(2½ hours)

Total Marks: 75

N. B.: (1) **All** questions are **compulsory**.
(2) Make **suitable assumptions** wherever necessary and **state the assumptions** made.
(3) Answers to the **same question** must be **written together**.
(4) Numbers to the **right** indicate **marks**.
(5) Draw **neat labeled diagrams** wherever **necessary**.
(6) Use of **Non-programmable** calculators is **allowed**.

1.	**Attempt *any two* of the following:**	**10**

a.	**What is an exception? Explain exception handling in C#.**
	Answer:
	Exception:
	An exception is any error, condition, or unexpected behavior that an application encounters. When an error occurs, an exception is thrown.
	
	Exception handling:
	The mechanism of Exception Handling is throwing an exception and catching it. C# uses try.... catch block. Code which may give rise to exceptions is enclosed in a try block, and catch block catches that exception and handles it appropriately. The try block is followed by one or more catch blocks.
	Example:
	using System;
	class tryCatch
	public static void Main()
	{
	int k=0;
	try
	{
	int n= 10/k;
	Console.WriteLine(”n=” + n);
	}
	catch(Exception e)
	{
	Console .Wri teLi ne (“Di Vi 5 Ofl by zero exception”);
	}
	Console.WriteLtne(”Statement executed after Exception because- of try catch”);
	}
	}
	Output:
	Division By zero exception
	
	Statement executed after Exception because of try catch.
	
	Finally block:
	Sometimes you want to define a block of code that will execute after a try....catch block.
	For instance, an exception might cause an error that terminates the current method, causing its premature return, and that program may have opened a file or a network connection that needs to be closed. To handle this, C# provides a convenient way: finally.	

b.	**What is polymorphism? Explain runtime polymorphism in C#.** **Answer:** **Polymorphism:**	

Polymorphism is the ability to take more than one form. For example, an operation may exhibit different behaviour in different situations. The behaviour depends upon the types of data used on the operation. Polymorphism is extensively used while implementing inheritance.

Runtime polymorphism:

It is also known as Inclusion polymorphism and achieved through the use of virtual functions. Assume that the class A implements a virtual method M & classes B & C that are derived from A override from A override the virtual method M. When B is cast to A, a call to the method M from A is dispatched to B. Similarly when C is cast to A, a call to M is dispatched to C. The decision on exactly which method to call is delayed until runtime & therefore it is also known as runtime polymorphism.

Since the method is linked with a particular class much later after compilation, this process is termed as late binding.

Example :

```
using System;
public class A
{
   public virtual void Print()
   {
      System.Console.WriteLine("Virtual Print method from A");
   }
}

public class B:A
{
   public override void Print()
   {
      System.Console.WriteLine("Override Print method from B");
   }
}
class Program
  {
     static void Main(string[] args)
     {
       A a=new A();
       a.Print();
     }
  }
```

Output:
Virtual Print method from A

c. What is the difference between for loop and foreach loop?

Answer:

	for loop		foreach loop
1	In case of for the control variable of the loop is always be int only.	1	In case of Foreach the control variable of the loop while be same as the type of values under the array.
2	The for loop executes the statement or block of statements repeatedly until specified expression evaluates to false.	2	The Foreach statement repeats a group of embedded statements for each element in

			an array or an object collection.	
3	There is need to specify the loop Bounds(Minimum, Maximum).	3	We do not need to specify the loop bounds (Minimum, Maximum).	
4	For loop is complex than foreach loop	4	Forech loop is simple than for loop	

5	example:
	using sytem;
	class class1
	{
	static void Main()
	{
	int j=0;
	for(int i=0; i<=10;i++)
	{
	j=j+1;
	}
	Console.ReadLine();
	}
	}
		5	example:
			using sytem;
			class class1
			{
			static void Main()
			{
			int j=0;
			int[] arr=new int[]
			{0,3,5,2,5,34,6,3,42,23};
			foreach(int i in arr)
			{
			j=j+1;
			}
			Console.ReadLine();
			}
			}	

d. **Write short note on Common Language Runtime (CLR) in .NET.**

Answer:

Common Language Runtime:

Common Language Runtime (CLR) is the programming (Virtual Machine component) that manages the execution of programs written in any language that uses the .NET Framework, for example C#, VB.Net, F# and so on.

We can say that it is heart and soul of .Net Framework or backbone.

Components of the CLR:

CTS

Common Type System (CTS) describes a set of types that can be used in different .Net languages in common. That is, the Common Type System (CTS) ensure that objects written in different .Net languages can interact with each other. For Communicating between programs written in any .NET complaint language, the types have to be compatible on the basic level.

