.NET PROGRAMMING

Q.1] Explain the .NET architechure in detail?

- Net Architecture is a software architecture used for building applications that run on Microsoft's .NET platform.

- It provides a set of libraries and tools that simplify the development of complex applications by providing a consistent programming model and a standardized set of APIs.

Key Components of .NET Architecture:

There are four essential parts to the.NET architecture:

1] Common Language Runtime (CLR):

- The.NET applications are operated by the CLR, which serves as the execution engine.

- It offers security, thread management, and Memory Management functions.

- It also provides a common type system, which allows objects written in different languages to interoperate seamlessly.

2] Base Class Library (BCL):

- The **BCL** is a set of reusable classes, interfaces, and value types that provide core functionality for .NET applications.

- It includes classes for input/output, collections, threading, networking, and more.

3] Just-In-Time Compiler (JIT):

-The **JIT** compiler is in charge of translating.

- NET applications' **Intermediate Language (IL)** code into native machine code that the CPU can execute.

- In order to increase performance, it compiles code as it is required.

4] Visual Studio:

- Visual Studio is a very popular **Integrated Development Environment (IDE)** that provides a set of tools for building .NET applications.

- It includes a code editor, debugger, project management tools, and more.

The.NET Architecture's Benefits:

1] Interoperability:

- C#, **VB.NET**, and F# are just a few of the programming languages that.NET supports.

- This frees developers from having to worry about compatibility difficulties when selecting the language that best meets their needs.

2] Consistency:

- The.NET framework offers a defined set of APIs and a consistent programming style, which makes it simpler to create and manage complex systems.

3] Rapid Application Development (RAD):

- With the.NET framework's tools, such as Visual Studio, developers may quickly and effectively create apps.

- This shortens the process and lowers expenses.

4] Memory Management:

- The **CLR** provides automatic memory management, which helps to prevent common memory-related errors, such as buffer overflows and memory leaks.

5] Security:

- .NET provides a comprehensive security model, which includes code access security, role-based security, and encryption.

- This helps to protect applications from unauthorized access and attacks.

Disadvantages of .NET Architecture:

1] Performance:

- .NET applications can be slower than native applications because of the overhead of the **CLR** and the **JIT** compiler.

- However, this performance penalty is usually small and is offset by the benefits of automatic memory management and other features.

2] Platform Dependency:

- .NET applications can only run on systems that have the .NET framework installed.

- This can be a disadvantage if the target platform does not support .NET or if the user does not have the necessary permissions to install the framework.

Q.2] Write a note on CLR?

- .NET CLR is a runtime environment that manages and executes the code written in any .NET programming language.

- CLR is the virtual machine component of the .NET framework.

- That language's compiler compiles the source code of applications developed using .NET compliant languages into CLR's intermediate language called MSIL, i.e., Microsoft intermediate language code.

- This code is platform-independent.

- It is comparable to byte code in java.

- Metadata is also generated during compilation and MSIL code and stored in a file known as the Manifest file.

- This metadata is generally about members and types required by CLR to execute MSIL code.

- A justin-time compiler component of CLR converts MSIL code into native code of the machine. This code is platform-dependent.

- CLR manages memory, threads, exceptions, code execution, code safety, verification, and compilation.

Functions of .NET CLR

- Exception handling
- Type safety
- Security
- Improved performance
- Language independency

Components of .NET CLR

- Class Loader - Used to load all classes at run time.
- MSIL to Native code - The Just In Time (JTI) compiler will convert MSIL code into native code.
- Code Manager - It manages the code at run time.

Q.3] What is mean by JIT? Explain its types?

- JIT stands for just-in-time compiler.

- Compilers are tools that translate source code to machine understandable language.

- Just-In-Time compiler(JIT) is a part of Common Language Runtime (CLR) in .NET which is responsible for managing the execution of .NET programs regardless of any .NET programming language.

- It converts the MSIL code to CPU native code as it is needed during code execution.

- It is called just-in-time since it converts the MSIL code to CPU native code;

- when it is required within code execution otherwise it will not do anything with that MSIL code.

- A language-specific compiler converts the source code to the intermediate language.

There are 3 types of JIT compilers which are as follows:

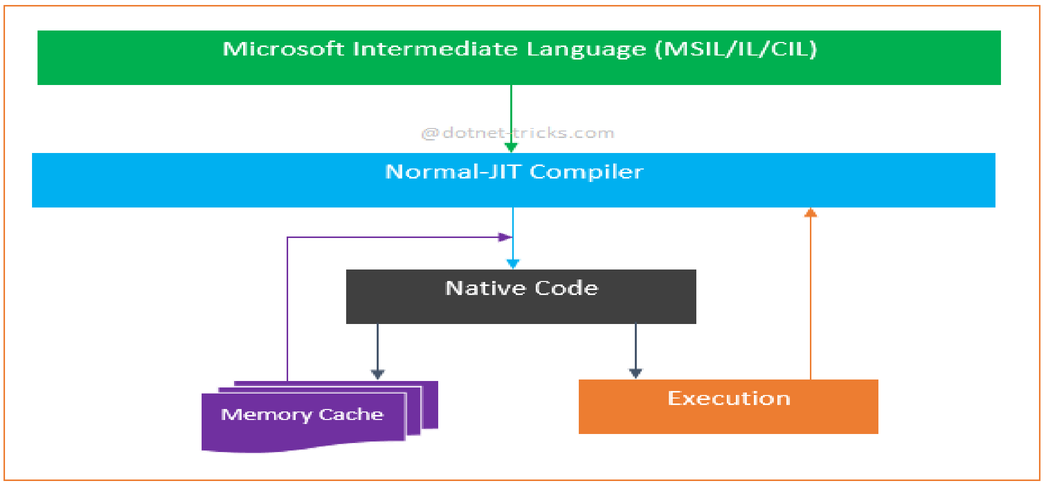
1] Normal JIT

- This complies only those methods that are called at runtime.

- These methods are compiled only first time when they are called, and then they are stored in memory cache.

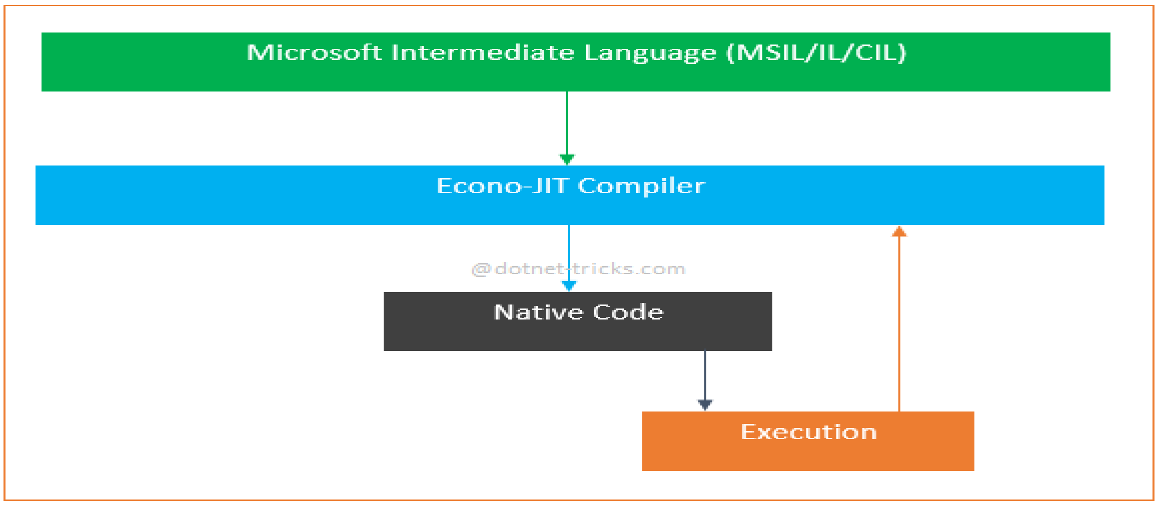
- This memory cache is commonly called as JITTED.

