

C# INTERVIEW QUESTIONS AND ANSWERS

1. Is it possible to inline assembly or IL in C# code?

- No.

2. Is it possible to have different access modifiers on the get/set methods of a property?

- No. The access modifier on a property applies to both its get and set accessors. What you need to do if you want them to be different is make the property read-only (by only providing a get accessor) and create a private/internal set method that is separate from the property.

3. Is it possible to have a static indexer in C#? allowed in C#.

- No. Static indexers are not

4. If I return out of a try/finally in C#, does the code in the finally-clause run?

-Yes. The code in the finally always runs. If you return out of the try block, or even if you do a goto out of the try, the finally block always runs:
using System;

```
class main
{
    public static void Main()
    {
        try
        {
            Console.WriteLine("In Try block");
            return;
        }
        finally
        {
            Console.WriteLine("In Finally block");
        }
    }
}
```

Both In Try block and In Finally block will be displayed. Whether the return is in the try block or after the try-finally block, performance is not affected either way. The compiler treats it as if the return were outside the try block anyway. If it's a return without an expression (as it is above), the IL emitted is identical whether the return is inside or outside of the try. If the return has an expression, there's an extra store/load of the value of the expression (since it has to be computed within the try block).

5. I was trying to use an out int parameter in one of my functions. How should I declare the variable that I am passing to it?

You should declare the variable as an int, but when you pass it in you must specify it as 'out', like the following: int i; foo(out i); where foo is declared as follows:
[return-type] foo(out int o) { }

6. How does one compare strings in C#?

In the past, you had to call .ToString() on the strings when using the == or != operators to compare the strings' values. That will still work, but the C# compiler now automatically compares the values instead of the references when the == or != operators are used on string types. If you actually do want to compare references, it

can be done as follows: if ((object) str1 == (object) str2) {} Here's an example showing how string compares work:

using System;

```
public class StringTest
{
    public static void Main(string[] args)
    {
        Object nullObj = null; Object realObj = new
StringTest();
        int i = 10;
        Console.WriteLine(@"Null Object is [" + nullObj +
"\"]\n\"
            + @"Real Object is [" + realObj + "]" + "\n\"
            + @"i is [" + i + "]" + "\n\"");
        // Show string equality operators
        string str1 = @"foo\";
        string str2 = @"bar\";
        string str3 = @"bar\";
        Console.WriteLine(@"{0} == {1} ? {2}\", str1,
str2, str1 == str2 );
        Console.WriteLine(@"{0} == {1} ? {2}\", str2,
str3, str2 == str3 );
    }
}
```

Output:

```
Null Object is []
Real Object is [StringTest]
i is [10]
foo == bar ? False
bar == bar ? True
```

7. How do you specify a custom attribute for the entire assembly (rather than for a class)?

Global attributes must appear after any top-level using clauses and before the first type or namespace declarations. An example of this is as follows:

using System;

```
[assembly : MyAttributeClass] class X {}
```

Note that in an IDE-created project, by convention, these attributes are placed in AssemblyInfo.cs.

8. How do you mark a method obsolete?

```
[Obsolete] public int Foo() {...}
```

or

```
[Obsolete(@"This is a message describing why this method is obsolete")] public int
Foo() {...}
```

Note: The O in Obsolete is always capitalized.

9. How do you implement thread synchronization (Object.Wait, Notify, and CriticalSection) in C#?

You want the lock statement, which is the same as Monitor Enter/Exit:

```
lock(obj) { // code }
```

translates to

```
try {  
    CriticalSection.Enter(obj);  
    // code  
}  
finally  
{  
    CriticalSection.Exit(obj);  
}
```

10. How do you directly call a native function exported from a DLL?

Here's a quick example of the DllImport attribute in action:

```
using System.Runtime.InteropServices; \  
class C  
{  
    [DllImport(@"user32.dll\  
    public static extern int MessageBoxA(int h, string m,  
string c, int type);  
    public static int Main()  
    {  
        return MessageBoxA(0, @"Hello World!\  
@"Caption\  
    }  
}
```

This example shows the minimum requirements for declaring a C# method that is implemented in a native DLL. The method C.MessageBoxA() is declared with the static and external modifiers, and has the DllImport attribute, which tells the compiler that the implementation comes from the user32.dll, using the default name of MessageBoxA. For more information, look at the Platform Invoke tutorial in the documentation.

11. How do I simulate optional parameters to COM calls?

You must use the Missing class and pass Missing.Value (in System.Reflection) for any values that have optional parameters.