These types can be Value Types or Reference Types. The Value Types are passed by values and stored in the stack. The Reference Types are passed by references and stored in the heap.

CLS

It is a subset of CTS. All instruction is in CLS i.e. instruction of CTS is written in CLS.

MSIL

It is language independent code. When you compile code that uses the .NET Framework library, you don't immediately create operating system - specific native code.

Instead, you compile your code into Microsoft Intermediate Language (MSIL) code. The MSIL code is not specific to any operating system or to any language.

Functions of the CLR

- Garbage Collector
- Exception handling
- Type safety
- Memory management (using the Garbage Collector)
- Security
- Improved performance

2.	**Attempt *any two* of the following:**	**10**
a.	**Differentiate between private assembly and shared assembly.** **Answer:** **Public Assembly** | **Private Assembly** Public assembly can be used by multiple applications. | Private assembly can be used by only one application. Public assembly is stored in GAC (Global Assembly Cache). | Private assembly will be stored in the specific application's directory or sub-directory. Public assembly is also termed as shared assembly. | There is no other name for private assembly. Strong name has to be created for public assembly. | Strong name is not required for private assembly. Public assembly should strictly enforce version constraint. | Private assembly doesn't have any version constraint.	
b.	**Explain how garbage collection works in .NET.** **Answer:** In the common language runtime (CLR), the garbage collector serves as an automatic memory manager. It provides the following benefits: Enables you to develop your application without having to free memory. Allocates objects on the managed heap efficiently. Reclaims objects that are no longer being used, clears their memory, and keeps the memory available for future allocations. Managed objects automatically get clean content to start with, so their constructors do not have to initialize every data field. Provides memory safety by making sure that an object cannot use the content of another object.. The managed heap: After the garbage collector is initialized by the CLR, it allocates a segment of memory to store and manage objects. This memory is called the managed heap, as opposed to a native heap in the operating system. Generations The heap is organized into generations so it can handle long-lived and short-lived objects. Garbage collection primarily occurs with the reclamation of short-lived objects that typically occupy only a small part of the heap. There are three generations of objects on the heap: **Generation 0**. This is the youngest generation and contains short-lived objects. An example of a short-lived object is a temporary variable. Garbage collection occurs most frequently in this generation. Newly allocated objects form a new generation of objects and are implicitly generation 0 collections, unless they are large objects, in which case they go on the large object heap in a generation 2 collection. Most objects are reclaimed for garbage collection in generation 0 and do not survive to the next generation. **Generation 1**. This generation contains short-lived objects and serves as a buffer between short-lived objects and long-lived objects. **Generation 2**. This generation contains long-lived objects. An example of a long-lived object is an object in a server application that contains static data that is live for the duration of the process.	
c.	**What is selector in CSS? Explain various types CSS selectors.** **Answer:** **Universal Selector**	

The Universal selector, indicated by an asterisk (*), applies to all elements in your page. The Universal selector can be used to set global settings like a font family. The following rule set changes the font for all elements in your page to Arial:

```
*{
font-family: Arial;
}
```

Type Selector

The Type selector enables us to point to an HTML element of a specific type. With a Type selector all HTML elements of that type will be styled accordingly.

```
h1
{
color: Green;
}
```

This Type selector now applies to all <h1> elements in your code and gives them a green color.
Type
Selectors are not case sensitive, so you can use both h1 and H1 to refer to the same heading.

ID Selector

The ID selector is always prefixed by a hash symbol (#) and enables us to refer to a single element in the page. Within an HTML or ASPX page, you can give an element a unique ID using the id attribute. With the ID selector, we can change the behavior for that single element, for example:

```
#IntroText
{
font-style: italic;
}
```

Because you can reuse this ID across multiple pages in your site (it only has to be unique within a single page), you can use this rule to quickly change the appearance of an element that you use once per page, but more than once in your site, for example with the following HTML code:

```
<p id="IntroText">I am italic because I have the right ID. </p>
```

Class Selector

The Class selector enables us to style multiple HTML elements through the class attribute. This is handy when we want to give the same type of formatting to a number of unrelated HTML elements. The following rule changes the text to red and bold for all HTML elements that have their class attributes set to highlight:

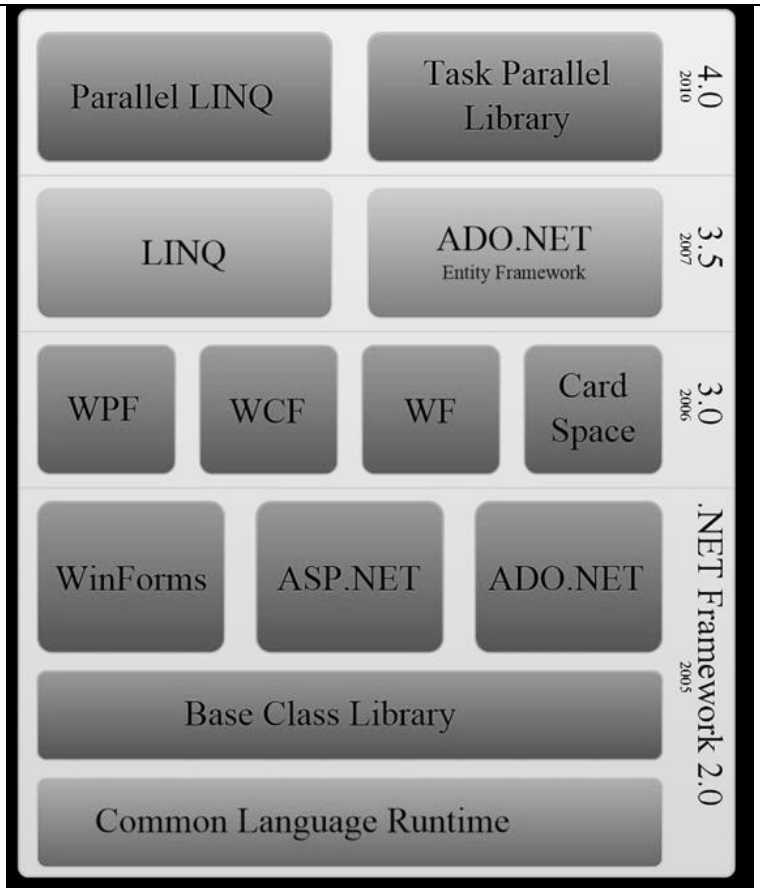
```
.Highlight
{
font-weight: bold;
color: Red;
}
```

d. What is .NET framework? Explain various components of .NET framework 4.0.

Answer:

The .NET Framework is a development platform for building apps for web, Windows, Windows Phone, Windows Server, and Microsoft Azure. It consists of the common language runtime (CLR) and the .NET Framework class library, which includes a broad range of functionality and support for many industry standards.

The .NET Framework provides many services, including memory management, type and memory safety, security, networking, and application deployment. It provides easy-to-use data structures and APIs that abstract the lower-level Windows operating system. You can use a variety of programming languages with the .NET Framework, including C#, F#, and Visual Basic.



3.	**Attempt *any two* of the following:**	**10**
a.	**List and explain the major events in global.asax file.** **Answer:** Application_Init The Application_Init event is fired when an application initializes the first time. Application_Start The Application_Start event is fired the first time when an application starts. Session_Start The Session_Start event is fired the first time when a user's session is started. This typically contains for session initialization logic code. Application_BeginRequest The Application_BeginRequest event is fired each time a new request comes in. Application_EndRequest The Application_EndRequest event is fired when the application terminates. Application_AuthenticateRequest The Application_AuthenticateRequest event indicates that a request is ready to be authenticated. If you are using Forms Authentication, this event can be used to check for the user's roles and rights. Application_Error The Application_Error event is fired when an unhandled error occurs within the application. Session_End The Session_End Event is fired whenever a single user Session ends or times out. Application_End The Application_End event is last event of its kind that is fired when the application ends or times out. It typically contains application cleanup logic.	
b.	**What is the code behind and inline code?** **Answer:** **Code Behind:**	

Code Behind refers to the code for an ASP.NET Web page that is written in a separate class file that can have the extension of .aspx.cs or .aspx.vb depending on the language used. It allows a separation of HTML from the presentation logic.

Example: code in .cs

```
Protected void MyButton_OnClick(Object sender, EventArgs e)
  {
    MyLabel.Text = MyTextbox.Text.ToString();
  }
```

Code in separate .aspx file:

```
    <body>
    <form id="MyForm" runat="server">
      <asp:textbox id="MyTextbox" text="Hello World" runat="server"/>
      <asp:button id="MyButton" text="Echo Input" OnClick="MyButton_OnClick"
runat="server"/>
      <asp:label id="MyLabel" runat="server"></asp:label>
    </form>
  </body>
</HTML>
```

Inline Code

Inline Code refers to the code that is written inside an ASP.NET Web Page that has an extension of .aspx. It allows the code to be written along with the HTML source code using a <Script> tag. It's major point is that since it's physically in the .aspx file it's deployed with the Web Form page whenever the Web Page is deployed.

