March-april-2018 paper solution

Q-1

a) 1 marks :-

1.full form of CLR .

→ Common Language Runtime.

2.Net framework based on Object Oriented Programming (true / false).

→ True

3. namespace keyword is used to include namespace.

4.define jagged array.

→ A Jagged array is an array of arrays.

→ A jagged array is an array whose elements are arrays, possibly of different sizes.

→ Syntax : int[][] jaggedArray = new int[3][];

b) 2 marks :-

1.explain foreach loop with example.

→ Foreach is a looping statement used with array.

→ We can perform loop based on array element using foreach.

→ Syntax : foreach (*type variableName* in *arrayName*)

{

```
        // code block to be executed
    }
```

→ Example:

```
string[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
foreach (string i in cars)
{
    Console.WriteLine(i);
}
```

2.what is managed code?

→ The code which is executed by CLR is known as Managed Code.

→ .NET framework is known as Managed Code. In other words, it is the code that gets executed directly in C#.

→ The managed code is always compiled by the CLR into an intermediate language known as MSIL and after that, the executable of the code is created. The intermediate language (MSIL) in the source code is compiled by the compiler named 'Just In Time Compiler' when the executable is run by the programmer.

→ Garbage collection is automatically implemented.

→ Dynamic type checking or runtime type checking is also provided.

c) 3 marks :-

1.what are value type and reference type?

→ value Types:-

- Value type are the typical primitive (પ્રાથમિક કક્ષાનું) types available in most programming language and are allocated on the stack.
- Value type variable contains the data itself
- When you copy a value type variable to another one, the actual data is copied and each variable can be independently manipulated.

- It has sub types like, Integer Types, Floating Point types, Decimal types, Boolean types, Character Types.

→ Reference Types :-

- Reference types are typically class instances and are allocated on the heap.
- When you copy a value type variable to another one, the actual data is copied and each variable can be independently manipulated.
- When coping a reference type variable to another variable, only the memory address is copied. Both variables will still point to the same memory location, which means, if you change value of one variable, the value will be changed for the other variable too.
- It has sub types like, The object types and the string types.

2.what are boxing and unboxing?

→ In C# it is possible to convert a value of one type into a value of another type.

→ Boxing:-

- The operation of Converting a Value type to a Reference type is called as Boxing.
- Type of Conversion: Implicit Conversion.
- Example: int i=1; Object obj=i;

→ Unboxing**:-**

- The operation of Converting a Reference type to Value type is called as Unboxing.
- Type of Conversion: Explicit Conversion
- Example: object obj=123; i=(int)obj;

d) 5 marks :-

1.explain the JIT and its type.

→ When any .NET application compile, it is not converted into machine code but it converted into MSIL (Microsoft Intermediate Language).

→ This code is a machine independent. So CLR provide Just-In-Time Compilation technology to convert the MSIL code into a platform/devicespecific code so that it can be executed on current machine.

→ The .NET provides three type of JIT compilers:-

- Pre-JIT:-
  - This JIT compiles an assembly's entire code into native code at one cycle.
  - Normally, it is used at installation time.
- Econo-JIT:-
  - This compiler is used on devices with limited resources.
  - It compiles the IL code bit-by-bit freeing resources used by the cached native code when required.
- Normal-JIT:-
  - The default JIT compiles code only as it called and places the resulting native code in the cache.
  - When, JIT compiles, the native code is placed into the cache, so that when the next call is made to the same method, the cached code is executed. So it increases the application speed.

2.explain architecture of .Net framework.

→Before starting the .NET framework we should know the definition of framework.

→A framework is a collection of classes, and applications which are libraries of SDKs (Software Development Kit) and APIs (Application Programming Interface) to help the different components all work together.

→A framework is a collection of classes, and applications which are libraries of SDKs (Software Development Kit) and APIs (Application Programming Interface) to help the different components all work together.