12. The C# keyword .int. maps to which .NET type?

1. System.Int16
2. System.Int32
3. System.Int64
4. System.Int128

13. Which of these string definitions will prevent escaping on backslashes in C#?

1. string s = #.n Test string.;
2. string s = ..n Test string.;
3. string s = @.n Test string.;

4. `string s = .n Test string.;`

14. Which of these statements correctly declares a two-dimensional array in C#?

1. `int[,] myArray;`
2. `int[][] myArray;`
3. `int[2] myArray;`
4. `System.Array[2] myArray;`

15. If a method is marked as protected internal who can access it?

1. Classes that are both in the same assembly and derived from the declaring class.
2. Only methods that are in the same class as the method in question.
3. Internal methods can be only be called using reflection.
4. Classes within the same assembly, and classes derived from the declaring class.

16. What is boxing?

1. Encapsulating an object in a value type.
2. Encapsulating a copy of an object in a value type.
3. Encapsulating a value type in an object.
4. Encapsulating a copy of a value type in an object.

17. What compiler switch creates an xml file from the xml comments in the files in an assembly?

1. `/text`
2. `/doc`
3. `/xml`
4. `/help`

18. What is a satellite Assembly?

1. A peripheral assembly designed to monitor permissions requests from an application.
2. Any DLL file used by an EXE file.
3. An assembly containing localized resources for another assembly.
4. An assembly designed to alter the appearance or .skin. of an application.

19. What is a delegate?

1. A strongly typed function pointer.
2. A light weight thread or process that can call a single method.
3. A reference to an object in a different process.
4. An inter-process message channel.

20. How does assembly versioning in .NET prevent DLL Hell?

1. The runtime checks to see that only one version of an assembly is on the machine at any one time.
2. .NET allows assemblies to specify the name AND the version of any assemblies they need to run.
3. The compiler offers compile time checking for backward compatibility.
4. It doesn't.

21. Which .Gang of Four. design pattern is shown below?

```
public class A {  
  
    private A instance;  
  
    private A() {  
  
    }  
  
    public  
    static A Instance {  
  
        get  
  
        {  
  
            if ( A == null )  
  
                A = new A();  
  
            return instance;  
  
        }  
  
    }  
  
}
```

1. Factory
2. Abstract Factory
3. Singleton
4. Builder

22. In the NUnit test framework, which attribute must adorn a test class in order for it to be picked up by the NUnit GUI?

1. TestAttribute
2. TestClassAttribute
3. TestFixtureAttribute
4. NUnitTestClassAttribute

23. Which of the following operations can you NOT perform on an ADO.NET DataSet?

1. A DataSet can be synchronised with the database.
2. A DataSet can be synchronised with a RecordSet.
3. A DataSet can be converted to XML.
4. You can infer the schema from a DataSet.

24. In Object Oriented Programming, how would you describe encapsulation?

1. The conversion of one type of object to another.
2. The runtime resolution of method calls.
3. The exposition of data.
4. The separation of interface and implementation.

25. What do you know about .NET assemblies?

Assemblies are the smallest units of versioning and deployment in the .NET application. Assemblies are also the building blocks for programs such as Web services, Windows services, serviced components, and .NET remoting applications.

26. What's the difference between private and shared assembly?

Private assembly is used inside an application only and does not have to be identified by a strong name. Shared assembly can be used by multiple applications and has to have a strong name.

27. What's a strong name?

A strong name includes the name of the assembly, version number, culture identity, and a public key token.

28. How can you tell the application to look for assemblies at the locations other than its own install?

Use the directive in the XML .config file for a given application.

```
< probing privatePath=c:\mylibs; bin\debug />
```

should do the trick. Or you can add additional search paths in the Properties box of the deployed application.

29. How can you debug failed assembly binds?

Use the Assembly Binding Log Viewer (fuslogvw.exe) to find out the paths searched.

30. Where are shared assemblies stored?

Global assembly cache.

31. How can you create a strong name for a .NET assembly?

With the help of Strong Name tool (sn.exe).

32. Where's global assembly cache located on the system?

Usually C:\winnt\assembly or C:\windows\assembly.

33. Can you have two files with the same file name in GAC?

Yes, remember that GAC is a very special folder, and while normally you would not be able to place two files with the same name into a Windows folder, GAC differentiates by version number as well, so it's possible for MyApp.dll and MyApp.dll to co-exist in GAC if the first one is version 1.0.0.0 and the second one is 1.1.0.0.

34. So let's say I have an application that uses MyApp.dll assembly, version 1.0.0.0. There is a security bug in that assembly, and I publish the patch, issuing it under name MyApp.dll 1.1.0.0. How do I tell the client applications that are already installed to start using this new MyApp.dll?

Use publisher policy. To configure a publisher policy, use the publisher policy configuration file, which uses a format similar app.config file. But unlike the app.config file, a publisher policy file needs to be compiled into an assembly and placed in the GAC.