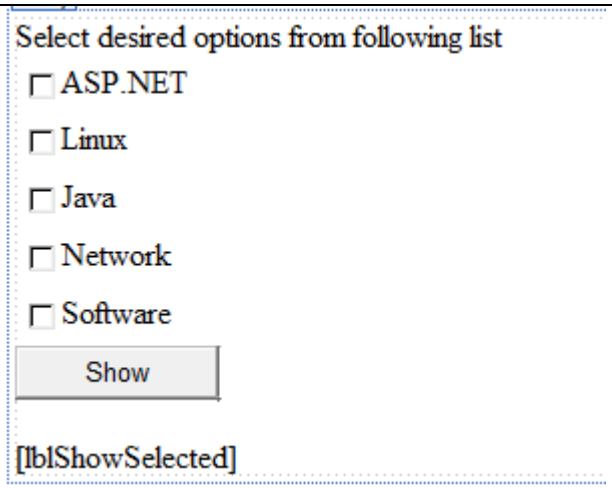
Following code is written in .aspx page:

```
<%@ Language=C# %>
<HTML>
  <script runat="server" language="C#">
  void MyButton_OnClick(Object sender, EventArgs e)
  {
    MyLabel.Text = MyTextbox.Text.ToString();
  }
  </script>
  <body>
    <form id="MyForm" runat="server">
      <asp:textbox id="MyTextbox" text="Hello World" runat="server"/>
      <asp:button id="MyButton" text="Echo Input" OnClick="MyButton_OnClick"
runat="server"/>
      <asp:label id="MyLabel" runat="server"></asp:label>
    </form>
  </body>
</HTML>
```

c. **Write ASP.NET code to display selected elements from the CheckBoxList on a Label control. Elements on the label must be separated by a whitespace.**

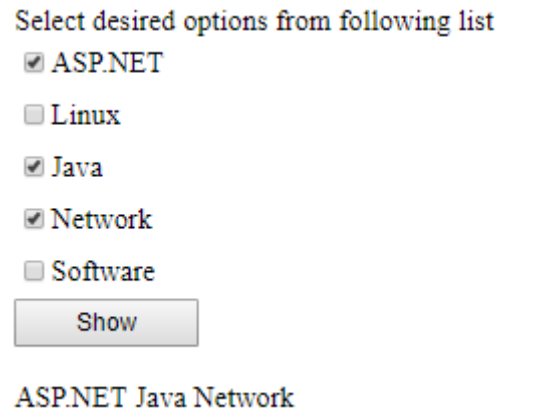
Answer:

Consider the following web page design




```
protected void btShow_Click(object sender, EventArgs e)
    {
        string strAllSelectect="";
        foreach (ListItem val in chkSubjectsList.Items)
        {
            if (val.Selected)
            {
                strAllSelectect += val.Value + " ";
            }
        }
        lblShowSelected.Text = strAllSelectect;
    }
```

Output:



d. **Explain following properties of ListBox control**
(i) AutoPostBack (ii) Items

Answer:

AutoPostBack

Gets or sets a value indicating whether a postback to the server automatically occurs when the user changes the list selection.(Inherited from ListControl.).
If we set **AutoPostBack** property of the ListBox to True, then form containing Listbox is submitted automatically whenever a new item is selected.

Items:

This property enables you to obtain a reference to the list of items that are currently stored in the ListBox. With this reference, you can add items, remove items, and obtain a count of the items in the collection.
Methods:
ListBox1.Items.Clear();
ListBox1.Items.Add(new ListItem(TextBox1.Text));

