C# - INTERVIEW QUESTIONS

Dear readers, these **C# Interview Questions** have been designed specially to get you acquainted with the nature of questions you may encounter during your interview for the subject of **C#**. As per my experience good interviewers hardly plan to ask any particular question during your interview, normally questions start with some basic concept of the subject and later they continue based on further discussion and what you answer –

What is C#?

C# is a modern, general-purpose, object-oriented programming language developed by Microsoft. C# is designed for Common Language Infrastructure CLI, which consists of the executable code and runtime environment that allows use of various high-level languages on different computer platforms and architectures.

What is Jagged Array in C#?

A Jagged array is an array of arrays.

You can initialize a jagged array as –

```
int[][] scores = new int[2][]{new int[]{92,93,94},new int[]{85,66,87,88}};
```

Where, scores is an array of two arrays of integers - scores[0] is an array of 3 integers and scores[1] is an array of 4 integers.

In how many ways you can pass parameters to a method?

There are three ways that parameters can be passed to a method –

- **Value parameters** – This method copies the actual value of an argument into the formal parameter of the function. In this case, changes made to the parameter inside the function have no effect on the argument.
- **Reference parameters** – This method copies the reference to the memory location of an argument into the formal parameter. This means that changes made to the parameter affect the argument.
- **Output parameters** – This method helps in returning more than one value.

Can you return multiple values from a function in C#?

Yes! Using output parameters. A return statement can be used for returning only one value from a function. However, using output parameters, you can return two values from a function.

What is the difference between ref and out parameters?

Output parameters are similar to reference parameters, except that they transfer data out of the method rather than into it. Reference parameter copies the reference to the memory location of an argument into the formal parameter. This means that changes made to the parameter affect the argument.

What is namespace in C#?

A namespace is designed for providing a way to keep one set of names separate from another. The class names declared in one namespace does not conflict with the same class names declared in another.

What is the purpose of using statement in C#?

using keyword is used to include a namespace in the program. A program generally has multiple using statements.

What are value types in C#

Value type variables can be assigned a value directly. They are derived from the class System.ValueType.

The value types directly contain data. Some examples are int, char, and float, which stores numbers, alphabets, and floating point numbers, respectively. When you declare an int type, the system allocates memory to store the value.

What are reference types in C#

The reference types do not contain the actual data stored in a variable, but they contain a reference to the variables.

In other words, they refer to a memory location. Using multiple variables, the reference types can refer to a memory location. If the data in the memory location is changed by one of the variables, the other variable automatically reflects this change in value. Example of built-in reference types are: object, dynamic, and string.

Which class acts as a base class for all the data types in .net?

The Object Type is the ultimate base class for all data types in C# Common Type System CTS. Object is an alias for System.Object class. The object types can be assigned values of any other types, value types, reference types, predefined or user-defined types. However, before assigning values, it needs type conversion.

What is boxing in C#?

When a value type is converted to object type, it is called boxing.

What is unboxing in C#?

When an object type is converted to a value type, it is called unboxing.

What are dynamic type variables in C#

You can store any type of value in the dynamic data type variable. Type checking for these types of variables takes place at run-time.

Syntax for declaring a dynamic type is –

```
dynamic <variable_name> = value;
```

For example,

```
dynamic d = 20;
```

What is the difference between dynamic type variables and object type variables?

Dynamic types are similar to object types except that type checking for object type variables takes place at compile time, whereas that for the dynamic type variables takes place at run time.

What are pointer types in C#?

Pointer type variables store the memory address of another type. Pointers in C# have the same capabilities as the pointers in C or C++.

Syntax for declaring a pointer type is –

```
type* identifier;
```

For example

```
char* cptr;
int* iptr;
```

What is the purpose of is operator in C#?

is operator determines whether an object is of a certain type.

If$FordisCar$ // checks if Ford is an object of the Car class.

What is the purpose of as operator in C#?

as operator casts without raising an exception if the cast fails.

```
Object obj = new StringReader("Hello");
StringReader r = obj as StringReader;
```

What is encapsulation?

