Question: What is C#? Write its features

Answer: C# is an object-oriented programming language that was developed by Microsoft in 2000. It is supported by different operating systems. C# is the primary language that is used to create .Net software applications. It allows us to create Windows UI apps, backend services, controls, libraries, android apps, and even blockchain applications. C# works on the concept of classes and objects just like Java.

Question: Describe the different C# classes in detail.

Answer: There are 4 types of classes that we can use in C#:

- **Static Class:** It is the type of class that cannot be instantiated, in other words, we cannot create an object of that class using the new keyword and the class members can be called directly using their class name.

- **Abstract Class:** Abstract classes are declared using the abstract keyword. Objects cannot be created for abstract classes. If you want to use it then it must be inherited in a subclass. You can easily define abstract or non-abstract methods within an Abstract class. The methods inside the abstract class can either have an implementation or no implementation.

- **Partial Class:** It is a type of class that allows dividing their properties, methods, and events into multiple source files, and at compile time these files are combined into a single class.

- **Sealed Class:** One cannot inherit a sealed class from another class and restricts the class properties. Any access modifiers cannot be applied to the sealed class.

Question: How can you describe object-oriented concepts in detail?

Answer: C# is an object-oriented programming language that supports 4 OOP concepts.

Encapsulation: defines the binding together code and the data and keeps it safe from any manipulation done by other programs and classes. It is a container that prevents code and data from being accessed by another program that is defined outside the container.

- **Abstraction**: this concept of object-oriented protects everything other than the relevant data about any created object in order to increase efficiency and security within the program.

- **Inheritance**: Inheritance is applied in such a way where one object uses the properties of another object.

- **Polymorphism**: is a feature that allows one interface to act as a base class for other classes. This concept is often expressed as a "single interface but multiple actions".

Question: Explain how code gets compiled in C#?

Answer: It takes 4 steps to get a code to get compiled in C#. Below are the steps:

1- Firstcompile the source code in the managed code compatible with the C# compiler.
2- Second, combine the above newly created code into assemblies.
3- Third, load the CLR.
4- Last, execute the assembly by CLR to generate the output.

Question: What is break and continue statements in C#, explain?

Answer: Below are the differences:

Break	**Continue**
You can use break statements in both switch and loop (for, while, and do-while) statements.	You can use continue statements only in the loop (for, while, do) statements.
The switch or loop statements terminate at the moment the break statement is executed and it ends abruptly from there.	You cannot make a continue statement terminate the loop, it carries on the loop to go to the next iteration level without executing the immediate next step.
The loop or switch exits immediately from the inner loop when the compiler encounters a break statement and comes out of the inner loop.	A continue that is placed inside a nested loop within a switch causes the next loop iteration.

Question: How you can explain the use of ‘using’ statements in C# in detail.

Answer: The using statement is used to control the usage of one or more resources that are being used within the program. The resources are continuously consumed and released. The main function of this statement is to manage unused resources and release them automatically. Once the object is created which is using the resource and when you are done you make sure that the object’s dispose metho d is called to release the resources used by that object, this is where using statements works well.

Question: How you can define the exception handling in C#?

Answer: An exception is a raised problem that may occur during the execution of the program. Handling exceptions offers a simple way to pass the control within the program whenever an exception is raised. C# exceptions are handled by using 4 keywords and those are try, catch, finally, throw.

- **try**: a raised exception finds a particular block of code to get handled. There is no limit on the number of catch blocks that you will use in your program to handle different types of exception raised.
- **catch:** you can handle the raised exception within this catch block. You can mention the steps that you want to do to solve the error or you can ignore the error by suppressing it by the code.
- **Finally:** irrespective of the error, if you still want some set of instructions to get displayed then you can use those statements within the finally block and it will display it on the screen.
- **throw:** you can throw an exception using the throw statement. It will display the type of error you are getting.

Question: Explain the concept of Destructor in detail. Explain it with an

example.

Answer: A destructor is a member that works just the opposite of the constructor. Unlike constructors, destructors mainly delete the object. The destructor name must match exactly with the class name just like a constructor. A destructor block always starts with the tilde (~) symbol.

A destructor is called automatically:

1. when the program finishes its execution.
2. Whenever a scope of the program ends that defines a local variable.
3. Whenever you call the delete operator from your program

Question: Define method overloading with example.

Answer: Method overloading allows programmers to use multiple methods but with the same name. Every defined method within a program can be differentiated on the basis of the number and the type of method arguments. It is a concept based on polymorphism.

Answer: Boxing- is a process of converting a value type to an object type where value type is placed on the stack memory, and the object type is placed in the heap memory. This conversion is an implicit conversion and you can directly assign any value to an object, and C# will handle the rest of the conversion on its own.

Example:

```
public void function()
{
Int a=111;
Object b=a; //implicit conversion
Console.WriteLine(b);
}
```

Unboxing- it is the reverse process of the boxing process. It is a conversion of the object type to the value type and the value of the boxed object type placed on the heap memory which will be transferred to the value type which is placed on the stack. This conversion of the unboxing process has to be done explicitly.