4. **Attempt *any two* of the following:** **10**

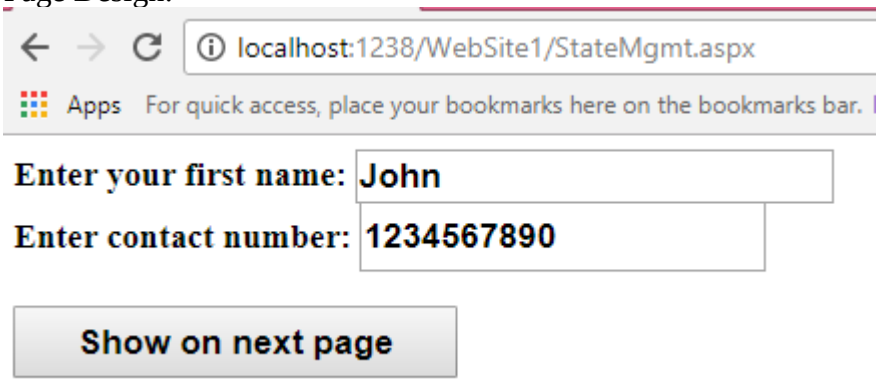
a.	**Write necessary properties which are common for all validation controls.** **Answer:** ControlToValidate ErrorMessage Text	
b.	**What is ViewState? How it works in ASP.NET?** **Answer:** **ViewState:** View State is the method to preserve the Value of the Page and Controls between round trips. It is a Page-Level State Management technique. View State is turned on by default and normally serializes the data in every control on the page regardless of whether it is actually used during a post-back. **Working:** All server controls have a property called ViewState. If this is enabled, the ViewState for the control is also enabled. Where and how is ViewState stored? When the page is first created all controls are serialized to the ViewState, which is rendered as a hidden form field named __ViewState. This hidden field corresponds to the server side object known as the ViewState. ViewState for a page is stored as key-value pairs using the System.Web.UI.StateBag object. When a post back occurs, the page de-serializes the ViewState and recreates all controls. The ViewState for the controls in a page is stored as base 64 encoded strings in name - value pairs. When a page is reloaded two methods pertaining to ViewState are called, namely the LoadViewState method and SaveViewState method. The following is the content of the __ViewState hidden field as generated for a page in my system. Listing 1 <input type="hidden" name="__VIEWSTATE" value="dNrATo45Tm5QzQ7Oz8AblWpxPjE9MMl0Aq765QnCmP2TQ==" /> **OR** **Using ViewState collection in code:** If we want to add one variable in View State, ViewState["Var"]=Count; For Retrieving information from View State string Test=ViewState["TestVal"];	
c.	**Explain TreeView control in ASP.NET.** **Answer:** A Tree View control displays a hierarchical list of data. When TreeView is displayed for the first time, it displays all its nodes. To display TreeView control in all your pages automatically, add it in the master page of the website. To create a TreeView control follows the following steps: (a) Open the web/master page in design View and add the TreeView control from the Navigation Toolbox to the page. **Syntax**: <asp:TreeView ID=”TreeView1” runat=”server’> </asp : TreeView> **Example:** <asp:TreeView ExpandDepth="1" runat="server"> <Nodes> <asp:TreeNode Text="Employees"> <asp:TreeNode Text="Bradley" Value="ID-1234" /> <asp:TreeNode Text="Whitney" Value="ID-5678" /> <asp:TreeNode Text="Barbara" Value="ID-9101" /> </asp:TreeNode> </Nodes> </asp:TreeView>	

d. **Write ASP.NET code to send data entered in two textboxes from one web page to another web page. Display the data on two separate labels.**

Answer:

[Answer must be written with the help of any one of the methods of state management like, QueryString, ViewState, and Session etc. Following example demonstrates use of QueryString]

Page Design:



Code(.cs file):

```
protected void btShowOnNextPage_Click(object sender, EventArgs e)
    {
        String name = Server.UrlEncode(txtName.Text);
        String contact = Server.UrlEncode(txtContactNo.Text);

        String Url = "DisplayData.aspx?" + "nm=" + name + "&contact=" + contact;

        Response.Redirect(Url);

    }
```

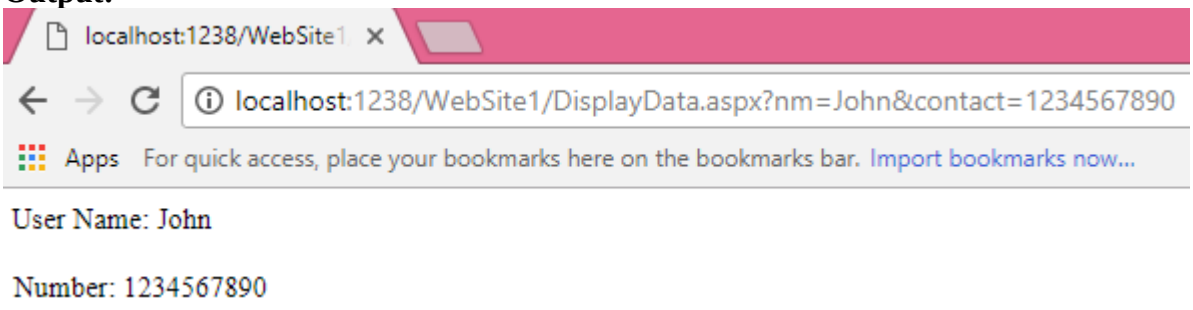
Code for next page(.cs file):

```
protected void Page_Load(object sender, EventArgs e)
    {
        String strName = Server.UrlDecode(Request.QueryString["nm"]);
        String strContact = Server.UrlDecode(Request.QueryString["contact"]);


        lblName.Text = "User Name: " + strName;
        lblContact.Text = "Number: " + strContact;

    }
```