35. What is delay signing?

Delay signing allows you to place a shared assembly in the GAC by signing the assembly with just the public key. This allows the assembly to be signed with the private key at a later stage, when the development process is complete and the component or assembly is ready to be deployed. This process enables developers to work with shared assemblies as if they were strongly named, and it secures the private key of the signature from being accessed at different stages of development.

36. Is there an equivalent of exit() for quitting a C# .NET application?

Yes, you can use System.Environment.Exit(int exitCode) to exit the application or Application.Exit() if it's a Windows Forms app.

37. Can you prevent your class from being inherited and becoming a base class for some other classes?

Yes, that is what keyword sealed in the class definition is for. The developer trying to derive from your class will get a message: cannot inherit from Sealed class WhateverBaseClassName. It is the same concept as final class in Java.

38. Is XML case-sensitive?

Yes, so and are different elements.

39. If a base class has a bunch of overloaded constructors, and an inherited class has another bunch of overloaded constructors, can you enforce a call from an inherited constructor to an arbitrary base constructor?

Yes, just place a colon, and then keyword base (parameter list to invoke the appropriate constructor) in the overloaded constructor definition inside the inherited class.

40. I was trying to use an "out int" parameter in one of my functions. How should I declare the variable that I am passing to it?

You should declare the variable as an int, but when you pass it in you must specify it as 'out', like the following:

```
int i;
```

```
foo(out i);
```

where foo is declared as follows:

```
[return-type] foo(out int o) { }
```

41. How do I make a DLL in C#?

You need to use the /target:library compiler option.

42. How do I simulate optional parameters to COM calls?

You must use the Missing class and pass Missing.Value (in System.Reflection) for any values that have optional parameters.

43. Will finally block get executed if the exception had not occurred?

Yes.

44. What is the C# equivalent of C++ catch (...), which was a catch-all statement for any possible exception? Does C# support try-catch-finally blocks?

Yes. Try-catch-finally blocks are supported by the C# compiler. Here's an example of a try-catch-finally block: using System;

```
public class TryTest
{
    static void Main()
    {
        try
        {
            Console.WriteLine("In Try block");
            throw new ArgumentException();
        }
        catch(ArgumentException n1)
        {
            Console.WriteLine("Catch Block");
        }
        finally
        {
            Console.WriteLine("Finally Block");
        }
    }
}
```

Output: In Try Block
Catch Block
Finally Block

If I return out of a try/finally in C#, does the code in the finally-clause run? Yes. The code in the finally always runs. If you return out of the try block, or even if you do a "goto" out of the try, the finally block always runs, as shown in the following

```
example: using System;
class main
{
    public static void Main()
    {
        try
        {
            Console.WriteLine("In Try block");
            return;
        }
        finally
        {
            Console.WriteLine("In Finally block");
        }
    }
}
```

Both "In Try block" and "In Finally block" will be displayed. Whether the return is in the try block or after the try-finally block, performance is not affected either way. The compiler treats it as if the return were outside the try block anyway. If it's a return without an expression (as it is above), the IL emitted is identical whether the return is

inside or outside of the try. If the return has an expression, there's an extra store/load of the value of the expression (since it has to be computed within the try block).

45. Is there regular expression (regex) support available to C# developers?

Yes. The .NET class libraries provide support for regular expressions. Look at the documentation for the System.Text.RegularExpressions namespace.

46. Is there a way to force garbage collection?

Yes. Set all references to null and then call System.GC.Collect(). If you need to have some objects destructed, and System.GC.Collect() doesn't seem to be doing it for you, you can force finalizers to be run by setting all the references to the object to null and then calling System.GC.RunFinalizers().

47. Does C# support properties of array types?

Yes. Here's a simple example: using System;

```
class Class1
{
    private string[] MyField;
    public string[] MyProperty
    {
        get { return MyField; }
        set { MyField = value; }
    }
}
class MainClass
{
    public static int Main(string[] args)
    {
        Class1 c = new Class1();
        string[] arr = new string[] { "apple", "banana" };
        c.MyProperty = arr;
        Console.WriteLine(c.MyProperty[0]); // "apple"
        return 0;
    }
}
```

48. What connections does Microsoft SQL Server support?

Windows Authentication (via Active Directory) and SQL Server authentication (via Microsoft SQL Server username and passwords).

49. What is a satellite assembly?

When you write a multilingual or multi-cultural application in .NET, and want to distribute the core application separately from the localized modules, the localized assemblies that modify the core application are called satellite assemblies.

50. How is method overriding different from overloading?

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

51. When do you absolutely have to declare a class as abstract (as opposed to free-willed educated choice or decision based on UML diagram)?

When at least one of the methods in the class is abstract. When the class itself is inherited from an abstract class, but not all base abstract methods have been overridden.