Example:

```
public void function()
{
Object b=111;
Int a=(int)b; //implicit conversion
Console.WriteLine(a);
}
```

Method overloading can be achieved by the following:

- By changing the number of parameters in the given method
- By changing the order of parameters passed to a method
- By using different data types as the passed parameters

Question: What is a different approach to the passing parameter in C#?

Answer: Parameters can be passed in three different ways to any defined methods and they are defined below:

Value Parameters: it will pass the actual value of the parameter to the formal parameter. In this case, any changes that are made into the formal parameter of the function will be having no effect on the actual value of the argument.

Reference Parameters: with this method, you can copy the argument that refers to the memory location into the formal parameter that means any changes made to the parameter affect the argument.

Output Parameters: This method returns more than one value to the method.

Question: How you can implement nullable<> types in C#? explain with the syntax of Nullable type.

Answer: In C#, you cannot put a null value directly into any variable and the compiler does not support it. So, the revised version **C# 2.0** provides you a special feature that will assign a null value to a variable that is called as the Nullable type. You cannot make the nullable types to work with value types. Nullable value can only work with the reference types as it already has a null value. System.Nullable<T> structure creates the instance nullable type, where T defines the data type. This T contains a non-nullable value type that can be any data type you want.

Syntax

```
Nullable<data_type> variable_name=null;
```

OR

```
Datatype? variable_name=null;
```

Question: What do you mean by value types and reference types in C#?

Answer:

Value type:

The memory allocated for the value type content or assigned value is stored on the stack. When we create any variable, space is allocated to that variable and then a value can be assigned to that variable. Also if we want to copy the value of that variable to another variable, its value gets copied and that creates two different variables.

Reference Type:

It holds the reference to the address of the object but not the object directly. Reference types represent the address of the variable and assigning a reference variable to another does not copy the data but it creates a second copy of the reference which represents the same location on the heap as the original value. Reference values are stored on the heap and when the reference variable is no longer required it gets marked for garbage collection.

Question: What are various types of comments in C#, explain with example?

Answer: C# supports three types of comments-

1. Single line comment

Syntax: //single line

2. Multiple line comment

Syntax: /* multiple lines

*/

3. XML comment

Syntax: /// set error

Question: What are the constructors?

Answer: In C#, there is a special method that is invoked automatically at the time of object creation. It initializes the data members of a new object and has the same name as the class or the structure. There are two types of constructors:

- **Default constructor**: it has no parameter to pass.
- **Parameterized constructor:** it is invoked with parameters that are passed to the class during object creation.

Question: What are the different collection classes in C#?

Answer: Collection classes are classes that are mainly used for data storage and retrieval. These collection classes will serve many purposes like allocating dynamic memory during run time and you can even access the items of the collection using the index value that makes the search easier and faster. These collection classes belong to the object class.

Array list: it refers to the ordered collection of the objects that are indexed individually. You can use it as an alternative to the array. Using index you can easily add or remove the items off the list and it will resize itself automatically. It works well for dynamic memory allocation, adding or searching items in the list.

Hash table: if you want to access the item of the hash table then you can use the key-value to refer to the original assigned value to the variable. Each item in the hash table is stored as a key/value pair and the item is referenced with its key value.

Stack: it works on the concept of last-in and first-out collection of the objects. Whenever you add an item to the list it is called pushing and when you remove the item off the list it is called popping.

Sorted list: this collection class uses the combination of key and the index to access the item in a list.

Queue: this collection works on the concept of first-in and first-out collection of the object. Adding an item to the list is call enqueue and removing the item off the list is call deque.

BitArray: this collection class is used to represent the array in binary form (0 and 1). You can use this collection class when you do not know the number and the items can be accessed by using integer indexes that start from zero.

Question: Explain file handling in C#.

Answer: Whenever you open a file for reading or writing it becomes a stream which is a sequence of bytes traveling from source to destination. The two commonly used streams are input and output. The included namespace is system.IO that includes many classes for file handling. The stream is an abstract class that is the parent class for the file handling process. The file is a static class with many static methods to handle file operation.

Below are the used classes:

The following table describes some commonly used classes in the System.IO namespace.

Class Name	**Description**
FileStream	This stream read from and write to any location within a file
BinaryReader	read primitive data types from a binary stream
DirectoryInfo	perform operations on directories
FileInfo	perform operations on files
BinaryWriter	write primitive data types in binary format
StreamReader	to read characters from a byte Stream
StreamWriter	write characters to a stream.
StringReader	read from a string buffer

StringWriter	write into a string buffer

Question: Explain the concept of thread in C#.

Answer: A thread can be defined as the execution flow of any program and defines a unique flow of control. You can manage these threads' execution time so that their execution does not overlap the execution of other threads and prevent deadlock or to maintain efficient usage of resources. Threads are lightweight programs that save the CPU consumption and increase the efficiency of the application. The thread cycle starts with the creation of the object of system.threading.thread class and ends when the thread terminates.

