {
        Single num=Convert.ToSingle(lblDisplay.Text );
        lblDisplay.Text = Convert.ToString(num * num);
    }
```

➢ In the same way go to the design mode double click on the sqrt button. It will open a new subroutine in the code view. Write the following code in that subroutine. The same

```
private void bsqrt_Click(object sender, EventArgs e)
        {
            lblDisplay.Text =
Convert.ToString(Math.Sqrt(Convert.ToDouble(lblDisplay.Text)));
        }

        private void blog_Click(object sender, EventArgs e)
        {
            lblDisplay.Text =
Convert.ToString(Math.Log10(Convert.ToDouble(lblDisplay.Text)));
        }

        private void bsin_Click(object sender, EventArgs e)
        {
            lblDisplay.Text =
Convert.ToString(Math.Sin(Convert.ToDouble(lblDisplay.Text)));
        }

        private void bcos_Click(object sender, EventArgs e)
        {
            lblDisplay.Text =
Convert.ToString(Math.Cos(Convert.ToDouble(lblDisplay.Text) ));
        }

        private void bfact_Click(object sender, EventArgs e)
        {
            int ans, num, i;
            num = Convert.ToInt16(lblDisplay.Text);
            ans = 1;
            for (i = 1; i <= num; i++)
                ans = ans * i;
            lblDisplay.Text = Convert.ToString(ans);
        }

        private void bclr_Click(object sender, EventArgs e)
        {
            cleardisplay = false;
            lblDisplay.Clear();
            op = "";
            operand = 0;
        }
```


➢ So, finally we have completed the design and code given in the question1. Run and test the application. The complete code of the calculator is given below:

```
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;
namespace Unit1_Q1
{
    public partial class Form1 : Form
    {
        Boolean cleardisplay;
        String op;
        float operand;
        public Form1()
        {
            InitializeComponent();
        }

        private void operand_Click(object sender, EventArgs e)
        {
            if (((Button)sender).Text == "." &&
lblDisplay.Text.IndexOf(".") >= 0)
            {
                return;
            }
            if (cleardisplay == true)
            {

                lblDisplay.Clear();
                cleardisplay = false;
            }
            lblDisplay.Text = lblDisplay.Text +
((Button)sender).Text ;
        }

        private void operator_Click(object sender, EventArgs e
        {
            if (op != "")
```

```
            {
                equal_Click(sender, e);
                op = "";
            }
            op = ((Button)sender).Text;
            operand = Convert.ToSingle(lblDisplay.Text);
            cleardisplay = true;
        }

        private void equal_Click(object sender, EventArgs e)
        {
            if (op == "+")
                lblDisplay.Text = Convert.ToString(operand +
Convert.ToSingle(lblDisplay.Text));
            if (op == "-")
                lblDisplay.Text = Convert.ToString(operand +
Convert.ToSingle(lblDisplay.Text));
            if (op == "*")
                lblDisplay.Text = Convert.ToString(operand +
Convert.ToSingle(lblDisplay.Text));
            if (op == "/")
                lblDisplay.Text = Convert.ToString(operand +
Convert.ToSingle(lblDisplay.Text));
            if(op=="pow")
                lblDisplay.Text =
Convert.ToString(Math.Pow(operand,Convert.ToDouble(lblDisplay.Text
)));
        }

        private void bsqure_Click(object sender, EventArgs e)
        {
            Single num=Convert.ToSingle(lblDisplay.Text );
            lblDisplay.Text = Convert.ToString(num * num);
        }

        private void bsqrt_Click(object sender, EventArgs e)
        {
            lblDisplay.Text =
Convert.ToString(Math.Sqrt(Convert.ToDouble(lblDisplay.Text)));
        }

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



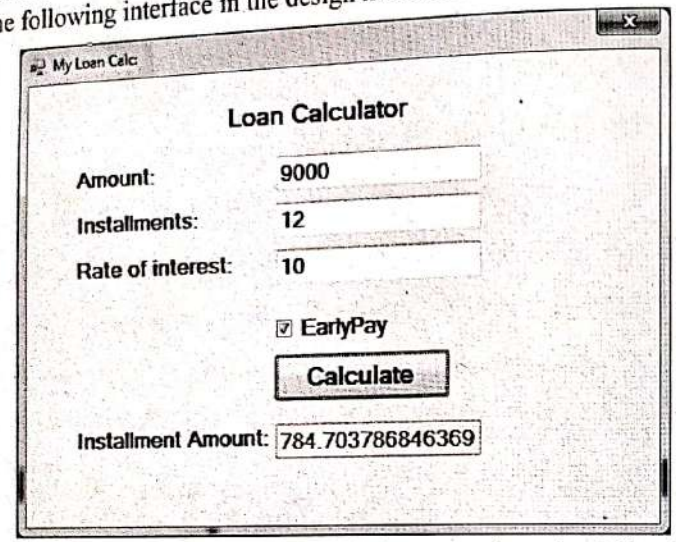
```
            lblDisplay.Text =
Convert.ToString(Math.Log10(Convert.ToDouble(lblDisplay.Text)));
        }

        private void bsin_Click(object sender, EventArgs e)
        {
            lblDisplay.Text =
Convert.ToString(Math.Sin(Convert.ToDouble(lblDisplay.Text)));
        }

        private void bcos_Click(object sender, EventArgs e)
        {
            lblDisplay.Text =
Convert.ToString(Math.Cos(Convert.ToDouble(lblDisplay.Text) ));
        }

        private void bfact_Click(object sender, EventArgs e)
        {
            int ans, num, i;
            num = Convert.ToInt16(lblDisplay.Text);
            ans = 1;
            for (i = 1; i <= num; i++)
                ans = ans * i;
            lblDisplay.Text = Convert.ToString(ans);

    }

        private void bclr_Click(object sender, EventArgs e)
        {
            cleardisplay = false;
            lblDisplay.Clear();
            op = "";
            operand = 0;
        }
    }
}
```

Program: 2

Design interface and implement functionalities for Loan calculator. Take Amount, No of installments and Rate of interest from the user. Also user can choose Early Pay option through a

checkbox. Calculate installment amount using pmt() function. Do proper validation for inputs taken by the user.

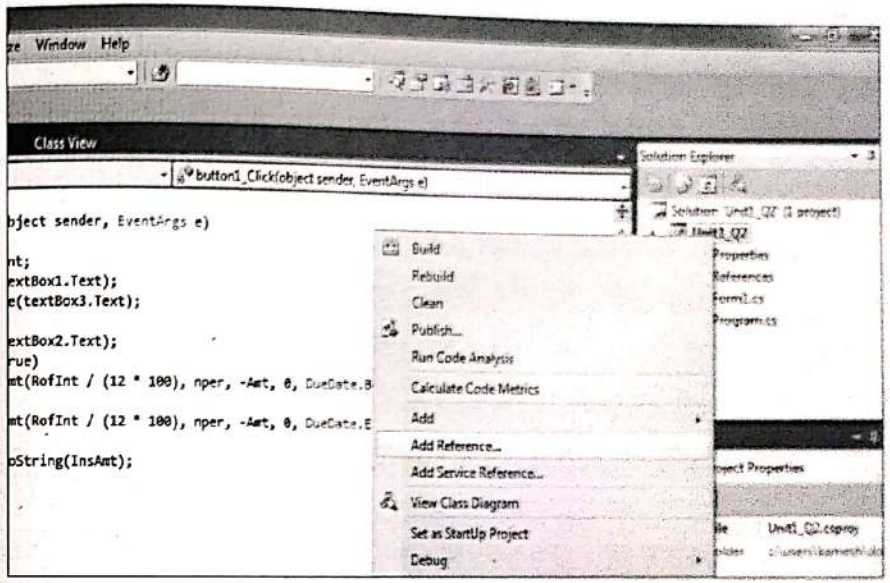
- Click on the start button, type command 'devenv' to launch visual studio and press Enter key. When the visual studio gets open select File -> New ->Project. In the dialogBox select C# language and select Windows Form Application template. Give project name to Unit1_Q2.
- Design the following interface in the design mode of the Form1.



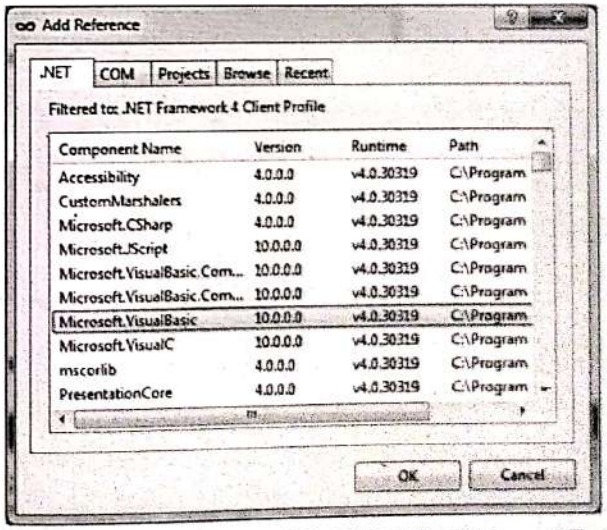
- Take 5 Labels, 4 textBoxes, a checkbox and a button and set the following properties.

Object	Property	Value	Explanation
Form	Text	My Loan Calc:	Shows My Loan Calc as form caption
	MinimizeBox	False	User can not minimize the form
	MaximizeBox	False	User can not maximize the form
	FormBorderStyle	FixedSingle	User can not resize the form
	StartPosition	CenterScreen	Form opens in the center of the screen
	Font -> Size	12	
	Font -> Bold	True	
Label1	Text	Loan Calculator	Set the caption of the label
	Font -> Size	14	
Label2	Text	Amount:	
Label3	Text	Installments:	
Label4	Text	Rate of Interest:	
Checkbox1	Text	Early Payment	
Label5	Text	Installment Amount:	
Button1	Text	Calculate	

- Now we need to add the Reference library Microsoft.Visualbasic, because we are going to use a function Pmt() is a built in function available in the namespace Microsoft.Visualbasic. We cannot use Microsoft.VisualBasic namespace unless we have not added its reference library. To do this, Right click on the solution name (Unit1_Q2) in the solution explorer and select option Add Reference as shown in the figure given below.



- Now from the dialogBox select '.NET' tab, and select 'Microsoft.Visualbasic' and click on the OK button, which shown in the following figure.



- Now add the namespace by writing 'using Microsoft.VisualBasic;' on the top of the code window, and type the following code on the click event of the button.

```
private void button1_Click(object sender, EventArgs e)
        {
            Double Amt, InsAmt, RofInt;
            Amt = Convert.ToSingle(textBox1.Text);
            RofInt = Convert.ToSingle(textBox3.Text);
            Int16 nper;
            nper = Convert.ToInt16(textBox2.Text);
            if(checkBox1.Checked ==true)
                InsAmt = Financial.Pmt(RofInt / (12 * 100), nper, -
Amt, 0, DueDate.BegOfPeriod);
            else
                InsAmt = Financial.Pmt(RofInt / (12 * 100), nper, -
Amt, 0, DueDate.EndOfPeriod);

            textBox4.Text =Convert.ToString(InsAmt);
        }
```

- Make sure amount entered by the user has to be negating while we are passing to the function, and rate of interest has to be divided by (12 * 100) as rate of interest of a year (12 months and rate of interest is calculated for Rs. 100).

- To validate the amount, we are allowing user to enter only numeric keys, back space key and Dot(.) keys. We restrict user not to enter any alphabet or special symbol in the amount field. For that we will write following code on the keypress event of the Textbox1 which takes amount from the user. So select Textbox1, go to property window, and click on event icon double click on keypress event. It will create a subroutine called Textbox1_keypress add the following code to this subroutine.

```
private void textBox1_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (Convert.ToInt16(e.KeyChar) == 8)
                return;
            if (e.KeyChar == '.' && textBox1.Text.IndexOf(".") >= 0)
                e.Handled = true;
            if(!Char.IsDigit(e.KeyChar) && !(e.KeyChar =='.'))
                e.Handled =true;
        }
```

- ASCII value of the key pressed is 8 (Back space) then allow (return). If the key pressed by the user is . Dot(.) and already Dot(.) is there in the textbox1 then do not

allow it (e.handled = true;). If key pressed by the user is not digit or Dot(.) then do not allow (e.handled=true;).

The same validation code you can use to validate other textboxes. The complete code of the loan calculator is given below.

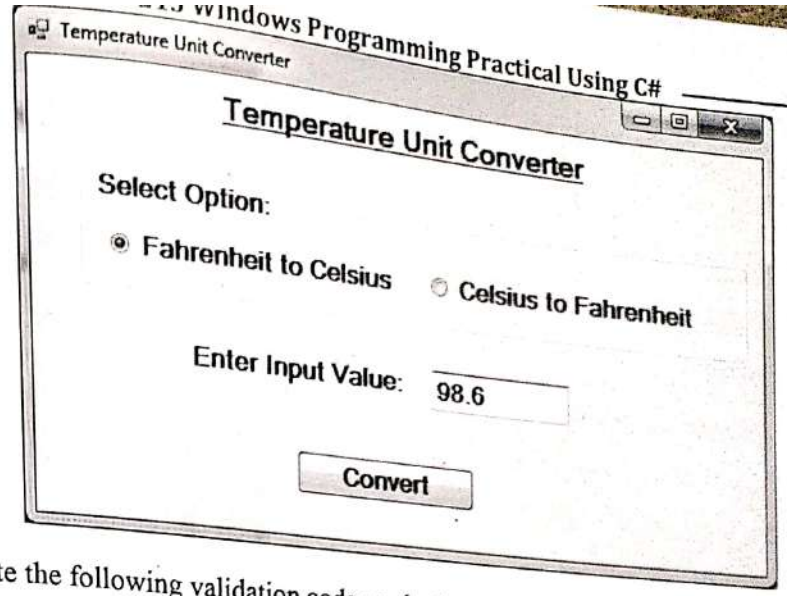
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using Microsoft.VisualBasic;
namespace Unit1_Q2
{

public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Double Amt, InsAmt, RofInt;
            Amt = Convert.ToSingle(textBox1.Text);
            RofInt = Convert.ToSingle(textBox3.Text);
            Int16 nper;
            nper = Convert.ToInt16(textBox2.Text);
            if(checkBox1.Checked ==true)
                InsAmt = Financial.Pmt(RofInt / (12 * 100), nper, -
Amt, 0, DueDate.BegOfPeriod);
            else
                InsAmt = Financial.Pmt(RofInt / (12 * 100), nper, -
Amt, 0, DueDate.EndOfPeriod);

            textBox4.Text =Convert.ToString(InsAmt);
        }

        private void textBox1_KeyPress(object sender,
KeyPressEventArgs e)
        {
```

```
if (Convert.ToInt16(e.KeyChar) == 8)
                return;
            if (e.KeyChar == '.' && textBox1.Text.IndexOf(".") >= 0)
                e.Handled = true;
            if(!Char.IsDigit(e.KeyChar) && !(e.KeyChar =='.'))
                e.Handled =true;
        }

        private void textBox2_KeyPress(object sender,
KeyPressEventArgs e)
        {
            if (Convert.ToInt16(e.KeyChar) == 8)

        return;
            if (!Char.IsDigit(e.KeyChar))
            {
                MessageBox.Show("Invalid Input");
                e.Handled = true;
            }
        }
    }
}
```

Program: 3

Design an application which will have 2 radio buttons. One will convert the Celsius to Fahrenheit and another will convert Fahrenheit to Celsius. Show the appropriate output depends on the user's selection. (Use radio button to take user choice and use textbox to enter value).

- Create a new project called Unit1_Q3 and design the following interface in the form1 using two labels, a group box, two radio buttons, a textbox and a button. Place both the radio buttons into the group box so that user can select any one from them.



- Write the following validation code on the keypress event of the Textbox1 so that user cannot enter any alphabet or special symbols in it. Textbox1 will allow only one Dot(.), digits and back space only.

```
private void textBox1_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (Convert.ToInt16(e.KeyChar) == 8)
                return;
            if (e.KeyChar == '.' && textBox1.Text.IndexOf(".") >= 0)
                e.Handled = true;
            if(!Char.IsDigit(e.KeyChar) && !(e.KeyChar =='.'))
                e.Handled =true;
        }
```

- Now write the following code on the click event of the Button1.

```
private void button1_Click(object sender, EventArgs e)
        {
            float f, c;
            if (radioButton1.Checked == true)
            {
                f = Convert.ToSingle(textBox1.Text);
                c = (5 * (f - 32)) / 9;
                MessageBox.Show(Convert.ToString(f) + " Fahrenheit
is Equal to " + Convert.ToString(c) + " Celsius");
            }
            else
```

```
            {
                c = Convert.ToSingle(textBox1.Text);
                f = ((9 * c) / 5) + 32;
                MessageBox.Show(Convert.ToString(c) + " Celsius is
Equal to " + Convert.ToString(f) + " Fahrenheit");
            }
        }
```

➢ We are starting the code by declaring two variables of type single that is f and c. Then we check if Radiobutton1 is checked then the value given in the Textbox1 by the user is in the Fahrenheit and we have to convert it to Celsius using the formula C=(5 * (f - 32)) / 9, otherwise user has selected second radio button, and the value given in the textbox1 is of Celsius and we have to converting it to Fahrenheit using formula: F=((9*C)/5)+32. We are calculating and showing the result using Messagebox.Show method. The complete code is given below:

```
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;

namespace Unit1_Q3
{
    public partial class Form1 : Form
    {
      public Form1()
        {
            InitializeComponent();
        }

      private void button1_Click(object sender, EventArgs e)
        {
            float f, c;
            if (radioButton1.Checked == true)
            {
                f = Convert.ToSingle(textBox1.Text);
                c = (5 * (f - 32)) / 9;
                MessageBox.Show(Convert.ToString(f) + " Fahrenheit
Equal to " + Convert.ToString(c) + " Celsius");
            }
```

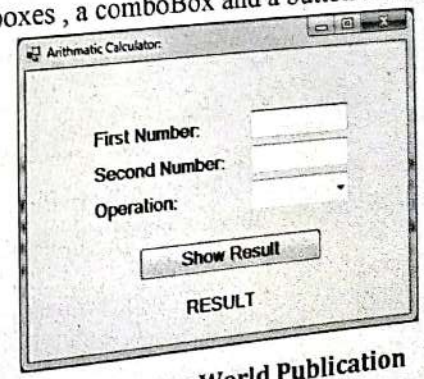


```
            else
            {
                c = Convert.ToSingle(textBox1.Text);
                f = ((9 * c) / 5) + 32;
                MessageBox.Show(Convert.ToString(c) + " Celsius is
Equal to " + Convert.ToString(f) + " Fahrenheit");
            }
        }

        private void textBox1_KeyPress(object sender,
KeyPressEventArgs e)
        {

            if (Convert.ToInt16(e.KeyChar) == 8)
                return;
            if (e.KeyChar == '.' && textBox1.Text.IndexOf(".") >= 0)
                e.Handled = true;
            if (!Char.IsDigit(e.KeyChar) && !(e.KeyChar == '.'))
                e.Handled = true;

        }
    }
}
```

Program: 4

Design a form having two text boxes, combo box and a label. Make the validation so that user can enter only numbers in both texboxes, if user has entered both numerical values then make the combo box visible. The combo box has options like 'ADD', 'SUB', 'MUL' and 'DIV'. According to user's choice from combo, result will display in label.

➢ Create a new project called Unit1_Q4 and design following interface for form1 using 4 labels, two textboxes , a comboBox and a button as shown in the following figure:



- Now double click on the form, it will open a new subroutine called 'form1_Load'. Write the following code in the load event, which will populate ADD, SUB, MUL and DIV options in the comboBox.

```
private void Form1_Load(object sender, EventArgs e)
        {
            comboBox1.Items.Add("ADD");
            comboBox1.Items.Add("SUB");
            comboBox1.Items.Add("MUL");
            comboBox1.Items.Add("DIV");
        }
```

- Write the code so that user can enter only back space, digits or single Dot(.) as discussed in the Unit1_Q3 on the keypress event of the Textbox1 and Textbox2. Double click on the Button1, which will create and open a new subroutine called button1_Click() and write the following code in it.

```
private void button1_Click(object sender, EventArgs e)

        {
            Single num1, num2;
            num1 = Convert.ToSingle(textBox1.Text);
            num2 = Convert.ToSingle(textBox2.Text);
            if (comboBox1.Text == "ADD")
                label4.Text = "RESULT is " +
Convert.ToString(num1 + num2);
            else if (comboBox1.Text == "SUB")
                label4.Text = "RESULT is " +
Convert.ToString(num1 - num2);
            else if (comboBox1.Text == "MUL")
                label4.Text = "RESULT is " +
Convert.ToString(num1 * num2);
            else if (comboBox1.Text == "DIV")
                label4.Text = "RESULT is " +
Convert.ToString(num1 / num2);
            else
                label4.Text = "RESULT";
        }
```

When user will click on the button result will be displayed in the Label4 like "RESULT is 28" based on the operation selected by the user from the comboBox1. The complete code is given below:

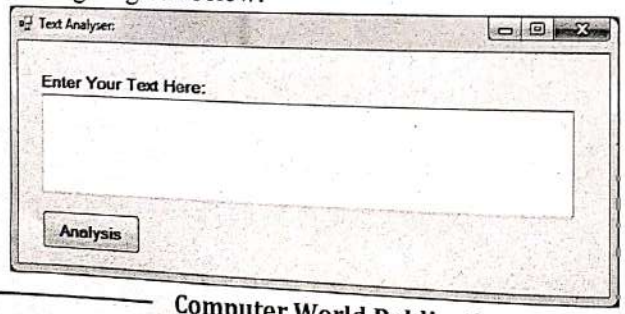

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace Unit1_Q4
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private void Form1_Load(object sender, EventArgs e)
        {
            comboBox1.Items.Add("ADD");
            comboBox1.Items.Add("SUB");
            comboBox1.Items.Add("MUL");
            comboBox1.Items.Add("DIV");
        }
        private void button1_Click(object sender, EventArgs e)
        {
            Single num1, num2;
            num1 = Convert.ToSingle(textBox1.Text);
            num2 = Convert.ToSingle(textBox2.Text);
            if (comboBox1.Text == "ADD")
                label4.Text = "RESULT is " + Convert.ToString(num1 +
num2);
            else if (comboBox1.Text == "SUB")
                label4.Text = "RESULT is " + Convert.ToString(num1 -
num2);
            else if (comboBox1.Text == "MUL")
                label4.Text = "RESULT is " + Convert.ToString(num1 *
num2);
            else if (comboBox1.Text == "DIV")
                label4.Text = "RESULT is " +
Convert.ToString(num1 / num2);
            else
```

```
            label4.Text = "RESULT";
        }
        private void textBox1_KeyPress(object sender,
KeyPressEventArgs e)
        {
            if (Convert.ToInt16(e.KeyChar) == 8)
                return;
            if (e.KeyChar == '.' && textBox1.Text.IndexOf(".") >= 0)
                e.Handled = true;
            if (!Char.IsDigit(e.KeyChar) && !(e.KeyChar == '.'))
                e.Handled = true;
        }
        private void textBox2_KeyPress(object sender,
KeyPressEventArgs e)
        {
            if (Convert.ToInt16(e.KeyChar) == 8)
                return;
            if (e.KeyChar == '.' && textBox2.Text.IndexOf(".") >= 0)
                e.Handled = true;
            if (!Char.IsDigit(e.KeyChar) && !(e.KeyChar == '.'))
                e.Handled = true;
        }
    }
}
```

Program: 5

Create an application with a textbox in which user can enter a sentence then displays
1) Number of vowels
2) Number of spaces
3) Number of digits
4) Number of special symbols
When user press "analysis" button.

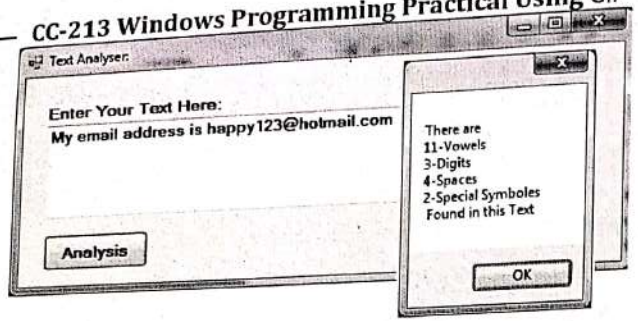
Create a new project called Unit1_Q5 and design the simple interface for form1 as shown in the figure given below:



- Set the Text property for the form is "Text Analyzer:", and Font property to size 10, and Bold=True. Take a Label and set the Text property of the Label1 to "Enter Your Text Here:". Take a Textbox and set its Multiline property to True. Place a Button as shown in the figure and set the Text property: "Analysis". Double click on the button it, which will create a new subroutine called button1_Click and add the following code to this soubroutine as given below:

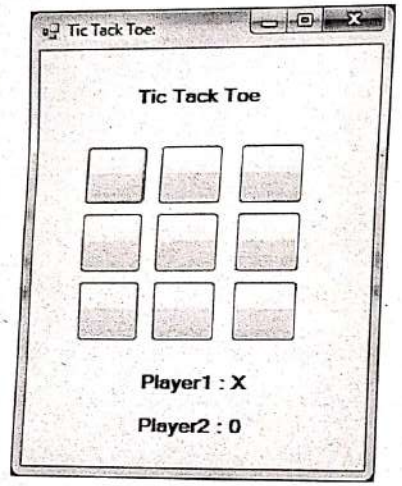
```
private void button1_Click(object sender, EventArgs e)
        {
            int vowels=0, spaces=0, digits=0,
spsymbols=0,consonant=0;
            int i;
            for (i = 0; i < textBox1.Text.Length; i++)
            {
                Char ch = Char.ToLower(textBox1.Text[i]);
                if (Char.IsDigit(ch))
                    digits++;
                else if (ch == ' ')
                    spaces++;
                else if (!Char.IsLetterOrDigit(ch))
                    spsymbols++;
                else if (ch == 'a' || ch == 'e' || ch == 'i' || ch
== 'o' || ch == 'u')
                    vowels++;
                else
                    consonant++;
            }
            String str = "There are \n" + vowels + "-Vowels\n" +
digits + "-Digits\n" + spaces + "-Spaces\n" + spsymbols + "-Special
Symboles\n Found in this Text";
            MessageBox.Show(str);
        }
```

- In this function we have declared variables vowels, spaces, digits, spsymbols and consonant and initialize them as 0. We have declared one more variable 'i' to run a loop. We are starting the loop from 0 to length of the string, and fetched one by one character into the character variable 'ch'. We are evaluating character, if it is ' ' (space) then spaces variable will be incremented, if it is digit then digits variable will be incremented and so on. To do this we have used some of the methods of Char class such as IsDigit, isletterOrDigit and so on. To know the character is vowel or not we are comparing char with a,e,i,o,u using OR operator by converting the character into the lowercase.
- The output of the program is given to the user, by using Messagebox.Show method as shown in the following figure:



Program: 6
Design and implement a Tic Tack Toe game (Two Player).

- Create a new project using Visual Studio called Unit1_Q6, and design the interface as shown in the figure given below using 3 Labels and 9 Buttons.



- Set the Text property for the form is "Tic Tack Toe:", also set the Font property to Size-10 and Bold-True. Place a label on the top of the form and set the Text property to "Tic Tack Toe". Arrange 9 buttons as shown in the figure and keep its Text property blank (null). Place two more labels at the bottom size and set its Text property to "Player1 : X" , and "Player2 : 0".
- Double click on the button1. It will create and open a subroutine called 'button1_Click()'. We want to execute the same code written on the click event of the button1 to all 9 buttons. So select the second button go to property window, click on the event icon, select click event and select 'button1_Click() subroutine.

- Repete the same procedure for all 9 buttons, so that user clicks on any button, but same subroutine button1_click() will be execute.
- Declare a Boolean variable 'Player' in the general declaration. By default it will be initialized with 'false'. If Player=false means its turn to Player1 and its sign is 'X'. When player1 will play we will execute the code like *Player=!Player;* which will simply negate the value of Boolean variable player and now variable Player will

becomes true (!False=True). When Player is true then it's a time to play for player2 and sign for player2 is 0.