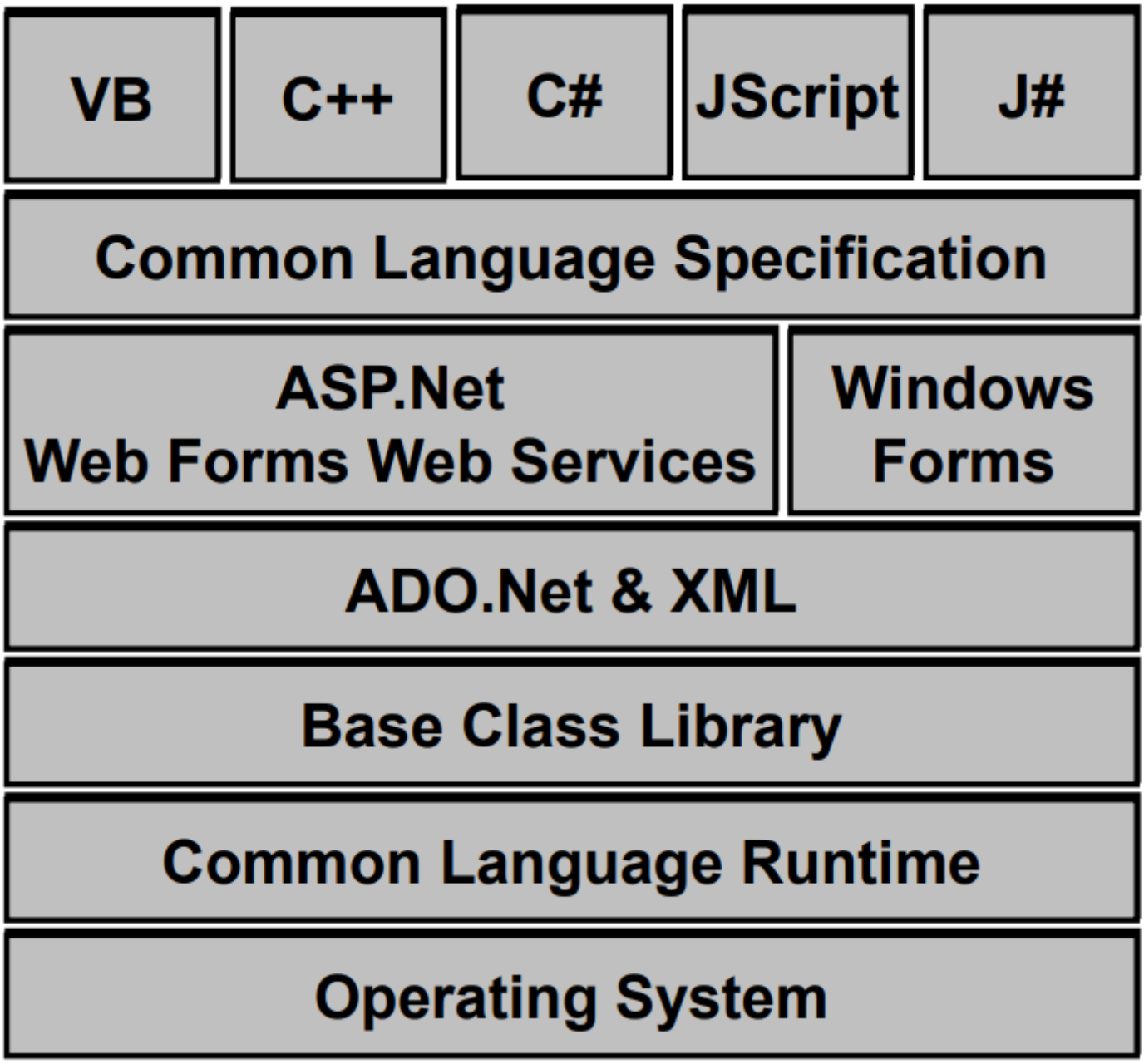
→It provides a large library and supports all the programming languages available in the market.

→It provides a large library and supports all the programming languages available in the market.

→Programs that are written in .NET framework are executed in a software environment, known as Common Language Runtime (CLR).

→Programs that are written in .NET framework are executed in a software environment, known as Common Language Runtime (CLR).

→Programs that are written in .NET framework are executed in a software environment, known as Common Language Runtime (CLR).



→The .NET Framework's Base Class Library provides user interface, data access, database connectivity, cryptography, web application development, numeric algorithms and network communications.

→Programmers produce software by combining their own source code with the .NET Framework and other libraries.

→Programmers produce software by combining their own source code with the .NET Framework and other libraries.

Q-2

a) 1 marks :-

1.define object:

→ An object is an instance of a class.

