Below is the 25 most common 2019 Computer Science interview Questions that are asked mostly:

**1. What is a file?**

**Answer:**  
A file is a named location which stores data or information permanently. A file is always stored inside a storage device using file name (e.g. STUDENT.MARKS). A file name normally has primary and secondary name separated by a “.”(DOT).

**2.What is a class?**

**Answer:**  
A class is a blueprint from which objects are created. A class contains methods and variables associated with an instance of a class.

**3.What is an object?**

**Answer:**  
An object is an instance of a class. For example  
class Abc{ —– This is a class  
int a; —— This is a variable  
public Abc(); —- This is contractor  
public static void main (String args[]) ——- This is a method  
{  
Abc a= new Abc(); —— This is object creation where ‘a’ is the reference variable or object name  
}  
}

**4.What is a constructor?**

**Answer:**  
A constructor is methods which are used to create an Object of class. There are two types of constructor Default & Parameterized constructor.

Popular Course in this category

Programming Languages Courses 40 Online Courses | 270+ Hours | Verifiable Certificate of Completion | Lifetime Access   
4.5 (826 ratings)

Course Price   
$199 ~~$499~~   
[View Course](https://www.educba.com/software-development/courses/programming-languages-course/?btnz=edu-blg-inline-banner1)

Related Courses

[C Programming Course](https://www.educba.com/software-development/courses/c-programming-course/?btnz=edu-blg-inline-banner1)[Selenium Training Certification](https://www.educba.com/software-development/courses/selenium-training-certification/?btnz=edu-blg-inline-banner1)

**5.What is the different OOPS principle?**

**Answer:**  
The basic OOPS principle are as follows,

* Encapsulation
* Abstraction
* Inheritance
* Polymorphism

**6.What is inheritance?**

**Answer:**  
Inheritance is property in which the property of a parent class(Superclass) is passed on to child class(Subclass). For example  
class Abc{ —– This is a class  
int a; —— This is a variable  
public void abc(){} — Methods  
}  
class Xyz extends Abc —–(Extend is the keyword, Xyz is the subclass which inherits the properties of ABC parent class.)  
{  
public static void main (String args[]) ——- This is a method  
{  
Abc a= new Abc(); —— This is object creation where ‘a’ is the reference variable or object name  
}  
}

**7.What is polymorphism?**

**Answer:**  
Polymorphism is the ability of an object to take on multiple forms. Most commonly polymorphism is used in OOP when a parent class reference is used to refer to a child class object.

**8. What are the instance and class variables?**

**Answer:**  
Instance variable belongs to a particular instance of that class whereas Class variable. A class variable is also known as static variables. For example  
public class Abc{  
public int a; …….. This is an instance variable  
public static int a1;…….. This is a static or class variable  
……………………..  
……………..  
}

**9.Compare method and constructor?**

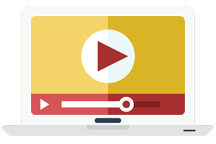
**Answer:**  
Constructor: Used to initialize the instance of a class.  
Method: Used to perform some function or operation.

Constructor: Doesn’t have a return type.  
Method: Has a return type.

**10. What is a singleton class?**

**Answer:**  
Singleton class limits the number of objects created for a class to one but gives the flexibility of creating more objects if the situation changes.

[Programming Languages Courses 40 Online Courses | 270+ Hours | Verifiable Certificate of Completion | Lifetime Access Watch The Course Preview](https://www.educba.com/software-development/courses/programming-languages-course/?btnz=edu-blg-inline-banner2" \t "_blank)

[[](https://www.educba.com/software-development/courses/programming-languages-course/?btnz=edu-blg-inline-banner2)](https://www.educba.com/software-development/courses/programming-languages-course/?btnz=edu-blg-inline-banner2" \t "_blank)

**11. What are the steps for creating the object?**

**Answer:**  
An object is first declared then instantiated and at last declared. For example  
Abc a= new Abc();

**12. What is the different type of access modifiers?**

**Answer:**  
There are four type of access modifiers as given below:-  
• Visible to the overall package. No modifier needed.  
• Private – Visible to class only.  
• Public – Visible to the world.  
• Protected – Visible to package and subclass.