Encapsulation is defined 'as the process of enclosing one or more items within a physical or logical package'. Encapsulation, in object oriented programming methodology, prevents access to implementation details.

How encapsulation is implemented in C#?

Encapsulation is implemented by using access specifiers.

What is the purpose of an access specifier in C#?

An access specifier defines the scope and visibility of a class member.

What is scope of a public member variable of a C# class?

Public access specifier allows a class to expose its member variables and member functions to other functions and objects. Any public member can be accessed from outside the class.

What is scope of a private member variable of a C# class?

Private access specifier allows a class to hide its member variables and member functions from other functions and objects. Only functions of the same class can access its private members. Even an instance of a class cannot access its private members.

What is scope of a protected member variable of a C# class?

Protected access specifier allows a child class to access the member variables and member functions of its base class. This way it helps in implementing inheritance.

What is scope of a Internal member variable of a C# class?

Internal access specifier allows a class to expose its member variables and member functions to other functions and objects in the current assembly. In other words, any member with internal access specifier can be accessed from any class or method defined within the application in which the member is defined.

What is scope of a Protected Internal member variable of a C# class?

The protected internal access specifier allows a class to hide its member variables and member functions from other class objects and functions, except a child class within the same application. This is also used while implementing inheritance.

What are nullable types in C#?

C# provides a special data types, the nullable types, to which you can assign normal range of values as well as null values.

For example, you can store any value from -2,147,483,648 to 2,147,483,647 or null in a Nullable<Int32> variable. Similarly, you can assign true, false, or null in a Nullable<bool> variable.

What is the use of Null Coalescing Operator ?? in C#?

The null coalescing operator is used with the nullable value types and reference types. It is used for converting an operand to the type of another nullable *ornot* value type operand, where an implicit conversion is possible.

If the value of the first operand is null, then the operator returns the value of the second operand, otherwise it returns the value of the first operand.

Can you create a function in C# which can accept varying number of arguments?

By using the params keyword, a method parameter can be specified which takes a variable number of arguments or even no argument.

Can you pass additional type of parameters after using params in function definition?

No! additional parameters are not permitted after the params keyword in a method declaration. Only one params keyword is allowed in a method declaration.

Which class acts as a base class for all arrays in C#?

The Array class is the base class for all the arrays in C#. It is defined in the System namespace. The Array class provides various properties and methods to work with arrays.

How to sort an array in C#?

Using Array.sort*array* function. It sorts the elements in an entire one-dimensional Array using the IComparable implementation of each element of the Array.

How to sort an array in C# in descending order?

First sort the array using Array.sort*array* then reverse the same using Array.reverse*array* method.

What is a structure in C#?

In C#, a structure is a value type data type. It helps you to make a single variable hold related data of various data types. The struct keyword is used for creating a structure.

Structures are used to represent a record. To define a structure, you must use the struct statement. The struct statement defines a new data type, with more than one member for your program.

What are the differences between a class and structure

Classes and Structures have the following basic differences –

- classes are reference types and structs are value types.
- structures do not support inheritance.
- structures cannot have default constructor.

What is a enumeration in C#?

An enumeration is a set of named integer constants. An enumerated type is declared using the enum keyword.

C# enumerations are value data type. In other words, enumeration contains its own values and cannot inherit or cannot pass inheritance.

What is the default access for a class?

Default access specifier for a class type is internal.

What is the default access for a class member?

Default access for the members is private.

What is inheritance?

One of the most important concepts in object-oriented programming is inheritance. Inheritance allows us to define a class in terms of another class, which makes it easier to create and maintain an application. This also provides an opportunity to reuse the code functionality and speeds up implementation time.

When creating a class, instead of writing completely new data members and member functions, the programmer can designate that the new class should inherit the members of an existing class. This existing class is called the base class, and the new class is referred to as the derived class.

