Here is a list of questions and answers from the sources, with question numbers as requested:

1. **What is C#?** C# is a high-level object-oriented programming language. It is used for building secure and robust applications.
2. **Why was C# invented?** Developed by Microsoft in 2000, C# was invented to match the increasing demand for web applications.
3. **What are the benefits of using C#?** Some top reasons to use C# are: Easy to learn, Fast development time, High scalability, Compiled on multiple computer platforms, and Modularity for easier troubleshooting allows developers to work on multiple objects simultaneously.
4. **Can you name the types of comments in C#?** There are two types of comments in C#: Single line (//) and Multiple line (/* */).
5. **Can you name a few IDEs given by Microsoft for C# development?** There are several IDEs for C# development: Visual Studio, Visual Studio Code, and Visual Studio Express.
6. **What does the acronym CLR stand for?** Common Language Runtime (CLR).
7. **Can we execute multiple catch blocks in C# program for one exception?** No. You can't use multiple catch blocks for same exception in C# because a catch block is preceded by a try block.
8. **What is the difference between C# and C programming language?** C# supports object-oriented programming whereas C supports procedural programming.
9. **What is .Net CLR equivalent to?** Java Virtual Machine (JVM).
10. **What does the acronym SOAP stand for?** Simple Object Access Protocol.
11. **What is Common Language Runtime (CLR)?** The CLR is a virtual machine component of the .NET Framework. It manages the code execution of .NET programs.
12. **What are indexers?** Indexers allow objects to be indexed just like arrays.
13. **Can you name the types of classes in C#?** There are mainly four types of classes in C#: Abstract class, Partial class, Sealed class, and Static class.
14. **Does C# support multiple inheritances?** No.
15. **Is C++ the same as C#?** No. C# is a high level programming language whereas C++ is a low level programming language. Another difference is C# compiles to CLR whereas C++ compiles to machine code.
16. **Can you tell us the extension of a C# language file?** ".cs" is used to save C# files.
17. **Can you tell us the symbols used to mark the start and end of a code block?** Curly braces {}.

18. **Define operators.** Operators are a set of symbols that tells the compiler to perform an action.
19. **Is C# a type-safe programming language?** Yes, it is a type-safe programming language.
20. **Is it possible to get the array index using the for each loop?** No, it is not possible to get the array index using the for each loop. To access the array index, you need to use a standard for loop.
21. **Name the keyword used to come out from the loop.** Break.
22. **Can you inherit a class into another class?** Yes, it is possible to inherit a class into another. It is of two types: Derived class-child and Base class-parent.
23. **Can override of a function be possible in the same class?** No. Method overriding is a process of calling functions from base class in the derived class. So overriding is not possible in the same class.
24. **What do you call a subroutine in C#?** Method.
25. **Name some members of namespaces in C#.** Namespaces delegates, interfaces, and structures can be the members of the namespace.
26. **Can you tell us if a private virtual method can be overridden?** No, you can't as private virtual methods can't be accessed outside the class.
27. **What is the symbol used to terminate a C#?** Every statement in C# is terminated by a semicolon (;).
28. **Is C# case-sensitive?** Yes.
29. **Can you use a "this" command within a static method?** No.
30. **Which symbol is used to mark the beginning of a single-line comment in C#?** //.
31. **What are the symbols for a multi-line comment?** /* is used to begin and */ to end the comment.
32. **What's the statement to declare a variable in C#?** type variableName = value;.
33. **Which data type should be used to store text value?** String.
34. **What's the syntax to define a constant?** const type constant_name = value;.
35. **What's a 'Console' in C#?** Class.
36. **Can you use foreach loop in C#?** Yes.
37. **What do you mean by throw statement in C#?** The throw statement allows you to manually throw an exception during the execution of a program.
38. **Can you name the class from which data type UInt is derived in C#?** System.UInt32.