- When the same methods are called again, the complied code from cache is used for execution.



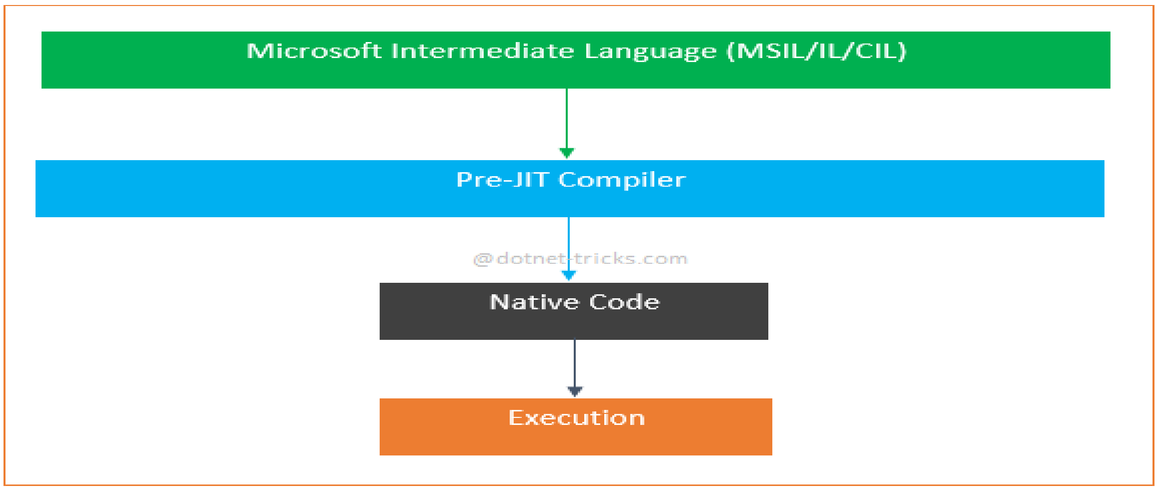
2] Econo JIT

- This complies only those methods that are called at runtime and removes them from memory after execution.



3] Pre JIT

- This complies entire MSIL code into native code in a single compilation cycle. This is done



Q.4] Write a note on managed code & unmanaged code?

A] managed code -

- A code which is written to aimed to get the services of the managed runtime environment execution like CLR(Common Language Runtime) in .NET Framework is known as Managed Code.

- It always implemented by the managed runtime environment instead of directly executed by the operating system.

- The managed runtime environment provides different types of services like garbage collection, type checking, exception handling, bounds checking, etc.

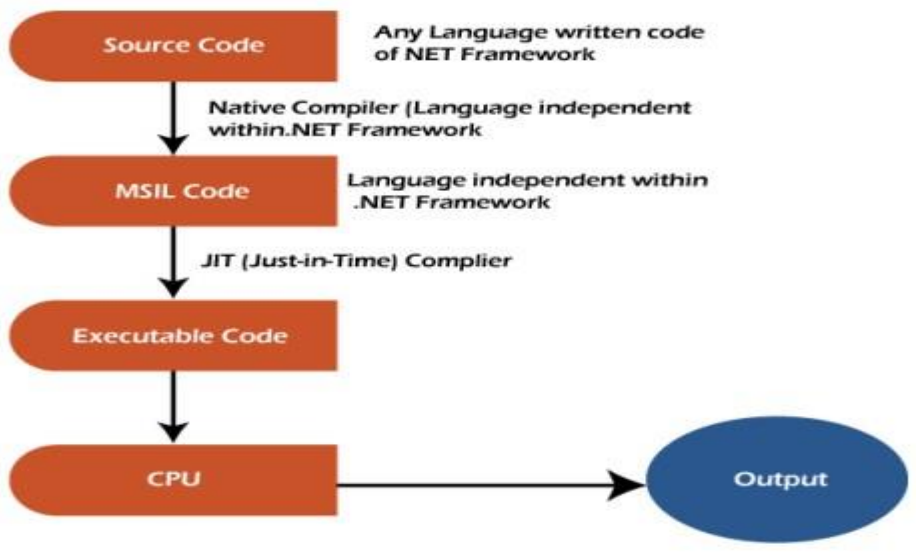
- It also provides memory allocation, type safety, etc to the code.

- The application is written in the languages like Java, C#, VB.Net, etc. are always aimed at runtime environment services to manage the execution and the code written in these types of languages are known as managed code.

Advantages Managed Code -

• It implement the garbage collection automatically.

• It also provides runtime type checking/dynamic type checking.


Source Code
Any Language written code of NET Framework
Native Compiler (Language independent within.NET Framework
MSIL Code
Language independent within .NET Framework
JIT (Just-in-Time) Complier
Executable Code
CPU
Output

Managed Code Execution Process

B] Unmanaged Code –

- Applications that do not run under the control of the CLR are said to be unmanaged.

- A code which is directly executed by the operating system is known as Unmanaged code.

- It always aimed for the processor architecture and depends upon computer architecture.

- It always compiles to the native code that is specific to the architecture.

- In unmanaged code, the memory allocation, type safety, security, etc are managed by the developer.

Advantages of Unmanaged Code -

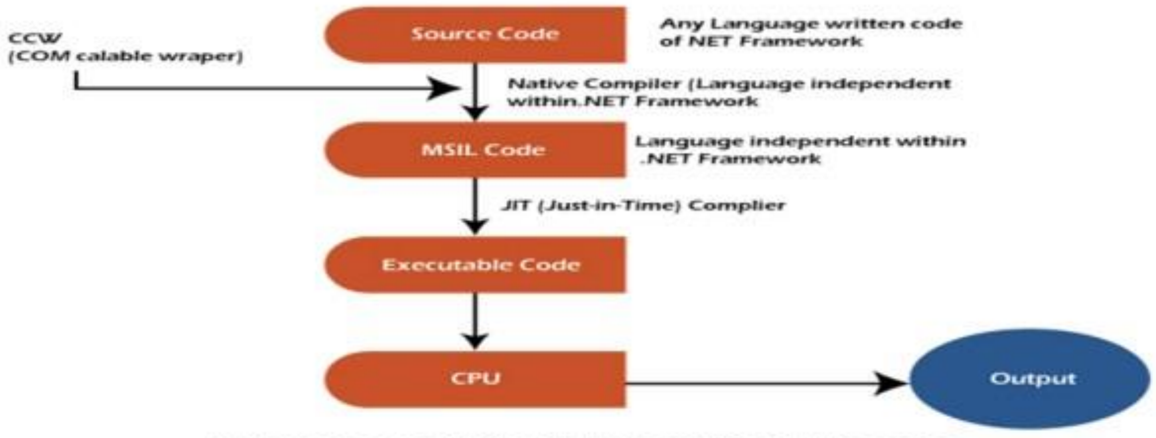
• It provides the low-level access to the programmer.