→ A class will not occupy any memory space. Therefore to work with the data represented by the class we have to create a variable for the class which is called as object.

→ When an object is created using the new operator, memory is allocated for the class in the big size (heap= ઢગલો), the object is called an instance and its starting address will be stored in the object in stack memory.

→ But when an object is created without the new operator then memory will not allocated in to the heap that means instance of the class is still not created.

→ An object is characterized by concept like, θ Attribute, Behavior & Identity.

2.overloading and overriding both are concept of polymorphism.(true/false)

→ **True**

3._____is used to implement multiple inheritance in c#.

→ Interfaces

4.by default a class member is _____to its class.

→ **public**

b) 2 marks :-

1.explain get and set accessor properties.

→ Property in C# is a member of a class that provides a flexible mechanism for classes to expose private fields.

→ Internally, C# properties are special methods called accessors.

→ A C# property have two accessors, get property accessor and set property accessor.

→ A get accessor returns a property value, and a set accessor assigns a new value.

→ The value keyword represents the value of a property.

→ Syntax:-

```
<access modifier> <return type>
<property name>
{
    Get
    {
    }
    Set
    {
```

```
        }
    }
```

Example : -

```
class MyClass
{
    private int x; public int X
    {
        Get
        {
            return x;
        }
        Set
        {
            x = value;
        }
    }
}

class PropertyDemo
{
    public static void Main()
    {
        MyClass mc = new MyClass();
        mc.X = 10;
        int xVal = mc.X;
        Console.WriteLine(xVal);
        //Displays 10
    }
}
```

2.explain indexer in c#.

→ C# indexers are usually known as smart arrays.

→ A C# indexer is a class property that allows you to access a member variable of a class or struct using the features of an array.

→ In C#, indexers are created using this keyword.

→ Indexers in C# are applicable on both classes and structs.

→ Defining an indexer allows you to create a class like that can allows its items to be accessed an array.

→ Instances of that class can be accessed using the [] array access operator.

→ Syntax :-

```
<modifier> <return type> this[argument list]
{
    Get
    {
        // your get block code
    }
    set
    {
        // your set block code
    }
}
```

→ Example :-

```
class IndexerClass
{
    private string[] names = new string[10];
        public string this[int i]
    {
        Get
        {
            return names[i];
        }
        Set
        {
            names[i] = value;
        }
```

```
    }
}
class IndexerDemo
{
        static void Main(string[] args)
        {
                IndexerClass Team = new IndexerClass();
                Team[0] = "Rajesh";
                Team[1] = "Teena";
                Team[2] = "Raj";
                Team[3] = "Mahesh";
                Team[4] = "Jignesh";
                Team[5] = "Viraj";
                Team[6] = "Sahil";
                Team[7] = "Parth";
                Team[8] = "Haresh";
                Team[9] = "Naresh";
                //Team[10] = "Naresh";
                //its out of bound array
                for (int i = 0; i < 10; i++)
                {
                        Console.WriteLine(Team[i]);
                }
        }
}
```

c) 3 marks :-

1.what is sealed class? Give example.

→ A sealed class, in C#, is a class that cannot be inherited by any class but can be instantiated.

→ The design intent of a sealed class is to indicate that the class is specialized and there is no need to extend it to provide any additional functionality through inheritance to override its behavior.

→ A sealed class is often used to encapsulate a logic that needs to be used across the program but without any alteration to it.

→ Example of Sealed Class:-

```
// Sealed class
sealed class SealedClass
{
    public int Add(int x, int y)
    {
        return x + y;
    }
}
class Class1
{
    static void Main(string[] args)
    {
            SealedClass sealedCls = new SealedClass();
            int total = sealedCls.Add(4, 5);
            Console.WriteLine("Total = " + total.ToString());
    }
}
```

2.what is delegate ? give example.

→ A delegate is a type that represents references to methods with a particular parameter list and return type.

→ When you instantiate a delegate, you can associate its instance with any method with a compatible signature and return type.

→ You can invoke (or call) the method through the delegate instance.

→ Delegates are used to pass methods as arguments to other methods.

