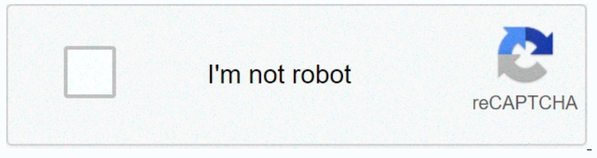

Continue

C language basic mcq questions and answers

Catch up on the latest daily buzz with the daily buzzFeed newsletter! Basic C# interview questions commonly used on programming and coding:C# is a fast-growing programming language that is widely used. It is highly sought after, versatile and also supports cross platforms. It is not only used for Windows but many other operating systems. Therefore, it is very important to gain a strong understanding of this language and land in any work in the software testing industry. Below you have enlisted not only a set of the most frequently asked questions of C# but also some very important issues to be understood to stand out from the audience of the C# population. As C# is a huge issue, for the ease of handling all concepts, I divided this topic into three parts as mentioned below: questions about basic conceptsSpeed questions about arrays and StringsAdvanced ConceptsThrest article includes a set of top 50 C# interview questions and answers covering almost all of its important topics in simple terms, in order to help you prepare for your interview. The most popular C# interview questions and answers concepts #1) What is object and status? Answer: A class is an enquirer of properties and methods used to represent an another in real time. This is a data structure that consolidates all instances in one unit. An object is defined as a class appearance. Q#2) What are basic OOP concepts? Answer: The four basic concepts of object-oriented programming are: an ankipsycholysation: here, the internal representation of an object is hidden from view outside the object definition. Only the required information can be accessed while the rest of the data application is hidden. Abstraction: This is the process of identifying the critical behavior and data of an object and eliminating irrelevant details. Inheritance: This is the ability to create new departments from another meth. This is done by accessing, modifying, and expanding the behavior of objects in the parent class. Polymorphism: the meaning of the name, one name, many forms. It is achieved by having multiple methods with the same name but different applications. Q#3) What is managed and unmanaged code? Answer: Managed code is code executed by CLR (common language runtime) meaning that each application code is based on the .Net platform. It is considered managed because of a .Net framework that uses the garbage collector internally to clean up the memory that is not in use. Unmanaged code is any code executed by an application runtime of any framework other than .Net. The application runtime will handle memory, security, and other performance operations. Q#4) What is an interface? Answer: An interface is a class without an application. The only thing it contains is the declaration of methods, properties, and events. Q#5) What are the different types of lessons in C#? Answer: Class types in C# are: Partial class: It allows you to divide or share its members with multiple .cs files. It is marked by the partial class keyword. Sealed: This is an inalieverable class. To access members of a sealed classroom, we need to create the object of the class. It is marked by the Sealed.Abstract class of the keyword: This is a class in which you cannot create a shape. The class can only be inherited. It should contain at least one method. It is marked by the keyword summary. Static class: This is a class that does not allow inheritance. Classmates are also static. The static keyword indicates it. This keyword instructs the rusher to check for accidental occurrences of the static class. Q #6) Explain code compilation in C#. Answer: Compiling code in C# includes the following four steps: compiling the source code into code managed by the C#compiler. Loads the shared language runtime (CLR). CIR is assembling. Q#7) What are the differences between class and bills? Answer: The following are the differences between a class and Struct:Q #8) What is the difference between the virtual method and the abstract method? Answer: The virtual method must always have a default application. However, it can be circumvented in a derivative class, although it is not mandatory. It can be override by using the override keyword. An abstract method does not include an application. It resides in the abstract classroom. It is mandatory that the derivative class implements the abstract method. An indirect keyword is not needed here even though it can be used. Q #9) Explains namespaces in C#. Answer: They are used to organize large code projects. A system is the most common namespace in C#. We can create our own namespace and also use one namespace in one namespace in another namespace, called nested namespaces.They are marked by the namespace keyword. Q #10) What is the use of a declaration in C#? Answer: The usage keyword indicates that the specific namespace is being used by the program. For example, by using SystemHere, System is a namespace. The class console is set under System. Therefore, we can use console.writeline (....) or readline in our program. Q #11) Abstraction Explanation.Answer: Abstraction is one of the concepts of OOP. It is used to show only the essential features of the class and to hide unnecessary information. Let us take an example of a car: a driver of the car needs to know the details about the car such as color, name, mirror, steering, gear, brake, etc. What he doesn't need to know is an internal engine, an exhaust system. Therefore, abstraction helps to know what is necessary and to hide the internal details from the outside world. You can obtain hiding the internal information by declaring parameters such as private by using the private keyword. Q #12) Explain polymorphism? Answer: Programically, polymorphism means the same method but different applications. It's of 2 types, time compilation and Polymorphism is achieved by overloading the operators. Runtime polymorphism is achieved by overtaking. Inheritance and virtual functions are used during runtime polymorphisms. For example, if a class includes a Void Add method,polymorphism is achieved by overloading the method, all of this, Void Add(int a, int b), void Add(int add) are all busy methods. Q #13) How is exception handling implemented in C#? Answer: Exceptions are handled using four keywords in C#:try: Contains a code block for which an exception will be checked.catch: This is a program that catches an exception with the exception handler.Finally: This is a code block written for execution whether an exception is caught or not. Thesis: Provides an exception when an issue occurs. Q#14) What are C# income rates? What are the common I/O rates? Answer: C# System.IO names, consisting of metals used to perform various actions on files such as creating, deleting, opening, closing, etc. Some common I/O classes are:File – Helps manipulate a file. StreamWriter – used to write characters to stream. StreamReader – used to read characters to stream. StringWriter – Used to read string buffer. StringReader – Used to write string buffer. Path – Used to perform actions related to the path information. Q#15) What is the StreamReader/StreamWriter class? Answer: StreamReader and StreamWriter are class of the namespace System.IO. They are used when we want to read or write charact90, reader-based data, respectively. Some StreamReader members are: Close(), Read(), Readline(). StreamWriter members are: close(), write(), line writing(). Class 1 program { using (StreamReader sr = New StreamReader (C:\ReadMe.txt) { //----------Readable Code----------// } using (StreamWriter sw = New StreamWriter (C:\ReadMe.txt)) { //----------Write Code----------// } Q #16) What is an exterminator in C#? Answer: The extermination is used to clean the memory and clean the resources, but in C it is done by the garbage collector on his own. System.GC.Collect() is internally called for cleaning. But sometimes the destroyers may need to be implemented manually. For example:~Car() { Console.writeline(...); Q #17) What is an abstract class? Answer: An abstract class is a class that is founded by an abstract keyword and cannot only be used as a base bag. This class should always be inherited. The class itself cannot be created. If we don't want any program to create a class object, then such classes can become abstract. Each method in the abstract class does not have applications in the same class. But they must be applied in a child's classroom. For example: AB1 Abstract Class { Add Public Space(); } Class Children's Department : AB1 { Children's Department cs = New Pediatric Ward (); Amount int = cs. Insert(); } All methods in an abstract class are virtual methods in the concept. Therefore, do not use the virtual keyword with any In the abstract classroom. Q #18) What are boxing and unboxing? Answer: Converting a value type to a reference type is called Boxing.For example: int Value1 -= 10; ———— Boxes————// Object with Value = Value1; Explicit conversion of the same reference type (generated by boxing) back to a value type is called Unboxing.For example: //————UnBoxing————// int UnBoxing = int (boxedValue); Q#19) What is the difference between continuing to make a pause statement? Answer: The Break declaration breaks the loop. It makes the show's control come out of the loop. The Continue declaration makes controlling the program exit only the current stream. It doesn't break the loop. Q#20) What's the difference between finally and a final block? Answer: Finally block called after making of try to grab block. It's used to treat exceptions. Whether or not an exception is caught, this code block will be executed. Usually, this block will be a cleaning code.A final method is called just before garbage collection. It is used to perform unmanaged code cleanup operations. It is automatically called when a given occurrence is not subsequently read. Arrays and strings #21) What is an array? Give the syntax for a single, multidimensional array? Answer: An array is used to store multiple variables of the same type. This is a collection of variables stored in a contiguous memory location. Example:Double Numbers = New Double[10]; int[] Score = New int[4] {25,24,23,25}; A one-dimensional array is a linear array in which the variables are stored in one row. In the example above, a one-dimensional array. Arrays can be more than one dimension. Multidimensional arrays are also called rectangular arrays. For example, int[,] numbers = int[3,2] { {1,2},{2,3},{3,4} }; Q#22) What is a jagged array? Answer: A jagged array is an array whose components are arrays. It's also called the array. It can also be single or dimensions.int[] jaggedArray = int[4][]; Q #23) Array.Answer property number name: Array properties include: Length: Accepts the total number of elements in an array. IsFixedSize: Indicates whether the array is fixed in size or not. IsReadOnly: Specifies whether the array is read-only or not. Q#24) What is an array class? Answer: An array class is the base class for all arrays. It provides many characteristics and methods. It exists in the namespace system. Q#25) What is a string? What are the properties of a string class? Answer: A string is a collection of character objects. We can also declare string variables named c#.string = C# questions; String class in C# represents a string. The properties of the string class are: Characters get the Char object in the current String.Length Getting the number of