• It also provides direct access to the hardware.

Disadvantages of Unmanaged Code -

• It does not provide security to the application.

• Error and exceptions are also handled by the programmer.


CCW
(COM calable wraper)
Source Code
Any Language written code of NET Framework
Native Compiler (Language independent within.NET Framework
MSIL Code
Language independent within .NET Framework
JIT (Just-in-Time) Complier
Executable Code
CPU
Output

Unmanaged Code Execution Process

Q.5] What is mean by type casting? Discuss the Implicit & Explicit?

- When the variable of one data type is changed to another data type is known as the Type Casting.

- According to our needs, we can change the type of data.

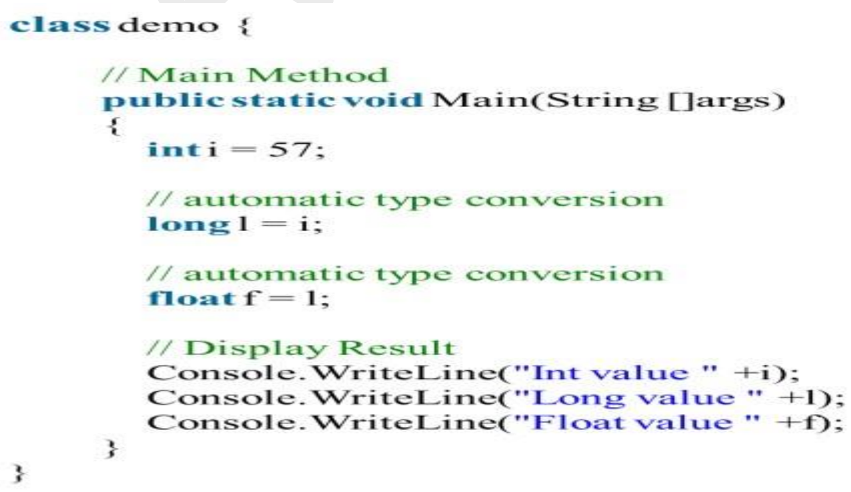
A] Implicit type conversion –

- For the implicit conversion, there is not any need for the special syntax.

- This type of conversion is safe; in this conversion, there is not any loss of the data.

- Implicit conversions include the conversion of the small type to large integral types, and from the derived class to the base class conversion.

- converting a smaller type to a larger type size char -> int -> long -> float -> double



```
class demo {

        // Main Method
        public static void Main(String []args)
        {
            int i = 57;

            // automatic type conversion
            long l = i;

            // automatic type conversion
            float f = l;

            // Display Result
            Console.WriteLine("Int value " +i);
            Console.WriteLine("Long value " +l);
            Console.WriteLine("Float value " +f);
        }
}
```

B] Explicit type conversion –

- These conversions are done explicitly by users using the pre-defined functions.

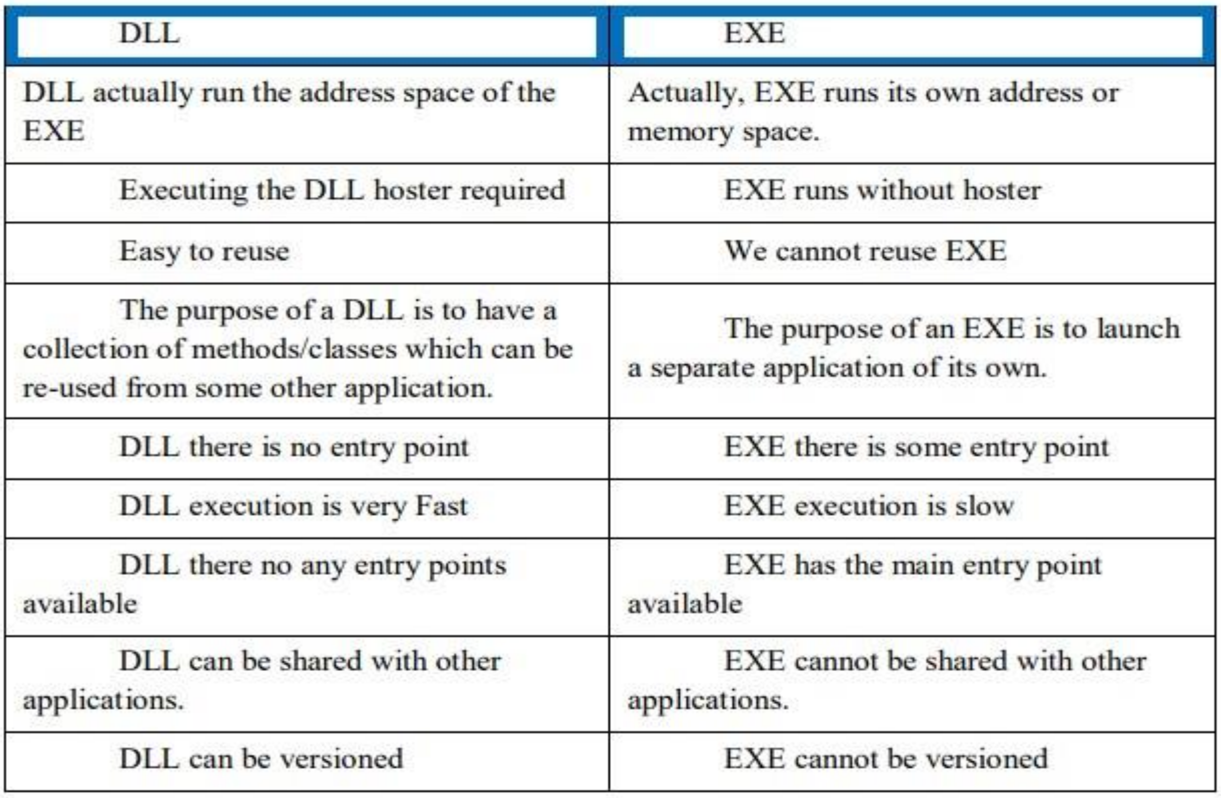
- Explicit conversions require a cast operator.

- We will do the casting when there is the situation of the data loss, or when the conversion is not succeeded.

- converting a larger type to a smaller size type double -> float -> long -> int -> char

```
using System;

class Program {

        static void Main(string[] args) {

                double myDouble = 9.78;

                int myInt = (int) myDouble; // Manual casting: double to int

                Console.WriteLine(myDouble);

                Console.WriteLine(myInt);

        }

}
```

Q.6] Write a note on DLL & EXE?



