- **Question 1. What Is Attribute In C#?**

it is access at compile time or run-time. Attributes are declare with a square brackets [] which is places above the elements.

```
[Obsolete(“Don’t use Old method, please use New method”, true)]
```

For example consider the bellow class. If we call the old method it will through error message.

```
public class myClass

{

   [Obsolete("Don't use Old method, please use New method", true)]

   public string Old() { return "Old"; }

   public string New() { return "New"; }

}

myClass omyClass = new myClass();

omyClass.Old();
```

- **Question 2. Why Attributes Are Used?**

In a program the attributes are used for adding metadata, like compiler instruction or other information (comments, description, etc).

- **Question 3. What Are The Types Of Attributes?**

The Microsoft .Net Framework **provides two types of attributes**: the pre-defined attributes and custom built attributes.

Pre-define attributes are three types:

AttributeUsage

Conditional

Obsolete

This marks a program that some entity should not be used.

- **Question 4. What Is Custom Attributes?**
- The Microsoft .Net Framework allows creating custom attributes that can be used to store declarative information and can be retrieved at run-time.
- **Question 5. What Is Reflection?**

Reflection is a process by which a computer program can monitor and modify its own structure and behavior.

```
public class MyClass
{
    public virtual int Add(int numb1, int numb2)
    {
        return numb1 + numb2;
    }
    public virtual int Subtract(int numb1, int numb2)
    {
        return numb1 - numb2;
    }
}
static void Main(string[] args)
{
    MyClass oMyClass = new MyClass();
    //Type information.
    Type oMyType = oMyClass.GetType();
    //Method information.
MethodInfo oMyMethodInfo = oMyType.GetMethod("Subtract");

Console.WriteLine("nType information:" + oMyType.FullName);

 Console.WriteLine("nMethod info:" + oMyMethodInfo.Name);

Console.Read();

}
```

- **Question 7. What Is Dynamic Keyword?**

The dynamic is a keyword which was introduced in .NET 4.0. Computer programming languages are two types: strongly typed and dynamically typed.

- In strongly types all types checks are happened at compile time
- in dynamic types all types of checks are happened at run time.

For example consider the following code

```
dynamic x = "c#";

x++;
```

It will not provide error at compile time but will provide error at run time.

- **Question 10. What Is Serialization?**

When we want to transport an object through network then we need to convert the object into a stream of bytes.

Serialization is a process to convert a complex objects into stream of bytes for storage (database, file, cache, etc) or transfer. Its main purpose is to save the state of an object.

De-serialization is the reverse process of creating an object from a stream of bytes to their original form.

- **Question 11. What Are The Types Of Serialization?**

The types of Serializations are given bellow:

1 Binary Serialization

2 SOAP Serialization

3 XML Serialization

- **Question 15. What Is Generics?**
- Generics are the most powerful features introduced in C# 2.0. It is a type-safe data structure that allows us to write codes that works for any data types.
- **Question 16. What Is A Generic Class?**
- A generic class is a special kind of class that can handle any types of data. We specify the data types during the object creations of that class.

```
public class Comparer

{

    public bool Compare(Unknown t1, Unknown t2)

  {

         if (t1.Equals(t2))

      {

            return true;

        }

        else

        {

            return false;

        }

    }

}

Comparer oComparerInt = new Comparer();
```

```
Console.WriteLine(oComparerInt.Compare(10, 10));

Comparer oComparerStr = new Comparer();

Console.WriteLine(oComparerStr.Compare("jdhsjhds", "10"));
```

- **Question 17. Why We Should Use Generics?**

It allows creating class, methods which are type-safe

It is faster. Because it reduce boxing/un-boxing

It increase the code performance

It helps to maximize code reuse, and type safety

- **Question 18. What Is Collections In C#?**

Arrays are most useful for creating and **working with a fixed number of strongly-typed objects.**

Collections are enhancement of array which **provides a more flexible way to work with groups of objects.**

Some commonly used collections under System.Collections namespace are given bellow:

- ArrayList
- SortedList
- Hashtable
- Stack
- Queue
- BitArray

- **Question 19. What Is Unsafe Code?**

In order to maintain security and type safety, C# does not support pointer generally. But by using unsafe keyword we can define an unsafe context in which pointer can be used.

The unsafe code or unmanaged code is a code block that uses a pointer variable. In the CLR, unsafe code is referred to as unverifiable code. In C#, the unsafe code is not necessarily dangerous. **The CLR does not verify its safety.**

- **Question 23. What Is Pointer?**

Pointer is a variable that stores the memory address of another variable. Pointers in C# have the same capabilities as in C or C++.

Some examples are given bellow:

```
int    *i    // pointer of an integer
float  *f    // pointer to a float
double *d    // pointer to a double
char   *ch   // pointer to a character
```

- **Question 24. Should I Use Unsafe Code In C#?**
- In C#, pointer is really used and Microsoft disengaged to use it. But there are some situations that require pointer. We can use pointer if required at our own risk.
- **Question 25. How Can We Sort The Elements Of The Array In Descending Order ?**

For This,First we call the Sort () method and then call Reverse() Methods.

- **Question 26. Can We Store Multiple Data Types In System.array ?**

No.

- **Question 28. What Is Difference Between String And Stringbuilder ?**

StringBuilder is more efficient than string.

String :- It is Immutable and resides within System Namespace.

StringBuilder:-It is mutable and resides System.Text Namespace.

- **Question 29. What Is Class Sortedlist Underneath?**

It is a Hash Table.

- **Question 30. What Is The .net Data Type That Allow The Retrieval Of Data By A Unique Key ?**

Hash Table

- **Question 31. Is Finally Block Get Executed If The Exception Is Not Occured ?**

Yes.

- **Question 32. Can Multiple Catch () Block Get Executed If The Exception Is Not Occured ?**

No,Once the proper catch code fires off ,the control is transferred to the finally block(if any),and the whatever follows the finally block.

- **Question 33. What Is Multicast Delegate ?**

The Multicast delegate is a delegate **that points to and eventually fires off several methods.**

- **Question 37. How Do We Inherit From A Class In C# ?**

In c#, we use a colon (:) and then the name of the base class.

- **Question 38. Does C# Support Multiple Inheritance ?**

No, we use interface for this purpose.

- **Question 39. Are Private Class -label Variables Inherited ?**

Yes, **but it is not accessible**.we generally know that they are inherited but not accessible.

- **Question 41. What Is The Top .net Class ?**

System.Object

- **Question 42. How Does Method Overloading Different From Overriding ?**
- A method overloading simply involves having a method with the **same name within the class.** whereas in method overriding we **can change method behaviour for a derived class.**
- **Question 43. Can We Override Private Virtual Method ?**

No.

- **Question 44. Can We Declare The Override Method Static While The Original Method Is Non Static ?**

No.

- **Question 45. Can We Prevent My Class From Being Inherited And Becoming A Base Class From The Other Classes ?**

Yes.

- **Question 46. What Is An Interface Class ?**

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

- **Question 47. Can We Inherit Multiple Interfaces ?**

Yes.

- **Question 48. Can We Allow Class To Be Inherited ,but Prevent The Method From Being Overridden ?**

Yes, first create class as public and make it's method sealed.

- **Question 49. What Is Signature Used For Overloaded A Method ?**
  - Use different data types
  - Use different number of parameters
  - Use different order of parameters
- **Question 50. What Is The Difference Between An Interface And Abstract Class ?**

In an interface, all methods must be abstract but in abstract class some methods can be concrete.