39. **Can you tell which access specifier in C# should be used for the Main() method?** Public. As the Main() method is called by the runtime, it should be defined as public.
40. **What's the use of the C# pointer?** A C# pointer allows the user to store the memory address of another type.
41. **Which symbol is used to access variables/fields inside a class?** (.) symbol or dot operator.
42. **Define a variable in C#.** Variables are containers used to store data values. We can change the value or reuse the variable as many times as we like.
43. **How do you do Exception Handling in C#?** The following four keywords are used for Exception Handling in C#: Try, Catch, Finally, and Throw.
    - **Try** - The try block recognizes which block of code has particular exceptions activated.
    - **Catch** - The catch keyword signifies a program for catching an exception using an exception handler.
    - **Finally** - The finally block executes a given block of code whether or not an exception is caught.
    - **Throw** - Using the throw keyword, the program throws an exception in the event of a problem.
44. **What's Recursion in C#?** Recursion refers to the process of making a function call itself.
45. **What is a multicasting delegate?** Multicasting delegates allow users to invoke multiple callbacks. It can refer to multiple methods and functions having the same signature at one time.
46. **What are Namespaces in C#?** Namespaces are used for differentiating one set of names different from another. They are used to organize code in distinct groups so that one group can be differentiated from another.
47. **List the steps of code compilation in C#.** Here are the four steps: Pre-processing, Compiling, Assembling, and Linking.
48. **Can you name some access modifiers available in C#?** Public, Private, Protected, Internal, and Protected Internal.
49. **Which parameter can be used to return multiple values from a function?** Reference or output parameters can be used to return multiple values from a function.
50. **Define Abstract class in C#.** Abstract class acts as a base class and doesn't have its own objects. It can't be used for creating objects.
51. **What's the by default, default interface method in C#?** Virtual.

52. **What's Polymorphism in C#?** It's the ability of an object to take different forms and behave differently in different cases. It is of two types: Compile time polymorphism and Runtime polymorphism.
53. **What's the role of the access modifier in C#?** The access modifiers are used to define the visibility of classes, methods, properties, and fields.
54. **Name the different types of constructors in C#.** There are five types of constructors in C#: Static constructor, Default constructor, Private constructor, Copy constructor, and Parameterized constructor.
55. **What's the syntax for declaring an object of the class?** Class_Name Object_Name = new Class_Name();.
56. **Which symbol is used to define the variable type when declaring an array?** [ ].
57. **What's String.Length in C#?** String.Length is used to get the count of characters present in a given string. It is a property of the System.String class.
58. **Name the string class operator used to decide whether two given strings have different values.** The inequality operator (!=). The syntax is: public static bool operator != (string? x, string? y);.
59. **What is the difference between object type variables and dynamic type variables in C#?** Dynamic and object type variables are similar in function. Object type variables type checks during the compile time, whereas dynamic type variables at run time.
60. **Name the different ways in which you can pass parameters to a method in C#.** There are three ways of passing parameters: Value parameters, Reference parameters, and Output parameters.
61. **How can you run an infinite loop using the for () statement?** Using for(;;).
62. **Name all data types present in C#.** There are four basic data types: Char, Int, Float, and Double.
63. **Define Keywords in C#?** Keywords are reserved words that have some predefined actions. They are special words that hold special meaning to the compiler.
64. **What are jagged arrays?** A jagged array, also called an array of arrays, is a multidimensional array that consists of other arrays of different sizes.
65. **Define a local variable in C#.** Local variables are referred to as variables that are defined in a code block. They are only visible in the code block they're declared in.
66. **Define Inheritance in C#.** Inheritance is a way of defining a class (child class) that is allowed to inherit the behavior of the other class (parent class).
67. **Why does C# not support multiple inheritances?** C# does not support multiple inheritance because of Name collision.
68. **Which keyword is used to implement duck typing?** Dynamic.