objects in String.Q #26) What is the Escape sequence? Name some string escape sequences in C#. Answer: Escape sequence is marked by a backslash (\). The rear-sedition indicates that the next character after it should be Literally or it's a special character. An escape sequence is considered a single character. String escape sequences are as follows: – New Line Character\b – Backspace \\ – Backspace \\ – Backslash \' – Single Quote \" – Double QuoteQ #27) What are regular expressions? Search the string using regular expressions? Answer: A regular expression is a format that matches an input format. The format can consist of operators, structures, or character stro-verbals. Regex is used for string parsing and replacing the string of characters. For example:* matches the nature of zero or more. So, regex a*b is equivalent to b, ab, aab, aaab and so on. Search the string using Regex: Primary static space (string[] args) { string[] languages = { C#, python, Java }; the example above searches for Python against the input scheme from the language set. Explain.Answer: Some of the basic string operations are:Concatenate: You can align two strings by using System.String.Concat or the + operator. Change: Replace(a,b) is used to replace a string with another string. Trim() is used to crop the string at the end or beginning. Compare: System.StringComparison() is used to compare two strings, a case-sensitive or case-independing comparison. It takes mostly two parameters, an original string, and a string to compare with. Search: StartWith, ends with methods used to search for a specific string. Q#29) What is surgery? How do I parse a date time string? Answer: Analyzing strings converts a string to another data type. For example: string text = 500; intnum = int. parsing(text); 500 is an insamation number. Therefore, the Analysis method converts the 500 string to its root type, that is, int. Follow the same method to convert a DateTime string. Date Time String = Jan 1, 2018; Date and time analyzed value = DateTime.parse (date and time); Advanced concepts #30) What is a delegate? Explain.Answer: A delegate is a variable that holds the method reference. Therefore, this is a function pointer or reference type. All delegates are derived from the System.Delegate namespace. Both the delegate and the method he is referring to can be the same signature. Delegate Announcement: Void AddNumbers(int n) public axis; After a delegate's declaration, the delegate must create the object by using the new keyword. Add numbers an1 = new additional numbers(number); The delegate provides a type of enquirer to the reference method, which will be read internally when a delegate is called. public representative int myDel(int number); Public class plan { number of additional public numbers (int a) { int Sum = a + 10; return amount; } Public space Start() { myDel DelgateExample = AddNumbers; } } in In example, we have a myDel delegate who takes an inemperation number value as a parameter. A class plan includes a method of the same signature as the delegate, called AddNumbers(). If there is another method called Start() that a delegate object is created, the object can be assigned to AddNumbers as it has the same signature as the delegate. Q#31) What are events? Answer: Events are user actions that generate messages for the application to which they need to respond. User actions can be mouse movements, press keys, and so on. Programmfully, a class that uploads an event is called an advertiser and a department that responds/accepts the event is called a subscription. An event should have at least one other subscription that this event will never upload. Delegates are used to announced events.Public representative Turn off PrintNumbers(); print events from myEvent numbers; Q#32) How to use delegates with events? Answer: Delegates are used to upload and handle events. A representative must always be announced first and then the events are announced. Let us see an example: Consider a class called Patient. Consider two other departments, insurance, and a bank that requires patient death information from the patient's condition. Here, insurance and bank are the subscribers and from the sick class becomes the mall. It triggers the death event and the other two girls should get the event. Namespace ConsoleApp2 { Public Ward Patient { Public Representative Cancel DeathInfo();//Announcement of Representative// Public Event DeathInfo deathDate;//Declaring on incident// Public Space Death() { deathDate(); } Public Class Insurance [MyPat Patient = New Patient(); void GetDeathDe tails() { //-------Date----------// } primary blank() { //--------Rerealized GetDeathDetails function----------// myPat.deathDate += GetDeathDetails; } Public Department Bank { myPat patient = new patient(); Void GetPatInfo () { //------- did something with a deathDate event----------// } void Main() { //-------- Sign up for the GetPatInfo function ----------// myPat.deathDate += GetPatInfo; I don't know if I can do that. Answer: Different types of delegates are: single delegate: a delegate who can call one method. Multicast delegate: A delegate who can call multiple actions. + and – Operators are used to register and not subscribe respectively. General Delegate: Do not require an instance definition of the delegate. It consists of three types, Action, Funcs, and Predicate.Action– In the Hanel example of delegates and events, we can replace the definition of a delegate and an event with an action keyword. The action delegate defines a method that can be timed by arguments but does not return the result of a public representative to cancel deathInfo(); Death at a public eventInfo DeathDate; Replace Action// Public Event Action DeathDate; An