1. What is a Datatype?

A: Datatype means the type of data which can be stored in a variable. It also identifies the way memory is allocated to a variable as well as the operations that can be performed on the same variable.

2. What are Nullable types?

A: The value types that can accept a null value are called the Nullable types.

3. What is (??) operator in C#?

A: The ?? operator is specifically known as the null coalescing operator, and is used for defining a default value for a nullable value type.

4. How to check that a nullable variable is having value?

A: To check whether a nullable variable has value or not, HasValue property is used.

5. What are the differences between Object and Var?

A: The introduction of object was done with C# 1.0, while for Var it was C# 3.0.

You can use object when you want to store several types of values in a single variable;

while Var is used when you are not sure about the type of assigned value. Object can be passed as a method argument, while the same cannot be done with Var.

6. What is the ref keyword in C#?

A: The ref keyword is known for causing an argument that is passed by reference and not value.

7. What is the params keyword in C#?

A: The params keyword allows a method parameter to obtain a huge number of arguments. These large number of arguments are changed with the help of the compiler into elements in a short-term array.

8. What do you mean by operators in C#?

A: An operator is used as a symbol that tells the compiler about the kind of operations that can be performed on an operand.

9. What are the different types of operators in C#?

A: There are a total of seven operators in C#, which include:

- Arithmetic Operators: These are the kind of operators that are used for performing arithmetic operations.
- Relational Operators: These operators are used for comparing values. These always result in true or false (>,<,!=, etc).
- Logical/Boolean Operators: These operators are also used for comparing values. These also always result in true or false (! , &&).
- Bitwise Operators: These operators perform a bit-by-bit operation.
- Assignment Operators: These operators help in assigning a new value to the variable.
- Type Information Operators: These operators provide information about a certain type.
- Miscellaneous Operators: ?: , => , & , && , *

10. What is ternary operator in C#?

A: A ternary operator is used for a conditional expression that returns a Boolean value. It is a short form of if-else.

11. What is the static keyword mean in C#?

A: Static keyword helps to specify a static number. This means static members are not tied to a particular object, rather common to all objects.

12. What do you mean by Typecasting in C#?

A: Typecasting in C# is a mechanism that covers a certain type of value to another value. It is only probable when both the data types are well-suited with each other.

13. What are the different types of casting in C#?

A: The different types of casting in C# are explicit conversion and implicit conversion. Explicit conversion means conversion from a larger data type to a smaller data type.

Implicit conversion is just the opposite in which smaller data types are converted into larger data types.

14. What is an Out keyword in C#?

A: Out keyword is used to pass the arguments to methods in the form of a reference type. It is mostly used when multiple values are returned by a method.

15. Can out and ref be used for overloading as the different signature of method?

A: That cannot be done. Even when ref as well as out is treated in a different way at runtime, they are treated the same way during compile time. Therefore, it cannot be loaded with similar kinds of arguments.

16. What do you mean by value type and reference type?

A: Actual values are stored by a value type variable; while references of actual values are stored by a reference type variable.

17. What is a safe and unsafe code in C#?

A: A safe code is the one that runs by the management of CLR; and, an unsafe code does not run by the management of CLR.

18. What is boxing and unboxing in C#?

A: Implicitly converting a value type into a reference type is called boxing.

Example:

```
1  int variable = 15
2  object boxingvar = variable
```

Explicitly converting the same reference into value type is called unboxing.

Example:

```
int variable = 15
object boxingvar = variable
int unboxed = (int)boxingvar
```

19. What is upcasting and downcasting?

A: Upcasting is implicitly converting the derived classes into a base class. Downcasting is just the opposite in which the base class is explicitly converted into a derived class.

20. What do you mean by static members?

A: Static members use static keywords. These can be called with class name.

21. What is the base class in .NET framework from which all the classes have been developed?

A: System.Object

22. What is a static class?

A: A static class is a class in which the CLR fills it with memory automatically during code execution.

23. When to use a static class?

A: A static class is significant if you want to provide common utilities such as configuration settings, driver functions, and many more.

24. What are sealed classes in C#?

A: Sealed classes are the special kinds of classes that are used to stop inheritance.

25. Can multiple catch blocks be implemented?

A: Multiple catch blocks cannot be implemented. Once you execute the proper catch code, the control is moved to the final block. After this, the code following the final block is implemented.

26. What is an object pool in .NET?

A: An object pool can be called a container with objects that can be used. It tracks the object that is in use at present.

27. What do you mean by partial method?

A: A partial method is basically a special method in a partial class, which is also called a struct. One part of a partial class has the only partial method declaration, which means signature, as well as the other part of the same struct, may have an execution.

28. What are the different ways a method can be overloaded?

A: Methods can be overloaded with the help of various types of data for a parameter, several orders of parameters, and a varied number of parameters.