**13.Which is the highest operator precedence in Java**

**Answer:**  
The operator with the highest preference is Postfix operators i.e () [].

**14.What is an array?**

**Answer:**  
The array is a container which holds the fixed number of similar data types.

**15. What is the difference between equals() and method and == operator?**

**Answer:**  
The equals() is a method and it matches the content of the strings whereas == is an operator and matches object or reference of the strings.

**16. Is string class final?**

**Answer:**  
Yes

**17. What is a wrapper class?**

**Answer:**  
To access the primitive data type as an object we use wrapper class. They are following:-

|  |  |
| --- | --- |
| **Primitive Type** | **Wrapper class** |
| boolean | Boolean |
| char | Character |
| byte | Byte |
| short | Short |
| int | Integer |
| long | Long |
| float | Float |
| double | Double |

**18.Difference between overloading and overriding?**

**Answer:**  
Overloading is when two or more methods in the same class have the same method name but different parameters(i.e different method signatures).  
Overriding is when two methods having the same method name and parameters (i.e., method signature) but one of the methods is in the parent class and the other is in the child class.

**19. What are multiple inheritances in Java?**

**Answer:**  
[Java](https://www.educba.com/course/java/) supports multiple inheritances i.e the ability of a class to implement more than one interface. A class can implement multiple Interfaces but cannot extends multiple classes.

**20. What is a stream?**

**Answer:**  
A stream can be defined as the sequence of data. There is two type of streams.  
InputStream: Used to read data from a source.  
OutPut Stream: Used to write data into a destination.

**21. What is a Character stream?**

**Answer:**  
Java Character stream is basically used to perform input and output for 16 bit Unicode. The main classes users are FileReader and FileWriter which internally uses FileInputStream and FileOutputStream so the basic difference is that FileReader and FileWriter read and writes two bites at a time respectively.

**22. What is a Byte stream?**

**Answer:**  
Java Byte stream is basically used to perform input and output for 8 bit Unicode.  
The main classes related to byte streams are FileInputStream and FileOutputStream.

**23. What is an Interface?**

**Answer:**  
The interface is a reference type in [Java](https://www.educba.com/course/design-patterns-java/), similar to the class but its collection of abstract methods. A class can implement multiple interfaces.

**24.Difference between class and interface?**

**Answer:**  
Below are the difference between Interface and class:-

* The interface cannot be instantiated.
* An interface doesn’t have any constructors.
* Interface only have abstract methods.
* A class implements an interface and extends a class.
* An interface can extend multiple interfaces.

**25. What is an abstract class?**

**Answer:**  
A class which contains the abstract keyword in a declaration is called abstract class. The properties of the abstract class are as follows:-

* Abstract classes may or may not contain *abstract methods* but, if a class has at least one abstract method, then it must be declared abstract.
* The abstract class cannot be instantiated.
* To use an abstract class, we have to inherit it from another class.
* If we inherit an abstract class, then we have to provide implementations to all the abstract methods in it

### 7. What is a class? What is a super-class?

As with any interview, there will be multiple technical questions to assess your general knowledge and familiarity with common terms and topics. You might answer:

“A class defines the characteristics of a certain type of object such as which messages they will respond to and what form of response they will take. On the other hand, a super-class is the basis of the class under consideration.”

### 8. What is a default constructor? What is a conversion constructor?

As you prepare, it is necessary to go over basic concepts and refresh your memory about important skills and subjects you may need to study again:

“A default constructor either has no arguments, or all of the arguments have default argument values. A conversion constructor accepts one argument of a different type and uses this as a way to infer conversion rules for a class.”

### 9. What is the difference between C and C++ ? Would you prefer to use one over the other?

While there is no right answer for a question like this, the interviewer is assessing your knowledge of these common programming languages while also checking to see if you can compare and contrast them.

You might answer something like:

“C++ supports the object-oriented programming paradigm while C is based on structured programming. I prefer C++ due to its object-oriented nature but also the fact that almost anything can be built using it.”

### 10. What is multiple inheritance? What are its advantages and disadvantages?

Being able to explain common topics but also point out their advantages and disadvantages will be critical to your interviews:

“Multiple inheritance is the process where a subclass can be derived from more than one super-class. Its advantage is that a class can inherit the functionality of more than one base class, but its disadvantage is that it can lead to a lot of confusion when two base classes implement a method with the same name.”

