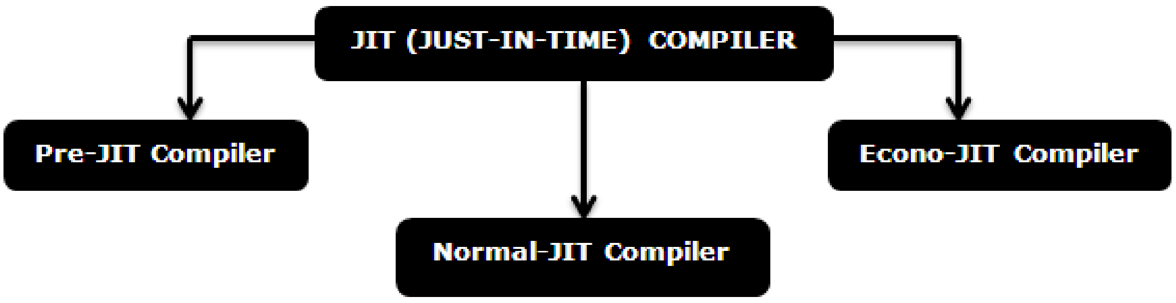
Programming with C Sharp (Assignment)
BCA SEMESTER -4

(1) Write a note on JIT and its types.

Before MSIL(MS Intermediate Language) can be executed, it must converted by .net Framework Just in time (JIT) compiler to native code, which is CPU specific code that run on some computer architecture as the JIT compiler. Rather than using time and memory to convert all the MSIL in portable executable (PE) file to native code, it converts the MSIL as it is needed during execution and stored in resulting native code so it is accessible for subsequent calls.

In Microsoft .NET there are three types of JIT (Just-In-Time) compilers which are Explained as Under:

- Pre-JIT Compiler (Compiles entire code into native code completely)
- Econo JIT Compiler (Compiles code part by part freeing when required)
- Normal JIT Compiler (Compiles only that part of code when called and places in cache



Description:

- **Pre-JIT COMPILER** Pre-JIT compiles complete source code into native code in a single compilation cycle. This is done at the time of deployment of the application.

- **Econo-JIT COMPILER:** Econo-JIT compiles only those methods that are called at runtime. However, these compiled methods are removed when they are not required.

- **Normal-JIT COMPILER:** Normal-JIT compiles only those methods that are called at runtime. These methods are compiled the first time they are called, and then they are stored in cache. When the same methods are called again, the compiled code from cache is used for execution. These methods are compiled the first time they are called, and then they are stored in cache. When the same methods are called again, the compiled code from cache is used for execution

Q 2 What is 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
- **THE ARRAYLIST:**
- ArrayList is one of the most flexible data structure from CSharp Collections. ArrayList contains a simple list of values. ArrayList implements the IList interface using an array and very easily we can add, insert, delete, view etc. It is very flexible because we can add without any size information, which is it will grow dynamically and also shrink.
- ***Add : Add an Item in an ArrayList***
- ***Insert : Insert an Item in a specified position in an ArrayList***
- ***Remove : Remove an Item from ArrayList***
- ***RemoveAt: remove an item from a specified position***
- ***Sort : Sort Items in an ArrayList***
- <u>How to add an Item in an ArrayList ?</u>
- ***Syntax : ArrayList.add(object)***
- ***object : The Item to be add the ArrayList***
- ***ArrayList arr;***
- ***arr.Add("Item1");***
- <u>How to Insert an Item in an ArrayList ?</u>
- ***Syntax : ArrayList.insert(index,object)***
- ***index : The position of the item in an ArrayList***
- ***object : The Item to be add the ArrayList***
- ***ArrayList arr;***
- ***arr.Insert(3, "Item3");***
- <u>How to remove an item from arrayList ?</u>
- ***Syntax : ArrayList.Remove(object)***
- ***object : The Item to be add the ArrayList***
- ***arr.Remove("item2")***
- <u>How to remove an item in a specified position from an ArrayList ?</u>
- ***Syntax : ArrayList.RemoveAt(index)***
- ***index : the position of an item to remove from an ArrayList***
- ***ItemList.RemoveAt(2)***
- <u>How to sort ArrayList ?</u>
- ***Syntax : ArrayList.Sort()***
- The System.Collections.ArrayList class is similar to arrays, but can store elements of any data type. We don’t need to specify the size of the collection when using an ArrayList (as we used to do in the case of simple arrays). The size of the ArrayList grows dynamically as the number of elements it contains changes. An ArrayList uses an array internally and initializes its size with a default value called Capacity. As the number of elements increase or decrease, ArrayList adjusts the capacity of the array accordingly by making a new array and copying the old values into it. The Size of the ArrayList is the total number of elements that are actually present in it while the Capacity is the number of elements the ArrayList can hold without instantiating a new array. An ArrayList can be constructed
- Like this:

- ***ArrayList list = new ArrayList();***
- We can also specify the initial Capacity of the ArrayList by passing an integer value to the constructor:
- ***ArrayList list = new ArrayList(20);***
- **PROGRAM:**

```
static void Main()
{
         ArrayList list = new ArrayList();
         list.Add(45);
         list.Add(87);
         list.Add(12);
         foreach(int num in list)
        {
           Console.Write(num);
         }
}
```

```
Output :45 87 12
```

Q 3 Explain MessageBox Class with all types of show() Method.