29. What is serialization and its main purpose in C#?

A: Serialization in C# means converting an object into a stream of bytes. This further helps in storing the object into memory or fill in the form of XML, JSON, etc. Serialization helps in saving the state of an object so that it can later be recreated as per requirement.

30. What is reflection in C#?

A: Reflection is used for observing objects and their types. Reflection namespace has classes that enable you to get information about modules, assemblies, and types.

31. When should reflection be used?

A: Reflection can be used to create an instance of a type. It is also used to dynamically fix the type to an object that exists.

32. What do you mean by exceptions in C#?

A: These are unpredicted errors that take place during the implementation of a program. This is mostly caused because of inappropriate use of user inputs, system errors, to name a few.

33. What is the role of System.Exception class?

A: .NET framework provides System.Exception class to handle various types of exceptions that take place. The exception class is the basic class among the other exception classes.

34. What is exception handling?

A: Exception handling is a way of capturing run-time errors as well as handling them appropriately. Try-catch blocks, as well as throw keywords, are used to do it.

35. What are the various types of serialization?

A: The several types of serialization include XML serialization, Binary serialization, and SOAP serialization.

36. What are Delegates?

A: Delegates are the reference types that help in holding the reference to a method class. Methods that have the same signature as delegates can be allotted to delegates.

37. What is Polymorphism in C#?

A: Polymorphism is the ability of a programming language to the objects in various ways, which totally depend on their data type. Polymorphism is of two types, which include Compile-time polymorphism and Runtime polymorphism.

38. What are the uses of delegates in C#?

A: Delegates have several uses. Some of them are Callback Mechanism, Multicasting, Asynchronous Processing, and Abstract and Encapsulate methods.

39. What is the property in C#?

A: Property is a wrapper around a field. Property is used for assigning as well as reading the value from the field with the help of set and get accessors. The property can be created for various fields such as private, public, protected, and internal.

40. What are the uses of the property?

A: The uses of the property are validating data before allocating it to a field and logging all access for a child, when you need it.

41. What is the difference between “as” and “is” operators in C#?

A: “as” operators are used to cast the object to class. “is” operators are used to check the object with type. This will then return a Boolean value.

43. What is an indexer in C#?

A: An indexer allows a struct instance to be indexed just like an array.

44. How encapsulation is implemented in C#?

A: Implementation of encapsulation is done by using access specifiers. An access specifier helps in defining the visibility of a class member.

44. Why do we use collections in C#?

A: Collection classes are used to allocate memory to elements dynamically and access a list of items based on the index.

45. What are Generics in C#?

A: Generic is a class that enables you to define classes as well as methods by using a placeholder. Generics were part of version 2.0 of the C#. The intention of using Generic is to enable type to be a parameter to classes, methods, and interfaces.

46. What is Anonymous type in C#?

A: Anonymous type allows the users to create a new type without the need to define them. This is a way to define read-only properties in a single object without defining the type explicitly.

47. Can you briefly explain Thread Pooling in C#?

A: Thread Pool in C# is basically a collection of threads. Those threads are used for performing tasks without making a disturbance in the implementation of the primary

thread. Once a thread from the thread pool completes implementation, it returns to the thread pool.

48. What is Multithreading in C#?

A: Multithreading enables users to perform more than one operation simultaneously. To perform threading in C#, The .NET framework System.Threading namespace is used.

49. What are the different states of a Thread in C#?

A: The various states of a Thread in C# include:

- Aborted – This means the thread is dead but not stopped
- Running – This means the thread is implementing
- Stopped – This means implementation has been stopped by thread
- Suspended – This means the thread has been put off

50. How can the singleton design pattern in C# be used?

A: The singleton design pattern is used in C# in a situation when the class has one instance and the access has been provided widely.

51. What are the different types of decision-making statements in C#?

A: There are various types of decision-making statements that are included in C#. The statement types can be if statement, if-else statement, switch statement, and if-else-if statement.

52. Which one is better/faster, switch or if-else-if statements, and why?

A: Among these two, the switch statement is considered faster than the if-else-if statement. This is because the switch does not check earlier statements, but in case of if-else-if each condition has to be checked.

53. What is the goto statement?

A: A goto statement, provides a labeled statement with the control of the program. For this, the statement has to be within the scope of the goto statement.

54. What is the return statement in C#?

A: The function of a return statement in C# is to terminate the method's execution where it appears, and the control is retrieved to the calling method.

55. What is the jump statement in C#?

A: The jump statement in C# is used to transfer the program control from one point to another point in the program.

56. What does the throw statement do?

A: The throw statement is used for throwing an exception, indicating an error encountered while executing the program.

57. What do you mean by an array and what are the different types of the array in C#?

A: A collection of similar elements accessible through a numeric index is called an array. You can have one-dimensional array, two-dimensional array, and jagged array.