52. Why would you use untrusted verification?

Web Services might use it, as well as non-Windows applications.

53. What is the implicit name of the parameter that gets passed into the class set method?

Value, and its datatype depends on whatever variable we are changing.

54. How do I register my code for use by classic COM clients?

Use the regasm.exe utility to generate a type library (if needed) and the necessary entries in the Windows Registry to make a class available to classic COM clients. Once a class is registered in the Windows Registry with regasm.exe, a COM client can use the class as though it were a COM class.

55. How do I do implement a trace and assert?

Use a conditional attribute on the method, as shown below:

```
class Debug
{
    [conditional("TRACE")]
    public void Trace(string s)
    {
        Console.WriteLine(s);
    }
}
class MyClass
{
    public static void Main()
    {
        Debug.Trace("hello");
    }
}
```

In this example, the call to Debug.Trace() is made only if the preprocessor symbol TRACE is defined at the call site. You can define preprocessor symbols on the command line by using the /D switch. The restriction on conditional methods is that they must have void return type.

56. How do I create a multilanguage, multifile assembly?

Unfortunately, this is currently not supported in the IDE. To do this from the command line, you must compile your projects into netmodules (/target:module on the C# compiler), and then use the command line tool al.exe (alink) to link these netmodules together.

57. C# provides a default constructor for me. I write a constructor that takes a string as a parameter, but want to keep the no parameter one. How many constructors should I write?

Two. Once you write at least one constructor, C# cancels the freebie constructor, and now you have to write one yourself, even if there is no implementation in

58. What is the equivalent to regsvr32 and regsvr32 /u a file in .NET development?

Try using RegAsm.exe. The general syntax would be: RegAsm. A good description of RegAsm and its associated switches is located in the .NET SDK docs. Just search on "Assembly Registration Tool".

59. Explain ACID rule of thumb for transactions.

Transaction must be Atomic (it is one unit of work and does not depend on previous and following transactions), Consistent (data is either committed or rolled back, no in-between case where something has been updated and something hasn't), Isolated (no transaction sees the intermediate results of the current transaction), Durable (the values persist if the data had been committed even if the system crashes right after).

60. Where is the output of TextWriterTraceListener redirected?

To the Console or a text file depending on the parameter passed to the constructor.

61. How do I create a multilanguage, single-file assembly?

This is currently not supported by Visual Studio .NET.

62. Why cannot you specify the accessibility modifier for methods inside the interface?

They all must be public. Therefore, to prevent you from getting the false impression that you have any freedom of choice, you are not allowed to specify any accessibility, it is public by default.

63. Is it possible to restrict the scope of a field/method of a class to the classes in the same namespace?

There is no way to restrict to a namespace. Namespaces are never units of protection. But if you're using assemblies, you can use the 'internal' access modifier to restrict access to only within the assembly.

64. Why do I get a syntax error when trying to declare a variable called checked?

The word checked is a keyword in C#.

65. Why are there five tracing levels in System.Diagnostics.TraceSwitcher?

The tracing dumps can be quite verbose and for some applications that are constantly running you run the risk of overloading the machine and the hard drive there. Five levels range from None to Verbose, allowing to fine-tune the tracing activities.

66. What is the syntax for calling an overloaded constructor within a constructor (this() and constructorname() does not compile)?

The syntax for calling another constructor is as follows:

```
class B
{
    B(int i)
    { }
}
class C : B
{
    C() : base(5) // call base constructor B(5)
    { }
    C(int i) : this() // call C()
    { }
    public static void Main() {}
}
```

67. Why do I get a "CS5001: does not have an entry point defined" error when compiling?

The most common problem is that you used a lowercase 'm' when defining the Main method. The correct way to implement the entry point is as follows:

```
class test
{
    static void Main(string[] args) {}
}
```

68. What does the keyword virtual mean in the method definition?

The method can be over-ridden.

69. What optimizations does the C# compiler perform when you use the /optimize+ compiler option?

The following is a response from a developer on the C# compiler team:

We get rid of unused locals (i.e., locals that are never read, even if assigned).

We get rid of unreachable code.

We get rid of try-catch w/ an empty try.

We get rid of try-finally w/ an empty try (convert to normal code...).

We get rid of try-finally w/ an empty finally (convert to normal code...).

We optimize branches over branches:

```
gotoif A, lab1
```

```
goto lab2:
```

```
lab1:
```

```
turns into: gotoif !A, lab2
```

```
lab1:
```

We optimize branches to ret, branches to next instruction, and branches to branches.