69. **What is a read only variable?** Read only variables are created using the readonly keyword. Its value can be modified only within a constructor.
70. **Can you change the value of a variable while debugging an application?** Yes, the values of variables can be altered during debugging.
71. **What is LINQ in C#?** Language-Integrated Query (LINQ) is a .NET Framework. It is used to retrieve information from different kinds of sources.
72. **What are the constructors?** Constructor is a method that gets executed when a new class object is created. It can be public or private.
73. **Can you tell us the difference between a constant from a read-only?** Read-only is a runtime constant. Const is a compile-time constant.
74. **What is method overloading?** Method overloading is a method of having two or more methods in the same scope with the same name but different parameter lists.
75. **What are the features of read-only variables?** The features of read-only variable are as follows: Initialized at runtime, Can be used with static modifiers, Only declared at the class level.
76. **Define dynamic type variables in C#.** Dynamic type variable was introduced in C# 4.0. It is used to skip type checking at compile-time. It is created using dynamic keywords. You can store any type of value in a dynamic variable.
77. **What is a "using" statement in C#?** Using a statement ensures the object is disposed of as soon as it goes out of scope without needing to write any code.
78. **Define nullable types in C#.** Nullable types allow you to assign a normal range to null values. You can also assign true or false to null types. The syntax is: < data_type> ? = null;.
79. **Can you tell us something about the stream reader and stream writer class in C#?** Stream reader and stream writer classes are used for reading and writing actions to a file. Both are inherited from the abstract base class stream.
80. **Can you tell the difference between overloading and overriding? Overloading** - When you have two or more methods in the same scope with the same name but different parameters. **Overriding** - It allows you to change the behavior of a method in a subclass or child class.
81. **Define file handling in C#.** File handling refers to the management of files. It consists of different actions like creating the file, writing to the file, reading from the file, etc. Read and write are the two operations used in file handling.
82. **Explain Boxing and Unboxing.** Both Boxing and Unboxing are used for converting the type. **Boxing** - Boxing converts the value type to the object or to the data type of an interface implemented by this particular value type. The CLR boxes a value, in other words, converts the value type to an object. For this, CLR wraps the value in System.

Object and store it in the heap area within the domain of the application. **Unboxing** - Unboxing extracts the value type from the object or any interface type that has been implemented. For Boxing, implicit code may be used, but for Unboxing explicit code must be used. Boxing and Unboxing highlight that C# has a unified view of the type system, meaning that all value types can be treated as objects.

83. **Differentiate between managed and unmanaged code Managed Code** - Managed Code is developed within the .NET framework. CLR directly executes such code by using managed code execution. Any language written in the .NET framework is considered to be managed code. **Unmanaged Code** - Unmanaged code is any code developed outside the .NET framework. Unmanaged applications are not executed by CLR. Some languages like C++ can write unmanaged applications such as an application for accessing the low-level functions of the operating system. Some examples of unmanaged code include background compatibility with the code of VB, ASP, and COM.

84. **Differentiate between Struct and Class in C#.** Class and struct both are user-defined data types. **Struct** - Struct is a value type in C# that inherits values from System.Value. It is mostly used for small quantities of data. It cannot be inherited to any other type. A Struct cannot have abstract values. **Class** - Class is a reference type in C#. Since it refers to objects, it inherits from System.Object. Classes are mostly used for large quantities of data. Classes can be inherited to other classes. Classes can have abstract values. A default constructor can be created for classes.

85. **What is the difference between Task and Thread in C#?** Task is an object used in the Task Parallel Library (TPL) to represent an asynchronous operation, while a Thread is a separate path of execution in a program. Tasks are a higher level of abstraction than threads and are used to manage the execution of code in parallel. Tasks are easier to use and manage than threads, and they can also be used to provide more efficient resource utilization. Threads on the other hand, provide a lower level of abstraction and are used to execute code directly in the processor.

86. **How is encapsulation done in C#?** Encapsulation, in C#, is implemented by using access specifiers. A class member's scope and visibility are defined by these access specifiers. With public access specifiers, a class can expose its member variables and functions to other objects and functions. Once a member is public, it can be reached from outside the class. With private access specifiers, a class can hide its member variables and functions from other objects and functions. The private members of a class can be accessed only by functions of the same class. Even instances of the same class do not have access to its private members. Protected access specifiers are similar to private access specifiers because they cannot be accessed outside the class. However, protected class members can be accessed by any subclass of that class as well. This enables implementing inheritance.

87. **What is a Destructor in C# and when is it used?** A destructor is a special method in C# that is automatically called when an object is destroyed. It is used to free up any

resources that the object may have been using, such as memory or files. Destructors are usually implemented in a class and are denoted by the keyword ~ followed by the class name. For example, if a class called MyClass was to have a destructor, it would be declared as follows: ~MyClass().