→ Syntax:-

Delegate <return type> <delegate name> <parameter list>

Example of Delegate:-

```
public delegate void printString(string s);
```

```
printString ps1 =new printString(WriteToScreen);
printString ps2 = new printString(WriteToFile);
class DelegateDemo
{
    public delegate void addnum(int a, int b);
    public delegate int subnum(int a, int b);
    public void sum(int a, int b)
    {
        Console.WriteLine("Sum of 2 Value = " + (a + b));
    }
    public int subtract(int a, int b)
    {
        return a - b;
    }
    public static void Main(String[] args)
    {
        DelegateDemo obj = new DelegateDemo();
        addnum del_obj1 = new addnum(obj.sum);
        subnum del_obj2 = new subnum(obj.subtract);
        del_obj1(100, 40);
        //delegate addnum
        Console.WriteLine("Subtraction =" + del_obj2(100, 60));
        //delegate subnum
    }
}
```

d) 5 marks :-

1.explain collection? explain any one class of collection.

→ C# collection types are designed to store, manage and manipulate similar data more efficiently. Data manipulation includes adding, removing, finding, and inserting data in the collection. Collection types implement the following common functionality:

- Adding and inserting items to a collection
- Removing items from a collection
- Finding, sorting, searching items
- Replacing items
- Copy and clone collections and items
- Capacity and Count properties to find the capacity of the collection and number of items in the collection

→ Types of collections in C#:

- ArrayList
- HashTable
- SortedList
- Stack
- Queue

ArrayList:

- ArrayList class is a collection that can be used for any types or objects.
- Arraylist is a class that is similar to an array, but it can be used to store values of various types.
- An Arraylist doesn't have a specific size.
- Any number of elements can be stored.

→ Example of ArrayList:

```
ArrayList al = new ArrayList();
string str = "kiran teja jallepalli";
int x = 7;
```

```
float f = 10.5f;
al.Add(str);
al.Add(x);
al.Add(f);
foreach (object o in al)
{
    Console.WriteLine(o);
}
al.Remove(7);
```

2.what is event ? explain with example.

→ Events are user actions such as key press, clicks, mouse movements, etc., or some occurrence such as system generated notifications. Applications need to respond to events when they occur. For example, interrupts. Events are used for inter-process communication.

→ The events are declared and raised in a class and associated with the event handlers using delegates within the same class or some other class. The class containing the event is used to publish the event. This is called the publisher class. Some other class that accepts this event is called the subscriber class.

→ Syntax of Events:

<access modifier> event <delegate name>
<event name>;

→ Example of Events:

```
//This is Publisher Class
class AddTwoNumbers
{
        public delegate void dg_OddNumber();
                //Declared Delegate public event dg_OddNumber
        ev_OddNumber;
                //Declared Events public void Add()
                {
                        int result;
                        result = 5 + 4;
                        Console.WriteLine(result.ToString());
```

```
                //Check if result is odd number then raise event
                if (result % 2 != 0)
                {
                    ev_OddNumber();
                    //Raised Event
                }
            }
        }
        //This is Subscriber Class
        class EventDemo
        {
            static void Main(string[] args)
            {
                AddTwoNumbers a = new AddTwoNumbers();
                //Event gets binded with delegates a.ev_OddNumber +=
            new AddTwoNumbers.dg_OddNumber(EventMessage);
                a.Add();
            }
            //Delegates calls this method when event raised. static void
            EventMessage()
            {
                Console.WriteLine("*********Event Executed : This is Odd
            Number**********");
            }
    }
```

Q-3.

a) 1 marks :-

1 . F5 key from the keyboard is used to run - window application.

2.MDI stands for multiple document interface.

3. AutoSize property of the label is used to autometically resize to display its content.

4. The Image **properties is used to display image on the button.**

b) 2 marks :-

1.write a program to change the ,background color of the form with the help of ColorDialogBox.

```
private void button1_Click(object sender, EventArgs e)
{
        DialogResult result = colorDialog1.ShowDialog();
        if (result == DialogResult.OK)
        {
                label1.BackColor = colorDialog1.Color;
        }
}
```

2.write a two difference between panel and groupbox.

→ Panel is Scrollable whereas GroupBox isn't.

→ GroupBox has a caption, whereas a Panel doesn't.

c) 3 marks :-

1.difference between comboox and listbox.

Listbox	Combobox
Occupies more space but shows more than one value.	Occupies less space but shows only one value for visibility.
We can select multiple items.	Multiple select is not possible.
we can use checkboxes with in the list box.	can't use checkboxes within combo boxes.

d) 5 marks :-

1.explain messagebox class with all types of show method.