- After each play we will check for the winner by calling subroutine 'checkwinner'. If any player wins then we will print the message for winner and clear the game by calling subroutine 'cleargame ()'. This subroutine will clear the Text of all 9 buttons and set the Player variable to false so that we get the game exactly in same state as it was in the beginning.
- **The complete code for Tic Tack Toe game is given bellow:**

```
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;

namespace Unit1_Q6
{
    public partial class Form1 : Form
    {
        Boolean Player;
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)

        {
            if (((Button)sender).Text != "X" &&
((Button)sender).Text != "0")
            {
                if (Player == false)
                ((Button)sender).Text = "X";
            else
                ((Button)sender).Text = "0";
            checkwinner();
            Player = !Player;
            }
        }
        private void checkwinner()
        {
```

```
            string smb;
            string msgplayer;
            if (Player == false)
            {
                smb = "X";
                msgplayer = "Player1";
            }
            else
            {
                smb = "0";
                msgplayer = "Player2";
                if (button1.Text == smb && button2.Text == smb &&
button3.Text == smb)
                {
                    MessageBox.Show(msgplayer + " Win!!!");
                    cleargame();
                }
                else if (button4.Text == smb && button5.Text ==
smb && button6.Text == smb)
                {
                    MessageBox.Show(msgplayer + " Win!!!");
                    cleargame();
                }
                else if (button7.Text == smb && button8.Text ==
smb && button9.Text == smb)
                {

                    MessageBox.Show(msgplayer + " Win!!!");
                    cleargame();
                }
                else if (button1.Text == smb && button4.Text == smb
&& button7.Text == smb)
                {
                    MessageBox.Show(msgplayer + " Win!!!");
                    cleargame();
                }
                else if (button2.Text == smb && button5.Text == smb
&& button8.Text == smb)
                {
                    MessageBox.Show(msgplayer + " Win!!!");
                    cleargame();
                }
                else if (button3.Text == smb && button6.Text
== smb && button9.Text == smb)
```

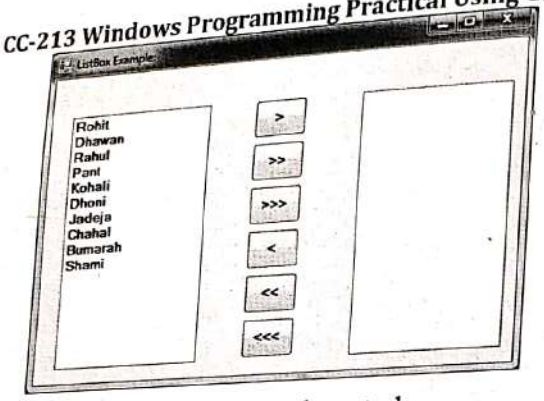


```
                {
                    MessageBox.Show(msgplayer + " Win!!!");
                    cleargame();
                }
                else if (button1.Text == smb && button5.Text == smb
&& button9.Text == smb)
                {
                    MessageBox.Show(msgplayer + " Win!!!");
                    cleargame();
                }
                else if (button3.Text == smb && button5.Text == smb
&& button7.Text == smb)
                {
                    MessageBox.Show(msgplayer + " Win!!!");
                    cleargame();
                }
                else
                {
                }
            }
        }

        private void cleargame()

        {

            foreach (var control in this.Controls)
            {
                if (control.GetType() == typeof(Button))
                {
                    ((Button)control).Text = "";
                }
            }
            Player =false;
        }
    }
}
```

Program: 7

Write a program to transfer an item from First Listbox to Second Listbox and from Second Listbox to First.

➤ Create a new project in the Visual Studio called Unit1_Q7 and design the following interface design using 2 listBoxes and 6 Buttons for form1 as shown in figure.



➢ Set the following properties for form and each control:

Object	Property	Value	Explanation
Form	Text	ListBox Example:	Shows ListBox Example: in the titlebar
	StartPosition	CenterScreen	Form opens in the center of the screen
	Font -> Size	10	
	Font -> Bold	True	
ListBox1	SelectionMode	MultiSimple	Allow user to select multiple items
ListBox2	SelectionMode	MultiSimple	Allow user to select multiple items
Button1	Text	>	Caption for Button1
Button2	Text	>>	Caption for Button2
Button3	Text	>>>	Caption for Button3
Button4	Text	<	Caption for Button4
Button5	Text	<<	Caption for Button5
Button6	Text	<<<	Caption for Button6

- Double click on the form1, which will create and open a subroutine form1_load and write the following code into it. It will populate the name of the Indian create players in the listbox1.

```
private void Form1_Load(object sender, EventArgs e)
    {
        listBox1.Items.Add("Rohit");
        listBox1.Items.Add("Dhawan");
        listBox1.Items.Add("Rahul");
        listBox1.Items.Add("Pant");
        listBox1.Items.Add("Kohali");
        listBox1.Items.Add("Dhoni");
        listBox1.Items.Add("Jadeja");
        listBox1.Items.Add("Chahal");
        listBox1.Items.Add("Bumarah");
        listBox1.Items.Add("Shami");
    }
```


➢ Double click on the button1 that is ">", it will create and open the subroutine button1_Click in the Code view. The main idea to place this button is user is selecting any one element from the Listbox1 and when user presses this button; selected element should be moved from Listbox1 to listbox2. If user has not selected any item in the listbox1 then error message is generated. Write the following code in the subroutine button1_Click to perform this task.

```
private void button1_Click(object sender, EventArgs e)
    {
        if (listBox1.SelectedIndex >= 0)
        {
            listBox2.Items.Add(listBox1.SelectedItem);
            listBox1.Items.Remove(listBox1.SelectedItem);
        }
        else
            MessageBox.Show("Select any one item in the ListBox1
and then click in this Button");
    }
```

➢ Now go to design mode and double click on the button2 that is ">>", it will create and open the subroutine called button2_Click(). The main idea to place this button is user will select more than one item from the listbox1 and click on the button ">>", so that all selected items will be moved from listbox1 to listbox2. User can select multiple item from the listbox1, because of we have selected SelectionMode property to 'MultiSimple'. User can select individual multiple items by holding Ctrl key and clicking on individual items. Similarly user can select range by using Shift key. The following code will move all selected items from the listbox1 to listbox2.

```
private void button2_Click(object sender, EventArgs e)
    {
        int i,n;
        n = listBox1.SelectedItems.Count;
        for (i = 0; i < n; i++)
        {
            listBox2.Items.Add(listBox1.SelectedItems[0]);
            listBox1.Items.Remove(listBox1.SelectedItems[0]);
        }
    }
```

➢ Again come back to design mode and double click on third button that is ">>>". This will create and open another subroutine called button3_Click (). This event will move all items (no selection is needed) from listbox1 to listbox2. Add the following code to the button3_Click () event.

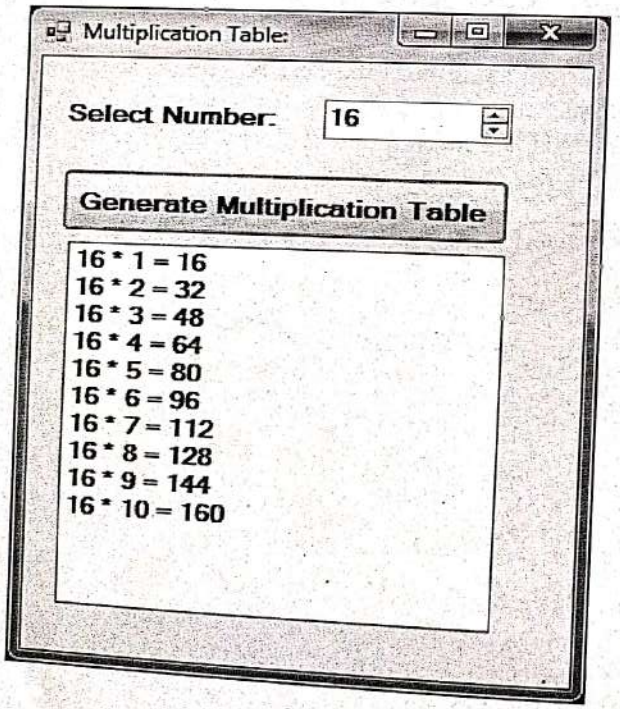
```
private void button3_Click(object sender, EventArgs e)
        {
            int i, n;
            n = listBox1.Items.Count;
            for (i = 0; i < n; i++)
            {
                listBox2.Items.Add(listBox1.Items[0]);
                listBox1.Items.Remove(listBox1.Items[0]);
            }
        }
```

- Button4, Button5 and Button6 have same functionality as Button1, Button2 and Button3. But they are used to transfer the list items from listbox2 to listbox1. I this you can do it easily by following the code we have discussed above.

Program: 8

Print multiplication table into Listbox. For multiplication take value using Numeric up down.

- Create a new project called "Unit1_Q8" and design the form as shown in the following figure using a Label, a NumericUpdown control, a Button and a listBox.



- Set the following properties of different controls as shown in the table given below:

Object	Property	Value	Explanation
Form	Text	Multiplication Tabel:	Shows multiplication Table in the titlebar of the form
	StartPosition	CenterScreen	Form opens in the center of the screen
	Font -> Size	10	
	Font -> Bold	True	
NumericupDown	Minimum	1	Set minimum value to 1
	Maximum	25	Set maximum value to 25
Label	Text	Select Number:	Caption for Label
Button	Text	Generate Multiplication Table:	Caption for Button

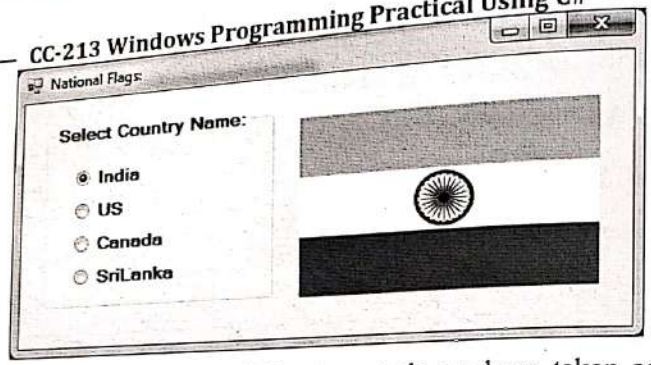
- After designing the form and setting all properties double click on the button which will create and open the subroutine called button1_Click(). Add the following code to this subroutine to generate the multiplication table in the listBox.

```
private void button1_Click(object sender, EventArgs e)
        {
            int i,num;
            num = Convert.ToInt16 ( numericUpDown1.Value);
            listBox1.Items.Clear();
            for (i = 1; i <= 10; i++)
            {
                listBox1.Items.Add( num + " * " + i + " = " +
num*i);
            }
```

Program: 9

Take 3 Radio buttons showing the name of 3 Countries. Load the image of the Flag of the country selected by the user from the given Radio buttons in the Picture box.

- Create a new project in Visual Studio called Unit1_Q9 and design the following interface for form1. Take a groupBox control and place 3 or 4 radio buttons inside the groupBox. Take a pictureBox at right side of the form.



- Set the following properties for different controls we have taken as shown in the following table:

Object	Property	Value
Form	Text	National Flags:
	StartPosition	CenterScreen
	Font -> Size	10
	Font -> Bold	True
GroupBox	Text	Select Country Name:
RadioButton1	Text	India
RadioButton2	Text	US
RadioButton3	Text	Canada
RadioButton4	Text	SriLanka
PictureBox1	SizeMode	StretchImage

- Now download or create the images (using paintbrush or other application) for different countries given in the groupBox as save all the images in a folder called flag. Double click on the radiobutton1. It will create and open the subroutine called *'radioButton1_CheckedChanged'* and the following code to the subroutine:

```
if(radioButton1.Checked ==true)
                pictureBox1.Image =
Image.FromFile("c:\\users\\kamesh\\documents\\visual studio
2010\\Projects\\Unit1_Q9\\Unit1_Q9\\Flags\\in.png");
```

- The same process you repeat for all the radioButtons, make sure you have to change the path. The complete code is given below:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
```



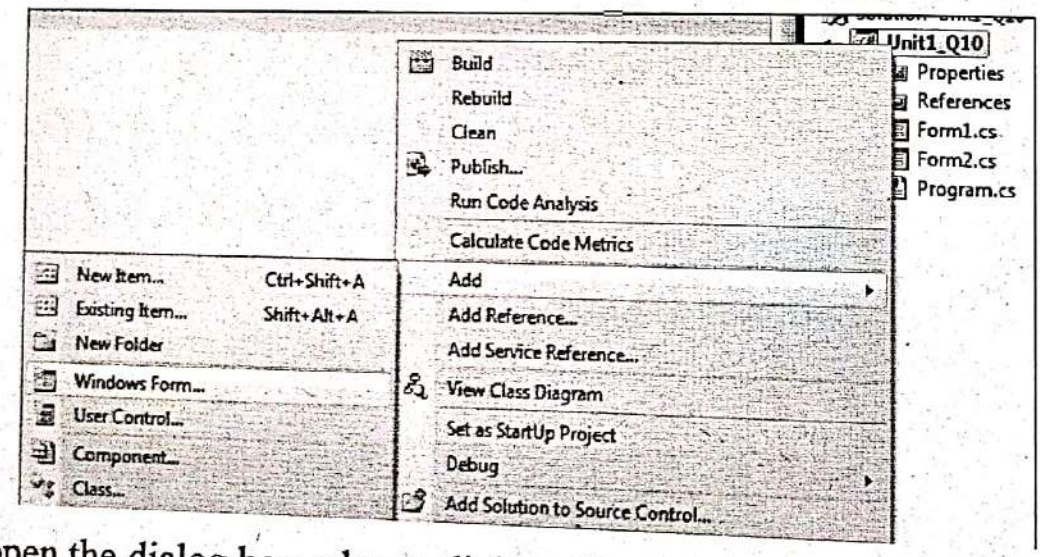
```
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Unit1_Q9
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void radioButton1_CheckedChanged(object sender,
EventArgs e)
        {
            if(radioButton1.Checked ==true)
                pictureBox1.Image =
Image.FromFile("c:\\users\\kamesh\\documents\\visual studio
2010\\Projects\\Unit1_Q9\\Unit1_Q9\\Flags\\in.png");
        }

        private void radioButton2_CheckedChanged(object sender,
EventArgs e)
        {
            if (radioButton2.Checked == true)
                pictureBox1.Image =
Image.FromFile("c:\\users\\kamesh\\documents\\visual studio
2010\\Projects\\Unit1_Q9\\Unit1_Q9\\Flags\\us.png");
        }

        private void radioButton3_CheckedChanged(object sender,
EventArgs e)
        {
            if (radioButton3.Checked == true)
                pictureBox1.Image =
Image.FromFile("c:\\users\\kamesh\\documents\\visual studio
2010\\Projects\\Unit1_Q9\\Unit1_Q9\\Flags\\ca.png");
        }

        private void radioButton4_CheckedChanged(object sender,
EventArgs e)
        {
```

```
            if (radioButton4.Checked == true)
                pictureBox1.Image =
Image.FromFile("c:\\users\\kamesh\\documents\\visual studio
2010\\Projects\\Unit1_Q9\\Unit1_Q9\\Flags\\lk.png");
        }
    }
}
```

Program: 10

Take a Timer control which will delay to load MainForm by 10 seconds. Show the progress bar in the wait time. Also use Time control to scroll a label having text "Gujarat University", also take two more labels to show date and time on the tick event of the timer.

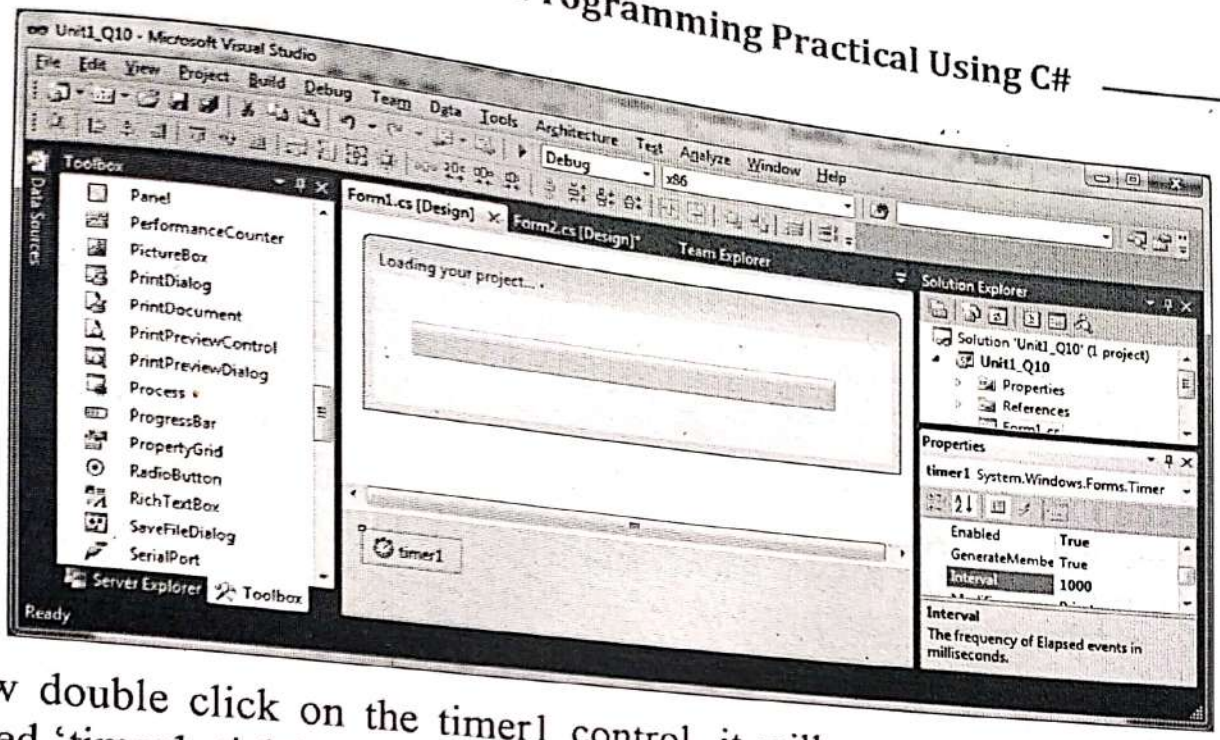
- Create a new project in the Visual Studio called 'Unit1_Q10'. By default, your project will have only one form that is form1. Add another form called Form2. To do this right click on the solution name in the solution explorer, Select option 'Add' and then 'new Windows form'.as shown in the figure.



- It will open the dialog box, please click on the OK button of the dialogBox and a new form (Form2) will be added to your project. Now place a progressBar and timer control to form1 and set the following properties to form1.

Object	Property	Value
Form1	Text	Loading your Project....
	ControlBox	False
Timer	Enabled	True
	Interval	1000

- After setting the properties your form should look like this:



Now double click on the timer1 control, it will create and open a new subroutine called 'timer1_tick()'. Write the following code to this subroutine.

```
private void timer1_Tick(object sender, EventArgs e)
        {
            if (progressBar1.Value < 100)
            {
                progressBar1.Value += 10;
            }
            else
            {
                timer1.Enabled = false;
                Form2 frm = new Form2();
                frm.Show();
                this.Hide();
            }
        }
```

- Minimum value of the progressBar is 0 and maximum value is 100. We have set the Interval property of the timer is '1000'. That means after each 1000 milli second (=1 Second) Tick event of timer will get executed. This will increase the value of the progressbar by 10. So, 1 second 10, 2 seconds 20 and so on. When the value of the progressBar becomes 100, we disabling the timer and opening the form2. Along with that we are also hiding the form1.
- Now place a label and a timer control on the form2, and set the following properties:

Object	Property	Value
Form2	Text	Marquee
	Size	1261, 658
Timer	Enabled	True
	Interval	100
Label	Text	Gujarat University

- In this form we have set the Interval property 100, that means (1000/100=) 10 times tick event will be executed in 1 second. Double click on the timer, it will create and open a subroutine called timer1_Tick() and write the following code:

```
private void timer1_Tick(object sender, EventArgs e)
    {
        label1.Left = label1.Left + 10;
        if (label1.Left > 1000)
        {
            label1.Left = 0;
        }
    }
```

- 10 times in each second we are increasing the value of Left property by 10 in a second. So, your label will move toward right side. When the Left property exceeds then 1000, we are again setting the Left property of the label to 0. This will give a marquee effect to the label on the Form2.

☺ ☺ ☺ ☺ ☺

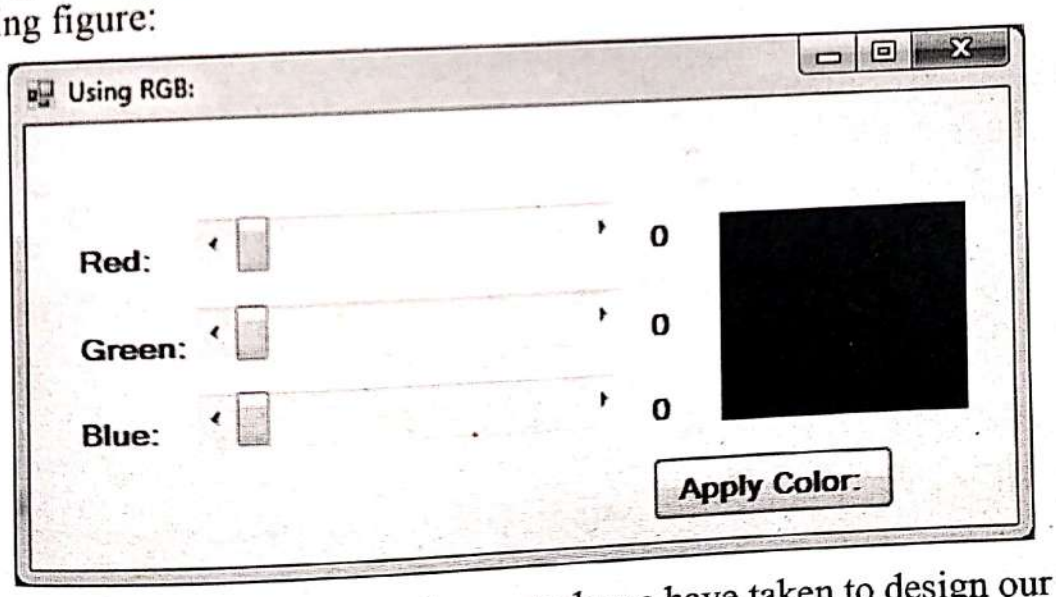
Programs of Unit-2

In this chapter we will discuss programs Scrollbar, RichTextbox, FontDialogbox, ColorDialogbox, OpenFileDialogbox, SaveFileDialogbox etc. We will also discuss how to read and write Text and Binary files and concept of Multiple Document Interface (MDI).

Program: 1

Take 3 Scrollbars indicating values of Red, Green and Blue colors from 0 to 255. Take 3 labels to show the values of the scroll bars. Depending upon values selected by the user with the help of scrollbars make a color and shows it in the picture box. Place a button having text 'Apply Color', and on the click event of the button apply that color to the form.

- Create a new Visual Studio project called 'Unit2_Q1' and design the interface for form1 by taking 3 scrollbars, 6 labels, a pictureBox and a button as shown in the following figure:



- Set the following properties for the controls we have taken to design our form.

Object	Property	Value
Form	Text	Using RGB:
	Font->Size	10
	Font->Bold	True
Label1	Text	Red
Label2	Text	Green
Label3	Text	Blue
Label4	Text	0

Label5	Text	0
Label6	Text	0
hScrollBar1	Maximum LargeChange	255 1
hScrollBar2	Maximum LargeChange	255 1
hScrollBar3	Maximum LargeChange	255 1
PictureBox	Size	108, 103
Button	Text	Apply Color:

- Double click on the Scrollbar1, which will create and open subroutine 'hScrollbar1_Scroll'. Now go back to the design view and select hScrollbar2. Go to the property window and click on the Event icon. From the Scroll property select 'hScrollbar1_Scroll' subroutine, and in the same way select hScrollbar3 assign the same subroutine to the Scroll event, As we want to execute same subroutine for the Scroll event to all 2 hScrollbar. Now write the following code to the subroutine 'hScrollBar1_Scroll' as well as 'button1_Click' event.

```
private void hScrollBar1_Scroll(object sender, ScrollEventArgs e)
{
int r, g, b;
r = hScrollBar1.Value;
g = hScrollBar2.Value;
b = hScrollBar3.Value;
label4.Text = Convert.ToString(r);
label5.Text = Convert.ToString(g);
label6.Text = Convert.ToString(b);
pictureBox1.BackColor = Color.FromArgb(255, r, g, b);
}

private void button1_Click(object sender, EventArgs e)
{
this.BackColor = pictureBox1.BackColor;
}
```

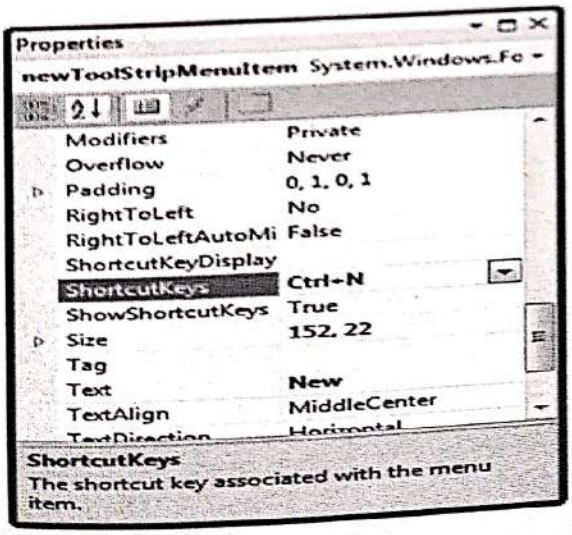
Program: 2

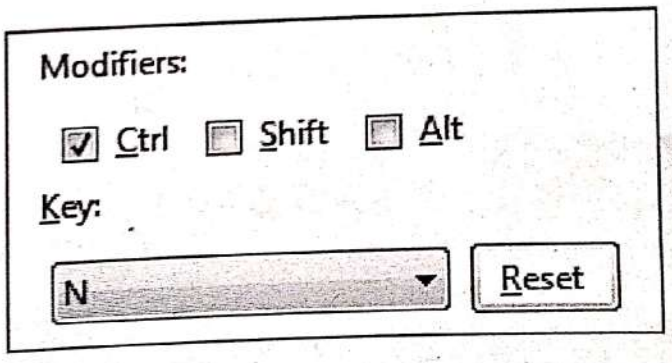
Implement Textpad application using Rich textbox. Make menus like File (New, Open Save, SaveAs and Exit), Edit (Cut, Copy, Paste, Undo, Redo), Format (Bold, Italic, Underline, Font, Color) etc. Use all common dialog controls and implement functionalities.

- Create a new project in the Visual Studio called 'Unit2_Q2'. In the Form1 place a MenuStrip control and a RichTextbox control from the ToolBox. Set the Dock property of the RichTextbox control to Fill and Design the Menu as per given hierarchy.

&File	&Edit	F&ormat
New	Cut	Bold
Open	Copy	Italic
Save	Paste	Underline
-	-	-
SaveAs	Undo	Font
-	Redo	-
Exit		Color

- Instead of writing File for the File menu we have written &File. This will provide access key to the key F. Runtime when user is pressing Alt key then user can see File in the menu strip and then suppose user is pressing 'F' then File menu will be opened. So user can press Alt+F to open the file menu. In the same way Edit menu has access key 'E' so to access Edit menu directly at run time user can press Alt+E. In the same way user can press Alt+O to open format menu.
- In the sub menu '-'indicate menu separator. Submenu Save and SaveAs will be separated by a line called menu separator. Now select submenu New and go to shortcut property and assign shortcut key Ctrl + N as shown in the figure given below.

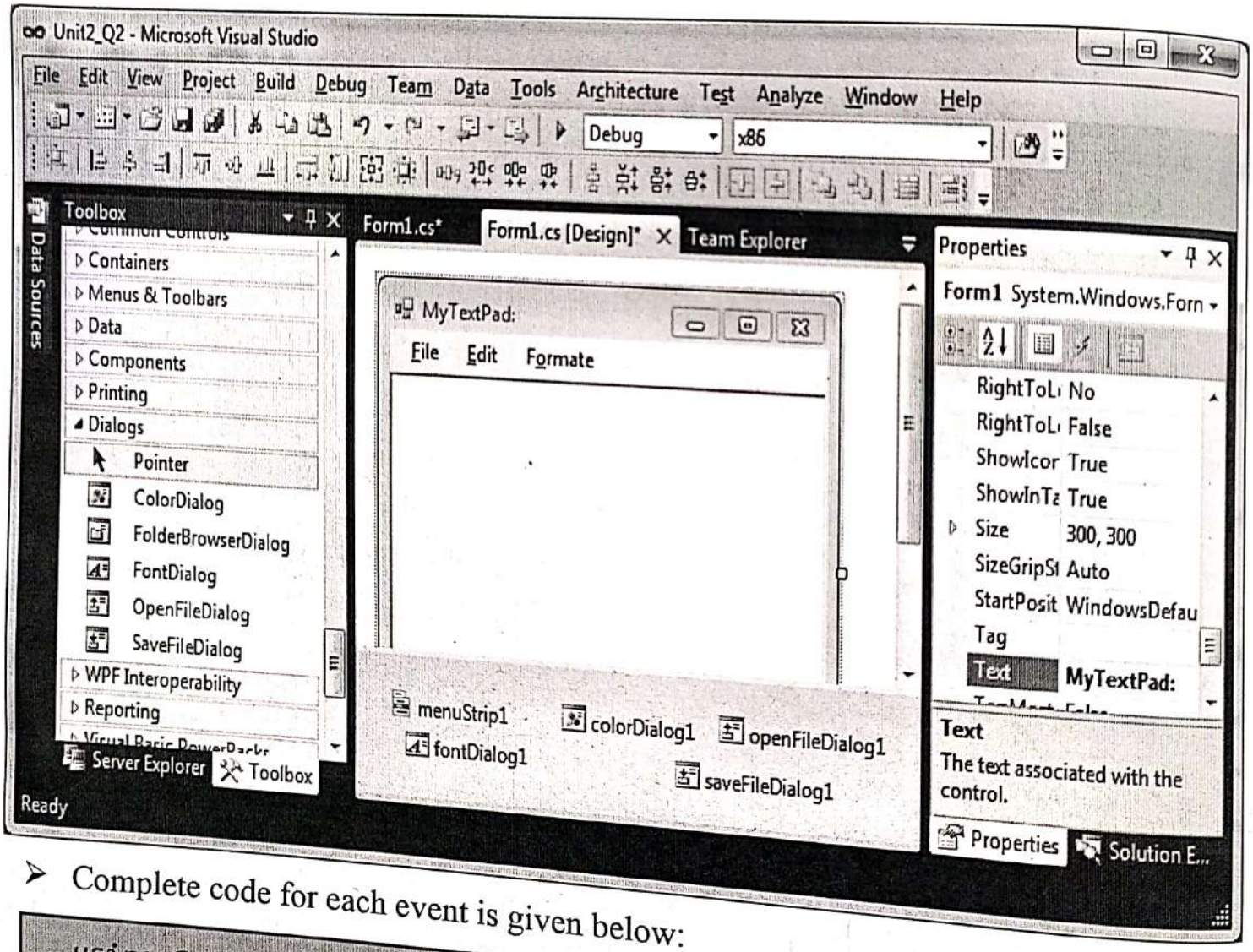




- In the same way give Ctrl+O to Open, Ctrl+S to Save, Ctrl +X to Exit menu, Ctrl + X to Cut menu Ctrl + C to copy menu, Ctrl + V to paste menu and so on.
- Once the menu has been design put several dialog boxes from the ToolBox as listed below to your form:
- ColorDialogBox
- FontDialogBox

- OpenFileDialogBox
- SaveFileDialogBox
- It is shown in the figure given below in the next page:

- Now each time you need to come to the design mode, and you have to double click on the menu to write the code. For example, if you double click on the New menu it will create and open the subroutine called 'newToolStripMenuItem_Click()' where you can write the code, which you want to execute on the click event of this menu item. In the same way, for each menu you need to write the code as given in the next page.



- Complete code for each event is given below:

```
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;
```

```
namespace Unit2_Q2
{
    public partial class Form1 : Form
    {
        string path;
        public Form1()
        {
            InitializeComponent();
        }

        private void newToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            richTextBox1.Clear();
            richTextBox1.Focus();
        }

        private void openToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            openFileDialog1.Filter = "All RichText Files|.rtf|All
Text Files|*.txt|All Files|*.*";
            if (openFileDialog1.ShowDialog()== DialogResult.OK)
            {
                path=openFileDialog1.FileName;
                richTextBox1.LoadFile(path);
            }
        }

            private void saveToolStripMenuItem_Click(object
    sender, EventArgs e)
        {
            if (path == null)
            {
                saveFileDialog1.Filter = "All RichText
Files|.rtf|All Text Files|*.txt|All Files|*.*";
                if (saveFileDialog1.ShowDialog() == DialogResult.OK)
                {
                    path = saveFileDialog1.FileName;
                    richTextBox1.SaveFile(path);
                }
            }
            else
            {
```

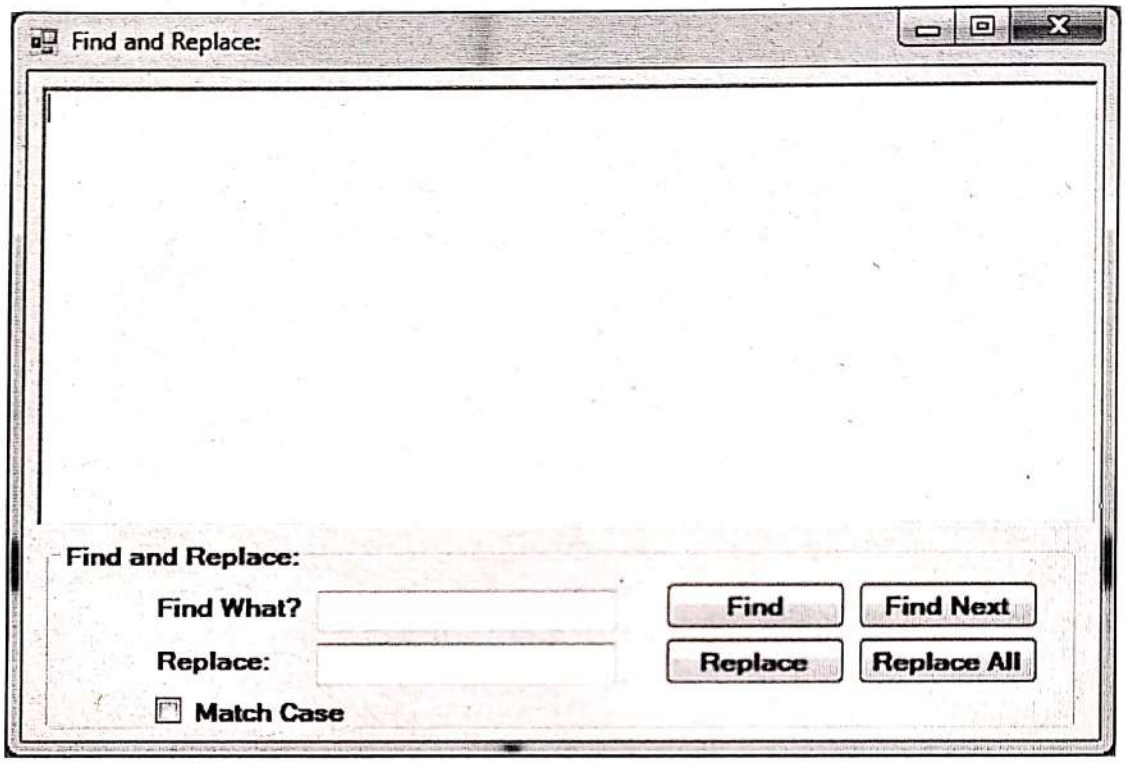
```
                richTextBox1.SaveFile(path);
            }
        }

        private void saveAsToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            saveFileDialog1.Filter = "All RichText Files|.rtf|All
Text Files|*.txt|All Files|*.*";
            if (saveFileDialog1.ShowDialog() == DialogResult.OK)
            {
                path = saveFileDialog1.FileName;
                richTextBox1.SaveFile(path);
            }
        }

        private void exitToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            this.Close();
        }
        private void cutToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            richTextBox1.Cut();
        }

        private void copyToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            richTextBox1.Copy();
        }

        private void pasteToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            richTextBox1.Paste();
        }
        private void undoToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            if (richTextBox1.CanUndo)
                richTextBox1.Undo();
```

```
        }
        private void redoToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            if (richTextBox1.CanRedo)
                richTextBox1.Redo();
        }
        private void boldToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            Font fnt = new Font(richTextBox1.SelectionFont,
FontStyle.Bold);
            richTextBox1.SelectionFont =fnt;
        }
        private void italicToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            Font fnt = new Font(richTextBox1.SelectionFont,
FontStyle.Italic);
            richTextBox1.SelectionFont = fnt;
        }

        private void underLineToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            Font fnt = new Font(richTextBox1.SelectionFont,
FontStyle.Underline);
            richTextBox1.SelectionFont = fnt;
        }
        private void fontToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            if (fontDialog1.ShowDialog() == DialogResult.OK)
                richTextBox1.SelectionFont = fontDialog1.Font;

        }
        private void colorToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            if (colorDialog1.ShowDialog() == DialogResult.OK)
                richTextBox1.SelectionColor = colorDialog1.Color;

        }
    }
}
```

Program: 3

Take a Rich Textbox and implement Find, FindNext, Replace and ReplaceAll functionalities.

- Create a new project in the Visual Studio called 'Unit2_Q3', and design the following interface for form1 by taking a richTextBox, two textBoxes (For Find and Replace), a checkBox (Form Match Case), a goupBox and 4 buttons:



- Set the following properties for the different controls we have taken to design the form interface.

Object	Property	Value
Form	Name	Editor
	Font->Size	10
	Font->Bold	True
GroupBox	Text	Find and Replace:
Label1	Text	Find What?
Label2	Text	Replace:
Textbox1	Name	txtFind
Textbox2	Name	txtReplace
Checkbox1	Name	chkMatchCase
Button1	Name	cmdFind
Button2	Name	cmdFindNext
Button3	Name	cmdReplace
Button4	Name	cmdReplaceAll

- Now declare variable pos for determining the position, of type int in the general declaration. Write the code to the click event for cmdFind, cmdFindNext, cmdReplace, and cmdReplaceAll as given below:

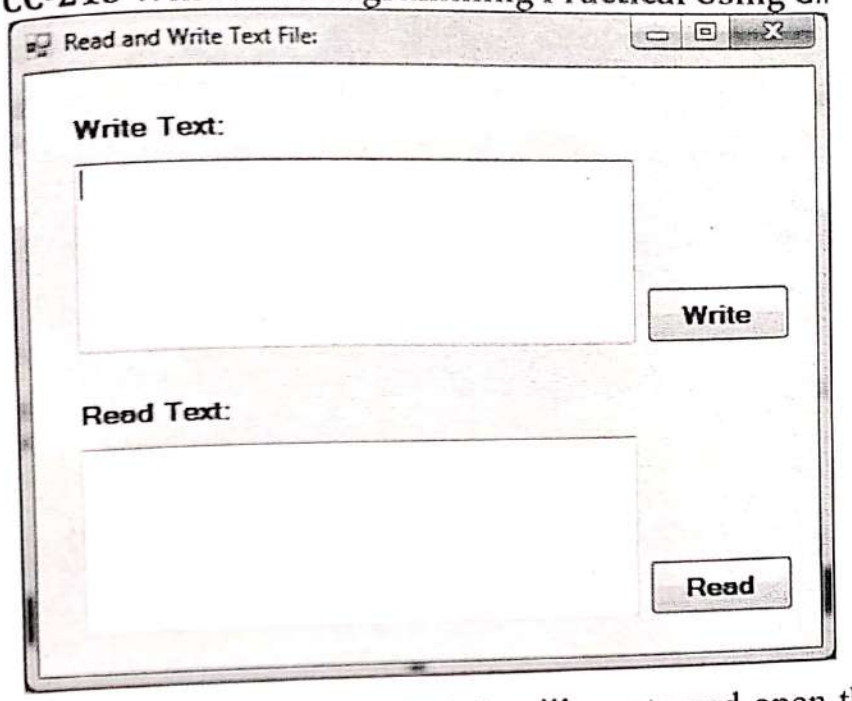
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Unit2_Q3
{
    public partial class Form1 : Form
    {
        int pos;
        public Form1()
        {
            InitializeComponent();
        }

        private void btnFind_Click(object sender, EventArgs e)
        {

            if (chkMatchCase.Checked == true)
                pos=Editor.Find(txtFind.Text, 0, Editor.Text.Length
- 1, RichTextBoxFinds.MatchCase);
            else
                pos = Editor.Find(txtFind.Text, 0,
Editor.Text.Length - 1, RichTextBoxFinds.WholeWord);
            if (pos >= 0)
                Editor.Select(pos, txtFind.Text.Length);
            else
                MessageBox.Show("The text you are looking for is not
Found:", "Error!!!");
        }

        private void btnReplace_Click(object sender, EventArgs e)
        {
            Editor.SelectedText = txtReplace.Text;
        }
```

```
private void btnFindNext_Click(object sender, EventArgs e)
        {
            if (chkMatchCase.Checked == true)
                pos = Editor.Find(txtFind.Text, pos+1,
Editor.Text.Length - 1, RichTextBoxFinds.MatchCase);
            else
                pos = Editor.Find(txtFind.Text, pos+1,
Editor.Text.Length - 1, RichTextBoxFinds.WholeWord);
            if (pos >= 0)
                Editor.Select(pos, txtFind.Text.Length);

        }

private void btnReplaceALL_Click(object sender, EventArgs e)
        {
            pos = 0;
            btnFind.PerformClick();
            if (pos != -1)
            {
                btnReplace.PerformClick();
                while (pos != -1)
                {
                    btnFindNext.PerformClick();
                    if (pos != -1)
                        btnReplace.PerformClick();
                }
            }
        }
    }
}
```

Program: 4

Write a program to Read and Write Text file.

- Create a new project in the Visual Studio called 'Unit2_Q4', and design the interface for form1 as shown in the following figure using two labels, two textboxes and two buttons.



- Now double click on the button 'Write'. It will create and open the subroutine called 'button1_Click()'. In this subroutine we will create the object fs for 'FileStream' class which is located under namespace 'System.IO'. Then we will create another object called sw of class 'StreamWriter' as we want to write the file. To write the data into the file we will use write() method and we will pass Text method of textbox1. Finally we close the instances of fs and sw.
- In the sameway to read the file, first we will create the object fs of FileStream class. This time we will create the object sr of 'StreamReader' class, as we want to read the file. We will the method 'ReadToEnd()' of sr and return string value we are assigning to the textbox2.
- The complete code is given below:

```
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;

namespace Unit2_Q4
{
    public partial class Form1 : Form
    {

        public Form1()
```

```
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            System.IO.FileStream fs=new
System.IO.FileStream("C:\\Sem4\\myfile.txt",System.IO.FileMode.
OpenOrCreate );
            System.IO.StreamWriter sw=new
 System.IO.StreamWriter(fs);
            sw.Write(textBox1.Text);
            MessageBox.Show ("Data is written to the file
 successfully:","Info:");
            sw.Close();
            fs.Close();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            System.IO.FileStream fs=new
System.IO.FileStream("C:\\Sem4\\myfile.txt",System.IO.FileMode.
OpenOrCreate );
            System.IO.StreamReader sr= new
 System.IO.StreamReader(fs);
            textBox2.Text =sr.ReadToEnd();
            sr.Close();
            fs.Close();
        }
    }
}
```

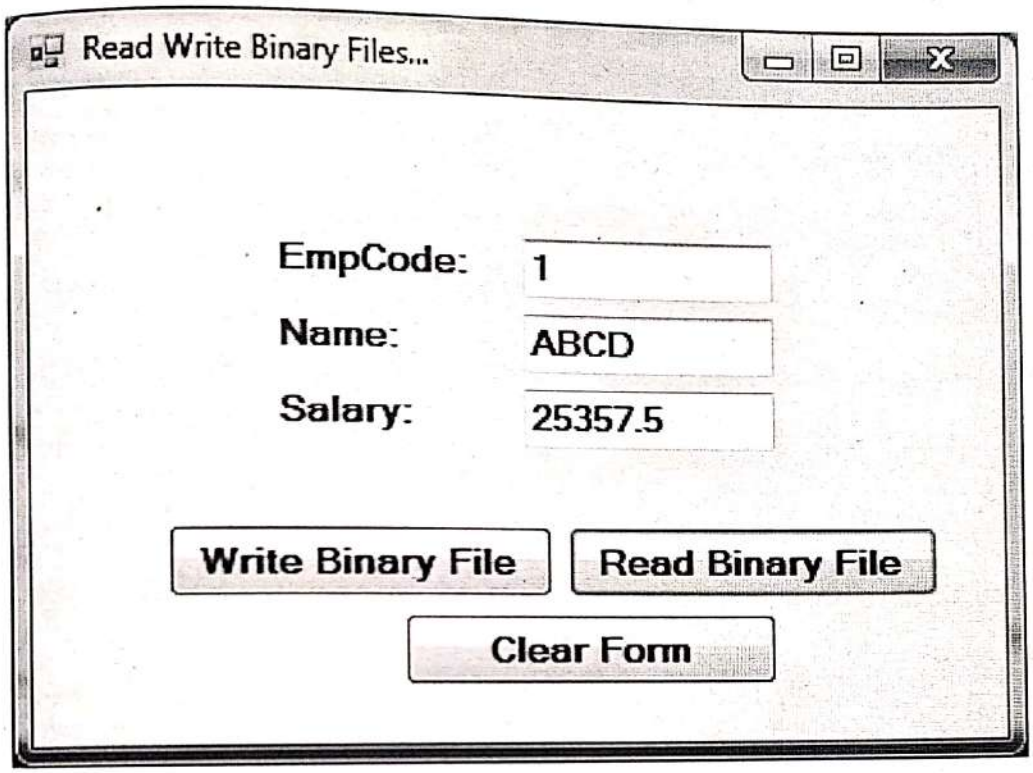
- Make a directory called Sem4 into your C: drive than run and test the application.
- Check you're C: (drive) and Sem4 folder which will have a text file called myfile.txt. You can open that file in the notepad editor to see the content you have typed in the Textbox1.

Program: 5

Write a program to Read and Write Binary file.

- Create a new project in Visual Studio called 'Unit2_Q5' and design the interface shown in the following figure using labels and Textboxes.

- Take two buttons. First button will create a binary file and write some sample data into it, and second button will open the same binary file to read it and load the content into the respective textboxes.



- Now to write the data first we are creating an object fs of FileStream class which is in the namespace "System.IO". We have given a path for the file "C:\\Sem4\\myfile.bin" and FileMode "OpenOrCreate". Make sure you need to create a folder named 'Sem4' into your C: drive. Then we are creating a object bw of type BinarayWriter. Using write() method of the object bw, first we are writing Int16 (EmpCode), String (EmpName) and then Single(Salary). At last we are closing the objects fs and bw using Close() method.
- To read the file we have created the objects of FileStream that is fs and another object of BinaryReader that is br. Now we are following the same sequnce in which we have written the data. First will read Int16 using method ReadInt16(),.then we will read string using method ReadString() and in the last we will read single using ReadSingle() method of BinaryReader class. The data will store in the respective variable, and then we are showing the data in the respective textboxes. Finally we are closing the object of br and fs.
- The complete code is given below:

```
private void button1_Click(object sender, EventArgs e)
        {
            System.IO.FileStream fs=new
System.IO.FileStream("C:\\Sem4\\myfile.bin",System.IO.FileMode
```

```
        .OpenOrCreate);

                System.IO.BinaryWriter bw=new
 System.IO.BinaryWriter(fs);
                Int16 Empcode=Convert.ToInt16(textBox1.Text);
                String EmpName=textBox2.Text;
                Single sal=Convert.ToSingle(textBox3.Text);
                bw.Write(Empcode);
                bw.Write(EmpName);
                bw.Write(sal);
                bw.Close();
                fs.Close();
                MessageBox.Show("Data is written to the file
 Successfullty:","Success");
            }

        private void button3_Click(object sender, EventArgs e)
            {
                System.IO.FileStream fs=new
System.IO.FileStream("C:\\sem4\\myfile.bin",System.IO.FileMode
.OpenOrCreate);
                System.IO.BinaryReader br=new
 System.IO.BinaryReader(fs);
                Int16 EmpCode=br.ReadInt16();
                String EmpName=br.ReadString();
                Single sal=br.ReadSingle();
                textBox1.Text =Convert.ToString(EmpCode);
                textBox2.Text =EmpName;
                textBox3.Text =Convert.ToString(sal);
            }
 private void button2_Click(object sender, EventArgs e)
            {
                textBox1.Text ="";
                textBox2.Text ="";
                textBox3.Text ="";
            }
```

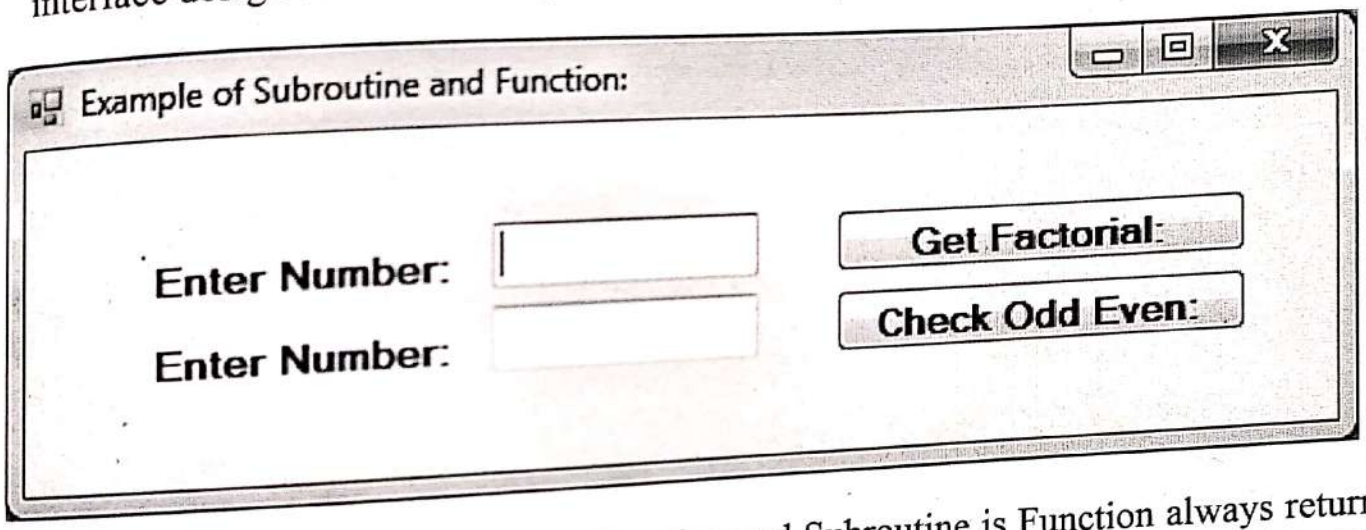
- To test the application, enter the data as shown in the figure. Click on the button "Write Binary File". Once you are getting success message, clear the form by click on the button "Clear Form". Once all the textboxes becomes clear then click on the button "Read Binary File" which will open the binary file, read the data and restore the data into the respective textboxes.

- You can see a file called myfile.bin in the C:\Sem4 directory. If you are trying to open this file into the notepad editor you can see some junk characters. Make sure you can't read the binary file as text file.

Program: 6

Accept no from user and perform following operations using user defined sub routines or functions:
1) Factorial of number
2) Odd/even

- Create a new project in the Visual Studio called "Unit2_Q6" and design the following interface design for form1 using 2 textboxes, 2 labels and 2 buttons.



- The main difference between the Function and Subroutine is Function always returns a value, where as subroutine never returns a value. So in the declaration of subroutine the return type must be 'void'.

- In this project on the click event of the button1, we will call a function called 'fact' which will return 'int' value, that is the factorial value of the number passed to it. Similarly, on the click event of the button2 we will call a subroutine 'chkEveOdd' which will check the value passed to the subroutine is even or odd and shows the result in the messagebox. Here is the complete code:

```
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;
```

```
namespace Unit2_Q6
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            int result;
            result = fact(Convert.ToInt16(textBox1.Text));
            MessageBox.Show("Factorial of " + textBox1.Text + " is =
" + Convert.ToString(result));
        }

public int fact(int num)
        {
            int ans,i;
            ans = 1;
            for (i = 1; i <= num; i++)
                ans = ans * i;
            return ans;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            chkEveOdd(Convert.ToInt16(textBox2.Text));
        }

        public void chkEveOdd(int num)
        {
            if (num % 2 == 0)
                MessageBox.Show(Convert.ToString(num) + " is an Even
Number:");
            else
                MessageBox.Show(Convert.ToString(num) + " is an Odd
Number:");
        }

    }
}
```

Program: 7

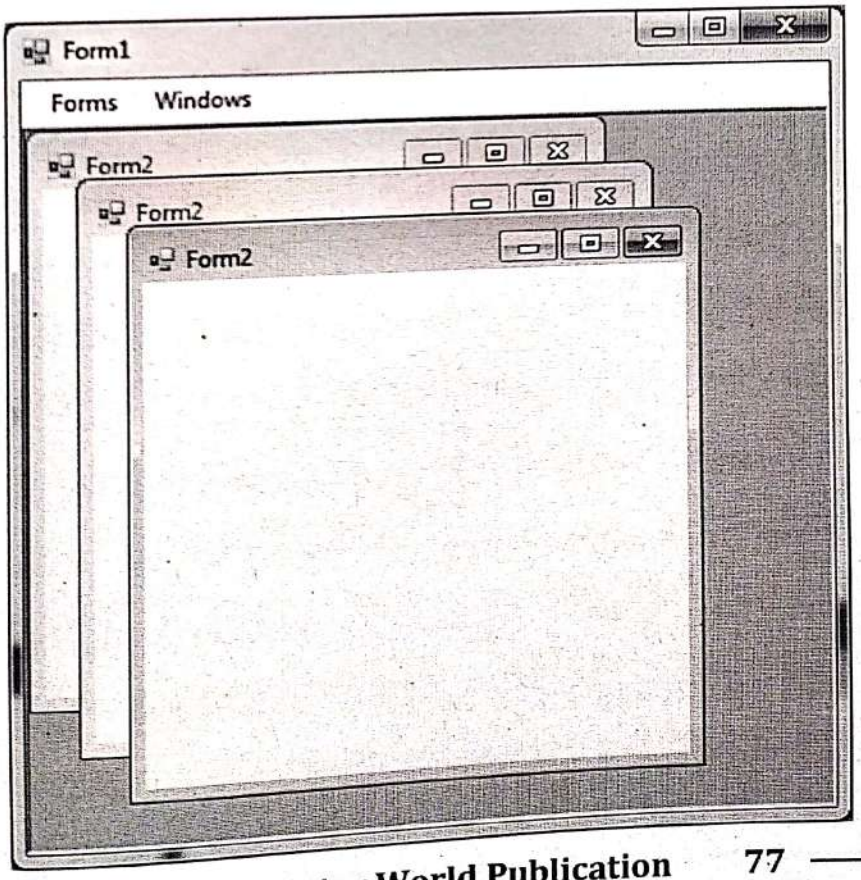
Create MDI form. It must have File menu with option open, Close and Exit. It should also have window menu to arrange the child forms like Tile Horizontal, Tile Vertical, Cascade and Arrange Icons.

- MDI stands for 'Multiple Document Interface'. It is a feature, allows user to open multiple document in the same window. For example, Notepad creates to separate windows to open two separate .txt files. It doesn't support MDI feature. But in the Microsoft Word you can open more than one document in the same window. It is an example of MDI. Another example of MDI is popular browser Google Chrome, which allows user to open multiple tabs in the same window.

- Create a project in the Visual Studio called "Unit2_Q7" and set the IsMDIContainer property of the Form to TRUE. Add two more forms called Form2 and Form3 to the project, by right click on the solution name in the solution explorer, Click on Add 'Windows form' option and then click on OK button of the dialog box.

- So, now our project has three forms. Form1 which will act as MDI container (MDI parent) and Form2 and Form3 will act as MDI child forms.

- Add a menu strip control to form1 (MDI Parent).



- See in the above image multiple instance of form2 (MDI Childs) we have opened in the form1(MDI Parent) and arranged them in Cascade ordered.
- Design the Menu as per given hierarchy for the form1 (MDI Parent).

• Forms	• Windows
➢ Form1 ➢ Form2 ➢ - ➢ Close ➢ - ➢ Exit	➢ Cascade ➢ Tile Horizontal ➢ Tile Vertical ➢ Arrange All

- Write the following code to the click event of each menu options as given below:

```
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;

namespace Unit2_Q7

{
    public partial class Form1 : Form

 {
        public Form1()
        {
            InitializeComponent();
        }
        private void form2ToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            Form2 frm =new Form2();
            frm.MdiParent = this;
            frm.Show();
        }

        private void form3ToolStripMenuItem_Click(object sender,
EventArgs e)
        {
```

```
            Form3 frm = new Form3();
            frm.MdiParent = this;
            frm.Show();
        }

        private void closeToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            Form frm = this.ActiveMdiChild;
            frm.Close(); // Close the active MDI child form
        }

        private void exitToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            this.Close(); //Close the MDI Parent
        }

        private void cascadeToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            this.LayoutMdi(MdiLayout.Cascade);
        }

    private void tileHorizontalToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            this.LayoutMdi(MdiLayout.TileHorizontal);
        }

    private void arrangeIconsToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            this.LayoutMdi(MdiLayout.ArrangeIcons);
        }

    private void tileVerticalToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            this.LayoutMdi(MdiLayout.TileVertical);
        }
    }
}
```

Program: 8

Create MDI form. It must have File menu with option Open, Close and Exit and one picture box. Allow users to open any picture using open dialog box, that picture should be displayed in the picture box.

