C++ Interview Questions

**1. What are the different data types present in C++?**

The 4 data types in C++ are given below:

* Primitive Datatype(basic datatype). Example- char, short, int, float, long, double, bool, etc.
* Derived datatype. Example- array, pointer, etc.
* Enumeration. Example- enum
* User-defined data types. Example- structure, class, etc.

### 2. What are class and object in C++?

A class is a user-defined data type that has data members and member functions. Data members are the data variables and member functions are the functions that are used to perform operations on these variables.

An object is an instance of a class. Since a class is a user-defined data type so an object can also be called a variable of that data type.

### 3. What is operator overloading?

Operator Overloading is a very essential element to perform the operations on user-defined data types. By operator overloading we can modify the default meaning to the operators like +, -, \*, /, <=, etc.

**4. What is polymorphism in C++?**

Polymorphism in simple means having many forms. Its behavior is different in different situations. And this occurs when we have multiple classes that are related to each other by inheritance.

For example, think of a base class called a car that has a method called car brand(). Derived classes of cars could be Mercedes, BMW, Audi - And they also have their own implementation of a cars

The two types of polymorphism in c++ are:

* Compile Time Polymorphism
  1. Function Overloading
  2. Operator Overloading
* Runtime Polymorphism
  1. Virtual Function
  2. Function Overriding

### 5. Explain constructor in C++

The constructor is a member function that is executed automatically whenever an object is created. Constructors have the same name as the class of which they are members so that compiler knows that the member function is a constructor. And no return type is used for constructors.

### 6. Tell me about virtual function

**Virtual function** is a member function in the base class that you redefine in a derived class. A virtual function is declared using the virtual keyword. When the function is made virtual, C++ determines which function is to be invoked at the runtime based on the type of the object pointed by the base class pointer.

### 7. What do you know about friend class and friend function?

A friend class can access private, protected, and public members of other classes in which it is declared as friends.

Like friend class, friend function can also access private, protected, and public members. But, Friend functions are not member functions.

### 8. What are the C++ access specifiers?

In C++ there are the following access specifiers:

**Public:** All data members and member functions are accessible outside the class.

**Protected:** All data members and member functions are accessible inside the class and to the derived class.

**Private:** All data members and member functions are not accessible outside the class.

### 9. Define inline function

If a function is inline, the compiler places a copy of the code of that function at each point where the function is called at compile time. One of the important advantages of using an inline function is that it eliminates the function calling overhead of a traditional function.

### 10 What is a reference in C++?

A reference is like a pointer. It is another name of an already existing variable. Once a reference name is initialized with a variable, that variable can be accessed by the variable name or reference name both.

### 11. What do you mean by abstraction in C++?

Abstraction is the process of showing the essential details to the user and hiding the details which we don’t want to show to the user or hiding the details which are irrelevant to a particular user.

### 12. Is deconstructor overloading possible? If yes then explain and if no then why?

No destructor overloading is not possible. Destructors take no arguments, so there’s only one way to destroy an object. That’s the reason destructor overloading is not possible.

### 13. What do you mean by call by value and call by reference?

In call by value method, we pass a copy of the parameter is passed to the functions. For these copied values a new memory is assigned and changes made to these values do not reflect the variable in the main function.

In call by reference method, we pass the address of the variable and the address is used to access the actual argument used in the function call. So changes made in the parameter alter the passing argument.

### 14. What is an abstract class and when do you use it?

A class is called an abstract class whose objects can never be created. Such a class exists as a parent for the derived classes. We can make a class abstract by placing a pure virtual function in the class.

### 15. What are destructors in C++?

A constructor is automatically called when an object is first created. Similarly when an object is destroyed a function called destructor automatically gets called. A destructor has the same name as the constructor (which is the same as the class name) but is preceded by a tilde.

### 16. What are the static members and static member functions?

When a variable in a class is declared static, space for it is allocated for the lifetime of the program. No matter how many objects of that class have been created, there is only one copy of the static member. So same static member can be accessed by all the objects of that class.

