CMR
INSTITUTE OF
TECHNOLOGY

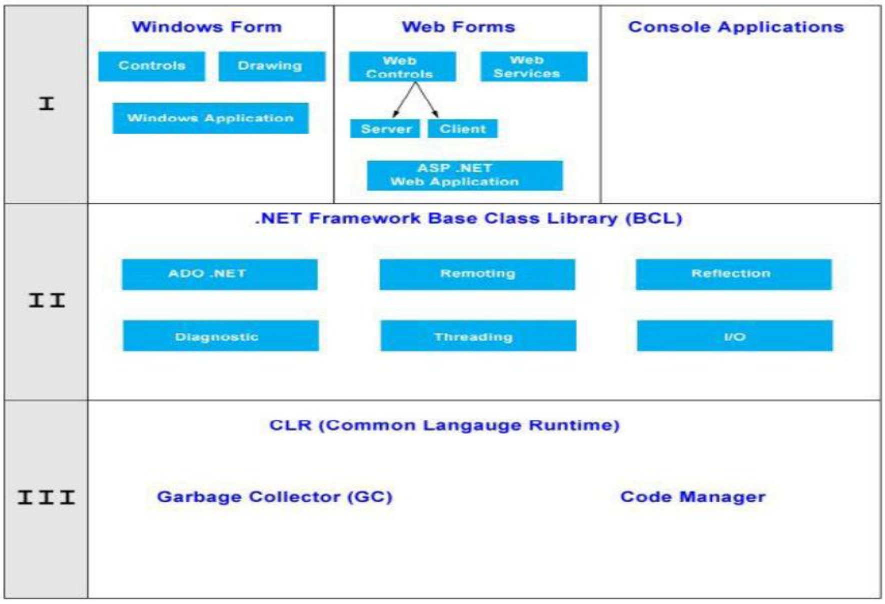
USN	1	C								

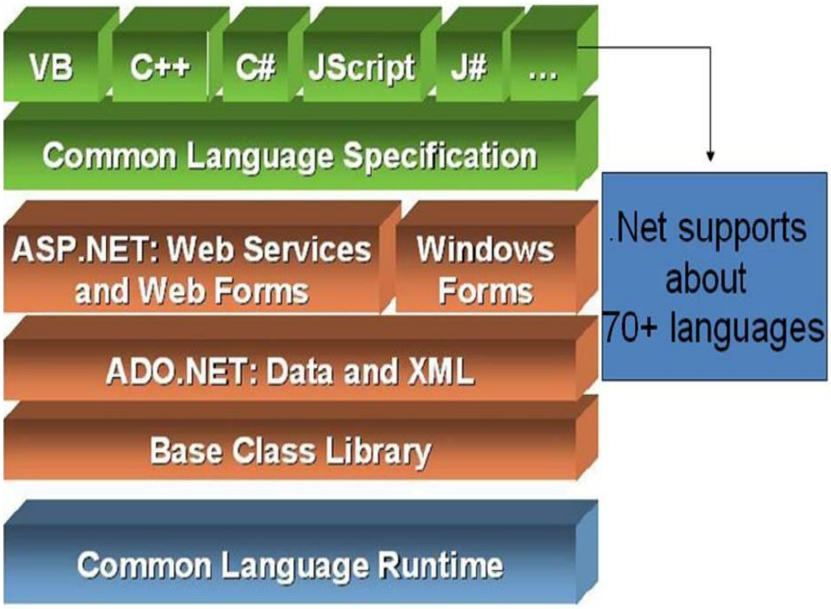
Internal Assessment Test 1 – September 2017

Sub:	**Programming using C#.NET**							**Code:**	13MCA53
Date:	19-09-17	Duration:	90 mins	Max Marks:	50	**Sem:**	V	**Branch:**	**MCA**

Note: Answer any 5 questions. All questions carry equal marks. Total marks: 50

1. a. **Explain the architecture and components of .NET framework**




VB
C++
C#
JScript
J#
…
Common Language Specification
ASP.NET: Web Services and Web Forms
Windows Forms
ADO.NET: Data and XML
Base Class Library
Common Language Runtime
.Net supports about 70+ languages

CLR: Provides run time environment to run the code and provide various services to develop the application.

