**Questions for this assignment**

Explain about the .NET Framework?

Explain about Common Language Infrastructure (CLI)?

Explain about Common Language Runtime (CLR)?

What is the difference between the .NET Framework and .NET Core?

Explain about Common Intermediate Language (CIL)?

What are the benefits of using CIL in .NET?

Explain about Just-In-Time (JIT) compilation?

What are the components of the CLR?

How does the Security Engine enforce security policies in .NET?

What is the difference between CIL and machine code?

What is meant by Unmanaged or Managed Code?

Mention the features of C# briefly

Explain about the .NET Framework?

The .NET Framework is a software development platform created by Microsoft that provides developers with a way to build applications for Windows. It includes a large library of pre-built code and tools for developing, deploying, and managing software.

Explain about Common Language Infrastructure (CLI)?

The Common Language Infrastructure is a specification that defines the runtime environment for .NET applications. It includes a set of rules for how programming languages can interact with the runtime and with each other.

Explain about Common Language Runtime (CLR)?

The Common Language Runtime is the component of the .NET Framework that manages the execution of .NET applications. It provides services such as memory management, security, and exception handling.

What is the difference between the .NET Framework and .NET Core?

The .NET Framework is a traditional Windows-only framework that has been around since 2002, while .NET Core is a newer, cross-platform framework that can run on Windows, Linux, and macOS. .NET Core is also open source and more modular than the .NET Framework.

Explain about Common Intermediate Language (CIL)?

The Common Intermediate Language (CIL) is a low-level, platform-agnostic language that is used by .NET compilers to generate executable code. CIL code is compiled into native code at runtime by the CLR, allowing .NET applications to run on any platform that has a compatible runtime environment.

What are the benefits of using CIL in .NET?

Using CIL provides several benefits, including platform independence, interoperability between different languages, and improved performance. Because CIL code is compiled at runtime, it can be optimized for the specific hardware and software environment, resulting in faster and more efficient code execution.

Explain about Just-In-Time (JIT) compilation?

Just-In-Time (JIT) compilation is the process by which the CLR compiles CIL code into native code at runtime, as needed. This allows the CLR to optimize the code for the specific hardware and software environment, resulting in faster and more efficient code execution.

What are the components of the CLR?

The CLR consists of several components, including the Garbage Collector (GC), Just-In-Time (JIT) compiler, Type Checker, Security Engine, exception hander and thread manager. The GC manages memory allocation and deallocation, the JIT compiler converts CIL code to native code, the Type Checker verifies type safety, the Security Engine enforces security policies, and the Execution Engine manages the execution of code.

How does the Security Engine enforce security policies in .NET?

The Security Engine is responsible for enforcing security policies in .NET applications. It checks that code has the necessary permissions to access resources such as files, network connections, and system information. It also provides a sandboxed environment for running untrusted code, helping to prevent malicious code from damaging the system.

What is the difference between CIL and machine code?

CIL is a high-level, platform-agnostic language that is compiled into machine code at runtime by the CLR. Machine code is the low-level, platform-specific code that is executed by the processor. CIL provides several benefits, such as platform independence and interoperability, while machine code provides the highest level of performance and efficiency.

What is meant by Unmanaged or Managed Code?

**Managed Code**

“Managed code is the code that is developed using the .NET framework and its supported programming languages such as C# or VB.NET. Managed code is directly executed by the Common Language Runtime (CLR or Runtime), and the Runtime manages its lifecycle, including object creation, memory allocation, and object disposal. Any language that is written in .NET Framework is managed code".

**Unmanaged Code**

The code that is developed outside of the .NET framework is known as unmanaged code.

“Applications that do not run under the control of the CLR are said to be unmanaged. For example, languages such as C or C++, or Visual Basic are unmanaged.

The programmers directly manage the object creation, execution, and disposal of unmanaged code. Therefore, if programmers write bad code, it may lead to memory leaks and unwanted resource allocations.”

The .NET Framework provides a mechanism for unmanaged code to be used in managed code and vice versa. The process is done with the help of wrapper classes.

Mention the features of C# briefly

Some of the main features of C# are -

* C# is a safely typed and managed language.
* C# is object-oriented in nature.
* C# is a Cross-platform friendly language.
* C# is a platform-independent language when it comes to compilation.
* C# is general purpose in nature.
* C# is used in implementing Destructors and Constructors.
* C# is part of the .NET framework.
* C# is an easy-to-learn and easy-to-grasp language.
* C# is a structured language.