→ MessageBox is a built-in feature of Windows

→ A message box is a predefined window that lets you display a message.

→ We can also obtain simple responses from the user, such as Yes, No or OK.

→ In a window application, a message box is supported by the Message Box class.

→ We cannot create an object of that class to display a message box.

→ We have to call the static method .Show() to display message box.

→ **MessageBox buttons is enumeration that defines the following values:**

Ok	**AbortRetryIgnore**	**OKCancel**
YesNo	**RetryCancel**	**YesNoCancel**

→ **Return Value of show method :**

Abort	DialogResult.Abort	Cancel	DialogResult.Cancel
Ignore	DialogResult.Ignore	No	DialogResult.No
OK	DialogResult.OK	Retry	DialogResult.Retry
Yes	DialogResult.Yes		

1. **Abort** - The dialog box return value is Abort (usually sent from a button labeled Abort).
2. **Cancel** - The dialog box return value is Cancel (usually sent from a button labeled Cancel).
3. **Ignore** - The dialog box return value is Ignore (usually sent from a button labeled Ignore).

4. **No** - The dialog box return value is No (usually sent from a button labeled No).
5. **None** - Nothing is returned from the dialog box. This means that the modal dialog continues running.
6. **OK** - The dialog box return value is OK (usually sent from a button labeled OK).
7. **Retry** - The dialog box return value is Retry (usually sent from a button labeled Retry).
8. **Yes** - The dialog box return value is Yes (usually sent from a button labeled Yes).

Example:-

```
public partial class Form1 : Form
{
        public Form1()
        {
                InitializeComponent();
        }
        private void Form1_Load(object sender, EventArgs e)
        {
                DialogResult result = MessageBox.Show("Testing of MESSAGEBOX",
                "Message", MessageBoxButtons.OKCancel,
                MessageBoxIcon.Information);
                if (result != DialogResult.OK)
                {
                        Application.Exit();
                }
        }
}
```

2.what is dialogbox ? explain savefile dialogbox and openfile dialogbox.

→ A dialog box is a form, defined with particular properties.

→ Like a form, a dialog box is referred to as a container.

→ Like a form, a dialog box is mostly used to host child controls, insuring the role of dialog between the user and the machine.

→ A dialog box has following characteristics :

- There is only close button.
- It can‟t minimized, maximized, or restored
- It is usually modal, in which case the user is not allowed to continue any other operation on the same application until the doalogbox is dismissed

→ Uses of dialog box are :

- Display specific information to users.
- Gather information from users.
- Both display and gather information.

→ How to create and use dialog box

- Ready Designed Dialog ::
- Add New AboutBox1.Cs from project menu
  - Add New Item (Ctrl+Shift+A)
  - Select About Box (It will create Aboutbox1.Cs file)
  - Now add required coding to code window

→ OpenFile Dialog box:-

- To call the Open File dialog we can use
- OpenFileDialog1.ShowDialog();

→ USE of OpenFile Dialog (txtFile)

```
private void button1_Click(object sender, EventArgs e)
{
        DialogResult result = openFileDialog1.ShowDialog();
        if (result == DialogResult.OK)
        {
                System.IO.StreamReader sr = new
        System.IO.StreamReader(openFileDialog1 .FileName);
                textBox1.Text= sr.ReadToEnd();
```

```
                sr.Close();
            }
        }
```

→ SaveFile Dialog box

- To call the Save File dialog we can use
- SaveFileDialog1.ShowDialog();

→ Demonstrate use of SaveFile Dialog

```
private void button1_Click(object sender, EventArgs e)
{
    DialogResult result =saveFileDialog1.ShowDialog();
    if(result == DialogResult.OK)
    {
        string name = saveFileDialog1.FileName;
        File.WriteAllText(name, textBox1.Text);

    }

}
```

Required -> using System;

using System.IO;

using System.Text;

Q-4

a) 1 marks :-

1.ADO stands for Activex Data Object.

2.data reader is a disconnected architecture component. True

3.dataset is a virtual set or localset of record set. False

4.OLEDB stands for Object Linking and Embedding, Database.

b) 2 marks :-

1.explain command object.

