C# Interview Questions

# What is C#?

C# is a type-safe object-oriented Programming language that enables developers to build a variety of secure and robust applications that run on the .NET Framework.

**Follow Up : What can we do with C#**

Windows client applications – XML Web services – Distributed components – Client-server apps – DB apps and much more

# What is CLR?

It is the execution engine of the .NET framework that all .NET apps run under. The CLR provides these apps with Security, Portability, Automatic memory mgmt. The CLR internally contains a Security Manager, JIT (Just In Time) compiler and a Garbage Collector.

# What is a Class and/or what is an object?

An object is an instance of a class through which we access that class’ methods. An object is created by using the “new” keyword. It is a real time object. i.e. Car,Laptop etc.

A class can be seen as a blueprint. It contains information about the data, methods and behavior of an object.

# What are Jagged Arrays?

A jagged array is an array that has elements of the type Array. It can be more plainly thought of as an Array of arrays, which can be of different dimensions and sizes.

# What is enum?

An enumeration is a set of named integer constants. An enumeration type is declared using the “enum” keyword. C# enumerations are of value data type. In other words, enumeration contains its own values and cannot inherit or cannot pass inheritance.

# What is the difference between constants and read-only?

Constant variables are declared and initialized at compile time. Their value cannot be changed afterwards. Read-only is used when we want to assign the value at run time.

# What are Static Classes?

A static class is basically the same as a non-static class, the only difference being that a static class cannot be instantiated. In other words, you can’t use the new keyword to create an instance variable of the class. Because there is no instance variable you can access the members of a static class by using the class name itself.

# What are sealed classes in C#?

A sealed class in C# is used when we want to restrict a class from being inherited, and to prevent derivation from another class. If we forcefully specify a sealed class as base class, then a compile-time error occurs.

# What is the difference between an interface and an abstract class?

An interface contains methods that have only declaration but no definition, whereas an abstract class can have concrete methods. In an interface all methods are public, where as an abstract class can have private methods.

# What is the difference between method overriding and method overloading?

In method overriding, we change the method definition in the derived class that changes the method behavior.

Method overloading is creating a method with the same name, within the same class, but they have different signatures.

# What is the difference between a Struct and a Class?

Structs are value-type variables and cannot be inherited, whereas classes are reference types and can be inherited. We define Struct using the “struct” keyword and Class with the “class” keyword.

# Which Access Modifiers are available in C#?

**public –** There are no restrictions on accessing public members

**private –** Access is limited to within the class definition. This is the default access modifier type if none is formally specified.

**protected –** Can only be accessed within the class definition and any class that inherits from it

**internal –** Access is limited exclusively to classes defined within the current project assembly.

**protected internal –** Access is limited to the current assembly and types derived from the containing class. All members in the current project and all members in a derived class can access these

# What is Encapsulation?

Encapsulation refers to an object’s ability to hide data and behavior that are not necessary to its user. Encapsulation enables a group of properties, methods and other members to be considered a single unit. It restricts access to the members of a class so as to prevent a given class’ objects from being manipulated in ways that were not intended by the designer.

# What is abstraction?

The word abstract itself means a concept or an idea not associated with any specific instance. In terms of OOP we apply the same meaning of abstraction by making classes that are not associated with any specific instance. The abstraction is done when we need to only inherit from a certain class, but do not need to instantiate objects of that class. So you cannot create a new instance of an abstract class.

**i.e.** A method is a good example of abstraction because when we call a method, we are only aware of how to call that method and what it may do or return, but not the actual definition.

# Explain Polymorphism

Polymorphism means one name many forms. With polymorphism we can have one object that behaves as multiple forms, or one function that can behave in different forms. In other words, many forms of a single object is called Polymorphism.

**Follow Up – How can Polymorphism be achieved?**

Polymorphism can be achieved through method overloading and method overriding.

# What is inheritance in C#?

Inheritance is the ability to create a class from another class, the “base/parent” class, and extends the functionality and state of the base class to the “derived/child” class. It allows derived classes to have overloaded methods from the “base/parent” class. Simply put, we’re establishing a parent/child relationship between classes.

# What is the difference between String and StringBuilder

A string is immutable, which means that if you create a string object then you cannot modify it. A new object of type string is always created in memory when working with strings.

StringBuilder is mutable, meaning that if you create a StringBuilder object then you can perform operations on it, such as insert, replace, or append, without creating a new instance of it every time. It will update the string at one place and not create a new space in memory every time.