action refers in a throes to a delegate. Func – A Func representative defines a method that can be used for a time when arguments are returned and returns a result. Func <int, string,= bool=> myDel is </int,>as a representative of bool myDel (int a, string B); Predicate – Defines a method that can be called on arguments and always returns the bool. Predicate<string> myDel is the same as bool myDel(string); Q#34) What do multicast delegates mean? Answer: A delegate indicates more than one method is called a multicast delegate. Multicast is achieved using operator + and +=. Consider the example from Q #32.There are two subscriptions for deathEvent, GetPatInfo and GetDeathDetails. And that's why we used a += operator. This means that every time myDel is called, both subscribers are called. Delegates will read in the order in which they will be added. Q #35) Explain publishers and subscribers in Events.Answer: Publisher is a class responsible for publishing a message from different types of other departments. The message is nothing but an event as discussed in the questions above. From the example in Q #32, a class patient is the Publisher class. It creates a event that is accepted by other departments. Subscribers capture the type of message they want. Again, from the example of Q#32, grade and bank insurance are subscriptions. They're interested in death in case of an empty type event. Q#36) What are synchronous and asynchronous operations? Answer: Synchronization is a way to create safe code on a thread where only one thread can access a resource at any given time. The asynchronous call waits for the method to complete before continuing with the program flow. Synchronous programming adversely affects user interface actions when the user tries to perform time-consuming actions because only one thread is used. In an asynchronous operation, a call to the method will return immediately so that the program can perform other actions when the method called completes its work in certain situations. In C#, asynchronous and waiting keywords are used to obtain asynchronous programming. You'll see #43 Q for more details on synchronous programming. Q #37) What is reflection in C#? Answer: Reflection is the ability of code to access assembly metadata during runtime. A program reflects itself and uses metadata to inform the user or change their behavior. Metadata refers to information about objects, methods of service. The System.Reflection namespace system contains methods and classes that manage the information of all types and methods loaded. It is used primarily for Windows applications, for example, to display the properties of a button on a Windows form. The Class Reflection MemberInfo object is used to discover the attributes associated with the class. Reflection is implemented in two steps, first, we get the object type, and then we use the type to identify members such as methods and properties. To get a type of class, we can simply use,type mytype=myClass.GetType();once we have a type of class, the other information about the class can be easily accessed. system.reflection.connect info information = mytype. GetMethod(AddNumbers); The sentence above is trying to find </string>Method with AddNumbers name in myClass.Q class #38) What is a global class? Answer: Global or global classes are used to create classes or objects that do not have a specific data type. The data type can be assigned during runtime, that is, when used in a program. For example: From the above code, we see 2 comparison methods initially, to compare string and int. In the case of other data type parameter comparisons, instead of creating many busy methods, we can create a global class and migrate an alternate data type, that is, T. Therefore, T acts as a data type until it is used specifically by the Main() method. Q #39) Explanation of capacitor properties and accessory definition? Answer: Accept and set up are called accessories. These are used by assets. The property provides a mechanism for reading, writing the value of a private field. To access this private field, these accessories are used. Get Property is used to return the value of the Set Property accessory used to set the value. The use of get and set is below:Q #40) What is a thread? What is multiple walks? Answer: A thread is a set of instructions that can be executed that will allow our program to perform simultaneous processing. Simultaneous processing helps us do more than one action at a time. By default, C# has only one thread. But you can create the other threads to execute the code in parallel with the original thread. Wire has a life cycle. It starts every time a thread class is created and stopped after execution. System.Threading is the namespace to include to create threads and use its members. Threads are created by expanding the thread class. The Start() method is used to start threading./CallThread is the target method// ThreadStart methodReading them = New ThreadStart (CallThread); Child ThreadRead = New Thread (MethodRead); ChildThread.Start(); C# can perform more than one task at a time. This is done by handling different processes by different threads. This is called MultiThreading.There are several threading methods used to handle multithreaded operations:start, sleep, cancel, suspend, resume, and Join.Most of these methods are self-explanatory. Q #41) The name of some Thread Class.Answer properties: Few thread class properties are:IsAlive – contains a true value when a thread is Active.Name – can return the thread name. You can also define a name for the thread. Priority – Returns the priority value of the activity determined by the operating system. IsBackground – Accepts or sets a value indicating whether a thread