System.threading.thread class allows you to handle multiple threads and the first thread always runs in a process called the main thread. Whenever you run a program in C#, the main thread runs automatically.

Question: Define structure in C# with example.

Answer: A structure is a data type of a value type. A struct keyword is used when you are going to define a structure. A structure represents a record and this record can have many attributes that define the structure. You can define a constructor but not destructor for the structure. You can implement one or more interfaces within the structure. You can specify a structure but not as abstract, virtual, or protected. If you do not use the new operator the fields of the structure remain unassigned and you cannot use the object till you initialize the fields.

Question: How will you differentiate between a Class and a Struct?

Answer: Although both class and structure are user-defined data types, they are different in several fundamental ways. A class is a reference type and stores on the heap. Struct, on the other hand, is a value type and is, therefore, stored on the stack. While the structure doesn't support inheritance and polymorphism, the class provides support for both. A class can be of an abstract type, but a structure can't. All members of a class are private by default, while members of a struct are public by default. Another distinction between class and struct is based on memory management. The former supports garbage collection while the latter doesn't.

Question: What are Namespaces in C#?

Answer: Use of namespaces is for organizing large code projects. The most widely used namespace in C# is System. Namespaces are created using the namespace keyword. It is possible to use one namespace in another, known as Nested Namespace

Question: What are I/O classes in C#? Define some of the most commonly used ones.

Answer: The System.IO namespace in C# consists of several classes used for performing various file operations, such as creation, deletion, closing, and opening. Some of the most frequently used I/O classes in C# are:

- File – Manipulates a file
- Path – Performs operations related to some path information
- StreamReader – Reads characters from a stream
- StreamWriter – Writes characters to a stream
- StringReader – Reads a string buffer
- StringWriter – Writes a string buffer

Question: What do you understand by regular expressions in C#? Write a program that searches a string using regular expressions.

Answer: Regular expression is a template for matching a set of input. It can consist of constructs, character literals, and operators. Regex is used for string parsing, as well as replacing the character string. Following code searches a string "C#" against the set of inputs from the languages array using Regex:

```
System.Delegate namespace.
```

Following example demonstrates declaring a delegate:

```
public delegate AddNumbers(int n);
```

After declaring a delegate, the object must be created of the delegate using the new keyword, such as:

```
AddNumbers an1 = new AddNumbers(number);
```

Question: Explain different states of a Thread in C#?

Answer: A thread in C# can have any of the following states:

- Aborted – The thread is dead but not stopped
- Running – The thread is executing
- Stopped – The thread has stopped execution
- Suspended – The thread has been suspended
- Unstarted – The thread is created but has not started execution yet
- WaitSleepJoin – The thread calls sleep, calls wait on another object, and calls join on some other thread

Question: Why do we use Async and Await in C#?

Answer: Processes belonging to asynchronous programming run independently of the main or other processes. In C#, using Async and Await keywords for creating asynchronous methods

Question: What is an Indexer in C#, and how do you create one?

Answer: Also known as an indexed property, an indexer is a class property allowing accessing a member variable of some class using features of an array. Used for treating an object as an array, indexer allows using classes more intuitively. Although not an essential part of the object-oriented programming, indexers are a smart way of using arrays. As such, they are also called smart arrays. Defining an indexer enables creating classes that act like virtual arrays. Instances of such classes can be accessed using the [] array access operator. The general syntax for creating an indexer in C# is:

Question: What do you understand by Get and Set Accessor properties?

Answer: Made using properties, Get and Set are called accessors in C#. A property enables reading and writing to the value of a private field. Accessors are used for accessing such private fields. While we use the Get property for returning the value of a property, use the Set property for setting the value.

1. Can multiple catch blocks be executed?

No, Multiple catch blocks can't be executed. Once the proper catch code executed, the control is transferred to the finally block, and then the code that follows the finally block gets executed.

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 are 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 Jagged Arrays?(O que são matrizes irregulares)

The Array which has elements of type array is called jagged Array. The elements can be of different dimensions and sizes. We can also call jagged Array as an Array of arrays.

6. 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.

7. What is the use of 'using' statement in C#?

The 'using' block is used to obtain a resource and process it and then automatically dispose of when the execution of the block completed.

8. What is serialization?

When we want to transport an object through a network, then we have to convert the object into a stream of bytes. The process of converting an object into a stream of bytes is called Serialization. For an object to be serializable, it should implement ISerialize Interface. De-serialization is the reverse process of creating an object from a stream of bytes.

9. Can we use "this" command within a static method?

We can't use 'This' in a static method because we can only use static variables/methods in a static method.

10. What is the difference between constants and read-only?

Constant variables are declared and initialized at compile time. The value can't be changed afterward. Read-only is used only when we want to assign the value at run time.