Although computer science is an extremely broad field, we know that these top ten computer science interview questions will prepare you for what lies ahead. By brushing up on basic concepts and becoming familiar with the best sample answers, you will be on your way to the job of your dreams in no time

 **Question 1. What Is An Object In C++?**

**Answer :**

An object is a package that contains related data and instructions. The data relates to what the object represents, while the instructions define how this object relates to other objects and itself.

 **Question 2. What Is A Message?**

**Answer :**

A message is a signal from one object to another requesting that a computation take place. It is roughly equivalent to a function call in other languages.

 **Question 3. What Is A Class?**

**Answer :**

A class defines the characteristics of a certain type of object. It defines what its members will remember, the messages to which they will respond, and what form the response will take.

 **Question 4. What Is An Instance?**

**Answer :**

An individual object that is a member of some class.

 **Question 5. What Is A Super-class?**

**Answer :**

Given a class, a super-class is the basis of the class under consideration. The given class is defined as a subset (in some respects) of the super-class. Objects of the given class potentially posses all the characteristics belonging to objects of the super-class.

 **Question 6. What Is Inheritance?**

**Answer :**

Inheritance is property such that a parent (or super) class passes the characteristics of itself to children (or sub) classes that are derived from it. The sub-class has the option of modifying these characteristics in order to make a different but fundamentally related class from the super-class.

 **Question 7. To What Does Message Protocol Refer?**

**Answer :**

An object’s message protocol is the exact form of the set of messages to which the object can respond.

 **Question 8. What Is Polymorphism?**

**Answer :**

Polymorphism refers to the ability of an object to respond in a logically identical fashion to messages of the same protocol, containing differing types of objects. Consider 1 + 5 and 1 + 5.1. In the former, the message “+ 5” is sent to an object of class integer (1). In the later, the message “+ 5.1” is sent to the same integer object. The form of the message (its protocol) is identical in both cases. What differs is the type of object on the right-hand side of these messages. The former is an integer object (5) while the later is a floating point object (5.1). The receiver (1) appears (to other objects) to respond in the same way to both messages. Internally, however, it knows that it must treat the two types of objects differently in order to obtain the same overall response.

 **Question 9. What Are Instance Variables?**

**Answer :**

These represent an object’s private memory. They are defined in an object’s class.

 **Question 10. What Are Class Variables?**

**Answer :**

These represent a class’s memory which it shares with each of its instances.

 **Question 11. What Is A Method?**

**Answer :**

A method is a class’s procedural response to a given message protocol. It is like the definition of a procedure in other languages.

 **Question 12. In C++ What Is A Constructor? A Destructor?**

**Answer :**

A constructors and destructors are methods defined in a class that are invoked automatically when an object is created or destroyed. They are used to initialize a newly allocated object and to cleanup behind an object about to be removed.

 **Question 13. Compare And Contrast C And C++.?**

**Answer :**

**Comparison:** C++ is an extension to the C language. When C++ is used as a procedural language, there are only minor syntactical differences between them.

**Contrast:** When used as a procedural language, C++ is a better C because:

* It vigorously enforces data typing conventions.
* It allows variables to be defined where they are used.
* It allows the definition of real (semantically significant) constants.
* It allows for automatic pointer dereferencing.
* It supports call-by-reference in addition to call-by-value in functions.
* It supports tentative variable declarations (when the type and location of a variable cannot be known before hand.

As an object oriented language, C++ introduces much of the OOP paradigm while allowing a mixture of OOP and procedural styles.

 **Question 14. What Is Operator Overloading?**

**Answer :**

It is the process of, and ability to redefine the way an object responds to a C++ operator symbol. This would be done in the object’s class definition.

 **Question 15. What Is Cin And Cout?**

**Answer :**

They are objects corresponding to a program’s default input and output files.

Contrast procedural and object oriented programming.

The procedural paradigm performs computation through a step-by-step manipulation of data items. Solving problems this way is akin to writing a recipe. ie: All the ingredients (data items) are defined. Next a series of enumerated steps (statements) are defined to transform the raw ingredients into a finished meal.

