C++ Programmer Job Interview Questions And Answers

Interview Questions Answers

https://interviewquestionsanswers.org/

About Interview Questions Answers

C++ Programmer Interview Questions And Answers Guide.

Question - 1:
Tell us what do you mean by internal linking and external linking in c++?

Ans:
A symbol is said to be linked internally when it can be accessed only from with-in the scope of a single translation unit. By external linking a symbol can be accessed from other translation units as well. This linkage can be controlled by using static and extern keywords.

View All Answers

Question - 2:
Explain what do you mean by pure virtual functions in C++?

Ans:
Pure virtual function is a function which doesn't have an implementation and the same needs to be implemented by the the next immediate non-abstract class. (A class will become an abstract class if there is at-least a single pure virtual function and thus pure virtual functions are used to create interfaces in c++).

View All Answers

Question - 3:
Tell me what is an abstract class in C++?

Ans:
A class with at least one pure virtual function is called as abstract class. We cannot instantiate an abstract class.

View All Answers

Question - 4:
Do you know the purpose of the keyword volatile?

Ans:
Declaring a variable volatile directs the compiler that the variable can be changed externally. Hence avoiding compiler optimization on the variable reference.

View All Answers

Question - 5:
Tell me what is this pointer?

Ans:
The 'this' pointer is passed as a hidden argument to all nonstatic member function calls and is available as a local variable within the body of all nonstatic functions. 'this' pointer is a constant pointer that holds the memory address of the current object. 'this' pointer is not available in static member functions as static member functions can be called without any object (with class name).

View All Answers

Question - 6:
Explain what is the role of protected access specifier?

Ans:
If a class member is protected then it is accessible in the inherited class. However, outside the both the private and protected members are not accessible.

View All Answers

Question - 7:
Please explain what is a reference variable in C++?

Ans:
A reference variable is an alias name for the existing variable. Which mean both the variable name and reference variable point to the same memory location. Therefore updation on the original variable can be achieved using reference variable too.

View All Answers

Question - 8:
What is an inline function in C++?

Ans:
A function prefixed with the keyword inline before the function definition is called as inline function. The inline functions are faster in execution when compared to normal functions as the compiler treats inline functions as macros.
View All Answers

Question - 9:
Tell me what do you mean by persistent and non persistent objects?

Ans:
Persistent objects are the ones which we can be serialized and written to disk, or any other stream. So before stopping your application, you can serialize the object and on restart you can deserialize it. [Drawing applications usually use serializations.]
Objects that can not be serialized are called non persistent objects. [Usually database objects are not serialized because connection and session will not be existing when you restart the application.]
View All Answers

Question - 10:
Tell me the types of inheritance supported in C++?

Ans:
* Single,
* Multilevel,
* Multiple,
* Hierarchical and
* Hybrid.
View All Answers

Question - 11:
Explain me the storage classes names in C++?

Ans:
The following are storage classes supported in C++
* auto,
* static,
* extern,
* register and
* mutable
View All Answers

Question - 12:
Tell me the default standard streams in C++?

Ans:
* cin,
* cout,
* cerr and
* clog.
View All Answers

Question - 13:
Do you know what is encapsulation?

Ans:
The process of binding the data and the functions acting on the data together in an entity (class) called as encapsulation.
View All Answers

Question - 14:
Please explain the volatile and mutable keywords?

Ans:
The volatile keyword informs the compiler that a variable may change without the compiler knowing it. Variables that are declared as volatile will not be cached by the compiler, and will thus always be read from memory.
The mutable keyword can be used for class member variables. Mutable variables are allowed to change from within const member functions of the class.
View All Answers

Question - 15:
Explain void* realloc (void* ptr, size_t size)?

Ans:

This function is used to change the size of memory object pointed by address ptr to the size given by size. If ptr is a null pointer, then realloc will behave like malloc(). If the ptr is an invalid pointer, then defined behaviour may occur depending the implementation. Undefined behaviour may occur if the ptr has previously been deallocated by free(), or dealloc() or ptr do not match a pointer returned by an malloc(), calloc() or realloc().

View All Answers

Question - 16:

Explain me what is implicit conversion/coercion in c++?

Ans:

Implicit conversions are performed when a type (say T) is used in a context where a compatible type (Say F) is expected so that the type T will be promoted to type F.
short a = 2000 + 20;
In the above example, variable a will get automatically promoted from short to int. This is called implicit conversion/coercion in c++.