70. How can I create a process that is running a supplied native executable (e.g., cmd.exe)?

The following code should run the executable and wait for it to exit before

continuing: using System;

using System.Diagnostics;

```
public class ProcessTest {
```

```
    public static void Main(string[] args) {
```

```
        Process p = Process.Start(args[0]);
```

```
        p.WaitForExit();
```

```
        Console.WriteLine(args[0] + " exited.");
```

```
    }
```

```
}
```

Remember to add a reference to System.Diagnostics.dll when you compile.

71. What is the difference between the System.Array.CopyTo() and System.Array.Clone()?

The first one performs a deep copy of the array, the second one is shallow.

72. How do I declare inout arguments in C#?

The equivalent of inout in C# is ref. , as shown in the following

example: public void MyMethod (ref String str1, out String str2)

```
{
```

```
...
```

```
}
```

When calling the method, it would be called like this: String s;

```
String s;
```

```
s1 = "Hello";
```

```
MyMethod(ref s1, out s2);
```

```
Console.WriteLine(s1);
```

```
Console.WriteLine(s2);
```

Notice that you need to specify ref when declaring the function and calling it.

73. Is there a way of specifying which block or loop to break out of when working with nested loops?

The easiest way is to use goto: using System;

```
class BreakExample
```

```
{
```

```
public static void Main(String[] args)
```

```
{
```

```
for(int i=0; i<3; i++)
```

```
{
```

```
Console.WriteLine("Pass {0}: ", i);
```

```
for( int j=0 ; j<100 ; j++ )
```

```
{
```

```
if ( j == 10) goto done;
```

```
Console.WriteLine("{0} ", j);
```

```
}
```

```
Console.WriteLine("This will not print");
```

```
}
```

```
done:
```

```
Console.WriteLine("Loops complete.");
```

```
}
```

```
}
```

74. What is the difference between const and static read-only?

The difference is that static read-only can be modified by the containing class, but const can never be modified and must be initialized to a compile time constant. To expand on the static read-only case a bit, the containing class can only modify it: -- in the variable declaration (through a variable initializer).

-- in the static constructor (instance constructors if it's not static).

75. What does the parameter Initial Catalog define inside Connection String?

The database name to connect to.

76. What is the difference between System.String and System.StringBuilder classes?

System.String is immutable; System.StringBuilder was designed with the purpose of having a mutable string where a variety of operations can be performed.

77. What is the top .NET class that everything is derived from?

System.Object.

78. Can you allow class to be inherited, but prevent the method from being over-ridden?

Yes, just leave the class public and make the method sealed

79. Can you change the value of a variable while debugging a C# application?

Yes, if you are debugging via Visual Studio.NET, just go to Immediate window.

80. Are private class-level variables inherited?

Yes, but they are not accessible, so looking at it you can honestly say that they are not inherited. But they are.

81. Can you inherit multiple interfaces?

Yes

82. From a versioning perspective, what are the drawbacks of extending an interface as opposed to extending a class?

With regard to versioning, interfaces are less flexible than classes. With a class, you can ship version 1 and then in version 2, decide to add another method. As long as the method is not abstract (i.e., as long as you provide a default implementation of the method), any existing derived classes continue to function with no changes. Because interfaces do not support implementation inheritance, this same pattern does not hold for interfaces. Adding a method to an interface is like adding an abstract method to a base class--any class that implements the interface will break, because the class doesn't implement the new interface method.

83. Which one is trusted and which one is untrusted?

Windows Authentication is trusted because the username and password are checked with the Active Directory, the SQL Server authentication is untrusted, since SQL Server is the only verifier participating in the transaction

84. What namespaces are necessary to create a localized application?

System.Globalization, System.Resources.

85. Does Console.WriteLine() stop printing when it reaches a NULL character within a string?

Strings are not null terminated in the runtime, so embedded nulls are allowed. Console.WriteLine() and all similar methods continue until the end of the string.

86. What is the advantage of using System.Text.StringBuilder over System.String?

StringBuilder is more efficient in the cases, where a lot of manipulation is done to the text. Strings are immutable, so each time it is being operated on, a new instance is created.

87. What are advantages and disadvantages of Microsoft-provided data provider classes in ADO.NET?

SQLServer.NET data provider is high-speed and robust, but requires SQL Server license purchased from Microsoft. OLE-DB.NET is universal for accessing other sources, like Oracle, DB2, Microsoft Access and Informix, but it is a .NET layer on top of OLE layer, so not the fastest thing in the world. ODBC.NET is a deprecated layer provided for backward compatibility to ODBC engines.

88. Why do I get a security exception when I try to run my C# app?

Some security exceptions are thrown if you are working on a network share. There are some parts of the frameworks that will not run if being run off a share (roaming profile, mapped drives, etc.). To see if this is what's happening, just move the executable over to your local drive and see if it runs without the exceptions. One of the common exceptions thrown under these conditions is System.Security.SecurityException.