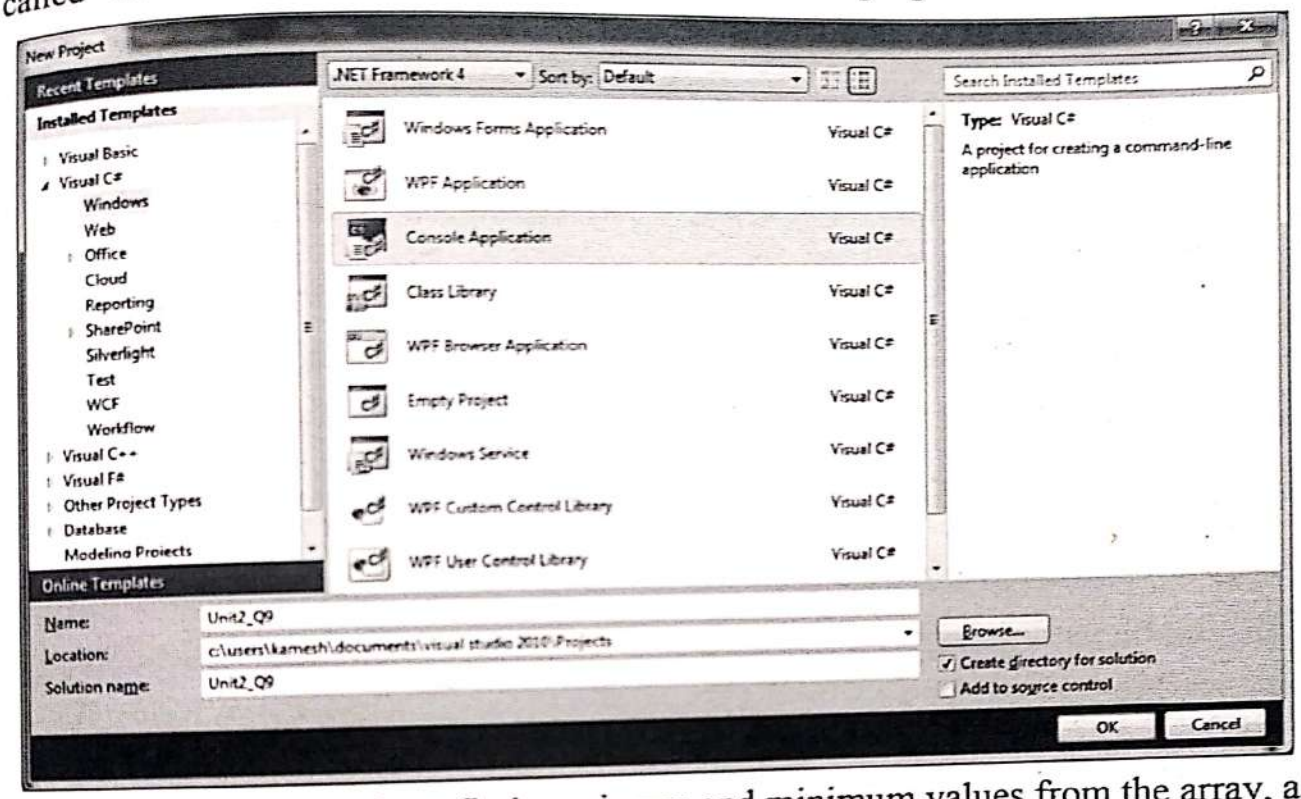
- Create a new project in the Visual Studio called "Unit2_Q8", and design the following interface for form1 by adding 'OpenFileDialog', 'MenuStrip' and a 'PictureBox' from the ToolBox. Create a menu File and submenus Open and Exit under it. Set the 'IsMDIContainer' property of the Form1 to true and sizeMode property to the Picturebox is 'StreachImage'.
- Write the following to Open sub menu item which will load the openFileDialog, and when you give a path to any jpg, gif or png, that image file will be loaded in the pictureBox of MDI Form, as shown in the following figure.

```
private void openToolStripMenuItem_Click(object sender, EventArgs e)
        {
            openFileDialog1.Filter="All JPG File | *.jpg | All GIF Files
| *.gif | All PNG Files | *.png";
            if(openFileDialog1.ShowDialog()==DialogResult.OK)
            {
                pictureBox1.Image
=Image.FromFile(openFileDialog1.FileName);
            }
        }

        private void exitToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            this.Close();
        }
```

Program: 9

Write function or subroutine to Find maximize, minimize value from an array and also to sort an array.

- Create a new project in the Visual Studio called 'Unit2_Q9' and select the template called 'Console Application' as shown in the following figure.



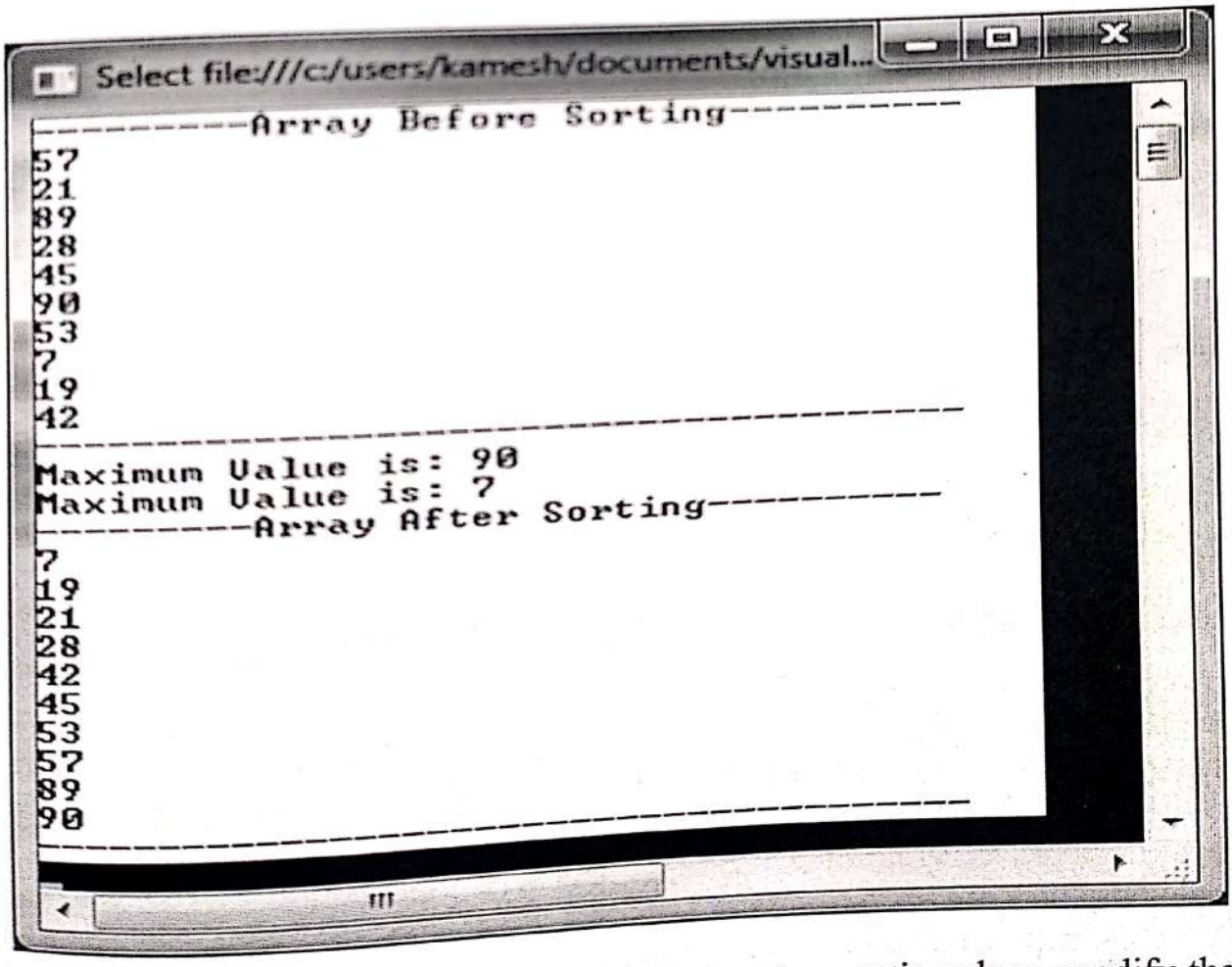
- Write the following code to find maximum and minimum values from the array, as well as sort the array.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace Unit2_Q9
{
    class Program
    {
        static void Main(string[] args)
        {
            /* Declaring and Initializing Array */
            int[] arr= new int[]{57,21,89,28,45,90,53,7,19,42};
        /* Creating the object of our class program */
            Program obj = new Program();
    Console.WriteLine("----Array Before Sorting----------");
            obj.printarray(arr);
    Console.WriteLine("-----------------------------------------");
```

```
        int max = obj.findmax(arr);
        Console.WriteLine("Maximum Value is: " + max);
        int min = obj.findmin(arr);
        Console.WriteLine("Maximum Value is: " + min);
        obj.sort(arr);
Console.WriteLine("-----Array After Sorting----------");
        obj.printarray(arr);
Console.WriteLine("-----------------------------------");
        Console.Read();
    }
    public int findmax(int[] x)
        {
            int max = x[0];
            int i;
        for (i = 1; i < 10; i++)
        {
            if (x[i] > max)
                max = x[i];
        }

        return max;
    }
    public int findmin(int[] x)
    {
        int min = x[0];
        int i;
        for (i = 1; i < 10; i++)
        {
            if (x[i] < min)
                min = x[i];
        }
        return min;
    }
    public void printarray(int[] x)
    {
        int i;
        for(i=0;i<10;i++)
        {
            Console.WriteLine(Convert.ToString(x[i]));
        }
    }
    public void sort(int[] x)
```

```
    {
        int i, j, tmp;
        for(i=0;i<10;i++)
            for(j=0;j<9-i;j++)
                if (x[j] > x[j + 1])
                {
                    tmp = x[j];
                    x[j] = x[j + 1];
                    x[j + 1] = tmp;
                }
    }

  }
}
```

- The output of the program is shown in the following figure:



- If you wish to take the value from the user rather than static values, modify the code of Main function as follows:

```
static void Main(string[] args)
        {
            /* Declaring and Initializing Array */
            int[] arr = new int[10];
            int i;
            for (i = 0; i < 10; i++)
            {
                Console.WriteLine("Enter Number:");
                arr[i] = Convert.ToInt16(Console.ReadLine());
              }

        /* Creating the object of our class program */
            Program obj = new Program();
    Console.WriteLine("-----Array Before Sorting---------");
            obj.printarray(arr);
    Console.WriteLine("-----------------------------------");

    int max = obj.findmax(arr);

 Console.WriteLine("Maximum Value is: " + max);
                int min = obj.findmin(arr);


Console.WriteLine("Maximum Value is: " + min);
          obj.sort(arr);
 Console.WriteLine("-----Array After Sorting----------");
          obj.printarray(arr);
 Console.WriteLine("-----------------------------------");
 Console.Read();
      }
```

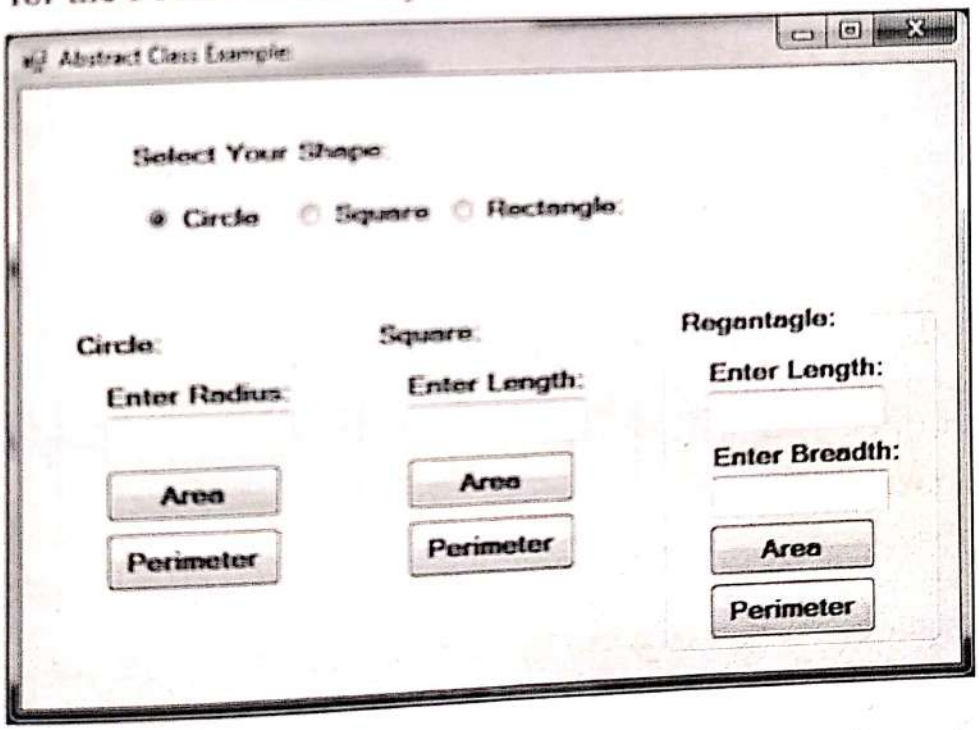
Programs of Unit-3

In this unit we will learn object oriented programming with C#.NET. We will learn how we can declare class, objects, properties and methods. We will also discuss Inheritance, Abstract class and virtual methods:

Program: 1

Create shape class as abstract class having area as must override function. Derive rectangle, triangle, square class based from this class.

- Create a project in the Visual Studio called "Unit3_Q1" and design the following Interface for the Form1 in the Project.



- To design the interface for the form1, take 4 groupBoxes as shown in the figure given above. Place 3 radioButtons for Circle, Square and Rectangle in the first groupBox. We know that to calculate area and perimeter for the circle we need only 'Radius' as an input factor from the user. So, in the groupbox2 we take radius as an input and we have placed two buttons for area and perimeter. Similarly to calculate area and perimeter we need only one input that is 'Length'. So in the Groupbox3 we are accepting only one input (Textbox) for length and two buttons for area and perimeter. To calculate area and perimeter for the rectangle, we need two inputs (length and breadth), so we have placed two textboxes in the GroupBox4 and two buttons for area and Perimeter.

- Now we will try to understand, what is Inheritance, Function (method) overriding abstract class and Virtual method?

Inheritance:
Inheritance is an important concept of OOP (Object Oriented Programming). It is th mechanism of Object Oriented Programming Languages like C#.NET, by which on class is allow, inheriting the features (properties means variables of the class an methods means functions of the class) of another class.

Important terminology of Inheritance:

- **Base Class:** The class whose features are inherited is known as base class (or super class or a parent class).
- **Derived Class:** The class that inherits the other class is known as derived class (o a sub class, extended class, or child class). The subclass can add its own fields an methods in addition to the base class properties and methods.
- **Reusability:** Inheritance supports the concept of "reusability". If we hav declared and defined a class with its properties (variables) and method (functions), and we want same properties and methods into another class, then w use inheritance. We are just inheriting another class which will act as a derive class and your original class will act as a base class. The derived class will hav properties and methods on the base class and we don't have to declare the sam variables and functions again into another class. That means inherence provide reusability of the code. The code done in the base class we can reuse it into th derive class. Make sure, in the case of inheritance the derive class has feature (properties and methods) of the base class, plus it can have its own additiona features (properties and methods).

Functions Overriding:
Function overriding is a feature that allows programmer to have a same function in derived class which is already present in the base class. A child class inherits the data members and member functions of base class, but when we want to override functionality in the derived class then we can use function overriding.

Abstract Class:
Abstract Class in Object oriented programming language like C#.NET can have an abstract method without body (virtual functions) and it can have methods with implementation also. Abstract keyword is used to create an abstract class and that must have at least one virtual method. We cannot create the instance (object) of the abstract class, but we can inherit the abstract class to derive new classes. Abstract class will act as a base class and it's derive classes must have abstract methods (virtual functions) with body (function overriding).

- In this example, we have to make a class called Shape and we know that every shape has area and perimeter. But we cannot calculate area and perimeter unless and until we are not aware the type of shape. Because for each type of shape different formula has been applied. If the shape is circle then area is: ($\pi * r^2$) and perimeter is $2*\pi*r$. If it is Square then formula to calculate area is l^2 and perimeter is 4*l. If the shape is rectangle then area is l*b and perimeter is 2 (l + b).
- So, here shape class becomes abstract class and has two methods area () and perimeter (). This methods are abstract methods and do not has its implementation. We will derive the classes called Circle, Square and Rectangle from the shape class and all the derived classes, will have implemented methods called area () and perimeter () [methods are overridden in the derived class].
- Now to make abstract shape class into your project, Right click to the solution name in the solution explorer, click on Add option and select class. In the dialog box give name to Shape and click on OK button of the dialog box. This will add a file Shape.cs file in your project. Write and modify the code in the Shape.cs file as shown below:

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

namespace Unit3_Q1
{
    abstract class Shape
    {
        public virtual int area()
        {
            return 0;
        }
        public virtual int perimeter()
        {
            return 0;
        }
    }
}
```

- Now add one more class file called Square.cs, same as Shape.cs file into your project and write following code to it.

```
sing System;
sing System.Collections.Generic;
sing System.Linq;
```

```
using System.Text;


namespace Unit3_Q1
{
    class Square : Shape
    {
        int l;
        public override int area()
        {
            return l * l;
        }
        public override int perimeter()
        {
            return 4 * l;
        }
      public Square (int length) //Constructor to initialize variable l
        {
            l = length;
        }
    }
}
```

- In the same way add two more class file called Circle.cs and Rectangle.cs and add following code to these class files.

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

namespace Unit3_Q1
{
    class Circle : Shape //Inheriting Circle class from Shape
    {
        int r;

        public override int area()
        {
            return Convert.ToInt16 (Math.PI * r * r);
        }
        public override int perimeter()
        {
```

```
            return Convert.ToInt16(2 * Math.PI * r);
        }
        public Circle(int radius) //Constructor
        {
            r = radius;
        }

    }
}
```

- And the code for Rectangle.cs is:

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

namespace Unit3_Q1
{
    class Rectangle:Shape
    {
        int l, b;
        public override int area()
        {
            return l * b;
        }
        public override int perimeter()
        {
            return 2 * (l+b);
        }
        public Rectangle(int Length, int Breadth) //Constructor with
two arguments
        {
            l = Length;

            b = Breadth;
        }
    }
}
```

- Now set the visible property to FALSE for GroupBox2, GroupBox3 and GroupBox4 and write the following code for the 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;

namespace Unit3_Q1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

      private void radioButton1_CheckedChanged(object sender,
EventArgs e)
        {
            if (radioButton1.Checked == true)
            {
                groupBox2.Visible = true;
                groupBox3.Visible = false;
                groupBox4.Visible = false;
            }
        }

      private void radioButton2_CheckedChanged(object sender,
EventArgs e)
        {
            if (radioButton2.Checked == true)
            {
                groupBox2.Visible = false;
                groupBox3.Visible = true;
                groupBox4.Visible = false;
            }
        }
private void radioButton3_CheckedChanged(object sender, EventArgs e)
        {
            if (radioButton3.Checked == true)
            {
                groupBox2.Visible = false;
```

```
                groupBox3.Visible = false;
                groupBox4.Visible = true;
            }
        }

private void btnCircleArea_Click(object sender, EventArgs e)
        {
            Circle c = new Circle(Convert.ToInt16(textBox1.Text));
            int area = c.area();
            MessageBox.Show ("Area of the Circle is: " +
Convert.ToString(area));
        }

private void btnCirclePeri_Click(object sender, EventArgs e)
        {
            Circle c = new Circle(Convert.ToInt16(textBox1.Text));
            int peri = c.perimeter();
            MessageBox.Show("Perimeter of the Circle is: " +
Convert.ToString(peri));
        }

private void btnSquareArea_Click(object sender, EventArgs e)
        {
            Square s = new Square(Convert.ToInt16(textBox2.Text));
            int area = s.area();
            MessageBox.Show("Area of the Square is: " +
Convert.ToString(area));
        }

private void BtnSquarePeri_Click(object sender, EventArgs e)
        {
            Square s = new Square(Convert.ToInt16(textBox2.Text));
            int peri = s.perimeter();
            MessageBox.Show("Perimeter of the Square is: " +
Convert.ToString(peri));
        }

private void BtnRectArea_Click(object sender, EventArgs e)
        {
            Rectangle r = new
Rectangle(Convert.ToInt16(textBox3.Text),
Convert.ToInt16(textBox4.Text));
            int area = r.area();
```

```
            MessageBox.Show("Area of the Rectangle is: " +
Convert.ToString(area));
        }


 private void btnRectPeri_Click(object sender, EventArgs e)
        {
            Rectangle r = new
Rectangle(Convert.ToInt16(textBox3.Text),
Convert.ToInt16(textBox4.Text));
            int peri = r.perimeter();
            MessageBox.Show("Perimeter of the Rectangle is: " +
Convert.ToString(peri));
        }
    }
}
```

Program: 2

Write a program to create class Person. Make at least three properties and one method "showdetail" of this class. Now inherit class Student and Faculty from class Person and override method "showdetail". Create objects of Student and Faculty class and call show detail function for both objects to show details in appropriate text boxes.

Create a new project using Visual Studio called "Unit3_Q2". Add a class file called Person and write the following code:

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

namespace Unit3_Q2
{
    abstract class Person
    {
        String name, City, Email;
        public virtual String showdetails()
        {
            String str;
            str = "Name:" + name + "\nCity:" + City + "\nEmail";
            return str;
```

```
        }
        protected void getDetails(String _name, String _City, String
_Email)
        {
            name = _name;
            City = _City;
            Email = _Email;
        }
    }
}
```

- Add another class called Faculty and inherit that class from the person. Add two more properties like Degree and Salary. Make sure Faculty class has now 5 attributes, 3 attributes comes from the base class person that is name, city and email and it has its own attributes like degree and salary. Write the following code for class faculty.

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

namespace Unit3_Q2
{
    class Faculty : Person //Every faculty is a Person so derived
from person class
    {
        /* Attributes Like name, city and email will
         * automatically come (derived) in this class */
        String Degree;
        Single Salary;
        public void getDetails (String _name, String _City, String
_Email, String _Degree, Single _Salary)
        {
            base.getDetails(_name, _City, _Email);
            Degree=_Degree;
            Salary =_Salary;
        }
        public override string  showdetails()
        {
            String str;
            str= base.showdetails();
            str = str + "\nDegree:" + Degree + "\nSalary:" +
Convert.ToString(Salary);
```

```
            return str;
        }
    }
}
```

- Now place a button on the form and double click it. On the click event of the button we will create an instance (object) of faculty class and use the methods of it as shown below:

```
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;

namespace Unit3_Q2
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Faculty f = new Faculty();
            f.getDetails("ABCD", "Ahmedabad",
"abcd@hotmail.com","MCA", 25050);
            MessageBox.Show(f.showdetails());
        }
    }
}
```

- In the same way you can create a class called Student, add two more attributes and create the instance same as faculty.

Program: 3

Write a program to implement the class Employee. Show Constructor Overloading.

Create a project in the Visual Studio and add a class file called employee. Declare 3 properties called EmpCode, Name and Salary and as shown in the following code.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace Unit3_Q3
{
    class Employee
    {
        int EmpCode;
        String EmpName;
        Single Salary;
        public Employee() //Default Constructor
        {
        }
        public Employee(int ECode) //One argument Constructor
        {
            EmpCode = ECode;
        }
        public Employee(int ECode, String EName)
        {
            EmpCode = ECode;
            EmpName = EName;
        }
        public Employee(int ECode, String EName,Single sal)
        {
            EmpCode = ECode;
            EmpName = EName;
            Salary = sal;
        }
        public String showdata()
        {
            String str;
            str = "EmpCode: " + EmpCode + "\nName: " + EmpName
+ "\nSalary: " + Salary;
            return str;
        }
    }
}
```

- Now place a button on the form1 as write the following code on the click event of the button:

```
private void button1_Click(object sender, EventArgs e)
        {
            Employee e1 = new Employee(); //Default Constructor
called
            MessageBox.Show(e1.showdata());


        Employee e2 = new Employee(5);//One argument constructor
called
            MessageBox.Show(e2.showdata());

            Employee e3 = new Employee(5,"ABCD");//Two arguments
constructor called
            MessageBox.Show(e3.showdata());
        }
```

Program: 4

Write a program to implement the class Book. Show Method Overloading.

- Create a new project in the Visual Studio called "Unit3_Q4" and place a button on the form. Add new class file to your solution by doing right click on the name of the solution in the solution explorer and click on the option Add -> class. Give the name "Book" to this new class which will add a new file to your solution called Book.cs. Now write the following code into the file.

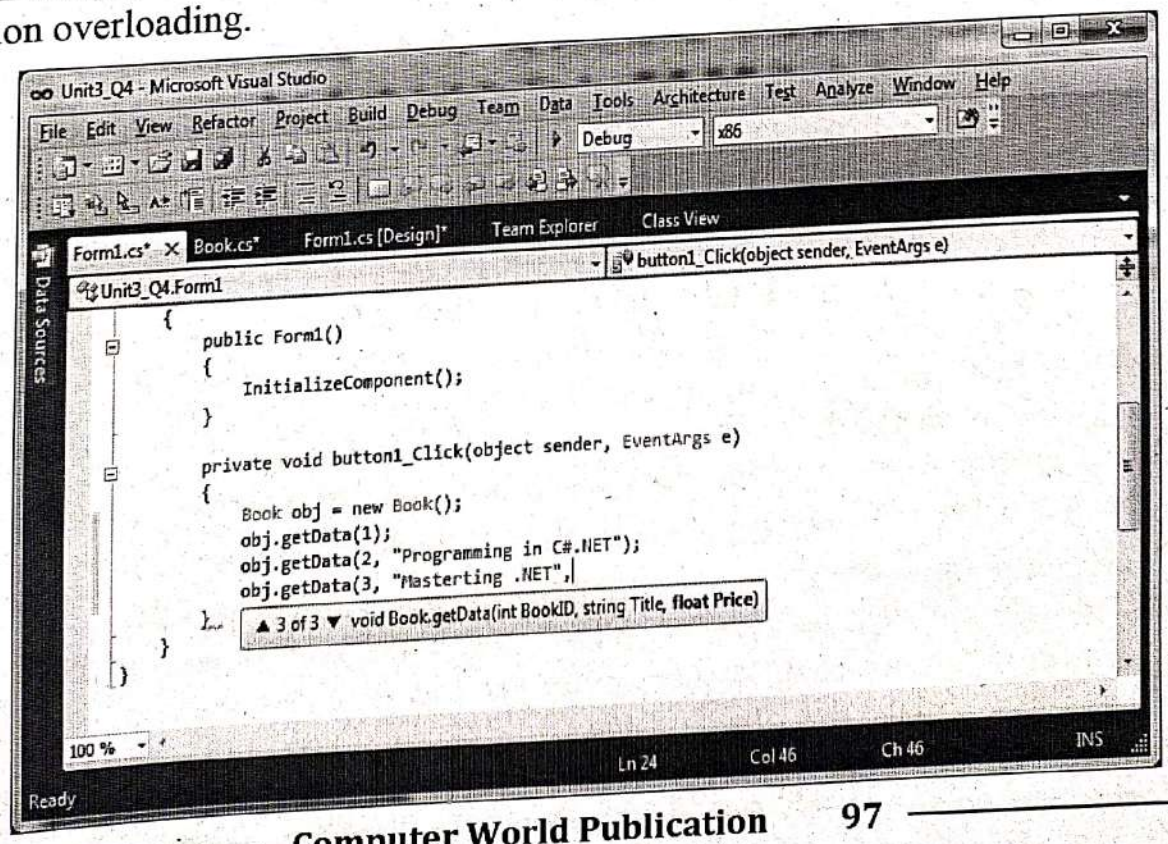
```
class Book
    {
        //Declaring Properties for the class
        int _BookID;
        String _Title;
        Single _Price;
        //Declaring overloaded methods for the class
        public void getData(int BookID)
        {
            _BookID = BookID;
        }
        public void getData(int BookID, String Title)
        {

    _BookID = BookID;
```

```
        _Title = Title;
    }
    public void getData(int BookID, String Title, Single Price)
    {
        _BookID = BookID;
        _Title = Title;
        _Price = Price;
    }
}
```

- Now on the click event of the button write following code:

```
private void button1_Click(object sender, EventArgs e)
        {
            Book obj = new Book();
            obj.getData(1);
            obj.getData(2, "Programming in C#.NET");
            obj.getData(3, "Masterting .NET", 575);
        }
```

- Now focus when you create the object of the Book class on the click event of the button you are getting (1 of 3 or 2 of3 or 3 of 3) as shown in the figure given below. That indicates there are three copies of the function 'getData()' exists. It is called function overloading.