CTS: Specify certain guidelines for declaring using and managing types at runtime.

Base Class library: Reusable types. Classes, interfaces, value types helps in speeding-up application development process.

CLS: Common Language Specification.

Windows forms: is the graphical representation of any windows displayed in an application.

Web application: Uses ASP.NET to build application.

ADO.NET: Provides functionality for database communication.

Programming Languages: C#, VB, J#,VC++ and more are supported in .NET environment

2. a. **Explain the benefits of .NET framework**.

Benefits of .NET framework:

Consistent Programming model

Cross-platform support

Language Interoperability

Automatic management of resources

Ease of Deployment

b. **What is an assembly? Explain each component of an assembly.**

In the .NET framework, an assembly is a compiled code library for use in deployment, Versioning and security.

There are two types: process assemblies (EXE) and library assemblies (DLL). A process assembly represents a process which will use classes defined in library assemblies.

.NET assemblies contain code in CIL, which is usually generated from a language, and then compiled into machine language at runtime by the CLR just-in-time compiler.

An assembly can consist of one or more files. Code files are called modules. An assembly can contain more than one code module and since it is possible to use different languages to create code modules it is technically possible to use several different languages to create an assembly.

3. a. **Discuss panel control with respect to windows forms**.

The Panel control is similar to the GroupBox control; however, only the Panel control can have scroll bars, and only the GroupBox control displays a caption.

How to use Panel Control
Drag and drop Panel control from toolbox on the window Form.
Collection of control can be placed in side Panel.

Transparent Panel

First set BackColor of Panel suppose you set red then
set Form's TransparencyKey property to the same color as Panel's background color –red in this case.
Example:

```
 private void frmPanel_Load(object sender, EventArgs e)
{
   //change back color of Panel
   panel1.BackColor = Color.Red;
   //set Form's TransparencyKey to the same color as Panel's back color
   this.TransparencyKey = Color.Red;
}
```

Now panel will be transparent when application run.

Panel Properties

BackColor: Panel BackColor can be changed through BackColor property.

Example:

```
private void frmPanel_Load(object sender, EventArgs e)
{
    //change back color of Panel
    panel1.BackColor = Color.CadetBlue;
}
```

BorderStyle: Get or set BorderStyle of Panel.

Example:

```
private void frmPanel_Load(object sender, EventArgs e)
{
        //Set Border style of Panel
        panel1.BorderStyle = BorderStyle.Fixed3D;
}
```

b. **Explain the implementation of combo box in C#**.

C# controls are located in the Toolbox of the development environment, and you use them to create objects on a form with a simple series of mouse clicks and dragging motions. A ComboBox displays a text box combined with a ListBox, which enables the user to select items from the list or enter a new value.

How add a item to combobox

```
comboBox1.Items.Add("Sunday");
comboBox1.Items.Add("Monday");
comboBox1.Items.Add("Tuesday");
```

ComboBox SelectedItem

retrieve the displayed item to a string variable ,

```
string var;
var = comboBox1.Text;
```

Or

```
var item = this.comboBox1.GetItemText(this.comboBox1.SelectedItem);
MessageBox.Show(item);
```

How to remove an item from ComboBox

```
  comboBox1.Items.RemoveAt(1);
```

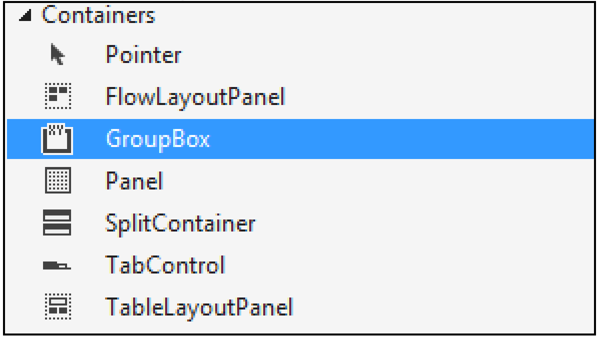
The above code will remove the second item from the combobox.

```
  comboBox1.Items.Remove("Friday");
```