The object oriented model, in contrast, combines related data and procedural information into a single package called an object. Objects are meant to represent logically separate entities (like real world objects). Objects are grouped together (and defined by) classes. (This is analogous to user defined data types in procedural languages.) Classes may pass-on their “makeup” to classes derived from them. In this way, Objects that are of a similar yet different nature need not be defined from scratch.

Computation occurs though the intercommunication of objects. Programming this way is like writing a play. First the characters are defined with their attributes and personalities. Next the dialog is written so that the personalities interact. The sum total constitutes a drama.

 **Question 16. How Do You Link A C++ Program To C Functions?**

**Answer :**

By using the extern “C” linkage specification around the C function declarations.

You should know about mangled function names and type-safe linkages. Then you should explain how the extern “C” linkage specification statement turns that feature off during compilation so that the linker properly links function calls to C functions.

 **Question 17. Explain The Scope Resolution Operator.?**

**Answer :**

The scope resolution operator permits a program to reference an identifier in the global scope that has been hidden by another identifier with the same name in the local scope.

The answer can get complicated. It should start with “colon-colon,” however. (Some readers had not heard the term, “scope resolution operator,” but they knew what :: means. You should know the formal names of such things so that you can understand all communication about them.) If you claim to be well into the design or use of classes that employ inheritance, you tend to address overriding virtual function overrides to explicitly call a function higher in the hierarchy. That’s good knowledge to demonstrate, but address your comments specifically to global scope resolution. Describe C++’s ability to override the particular C behavior where identifiers in the global scope are always hidden by similar identifiers in a local scope.

 **Question 18. What Are The Differences Between A C++ Struct And C++ Class?**

**Answer :**

The default member and base class access specifiers are different.

This is one of the commonly misunderstood aspects of C++. Believe it or not, many programmers think that a C++ struct is just like a C struct, while a C++ class has inheritance, access specifiers, member functions, overloaded operators, and so on. Some of them have even written books about C++. Actually, the C++ struct has all the features of the class.

The only differences are that a struct defaults to public member access and public base class inheritance, and a class defaults to the private access specifier and private base class inheritance. Getting this question wrong does not necessarily disqualify you because you will be in plenty of good company. Getting it right is a definite plus.

 **Question 19. How Many Ways Are There To Initialize An Int With A Constant?**

**Answer :**

There are two formats for initializers in C++ as shown in Example 1. Example 1(a) uses the traditional C notation, while Example 1(b) uses constructor notation. Many programmers do not know about the notation in Example 1(b), although they should certainly know about the first one. Many old-timer C programmers who made the switch to C++ never use the second idiom, although some wise heads of C++ profess to prefer it.

A reader wrote to tell me of two other ways, as shown in Examples 2(a) and 2(b), which made me think that maybe the answer could be extended even further to include the initialization of an int function parameter with a constant argument from the caller.

 **Question 20. How Does Throwing And Catching Exceptions Differ From Using Setjmp And Longjmp?**

**Answer :**

The throw operation calls the destructors for automatic objects instantiated since entry to the try block.

Exceptions are in the mainstream of C++ now, so most programmers, if they are familiar with setjmp and longjmp, should know the difference. Both idioms return a program from the nested depths of multiple function calls to a defined position higher in the program.

The program stack is “unwound” so that the state of the program with respect to function calls and pushed arguments is restored as if the calls had not been made. C++ exception handling adds to that behavior the orderly calls to the destructors of automatic objects that were instantiated as the program proceeded from within the try block toward where the throw expression is evaluated.

It’s okay to discuss the notational differences between the two idioms. Explain the syntax of try blocks, catch exception handlers, and throw expressions. Then specifically address what happens in a throw that does not happen in a longjmp. Your answer should reflect an understanding of the behavior described in the answer just given.

One valid reason for not knowing about exception handling is that your experience is exclusively with older C++ compilers that do not implement exception handling. I would prefer that you have at least heard of exception handling, though.

It is not unusual for C and C++ programmers to be unfamiliar with setjmp/ longjmp. Those constructs are not particularly intuitive. A C programmer who has written recursive descent parsing algorithms will certainly be familiar with setjmp/ longjmp.

Others might not, and that’s acceptable. In that case, you won’t be able to discuss how setjmp/longjmp differs from C++ exception handling, but let the interview turn into a discussion of C++ exception handling in general. That conversation will reveal to the interviewer a lot about your overall understanding of C++.