To get around this, you can change your security policy for the intranet zone, code group 1.2 (the zone that running off shared folders falls into) by using the caspol.exe tool.

89. Is there any sample C# code for simple threading?

Some sample code follows: using System;

```

using System.Threading;
class ThreadTest
{
public void runme()
{
Console.WriteLine("Runme Called");
}
public static void Main(String[] args)
{
ThreadTest b = new ThreadTest();
Thread t = new Thread(new ThreadStart(b.runme));
t.Start();
}
}

```

90. What is the difference between // comments, /* */ comments and /// comments?

Single-line, multi-line and XML documentation comments.

91. What is the difference between and XML documentation tag?

Single line code example and multiple-line code example.

92. Explain the three services model (three-tier application). Presentation (UI), business (logic and underlying code) and data (from storage or other sources).

What are three test cases you should go through in unit testing? Positive test cases (correct data, correct output), negative test cases (broken or missing data, proper handling), exception test cases (exceptions are thrown and caught properly).

93. How do you inherit from a class in C#?

Place a colon and then the name of the base class. Notice that it is double colon in C++.

94. How do I port "synchronized" functions from Visual J++ to C#?

```

Original Visual J++ code: public synchronized void Run()
{
// function body
}
Ported C# code: class C
{
public void Run()
{
lock(this)
{
// function body
}
}
public static void Main() {}
}

```

95. Can I define a type that is an alias of another type (like typedef in C++)?

Not exactly. You can create an alias within a single file with the "using" directive: using System; using Integer = System.Int32; // alias

But you can't create a true alias, one that extends beyond the file in which it is declared. Refer to the C# spec for more info on the 'using' statement's scope.

96. Is it possible to have different access modifiers on the get/set methods of a property?

No. The access modifier on a property applies to both its get and set accessors. What you need to do if you want them to be different is make the property read-only (by only providing a get accessor) and create a private/internal set method that is separate from the property.

97. Is it possible to have a static indexer in C#?

No. Static indexers are not allowed in C#.

98. Does C# support# define for defining global constants?

No. If you want to get something that works like the following C code:

```
#define A 1
```

use the following C# code: class MyConstants

```
{  
    public const int A = 1;  
}
```

Then you use MyConstants.A where you would otherwise use the A macro.

Using MyConstants.A has the same generated code as using the literal 1.

100. Does C# support templates?

No. However, there are plans for C# to support a type of template known as a generic. These generic types have similar syntax but are instantiated at run time as opposed to compile time. You can read more about them here.

101. Does C# support parameterized properties?

No. C# does, however, support the concept of an indexer from language spec. An indexer is a member that enables an object to be indexed in the same way as an array. Whereas properties enable field like access, indexers enable array-like access. As an example, consider the Stack class presented earlier. The designer of this class may want to expose array-like access so that it is possible to inspect or alter the items on the stack without performing unnecessary Push and Pop operations. That is, Stack is implemented as a linked list, but it also provides the convenience of array access.

Indexer declarations are similar to property declarations, with the main differences being that indexers are nameless (the name used in the declaration is this, since this is being indexed) and that indexers include indexing parameters. The indexing parameters are provided between square brackets.

102. Does C# support C type macros?

No. C# does not have macros. Keep in mind that what some of the predefined C macros (for example, __LINE__ and __FILE__) give you can also be found in .NET classes like System.Diagnostics (for example, StackTrace and StackFrame), but they'll only work on debug builds.

103. Can you store multiple data types in System.Array?

No.

104. Is it possible to inline assembly or IL in C# code?

No.

105. Can you declare the override method static while the original method is non-static?

No, you cannot, the signature of the virtual method must remain the same, only the keyword virtual is changed to keyword override

106. Does C# support multiple inheritance?

No, use interfaces instead.

107. Can multiple catch blocks be executed?

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

108. Can you override private virtual methods?

No, moreover, you cannot access private methods in inherited classes, have to be protected in the base class to allow any sort of access.

109. What is a pre-requisite for connection pooling?

Multiple processes must agree that they will share the same connection, where every parameter is the same,

110. What is the data provider name to connect to Access database?

Microsoft.Access.

111. Why does my Windows application pop up a console window every time I run it?

Make sure that the target type set in the project properties setting is set to Windows Application, and not Console Application. If you're using the command line, compile with /target:winexe & not target:exe.

112. What is the wildcard character in SQL?

Let us say you want to query database with LIKE for all employees whose name starts with La. The wildcard character is %, the proper query with LIKE would involve La%.

113. What is the role of the DataReader class in ADO.NET connections?

It returns a read-only dataset from the data source when the command is executed

114. What does the This window show in the debugger?

It points to the object that is pointed to by this reference. Object's instance data is shown.

