**1. What is C#?**

C# is an object-oriented, type-safe, and managed language that is compiled by .Net framework to generate Microsoft Intermediate Language.

### 2. What is the difference between public, static, and void?

Public declared variables or methods are accessible anywhere in the application. Static declared variables or methods are globally accessible without creating an instance of the class. Static member is by default not globally accessible it depends upon the type of access modified used. The compiler stores the address of the method as the entry point and uses this information to begin execution before any objects are created. And Void is a type modifier that states that the method or variable does not return any value.

### 3. What is an object?

An object is an instance of a class through which we access the methods of that class. “New” keyword is used to create an object. A class that creates an object in memory will contain the information about the methods, variables, and behavior of that class.

### 4. Define Constructors

A constructor is a member function in a class that has the same name as its class. The constructor is automatically invoked whenever an object class is created. It constructs the values of data members while initializing the class.

### 5. What is the difference between ref & out parameters?

An argument passed as ref must be initialized before passing to the method whereas out parameter needs not to be initialized before passing to a method

### 6. What is method overloading?

Method overloading is creating multiple methods with the same name with unique signatures in the same class. When we compile, the compiler uses overload resolution to determine the specific method to be invoke

### 7. What is the difference between Array and Arraylist?

In an array, we can have items of the same type only. The size of the array is fixed when compared. To an arraylist is similar to an array, but it doesn’t have a fixed size.

### 8. What’s the difference between an interface and abstract class?

Interfaces have all the methods having only declaration but no definition. In an abstract class, we can have some concrete methods. In an interface class, all the methods are public. An abstract class may have private methods

**9. Summarize the Different Class Types in C#.**

While all classes in C# have a base type of *Object*, there are 4 distinct types.

* **Static Class:**Cannot instantiate objects with the *New* keyword, and class members can only be accessed by using the actual class name
* **Abstract class:** Restricted, which means it cannot be used to create objects directly, so we need to inherit from this via a sub-class if we want to access its members
* **Partial Class:**Allows methods, events, and properties to be split into separate/multiple .cs source files, which are combined into one class at compile time
* **Sealed Class:**Prevents users from inheriting from the class, and restricts access to the class members such as properties or methods

**10.Summarize the Four Access Modifiers in C#.**

1. **Private:** Only accessible from within the class
2. **Public:** Accessible from anywhere in the code
3. **Internal:** Only accessible at the current assembly point of the class
4. **Protected:** Only accessible by class members and classes that inherit from it

**11. How Do You Use Nullable Types in C#?**

In order to assign a null value to *Value Types* in C#, we have to use the *Nullable Type*. This can be done with either the *Nullable* keyword or the ‘?’ operator shorthand, as shown below.

It is not possible to use the *Nullable Type* with *vartype* variables because these are not explicitly defined but rather implicitly defined by the value assigned to the variable.

*Note that this is one of those C# important questions you need to understand properly.*

**Nullable Type example:**

Nullable<**int**> i = **null**;

**int**? i = **null**;

**12. What are the fundamental OOP concepts?**

Answer: The four fundamental concepts of Object-Oriented Programming are:

* **Encapsulation**: Here, the internal representation of an object is hidden from the view outside the object’s definition. Only the required information can be accessed whereas the rest of the data implementation is hidden.
* **Abstraction:** It is a process of identifying the critical behavior and data of an object and eliminating the irrelevant details.
* **Inheritance**: It is the ability to create new classes from another class. It is done by accessing, modifying and extending the behavior of objects in the parent class.
* **Polymorphism**: The name means, one name, many forms. It is achieved by having multiple methods with the same name but different implementations.

**13. What are the differences between a Class and a Struct?**

Answer: Given below are the differences between a Class and a Struct:

| **Class** | **Struct** |
| --- | --- |
| Supports Inheritance | Does not support Inheritance |
| Class is Pass by reference (reference type) | Struct is Pass by Copy (Value type) |
| Members are private by default | Members are public by default |
| Good for larger complex objects | Good for Small isolated models |
| Can use waste collector for memory management | Cannot use Garbage collector and hence no Memory management |

**14. How is Exception Handling implemented in C#?**

Answer: Exception handling is done using four keywords in C#:

* **try**: Contains a block of code for which an exception will be checked.
* **catch**: It is a program that catches an exception with the help of the exception handler.
* **finally**: It is a block of code written to execute regardless of whether an exception is caught or not.
* **Throw**: Throws an exception when a problem occurs.

### 15. Write a program in C# Sharp to reverse a string?

**internal** **static** **void** **ReverseString**(string str)

{

char[] charArray = str.ToCharArray();

**for** (int i = 0, j = str.Length - 1; i < j; i++, j--)

{

charArray[i] = str[j];

charArray[j] = str[i];

}

string reversedstring = **new** string(charArray);

Console.WriteLine(reversedstring);

## **16. What are the advantages of ASP.NET Core over ASP.NET?**

There are the following advantages of ASP.NET Core over ASP.NET:

* + It is cross-platform, so it can be run on Windows, Linux, and Mac.
  + There is no dependency on framework installation because all the required dependencies are shipped with our application
  + ASP.NET Core can handle more requests than the ASP.NET
  + Multiple deployment options available withASP.NET Core

## **17. How to enable Session in ASP.NET Core?**

The middleware for the session is provided by the package Microsoft.AspNetCore.Session. To use the session in the ASP.NET Core application, we need to add this package to the csproj file and add the Session middleware to the ASP.NET Core request pipeline.

public class Startup

{

public void ConfigureServices(IServiceCollection services)

{

services.AddSession();

services.AddMvc();

}public void Configure(IApplicationBuilder app, IHostingEnvironment env)

{

app.UseSession();} }