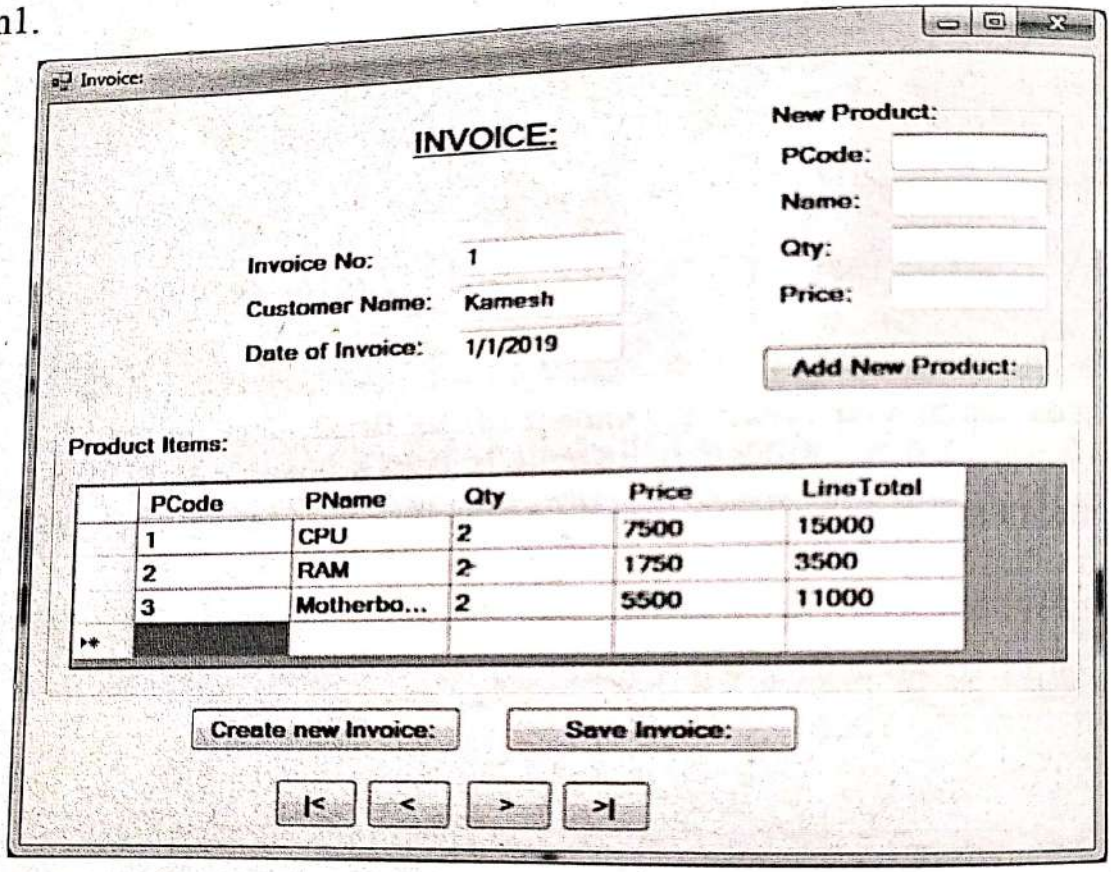


- Function with the same name but with different signature (different number of arguments or different types of arguments) is called function overloading.

Program: 5

Create an Invoice application in which user enters the customer name, description, unit price and quantity for the item ordered, then clicks the add item button. The application calculates the order total by multiplying the unit price by the quantity. And calculates a discount based on the order total. The user can then add another item to the order by using all information.

- Create a new project called "Unit3_Q5" and design the following interface for the form1.



- To design this form we have taken Textbox1, Textbox2 and Textbox3 for storing Invoice No, Customer Name, and Date of Invoice. We have taken a GroupBox1 for adding new product to invoice and inside it we have placed Textbox4, Textbox5, Textbox6 and Textbox7 to store Product code, Product Name, Qty and Price. We have placed a button inside this Group box, and on the click event of it we will add the product to the DataGridView1 which is placed inside another GroupBox2.
- Below the GroupBox2 (which contains dataGridView) we have placed two buttons. When user clicks on 'Save Invoice:' button the new object of the Invoice class is created, all the Invoice, Customer and Product details we will store in that object and then we will place this object in the List (a kind of array) of type Invoice class. On the click event of 'Create new Invoice' button we are clearing all the Textboxes and

dataGridView to give a space to the user to enter the data for new Invoice. Make sure here the relationship is One Invoice can have multiple Products.

- Under these two buttons, we have placed four more buttons, that are |< (To View the first Invoice), < t(To view the Previous Invoice), > (To view the next Invoice) and >| (To view the last invoice). So, by using these four buttons user can navigate from one invoice to another.

- Now to implement functionality for this project, we have to add one class file called 'ProductItem' and add the following code to it.

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

namespace Unit3_Q5
{
    class ProductItem
    {
        //Properties of the ProductItem
        public int _pcode;
        public String _pname;
        public int _qty;
        public int _price;
        public int _linetotal;
    }
}
```

ProductItem.cs

- No we will create a class called 'Invoice' which will have attributes like Invoice No, Customer name, Invoice date and a list of products included in this invoice. Add a class file called 'Invoice.cs' and write the following code.

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

namespace Unit3_Q5
{
    class Invoice

    {
```

```
        public int _invcode;
        public String _custname;
        public DateTime _dt;
        public List<ProductItem> PI = new List<ProductItem>();
        public void addProductItem(int Pcode, String Pname, int Qty,
int Price)
        {
            ProductItem p = new ProductItem();
            p._pcode = Pcode;
            p._pname = Pname;
            p._qty = Qty;
            p._price = Price;
            p._linetotal = Qty * Price;
            PI.Add(p);
        }
        public ProductItem  getProductItem(int i)
        {
            ProductItem p =PI[i];
            return p;
        }
        public int countProducts()
        {
            int i = PI.Count;
            return i;
        }
    }
}
```

Invoice.cs

Now add the following code to the various events to your 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;

namespace Unit3_Q5
{
    public partial class Form1 : Form
```

```
    {
        List<Invoice> Invs = new List<Invoice>();
        int pos;
        public Form1()
        {
            InitializeComponent();
        }
       private void Form1_Load(object sender, EventArgs e)
        {
            dataGridView1.ColumnCount = 5;
            dataGridView1.RowCount = 1;
            dataGridView1.Columns[0].HeaderText = "PCode";
            dataGridView1.Columns[1].HeaderText = "PName";
            dataGridView1.Columns[2].HeaderText = "Qty";
            dataGridView1.Columns[3].HeaderText = "Price";
            dataGridView1.Columns[4].HeaderText = "LineTotal";
            pos = 0;
        }
    private void button1_Click_1(object sender, EventArgs e)
        {
        int i = dataGridView1.RowCount - 1;
        dataGridView1.RowCount = dataGridView1.RowCount + 1;
        int LineTotal = Convert.ToInt16(textBox6.Text) *
Convert.ToInt16(textBox7.Text);
            dataGridView1.Rows[i].Cells[0].Value = textBox4.Text;
            dataGridView1.Rows[i].Cells[1].Value = textBox5.Text;
            dataGridView1.Rows[i].Cells[2].Value = textBox6.Text;
            dataGridView1.Rows[i].Cells[3].Value = textBox7.Text;
            dataGridView1.Rows[i].Cells[4].Value =
Convert.ToString(LineTotal);
            textBox4.Text = "";
            textBox5.Text = "";
            textBox6.Text = "";

             textBox7.Text = "";
            textBox4.Focus();
        }
    private void btnSaveInv_Click(object sender, EventArgs e)
        {
          Invoice objInv = new Invoice();
          objInv._invcode = Convert.ToInt16( textBox1.Text);
          objInv._custname = textBox2.Text;
          objInv._dt = Convert.ToDateTime(textBox3.Text);
          int i;
```

```
            for (i = 0; i < dataGridView1.RowCount - 1; i++)
            {
                int pcode =
Convert.ToInt16(dataGridView1.Rows[i].Cells[0].Value);
                String pname =
Convert.ToString(dataGridView1.Rows[i].Cells[1].Value);
                int qty =
Convert.ToInt16(dataGridView1.Rows[i].Cells[2].Value);
                int price =
Convert.ToInt16(dataGridView1.Rows[i].Cells[3].Value);
        objInv.addProductItem(pcode, pname, qty, price);
            }
            Invs.Add(objInv);
        }

  private void btnAddInv_Click(object sender, EventArgs e)
        {
            foreach (var control in this.Controls)
            {
                if (control.GetType() == typeof(TextBox))
                {
                    ((TextBox)control).Text = "";
                }
            }
            textBox1.Focus();
            dataGridView1.Rows.Clear();
            dataGridView1.RowCount = 1;
        }
        private void showData()
        {
            Invoice tmp = Invs[pos];
            button2.PerformClick();
            textBox1.Text = Convert.ToString ( tmp._invcode);
            textBox2.Text = tmp._custname;
            textBox3.Text = Convert.ToString(textBox3.Text);
            int i;
            for (i = 0; i < tmp.PI.Count; i++)
            {
                dataGridView1.RowCount = dataGridView1.RowCount + 1;
                dataGridView1.Rows[i].Cells[0].Value =
tmp.PI[i]._pcode ;
                dataGridView1.Rows[i].Cells[1].Value =
tmp.PI[i]._pname ;
```

```
                dataGridView1.Rows[i].Cells[2].Value =
tmp.PI[i]._qty ;
                dataGridView1.Rows[i].Cells[3].Value =
tmp.PI[i]._price;
                dataGridView1.Rows[i].Cells[4].Value =
tmp.PI[i]._linetotal;
            }

        }
        private void btnFirst_Click(object sender, EventArgs e)
        {
            pos = 0;
            showData();
        }
        private void btnPrev_Click(object sender, EventArgs e)
        {
            if (pos > 0)
            {
                pos -= 1;
                showData();
            }
        }
        private void btnNext_Click(object sender, EventArgs e)
        {
            if (pos < Invs.Count - 1)
            {

                pos += 1;

                showData();
            }
        }
        private void btnLast_Click(object sender, EventArgs e)
        {
            pos = Invs.Count - 1;
            showData();
        }
    }
}
```

Form1.cs

- Insert the Invoice Number, Customer Name, Date of invoice and Product details to the form. You can add multiple product details to a single Invoice. Finally click on a

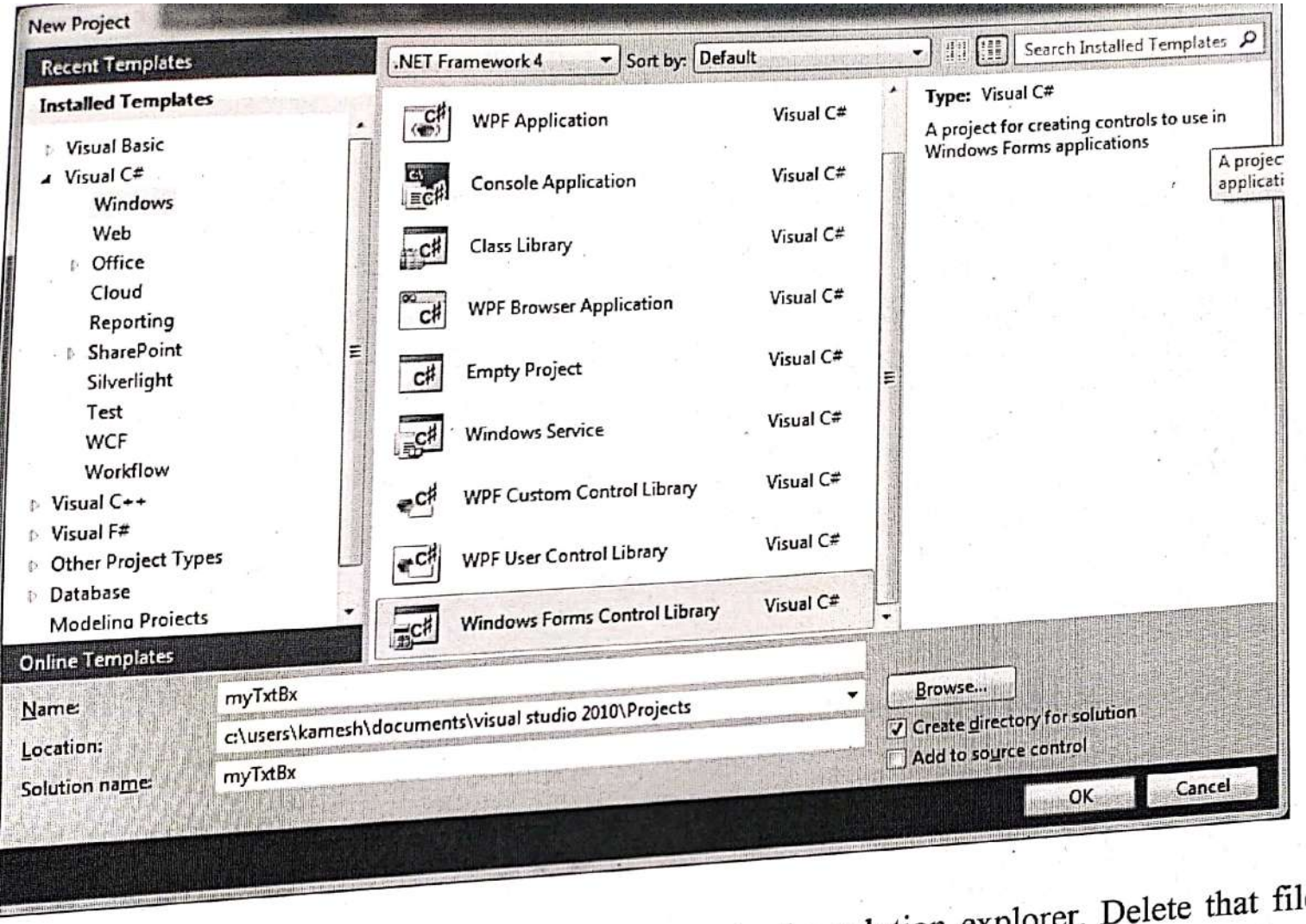
save invoice button, again you can add new invoice by clicking on create new invoice button, again you can enter the data and save the invoice. You can also navigate between the invoices by clicking on the four navigation buttons we have placed at the bottom of the form. This application is important as the code can be used to implement in the shopping cart in the asp.net which you will learn in the next semester.

Program: 6

Create a user control called myTextBx, having all the properties of the TextBox. Add Four Properties to it called EnterClr as Color, LeaveClr as Color and MandatoryClr as Color and Mandatory as Boolean. Use this control on Windows Application. Object of this control will show EnterClr when it is being focused. On Leave if Mandatory property is TRUE and Textbox is Empty then MandatoryClr has to be applied otherwise LeaveClr is applied to the textbox.

- This is a beautiful example of learning how can we create our own control. In this application we will learn 'how can we create our own TextBox control?', which will have all the features inherited from the original textbox, plus it has some additional features, mentioned in the question.
- Create a new project in the Visual Studio, but this time rather than taking 'Windows Forms Application' template take 'Windows Forms Control Library' template as shown in the figure. Give a name to this project as 'myTxtBx' project.
- This will create a new project called 'myTxtBx' which is a type of 'Windows control library' and inside this project, you can get 'UserControl1.cs' as a user interface, instead of 'Form1.cs'. Make sure, we cannot run this project. This project is to develop a new control. Once the control becomes ready we can use this control in the other project of type 'Windows Forms Application'.
- We want to develop a new control that is 'myTextBox' which will all features of our regular 'TextBox', plus it should have its own additional features. For that we will inherit our class from Textbox class.

First Right click on the 'UserControl1.cs' in the solution explorer and rename it to 'myTextBox.cs'. Now click on the View code button of the solution explorer to load code windows.



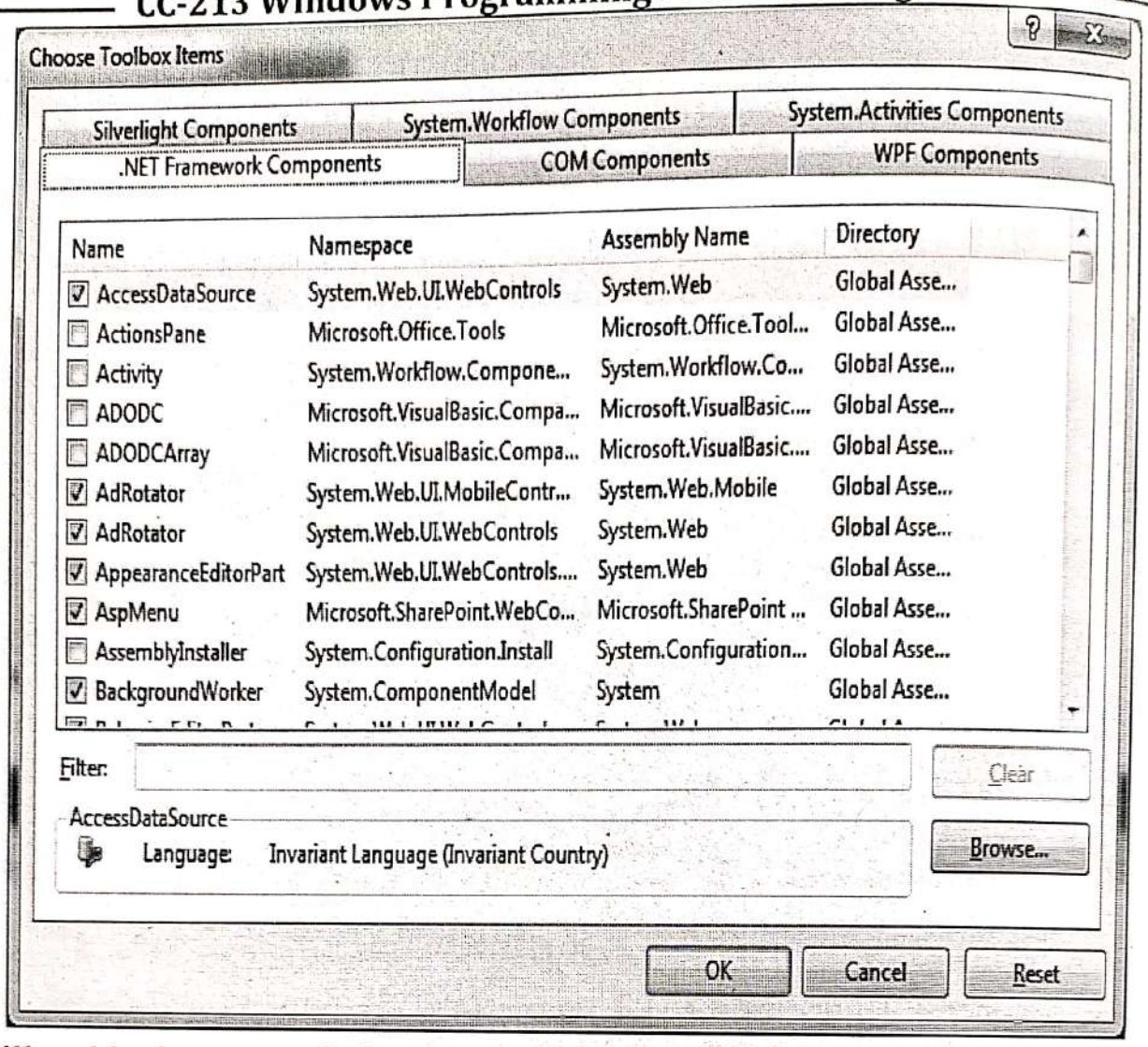
➢ Now you can find a 'UserControl1.cs' file in the solution explorer. Delete that file and right click on the solution explorer, click on Add and select 'UserControl…' option. In the dialogbox give name 'myTextBox' and click on OK button. This will add a new user control called 'myTextBox.cs' to your project. Write the following code in it as shown below.

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

namespace myTxtBx
{
    public partial class MyTextBox : TextBox
    {
        public Boolean _Mandatory;
```

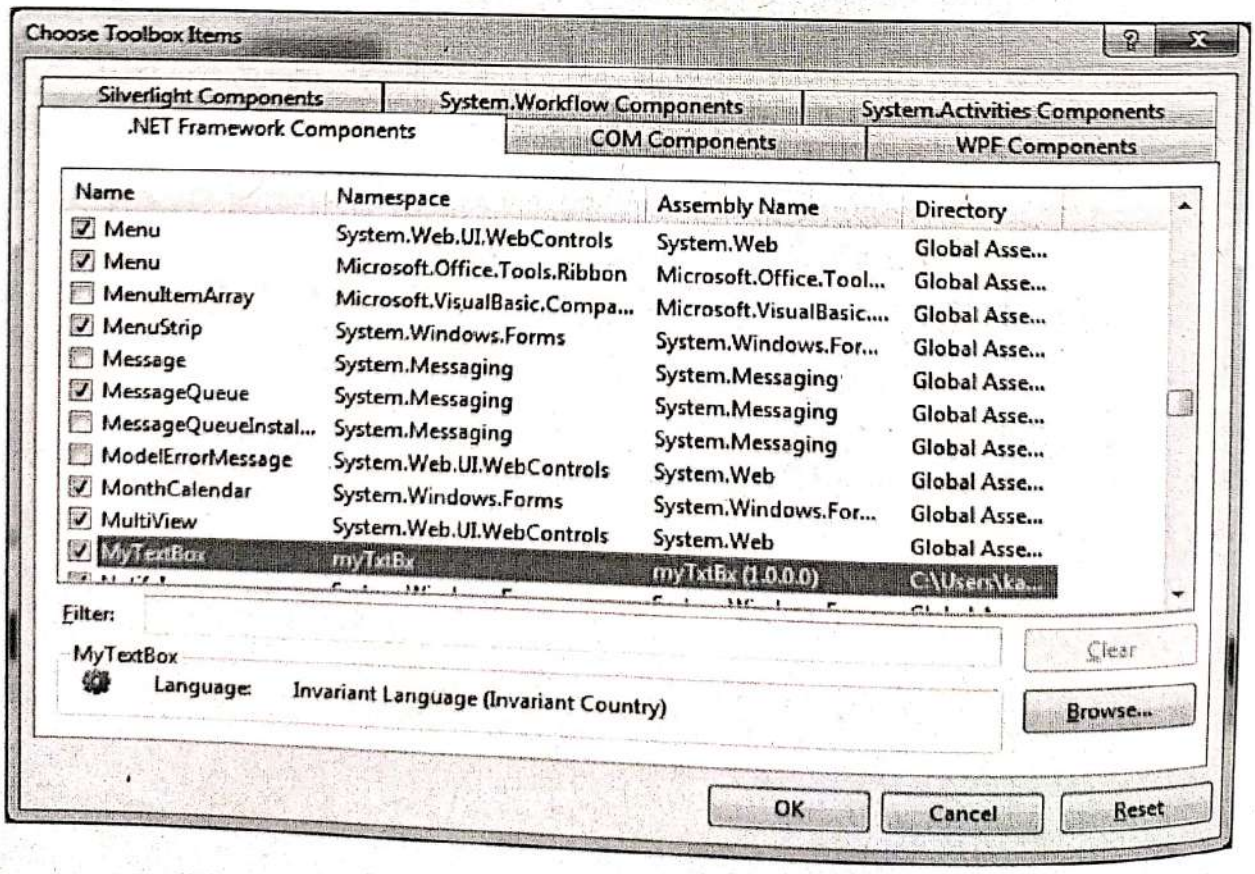
```
        public Color _EnterColor;
        public Color _LeaveColor;
        public Color _MandatoryColor;
        public MyTextBox()
        {
            InitializeComponent();
        }

        [System.ComponentModel.Browsable(true)]
        [Description("Set TRUE if you want to show MandatoryColor
when user keeps Null")]
        [Category("Category1")]
        public Boolean Mandatory
        {
            get { return _Mandatory; }
            set { _Mandatory = value; }
        }

        [System.ComponentModel.Browsable(true)]
        [System.ComponentModel.Category("Category1")]
        public Color MandatoryColor
        {
            get { return _MandatoryColor; }
            set { _MandatoryColor = value; }
        }


         [System.ComponentModel.Browsable(true)]
        [System.ComponentModel.Category("Category1")]
        public Color EnterColor
        {
            get { return _EnterColor; }
            set { _EnterColor = value; }
        }

        [System.ComponentModel.Browsable(true)]
        [System.ComponentModel.Category("Category1")]
        public Color LeaveColor
        {
            get { return _LeaveColor; }
            set { _LeaveColor = value; }
        }
    private void MyTextBox_Enter(object sender, EventArgs e)
        {
```

```
            this.BackColor = _EnterColor;
        }

    private void MyTextBox_Leave(object sender, EventArgs e)
        {
            if (this.Text.Length == 0 && Mandatory == true)
            {
                this.BackColor = _MandatoryColor;
            }
            else
            {
                this.BackColor = _LeaveColor;
            }
        }

    }
}
```

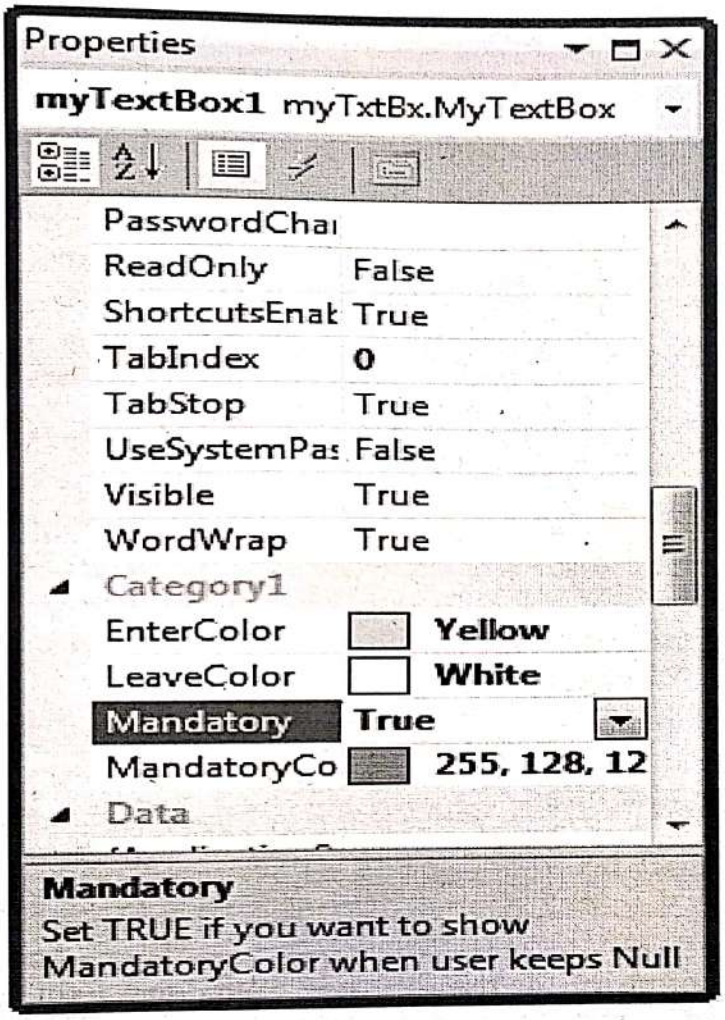
- Now go to the 'Build' menu and click on the 'Build Solution'. Your project will be compiled and you will get the message that: '========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========' in the output window.
- Once this is done successfully, you will get one DLL (Dynamic Linked Library) file that is myTxtBx.dll into the path:
  <<Your project th>>\myTxtBx\myTxtBx\bin\Debug directory. Copy that file and put it on the desktop or any other convenient and accessible path.
- Save and Close the project. Create a new project in the Visual Studio called "Unit3_Q6". Make sure this time we will take "Windows Forms Application" template.
- Now right click in the ToolBox, select 'Choose Item' option. It will open a dialog box. From this dialogBox, click on the 'Browse' button, give a path the 'myTxtBx.dll' file, and then click on OK button.



➢ It will add the user defined control myTextBox to your .NET Framewor Components. Click on the OK button for this dialogbox.



- This will add a new control called 'myTextBox' in the toolbox. Drag and drop this control for 3 times on the form, this will add myTextBox1, myTextbox2 and mytextBox3 on the form1. Now, Select myTextBox1 and go to property window. In the property window you can get separate category called 'Category1', and in that category you will get 4 properties, that are 'Mandatory', 'EnterColor', 'MandatoryColor', and 'LeaveColor'. Set these properties for all three control as shown below:



- Now you have set the Mandatory property TRUE for all 3 controls, and you have given Mandatory color (Orange), EnterColor (Yellow) and LeaveColor(White).