DLL	EXE
DLL actually run the address space of the EXE	Actually, EXE runs its own address or memory space.
Executing the DLL hoster required	EXE runs without hoster
Easy to reuse	We cannot reuse EXE
The purpose of a DLL is to have a collection of methods/classes which can be re-used from some other application.	The purpose of an EXE is to launch a separate application of its own.
DLL there is no entry point	EXE there is some entry point
DLL execution is very Fast	EXE execution is slow
DLL there no any entry points available	EXE has the main entry point available
DLL can be shared with other applications.	EXE cannot be shared with other applications.
DLL can be versioned	EXE cannot be versioned

Q.7] Write a note on Boxing & Unboxing?

A] Boxing -

- The process of converting a Value Type variable (char, int etc.) to a Reference Type variable (object) is called Boxing.

• Boxing is an implicit conversion process in which object type (super type) is used.

• Value Type variables are always stored in Stack memory, while Reference Type variables are stored in Heap memory.

```
using System;

class Boxing {

        static public void Main() {

                int num = 2020; // boxing

                object obj = num; // value of num to be change

                num = 100;
        System.Console.WriteLine("Value - type value of num is : {0}",num);

        System.Console.WriteLine("Object - type value of obj is : {0}", obj);

    }

}
```

B] Unboxing –

- Unboxing is the reverse of boxing, i.e., converting a reference type into a value type.

- This is an explicit conversion and requires type casting.

- The boxed object can be unboxed, and the value can be assigned to a value type.

```
using System;

class unboxing {

        static public void Main() {

                int num = 23; // boxing

                object obj = num; // unboxing

                int i = (int)obj;

                // Display result

                Console.WriteLine("Value of ob object is : " + obj);
                Console.WriteLine("Value of i is : " + i);

        }

}
```

Q.8] Explain the Abstraction & Encapsulation in C#?

A] Abstraction

- The process of representing the essential features without including the background details is called Abstraction.

- In simple words, we can say that it is a process of defining a class by providing the necessary details to call the object operations (i.e., methods) by hiding its implementation details. It is called abstraction in C#.

- It is the process of gaining information.

- The problems in this technique are solved at the interface level.

- It helps hide the unwanted details/information.

- It can be implemented using abstract classes and interfaces.

Advantages of abstraction –

• Privacy of data is maintained since only relevant data is visible to the user.

• Reduces the complexity of code and increases readability.

B] Encapsulation -

- Encapsulation is defined as the wrapping up of data and information under a single unit.

- It is the mechanism that binds together the data and the functions that manipulate them.

- Problems in encapsulation are solved at the implementation level.

- It helps hide data using a single entity, or using a unit with the help of method that helps protect the information.

- It can be implemented using access modifiers like public, private and protected.

- As in encapsulation, the data in a class is hidden from other classes, so it is also known as data-hiding.

Advantages of Encapsulation –

• Data Hiding –

-The user will have no idea about the inner implementation of the class.

- It will not be visible to the user that how the class is stored values in the variables.

- He only knows that we are passing the values to accessors and variables are getting initialized to that value.

• Reusability –

-Encapsulation also improves the re-usability and easy to change with new requirements.

• **Testing code is easy –**

-Encapsulated code is easy to test for unit testing.

Q.9] Explain the inheritance concept with its types?

- In C#, inheritance allows us to create a new class from an existing class.

- It is a key feature of Object-Oriented Programming (OOP).

- The class from which a new class is created is known as the base class (parent or superclass).

- And, the new class is called derived class (child or subclass) The derived class inherits the fields and methods of the base class.

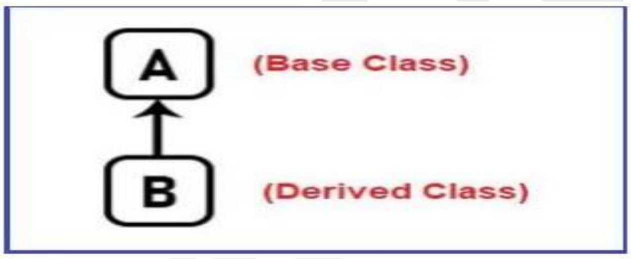
Types of Inheritance -

1] Single Inheritance –

- When a class is inherited from a single base class then the inheritance is called single inheritance.

- We have a class called A that is the Parent class and another class called B that is the Child class, and class B is inheriting from class A. I.e. Class B has a single Parent class i.e. class A.

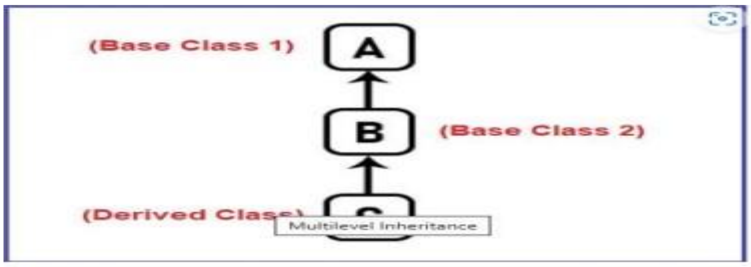
- This type of inheritance is called Single Inheritance.



2] Multilevel Inheritance –

- When a derived class is created from another derived class, then such a type of inheritance is called Multilevel Inheritance.

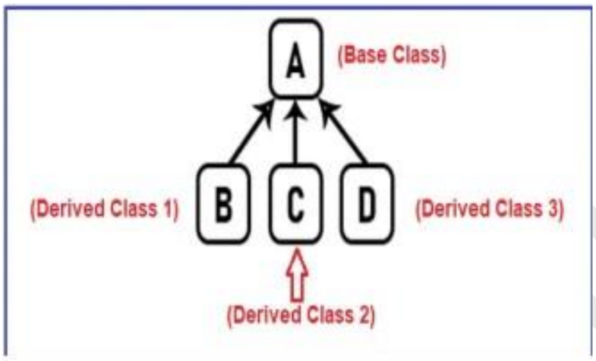
- For a better understanding, please have a look at the below image. If there is a class called A and from class A, class B is inheriting and from class B, class C is inheriting, then such type of inheritance is called Multilevel Inheritance.



3] Hierarchical Inheritance –

- When more than one derived class is created from a single base class then it is called Hierarchical inheritance.

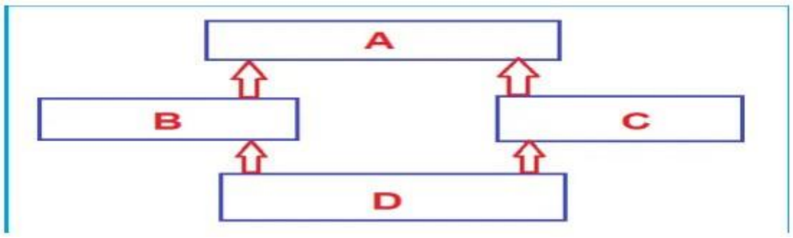
- Now, if you have a class called A and from this class A, if more than one class inheriting i.e. class B is inheriting, class C is inheriting as well as class D is inheriting i.e. when more than one child class is inheriting from a Single Base Class, then such a type of inheritance is called Hierarchical Inheritance.