11. What is an interface class? Give one example of it

An Interface is an abstract class which has only public abstract methods, and the methods only have the declaration and not the definition. These abstract methods must be implemented in the inherited classes.

```
using System;
namespace DemoApplication
{
   interface Guru99Interface
   {
      void SetTutorial(int pID, string pName);
      String GetTutorial();
   }

   class Guru99Tutorial : Guru99Interface
   {
      protected int TutorialID;
      protected string TutorialName;

      public void SetTutorial(int pID, string pName)
      {
         TutorialID = pID;
         TutorialName = pName;
      }

      public string GetTutorial() => $"TutorialID: {TutorialID}" + "\n" + $"TutorialName:
{TutorialName}";

      static void Main(string[] args)
      {
         Guru99Tutorial pTutor = new Guru99Tutorial();

         pTutor.SetTutorial(1, ".Net by Guru99");

         Console.WriteLine(pTutor.GetTutorial());

         Console.ReadKey();
      }
   }
```

}

14. What are value types and reference types?

Value Type

A value type holds a data value within its own memory space.

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.

Example:int a = 30;

The following table lists the available value types in C# 2010 −

Type	Represents	Range	Default Value
bool	Boolean value	True or False	False
byte	8-bit unsigned integer	0 to 255	0
char	16-bit Unicode character	U +0000 to U +ffff	'\0'
decimal	128-bit precise decimal values with 28-29 significant digits	(-7.9 x 10^{28} to 7.9 x 10^{28}) / 10^{0} to 28	0.0M
double	64-bit double-precision floating point type	(+/-)5.0 x 10^{-324} to (+/-)1.7 x 10^{308}	0.0D
float	32-bit single-precision floating point type	-3.4 x 10^{38} to + 3.4 x 10^{38}	0.0F

int	32-bit signed integer type	-2,147,483,648 to 2,147,483,647	0
long	64-bit signed integer type	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	0L
sbyte	8-bit signed integer type	-128 to 127	0
short	16-bit signed integer type	-32,768 to 32,767	0
uint	32-bit unsigned integer type	0 to 4,294,967,295	0
ulong	64-bit unsigned integer type	0 to 18,446,744,073,709,551,615	0
ushort	16-bit unsigned integer type	0 to 65,535	0

.

Reference Type

Reference type stores the address of the Object where the value is being stored. It is a pointer to another memory location.

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**.

Object Type

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.

When a value type is converted to object type, it is called **boxing** and on the other hand, when an object type is converted to a value type, it is called **unboxing**.

```
object obj;
obj = 100; // this is boxing
```

Dynamic Type

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;
```

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.

String Type

The **String Type** allows you to assign any string values to a variable. The string type is an alias for the System.String class. It is derived from object type. The value for a string type can be assigned using string literals in two forms: quoted and @quoted.

For example,

```
String str = "Tutorials Point";
```

A @quoted string literal looks as follows –

```
@"Tutorials Point";
```

The user-defined reference types are: class, interface, or delegate. We will discuss these types in later chapter.

Pointer Type

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;
```

```
string b = "Hello Guru99!!";
```

15. What are Custom Control and User Control?

Custom Controls are controls generated as compiled code (Dlls), those are easier to use and can be added to toolbox. Developers can drag and drop controls to their web forms. Attributes can, at design time. We can easily add custom controls to Multiple

Applications (If Shared Dlls). So, If they are private, then we can copy to dll to bin directory of web application and then add reference and can use them.

User Controls are very much similar to ASP include files, and are easy to create. User controls can't be placed in the toolbox and dragged - dropped from it. They have their design and code-behind. The file extension for user controls is ascx.

16. What are sealed classes in C#?

We create sealed classes when we want to restrict the class to be inherited. Sealed modifier used to prevent derivation from a class. If we forcefully specify a sealed class as base class, then a compile-time error occurs.

17. 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.

18. 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.

20. Describe the accessibility modifier "protected internal".

Protected Internal variables/methods are accessible within the same assembly and also from the classes that are derived from this parent class.

21. What are the differences between System.String and System.Text.StringBuilder classes?

System.String is immutable. When we modify the value of a string variable, then a new memory is allocated to the new value and the previous memory allocation released. System.StringBuilder was designed to have a concept of a mutable string where a variety of operations can be performed without allocation separate memory location for the modified string.

22. What's the difference between the System.Array.CopyTo() and System.Array.Clone() ?

Using Clone() method, we creates a new array object containing all the elements in the original Array and using CopyTo() method. All the elements of existing array copies into another existing array. Both methods perform a shallow copy.

23. How can we sort the elements of the Array in descending order?

Using Sort() methods followed by Reverse() method.

24. Write down the C# syntax to catch an exception

To catch an exception, we use try-catch blocks. Catch block can have a parameter of system.Exception type.

Eg:

```
try {
   GetAllData();
}
catch (Exception ex) {
}
```

25. 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.

27. What are circular references?

Circular reference is situation in which two or more resources are interdependent on each other causes the lock condition and make the resources unusable.