You can get the Description for the property here ⟶

- When you run the application, that textbox which is focused automatically gets EnterColor (Yellow), If you keep textbox empty and press tab button, it automatically takes MandatoryColor (Orange), But if you type something in the textbox (not null) and press enter then it automatically takes LeaveColor(White).
- Here we have tried to extend the capability of the existing textbox control.

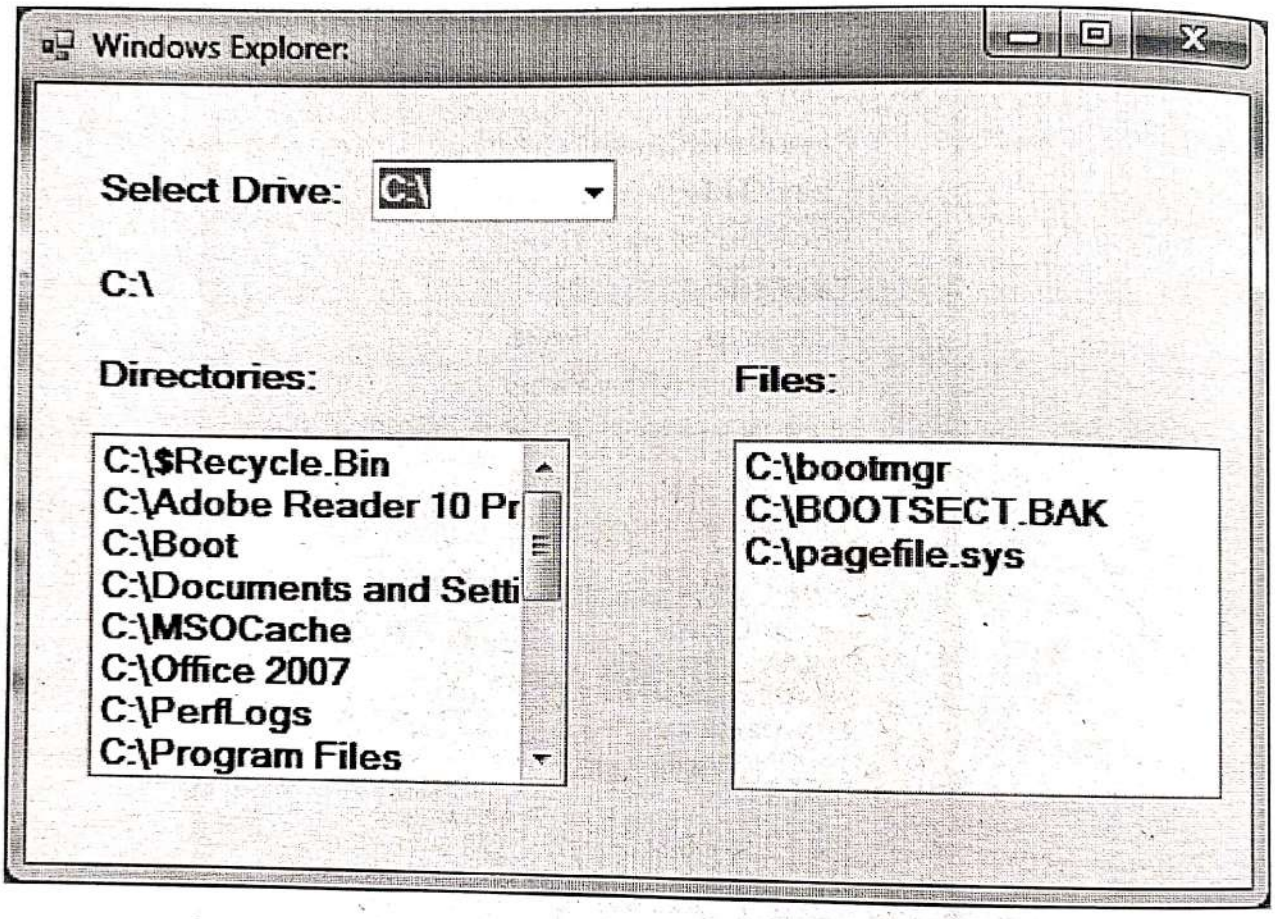
Program: 7

Write an application, which works like a window's explorer, using combo box and list box.

1. The combo box should display the list of drives.

2. The list box should display the list of files and directories.

- Create a new project in the Visual Studio called "Unit2_Q7" and place a Combobox and 2 Listbox and 4 label controls in the form1 as shown below:



- Set the Text property for the form1 is "Windows Explorer", and Text property for Label1 is "Select Drive:", Label2 is "Path", Label3 is "Directories:" and Label4 is "Files:" as shown in the figure. Now double click on the ComboBox. It will create and open a subroutine called 'comboBox1_SelectedIndexChange()'. Write the following code in the subroutine.

```
private void comboBox1_SelectedIndexChanged(object sender, EventArgs
e)
        {
            label2.Text = comboBox1.Text;
            refreshListBox();
        }
```

- Here 'refressListBox()' is a subroutine that we have to create, and we need to write the following code in it.

```
private void refreshListBox()
        {
            listBox1.Items.Clear();
            listBox2.Items.Clear();
            String[] dirs;
            dirs =
System.IO.Directory.GetDirectories(label2.Text);
            foreach (String str in dirs)
            {
                listBox1.Items.Add(str);
            }
            String[] files;
            files = System.IO.Directory.GetFiles(label2.Text);
            foreach (String str in files)
            {
                listBox2.Items.Add(str);
            }
        }
```

- Now go to the design view of the form1, and double click on it. It will create and open the subroutine called 'form1_Load' in the code editor. Write the following code on the load event of the form:

```
private void Form1_Load(object sender, EventArgs e)
        {
            String[] drvs;
            drvs = System.IO.Directory.GetLogicalDrives();
            foreach (String str in drvs)
            {
                comboBox1.Items.Add(str);
            }
        }
```

- Again go back to the Design view of the form1. Select the Listbox1 (where we are showing directories). Open the property window by pressing F4. Click on the event

icon in the property window. Now Locate 'DoubleClick' event and do double click on it. It will create and open a subroutine called "listBox1_DoubliClick()' as shown below. Type the following code in that subroutine. Run and test the application.

```
private void listBox1_DoubleClick(object sender, EventArgs e)
        {
            label2.Text = listBox1.SelectedItem.ToString();
            refreshListBox();
        }
```

☺ ☺ ☺ ☺ ☺

Programs of Unit-4

In this unit we will focus, how we can build database applications. We will discuss different classes which will help us to handle database from our Visual Studio .NET application. Set of classes used to establish connectivity with different databases is called ADO.NET. In this chapter we will learn how to develop application using ADO.NET classes.

❖ Introduction to ADO.NET:

Many different types of application, we can develop in the .NET. It can be a game, small mobile application, windows services, network application or it can be a database application. In most of the cases, when we develop an application for any organization it will be a database application. There are two main components are there for such type of application. Those are [1] Application [2] Database.

[1] Application:

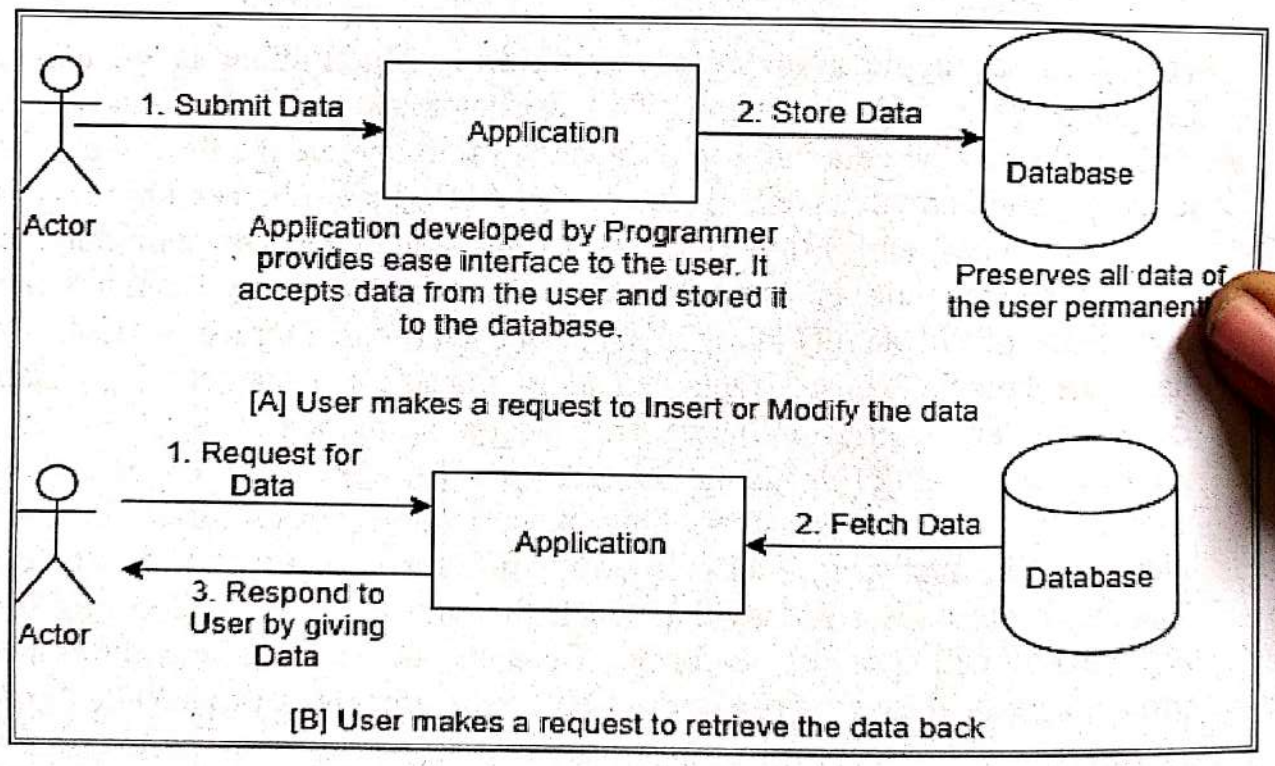
To design the application we are using any programming language, such as C-Language, C++, JAVA, Python, PHP, Visual Basic .NET or C#.NET. Application basically stores the data into the Database. But to operate the database, person should know complex structure of DBMS. Person can manage the database if he/she knows about data types, tables, relationships among tables, normalization and finally SQL. Now, a person who is not belongs to Computer Science, cannot have in depth knowledge of DBMS (Think of person who works in Library or Bank. Should they have the knowledge of Oracle?). So, to manage the database, programmers are developing an application using any programming language (we will use .NET platform and C#.NET language here), to provide ease and GUI based user friendly environment. For the user everything is application (because user does not have any idea about the database). User is inserting the data in the application, and clicks on the 'SAVE' button. User knows that when the same data is needed in the future and application will provide it back. Because the application developed by the programmer is always remain in the front to the user it is also called as 'Front-End'.

[2]Database:

The data entered by the user in the application, will be stored in the database, by the application, and when the user needs that data back, application will retrieve the data from the database and give it to the user. We know that to insert the data in the database we need to write insert SQL query, and to retrieve the data from the database we need to write select SQL query. Obviously these queries are written by the programmer in the application, which will be executed by the application when user is

just clicking on the different buttons provided in the GUI based interface of application. Here user does not have to learn about complex database structure or SQL queries. Databases are usually created on the secondary memory and it can store the data permanently. Because the user is not aware about database, which actually stores the data of user's organization, it is called 'Back-End' part of the application. Many DBMS applications are available. Microsoft Access, Microsoft SQL server, Oracle, MySQL etc. are example of Databases. Oracle and Microsoft SQL Server databases are very strong databases and can support trillions of bytes of data. The following figure will clear your concept about a database application:

See the figure given below, in [A] we have shown user is inserting or updating the data. User submits the data to the application which provides GUI base user friendly interface. User is just clicking on the 'Save' or 'Update' button. Application then fire 'Insert' or 'Update' query and the data will be inserted or modified in the database. In [B] user wants to retrieve the data back. User just enters the criteria in the textbox and clicks on the 'Search' button of application. Application then fire 'Select' SQL query (based on the user's criteria), and fetch the data from the database and show the result to the user on the application screen.



ADO.NET stands for ActiveX Data Object, it is a collection of the classes of .NET framework, provides functionality to handle database. With the help of these classes developer can write the code in the application for various database functionalities such as SELECT, INSERT, UPDATE and DELETE operations. All the classes (of ADO.NET) which allows programmer to operate various database functionality by writing a code, are available in the 'System.Data' namespace of .NET framework

class library. We have tried to explain System.Data namespace by the following figure:

Basic data types such as, Boolean, Int16, Int32, Single, Double, etc. are available in the system namespace, and by default when we create a project in the Visual Studio, then system namespace is included in that project. Therefore we can directly declare their variables (without using namespace system).

For example,

We are not writing, *'System.Boolean X'* but instead we are simply writing *'Boolean X'*. Look at to your coding window. On the top of it you can find a line called: *'using System"*. .NET do not enforce programmer to use a particular database. You can use Microsoft Access, SQL Server, Oracle or any other database. So, for each of the database ADO.NET provides separate classes such as connection, data adapter, command, data reader, command builder of each of the database into separate namespace (as you can see in the following figur

Class Hierarchies of ADO.NET Classes

- SQLConnection
- SQLCommand
- SQLDataAdapter
- SQLDataReader
- SQLCommandBuilder

SQLClient

- OracleConnection
- OracleCommand
- OracleDataAdapter
- OracleDataReader
- OracleCommandBuilder

ORACLEClient

- OLEDBConnection
- OLEDBCommand
- OLEDBDataAdapter
- OLEDBDataReader
- OLEDBCommandBuilder

OLEDB

- DataSet

DATA

This System namespace contains many namespaces like "System.IO", "System.Net", "System.Data" and so on. And the study of the classes which are located in the "System.Data" namespace is called study of ADO.NET.

This namespace "System.Data" has a class called 'DataSet'. Along with that it contains few more namespaces such as "SQLClient", "ORACLEClient", "OLEDB" etc.

1. **System.Data.SQLClient:** This namespace contains classes which are used to perform connectivity between C#.NET application and SQL-Server.

2. **System.Data.OracleClient:** This namespace is used if our database is in the ORACLE. Classes under this namespace are used to perform connectivity with C#.NET application and Oracle Server.

3. **System.Data.OLEDB:** This namespace can be used to perform connectivity between C#.NET application and any Database. The namespace provides classes to be used to fetch the data located in either, MS-Access, Oracle, SQL-Server or any other database.

Example:

1. If the database in the SQLServer, then we have two choices: Either we can use [1] System.Data.SQLClient OR we can use [2] System.Data.OLEDB. In this case, classes located under 'System.Data.SQLClient' gives us much better performance than the classes located under the namespace 'System.Data.OLEDB'. Because of the classes under 'System.Data.SQLClient' is specifically designed to operate SQL Server database only.

2. If the database is in the Oracle, then again, we have again two choices: Either we can use all the classes of namespace: [1] System.Data.ORACLEClient OR we can use the classes of [2] System.Data.OLEDB. In this case, classes located under 'System.Data.ORACLEClient' gives much better performance than the classes located under 'System.Data.OLEDB'. Because the classes under 'System.Data.OracleClient' is specifically designed to operate Oracle Server only.

3. If the database is in the MS-Access then, we have only one choice, which is "System.Data.OLEDB".

Now, we will go through the various classes, and we discuss the various properties and methods of it. In the example I have used MS-Access as a Database and OLEDB as a namespace.

[1] Connection Object: Connection object on an ADO.NET is preserving all the necessary information about the database, so that it can open the database. Some information is required to open the database, such as file format (Database is in the Oracle, SQL-Server or in the MS-Access), path (Where the data base is?), username and password (if the database is password protected). Connection object of the database is opening the database first, and then the other ADO.NET object can perform the necessary action in it.

Declaration:

System.Data.OLEDB.OLEDBConnection cn = new System.Data.OLEDB.OLEDBConnection();

This type of declaration has to be done in the general declaration section. So that, object 'cn (object of the connection class)' can be used in the entire form (in any function or subroutine).

Properties:

- **ConnectionString:** ConnectionString consists of provider, data source, username and password. This information is necessary to open any database.
- **State:** It is a Boolean type property which is returning either 0 (if the connection is closed) or 1 (if the connection is opened).

Syntax:

cn.ConnectionString="Provider=Microsoft.JET.OLEDB.4.0; Data Source=D:\\MyDB\\Employee.mdb";

In the above example Data Source is a simple path to the database, which contains the path, and file name with extension.

Provider is depending on the file format of the data base. Different providers are used for different databases, which are explained in the table given below.

Provider	File Format of the Database
Microsoft.JET.OLEDB.3.51	MS-Access 97
Microsoft.JET.OLEDB.4.0	MS-Access 2000, XP or 2003
Microsoft.ACE.OLEDB.12.0	MS-Access 2007
MSDAORA	Oracle
SQLOLEDB	SQL-Server

Note: The provider is needed only if, we are using OLEDB Connection. Because OLEDB allows us to operate all type of databases such as MS-Access, Oracle or SQL-Server. If we are using 'OracleClientConnection' then, it is specifically designed to establish connection, with only Oracle server. If 'SQLConnection' is used then, it

is specifically designed to establish connection with 'SQL Server' only. So, in the case of 'OracleClient' or 'SQLClient' connection, provider is not needed.

Methods:

- **Open( ):** Open is the method to open the connection. Before using this method 'ConnectionString' property must be initialized properly. Any mistake in the 'ConnectionString' will generate error in the 'Open' method.
- **Close( ):** Close is the method of the Connection, which is basically used to close the connection. Once it is closed then no other operation we can perform to the database.

Syntax:

cn.ConnectionString="Provider=Microsoft.JET.OLEDB.4.0; Data Source=D:\\MyDB\\Employee.mdb";

cn.Open();

[2] Command Object: Command object of an ADO.NET can store any single SQL query (SQL Statement) and it can execute it, with the help of a connection object.

Declaration:

System.Data.OLEDB.OLEDBCommand cmd = new System.Data.OLEDB.OLEDBCommand();

Properties:

- **CommandText: This property of the command object is storing SQL statement which we want to execute.**
- **Connection:** The command object can store SQL statement and it can execute it. But it requires open or active connection. The connection object knows everything about the database. By using this property, we can assign connection object (*cn*) to the command object (*cmd*), which we have already declared, configured and opened earlier.

Syntax:

System.Data.OLEDB.OLEDBConnection cn = new System.Data.OLEDB.OLEDBConnection();

cn.ConnectionString="Provider=Microsoft.JET.OLEDB.4.0; Data Source=D:\\MyDB\\Employee.mdb";

```
cn.Open();

System.Data.OLEDB.OLEDBCommand cmd = new
System.Data.OLEDB.OLEDBCommand();
cmd.CommandText= "Insert into Table1 values (1, 'Abc', 23)";
cmd.Connection = cn;
cmd.ExecuteNonQuery();
```

To test above syntax, create form having button. Write down the code on the click event of the button. Create a database called 'Employee.mdb' under the path mention in the connection string. Create a blank table called Table1 having 3 fields, EmpCode (Number), EmpName (Text) and Age (Number).

To test the application, run it, click on the button, close the running form and open the database. Open the table we can get one record is inserted in the table1.

Methods:

- **ExecuteNonQuery( ):** This method of the command object is used to execute any action query stored in the 'CommandText' property of the command object. Action queries are such, which can perform action in the database, and not return any value. 'Create table', 'Insert', 'Update' or 'Delete' query are the example of the action queries.

  Syntax:

  If we want to execute the query mentioned in the earlier example write the following syntax at the end of the code.

  ```
  cmd.ExecuteNonQuery();
  ```

- **ExecuteScalar( ):** This method of the command object is used to execute select query which is returning single value. Consider the following queries which are returning single value that can be of any type such as Boolean, String, Integer, or Single.

  1. Select count(*) from Table1
  2. Select Max(Age) from Table1
  3. Select Min(Age) from Table1
  4. Select Avg(Age) from Table1
  5. Select age from Table1 where EmpName='Abc'

Syntax:

```
System.Data.OLEDB.OLEDBConnection cn = new
System.Data.OLEDB.OLEDBConnection();

cn.ConnectionString="Provider=Microsoft.JET.OLEDB.4.0; Data
Source=D:\\MyDB\\Employee.mdb";

cn.Open();

System.Data.OLEDB.OLEDBCommand cmd = new
System.Data.OLEDB.OLEDBCommand();
cmd.CommandText= "Select Max(Age) from Table1";
cmd.Connection = cn;
int MaxAge;
MaxAge=cmd.ExecuteScalar();
MessageBox.Show(Convert.ToString (MaxAge ));
cn.Close();
```

- **ExecuteReader( ):** This method of the Command object is used if we want to execute, Select query which returns multiple values from the database table. The query returns many values, so those are stored in the special variable called the object of 'DataReader', which explained in details the discussion of command object.

Syntax:

```
System.Data.OLEDB.OLEDBConnection cn = new
System.Data.OLEDB.OLEDBConnection();

cn.ConnectionString="Provider=Microsoft.JET.OLEDB.4.0; Data
Source=D:\\MyDB\\Employee.mdb";

cn.Open();
System.Data.OLEDB.OLEDBCommand cmd = new
System.Data.OLEDB.OLEDBCommand();

cmd.CommandText= "Select * from Table1";

cmd.Connection = cn;

System.Data.OLEDB.OLEDBDataReader dr;
dr=cmd.ExecuteReader();
```

[3] DataReader Object:_ DataReader object of an ADO.NET is used to store the output of the execution of the 'Select' query which returns multiple values. 'DataReader' object is used to just read the data and no modification (Insert, Update or Delete) is allowed in it. Probably to fetch the values from the database table and populate them in the comboBox or listBox, we can use the object of DataReader. It is faster than 'DataSet' object. DataReader object provides multiple values row wise and we can access them in the forward direction only. That means we cannot fetch or process record number 3 after fetching or processing record number 5.

Declaration:

System.Data.OLEDB.OLEDBDataReader dr;

Properties:

- **FieldCount:** This property of the data reader object returns an integer number depends on how much fields are there in the table.
- **Item:** This property returns collection (array) of the different items. Because it is very much possible that one can have many fields with the different type. To fetch the value from the specific field, index number is passed. For example in the above table if we want to fetch EmpCode then dr.item[0], if we want to fetch EmpName then dr.item[1] is written.

Methods:

- **Read ( ):** This method reads a record, every time. The records are fetched sequentially only. Normally it is used with the while loop. The following example demonstrates the use of data reader object.

[4] DataSet Object: DataSet object of an ADO.NET is a collection (array) of one or more data tables. DataSet allow us to modify the data which stored inside the data table. Compare to Data Reader, DataSet object also allow us to access any part of the table, whether it is in the forward direction or reverse direction. Data set object is slower than Data Reader object. Class DataSet doesn't have different version for different databases, so it is located centrally in the 'System.Data' namespace.

Declaration:

System.Data.Dataset ds= new System.Data.Dataset();

Properties:

- **Tables:** As we know dataset is collection of the data tables, these data tables we can access by using this property.

Syntax:

MessageBox.Show (Convert.ToString (Ds.Tables[0].Rows.Count));

OR

MessageBox.Show (Convert.ToString (Ds.Tables["Emp"].Rows.Count));

Comparision of DataSet and DataReader Object:

No.	DataSet	DataReader
1.	DataSet is collection of datatables. It can store the output of many queries, by making separate data table for each query.	DataReader can store the output of only one query. To store the output of another query, another instance of the data reader object is required.
2.	DataSet allows us to modify the data stored in the data tables.	DataReader is read only, and does not allow us to do modification the data.
3.	We can access the data from any part of the data table.	It is forward only we cannot access the data in backward direction.
4.	It can work in the disconnected environment.	It can work only in connected environment.
5.	DataSet is slower than DataReader.	DataReader is faster than Dataset

[5] DataAdapter Object:_ DataAdpter object of an ADO.NET act as a bridge between dataset and a table of database. It brings the data from the database table to DataSet, by creating a new DataTable in it. Also, when the DataTable of Dataset is modified, these changes will be reflected to the database by data adapter.

Declaration:

*System.Data.OLEDB.OLEDBDataAdapter da = new System.Data.OLEDB.OLEDBDataAdapter ("Select * from Emp, cn);*

It requires two things while it is initialized. One is select query of a particular table and second is Connection.

Properties:

- **SelectCommand:** This property gets or sets OLEDB Command object. This command contains select query.

- **InsertCommand:** This property gets or sets OLEDB command with insert query.
- **UpdateCommand:** This property gets or sets OLEDB Command with update query.
- **DeleteCommand:** This property gets or sets OLEDB Command with delete query.

Methods:

- **Fill( ):** This method brings the data from the database and create a new DataTable in to DataSet.

  Syntax:

  da.Fill(ds,"Emp");

- **Update( ):** It will reflect the changes done in the DataSet to the Database table.

  Syntax:

  da.update(ds,"Emp");

[6] CommandBuilder Object:_ CommandBuilder object of an ADO.NET is used to generate Insert, Update and Delete command in the data adapter. We know that data adapter is having four properties: SelectCommand, InsertCommand, UpdateCommand and Deletecommand. If we are using command builder then it is automatically initialized InsertCommand, UpdateCommand and DeleteCommand properties of the DataAdapter object. It requires two things:

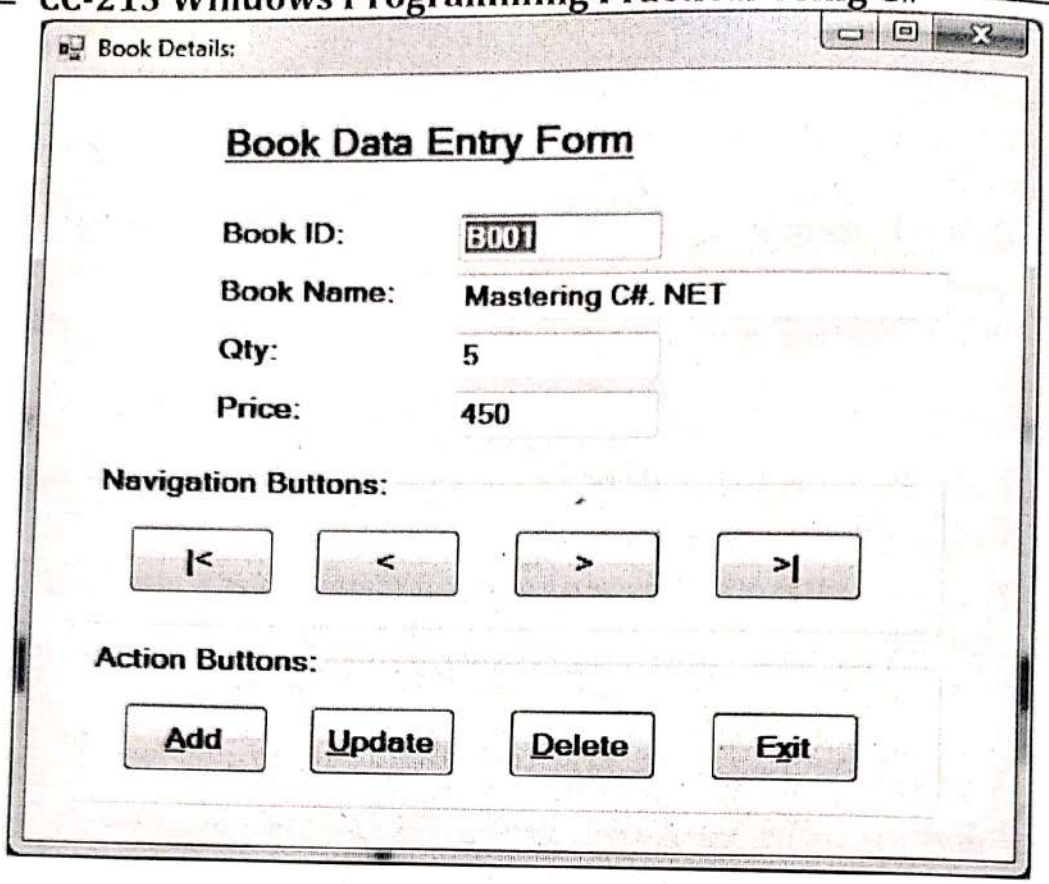
1. The SelectCommand property is properly initialized, and
2. The table must have either primary or composite primary key.