→ After establishing connection with the database, we can use the command object for processing request in the form of command and returning the results of those requests from the database.

The command object is one of the basic components of ADO .NET.

→ The Command Object uses the connection object to execute SQL queries.

→ The queries can be in the Form of Inline text, Stored Procedures or direct Table access.

→ An important feature of Command object is that it can be used to execute queries and Stored Procedures with Parameters.

→ If a select query is issued, the result set it returns is usually stored in either a DataSet or a DataReader object.

2.shortnote on data gridview.

→ The command object is one of the basic components of ADO .NET.

→ The Command Object uses the connection object to execute SQL queries.

→ The queries can be in the Form of Inline text, Stored Procedures or direct Table access.

→ An important feature of Command object is that it can be used to execute queries and Stored Procedures with Parameters.

→ If a select query is issued, the result set it returns is usually stored in either a DataSet or a DataReader object.

→ You can extend the DataGridView control in a number of ways to build custom behaviors into your applications.

For example, you can programmatically specify your own sorting algorithms, and you can create your own types of cells.

→ We can easily customize the appearance of the DataGridView control by choosing among several properties. ν Many types of data stores can be used

as a data source, or the DataGridView control can operate with no data source bound to it.

→ We have manipulated DataGrid with either DataSet or DataReader. ν In example of DataReader we have added individual row in to the object of DataGridView. ν But in example of DataSet we have added a whole table using DataSource property of DataGridView.

c) 3 marks :-

1. Difference between executeNonQuery() and execute scaler.

ExecuteScalar Method:
Namespace: System.Data.SqlClient
Return DataType : System.Object (returns a single value)
Use : Executes the query, and returns the first column of the first row in the result set returned by the query. Additional columns or rows are ignored.
Example:

```
public int getSomeProdId()
{
      int count=0;
      SqlConnection conn = new SqlConnection(connString))
      String sqlQuery = "SELECT COUNT(*) FROM dbo.region";
      SqlCommand cmd = new SqlCommand(sqlQuery, conn);
      try
      {
         conn.Open();
         //Since return type is System.Object, a typecast is must
         count = (Int32)cmd.ExecuteScalar();
      }
      catch (Exception ex)
      {
         Console.WriteLine(ex.Message);
      }
       finally
       {
         conn.Close();
```

```
        }
        return count;
}
```

ExecuteNonQuery Method:

Namespace: System.Data.SqlClient

Return DataType: System.Int32 (number of rows affected)

Use : You can use the ExecuteNonQuery to perform catalog operations by executing UPDATE, INSERT, or DELETE statements.

For UPDATE, INSERT, and DELETE statements, the return value is the number of rows affected by the command. When a trigger exists on a table being inserted or updated, the return value includes the number of rows affected by both the insert or update operation and the number of rows affected by the trigger or triggers. For all other types of statements, the return value is -1.

Example:

```
public void UpdateEmployeeEmail()
{
      SqlConnection conn = new SqlConnection(connString))
      String sqlQuery = "UPDATE Employee SET
empemail='suresh.paldia@xyz.com' WHERE empid=4;
      SqlCommand cmd = new SqlCommand(sqlQuery, conn);
      try
      {
         conn.Open();
         cmd.ExecuteNonQuery();
      }
      catch (Exception ex)
      {
         Console.WriteLine(ex.Message);
      }
       finally
       {
         conn.Close();
       }
       return count;
}
```

2. What is datatable? explain.

→ C# DataTable is a central object. It represents the database tables that provide a collection of rows and columns in grid form.

→ There are different ways to create rows and columns in the DataTable. In this article, we will be discussing how to add data to DataTable, bind DataTable to the DataGridView using data binding, etc.

→ Other objects that use DataTable include DataSet and DataView. Whenever we want to access the "C# datatable" objects, we must remember they are case-sensitive. For example, if there are two datatables, "NewDataTable" and "newDataTable", the string you will use for searching the data tables must be case sensitive.

→ So basically, "C# DataTable" is an in-memory tabular data representing the tabular cache of constraints, rows, and columns.

→ Creating a DataTable

→ We first need to create an instance of a "DataTable class" for creating a data table in "C# DataTable".

→ Then we will add DataColumn objects that define the type of data we will insert.

→ And then add DataRow objects which contain the data.

