Q1. What is the main component in the .Net Framework? (T)

Ans. **.NET Components**

The architecture of the .Net framework is based on the following key components;

### **1. Common Language Runtime**

The "Common Language Infrastructure" or CLI is a platform on which the .Net programs are executed.

The CLI has the following key features:

* Exception Handling - Exceptions are errors which occur when the application is executed.  
  Examples of exceptions are:
  + If an application tries to open a file on the local machine, but the file is not present.
  + If the application tries to fetch some records from a database, but the connection to the database is not valid.
* Garbage Collection - Garbage collection is the process of removing unwanted resources when they are no longer required.  
  Examples of garbage collection are
  + A File handle which is no longer required. If the application has finished all operations on a file, then the file handle may no longer be required.
  + The database connection is no longer required. If the application has finished all operations on a database, then the database connection may no longer be required.
* Working with Various programming languages –

A developer can develop an application in a variety of .Net programming languages-

1. Language - The first level is the programming language itself, the most common ones are VB.Net and C#.
2. Compiler – There is a compiler which will be separate for each programming language. So underlying the VB.Net language, there will be a separate VB.Net compiler. Similarly, for C#, you will have another compiler.
3. Common Language Interpreter – This is the final layer in .Net which would be used to run a .net program developed in any programming language. So the subsequent compiler will send the program to the CLI layer to run the .Net application.

### **2. Class Library**

The .NET Framework includes a set of standard class libraries. A class library is a collection of methods and functions that can be used for the core purpose.

For example, there is a class library with methods to handle all file-level operations. So there is a method which can be used to read the text from a file. Similarly, there is a method to write text to a file.

### **3. Languages**

The types of applications that can be built in the .Net framework are classified broadly into the following categories.

* WinForms – This is used for developing Forms-based applications, which would run on an end user machine. Notepad is an example of a client-based application.
* ASP.Net – This is used for developing web-based applications, which are made to run on any browser such as Internet Explorer, Chrome or Firefox.
  + The Web application would be processed on a server, which would have Internet Information Services Installed.
  + Internet Information Services or IIS is a Microsoft component which is used to execute an ASP.Net application.
  + The result of the execution is then sent to the client machines, and the output is shown in the browser.
* ADO.Net – This technology is used to develop applications to interact with Databases such as Oracle or Microsoft SQL Server.

Q2. Current .Net Standard framework and Current core framework?`(T)

Ans. .Net Standard Framework 4.8

Core Framework 3.1

Q3. Difference between Managed and Unmanaged code? (T)

Ans. **Managed Code**

Managed code is a code whose execution is managed by Common Language Runtime. It gets the managed code and compiles it into machine code. After that, the code is executed.The runtime here i.e. CLR provides automatic memory management, type safety, etc.

Managed code is written in high-level languages run on top of .NET. This can be C#, F#, etc. A code compiled in any of this language with their compilers, a machine code is not generated. However, you will get the Intermediate Language code, compiled and executed by runtime

C/C++ code, called "unmanaged code” does not have that privilege. The program is in binary that is loaded by the operating system into the memory. Rest, the programmer has to take care of.

## **Unmanaged Code**

* Applications that are not under the control of the CLR are unmanaged
* The unsafe code or the unmanaged code is a code block that uses a pointer variable.
* The unsafe modifier allows pointer usage in unmanaged code.

Q4. What do you mean by MSIL Code? (T)

Ans. MSIL stands for Microsoft Intermediate Language. We can call it as Intermediate Language (IL) or Common Intermediate Language (CIL). During the compile time , the compiler convert the source code into Microsoft Intermediate Language (MSIL) .Microsoft Intermediate Language (MSIL) is a CPU-independent set of instructions that can be efficiently converted to the native code.

Q5. What is the difference between .Net and .Net Core Frameworks? (T)

Ans. Dot Net is the full fat original MS development framework. It’s been around now since the early 2000’s and is considered to be very mature, and very stable.

It’s still an active project, and is still supported, but is not changed very often due to the massive amount of enterprise projects that depend on it.

Dot Net is also not portable, it’s restricted to the Windows Platform only, and many things like web applications require the support of extra platform features such as the IIS Web server.

Dot Net Core is the newcomer, it’s Microsoft’s vision of “Dot Net” for the future, it’s designed to directly compete with NodeJS in the web applications space, while still allowing you to build standard Dot Net applications.

It’s fast moving, has a very high octane release cycle, and should be expected to have breaking changes between major version releases.

It’s also cross platform, there are one or two features (Namely the desktop app stuff) coming in the upcoming Release 3 that will be initially restricted to windows, but in general everything you can do on windows with core, you can also do on Linux & mac.

In terms of API support, there are still some features that only full Dot Net supports, but as Core matures there’s a very good chance that this gap will increasingly close.

Q6. Explain the execution process of managed code in .Net? (T)

Ans. **Execution Process**

Managed execution process includes the following steps:

**Choosing the right compiler**

.Net Framework is a multilanguage execution environment, the runtime supports a wide variety of data types and language features. In order to obtain the full benefits provided by the common language runtime , you should use one or more language (VB.Net, C# etc.) compilers that target the runtime

**Compiling the code to MSIL**

Unlike the execution style of compiling source code into machine level code, .Net language compilers translates the source code into Microsoft Intermediate Language. This ensures language interoperability because no matter which language has been used to develop the application, it always gets translated to Microsoft Intermediate Language. During the compile time the compiler produces metadata, that contains description of the program like dependencies, versions etc

**Compiling MSIL to native code**

Before the program execution , Jist In Time compiler (JIT) compiles the MSIL into native code and stores it in a memory buffer. During JIT compilation, the code is also checked for type safety. Type safety ensures that objects are always accessed in a compatible way. The compiled native code is in memory and is not persisted. So every time we run our application this whole thing has to happen again.

**Excecution of Code**

After translating the IL into native code, it is sent to .Net runtime manager. The .Net runtime manager executes the code. During execution, managed code receives services such as garbage collection, security, interoperability with unmanaged code, cross-language debugging support, and enhanced deployment and versioning support.

Q7. Write a sample program to understand CLS and CTS? (P)

Ans. C#

using System;

namespace ConsoleApp1

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Hello World!");

int a = 5;

Console.WriteLine(a);

}

}

}

VB.Net

Module Program

Sub Main(args As String())

Console.WriteLine("Hello World!")

Dim a As Integer = 5

Console.WriteLine(a)

End Sub

End Module