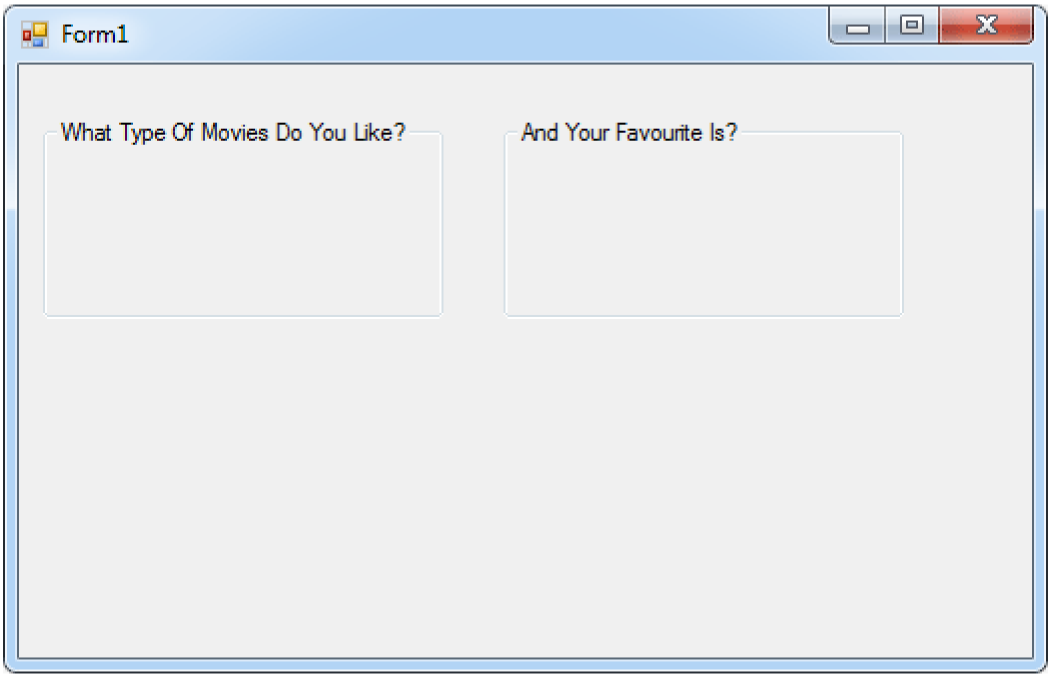
4. a. **Illustrate working with checkbox, Radio button and Group Box controls with a windows forms application example.**

Checkboxes allow a user to select multiple options, whereas Radio Buttons allow only one.

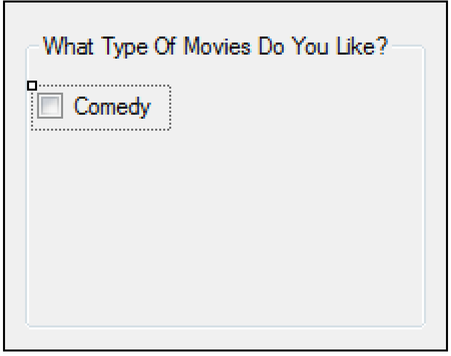
Locate the Groupbox control in the Toolbox on the left, under Containers. It looks like this:



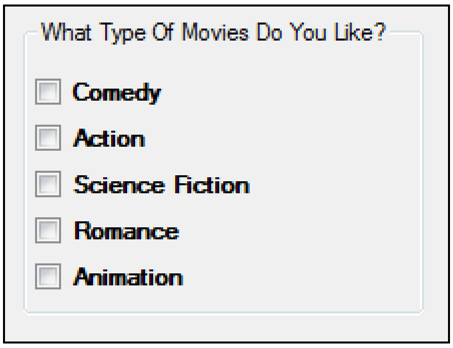
Draw one out on your form. Locate the Text property in the properties window on the right of C#. Change the Text property to What Type of Movies Do You Like?.
Add a second Groupbox along side of the first one, and set the Text property as And Your Favourite Is?. Your form will then look like this:



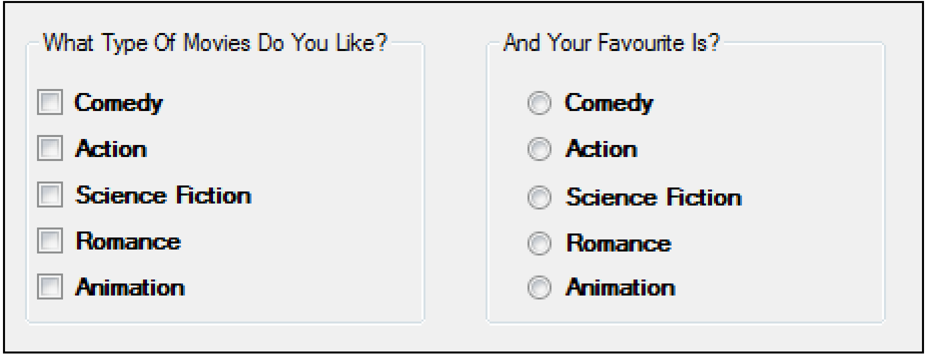
We'll place some Checkboxes on the first Groupbox, and some Radio Buttons on the second one.
Locate the Checkbox control on the toolbox, under Common Controls. Draw one out on your first Groupbox.
In the properties area on the right, notice that the default Name property is checkBox1. Leave it on this, but locate the Text property and change it to Comedy:



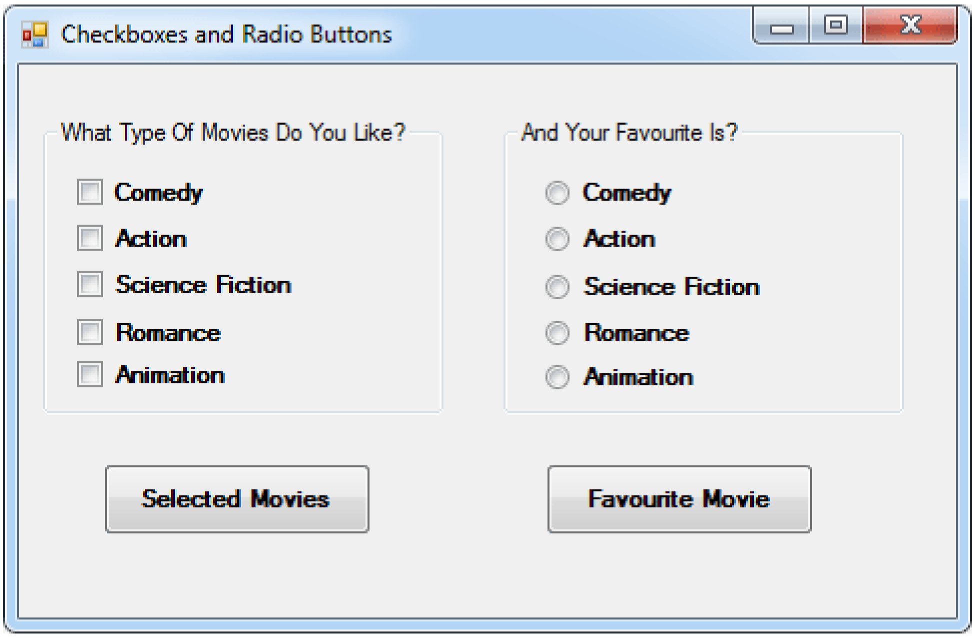
Draw four more checkboxes on the Groupbox, and set the Text properties as follows: Action, Science Fiction, Romance, Animation. (You can copy and paste the first one, instead of drawing them out.) Make the Text bold, and your Groupbox should look like this:



You add Radio Buttons in the same. So add five Radio Buttons to the second Groupbox. Leave the Name property on the defaults. But change the Text to the same as for the Checkboxes. Your form should look like ours below when you are finished:



Now add two buttons, one below each group box. Set the Text for the first one as Selected Movies. Set the Text for the second one as Favourite Movie. Here's what your form should look like now:



Run your form and test it out. What you should find is that you can select as many checkboxes as you like, but only one of the Radio Buttons.

Stop your program and return to Design Time.

What we'll do now is to write code to get at which selections a user made. First, the Checkboxes.

Double click your Selected Movies button to open up the code window. Our code will make use of the Checked property of Checkboxes. This is either true or false. It will be true if the user places a check in the box, and false if there is no check.

We can use if statements to test the values of each checkbox. We only need to test for a true value:

```
if (checkBox1.Checked)
{
}
```

We can also build up a message, if an option was selected:

```
string movies = "";
if (checkBox1.Checked)
```

```
{
movies = movies + checkBox1.Text;
}
MessageBox.Show(movies);
```

Inside of the if statement, we are building up the string variable we've called movies. We're placing the Text from the Checkbox into this variable.
Add a second if statement to your code:

```
string movies = "";
if (checkBox1.Checked)
{
movies = movies + checkBox1.Text;
}
if (checkBox2.Checked)
{
movies = movies + checkBox2.Text;
}
MessageBox.Show(movies);
```

5. a. **Write a short note on following:**

1. MDI

MDI child forms are an essential element of Multiple-Document Interface (MDI) Applications, as these forms are the center of user interaction.

create a new Windows Forms project. In the Properties Windows for the form, set its IsMdiContainer property to true, and its WindowsStateproperty to Maximized. This designates the form as an MDI container for child windows.
From the Toolbox, drag a MenuStrip control to the form. Set its Text property to File.
Click the ellipses (…) next to the Items property, and click Add to add two child tool strip menu items. Set the Text property for these items toNew and Window.
In Solution Explorer, right-click the project, point to Add, and then select Add New Item.
In the Add New Item dialog box, select Windows Form (in Visual Basic or in Visual C#) or Windows Forms Application (.NET) (in Visual C++) from the Templates pane. In the Name box, name the form Form2. Click the Open button to add the form to the project.

2. Event Driven GUI

In Visual C#, we can use either the Windows Form Designer or the Windows Presentation Foundation (WPF) Designer to quickly and conveniently create user interfaces. For information to help you decide what type of application to build, see Overview of Windows-based Applications. There are three basic steps in creating user interfaces:
Adding controls to the design surface.
Setting initial properties for the controls.
Writing handlers for specified events.

Programs with graphical user interfaces are primarily event-driven. They wait until a user does something such as typing text into a text box, clicking a button, or changing a selection in a list box. When that occurs, the control, which is just an instance of a .NET Framework class, sends an event to your application. We have the option of handling an event by writing a special method in our application that will be called when the event is received.

6 a.**List and explain any 5 ADO.NET Data Providers**.

ADO.NET Data Providers are class libraries that allow a common way to interact with specific data sources or protocols. The library APIs have prefixes that indicate which provider they support.