The idea of inheritance implements the IS-A relationship. For example, mammal IS A animal, dog IS-A mammal hence dog IS-A animal as well, and so on.

Is multiple inheritance supported in C#?

No! C# does not support multiple inheritance.

How to inherit a class in C#?

A class can be derived from more than one class or interface, which means that it can inherit data and functions from multiple base classes or interfaces. The syntax used in C# for creating derived classes is as follows –

```
<acess-specifier> class <base_class>
{
   ...
}
class <derived_class> : <base_class>
{
   ...
}
```

What is polymorphism?

The word polymorphism means having many forms. In object-oriented programming paradigm, polymorphism is often expressed as 'one interface, multiple functions'.

What is the difference between static polymorphism and dynamic polymorphism?

Polymorphism can be static or dynamic. In static polymorphism, the response to a function is determined at the compile time. In dynamic polymorphism, it is decided at run-time.

How C# supports static polymorphism?

C# provides two techniques to implement static polymorphism. They are –

- Function overloading
- Operator overloading

What is early binding?

The mechanism of linking a function with an object during compile time is called early binding. It is also called static binding.

What is function overloading?

You can have multiple definitions for the same function name in the same scope. The definition of the function must differ from each other by the types and/or the number of arguments in the argument list. You cannot overload function declarations that differ only by return type.

How C# supports dynamic polymorphism?

Dynamic polymorphism is implemented by abstract classes and virtual functions.

What is a sealed class in C#?

When a class is declared sealed, it cannot be inherited.

How will you create sealed abstract class in C#?

No! It can not be created as abstract classes cannot be declared sealed.

What are virtual functions in C#?

When you have a function defined in a class that you want to be implemented in an inherited class*es*, you use virtual functions. The virtual functions could be implemented differently in different inherited class and the call to these functions will be decided at runtime.

Is operator overloading supported in C#?

You can redefine or overload most of the built-in operators available in C#. Thus a programmer can use operators with user-defined types as well.

Overloaded operators are functions with special names the keyword operator followed by the symbol for the operator being defined. Similar to any other function, an overloaded operator has a return type and a parameter list.

What is an interface?

n interface is defined as a syntactical contract that all the classes inheriting the interface should follow. The interface defines the 'what' part of the syntactical contract and the deriving classes define the 'how' part of the syntactical contract.

Interfaces define properties, methods, and events, which are the members of the interface. Interfaces contain only the declaration of the members. It is the responsibility of the deriving class to define the members. It often helps in providing a standard structure that the deriving classes would follow.

What is a preprocessor directives in C#?

The preprocessor directives give instruction to the compiler to preprocess the information before actual compilation starts.

All preprocessor directives begin with #, and only white-space characters may appear before a preprocessor directive on a line. Preprocessor directives are not statements, so they do not end with a semicolon ;.

What is the use of conditional preprocessor directive in C#?

You can use the #if directive to create a conditional directive. Conditional directives are useful for testing a symbol or symbols to check if they evaluate to true. If they do evaluate to true, the compiler evaluates all the code between the #if and the next directive.

Which class acts as a base class for all exceptions in C#?

C# exceptions are represented by classes. The exception classes in C# are mainly directly or indirectly derived from the System.Exception class. Some of the exception classes derived from the System.Exception class are the System.ApplicationException and System.SystemException classes.

What is the difference between System.ApplicationException class and System.SystemException class?

The System.ApplicationException class supports exceptions generated by application programs. Hence the exceptions defined by the programmers should derive from this class.

The System.SystemException class is the base class for all predefined system exception.

What is Next ?

Further you can go through your past assignments you have done with the subject and make sure you are able to speak confidently on them. If you are fresher then interviewer does not expect you will answer very complex questions, rather you have to make your basics concepts very strong.

Second it really doesn't matter much if you could not answer few questions but it matters that whatever you answered, you must have answered with confidence. So just feel confident during your interview. We at tutorialspoint wish you best luck to have a good interviewer and all the very best for your future endeavor. Cheers :-)