A static member function can be called even if no objects of the class exist and the static function are accessed using only the class name and the scope resolution operator ::

### 17. Explain inheritance

Inheritance is the process of creating new classes, called derived classes, from existing classes. These existing classes are called base classes. The derived classes inherit all the capabilities of the base class but can add new features and refinements of their own.

### 18. What is a copy constructor?

A copy constructor is a member function that initializes an object using another object of the same class.

### 19. What is the difference between virtual functions and pure virtual functions?

A virtual function is a member function in the base class that you redefine in a derived class. It is declared using the virtual keyword.

A pure virtual function is a function that has no implementation and is declared by assigning 0. It has no body.

### 20.  Can we call a virtual function from a constructor?

Yes, we can call a virtual function from a constructor. But the behavior is a little different in this case. When a virtual function is called, the virtual call is resolved at runtime. It is always the member function of the current class that gets called. That is the virtual machine doesn’t work within the constructor.

### 21.  Can we call a virtual function from a constructor?

Yes, we can call a virtual function from a constructor. But the behavior is a little different in this case. When a virtual function is called, the virtual call is resolved at runtime. It is always the member function of the current class that gets called. That is the virtual machine doesn’t work within the constructor.

### 22. How do you allocate and deallocate memory in C++?

The new operator is used for memory allocation and deletes operator is used for memory deallocation in C++.

### 23. 1. What is C++? What are the advantages of C++?

C++ is an object-oriented programming language that was introduced to overcome the jurisdictions where C was lacking. By object-oriented we mean that it works with the concept of [*polymorphism*](https://www.geeksforgeeks.org/polymorphism-in-c/)*,*[*inheritance*](https://www.geeksforgeeks.org/inheritance-in-c/)*,*[*abstraction*](https://www.geeksforgeeks.org/abstraction-in-c/)*,*[*encapsulation*](https://www.geeksforgeeks.org/encapsulation-in-c/)*,*[*object, and class*](https://www.geeksforgeeks.org/c-classes-and-objects/)*.*

***Advantages of C++****:*

1. C++ is an OOPs language that means the data is considered as objects.
2. C++ is a multi-paradigm language; In simple terms, it means that we can program the logic, structure, and procedure of the program.
3. Memory management is a key feature in C++ as it enables dynamic memory allocation
4. It is a Mid-Level programming language which means it can develop games, desktop applications, drivers, and kernels

### 24. ****Define ‘std’?****

‘**std’**is also known as Standard or it can be interpreted as a namespace. The command “*using namespace std”*informs the compiler to add everything under the *std namespace* and inculcate them in the *global namespace*. This all inculcation of global namespace benefits us to use “**cout**” and “**cin”**without using “**std::\_operator\_”.**

For more information, refer to [namespace and std](https://www.geeksforgeeks.org/why-it-is-important-to-write-using-namespace-std-in-cpp-program/).

### 25. ****What do you mean by Call by Value and Call by Reference?****

| **Call by Value** | **Call by Reference** |
| --- | --- |
| A copy of a variable is passed. | A variable itself is passed fundamentally. |
| Calling a function by sending the values by copying variables. | Calling a function by sending the address of the passed variable. |
| The changes made in the function are never reflected outside the function on the variable. In short, the original value is never altered in Call by Value. | The changes made in the functions can be seen outside the function on the passed function. In short, the original value is altered in Call by reference. |
| Passed actual and formal parameters are stored in different memory locations. Therefore, making Call by Value a little memory insufficient | Passed actual and formal parameters are stored in the same memory location. Therefore, making Call by Reference a little more memory efficient. |

**26 What is the difference between C and C++?**

| **C** | **C++** |
| --- | --- |
| It is a procedural programming language. In simple words, it does not support classes and objects | It is a mixture of both procedural and object-oriented programming languages. In simple words, it supports classes and objects. |
| It does not support any OOPs concepts like polymorphism, data abstraction, encapsulation, classes, and objects. | It supports all concepts of data |
| It does not support Function and Operator Overloading | It supports Function and Operator Overloading respectively |
| It is a function-driven language | It is an object-driven language |