 **Question 21. What Is Your Reaction To This Line Of Code?**

**Answer :**

**delete this;**

It’s not a good practice.

A good programmer will insist that the statement is never to be used if the class is to be used by other programmers and instantiated as static, extern, or automatic objects. That much should be obvious.

The code has two built-in pitfalls. First, if it executes in a member function for an extern, static, or automatic object, the program will probably crash as soon as the delete statement executes. There is no portable way for an object to tell that it was instantiated on the heap, so the class cannot assert that its object is properly instantiated.

Second, when an object commits suicide this way, the using program might not know about its demise. As far as the instantiating program is concerned, the object remains in scope and continues to exist even though the object did itself in. Subsequent dereferencing of the pointer can and usually does lead to disaster.

A reader pointed out that a class can ensure that its objects are instantiated on the heap by making its destructor private. This idiom necessitates a kludgy DeleteMe kind of function because the instantiator cannot call the delete operator for objects of the class. The DeleteMe function would then use “delete this.”

I got a lot of mail about this issue. Many programmers believe that delete this is a valid construct. In my experience, classes that use delete this when objects are instantiated by users usually spawn bugs related to the idiom, most often when a program dereferences a pointer to an object that has already deleted itself.

 **Question 22. What Is A Default Constructor?**

**Answer :**

A constructor that has no arguments or one where all the arguments have default argument values.

If you don’t code a default constructor, the compiler provides one if there are no other constructors. If you are going to instantiate an array of objects of the class, the class must have a default constructor.

 **Question 23. What Is A Conversion Constructor?**

**Answer :**

A constructor that accepts one argument of a different type.

The compiler uses this idiom as one way to infer conversion rules for a class. A constructor with more than one argument and with default argument values can be interpreted by the compiler as a conversion constructor when the compiler is looking for an object of the type and sees an object of the type of the constructor’s first argument.

 **Question 24. What Is The Difference Between A Copy Constructor And An Overloaded Assignment Operator?**

**Answer :**

A copy constructor constructs a new object by using the content of the argument object. An overloaded assignment operator assigns the contents of an existing object to another existing object of the same class.

First, you must know that a copy constructor is one that has only one argument, which is a reference to the same type as the constructor. The compiler invokes a copy constructor wherever it needs to make a copy of the object, for example to pass an argument by value. If you do not provide a copy constructor, the compiler creates a member-by-member copy constructor for you.

You can write overloaded assignment operators that take arguments of other classes, but that behavior is usually implemented with implicit conversion constructors. If you do not provide an overloaded assignment operator for the class, the compiler creates a default member-by-member assignment operator.

This discussion is a good place to get into why classes need copy constructors and overloaded assignment operators. By discussing the requirements with respect to data member pointers that point to dynamically allocated resources, you demonstrate a good grasp of the problem.

 **Question 25. When Should You Use Multiple Inheritance?**

**Answer :**

There are three acceptable answers: “Never,” “Rarely,” and “When the problem domain cannot be accurately modeled any other way.”

There are some famous C++ pundits and luminaries who disagree with that third answer, so be careful.

Let’s digress to consider this issue lest your interview turn into a religious debate. Consider an Asset class, Building class, Vehicle class, and CompanyCar class. All company cars are vehicles. Some company cars are assets because the organizations own them. Others might be leased. Not all assets are vehicles. Money accounts are assets. Real-estate holdings are assets. Some real-estate holdings are buildings. Not all buildings are assets. Ad infinitum.

When you diagram these relationships, it becomes apparent that multiple inheritance is an intuitive way to model this common problem domain. You should understand, however, that multiple inheritance, like a chainsaw, is a useful tool that has its perils, needs respect, and is best avoided except when nothing else will do. Stress this understanding because your interviewer might share the common bias against multiple inheritance that many object-oriented designers hold.

 **Question 26. What Is A Virtual Destructor?**

**Answer :**

The simple answer is that a virtual destructor is one that is declared with the virtual attribute.

The behavior of a virtual destructor is what is important. If you destroy an object through a pointer or reference to a base class, and the base-class destructor is not virtual, the derived-class destructors are not executed, and the destruction might not be complete.