4] Hybrid Inheritance –

- Hybrid Inheritance is the inheritance that is the combination of any Single, Hierarchical, and Multilevel inheritances.

- There are two subclasses i.e. B and C which are inheriting from class A (this is Hierarchical inheritance).



5] Multiple Inheritance –

- C# does not support multiple inheritances of classes; the same thing can be done using interfaces.

- Private members are not accessed in a derived class when one class is derived from another.

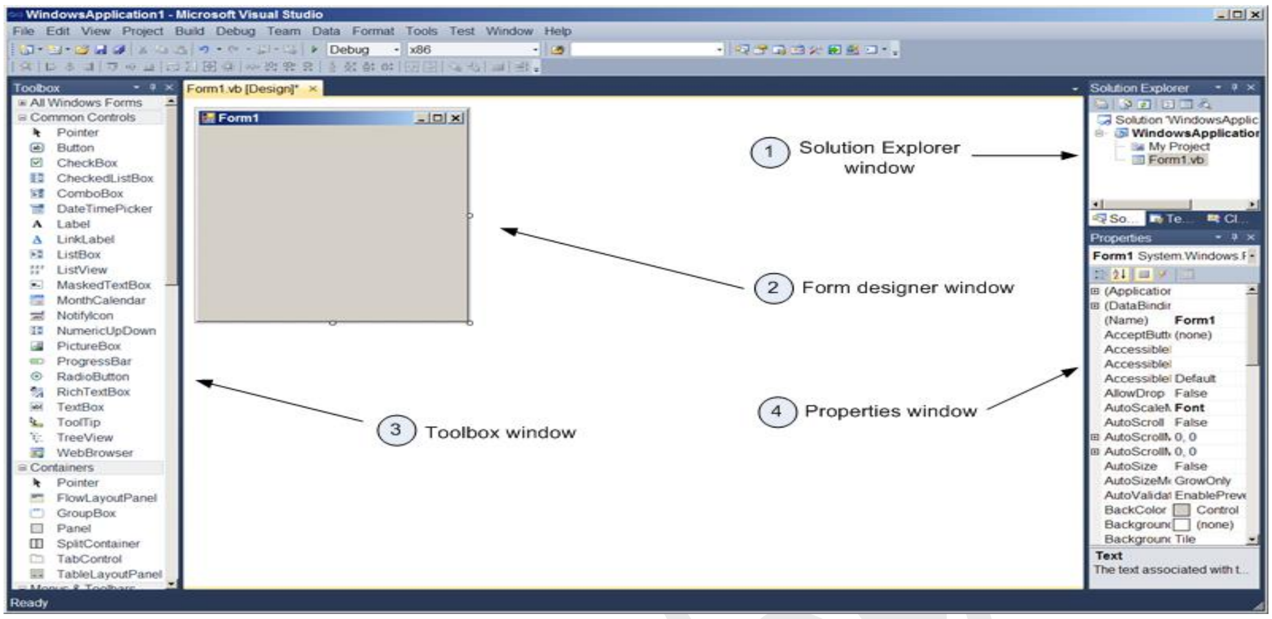
Q.10] Write a note on .NET IDE?

- Visual Studio is a powerful developer tool that you can use to complete the entire development cycle in one place.
- It is a comprehensive integrated development environment (IDE) that you can use to write, edit, debug, and build code, and then deploy your app.
- An *integrated development environment* (IDE) is a feature-rich program that supports many aspects of software development.
- The Visual Studio IDE is a creative launching pad that you can use to edit, debug, and build code, and then publish an app.

- The Visual Studio .NET integrated development environment (IDE) consists of windows for visual design of forms, code-editing windows, menus and toolbars providing access to commands and features, toolboxes containing controls for use on the forms, and windows providing properties and information about forms, controls, projects, and the solution.

Visual Studio IDE Consisting following windows-

- 1. Menu Bar
- 2. Standard Toolbar
- 3. ToolBox
- 4. Forms Designer
- 5. Output Window
- 6. Solution Explorer
- 7. Properties Window


WindowsApplication1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Format Tools Test Window Help
Toolbox
Form1.vb [Design]*
Form1
Solution Explorer
Properties
1 Solution Explorer window
2 Form designer window
3 Toolbox window
4 Properties window
Ready

1] Toolbox Window -

- The Toolbox window contains all the controls you can use to build your application's interface.

- This window is usually retracted, and you must move the pointer over it to view the Toolbox.

- The controls in the Toolbox are organized in various tabs,

2] Solution Explorer Window -

- The Solution Explorer window contains a list of the items in the current solution.
- A solution can contain multiple projects, and each project can contain multiple items.

- The Solution Explorer displays a hierarchical list of all the components, organized by project.

3] Properties Window -

- This window (also known as the Properties Browser) displays all the properties of the selected component and its settings.

- Every time you place a control on a form, you switch to this window to adjust the appearance of the control.

4] Forms Designer -

- In a “Windows Form Application” project, the form is the heart of your application; it is the "screen" or "window" that your users will interact with.

- You can think of the form as a "canvas" on which you will place the various objects that make up your application.

5] Output Window –

- The output window is a child window used to capture the output from user tools, find in files output, scripting output, version control (in UEStudio), and various other areas of functionality that may write messages or output.

Q.11] What is mean by event handling ? Explain the Mouse & Keyboard?

Event Handling -

- Events are basically a user action like key press, clicks, mouse movements, etc., or some occurrence like system generated notifications.

- Applications need to respond to events when they occur.

- Clicking on a button, or entering some text in a text box, or clicking on a menu item, all are examples of events.

- An event is an action that calls a function or may cause another event.

- Event handlers are functions that tell how to respond to an event.

VB.Net is an event-driven language. There are mainly two types of events −

- Mouse events
- Keyboard events

A] Handling Mouse Events

-Mouse events occur with mouse movements in forms and controls.

- Following are the various mouse events related with a Control class −

1. **MouseDown** − it occurs when a mouse button is pressed
2. **MouseEnter** − it occurs when the mouse pointer enters the control
3. **MouseHover** − it occurs when the mouse pointer hovers over the control

4. **MouseLeave** − it occurs when the mouse pointer leaves the control
5. **MouseMove** − it occurs when the mouse pointer moves over the control
6. **MouseUp** − it occurs when the mouse pointer is over the control and the mouse button is released
7. **MouseWheel** − it occurs when the mouse wheel moves and the control has focus

B] Handling Keyboard Events -

Following are the various keyboard events related with a Control class:

1. **KeyDown -** occurs when a key is pressed down and the control has focus
2. **KeyPress -** occurs when a key is pressed and the control has focus
3. **KeyUp -** occurs when a key is released while the control has focus

Q.12] Write a note on MDI form with its features?

- MDI stands for **Multiple Document Interface** applications that allow users to work with multiple documents by opening more than one document at a time.

- Whereas, a **Single Document Interface (SDI)** application can manipulate only one document at a time.

- The MDI applications act as the parent and child relationship in a form.