**27 . What is the difference between struct and class?**

| **Struct** | **Class** |
| --- | --- |
| Members of the struct are always by default public mode | Members of the class can be in private, protected, and public modes. |
| Structures are of the value type. They only hold value in memory. | Classes are of reference type. It holds a reference of an object in memory. |
| The memory in structures is stored as stacks | The memory in classes is stored as heaps. |

**28 What is the difference between reference and pointer?**

| **Reference** | **Pointer** |
| --- | --- |
| The value of a reference cannot be reassigned | The value of a pointer can be reassigned |
| It can never hold a *null*value as it needs an existing value to become an alias of | It can hold or point at a *null* value and be termed as a *nullptr* or *null pointer* |
| It cannot work with arrays | It can work with arrays |
| To access the members of class/struct it uses a ‘ **.**‘ | To access the members of class/struct it uses a ‘ **->**‘ |
| The memory location of reference can be accessed easily or it can be used directly | The memory location of a pointer cannot be accessed easily as we have to use a dereference ‘ **\***‘ |

**29. What is the difference between function overloading and operator overloading?**

| **Function Overloading** | **Operator Overloading** |
| --- | --- |
| It is basically defining a function in numerous ways such that there are many ways to call it or in simple terms you have multiple versions of the same function | It is basically giving practice of giving a special meaning to the existing meaning of an operator or in simple terms redefining the pre-redefined meaning |
| Parameterized Functions are a good example of Function Overloading as just by changing the argument or parameter of a function you make it useful for different purposes | Polymorphism is a good example of an operator overloading as an object of allocations class can be used and called by different classes for different purposes |
| Example of Function Overloading:   1. int GFG(int X, int Y); 2. int GFG(char X, char Y); | Example of Operator Overloading:   1. int GFG() = X() + Y(); 2. int GFG() = X() – Y(); |

**30. What is the difference between an array and a list?**

| **Arrays** | **Lists** |
| --- | --- |
| Array are contiguous memory locations of homogenous data types stored in a fixed location or size. | Lists are classic individual elements that are linked or connected to each other with the help of pointers and do not have a fixed size. |
| Arrays are static in nature. | Lists are dynamic in nature |
| Uses less memory than linked lists. | Uses more memory as it has to store the value and the pointer memory location |

**31. What is the difference between a while loop and a do-while loop?**

| **While Loop** | **do-while Loop** |
| --- | --- |
| While loop is also termed an entry-controlled loop | The do-while loop is termed an exit control loop |
| If the condition is not satisfied the statements inside the loop will not execute | Even if the  condition is not satisfied the statements inside the loop will execute for at least one time |
| **Example**of a while loop:  while(condition)  {statements to be executed;}; | **Example**of a do-while loop:  do {  statements to be executed;  } while(condition or expression); |

### 32.  What are the various OOPs concepts in C++?

* ***Classes*:** It is a user-defined datatype
* ***Objects*:**It is an instance of a class
* **Abstraction:** It is a technique of showing only necessary details
* **Encapsulation:** Wrapping of data in a single unit
* **Inheritance:** The capability of a class to derive properties and characteristics from another class
* **Polymorphism:**Polymorphism is known as many forms of the same thing.

**33. Compare compile-time polymorphism and Runtime polymorphism**

| **Compile-Time Polymorphism** | **Runtime Polymorphism** |
| --- | --- |
| It is also termed static binding and early binding. | It is also termed Dynamic binding and Late binding. |
| It is fast because execution is known early at compile time. | It is slow as compared to compile-time because execution is known at runtime. |
| It is achieved by function overloading and operator overloading. | It is achieved by virtual functions and function overriding. |

### 34.  What are destructors in C++?

Destructors are members of functions in a class that delete an object when an object of the class goes out of scope. Destructors have the same name as the class preceded by a tilde (~) sign. Also, destructors follow a **down-to-top**approach, unlike constructors which follow a top-to-down.

**Syntax:**

~constructor\_name(); // tilde sign signifies that it is a destructor