 **Question 27. Explain The Isa And Hasa Class Relationships. How Would You Implement Each In A Class Design?**

**Answer :**

A specialized class “is a” specialization of another class and, therefore, has the ISA relationship with the other class. An Employee ISA Person. This relationship is best implemented with inheritance. Employee is derived from Person.

A class may have an instance of another class. For example, an Employee “has a” Salary, therefore the Employee class has the HASA relationship with the Salary class. This relationship is best implemented by embedding an object of the Salary class in the Employee class.

The answer to this question reveals whether you have an understanding of the fundamentals of object-oriented design, which is important to reliable class design.

There are other relationships. The USESA relationship is when one class uses the services of another. The Employee class uses an object (cout) of the ostream class to display the employee’s name onscreen, for example. But if you get ISA and HASA right, you usually don’t need to go any further.

 **Question 28. When Is A Template A Better Solution Than A Base Class?**

**Answer :**

When you are designing a generic class to contain or otherwise manage objects of other types, when the format and behavior of those other types are unimportant to their containment or management, and particularly when those other types are unknown (thus the genericity) to the designer of the container or manager class.

Prior to templates, you had to use inheritance; your design might include a generic List container class and an application-specific Employee class. To put employees in a list, a ListedEmployee class is multiply derived (contrived) from the Employee and List classes. These solutions were unwieldy and error-prone. Templates solved that problem.

 **Question 29. What Is The Difference Between C And C++ ? Would You Prefer To Use One Over The Other ?**

**Answer :**

C is based on structured programming whereas C++ supports the object-oriented programming paradigm.Due to the advantages inherent in object-oriented programs such as modularity and reuse, C++ is preferred. However almost anything that can be built using C++ can also be built using C.

 **Question 30. What Are The Access Privileges In C++ ? What Is The Default Access Level ?**

**Answer :**

The access privileges in C++ are private, public and protected. The default access level assigned to members of a class is private. Private members of a class are accessible only within the class and by friends of the class. Protected members are accessible by the class itself and it’s sub-classes. Public members of a class can be accessed by anyone.

 **Question 31. What Is Data Encapsulation ?**

**Answer :**

Data Encapsulation is also known as data hiding. The most important advantage of encapsulation is that it lets the programmer create an object and then provide an interface to the object that other objects can use to call the methods provided by the object.

The programmer can change the internal workings of an object but this transparent to other interfacing programs as long as the interface remains unchanged.

 **Question 32. What Is Inheritance ?**

**Answer :**

Inheritance is the process of deriving classes from other classes. In such a case, the sub-class has an ‘is-a’ relationship with the super class. For e.g. vehicle can be a super-class and car can be a sub-class derived from vehicle. In this case a car is a vehicle.

The super class ‘is not a’ sub-class as the sub- class is more specialized and may contain additional members as compared to the super class. The greatest advantage of inheritance is that it promotes generic design and code reuse.

 **Question 33. What Is Multiple Inheritance ? What Are It’s Advantages And Disadvantages ?**

**Answer :**

Multiple Inheritance is the process whereby a sub-class can be derived from more than one super class. The advantage of multiple inheritance is that it allows a class to inherit the functionality of more than one base class thus allowing for modeling of complex relationships.

The disadvantage of multiple inheritance is that it can lead to a lot of confusion when two base classes implement a method with the same name.

 **Question 34. What Do The Keyword Static And Const Signify?**

**Answer :**

When a class member is declared to be of a static type, it means that the member is not an instance variable but a class variable. Such a member is accessed using Classname.Membername (as opposed to Object.Membername). Const is a keyword used in C++ to specify that an object’s value cannot be changed.

 **Question 35. How Is Memory Allocated/deallocated In C ? How About C++ ?**

**Answer :**

Memory is allocated in C using malloc() and freed using free(). In C++ the new() operator is used to allocate memory to an object and the delete() operator is used to free the memory taken up by an object.

 **Question 36. What Is Uml?**

**Answer :**

UML refers to Unified Modeling Language. It is a language used to model OO problem spaces and solutions.

 **Question 37. What Is The Difference Between A Shallow Copy And A Deep Copy?**

**Answer :**

A shallow copy simply creates a new object and inserts in it references to the members of the original object. A deep copy constructs a new object and then creates in it copies of each of the members of the original object