A message box or dialog box is used to interact with the users of your application. The purpose of using a message box may include notifying about a particular action e.g. success message after entering a record. Similarly, an error message if an operation was unsuccessful. In both cases, you may display the "OK" button in the message box with the message.

In other cases, you may display a message box for the user confirmation before performing a critical action e.g. deleting a record permanently. In that case, a Yes/No button in the dialog box makes sense.

Syntax	Use
MessageBox.Show(String)	It will display only the message box with the string that is passed. An ok button is also present to close the dialog. Example:`Messagebox.Show("Test")`
MessageBox.Show(String, String)	It will display only the message box with the string that is passed as first parameter. The second parameter is the title of the Message Box. An ok button is also present to close the dialog. Example:`MessageBox.Show( "Message", "Title").`
MessageBox.Show(String,String, MessageBoxButtons)	It will display the message box with the supplied text, title and the corresponding buttons to be displayed on the Message Box. For eg the below will display Yes and No buttons. `MessageBox.Show( "Message", "Title", MessageBoxButtons.YesNo);`

Show(String, String, MessageBoxButtons, MessageBoxIcon)	It will display the message box with the supplied text, title and the corresponding buttons to be displayed on the Message Box. It will also display the icon that is specified before the text.
	For eg the below will display Yes and No buttons with a question mark in front of message.
	`MessageBox.Show( "Message”, "Title",`
	`MessageBoxButtons.YesNo,`
	`MessageBoxIcon.Question);`

Show(String, String, MessageBoxButtons, MessageBoxIcon, MessageBoxDefaulButton)	It will display the message box with the supplied text, title and the corresponding buttons to be displayed on the Message Box. It will also display the icon that is specified before the text. The last parameter denotes which button must be selected by default on load.
	For eg the below will display Yes and No buttons with a question mark in front of message.
	`MessageBox.Show( "Message”, "Title",`
	`MessageBoxButtons.YesNo,`
	`MessageBoxIcon.Question,`
	`MessageBoxDefaultButton.Button2);`

Show(String, String, MessageBoxButtons, MessageBoxIcon, MessageBoxDefaulButton, MessageBoxOptions)	It will display the message box with the supplied text, title, and the corresponding buttons to be displayed on the Message Box. It will also display the icon that is specified before the text. The last parameter denotes which button must be selected by default on load and the contents of the messagebox will be right-aligned.
	For eg the below will display Yes and No buttons with a question mark in front of message.
	`MessageBox.Show( "Message”, "Title",`
	`MessageBoxButtons.YesNo,`
	`MessageBoxIcon.Question,`
	`MessageBoxDefaultButton.Button2,`
	`MesageBoxOptions.RightAlign, true);`

Q 4 Explain the Architecture of ADO.Net.

ADO.NET uses a multilayer architecture that has components such as the *Connection, Reader, Command, Adapter* and *DataSet* objects. ADO.NET introduced *data providers* that are a set of special classes to access a specific database, execute SQL commands and retrieve data. Data providers are extensible; developers can create their own providers for proprietary data source Some examples of data providers include SQL server providers, OLE DB and Oracle providers.

ADO.NET provides two types of classes of objects:

- **Connection-based**: They are the data provider objects such as Connection, Command, DataAdapter and DataReader. They execute SQL statements and connect to a database.
- **Content-based**: They are found in the *System.Data* namespace and includes DataSet, DataColumn, DataRow, and DataRelation. They are completely independent of the type of data source.

- **ADO.NET Namespaces**

Namespaces	Description
System.Data	Contains the definition for columns, relations, tables, database, rows, views and constraints.
System.Data.SqlClient	Contains the classes that are used to connect to a MS SQL server database such *as SqlCommand, SqlConnection,* and *SqlDataAdapter*.
System.Data.Odbc	Contains classes required to connect to most ODBC drivers. These classes include *OdbcCommand* and *OdbcConnection*.
System.Data.OracleClient	Contains classes such as *OracleConnection* and *OracleCommand* required to connect to an Oracle database.

Connection Class

You need to establish a connection class object to insert, update, delete or retrieve data from a database. The *Connection* class allows you to establish a connection to the data source, but to do so, it needs the necessary information to discover the data source, which is provided by a connection string.

Connection Strings

You need to supply a *connection string* in the *Connection* class object. The connection string is a series of name/value settings separated by semicolons (;). A connection string contains information such as the location of the database, its name and its authentication mechanism.
This connection is used to connect to the Master database on the current computer using integrated security, indicated by a currently logged-in Windows user to access the database.

Q 5 Explain All Section of Crystal Reports.

When creating a report, Crystal Reports automatically creates five areas in the Design tab:

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.