28. What are generics in C#.NET?

Generics are used to make reusable code classes to decrease the code redundancy, increase type safety, and performance. Using generics, we can create collection classes. To create generic collection, System.Collections.Generic namespace should be used instead of classes such as ArrayList in the System.Collections namespace. Generics promotes the usage of parameterized types.

29. What is an object pool in .NET?

An object pool is a container having objects ready to be used. It tracks the object that is currently in use, total number of objects in the pool. This reduces the overhead of creating and re-creating objects.

30. List down the commonly used types of exceptions in .net

ArgumentException, ArgumentNullException , ArgumentOutOfRangeException, ArithmeticException, DivideByZeroException ,OverflowException , IndexOutOfRangeException ,InvalidCastException ,InvalidOperationException , IOEndOfStreamException , NullReferenceException , OutOfMemoryException , StackOverflowException etc.

31. What are Custom Exceptions?

Sometimes there are some errors that need to be handled as per user requirements. Custom exceptions are used for them and are used defined exceptions.

32. What are delegates?

Delegates are same are function pointers in C++, but the only difference is that they are type safe, unlike function pointers. Delegates are required because they can be used to write much more generic type-safe functions.

33. How do you inherit a class into other class in C#?

Colon is used as inheritance operator in C#. Just place a colon and then the class name.

public class DerivedClass : BaseClass

34. What is the base class in .net from which all the classes are derived from?

ANSWER : System.Object

35. 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 having different signatures.

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

Methods can be overloaded using different data types for a parameter, different order of parameters, and different number of parameters.

37. Why can't you specify the accessibility modifier for methods inside the interface?

In an interface, we have virtual methods that do not have method definition. All the methods are there to be overridden in the derived class. That's why they all are public.

38. How can we set the class to be inherited, but prevent the method from being over-ridden?

Declare the class as public and make the method sealed to prevent it from being overridden.

Ques : What are the modifiers in C#?

Abstract

Sealed

Virtual

Const

Event

Extern

Override

Readonly

Static

New

Ques : What are the types of access modifiers in C#?

Access modifiers in C# are :

public

protect

private

internal

internal prote

Ques : What is boxing and unboxing?

Implicit conversion of value type to reference type of a variable is known as **BOXING**, for example integer to object type conversion.

Conversion of reference type variable back to value type is called as **UnBoxing**.

Ques : What is object?

An object is an instance of a class. An object is created by using operator new. A class that creates an object in memory will contain the information about the values and behaviours (or methods) of that specific object.

Ques : Where are the types of arrays in C#?

Single-Dimensional

Multidimensional

Jagged arrays.

Ques : What is the difference between Object and Instance?

An instance of a user-defined type is called an object. We can instantiate many objects from one class.

An object is an instance of a class.

Ques : Define destuctors?

A destructor is called for a class object when that object passes out of scope or is explicitly deleted.A destructors as the name implies is used to destroy

the objects that have been created by a constructors.Like a constructor , the destructor is a member function whose name is the same as the class name but is precided by a tilde.

Ques : What is the use of enumerated data type?

An enumerated data type is another user defined type which provides a way for attaching names to numbers thereby increasing comprehensibility of the code. The enum keyword automatically enumerates a list of words by assigning them values 0,1,2, and so on.

Ques : Define Constructors?

A constructor is a member function with the same name as its class. The constructor is invoked whenever an object of its associated class is created.It is called constructor because it constructs the values of data members of the class.

Ques : What is encapsulation?

The wrapping up of data and functions into a single unit (called class) is known as encapsulation. Encapsulation containing and hiding information about an object, such as internal data structures and code.

Ques : Does c# support multiple inheritance?

No,its impossible which accepts multi level inheritance.

Ques : What is ENUM?

Enum are used to define constants.

Ques : What is a data set?

A DataSet is an in memory representation of data loaded from any data source.

Ques : What is the difference between private and public keyword?

Private : The private keyword is the default access level and most restrictive among all other access levels. It gives least permission to a type or type member. A private member is accessible only within the body of the class in which it is declared.

Public : The public keyword is most liberal among all access levels, with no restrictions to access what so ever. A public member is accessible not only from within, but also from outside, and gives free access to any member declared within the body or outside the body.

Ques : Define polymorphism?

Polymorphism means one name, multiple forms. It allows us to have more than one function with the same name in a program.It allows us to have overloading of operators so that an operation can exhibit different behaviours in different instances.

Ques : What is Jagged Arrays?

A jagged array is an array whose elements are arrays.

The elements of a jagged array can be of different dimensions and sizes.

A jagged array is sometimes called an array-of-arrays.

Ques : what is an abstract base class?

An abstract class is a class that is designed to be specifically used as a base class. An abstract class contains at least one pure virtual function.

Ques : How is method overriding different from method overloading?

When overriding a method, you change the behavior of the method for the derived class. Overloading a method simply involves having another method with the same name within the class.