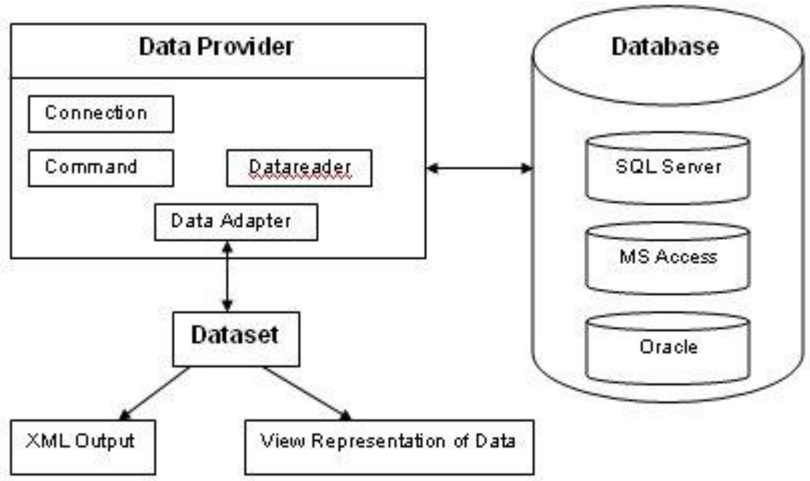
Provider Name	API prefix	Data Source Description
ODBC Data Provider	Odbc	Data Sources with an ODBC interface. Normally older data bases.
OleDb Data Provider	OleDb	Data Sources that expose an OleDb interface, i.e. Access or Excel.
Oracle Data Provider	Oracle	For Oracle Databases.
SQL Data Provider	Sql	For interacting with Microsoft SQL Server.
Borland Data Provider	Bdp	Generic access to many databases such as Interbase, SQL Server, IBM DB2, and Oracle.

b.**What is Connecting Object? Explain the procedure of connecting Database and retrieving value in List box in C#.NET.**

```
string cs = "Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=C:/Users/MCA/Desktop/samp1.accdb";
OleDbConnection con = new OleDbConnection(cs);
con.Open();
OleDbCommand cmd = new OleDbCommand();
cmd.Connection = con;
cmd.CommandText = "select ID from t1";
OleDbDataAdapter da = new OleDbDataAdapter(cmd);
DataSet ds = new DataSet();
da.Fill(ds);
DataTable dt = new DataTable();
dt = ds.Tables[0];
listBox1.DataSource = dt;
listBox1.DisplayMember = "ID";
con.Close();
```

7 a.**Explain the components of ADO.NET entity framework**

Entity Data Model: Defines the conceptual entities that can be read in serialized form using a DataReader
Entity SQL: Defines a common SQL based query language that is extended to express queries in terms of EDM concepts.
Entity Client: Provides a gateway for the queries of the entity level queries, which is queried through a common Entity SQL language.
Object Services: Allows us provides a more generic approach towards working with data - it uses anObject Query instance to process the data internally and execute your queries



8 a.**How data adapter is used to build database applications?**

Data adapters are an integral part of ADO.NET managed providers, which are the set of objects used to communicate between a datasource and a dataset. (In addition to adapters, managed providers include connection objects, data reader objects, and command objects.)

To create a data adapter manually

1. Make sure a connection object is available to the form or component that you are working with. For details about adding a standalone connection, see Establishing a Connection.
2. From the Data tab of the Toolbox, drag an OleDbDataAdapter, SqlDataAdapter, OdbcDataAdapter, OracleDataAdapter object onto the design surface.
   The designer adds an instance of the adapter to the form or component and launches the Data Adapter Configuration Wizard.
3. Close the wizard

b.**Write a C# program to add multiple tables to a DataSet**

```
string cs = "Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=C:/Users/MCA/Desktop/samp1.accdb";
OleDbConnection con = new OleDbConnection(cs);
con.Open();

OleDbDataAdapter d1 = new OleDbDataAdapter("select *from t1",con);
DataSet ds = new DataSet();
d1.Fill(ds,"[t1]");
dataGridView1.DataSource = ds.Tables["[t1]"];

OleDbDataAdapter d2 = new OleDbDataAdapter("select *from t2",con);
d2.Fill(ds,"[t2]");
dataGridView2.DataSource = ds.Tables["[t2]"];
  con.Close();
```