58. What is a multi-dimensional array?

A: A multidimensional array contains more than one level or dimension of array such as 2D and 3D array

```
1  int[,] arr2D = new int[6,8]; // declaration of 2D array
2  arr2D[0,0] = 1;
```

59. What is a jagged array?

A: An array, with elements consisting of arrays of different sizes and dimensions, is called a jagged array.

60. What do you mean by an object in C#?

A: A real-world entity having behaviors and attributes, an object in C# represents the class it belongs to. For its member functions, it carries out memory allocation.

61. What is a constructor?

A: It is a special function having the same name as its class. Whenever an object of a class is created, the constructor is invoked.

62. Why does the static constructor have no parameter?

A: As the static constructor is invoked automatically and called through the Common Language Runtime (CLR). It can't be directly called and that's why it doesn't have any parameters.

63. Why can you have only one static constructor?

A: Constructors must be overloaded to define multiple constructors for any given class. For this parameterized constructors must be defined that can accept outside parameters. The static constructors cannot be called directly and only through CLR which can't pass the parameter to the parameterized constructor.

64. What is a destructor in C#?

A: A special member function that is utilized for releasing the unmanaged resources allocated by the object. It has the same name as its class name following the ~(tilde) sign, and it can't be called explicitly.

65. What purpose does the *"using"* statement serve in C#?

A: Using statement fetches the resources that are specified, followed by using it, and then cleaning up through the dispose method when the statement is completely executed.

66. What is the purpose of an access modifier?

A: The access modifier can be utilized to specify the accessibility of the class member.

67. Can destructors have access modifiers?

A: As the destructors are directly called by the compiler, it cannot have access modifiers.

68. What is enum and when should it be used?

A: The enum is a value type that is used to store enumerators which is a list of named constants. Enum can be used to define static constants and constant flags.

69. What is the difference between class and structure?

A: Class is a reference type while the structure is a value type. Class can have a destructor while for a structure having a destructor is not possible. Class can have both parameterized constructor and default, while a structure can only have a default.

70. What is the difference between direct cast and ctype?

A: A direct cast is utilized for conversion of object type, that requires runtime similar to the specified type in direct cast. Ctype is instead used when converting the conversion defined for the expression and type.

71. When should you use Async and Await?

A: Async and Await are used in C# to create asynchronous methods. Asynchronous programming processes can run without any dependency on other processes including main processes.

72. What is Managed and Unmanaged code?

A: Managed code is written to get the services of managed runtime environment execution such as the CLR within the .NET framework. Unmanaged code on the other hand is executed directly by the operating system, providing low-level access and direct hardware access to the programmer.

73. Explain Namespaces in C#?

A: Namespace is utilized for organizing large code projects. You can create a custom namespace and also create them within another namespace which is known as nested namespaces.

74. What is a Deadlock?

A: Deadlock is an occurrence where two or more processes are waiting for the other to finish, and therefore a process cannot finish its execution. It is commonly found in multi-threading.

Going through these questions is a sure way to feel prepped up about your upcoming interview. It is also a great resource even if you do not have an interview lined up but wish to grasp the basics of the language and building a strong foundation for your career.

Read More Interview Questions

- **SQL Interview Questions**
- **Python Interview Questions**
- **Java Interview Questions**

FAQs Related to C#

Q: Is C# worth learning?

A: Learning C# is actually valuable. You can build any kind of application if you have learned C#. The applications that can be built are web services and web applications, games, console applications, desktop applications, IoT applications, windows services, AI applications, native mobile applications, cloud applications, and reusable libraries.

Q: What is the oops concept in C#?

A: Object-Oriented Programming (OOPs) is the programming paradigm that is defined by using objects. The objects can be considered as examples of real-world entities such as classes that have characteristics as well as behaviors.

Q: What is a collection in C#?

A: Collections in C# are specific classes for storing data and retrieval. These classes help in supporting queues, stacks, lists, and hash tables. The majority of the collection classes incorporate the same interfaces. These classes also help in creating collections of objects of the object classes, which is happen to be the basic class for all types of data in C#.

Q: What is the future of C#?

A: C# can be used for building Windows applications as well as creating applications that target Linux, iOS, MacOS, and Android operating systems. C# is one of the programming languages that is evolving quickly. Therefore, it can be said that the future of C# is bright.

Q: What is the type safety in C#?

A: Type safety in C# was introduced to stop the object of one type from peeking into the memory which was assigned for other objects. Safe code is also written to stop losing data during the conversion of one type to the other.

Q: What is a private constructor in C#?

A: A private constructor in C# is an instance constructor of a specific type. It is mostly used in classes that comprise only static members. If a class has more than one private constructor and zero public constructors, other classes cannot come up with examples of this class.

Q: Why do we use a private constructor?

A: A private constructor is used for putting a stop to creating instances of classes, mainly when there are instance fields or even methods, like the Math class or when a method is called for getting an instance of classes. It is majorly used for creating a singleton class. It also helps in stopping creating an instance of a class.