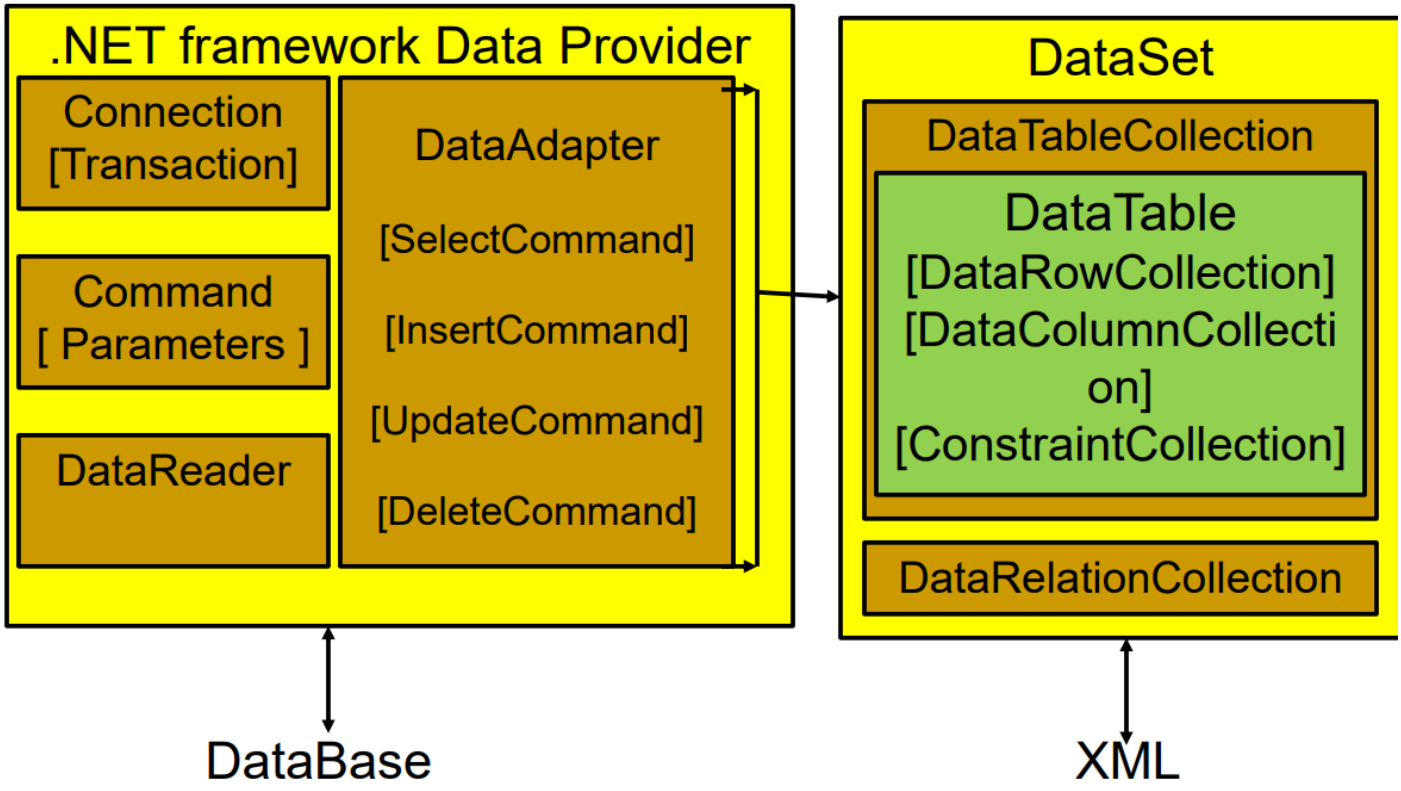
d) 5 marks :-

1. Explain the architecture of ADO.NET.

→ ADO.NET architecture provides two main components to enable data access and manipulate.

→ The components are...

- .Net Data Provider
- DataSet



Connection:

The connection object is the first important component of your application. It is required to connect to a backend database that may be SQL Server, Oracle, MySQL, etc. You must have two things to create a connection object. Your database Machine name or IP address or someplace where it is stored is where it is. The second thing is security credentials, such as whether it's a Windows authentication or username and password-based authentication. The connection is created using the connection object and a backend data source must be connected to using the connection.