should be a background or foreground process. ThreadState – Describes the status of the thread. Q#42) What are different thread modes? Answer: Different thread modes are: Not Started – A thread is created. Verb – A thread begins with execution. WaitSleep Joinin – sleep reader thread, wait calls on another object and call to join on another thread. Suspended – Thread suspended. Canceled - The thread is dead but has not changed to a stopped state. Discontinued – Thread Has #43) What are a-cinck and wait? Answer: Asynchronous keywords and standby are used to create asynchronous asynchronous programming methods, meaning that the process works regardless of primary or other processes. The use of Async and Await is shown below: The A-Cink keyword is used for the method declaration. The enumeration is an int task that calls the CalculateCount() method. Calculatecount() starts execution and calculates something. Standalone work is done on my thread and then wait for a counting declaration. If Calculatecount is not finished, myMethod will revert to its read method, so that the primary thread is not blocked. If the account calculation has already finished, the result will be available when the control reaches a pending count. So the next step will continue with the same thread. However, this is not the case in Hannel where the delay of one second is involved. Q #44) What is a dead end? Answer: A deadlock is a situation in which a process is unable to complete its execution because two or more processes are waiting to finish each other. This usually occurs in a thread decision. This is where a shared resource is held by a process, and another process waits for the first process to release it, and the thread that holds the locked item waits for another process to complete. Consider the example below: Performing tasks accesses objB and waits one second. In the meantime, PerformtaskB tries to access ObjA.After 1 second, PeformtaskA tries to access ObjA which is locked by PerformtaskB.PerformtaskB and tries to access ObjB which is locked by PerformtaskA.This creates Deadlock.Q #45) explain locking, monitors, and Mutex object in Threading.Answer: Keyword locking ensures that only one thread can enter a specific section of the code at any given time. In the example above, lock(ObjA) means that the lock is placed on objA until this process releases it, no other thread can access ObjA.Mutex and is also like a lock but it can work across multiple processes at a time. WaitOne() is used for locking and ReleaseMutex() is used to release the lock. But Mutex is slower than locking up as it takes time to purchase and release it. Monitor.Enter and Monitor.Exit have an internal lock. A lock is a shortcut for monitors. Internal calls are locked. Monitor. Enter(ObjA); Try { } finally {Monitor.Exit(ObjA));} Q #46) What is race mode? Ans: Race mode occurs when two threads access the same resource and try to change it at the same time. The thread that can access the resource cannot be predicted first. If we have two threads, T1 and T2, and they try to access a shared resource called X. Q #47) What is a thread pool? Ans: A thread pool is a collection of threads. These threads can be used to perform tasks without interfering with the primary thread. After the thread completes the task, the thread returns to the repository. system.threadpool.threadpool namespace includes classes Conduct the pool threads and its operations. System, threads. ThreadPool.QueueUserWorkItem (new system.threads.WaitCallback(SomeTask)); The row above queues a task. SomeTask methods should have an Object.Q parameter #48) What is serialization? Answer: Serialization is the process of converting code to its binary format. Once converted to home, it can be easily stored and written to disk or any such storage device. Serializations are most useful when we don't want to lose the original form of the code and retrieve it at any time in the future. Each class marked with [Serializable] will be converted to its binary form. The reverse process of getting C# code back from the binary form is called Deserialization.To Make an object serialize an object that we need to be serialized, a stream that can contain the object in a series, and the System.Runtime.ization namespace system can contain classes for serialization. Q#49) What are serialization types? Answer: The different types of serialization are: XML serialization - it serializes all public properties for the XML document. Because the data is xml format, it can be read and easily manipulated in different formats. The lessons reside in System.sml.Serialization.SOAP – the lessons reside in System.Runtime.Serialization. Similar to XML but produces a full SOAP-compatible envelope that can be used by any system that understands SOAP. Binary Serialization – Allows you to convert any code to its binary form. You can serialize and restore public and non-public properties. It's faster and take up less space. Q#50) What is an XSD file? Answer: An XSD file represents an XML Schema definition. It structures the XML file. This means that it decides which elements XML should be in and in what order and what properties should be. Without an XSD file associated with XML, the XML can contain any tags, attributes, and elements. The Xsd tool.exe converts the files to XSD format. During serialization of C# code, the lessons are converted to an XSD-compliant format by xsd.exe.ConclusionC# is growing rapidly by the day and it plays a major role in industry testing software.I am sure that this article will make your preparation for the interview much easier and give you quite a amount of knowledge of most C# topics. Hope you'll be ready to deal with any C# interview with confidence!! Confidently!!