- A parent form is a container that contains child forms, while child forms can be multiple to display different modules in a parent form.

- MDI enables users to open more than one file at once.

- It was developed to enable people working with large amounts of data to easily view several documents simultaneously.

- The main advantage of MDI is that it makes it easier to manage files because they can be viewed side by side.

- For example, comparing two versions of the same spreadsheet is often helpful when working with multiple spreadsheets. With MDI, this becomes much simpler.

Features of MDI

• If one document is maximized all the document are maximized.

• If no document is maximized, each document can be in two states:

minimized or restored.

• The child document will be on main frame. If a document get minimized it could display inside the main frame.

• If the document is not minimized it will be on the main frame as original size or user defined size.

Q.13] Explain the different dialog boxes presented in VB.net?

1] **ColorDialog -**
- It represents a common dialog box that displays available colors along with controls that enable the user to define custom colors.

2] **FontDialog -**
- It prompts the user to choose a font from among those installed on the local computer and lets the user select the font, font size, and color.

3] **OpenFileDialog -**
- It prompts the user to open a file and allows the user to select a file to open.

4] **SaveFileDialog -**
- It prompts the user to select a location for saving a file and allows the user to specify the name of the file to save data.

5] **PrintDialog -**
- It lets the user to print documents by selecting a printer and choosing which sections of the document to print from a Windows Forms application.

Q.14] What is mean by state management? Explain the server side state management?

-State Management is a process by which state and page information is maintained over multiple requests for same or different pages.

- As HTTP is a stateless protocol, server does not store any information once the response is sent back to client based on his request.

- When user submits request again, the server treats it as a new user.
- This is called stateless model.

- This model was workable when static web sites were developed and hosted in the past.

- Now, with interactive web sites or dynamic web site, there is a need to preserve some information to identify user, interact with user again and again within same session and same application.

- This concept is known as Stateful protocol.
- The information can be related to user, data objects, web pages or server objects.

Server-Side state management

-Server-Side is a type of state management where all the information is stored or retained in the user's memory.

- Due to this functionality, there are more secure domains of the website on the server side as compared to client-side state management

Q.15] Write a note on client-side and server-side state management?

A] Client-Side – State Management -

- Client-side state management refers to the management and storage of data on the client side of a web application, typically within the user's browser.

- This is in contrast to server-side state management, where data is stored on the web server.

1] View State –

- View State within ASP.NET serves as a server-side mechanism used to preserve the values of controls and variables at the page level throughout postbacks.

2] Cookies –

- A set of Cookies is a small text file that is stored in the user's hard drive using the client's browser.

- Cookies are just used for the sake of the user's identity matching as it only stores information such as sessions id's, some frequent navigation or post-back request objects.

- Whenever we get connected to the internet for accessing a specific service, the cookie file is accessed from our hard drive via our browser for identifying the user.

- The cookie access depends upon the life cycle or expiration of that specific cookie file.

B] Server-Side – State Management

- Server-Side is a type of state management where all the information is stored or retained in the user's memory.

- Due to this functionality, there are more secure domains of the website on the server side as compared to client-side state management.

1] Session –

- The session is a very important technique to maintain state.

- Normally session is used to store information and identity.

- The server stores information using Session id.

Session Event –

- The session event can be seen in the project Global.asax file.

Two types of Session Events -

I] Session_Start -

- The Session_start event is raised every time a new user requests without a session ID.

II] Session_End -

- The Session_End event is raised when the session is ended by a user or a time out using the Session end method.

2] Application –

-The application State is a server-side management state.

- It is also called application-level state management.

- This mainly stores user activity in server memory and application events shown in Global.asax file.

There are three types of applications in ASP.NET.

I] Application_Start – This event begins with domain start.

II] Application_Error - In this section manage unhandled exception errors.

III] Application_ End - This ends with domain or restarts IIS.

Q.16] Explain the validation controls with its types?

-ASP.NET is a popular, open-source app development framework that enables developers to create dynamic web pages and design robust websites.

- This framework comes with various validation controls that enable the developers to set properties that can validate the user data to ensure that the entered data satisfies the condition.

1] Required Field Validator

- RequiredFieldValidator is known as an elementary validation control.
- There is no form that doesn’t consist of fields that are mandatory to be filled.
- If the users want to proceed with the form, they will have to mandatorily fill these fields.

- To ensure that such fields are not left empty, RequiredFieldValidator is used.

- Basically, this validation checks that there must be some value-added within the control.

2] Compare Validator

-The CompareValidator control compares the value of one control with either a fixed value or a value in another control.

• **ControlToCompare** – It holds the ControlToValidate Id of another form of control.

-The value of both the form fields is then compared.

• **ValueToCompare** – A fixed value with which the comparison has to be made.

3] Range Validator

- The RangeValidator is a validation control that is used by the .NET developers to check whether the value of the input control is inside some specific range or not.

- This type of control is used when it comes to getting inputs like Age, Date of Birth, or mobile numbers from the user on the websites.

4] Regular Expression Validator

-RegularExpressionValidator or Regex is a control that holds various patterns which clearly define the format of the text.

-This means that if the text that has been added is in the same format then the Regex control will return true or else false.

-This type of validation control is generally used to validate input fields like email, phone, Zip code etc.

5] Custom Validator

- The CustomValidator control allows developers to customize the validation code to their specific user inputs.

- This is primarily useful when the developers need specific business logic in validation which cannot be implemented by the built-in controls.

- This control can operate both client-side and server-side validation.

- However, the server-side approach is often preferred as it is more powerful.

6] Validation Summary

-ValidationSummary is a control that is used to display error messages.

- It is a control that collects every type of validation control error message and is used by the other validation controls on a web page.

- It then displays the error message on the screen.

Q.17] What is mean by caching? Explain the mechanism of caching?

-Caching is a technique of storing frequently used data/information in memory, so that, when the same data/information is needed next time, it could be directly retrieved from the memory instead of being generated by the application.

-Caching is extremely important for performance boosting in ASP.NET, as the pages and controls are dynamically generated here.

-It is especially important for data related transactions, as these are expensive in terms of response time.

-Caching places frequently used data in quickly accessed media such as the random access memory of the computer.

-The ASP.NET runtime includes a key-value map of CLR objects called cache.

-This resides with the application and is available via the HttpContext and System.Web.UI.Page.

-In some respect, caching is similar to storing the state objects.