❖ Practical programs of ADO.NET:

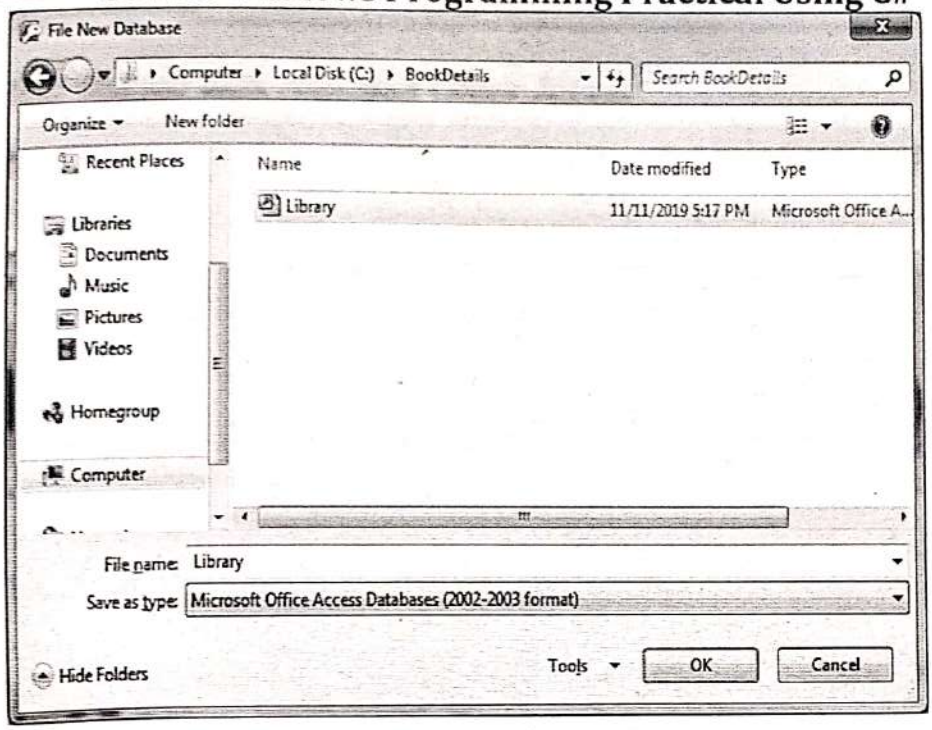
Program: 1

Write a Program to develop a Database Application with ADO.NET with Record Navigation and Add, Delete, Save and Update Facility for Books table (BookID, BookName, Qty, UnitPrice, TotalPrice).

- Create the project in Visual Studio .NET called "Unit4_Q1", and design the following interface for the form1.



- Create a folder called BookDetails into the C: Drive of you system. Open MS-Access and create the database called library.mdb. Make sure your access should be a XP, 2000 or 2003 file format, which will create a .mdb file. If you are using MS-Access 2007 then, at the time of database creation, select Save As type: to "Microsoft Office Access Databases (2002-2003 format)" File format as shown in the following figure.
- In the library.mdb File, create a table called "Books", with following fields: **Books (BookID Text, BookName (Text), Qty (Number), Price (Number))**
- Now give a Primary key to the field BookID. Save the table and open it to 'DataSheet' mode and insert at least 2 records in it (as shown in the form). Click on the Save button, Close MS-Access and come back to the Visual Studio.



- After designing the form change the name property of textboxes and buttons. For example for BookID, BookName, Qty and Price textboxes change the name to txtBookID, txtBookName, txtQty and txtPrice. Similarly for the buttons |<, <, > and > set the name properties to cmdFirst, cmdPrev, cmdNext and cmdLast. For the action buttons Add, Update, Delete and Exit set the name property to cmdAdd, cmdUpdate, cmdDelete and cmdExit.
- Write the following code on different events as shown below:

```
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.OleDb;

namespace Unit4_Q1
{
    public partial class Form1 : Form
    {
        public Form1()

        {
```

```
            InitializeComponent();
        }
        public static OleDbConnection cn = new OleDbConnection();
        public static OleDbDataAdapter da = new
OleDbDataAdapter("Select * from Books", cn);
        System.Data.DataSet ds = new System.Data.DataSet();
        OleDbCommandBuilder cmdbldr = new OleDbCommandBuilder(da);
        int pos;
        Boolean addmode;
        private void Form1_Load(object sender, EventArgs e)
        {
            cn.ConnectionString =
"Provider=Microsoft.JET.OLEDB.4.0; Data
Source=C:\\BookDetails\\Library.mdb";
            cn.Open();
            da.Fill(ds, "Books");
            pos=0;
            ShowData();
        }
        private void ShowData()
        {
            txtBookId.Text=
ds.Tables[0].Rows[pos].ItemArray[0].ToString();
            txtBookName.Text
=ds.Tables[0].Rows[pos].ItemArray[1].ToString();
            txtQty.Text
=ds.Tables[0].Rows[pos].ItemArray[2].ToString();
            txtPrice.Text
=ds.Tables[0].Rows[pos].ItemArray[3].ToString();
        }

 private void txtFirst_Click(object sender, EventArgs e)
        {
            pos = 0;
            ShowData();
        }

   private void txtPrev_Click(object sender, EventArgs e)
        {
            if (pos > 0)
            {
                pos = pos - 1;
                ShowData();
```

```
}
        }

    private void txtNext_Click(object sender, EventArgs e)
        {
            if (pos < ds.Tables[0].Rows.Count - 1)
            {
                pos = pos + 1;
                ShowData();
            }
        }

    private void txtLast_Click(object sender, EventArgs e)
        {
            pos = ds.Tables[0].Rows.Count - 1;
            ShowData();
        }

    private void txtAdd_Click(object sender, EventArgs e)
        {
            txtBookId.Text = "";
            txtBookName.Text = "";
            txtQty.Text = "";
            txtPrice.Text = "";
            txtBookId.Focus();
            addmode = true;
        }

    private void txtUpdate_Click(object sender, EventArgs e)
        {
            if (addmode == true)
            {
                DataRow dt = ds.Tables[0].NewRow();
                dt[0] = txtBookId.Text;

                 dt[1] = txtBookName.Text;
                dt[2] = txtQty.Text;
                dt[3] = txtPrice.Text;
                ds.Tables[0].Rows.Add(dt);
                da.Update(ds, "Books");
                addmode = false;
            }
            else
```

```
            DataRow dt = ds.Tables[0].Rows[pos];
            dt[0] = txtBookId.Text;
            dt[1] = txtBookName.Text;
            dt[2] = txtQty.Text;
            dt[3] = txtPrice.Text;
            da.Update(ds, "Books");
        }
    }

    private void txtDelete_Click(object sender, EventArgs e)
    {
        DialogResult dl;
        dl = MessageBox.Show("Are you Sure? Do you want to
Delete this Record?", "Confirm!!!", MessageBoxButtons.YesNo);
        if (dl == DialogResult.Yes)
        {
            ds.Tables[0].Rows[pos].Delete();
            da.Update(ds, "Books");
            pos = 0;
            ShowData();
        }
    }

private void txtExit_Click(object sender, EventArgs e)
    {
        this.Close();
    }
  }
}
```

- In this example 'Insert' and 'Update' both work is done by a single button called 'Update' button. We have taken a Boolean variable called 'addmode'. Initially variable 'addmode' will be 'false'. If user clicks on a Update button then, because of addmode variable is false, we are executing the code for 'Update' query. When user clicks on 'Add' button, we are clearing each field and set the value of 'addmode' variable to true. After that suppose user clicks on 'Update' button, then because of 'addmode' variable is true, we are executing the code to 'Insert a new record'. In short, it user clicks on Update button only, then we are executing the code to update record, but if user clicks on 'Add' button first and then 'Update' button then we are executing 'Insert' query.

- Variable 'pos' is of type integer, it will be initialized with value 0 on the load event of the form. It is useful to show the record on the screen. When variable 'pos' is 0, we are showing first record on the screen. On the click event of 'Next' button it will be incremented by 1, show subroutine 'ShowData()' will show second record. Similarly on the 'Prev' button we are decremented the value of 'pos' variable by 1, so subroutine 'ShowData()' will show the previous record on the screen.

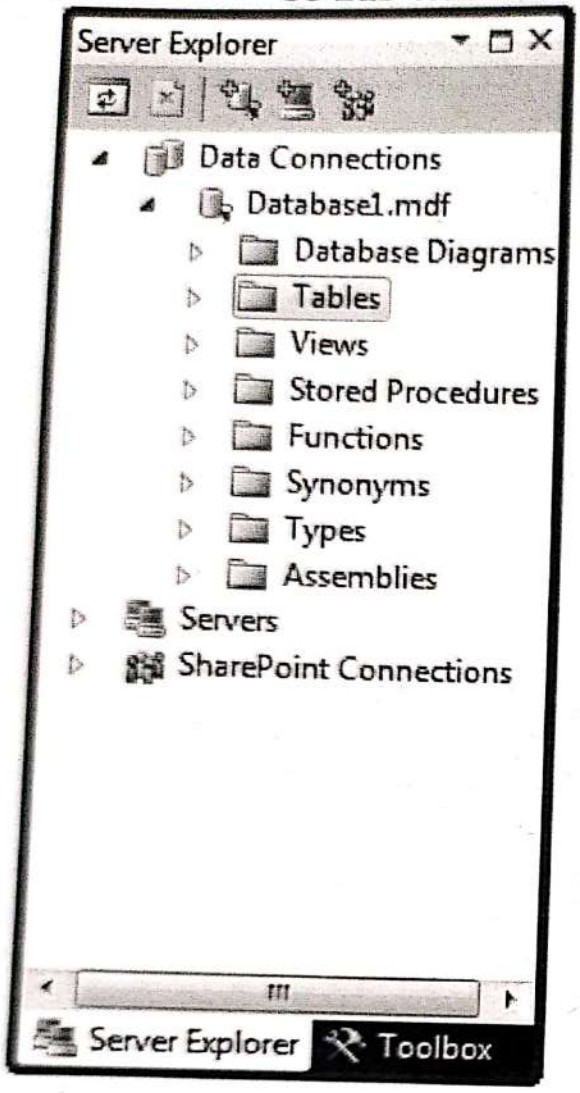
- In this application we have used MS-Access as a database, we need to use connection, data adapter, command builder etc. ADO.NET classes from OLEDB. So that in the 'using' section we have written 'using System.Data.OLEDB' to include the namespace OLEDB to our project.

Program: 2

Create a login form which will validate user name and password from the database. On validation show MDI form. Show appropriate message if user name and password is incorrect.

- In the previous question, we have used MS-Access (Library.mdb) as a database. As we have already discussed that ADO.NET provides, number of classes to connect our .NET application with any database, whether it is MS-Access, Oracle or MS-SQL Server. In this application we will use Express edition of MS-SQL server, which comes up with the 'Visual Studio' as a bundled. Obviously because in the Unit4_Q1, we have taken MS-Access as a database, we have used a namespace called 'System.Data.OLEDB'. In this application we are going to use MS-SQL Server as our application database (which is mostly preferred by majority of programmers because it provides strong database), we will use the classes from 'System.Data.SqlClient' namespace.

- To add the MS-SQL Server database file to the project, do right click on the solution name in the solution explorer. Click on Add, and then New Item option. It will open a dialog box. From the dialog box, select option 'Database' (Make sure when you select this option in FileName it should show Database1.mdf). If it is then click on OK button of the dialog box. This will add a file Database1.mdf to you project. Now double click on Database1.mdf file in the solution explorer. It will load new window for server explorer at the left hand side as shown below:



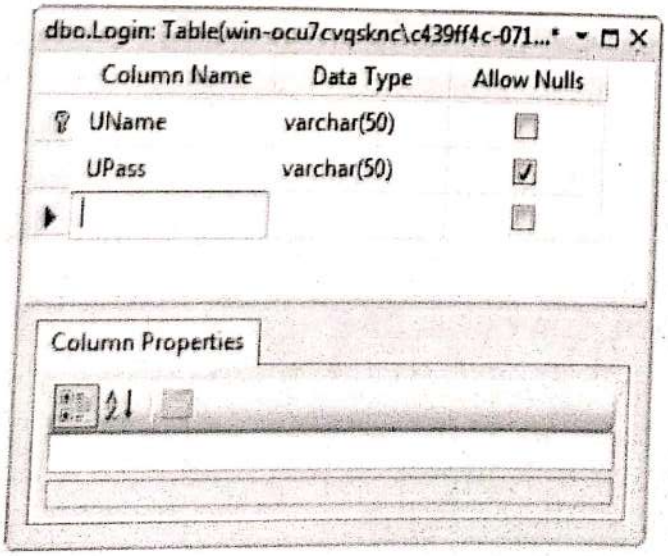
- Once the Server Explorer is loaded, then do right click on the Tables and select 'Add New Table' option. This will create a new table for you and it will be opened in another window in the designing mode. Create two fields in it, which are UName and UPass for user name and password.

- Set the data type to varchar(50) to both the fields, as shown in the figure given below. Now click on the Save button.

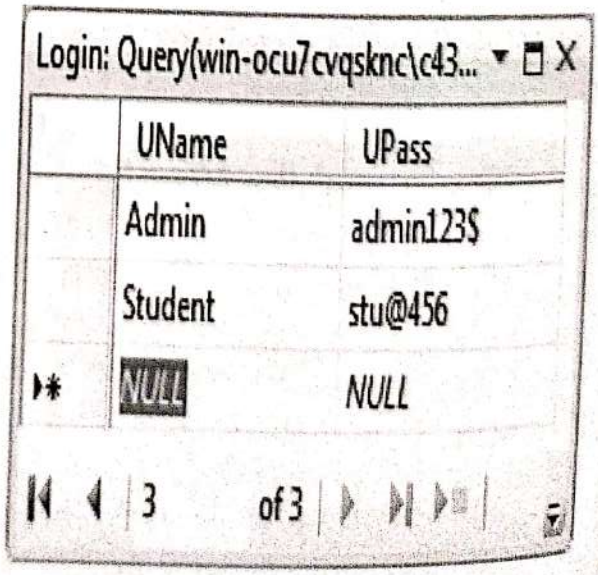
- It will prompt you to give a name to your table. It takes 'Table1' as a default table name, change it 'Login' and then click on OK button.

- Now if you expand the Tables node in the Server Explorer then you can see 'Login' table. Right click on the 'Login' table under Tables node of Server Explorer and select 'Show Table Data' option.

- This option will allows you to add the data in the 'Login' table. Enter at least two to three records in it. This is also shown in the following figure:

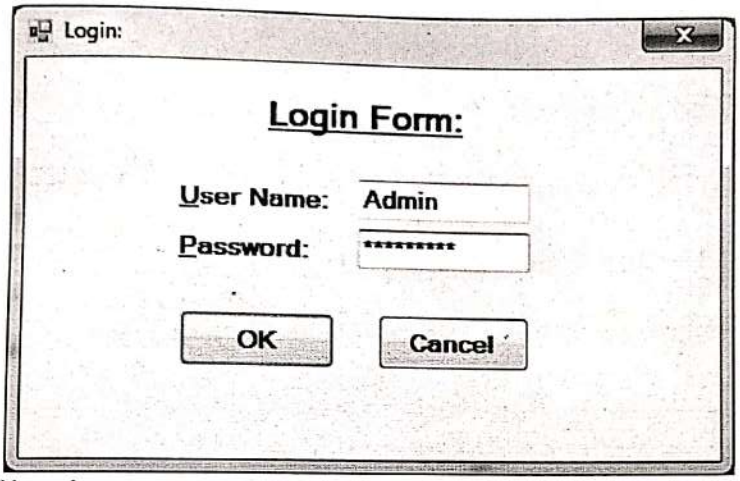


Design Mode



Data Entry Mode

- Once the data entry is done, close the table. Now add another form called to your solution by doing Right Click on the solution name in the Solution Explorer, select option 'Add' and the selecting option 'Windows Form…'. Now design the form1 as shown in the following figure.



- Now set the following properties for the different controls we have taken in our application:

Object	Property	Value	Explanation
Button1	Name	btnOK	
Button2	Name	btnCancel	
Textbox1	Name	txtUserName	
Textbox2	Name	txtPassword	
Form	Text	Login:	Shows Login in the caption of the form
	MinimizeBox	False	User can not minimize the form
	MaximizeBox	False	User can not maximize the form
	FormBorderStyle	FixedSingle	User can not resize the form
	StartPosition	CenterScreen	Form opens in the center of the screen
	Font -> Size	10	
	Font -> Bold	True	
	AcceptButton	btnOK	Button OK will

			be clicked automatically when user presses ENTER key.
	CancelButton	btnCancel	Button cancel will be clicked automatically when user presses Esc key.
Label1	Text	Login Form:	Shows the caption
	Font -> Size	14	
	Font->Underline	TRUE	
Form2	WindowState	Maximize	Form will be opened in the maximize mode.
	IsMDIContainer	TRUE	MDI Form

➢ Now write the following code on the form1 (Login).

```
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 Unit4_Q2
{
    public partial class frmLogin : Form
    {
        public frmLogin()
        {
         InitializeComponent();
        }

        public static SqlConnection cn;
        SqlCommand cmd;
        private void cmdCancel_Click(object sender, EventArgs e)
        {
```

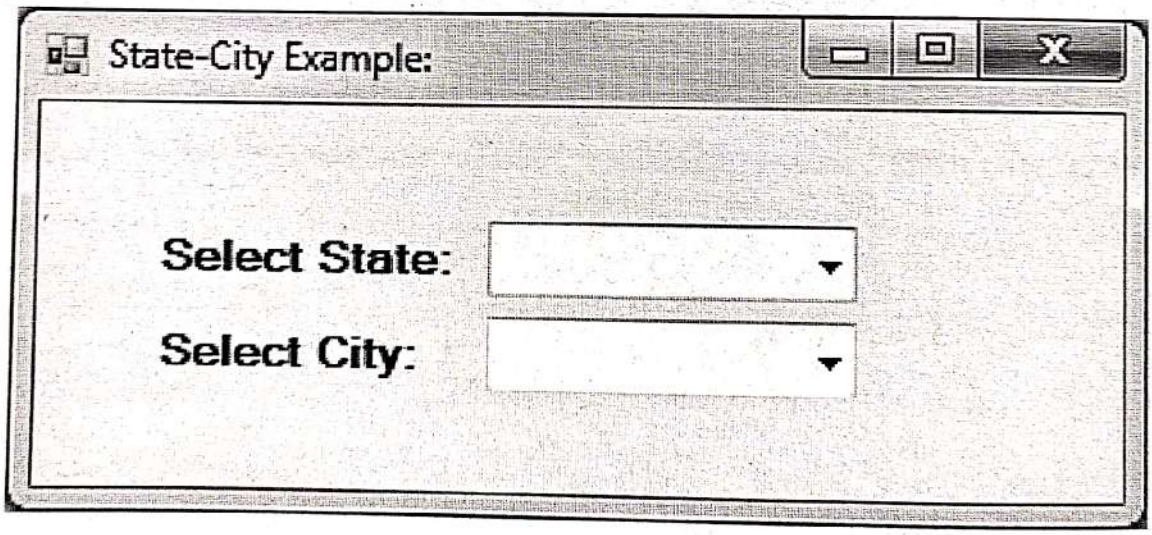
```
            this.Close();
        }

    private void frmLogin_Load(object sender, EventArgs e)
        {
    cn = new SqlConnection();
    cn.ConnectionString ="Data
Source=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\Database1.
mdf;Integrated Security=True;User Instance=True";
            cn.Open();
            MessageBox.Show(cn.State.ToString());
        }

     private void cmdOK_Click(object sender, EventArgs e)

    {
            cmd = new SqlCommand();
            cmd.CommandText = "Select UPass from Login where
Uname='" + txtUserName.Text + "'";
            cmd.Connection = cn;
            String opass;
            opass =Convert.ToString ( cmd.ExecuteScalar());
            if (opass != "")
            {
                if (opass == txtPassword.Text)
                {
                    Form2 frm = new Form2();
                    frm.Show();
                    this.Hide();
                }

                else

                {
                    MessageBox.Show("Invalid UserName or
Password:", "Error");
                    txtUserName.Clear();
                    txtPassword.Clear();
                    txtUserName.Focus();
                }
            }
            else
            {
```

```
                MessageBox.Show("Invalid UserName or Password:",
"Error");
                    txtUserName.Clear();
                    txtPassword.Clear();
                    txtUserName.Focus();
                }
            }
        }
    }
```

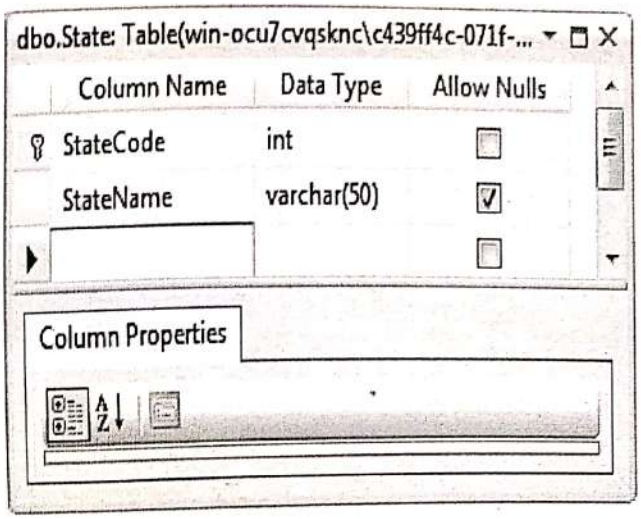
Program: 3

Create two tables State(StateCode, StateName) and City(CityCode, CityName, StateCode). Design a form which will list all the state in the ComboBox. After selection of the StateName, another ComboBox should display all the Cities which belong to the state selected by the user in the first ComboBox.

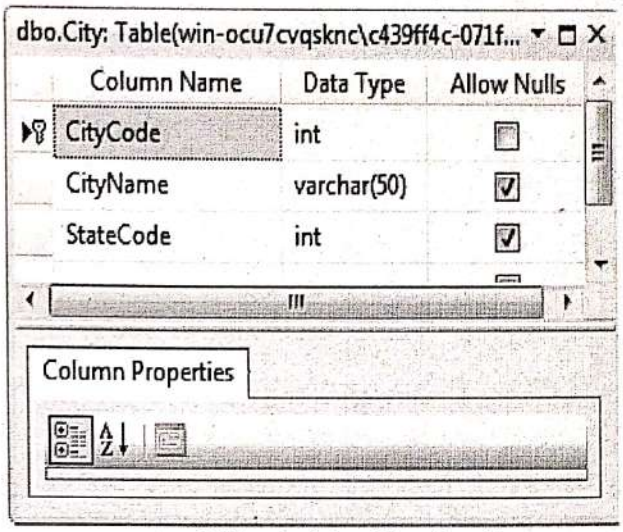
- Create a new project in the Visual Studio called "Unit4_Q3" and design the following interface for form1 by taking two Comboboxes and two labels, one for State and one for City.



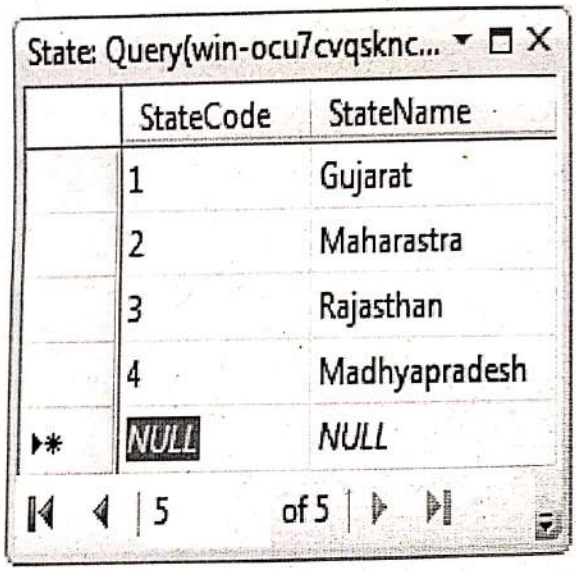
- Now add the database file called "Database1.mdf" to your project (as we have already discussed it in "Unit4_Q2"). Add two tables State and City with field given in the question. Give primary key to the field StateCode in the State table and CityCode in the City table. Do the following data entry in both the tables:



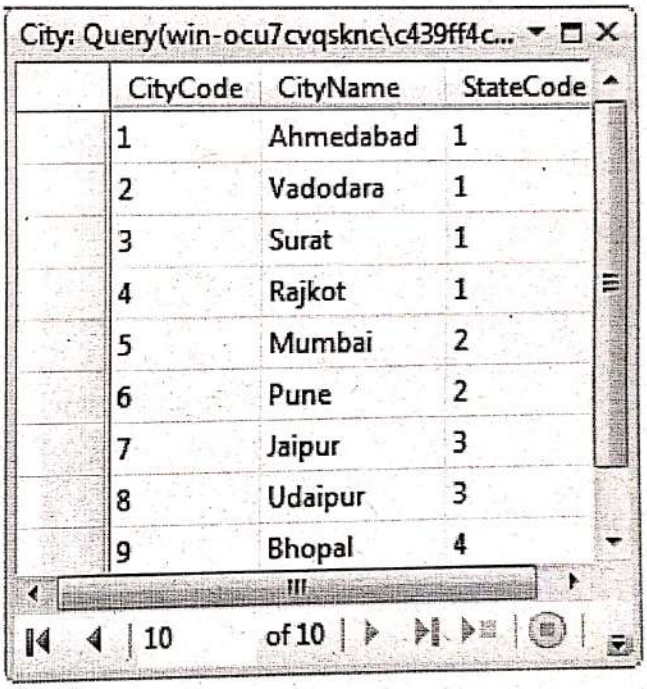
State table [Design]



City table [Design]



State table [Data Entry]



City table [Data Entry]

- Now write the following code on the Form_Load event as well as the Combobox1_SelectedIndexChange event of the 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.Data.SqlClient;

namespace Unit4_Q3
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        public static SqlConnection cn;
        private void Form1_Load(object sender, EventArgs e)
        {

 cn = new SqlConnection();
 cn.ConnectionString =
"DataSource=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\
Database1.mdf;Integrated Security=True;User Instance=True";
            cn.Open();
            SqlCommand cmd1 = new SqlCommand();
            cmd1.CommandText = "Select * from State";
            cmd1.Connection = cn;
            SqlDataReader dr;
            dr = cmd1.ExecuteReader();
            while (dr.Read())
            {
                comboBox1.Items.Add(dr[1]);
            }
            dr.Close();
            cmd1.Dispose();
        }

private void comboBox1_SelectedIndexChanged(object sender,
EventArgs e)
        {
         comboBox2.Items.Clear();
         SqlCommand cmd1 = new SqlCommand();
         cmd1.CommandText = "Select StateCode from  State where
StateName='" + comboBox1.Text + "'";
            cmd1.Connection = cn;
```

```
                int i = Convert.ToInt16 (cmd1.ExecuteScalar());
                cmd1.CommandText = "Select * from City Where
StateCode=" + i;
                SqlDataReader dr;
                dr = cmd1.ExecuteReader();
                while (dr.Read())
                {
                    comboBox2.Items.Add(dr[1]);
                }
                dr.Close();
                cmd1.Dispose();
            }
        }
    }
```

- Now run and test the application. Change different states from the comboBox1 and you can see comboBox2 will show different cities name depending upon state selected in comboBox1.

Program: 4

Write a program to create a table Emp with the following fields:

Field Name	Datatype
Eno	Integer
Ename	Varchar(20)
Salary	Single
Birthdate	Date
Designation	Varchar(20)

Design a form which performs Select, Insert, Update and Delete operations on the table Emp. On Insert Eno has been generated automatically.

- Create a new project in the Visual Studio called "Unit4_Q4". Add database file "Database1.mdf" in the solution (as we have added in the previous questions). Create a table in the database called "Emp" with the design format given in the question. Open the table in data entry mode and add at least 2 records in it.
- Design an appropriate interface for the form1, from the fields given in the question.
- Now declare the object of SqlConnection class, configure the connection string and open the connection (as per question 'Unit4_Q2'). Now rest of the code will be same as 'Unit4_Q1'. Instead of OLEDB namespace, we are using 'SqlClient'

namespace. So, obviously our OLEDBDataAdapter will be change to SqlDataAdapter.

- You just have to add the following code on the click event of cmdAdd, so that we can generate Eno automatically.