Output:



5. **Attempt *any two* of the following:** **10**

a. **What are the authentication modes in ASP.NET for security?**

Answer:

There are 3 types of authentication as follows

1.Windows authentication

2.Forms authentication
3.Passport authentication

1.Windows authentication

It causes the browser to display a login dialog box when the user attempts to access restricted page. It uses windows user accounts and directory rights to grant access to restricted pages.

```
<system.web>
  <authentication mode="windows" >
      <Forms name="authDemo" loginurl="login.aspx"/>
  </authentication
</system.web>
```

2.Forms authentication

Developer codes a login form that gets the user name and password. The forms authentication provider uses custom HTML forms to collect authentication information and lets you use your own logic to authenticate users.

```
<system.web>
  <authentication mode="forms" >
      <Forms name="authDemo" loginurl="login.aspx"/>
  </authentication
</system.web>
```

3.Passport authentication

Passport authentication is a centralized authentication service provided by Microsoft that offers a single logon and core profile services for member sites.

b. **What is the difference between ExecuteScalar and ExecuteNonQuery?**

Answer:

ExecuteScalar():

Execute Scalar will return single row single column value i.e. single value, on execution of SQL Query or Stored procedure using command object. It's very fast to retrieve single values from database. Used to execute SQL Select command which is used to return a single value.
ExecuteScalar only returns the value from the first column of the first row of your query.

Example: string result = (string)cmd.ExecuteScalar();
Where cmd-is an object of SqlCommand class.

ExecuteNonQuery():

ExecuteNonQuery method will return number of rows effected with INSERT, DELETE or UPDATE operations. This ExecuteNonQuery method will be used only for insert, update and delete, Create, and SET statements.
ExecuteNonQuery does not return data at all. It returns only the number of rows affected by an insert, update, or delete.

Example:

int result= cmd.ExecuteNonQuery();
Where cmd-is an object of SqlCommand class.

c. **Write a code to insert and update data into a SqlServer database from an ASP.NET web page.**

Answer:

Insert Data:

```
protected void cmdAdd_Click(object sender, EventArgs e)
  {
    SqlConnection conn = new SqlConnection();
```

```
conn.ConnectionString = (@"Data Source=ADMIN-PC\SQLEXPRESS;Initial
Catalog=TYIT;Integrated Security=True");
string query = "INSERT INTO Student(RollNo,FName,Marks,Address)VALUES
('" + txtRollNo.Text + "','" + txtFName.Text + "','" + txtMarks.Text + "','" + txtAdd.Text + "')";

try
{
   conn.Open();
   SqlCommand cmd = new SqlCommand(query, conn);
   cmd.ExecuteNonQuery();

   lblMessage.Text="Data Added Successfully";
}
catch (Exception ex)
{
   lblMessage.Text="Problem in connection or in database";
}

finally
{
   conn.Close();
}
}
```

Update Data:

d. **Explain DataAdapter class in ADO.NET?**

Answer:

The DataAdapter is the class at the core of ADO NET's disconnected data access. The DataAdapter object which populates a disconnected DataSet with data and performs update It essentially the middleman facilitating all communication between the database and a DataSet. The DataAdapter is used either to fill a DataTable or DataSet with data from the database. After the memory-resident data has been manipulated, the DataAdapter can commit changes to the database by calling the Update method.

Properties:

DeleteCommand:
Represents a DELETE statement or stored procedure for deleting records from the data source

InsertCommand:
Represents an INSERT statement or stored procedure for inserting a new record to The data source

SelectCommand:
Represents a SELECT statement or stored procedure can be used to select records from a data source

UpdateCommand:
Represents an UPDATE statement or stored procedure for Updating recording in a data source

Important methods:

Fill():
This method fills data records from a DataAdapter to a DataSet object.

FillSchema():
This method adds a DataTable to a DataSet.