View All Answers

Question - 17:

Explain me what is virtual destructors? Why they are used?

Ans:

Virtual destructors are used for the same purpose as virtual functions. When you remove an object of subclass, which is referenced by a parent class pointer, only destructor of base class will get executed. But if the destructor is defined using virtual keyword, both the destructors [of parent and sub class] will get invoked.

View All Answers

Question - 18:

Do you know what is an object?

Ans:

An instance of the class is called as object.

View All Answers

Question - 19:

Tell me what is the block scope variable in C++?

Ans:

A variable whose scope is applicable only within a block is said so. Also a variable in C++ can be declared anywhere within the block.

View All Answers

Question - 20:

Tell me what is the full form of OOPS?

Ans:

Object Oriented Programming System.

View All Answers

Question - 21:

Tell us what is a storage class?

Ans:

A class that specifies the life and scope of its variables and functions is called a storage class.
In C++ following the storage classes are supported: auto, static, register, extern, and mutable.
Note, however, that the keyword register was deprecated in C++11. In C++17, it was removed and reserved for future use.

View All Answers

Question - 22:

Do you know which access specifier/s can help to achive data hiding in C++?

Ans:

Private & Protected.

View All Answers

Question - 23:

Tell me do we have a String primitive data type in C++?

Ans:

No, it's a class from STL (Standard template library).

View All Answers

Question - 24:

What is a storage class in C++?

Ans:

Storage class specifies the life or scope of symbols such as variable or functions.
View All Answers

Question - 25:
Tell me what is abstraction?

Ans:
Abstraction refers to hiding the internal implementation and exhibiting only the necessary details.
View All Answers

Question - 26:
Explain me what is meant by reference variable in C++?

Ans:
In C++, reference variable allows you create an alias (second name) for an already existing variable. A reference variable can be used to access (read/write) the original data. That means, both the variable and reference variable are attached to same memory location. In effect, if you change the value of a variable using reference variable, both will get changed (because both are attached to same memory location).
View All Answers

Question - 27:
Tell us in how many ways we can initialize an int variable in C++?

Ans:
In c++, variables can be initialized in two ways, the traditional C++ initialization using "=" operator and second using the constructor notation.
Traditional C++ initilization
int i = 10;
variable i will get initialized to 10.
Using C++ constructor notation
int i(10);
View All Answers

Question - 28:
Explain me what do you mean by translation unit in c++?

Ans:
We organize our C++ programs into different source files (.cpp, .cxx etc). When you consider a source file, at the preprocessing stage, some extra content may get added to the source code (for example, the contents of header files included) and some content may get removed (for example, the part of the code in the #ifdef of #ifndef block which resolve to false/0 based on the symbols defined). This effective content is called a translation unit. In other words, a translation unit consists of
* Contents of source file
* Plus contents of files included directly or indirectly
* Minus source code lines ignored by any conditional pre processing directives (the lines ignored by #ifdef,#ifndef etc)
View All Answers

Question - 29:
Tell me when you should use virtual inheritance?

Ans:
While it's ideal to avoid virtual inheritance altogether (you should know how your class is going to be used) having a solid understanding of how virtual inheritance works is still important:
So when you have a class (class A) which inherits from 2 parents (B and C), both of which share a parent (class D), as demonstrated below:
```
#include <iostream>
class D {
public:
    void foo() {
        std::cout << "Foooooo" << std::endl;
    }
};
class C:  public D {
};
class B:  public D {
};
class A: public B, public C {
};
int main(int argc, const char * argv[]) {
    A a;
    a.foo();
}
```
If you don't use virtual inheritance in this case, you will get two copies of D in class A: one from B and one from C. To fix this you need to change the declarations of classes C and B to be virtual, as follows:
```
class C:  virtual public D {
};
class B:  virtual public D {
};
```
View All Answers

Question - 30:

Tell me which compiler switch to be used for compiling the programs using math library with g++ compiler?

Ans:

Opiton -lm to be used as > g++ -lm <file.cpp>

View All Answers

Question - 31:

Tell us does C++ supports exception handling? If so what are the keywords involved in achieving the same?

Ans:

C++ does supports exception handling. try, catch & throw are keyword used for the same.

View All Answers

Question - 32:

Explain me what do you mean by C++ access specifiers?