```
sqlCommand cmd = new sqlCommand();
cmd.CommandText = "Select Max(Eno) from Emp";
cmd.Connection = cn;
int Eid = Convert.ToInt16(cmd.ExecuteScalar());//Fetching last Eno
Eid = Eid + 1; //Adding 1 to Last Eno
txtENo.Text = Conver.Tostring(Eid); //Placing Eno to textbox
```

Reporting:

When we develop application, we are mainly designing two types of components in it: [1] Forms and [2] Reports. Forms are basically developed to give data input to the application. In all of the previous examples we have learn, how we can design the form using GUI based components. To take input from the user we are designing forms, and to make user input easier we are using GUI based (Window based) controls such as textbox, RichTextBox, ComboBox, listBox, radioButtons, checkbox etc. With the help of these control, we can provide user friendly environment as well as we can reduce overhead of data entry (For example selection of radio buttons to take input for gender is much more easier than using textbox).

We know that the data given by the user to the forms are stored in the database. User is giving data to the application because later on when it is required, application returns the data after processing it, in the form of information. Application takes data from the database; it will process the data (by grouping, by filtering and by summarizing) and giving that data to the user in the form of printable format. It called the output or Report. Reports are summarized and user can print it or export it into other formats like Word document or PDF format. Report do not allows user to do data entry. For example when you are buying something from the supermarket, shopkeeper enters your details as well as product details using form, and finally shopkeeper takes a printout of the Invoice (Bill) and give it to you. The copy of Invoice is called a report.

So, in the application user can use Forms for data entry or data manipulation, which will be recorded in the database. Whereas, reports takes the data from the database process it and produce printable or exportable output. In short, we can say Forms are for input and Reports are for output.

In the Visual Studio, there are many different ways are there to produce reports. 'Crystal Report' is popular tool to produce various types of reports in the Visual

Studio. In the next examples we will learn how to design reports into the Visual Studio using reporting tool 'Crystal Report'.

Program: 5

Create below mentioned Crystal Reports for the above application.

- **List all the Employees having salary more than 25000**
- **List all the Employees who are above the age 35.**
- **List all the Employees designation wise.**

[5.1] Generating Report for: List all the Employees having salary more than 25000

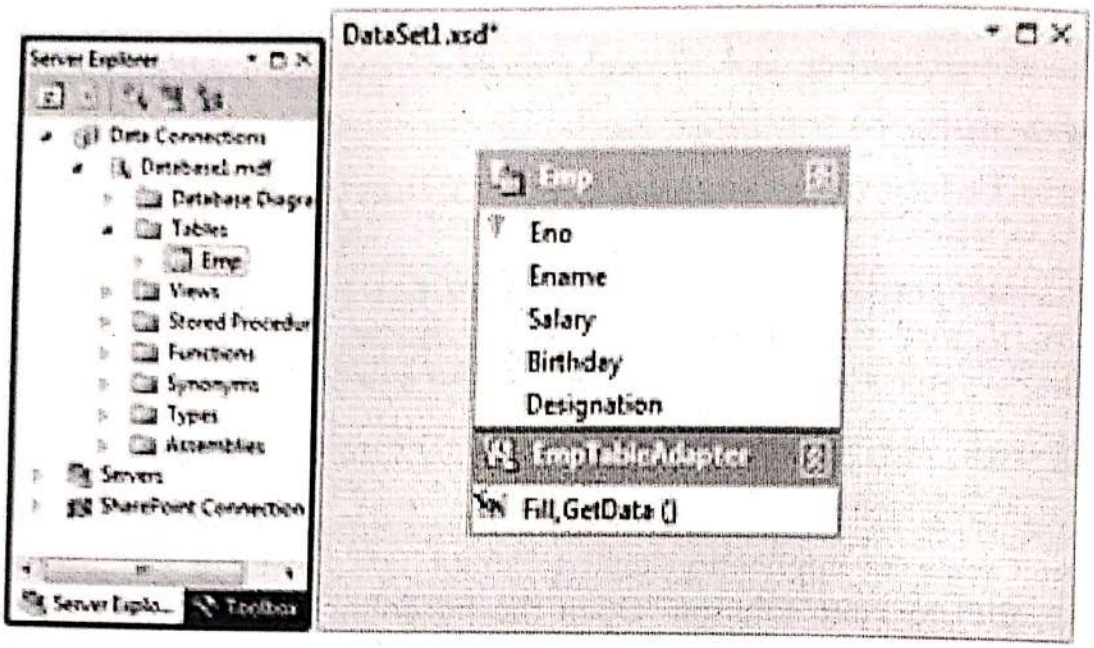
- Create an application called "Unit4_Q5". Insert a Database1.mdf (database file) to your solution. Open the by doing double click on the database file name in the solution explorer. Now right click on the 'tables' node of server explorer and click on 'Add New Table' option. This will create a new table in the design mode. Type the field names and its data type as given in the Program4 of Unit4. Now save the table as 'Emp'. Open the 'Emp' table in the datasheet mode for data entry by doing right click on 'Emp' in the server explorer and selecting the option 'Show table data' as given below:

Emp: Query(win-ocu7cvqsknc\c439ff4c-071f-47.C:\USERS\KAMES...

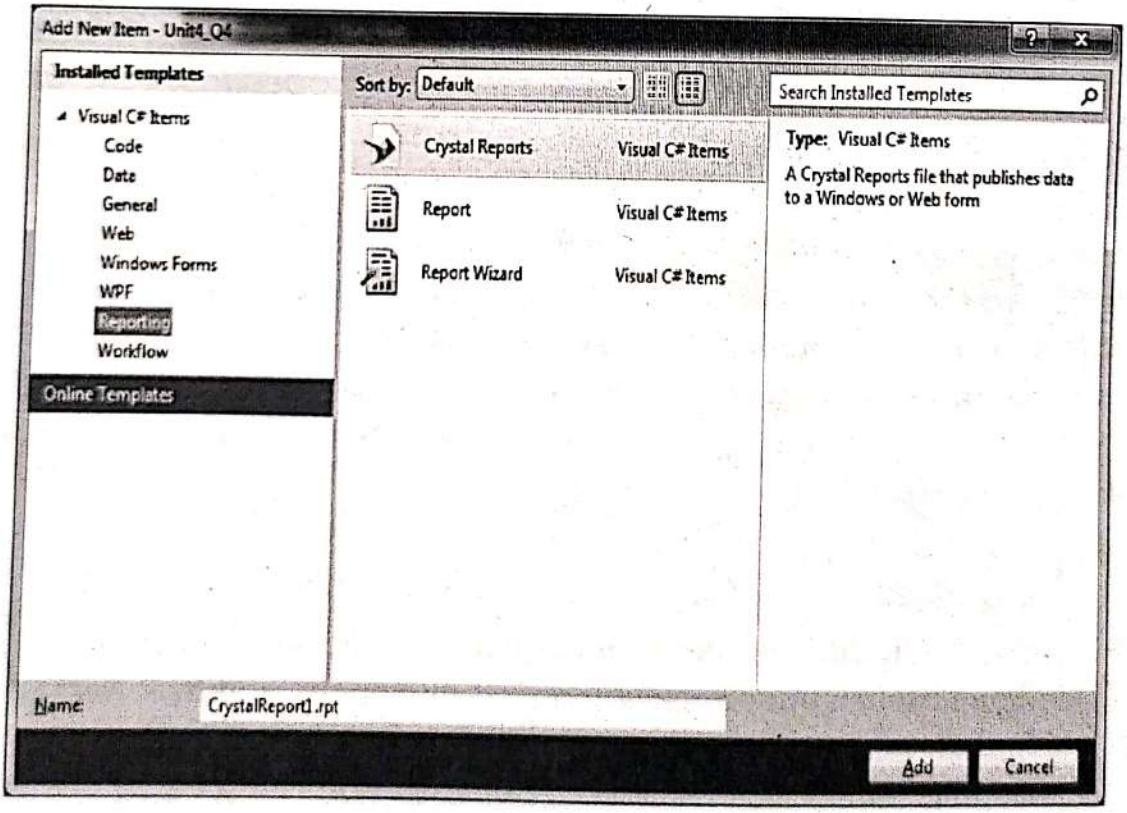
Eno	Ename	Salary	Birthday	Designation
1	Jack	35000	1/1/1978	General Manager
2	Jill	24000	2/2/1990	Clerk
3	Marry	20000	3/3/2000	Receptionist
4	Anthony	32000	4/4/1995	General Manager
5	Amar	25000	5/5/1990	Clerk
6	Geeta	28000	6/6/1976	Clerk
7	Supriya	32000	7/7/1999	Receptionist
NULL	NULL	NULL	NULL	NULL

8 of 8

- Now, Right click on the solution name, in the solution explorer, select option Add and select option 'New Item'. It will open 'Add new Item' dialog box. In this dialog box select 'Data' option from the Installed Template from the left side panel, and select DataSet option. Click on Add button, this will add a new file called DataSet1.xsd file to your project. Double click on the DataSet1.xsd file from the solution explorer. It will open DataSet1.xsd file in the center of the window (place where we are designing a form). Now drag and drop the table Emp from 'Server Explorer' (Left side panel) to 'DataSet1.xsd' (Center Window).

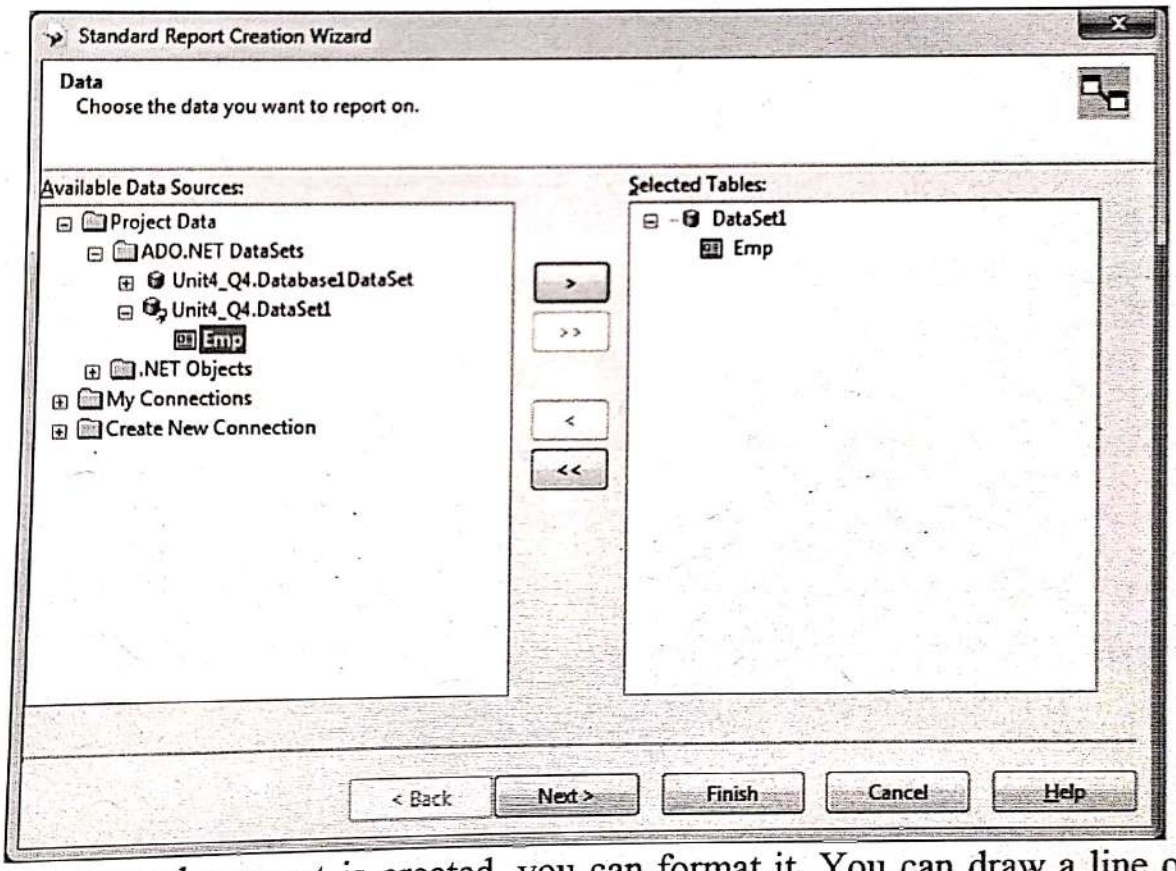


- Now, click on the Save button and save DataSet1.xsd file. Once the DataSet1.xsd file is saved, right click on the solution name of the solution explorer, click on Add and then select 'New Item' option. It will open 'Add New Item' dialog box. In this dialog box select 'Reporting' option from the Installed Template from the left side panel, and the select 'Crystal Reports' option.



- Now click on Add button. It will add a file called CrystalReport1.rpt to your project, and it will open the wizard to design the crystal report. From the first step of the wizard select option 'Using the Report Wizard' and the click on 'OK' button.

- In the second step of the wizard Expand 'Project Data' node, under this node expand 'ADO.NET Dataset' node, and under this you can get Unit4_Q4.DataSet1 node. This is our DataSet1.xsd file. If you expand this node you will get 'Emp' table (which we have dragged and dropped from server explorer to DataSet1.xsd file). Select the table "Emp" and click on > button as shown in the next figure.
- Now click on the 'Next' button it will show all the fields of "Emp" table. Here you have to select those fields which you want to display on the report. If you want to show all the fields then click on the button '>>', you can see all the fields of "Emp" table comes in the right side box. Click on the 'Next' button.
- Next screen of the wizard allows you to group the data. Read the question in the last report (designation wise report) we have to click on the designation field and then we have to click on > button. But for first 2 reports we do not have to do grouping. So, simply click on the 'Next', 'Next' and 'Finish' button.

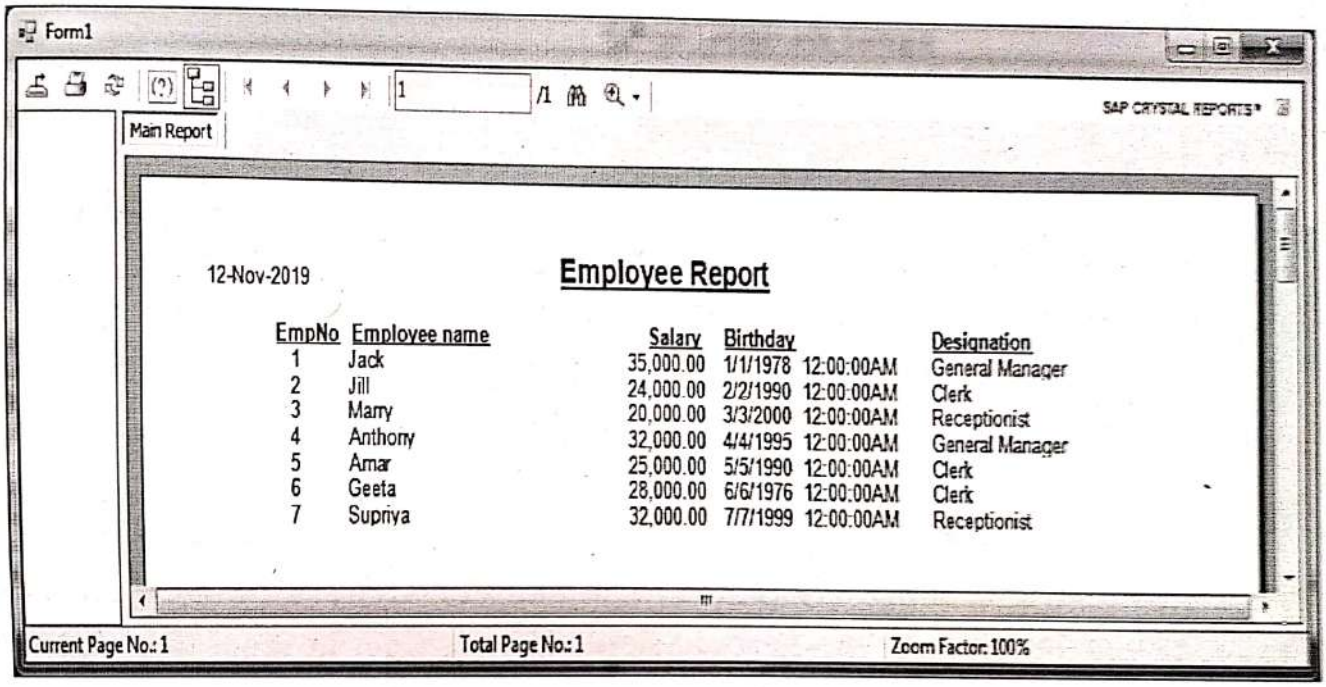


- Now once the report is created, you can format it. You can draw a line or a box or oval by doing right click on the report and the selecting option 'Insert'. You can add the TextBox, by doing right click on the report, selecting option 'Insert' and then clicking on the TextBox. You can also select any label or text and then by click on Bold, Italic, Underline from the toolbox. By using a toolbox you can also change the Font. You can select any text and change the format of it by right clicking on that object and selecting the option Format object. We are not going into the details of how the report will be formatted.

- Now if you are using Visual Studio 2010 then do the right click on the solution name of the solution explorer, go to the property and in the property window change the target framework option to "Framework 3.5".
- Now, from the toolbox, take a CrystalReportViewer control from the 'Reporting' category and place it on the form. Then write the following code on the 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.Data.SqlClient;
namespace Unit4_Q4
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        public static SqlConnection cn;
        SqlDataAdapter da;
        System.Data.DataSet ds;
        private void Form1_Load(object sender, EventArgs e)
        {
            cn = new SqlConnection();
            da = new SqlDataAdapter("Select * from Emp", cn);
            ds = new System.Data.DataSet();
            cn.ConnectionString = "Data
Source=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\Database1.mdf
;Integrated Security=True;User Instance=True";
            cn.Open();
            da.Fill(ds, "Emp");
            CrystalReport1 cr = new CrystalReport1();
            cr.SetDataSource(ds.Tables[0]);
            crystalReportViewer1.ReportSource = cr;
        }
    }
}
```

➢ **Run the application, you can find the report as shown below:**



[5.2] Generating Report for: List all the Employees who are above the age 35

➢ Now to generate second report, you just have to modify the code on the form_load event. Write the following code which will print the details of only those employees, whose age is more than 35.

```
private void Form1_Load(object sender, EventArgs e)
        {
            cn = new SqlConnection();
            DateTime dt, dt1;
            dt = DateTime.Now;
            dt1 = dt.AddYears(-35);
            da = new SqlDataAdapter("Select * from Emp where
BirthDay <'" + dt1 + "'", cn);
            ds = new System.Data.DataSet();
            cn.ConnectionString = "Data
Source=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\Database1.mdf
;Integrated Security=True;User Instance=True";
            cn.Open();
            da.Fill(ds, "Emp");
            CrystalReport1 cr = new CrystalReport1();
            cr.SetDataSource(ds.Tables[0]);
            crystalReportViewer1.ReportSource = cr;
        }
```

[5.3] Generating Report for: List all the Employees designation wise:

- In the third report, we have to group the data on the basis of 'designation' field. We have already added DataSet1.xsd file, and it contains schema for "Emp" table. Add a new crystal report called CrystalReport2.rpt.
- In the first Step of wizard select option: "using the Report Wizard" and "Standard" from "Choose an Expert" and click on OK button.
- In the next step of wizard expand "Project Data" node, expand "ADO.NET Datasets" node, expand "Unit4_Q4.DataSet1" node. In this node you will find "Emp" table. Click on "Emp" to select it and click on the button ">" and then click on the Next button of the wizard.
- In the next step of the wizard click on >> button to select all the fields for the reports and then click on Next button.
- Next step of the Wizard is for "Grouping" the data. We want to Group the data on the basis of 'designation' field. So select 'designation' field and click on > button. Now click on Next button.
- The next step of the wizard is summary step. In the 'Summarized Field' list you can see 'Sum of Emp.Eno' and 'Sum of Emp.Salary' because both are numerical fields. We don't want to do sum of EmpNo at the eand of each group. So, select field Emp.Eno from the "Summarized Field"and then select on < button to remove it. Now click on Finish button. Now you can get your report ready. Make the desire formatting into the report.
- Now add another Form to you project called 'Form2' and place Crystal Report Viewer control from the 'Reporting' reporting tab.
- Write the same code, which we have written to design the first report. It is also given below:

```
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 Unit4_Q4
{
    public partial class Form2 : Form
    {
```

```
        public Form2()

        {
            InitializeComponent();
        }
        public static SqlConnection cn;
        SqlDataAdapter da;
        System.Data.DataSet ds;
        private void Form2_Load(object sender, EventArgs e)
        {
            cn = new SqlConnection();

            da = new SqlDataAdapter("Select * from Emp", cn);
            ds = new System.Data.DataSet();
            cn.ConnectionString =
"DataSource=.\\SQLEXPRESS;AttachDbFilename=|DataDirectory|\\Database
1.mdf;Integrated Security=True;User Instance=True";
            cn.Open();
            da.Fill(ds, "Emp");
            CrystalReport3 cr = new CrystalReport3();
            cr.SetDataSource(ds.Tables[0]);
            crystalReportViewer1.ReportSource = cr;
        }
    }
}
```

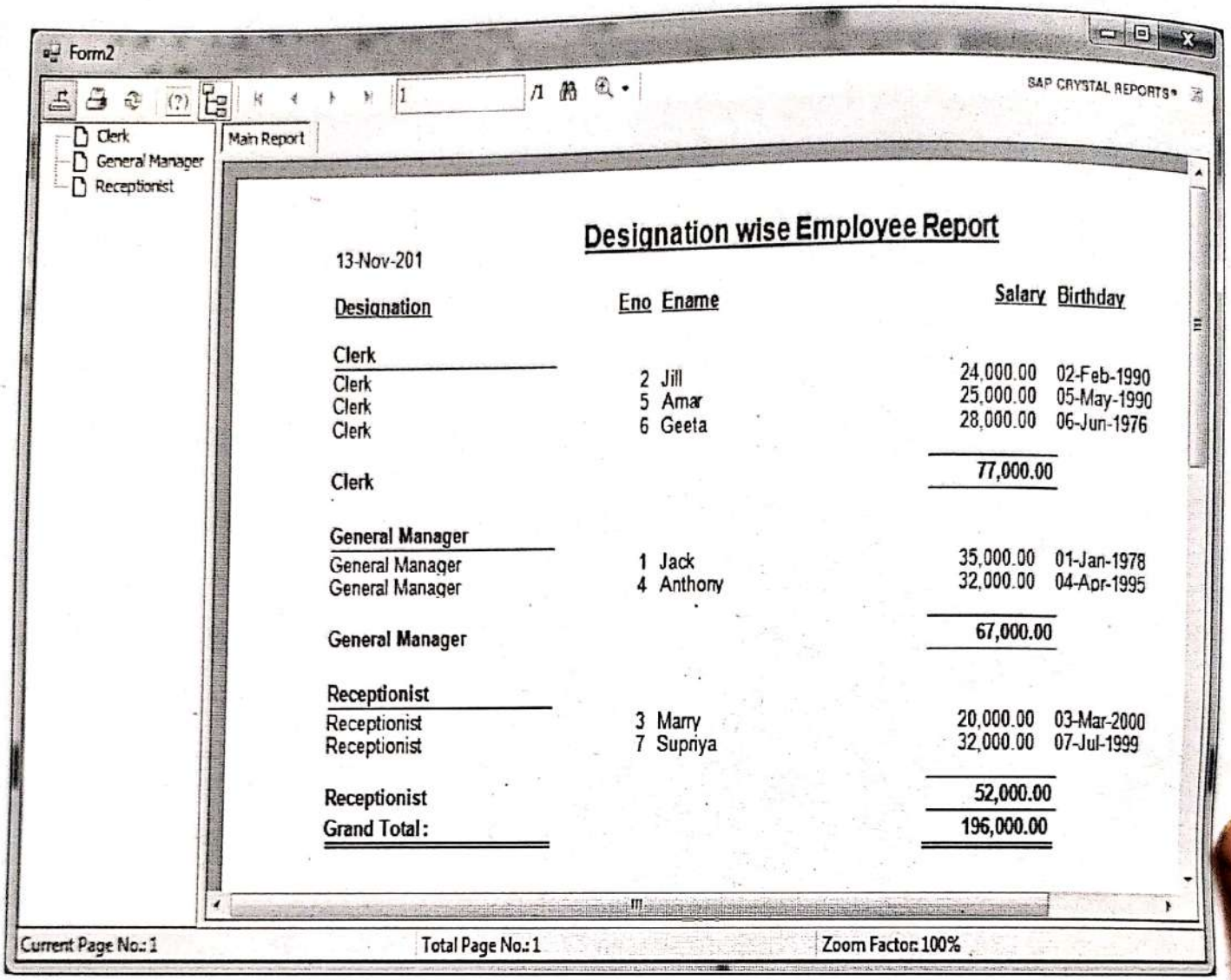
- Now we have to change the startup option of the project to Form2. So double click on the file Program.cs from solution explorer and change the line:

Application.Run(new Form1());

To

Application.Run(new Form2());

- **Run the application, it will load the CrystalReport2 in the Form2 as shown below:**



- See the toolbar of the window where you can get the icons (toolbar buttons) to Export the report in the PDF or .Doc format. You can also find another icon to print the Report.

Program: 6

Design a following table in Access. Table name = Books (Bookcode, Booktype, Bookpublisher, Bookauther, Bookpage, Bprice). Design Form that Display Above detail. Provide Add, Update, Delete, Next, first, last, pervious functionalities. Create Crystal Report for above application.

- Create a new Project in the Visual Studio called "Unit4_Q6", design the form1 according to the form fields given in the question. Place navigation buttons for First, Previous, Next and Last. Also placed Action buttons like Add, Update, Delete and Exit.

- Make a database in the MS-Access (As we have done in the Question:1 of this chapter) called Library.mdb. Design a table called "Books" with the fields given in the question: Books (Bookcode, Booktype, Bookpublisher, BookAuthor, Bookpage, Bprice) with appropriate fields, and enter at least 10 records in it. Make sure data entry you have to do in such as way that at least 2 or more authors should have 2 to 3 books (As we are designing the group by report on the basis of 'Author' field.)
- Perform the connectivity by writing the code as we have already discussed in the Program1 of Unit4.
- Now open the Server Explorer, right click on 'Data Connections' and then 'Add new connection' option. From the dialog box select 'Microsoft Access Database File' and then click on 'Continue' button.
- In the 'Add connection' dialog box click on the Browse button, and select your Access (.mdb) file and click on OK button.
- Now right click on the solution name (Unit4_Q6) in the solution explorer. Click on 'Add' and then 'Add New Item'. From the dialog box select 'Reporting' and then select 'Crystal Report' (which we are doing to add crystal report every time) and click on 'Add' button. This will add Database1.xsd file to your project.
- Go to the Solution Explorer, and expand data connection node, inside it you will get your database (Library.mdb) file. Also expand this node, under this node you can find Tables node. Under tables node you can find table 'Books'. Drag and drop this table to your opened Database1.xsd file. You can see the schema of Books table in the Database1.xsd file.
- Now add 'CrystalReport1.rpt' file as we have already done in the previous question 5(1).
- Write the code for fetching data, from the database to .NET application (as we have done in the previous question (5.1).
- Run and test the application.

Program: 7

Design a crystal report for the table given in the Q:6. Report should be Author wise and summarize it with Book count and Total of the price (Author wise).

- Create a new project in the 'Visual Studio'. Place the 'crystal report viewer' control in the form1, which you can get it from 'Reporting' tab of toolbox.
- If you fail to locate 'Crystal Report Viewer' in the reporting tab then you have to change the framework from the property of solution (refer question 5.1).
- If you have performed previous question (program: 6 of this chapter) then you have database file Library.mdb, and table 'Books' under this database.
- Add and design the DataSet1.xsd file with the schema of 'Books' table (refer question 6). Also add crystal report.
- In the previous question we have designed a simple report, but here you have to design a group by report.
- We have already discussed how to design group by report in question 5.3 of this chapter. In the question 5.3 we have designed a group by on the basis of 'Designation' field, and in the summary step of the wizard we have added 'sum of salary'. In this report you have to select 'Author' filed for grouping and in the summary step you have to add sum of price, and count of book code.
- You need to write the same code to connect your application to database, and bring the data from the database and load it to the report.
- Run and test the application.

☺ ☺ ☺ ☺ ☺

According to the New Syllabus of the

Gujarat University

Study Material for

Bachelor of Computer Application (BCA)

Exercise (MCQ, True-False, Fill in the Blanks) with Solution

More Books for BCA

Semester-1, Semester-2, Semester-3, Semester-4, Semester-5 and Semester-6

Shop Online
@
www.computerworld.ind.in

COMPUTER WORLD PUBLICATION
A Division of Live Education System Pvt. Ltd.
An ISO 9001:2008 Certified Company

COMPUTER WORLD PUBLICATION
43, 5th Floor, Sanidhya Complex, Opp. Sanyas Ashram,
Nr. M. J. Library, Ashram Road, Ahmedabad-380009.
Ph. : (079) 26580723 / 823, 97240 11150, 97250 22917
URL: www.computerworld.ind.in, www.cworld.co.in
e-Mail : info@computerworld.ind.in

978-93-88092-22-7
9 789388 092227
Computer World

Book Code : CEBCA118 | Price ₹ 125/-