6.	**Attempt *any two* of the following:**	**10**
a.	**Write the basic syntax of a LINQ query in C#.** **Answer:** **Syntax:** from [identifier] in [source collection] where [Boolean expression] order by [[expression](ascending/descending)], [optionally repeat] select [expression] **Where**, A query expression must begin with a from clause and must end with a select or group clause. Between the first from clause and the last select or group clause, it can contain one or more of these optional clauses: where, orderby, join, let and even additional from clauses. You can also use the into keyword to enable the result of a join or group clause to serve as the source for additional query clauses in the same query expression.	

b.	**What are the benefits using Ajax? Explain UpdateProgress control in Ajax.**
	Answer:
	Benefits of Ajax:
	• Reduce the traffic travels between the client and the server.
	• Response time is faster so increases performance and speed.
	• Ready Open source JavaScript libraries available for use – JQuery, etc..
	• AJAX communicates over HTTP Protocol.
	
	The UpdateProgress control
	There are scenarios where a request may take time, an image may take time to load, business logic may a lot of time to calculate, and data may take time to load due to diversities in the source. In UpdatePanels, because postbacks are partial in nature, the browser's default progress bar does not appear. For avoiding any confusion to the end user that a page process is going on in the background the UpdateProgress control may be used. This is like a progress bar that appears to the end user, signifying that a process is going on. Usually, image is a moving .gif file.
	Example:
	<form id="form1" runat="server">
	<asp:ScriptManager ID="ScriptManager1" runat="server" />
	<asp:UpdateProgress runat="server" id="PageUpdateProgress">
	<ProgressTemplate>
	Loading...
	</ProgressTemplate>
	</asp:UpdateProgress>
	<asp:UpdatePanel runat="server" id="Panel">
	<ContentTemplate>
	<asp:Button runat="server" id="UpdateButton" onclick="UpdateButton_Click" text="Update" />
	</ContentTemplate>
	</asp:UpdatePanel>
	</form>
	Code-Behind:
	protected void UpdateButton_Click(object sender, EventArgs e)
	{
	System.Threading.Thread.Sleep(5000);
	}	

c.	**Write a code to display all the number in array greater than 10 using LINQ.** **Answer:** static void Main() {	

```
    // array as data source.
    int[] arr = { 34, 7, 82, 8, 75, 6,12,3,38 };

    // Query Expression.
    IEnumerable<int> resultQuery =
        from n in arr
        where n > 10
        select score;

    // Execute the query to produce the results
    foreach (int num in resultQuery)
    {
        Console.WriteLine(num);
    }
}
```

Output:
34
82
75
12
38

d. **Explain the use of document.ready() function in jQuery.**

Answer:

Document.Ready:
All You jQuery methods in our examples are inside a document ready event:

```
$(document).ready(function(){

// jQuery methods go here...

});
```

This is to prevent any jQuery code from running before the document is finished loading (is ready). It is good practice to wait for the document to be fully loaded and ready, before working with it. This also allows you to have your JavaScript code before the body of your document, in the head section.

Here are some examples of actions that can fail if methods are run before the document is fully loaded:

- Trying to hide an element that is not created yet
- Trying to get the size of an image that is not loaded yet

The jQuery team has also created an even shorter method for the document ready event:

```
$(function(){

// jQuery methods go here...

});
```

7. **Attempt *any three* of the following:** **15**

a. **Explain switch statement. What is fallthrough in switch? Is fallthrough permitted in C#?**

Answer:

The switch statement tests the value of a given variable against a list of case values & when a match is found, a block of statements associated with that case is executed. The general form of the switch statement is as shown below:

```
switch(expression)
{
    case value-1:
                block-1
                break;
    case value-2:
                block-2
                break;
               …….
               …….
     default:
                default-block
                  break;
}
```

statement-x;

The switch statement is executed in the following order:

1. The expression is evaluated first.
2. The value of the expression is successively compared against the values, value-1, value-2,…. If a case is found whose value matches the value of the expression, then the block of statements that follows the case are executed.
3. The break statement at the end of each block signals the end of a particular case & causes an exit from the switch statement, transferring the control to the statement-x following the switch.
4. The default is an optional case. When present, it will be executed if the value of the expression does not match any of the case values. If not present, no action takes place when all matches fails & the control goes to the statement-x.

Fallthrough in switch:

Fallthrough means To pass into a particular condition, or situation. In the absence of break statement from the case the next case that come after the current case is additionally executed. This behaviour is called fallthough.

C# does not support an implicit fall through from one case label to another.

b. **How to create and use external style sheet using visual studio developer?**

Answer:

Steps creating a CSS Style Sheet:

1) In **Solution Explorer**, right-click the name of the website, then choose **Add New Item**…
2) From Add New Item… dialog box, select SyleSheet. You can change name of the style sheet from name box. Default name is StyleSheet.css.
3) Click Add button.