Ques : What is the difference between ref & out parameters?

An argument passed to a ref parameter must first be initialized. Compare this to an out parameter, whose argument does not have to be explicitly initialized before being passed to an out parameter.

Ques : What is the use of using statement in C#?

The using statement is used to obtain a resource, execute a statement, and then dispose of that resource.

Ques : What is serialization?

Serialization is the process of converting an object into a stream of bytes.

De-serialization is the opposite process of creating an object from a stream of bytes. Serialization / De-serialization is mostly used to transport objects.

Ques : What are the difference between Structure and Class?

Structures are value type and Classes are reference type

Structures can not have contractors or destructors.

Classes can have both contractors and destructors.

Structures do not support Inheritance, while Classes support Inheritance.

Ques : What is difference between Class And Interface?

Class : is logical representation of object. It is collection of data and related sub procedures with defination.

Interface : is also a class containg methods which is not having any definations.Class does not support multiple inheritance. But interface can support.

Ques : What is Delegates?

Delegates are a type-safe, object-oriented implementation of function pointers and are used in many situations where a component needs to call back to the component that is using it.

Ques : What is Authentication and Authorization?

Authentication is the process of identifying users. Authentication is identifying/validating the user against the credentials (username and password).

Authorization performs after authentication. Authorization is the process of granting access to those users based on identity. Authorization allowing access of specific resource to user.

Ques : What is a base class?

A class declaration may specify a base class by following the class name with a colon and the name of the base class. omitting a base class specification is the same as deriving from type object.

Ques : Can "this" be used within a static method?

No 'This' cannot be used in a static method. As only static variables/methods can be used in a static method

Ques : What is difference between constants, readonly and, static ?

Constants: The value can't be changed.

Read-only: The value will be initialized only once from the constructor of the class.

Static: Value can be initialized once

Ques : What are the different types of statements supported in C#?

C# supports several different kinds of statements are

Block statements

Declaration statements

Expression statements

Selection statements

Iteration statements

Jump statements

Try catch statements

Checked and unchecked

Lock statement

Ques : What is an interface class?

It is an abstract class with public abstract methods all of which must be implemented in the inherited classes.

Ques : what are value types and reference types?

Value types are stored in the Stack.

Examples : bool, byte, chat, decimal, double, enum , float, int, long, sbyte, short, strut, uint, ulong, ushort.

Reference types are stored in the Heap.

Examples : class, delegate, interface, object, string.

Ques : What is the difference between string keyword and System.String class?

String keyword is an alias for Syste.String class. Therefore, System.String and string keyword are the same, and you can use whichever naming convention you

prefer. The String class provides many methods for safely creating, manipulating, and comparing strings.

Ques : What are the two data types available in C#?

Value type

Reference type

Ques : What are the different types of Caching?

There are three types of Caching :

Output Caching: stores the responses from an asp.net page.

Fragment Caching: Only caches/stores the portion of page (User Control)

Data Caching: is Programmatic way to Cache objects for performance.

Ques : What is methods?

A method is a member that implements a computation or action that can be performed by an object or class. Static methods are accessed through the class.

Instance methods are accessed through instances of the class.

Ques : What is fields?

A field is a variable that is associated with a class or with an instance of a class.

Ques : What is events?

An event is a member that enables a class or object to provide notifications. An event is declared like a field except that the declaration includes an event keyword and the type must be a delegate type.

Ques : What is literals and their types?

Literals are value constants assigned to variables in a program. C# supports several types of literals are

Integer literals

Real literals

Boolean literals

Single character literals

String literals

Backslash character literals

Ques : What is the difference between value type and reference type?

Value types are stored on the stack and when a value of a variable is assigned to another variable.

Reference types are stored on the heap, and when an assignment between two reference variables occurs.

Ques : What are the features of c#?

C# is a simple and powerful programming language for writing enterprise edition applications.

This is a hybrid of C++ and VB. It retains many C++ features in the area statements,expressions, and operators and incorporated the productivity of VB.

C# helps the developers to easily build the web services that can be used across the Internet through any language, on any platform.

C# helps the developers accomplishing with fewer lines of code that will lead to the fewer errors in the code.

C# introduces the considerable improvement and innovations in areas such as type safety, versioning. events and garbage collections.

Ques : What are the types of comment in C#?

There are 3 types of comments in C#.

Single line (//)

Multi (/* */)

Page/XML Comments (///).

Ques : What are the namespaces used in C#.NET?

Namespace is a logical grouping of class.

using System;

using System.Collections.Generic;

using System.Windows.Forms;

Ques : What are the characteristics of C#?

There are several characteristics of C# are :

Simple

Type safe

Flexible

Object oriented

Compatible

Consistent

Interoperable

Modern

Ques : How does C# differ from C++?

C# does not support #include statement. It uses only using statement.

In C# , class definition does not use a semicolon at the end.

C# does not support multiple code inheritance.

Casting in C# is much safer than in c++.

In C# switch can also be used on string values.