88. **For methods inside the interface, why can you not specify the accessibility modifier?** Virtual methods in an interface have no method definition. The methods here are written to be overridden in the derived class and hence, they are publicly available.

89. **Differentiate between ref and out keywords.** The main difference between ref and out keywords in C# is that **ref requires that the variable be initialized before being passed to the method** whereas **out keyword doesn't require the variable to be initialized before being passed to the method**.

90. **Why is finally block used in C#?** The finally block always gets executed if there is an exception or not. When the code is executed in the try block and an exception occurs, control returns to the catch block, and in the end, the finally block gets executed. The finally block therefore can contain closing connections to the database and the release of file handlers.

20.

Write a program in C# to reverse a string?

Hide Answer

```
public class Program
{
   static void Main(string[] args)
   {
      string str;

      Console.Write("Enter a string : ");
      str = Console.ReadLine();

      char[] arr = str.ToCharArray();

      Array.Reverse(arr);

      Console.Write("Reversed string is : ");
      foreach (char ch in arr)
      {
         Console.Write(ch);
      }
   }
}
```

21.

Write a program in C# to reverse the order of the given words?

Hide Answer

```
public class Program
{
   public static void Main()
   {
      string s = "Hello World";
      string[] words = s.Split(' ');

      Console.Write("Reversed order of the given words: ");
      for (int i = words.Length - 1; i >= 0; i--)
      {
         Console.Write(words[i] + " ");
      }
   }
}

// Output: Reversed order of the given words: World Hello
```

22.

Write a program in C# to find if a given string is palindrome or not?

Hide Answer

```
internal static void chkPalindrome(string str)
{
  bool flag = false;
  for (int a = 0, b = str.Length - 1; a < str.Length / 2; a++, b--)
  {
     if (str[a] != str[b])
     {
        flag = false;
        break;
     }
     else
        flag = true;
  }
  if (flag)
  {
     Console.WriteLine("The entered string is palindrome");
  }
  else
     Console.WriteLine("The entered string is not palindrome");
```

23.

Write a C# program to find the substring from a given string?

Hide Answer

```
internal static void findallsubstring(string str)
{
  for (int a = 0; a < str.Length; ++a)
  {
     StringBuilder subString = new StringBuilder(str.Length - a);
     for (int b = a; b < str.Length; ++b)
     {
        subString.Append(str[b]);
        Console.Write(subString + " ");
     }
  }
}
```

24.

Write a program to remove duplicate characters from a string?

Hide Answer

```
internal static void removeduplicate(string str)
{
    string result = string.Empty;

    for (int a = 0; a < str.Length; a++)
    {
        if (!result.Contains(str[a]))
        {
            result += str[a];
        }
    }
    Console.WriteLine(result);
}
```

25.

Write a program to find the sum of digits of a positive integer?

Hide Answer

```
internal static void sumofnum (int num)
{
    int sum = 0;
    while (num > 0)
    {
        sum += num % 10;
        num /= 10;
    }
    Console.WriteLine(sum);
}
```

26.

Write a program to check whether the entered number is not divisible by 3 and 7?

Hide Answer

```
  static void Main()
   {
      Console.WriteLine("Please enter your number: ");
      int n = int.Parse(Console.ReadLine());

      for (int a = 1; a <= n; a++)
      {
         if (a % 3 == 0)
         {
            continue;
         }
         else if (a % 7 ==0)
         {
            continue;
         }
         Console.Write("{0} ", a);
      }
      Console.WriteLine();
   }
}
```

27.

What will be the output of the following C# code?

```
static void Main(string[] args)
{
   int x, y, z, r;
   x = 80;
   y = 30;
   z = 5;
   r = x - y / 3 + z * 2 - 1;
   Console.WriteLine(r);
   Console.ReadLine();
}
```

Hide Answer

Output:

79

28.

What will be the output of the following code?

```
using System;
public class multiply{
   public static void Main(string[] args) {
      int num1=25;
      int num2=20;
      int mul=num1*num2;
      Console.WriteLine ("The multiplication of two numbers: "+mul); } }}
```

Hide Answer

Output:

500

29.

What will be the output of the following code?

```
int a = 4;
int b = 7;
int sum = a + b;
Console.WriteLine(sum);
```

Hide Answer

Output:

11