Style sheet has been added in the website. Add formatting elements in style sheet file.

Using style sheet:

```
<head runat="server">
    <link rel="Stylesheet" href="styletname.css" type="text/css" />
</head>
```

c. **Explain following two web server controls**
(i) LinkButton (ii) ImageButton
Answer:

LinkButton Control

The LinkButton control is used to create a hyperlink-style button on the Web page.

Important properties/methods:

PostBackUrl: This Specifies the URL of the page to post to from the current page when the LinkButton control is clicked.

OnClick: Attach a server side method that will fire when this button will be clicked.

ImageButton Control:

It is like an ASP.NET Button control, the only difference is, you have the ability to place your own image as a button. You use an image Button when you want your button to look different than the plain rectangular button.

Important properties/methods:

ImageUrl: Gets or Sets the location of the image to display as button control.

PostBackUrl: This Specifies the URL of the page to post to from the current page when the **LinkButton** control is clicked.

OnClick: Attach a server side method that will fire when this button will be clicked.

d. **What are cookies? Explain various properties of HttpCookie class**

Answer:

Cookie is a small amount of data that server creates on the client. When a web server creates a cookie, an additional HTTP header is sent to the browser when a page is served to the browser. The HTTP header looks like this:

Creating cookie

```
protected void btnAdd_Click(object sender, EventArgs e)
{
   Response.Cookies[“message”].Value = txtMsgCookie.Text;
}
```

// Here txtMsgCookie is the ID of TextBox.

// cookie names are case sensitive. Cookie named message is different from setting a cookie named Message.

The above example creates a session cookie. The cookie disappears when you close your web browser. If you want to create a persistent cookie, then you need to specify an expiration date for the cookie.

```
Response.Cookies[“message”].Expires = DateTime.Now.AddYears(1);
```

Reading Cookies

```
void Page_Load()
{
    if (Request.Cookies[“message”] != null)
    lblCookieValue.Text = Request.Cookies[“message”].Value;
}
```

// Here lblCookieValue is the ID of Label Control.

.

e. **What is the authentication and authorization in ASP.NET?**

Answer:

<u>Authentication</u>:

Authentication is the process of verifyng the identity of a user using some credentials like username and password while authorization determines the parts of the system to which a particular identity has access. Authentication is required before authorization.

Authorization: Authorization is the process of allowing an authenticated user access to resources. Authentication is always precedes to Authorization; even if your application lets anonymous users connect and use the application, it still authenticates them as being anonymous.

Here is an overview of the steps in the joint IIS and ASP.net authentication process.

IIS first checks to make sure the incoming request comes from an IP address that is allowed access to the domain. If not it denies the request.

Next IIS performs its own user authentication if it configured to do so. By default IIS allows anonymous access, so requests are automatically authenticated, but you can change this default on a per - application basis with in IIS.

If the request is passed to ASP.net with an authenticated user, ASP.net checks to see whether impersonation is enabled. If impersonation is enabled, ASP.net acts as though it were the authenticated user. If not ASP.net acts with its own configured account.

Finally the identity from step 3 is used to request resources from the operating system. If ASP.net authentication can obtain all the necessary resources it grants the users request otherwise it is denied. Resources can include much more than just the ASP.net page itself you can also use .Net's code access security features to extend this authorization step to disk files, Registry keys and other resources.

For e.g. If an employee authenticates himself with his credentials on a system, authorization will determine if he has the control over just publishing the content or also editing it.

f. **Write jQuery code to demonstrate the use of hide () and SlideUp() functions on <p> element.**
Answer:

Use of Hide() method:

```
<!DOCTYPE html>
<html>
<head>
  <style>
  p { color:red; margin:5px; cursor:pointer; }
  p:hover { background:yellow; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
  <p>First Paragraph</p>

  <p>Second Paragraph</p>
  <p>Yet one more Paragraph</p>
<script>
    $("p").click(function () {
      $(this).hide();
    });
</script>

</body>
</html>
```

Use of SlideUp() method:

```
<!DOCTYPE html>
<html>
<head>
  <style>
  p { color:red; margin:5px; cursor:pointer; }
  p:hover { background:yellow; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
```

```
</head>
<body>
  <p>First Paragraph</p>

  <p>Second Paragraph</p>
  <p>Yet one more Paragraph</p>
<script>
    $("p").click(function () {
      $(this).slideUp();
    });
</script>

</body>
</html>
```