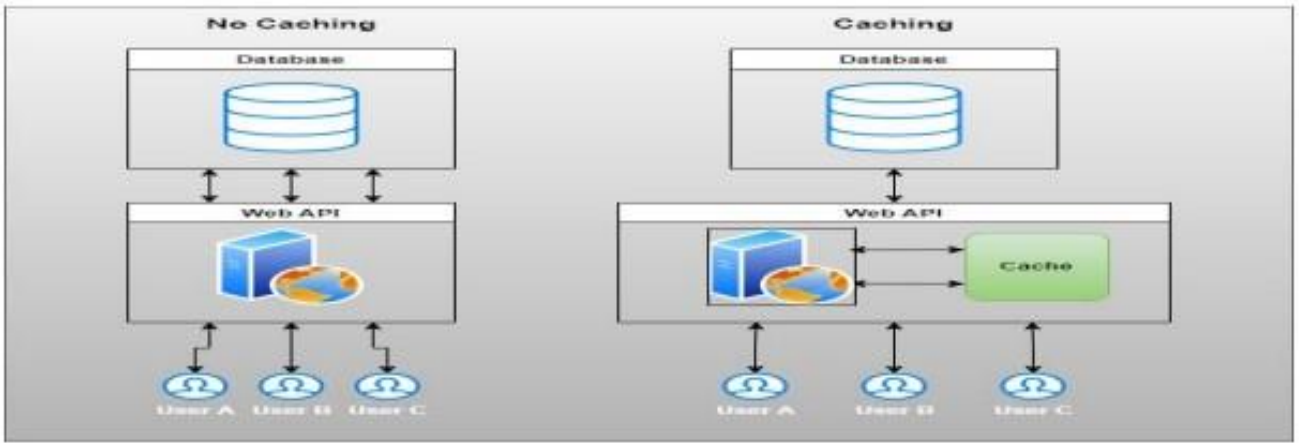
Mechanism –

-Sometimes our application frequently calls the same method and fetches the data from the database.

-The output of these requests is the same at all times.

-It doesn't get changed or updated in the database.

In this case, we can use caching to reduce the database calls and retrieve the data directly from the cache memory.



Q.18] Write a note on Authorization and Authentication ?

A] Authentication -

- Authentication is knowing the identity of the user.

-For example, Sham logs in with her username and password, and the server uses the password to authenticate Sham.

-Authentication is the process of validating user credentials and authorization is the process of checking privileges for a user to access specific modules in an application.

- Authentication is the process of obtaining some sort of credentials from the users and using those credentials to verify the user's identity.

- 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.

B] Authorization - is deciding whether a user is allowed to perform an action.

-For example, sham has permission to get a resource but not create a resource.

- Authorization is the process of allowing an authenticated user access to resources.

There are two closely interlinked concepts at the heart of security for distributed applications - authentication and authorization.

Q.19] What is mean by thread and difference between process and thread?

-A thread is defined as the execution path of a program.

- Each thread defines a unique flow of control.

- If your application involves complicated and time consuming operations, then it is often helpful to set different execution paths or threads, with each thread performing a particular job.

Difference between Process and Thread:		
S.NO	**Process**	**Thread**
1.	Process means any program is in execution.	Thread means a segment of a process.
2.	The process takes more time to terminate.	The thread takes less time to terminate.
3.	It takes more time for creation.	It takes less time for creation.
4.	It also takes more time for context switching.	It takes less time for context switching.
5.	The process is less efficient in terms of communication.	Thread is more efficient in terms of communication.
6.	Multiprogramming holds the concepts of multi-process.	We don’t need multi programs in action for multiple threads because a single process consists of multiple threads.
7.	The process is isolated.	Threads share memory.
8.	The process is called the heavyweight process.	A Thread is lightweight as each thread in a process shares code, data, and resources.

9.	Process switching uses an interface in an operating system.	Thread switching does not require calling an operating system and causes an interrupt to the kernel.
10.	If one process is blocked then it will not affect the execution of other processes	If a user-level thread is blocked, then all other user-level threads are blocked.

Q.20] Explain the try catch block with its syntax?

C# try and catch

- The `try` statement allows you to define a block of code to be tested for errors while it is being executed.
- The `catch` statement allows you to define a block of code to be executed, if an error occurs in the try block.

The `try` and `catch` keywords come in pairs:

Syntax -

```
try
{
  //  Block of code to try
}
catch (Exception e)
{
  //  Block of code to handle errors
}
```

C# try/catch example

```
using System;
public class ExExample
{
    public static void Main(string[] args)
    {
        try
        {
            int a = 10;
            int b = 0;
            int x = a / b;
        }
        catch (Exception e) { Console.WriteLine(e); }

        Console.WriteLine("Rest of the code");
    }
}
```

Output:

```
System.DivideByZeroException: Attempted to divide by zero.
Rest of the code
```

Q.21] Explain the thread life cycle in detail?

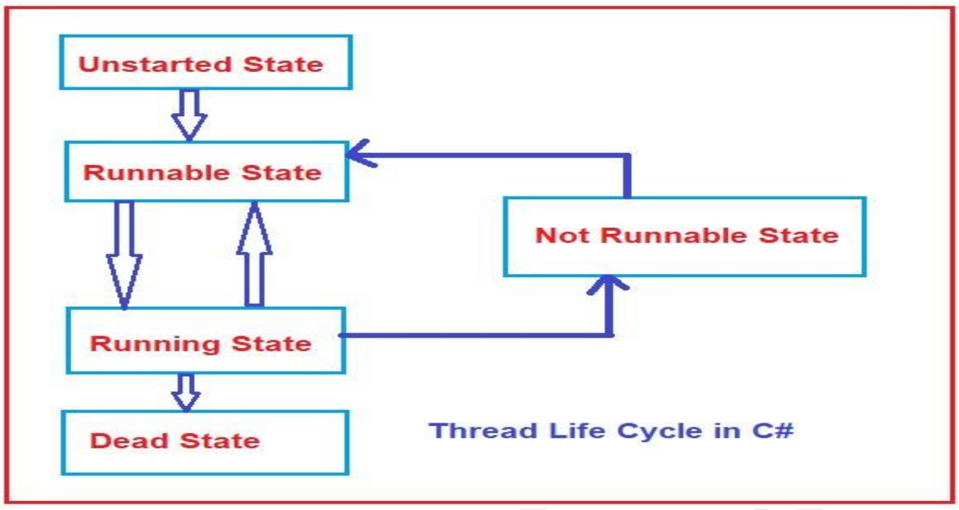
- In C#, each thread has a life cycle.

- The life cycle of a thread is started when instance of system *Threading*.

- *Thread class* is created. When the task execution of the thread is completed, its life cycle is ended.

There are following states in the life cycle of a Thread in C#.

- Unstarted
- Runnable (Readrun)
- Running
- Not Runnable
- Dead (Terminated)



1] Unstarted state:

-When an instance of a Thread class is created, it is in the unstarted state, means the thread has not yet started to run when the thread is in this state. Or in other words Start() method is not called.

-Thread thr = new Thread();

-Here, thr is at unstarted state.

2] Runnable State -

-A thread that is ready to run is moved to runnable state. In this state, a thread might actually be running or it might be ready to run at any instant of time.

-It is the responsibility of the thread scheduler to give the thread, time to run. Or in other words, the Start() method is called.

3] Running State –

-A thread that is running. Or in other words, the thread gets the processor.

4] Not Runnable State –

-A thread that is not executable because Sleep() method is called.
Wait() method is called.

Due to I/O request. Suspend() method is called.

5] Dead State –

-When the thread completes its task, then thread enters into dead, terminates, abort state.

Q.22] What is mean by exception? Explain types of error in application?

-An exception is an unwanted error that occurs during the execution of a program and can be a system exception or application exception.

- Exceptions are nothing but some abnormal and typically an event or condition that arises during the execution, which may interrupt the normal flow of the program.