115. Describe the accessibility modifier protected internal?

It is available to derived classes and classes within the same Assembly (and naturally from the base class it is declared in).

116. What is an interface class?

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

117. What is a multicast delegate?

It is a delegate that points to and eventually fires off several methods.

118. How does one compare strings in C#?

In the past, you had to call .ToString() on the strings when using the == or != operators to compare the strings' values. That will still work, but the C# compiler now automatically compares the values instead of the references when the == or != operators are used on string types. If you actually do want to compare references, it can be done as follows: if ((object) str1 == (object) str2) { ... } Here's an example showing how string compares work: using System;

```
public class StringTest
{
    public static void Main(string[] args)
    {
        Object nullObj = null;
        Object realObj = new StringTest();
        int i = 10;
        Console.WriteLine("Null Object is [" + nullObj + "]" +
            "Real Object is [" + realObj + "]" +
            "i is [" + i + "]");
        // Show string equality operators
        string str1 = "foo";
        string str2 = "bar";
        string str3 = "bar";
        Console.WriteLine("{0} == {1} ? {2}", str1, str2, str1 == str2
        );
        Console.WriteLine("{0} == {1} ? {2}", str2, str3, str2 == str3
        );
    }
}

Output: Null Object is []
Real Object is [StringTest]
i is [10]
foo == bar ? False
bar == bar ? True
```

119. What does assert() do?

In debug compilation, assert takes in a Boolean condition as a parameter, and shows the error dialog if the condition is false. The program proceeds without any interruption if the condition is true.

120. How do I get deterministic finalization in C#?

In a garbage collected environment, it's impossible to get true determinism. However, a design pattern that we recommend is implementing IDisposable on any class that contains a critical resource. Whenever this class is consumed, it may be placed in a using statement, as shown in the following example:

```
using(FileStream myFile = File.Open(@"c:temptest.txt",
    FileMode.Open))
{
    int fileOffset = 0;
    while(fileOffset < myFile.Length)
    {
        Console.Write((char)myFile.ReadByte());
        fileOffset++;
    }
}
```

When myFile leaves the lexical scope of the using, its dispose method will be called.

121. How can I get around scope problems in a try/catch?

If you try to instantiate the class inside the try, it'll be out of scope when you try to access it from the catch block. A way to get around this is to do the following:

```

Connection conn = null;
try
{
    conn = new Connection();
    conn.Open();
}
finally
{
    if (conn != null) conn.Close();
}

```

By setting it to null before the try block, you avoid getting the CS0165 error (Use of possibly unassigned local variable 'conn').

122. Why do I get an error(CS1006) when trying to declare a method without specifying a return type?

If you leave off the return type on a method declaration, the compiler thinks you are trying to declare a constructor. So if you are trying to declare a method that returns nothing use void. The following is an example: // This results in a CS1006 error

```

public static staticMethod (mainStatic obj) // This will work as wanted
public static void staticMethod (mainStatic obj)

```

123. How do I convert a string to an int in C#?

```

Here's an example: using System;
class StringToInt
{
    public static void Main()
    {
        String s = "105";
        int x = Convert.ToInt32(s);
        Console.WriteLine(x);
    }
}

```

124. How do you directly call a native function exported from a DLL?

Here's a quick example of the DllImport attribute in action: using System.Runtime.InteropServices;

```

class C
{
    [DllImport("user32.dll")]
    public static extern int MessageBoxA(int h, string m, string c, int type);
    public static int Main()
    {
        return MessageBoxA(0, "Hello World!", "Caption", 0);
    }
}

```

This example shows the minimum requirements for declaring a C# method that is implemented in a native DLL. The method C.MessageBoxA() is declared with the static and external modifiers, and has the DllImport attribute, which tells the compiler that the implementation comes from the user32.dll, using the default name of MessageBoxA. For more information, look at the Platform Invoke tutorial in the documentation.

125. What is the .NET datatype that allows the retrieval of data by a unique key?

HashTable.

126. What is class SortedList underneath?

127. How do you specify a custom attribute for the entire assembly (rather than for a class)?

Global attributes must appear after any top-level using clauses and before the first type or namespace declarations. An example of this is as follows:

using System;

[assembly : MyAttributeClass]

class X {}

Note that in an IDE-created project, by convention, these attributes are placed in AssemblyInfo.cs.

128. What is the difference between a struct and a class in C#?

From language spec:

The list of similarities between classes and structs is as follows. Longstructs can implement interfaces and can have the same kinds of members as classes. Structs differ from classes in several important ways; however, structs are value types rather than reference types, and inheritance is not supported for structs. Struct values are stored on the stack or in-line. Careful programmers can sometimes enhance performance through judicious use of structs. For example, the use of a struct rather than a class for a Point can make a large difference in the number of memory allocations performed at runtime. The program below creates and initializes an array of 100 points. With Point implemented as a class, 101 separate objects are instantiated-one for the array and one each for the 100 elements

129. What is the difference between the Debug class and Trace class?

Documentation looks the same. Use Debug class for debug builds, use Trace class for both debug and release builds.