Command:

The second important component is the command object. When we discuss databases such as SQL Server, Oracle, MySQL, then speak SQL, it is the command object that we use to create SQL queries. After you create your SQL queries, you can execute them over the connection using the command object. You can go either the DataSet or the DataReader way with DTS. In general, you should choose which

method you require based on the situation. Note: You can go either the DataSet or the DataReader way with DTS.

DataReader:

We can only read the records in the forward mode with DataReader. Here, you should familiarize yourself with three things: read-only, connected, and forward modes.

DataSet:

A disconnected recordset can be browsed in both directions, and it is also possible to insert, update, or delete data sets. The DataAdapter fills a DataSet using data.

DataAdapter:

The DataAdapter performs an operation on the data from the command object and then writes the data set to the dataset.

DataView Class:

A DataView enables you to modify the appearance of the data stored in a DataTable, a data-binding skill that is frequently employed in data-view applications. You may alter the sort order of data in a table or filter it based on row state or on a filter expression using a DataView.

Q-5

a) 1 marks :-

1.LINQ stands for Language-Integrated Querybase.

2.dropdown is a type of crystal report.

→ true

3. formula **field of crystal report is used to create formula.**

4. CrystalReportViewer **control is used to view the cristal report.**

b) 2 marks :-

1.What is user control ? write step to create user control.

→ A custom control is user defined control. ¬ It is created by the user as per requirement. ¬ User control is derived from the UserControl class or an existing windows forms control. ¬ User control provides front-end functionality through its graphical interface, but provides back-end functionality through its methods, properties and events.

→ Creating User Control

- Step 1:
  Right Click on Project Name
- Step 2:

  Click on add new item
- Step 3:

  elect type User control
- Step 4

  Design Control according to usage

2.write the names of any four special fields of crystal report.

1. File Author – or the person who created the report.
2. File Creation Date – or the date the report is created.
3. Report Title – or the title of the report.
4. Report Comments – any comments added to the report.

c) 3 marks :-

1.explain types of setup project.

Project type	When to use
Setup Project	For applications that will be installed on the end user's computer.
Web Setup Project	For applications that will be installed on or deployed through a web server.
Merge Module Project	For components that will be used by other applications. (Merge modules can be imported into either normal applications or web applications.)
Cab Project	For legacy component installation through a web browser. (Typically used for ActiveX controls.)
Setup Wizard	To create one of the four other project types, according to the selection made in the wizard.

2.explain types of crystal report.

1) Using report wizard

2) As a Blank Report

3) From an Existing Report

1. Using report wizard:-

1) Standard :- Used to creation of typical report

2) Cross-Tab :- Guides the creation of a report that contains a summarized grid. Generally used when need to create report like a row a column type or table type report.

3) Mail Label :- Guides the creation a report with multipalcolumns.Generally used to create a sheet of address labels.

2) As a Blank Report :-

When we want to create blank report then we can select this type of report.

3) From an Existing Report

Used when we create a report from the existing report.]

d) 5 marks :-

1.write a steps to create a crystal report.

→ Step 1 :-

Create a table in the database. Create an Employee table in the database.

→ Step 2 :-

Go to Visual Studio.

→ Step 3 :-

Go to the Solution Explorer and right-click on your project name and seelct Add -> New Item.

→ Step 4 :-

Add New Item-> Crystal Report.

→ Step 5 ;-

Click the Ok Button.

→ Step 6 ;-

Choose the data source

→ Step 7 ;-

Select OLE DB (ADO) from Create New Connection

→ Step 8 :-

Select Microsoft OLE DB Provider for SQL Server

2.explain all sections of crystal report.

1. **Report Header**

   This section is typically used for the report title and other information you want to appear at the beginning of the report. It can also be used for charts and cross-tabs that include data for the entire report.

2. **Page Header**

   This section is typically used for information that you want to appear at the top of each page. This can include such things as chapter names, the name of the document, and other similar information. This section can also be used to display field titles above the fields on a report.

3. **Details**

   This section is used for the body of the report, and is printed once per record. The bulk of the report data typically appears in this section.

4. **Report Footer**

   This section is used for information you want to appear only once at the end of the report (such as grand totals) and for charts and cross-tabs that include data for the entire report.

5. **Page Footer**

   This section typically contains the page number and any other information you want to appear on the bottom of each page.