Command line parameters array behave differently in C# as compared to C++

Ques : What are the basic concepts of object oriented programming?

It is necessary to understand some of the concepts used extensively in object oriented programming.These include

Objects

Classes

Data abstraction and encapsulation

Inheritance

Polymorphism

Dynamic Binding

Message passing.

Ques : Can you inherit multiple interfaces?

Yes. Multiple interfaces may be inherited in C#.

Ques : What is inheritance?

Inheritance is deriving the new class from the already existing one.

Ques : Define scope?

Scope refers to the region of code in which a variable may be accessed.

Ques : What is the difference between public, static and void?

public :The keyword public is an access modifier that tells the C# compiler that the Main method is accessible by anyone.

static :The keyword static declares that the Main method is a global one and can be called without creating an instance of the class. The compiler stores the address of the method as the entry point and uses this information to begin execution before any objects are created.

void : The keyword void is a type modifier that states that the Main method does not return

Ques : What are the modifiers in C#?

Abstract

Sealed

Virtual

Const

Event

Extern

Override

Readonly

Static

New

any value.

39. What happens if the inherited interfaces have conflicting method names?

Implement is up to you as the method is inside your own class. There might be a problem when the methods from different interfaces expect different data, but as far as compiler cares you're okay

40. What is the difference between a Struct and a Class?

Structs are value-type variables, and classes are reference types. Structs stored on the Stack causes additional overhead but faster retrieval. Structs cannot be inherited.

41. How to use nullable types in .Net?

Value types can take either their normal values or a null value. Such types are called nullable types.

```
Int? someID = null;
If(someID.HasVAlue)
{}
```

42. How we can create an array with non-default values?

We can create an array with non-default values using Enumerable.Repeat.

43. What is difference between "is" and "as" operators in c#?

"is" operator is used to check the compatibility of an object with a given type, and it returns the result as Boolean.

"as" operator is used for casting of an object to a type or a class.

44. What's a multicast delegate?

A delegate having multiple handlers assigned to it is called multicast delegate. Each handler is assigned to a method.

45. What are indexers in C# .NET?

Indexers are known as smart arrays in C#. It allows the instances of a class to be indexed in the same way as an array.

Eg: public int this[int index]   // Indexer declaration

46. What is difference between the "throw" and "throw ex" in .NET?

"Throw" statement preserves original error stack whereas "throw ex" have the stack trace from their throw point. It is always advised to use "throw" because it provides more accurate error information.

47. What are C# attributes and its significance?

C# provides developers a way to define declarative tags on certain entities, eg. Class, method, etc. are called attributes. The attribute's information can be retrieved at runtime using Reflection.

48. How to implement a singleton design pattern in C#?

In a singleton pattern, a class can only have one instance and provides an access point to it globally.

Eg:

```
Public sealed class Singleton
```

```
{
Private static readonly Singleton _instance = new Singleton();
}
```

49. What is the difference between directcast and ctype?

DirectCast is used to convert the type of object that requires the run-time type to be the same as the specified type in DirectCast.

Ctype is used for conversion where the conversion is defined between the expression and the type.

50. Is C# code is managed or unmanaged code?

C# is managed code because Common language runtime can compile C# code to Intermediate language.

51. What is Console application?

A console application is an application that can be run in the command prompt in Windows. For any beginner on .Net, building a console application is ideally the first step, to begin with.

52. Give an example of removing an element from the queue

The dequeue method is used to remove an element from the queue.