- An exception can occur due to different reasons, including the following:

1. A user has entered incorrect data or performs a division operator, such as an attempt to divide by zero.
2. A connection has been lost in the middle of communication, or system memory has run out.

1] Compilation errors –

- When we compile the code, the errors we get when the code doesn't get compiled are called Compilation errors.
- When we do typographical errors, type mismatches, misspelling and many other things and try to execute the code, it gives a compilation error.
- The syntax errors are caught by the compiler and it throws a compilation error.
- C# not only tells us there is compilation error but it also tells us where it is giving the error.
- For Example:- If I write Console.WriteLine("Hi") instead of Console.WriteLine("Hi");

- it will give you a compilation error and the error will be error: ;expected .

2] Logical Eroors –

- Apart from compilation errors, there is one more type of error, logical error.

- This error occurrs when we write the logic of the code incorrectly.

- We are the only ones who can find these types of errors.

- The compiler will find no logical error as the syntax may be right so it will pass the code with no compile errors.

- For example: I want to get the perimeter of a square but instead of the right formula i.e. 4*side we write 2*side.

- The syntax is right but we are not getting the right answer, the reason is incorrect logic.

3] Runtime errors –

- Thrown during the program execution. Runtime errors are the ones that occur when the program is running.

- The program must have been compiled correctly, so they were no compilation errors, but something goes wrong during the execution of the program.

- For example, we might try to access the first element of an empty list.

- As you can see, there are no compilation errors present, and I'm able to run this program.

- But when I do, we can see an exception: "Sequence contains no element's.

- It may be a bit more tricky to fix a runtime error.

Q.23] Write a note on Data Provider available in ADO.NET?

Selecting an appropriate data provider for a client application depends on the type of data source being accessed. There are four .Net data providers available.

1. **SQL Server**: It's used to work specifically with Microsoft SQL Server. It exists in a namespace within the System.Data.SqlClient.

2. **OLE DB**: It's used to work with the OLEDB provider. The System.Data.dll assembly implements the OLEDB .NET framework data provider in the System.Data.OleDb namespace.

3. **ODBC:** To use this type of provider, you must use an ODBC driver. The System.Data.ODBC.dll assembly implements the ODBC .NET framework data provider. This assembly is not part of the Visual Studio .NET installation.

4. **Oracle:** The System.Data.OracleClient.dll assembly implements the Oracle .NET framework data provider in the System.Data.OracleClient namespace. The Oracle client software must be installed on the system before you can use the provider to connect to an Oracle data source.

Data Provider Components –

1] Connection: Establishes a connection to a specific data source.

Open(): Opens the connection for accessing the database.
Close(): Closes the connection to the database.

2] Command: Executes a command against a data source. Exposes Parameters and can execute in the scope of a Transaction from a Connection.

3] Data Reader

- DataReader is used to read the data from the database and it is a read and forward only connection oriented architecture during fetch the data from database.

4] Data Set

- DataSet is a disconnected orient architecture that means there is no need of active connections during work with datasets and it is a collection of Data Tables and relations between tables

5] Data Adapter

- Data Adapter will acts as a Bridge between DataSet and database. Dataadapter is a disconnected oriented architecture.

6] Data Table

- DataTable represents a single table in the database. It has rows and columns.

7] Execute Non Query

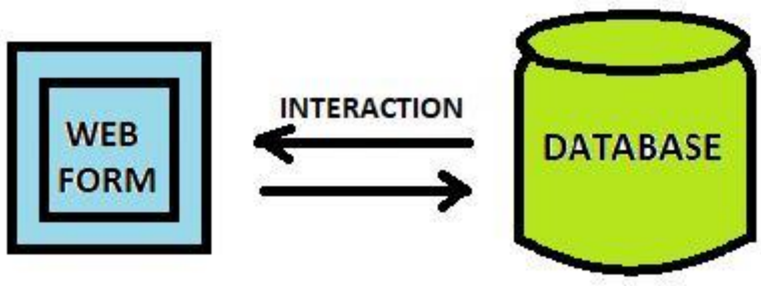
- ExecuteNonQuery method is used to execute SQL Command or the stored procedure performs INSERT, UPDATE, or Delete operations.

8] Execute Reader method is used to execute a SQL Command or stored procedure returns a set of rows from the database.

Q.24] Explain the connected and disconnected architecture?

1. **Connected Architecture -**

- Connected architecture refers to the fact that the connection is established for the full time between the database and application.
- For e.g. we make a program in C# that is connected with the database for the full time, so that will be connected architecture.
- Connected architecture is forward only and read-only.
- This means the connected mode will work only in one particular direction i.e. forward and that too for read-only purpose.
- Application issues query then read back results and process them.
- For connected architecture, we mainly use the object of the DataReader class.
- DataReader is used to retrieve the data from the database and it also ensures that the connection is maintained for the complete interval of time.
- In connected architecture, the application is directly linked with the Database.



2. Disconnected Architecture

- Disconnected architecture refers to the mode of architecture in Ado.net where the connectivity between the database and application is not maintained for the full time.
- Connectivity within this mode is established only to read the data from the database and finally to update the data within the database.
- This means during the processing of the application, we need data so that data is fetched from the database and kept in temporary tables.
- After that whenever data is required, it is fetched from the temporary tables.
- And finally, when the operations were completed, the connection was established to update the data within the database from the temporary tables.
- In this mode, application issues query then retrieves and store results for processing.
- For this purpose, we use objects of SqlDataAdapter and DataSet classes.

In disconnected architecture, a Dataset is used for retrieving data from the database. This way there is no need to establish a connection for the full time because DataSet acts as temporary storage. All the operations can be performed on the data using the Dataset and finally modified at the database

Q.25] Write a note on data reader & dataset?

1] Dataset -

1. DataSet object can contain multiple rowsets from the same data source as well as from the relationships between them.
2. Dataset is a disconnected architecture
3.Dataset can persist data.
4.Disconnected.
5.Can traverse data in any order front, back.
6.Data can be manipulated within the dataset.
7.More expensive than datareader as it stores multiple rows at the same time.

2] Datareader -

1.DataReader provides forward-only and read-only access to data.
2.Datareader is connected architecture
3.Datareader can not persist data.
4.Connection needs to be maintained all the time.
5.Can traverse only forward.
6.It is read only therefore, data cannot be manipulated.
7.It is less costly because it stores one row at a time.

Q.26] Explain the Data Adapter, Data Table & Execute Non Query in detail?

1] Data Adapter –

- DataAdapter will acts as a Bridge between DataSet and database.

- This dataadapter object is used to read the data from database and bind that data to dataset.

- Dataadapter is a disconnected oriented architecture.

2] Data Table -

- DataTable represents a single table in the database.

- It has rows and columns.

- There is no much difference between dataset and datatable, dataset is simply the collection of datatables.

3] Execute Non Query -

- Execute Non Query method is used to execute SQL Command or the stored procedure performs INSERT, UPDATE, or Delete operations. It doesn't return any data from the database. Instead, it returns an integer specifying the number of rows inserted, updated or deleted.