130. How can you overload a method?

Different parameter data types, different number of parameters, different order of parameters.

131. What debugging tools come with the .NET SDK?

CorDBG - command-line debugger, and DbgCLR - graphic debugger. Visual Studio .NET uses the DbgCLR. To use CorDbg, you must compile the original C# file using the /debug switch.

132. What does Dispose method do with the connection object?

Deletes it from the memory.

133. How do you generate documentation from the C# file commented properly with a command-line compiler?

Compile it with a /doc switch.

134. When you inherit a protected class level variable, who is it available to?

Classes in the same namespace.

135. How can I get the ASCII code for a character in C#?

Casting the char to an int will give you the ASCII value: char c = 'f';
System.Console.WriteLine((int)c); or for a character in a string:

System.Console.WriteLine((int)s[3]); The base class libraries also offer ways to do this with the Convert class or Encoding classes if you need a particular encoding.

136. How do I create a Delegate/MulticastDelegate?

C# requires only a single parameter for delegates: the method address. Unlike other languages, where the programmer must specify an object reference and the method to invoke, C# can infer both pieces of information by just specifying the method's name. For example, let's use System.Threading.ThreadStart: Foo MyFoo = new Foo(); ThreadStart del = new ThreadStart(MyFoo.Baz); This means that delegates can invoke static class methods and instance methods with the exact same syntax!

137. Is there an equivalent to the instanceof operator in Visual J++?

C# has the is operator:
expr is type

138. How do destructors and garbage collection work in C#?

C# has finalizers (similar to destructors except that the runtime doesn't guarantee they'll be called), and they are specified as follows:

```
class C
{
    ~C()
    {
        // your code
    }
    public static void Main() {}
}
```

Currently, they override object.Finalize(), which is called during the GC process.

139. My switch statement works differently! Why?

C# does not support an explicit fall through for case blocks. The following code is not legal and will not compile in C#:

```
switch(x)
{
    case 0:
        // do something
    case 1:
        // do something in common with 0
    default:
        // do something in common with
        //0, 1 and everything else
    break;
}
```

To achieve the same effect in C#, the code must be modified as shown below (notice how the control flows are explicit):

```
class Test
{
    public static void Main()
    {
        int x = 3;
        switch(x)
        {
            case 0:
                // do something
```

```

goto case 1;
case 1:
// do something in common with 0
goto default;
default:
// do something in common with 0, 1, and anything else
break;
}
}
}

```

140. How can I access the registry from C# code?

By using the Registry and RegistryKey classes in Microsoft.Win32, you can easily access the registry. The following is a sample that reads a key and displays its value:

```

using System;using Microsoft.Win32;
class regTest
{
public static void Main(String[] args)
{
RegistryKey regKey;
Object value;
regKey = Registry.LocalMachine;
regKey =
regKey.OpenSubKey("HARDWAREDESCRIPTIONSystemCentralProcessor ");
value = regKey.GetValue("VendorIdentifier");
Console.WriteLine("The central processor of this machine is: {0}.", value);
}
}

```

141. How can you sort the elements of the array in descending order?

By calling Sort() and then Reverse() methods.

142. How do you debug an ASP.NET Web application?

Attach the aspnet_wp.exe process to the DbgClr debugger.

143. How do you mark a method obsolete?

```

Assuming you've done a "using System;": [Obsolete]
public int Foo() {...}
or [Obsolete("This is a message describing why this method is obsolete")]
public int Foo() {...}

```

Note: The O in Obsolete is capitalized.

144. How is the DLL Hell problem solved in .NET?

Assembly versioning allows the application to specify not only the library it needs to run (which was available under Win32), but also the version of the assembly

145. What are the ways to deploy an assembly?

An MSI installer, a CAB archive, and XCOPY command.

146. Why does DllImport not work for me?

All methods marked with the DllImport attribute must be marked as public static extern.

147. What is a delegate?

A delegate object encapsulates a reference to a method. In C++ they were referred to as function pointers.

148. What is the difference between an interface and abstract class?

In the interface all methods must be abstract; in the abstract class some methods can be concrete. In the interface no accessibility modifiers are allowed, which is ok in abstract classes.

150. What is an abstract class?

A class that cannot be instantiated. A concept in C++ known as pure virtual method. A class that must be inherited and have the methods over-ridden. Essentially, it is a blueprint for a class without any implementation.