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DemoApplication
{
 class Program
 {
  static void Main(string[] args)
  {
   Queue qt = new Queue();
   qt.Enqueue(1);
   qt.Enqueue(2);
   qt.Enqueue(3);
  foreach (Object obj in qt)
  {
   Console.WriteLine(obj);
  }
   Console.WriteLine(); Console.WriteLine();
   Console.WriteLine("The number of elements in the Queue " + qt.Count);
   Console.WriteLine("Does the Queue contain " + qt.Contains(3));
   Console.ReadKey();
  }
 }
}
```

Ques 1 : ***Which of these statements correctly declares a two-dimensional array in C#?***

(A) int[][] myArray;

(B) int[,] myArray;

(C) System.Array[2] myArray;

(D) int[2] myArray;

Answer : int[,] myArray;

Ques 2 : ***Which of the following statements is correct about the .NET Framework?***

(A) .NET Framework uses COM+ services while creating Distributed Applications.

(B) .NET Framework uses DCOM for creating unmanaged applications.

(C) .NET Framework uses DCOM for making transition between managed and unmanaged code.

(D) .NET Framework uses DCOM for achieving language interoperability

Answer : .NET Framework uses DCOM for making transition between managed and unmanaged code.

Ques 3 : ***Which of following is correct***

(A) enum Day{Sunday= 01,Monday= 02,Tuesday= 03,Wednesday=04,Thursday= 05,Friday= 06,Saturday=07}

(B) Day{Sunday= 01,Monday= 02,Tuesday= 03,Wednesday=04,Thursday= 05,Friday= 06,Saturday=07}

(C) enumeration Day{Sunday= 01,Monday= 02,Tuesday= 03,Wednesday=04,Thursday= 05,Friday= 06,Saturday=07}

(D) Day enum{Sunday= 01,Monday= 02,Tuesday= 03,Wednesday=04,Thursday= 05,Friday= 06,Saturday=07}

Answer : enum Day{Sunday= 01,Monday= 02,Tuesday= 03,Wednesday=04,Thursday= 05,Friday= 06,Saturday=07}

Ques 4 : ***Which of the following is/are not true about the class?***

(A) Classes can have both contractors and destructors

(B) Classes are reference type

(C) Classes does not support Inheritance

(D) All of the above are true

Answer : Classes does not support Inheritance

Ques 5 : ***Boxing in .Net allows the user to convert***

(A) a double type to interger

(B) a interger type to double

(C) a value type to a reference type

(D) a reference type to a value type

Answer : a value type to a reference type

Ques 6 : ***If a method is marked as protected internal who can access it?***

(A) Classes within the same assembly, and classes derived from the declaring class.

(B) Internal methods can be only be called using reflection.

(C) Only methods that are in the same class as the method in question.

(D) Classes that are both in the same assembly and derived from the declaring class.

Answer : Classes within the same assembly, and classes derived from the declaring class.

Ques 7 : ***Which of the following is the root of the .NET type hierarchy?***

(A) System.Object

(B) System.Base

(C) System.Root

(D) System.Parent

Answer : System.Object

Ques 8 : ***In C#, by default structs are passed how?***

(A) By reference.

(B) By value.

(C) By address.

(D) By memory.

Answer : By value.

Ques 9 : ***Polymorphism?***

(A) Polymorphism means multiple forms and different name

(B) Polymorphism means multiple forms but one name

(C) Polymorphism means single forms and single name

(D) All of the above are true

Answer : Polymorphism means multiple forms but one name

Ques 10 : ***The RangeValidator control supports the following data types***

(A) Integer only

(B) Date, Integer and String

(C) only string

(D) Date and Integer

Answer : Date, Integer and String

Ques 11 : ***What is boxing?***

(A) Encapsulating a value type in an object.

(B) Encapsulating an object in a value type.

(C) Encapsulating a copy of a value type in an object.

(D) Encapsulating a copy of an object in a value type.

Answer : Encapsulating a copy of a value type in an object.

Ques 12 : ***Which of the following benefits do we get on running managed code under CLR?***

1. Type safety of the code running under CLR is assured.
2. It is ensured that an application would not access the memory that it is not authorized to access.
3. It launches separate process for every application running under it.
4. The resources are Garbage collected.

(A) Only 1, 2 and 4

(B) Only 2, 3 and 4

(C) Only 1 and 2

(D) All of the above

Answer : All of the above

Ques 13 : ***Which of the following characters ends every C# statement?***

(A) Period (.)

(B) Colon (:)

(C) Semicolon (;)

(D) Comma (,)

Answer : Semicolon (;)

Ques 14 : ***How to kill a user session explicitly?***

(A) Session.Discard()

(B) Session.Close()

(C) Session.Abandon()

(D) Session.kill()

Answer : Session.Abandon()

Ques 15 : ***What compiler switch creates an xml file from the xml comments in the files in an assembly?***

(A) /text

(B) /xml

(C) /doc

(D) /help

Answer : /doc

Ques 16 : ***Which of the following security features can .NET applications avail?***

1. PIN Security
2. Code Access Security
3. Role Based Security
4. Authentication Security
5. Biorhythm Security

(A) 2, 5

(B) 2, 3

(C) 1, 4, 5

(D) 3, 4

Answer : 2, 3

Ques 17 : ***You need to identify a type that meets the following criteria: Is always a number Is not greater than 65,535 Which type should you choose?***

(A) System.UInt16

(B) System.IntPtr

(C) Int

(D) System.String

Answer : System.UInt16

Ques 18 : ***Convert vs. Parse methods***

(A) Convert converts the value, Parse is for parsing

(B) Convert allows null values, Parse cannot

(C) Both are same

(D) None of these

Answer : Convert allows null values, Parse cannot

Ques 19 : ***What is a satellite Assembly?***

(A) A peripheral assembly designed to monitor permissions requests from an application.

(B) An assembly containing localized resources for another assembly.

(C) An assembly designed to alter the appearance or .skin. of an application.

(D) Any DLL file used by an EXE file.

Answer : An assembly containing localized resources for another assembly.

Ques 20 : ***Which of the following jobs are done by Common Language Runtime?***

1. It provides core services such as memory management, thread management, and remoting.
2. It enforces strict type safety.
3. It provides Code Access Security.
4. It provides Garbage Collection Services.

(A) Only 1 and 2

(B) Only 3, 4

(C) Only 2, 3 and 4

(D) All of the above

Answer : All of the above