Ans:

Access specifiers are used to define how the members (functions and variables) can be accessed outside the class. There are three access specifiers defined which are public, private, and protected
* private:
Members declared as private are accessible only with in the same class and they cannot be accessed outside the class they are declared.
* public:
Members declared as public are accessible from any where.
* protected:
Members declared as protected can not be accessed from outside the class except a child class. This access specifier has significance in the context of inheritance.

View All Answers

Question - 33:

Please explain what is difference between shallow copy and deep copy? Which is default?

Ans:

When you do a shallow copy, all the fields of the source object is copied to target object as it is. That means, if there is a dynamically created field in the source object, shallow copy will copy the same pointer to target object. So you will have two objects with fields that are pointing to same memory location which is not what you usually want.
In case of deep copy, instead of copying the pointer, the object itself is copied to target. In this case if you modify the target object, it will not affect the source. By default copy constructors and assignment operators do shallow copy. To make it as deep copy, you need to create a custom copy constructor and override assignment operator.

View All Answers

Question - 34:

Explain me are you allowed to have a static const member function?

Ans:

A const member function is one which isn't allowed to modify the members of the object for which it is called. A static member function is one which can't be called for a specific object.
Thus, the const modifier for a static member function is meaningless, because there is no object associated with the call.
A more detailed explanation of this reason comes from the C programming language. In C, there were no classes and member functions, so all functions were global. A member function call is translated to a global function call. Consider a member function like this:
void foo(int i);
A call like this:
obj.foo(10);
...is translated to this:
foo(&obj, 10);
This means that the member function foo has a hidden first argument of type T*:
void foo(T* const this, int i);
If a member function is const, this is of type const T* const this:
void foo(const T* const this, int i);
Static member functions don't have such a hidden argument, so there is no this pointer to be const or not.

View All Answers

Question - 35:

Explain me which operator can be used in C++ to allocate dynamic memory?

Ans:

'new' is the operator can be used for the same.

View All Answers

Question - 36:

Tell me what is a class?

Ans:

Class defines a datatype, it's type definition of category of thing(s). But a class actually does not define the data, it just specifies the structure of data. To use them you need to create objects out of the class. Class can be considered as a blueprint of a building, you can not stay inside blueprint of building, you need to construct building(s) out of that plan. You can create any number of buildings from the blueprint, similarly you can create any number of objects from a class.

```
class Vehicle
{
   public:
      int numberOfTyres;
      double engineCapacity;
      void drive(){
         // code to drive the car
      }
};
```

View All Answers

Question - 37:

Tell me what are C++ inline functions?

Ans:

C++ inline functions are special functions, for which the compiler replaces the function call with body/definition of function. Inline functions makes the program execute faster than the normal functions, since the overhead involved in saving current state to stack on the function call is avoided. By giving developer the control of making a function as inline, he can further optimize the code based on application logic. But actually, it's the compiler that decides whether to make a function inline or not regardless of it's declaration. Compiler may choose to make a non inline function inline and vice versa. Declaring a function as inline is in effect a request to the compiler to make it inline, which compiler may ignore. So, please note this point for the interview that, it is upto the compiler to make a function inline or not.

```
inline int min(int a, int b)
{
   return (a < b)? a : b;
}

int main( )
{
   cout << "min (20,10): " << min(20,10) << endl;
   cout << "min (0,200): " << min(0,200) << endl;
   cout << "min (100,1010): " << min(100,1010) << endl;
   return 0;
}
```

If the complier decides to make the function min as inline, then the above code will internally look as if it was written like

```
int main( )
{
   cout << "min (20,10): " << ((20 < 10)? 20 : 10) << endl;
   cout << "min (0,200): " << ((0 < 200)? 0 : 200) << endl;
   cout << "min (100,1010): " << ((100 < 1010)? 100 : 1010) << endl;
   return 0;
}
```

View All Answers

Question - 38:

Explain what is 'Copy Constructor' and when it is called?

Ans:

This is a frequent c++ interview question. Copy constructor is a special constructor of a class which is used to create copy of an object. Compiler will give a default copy constructor if you don't define one. This implicit constructor will copy all the members of source object to target object.

Implicit copy constructors are not recommended, because if the source object contains pointers they will be copied to target object, and it may cause heap corruption when both the objects with pointers referring to the same location does an update to the memory location. In this case its better to define a custom copy constructor and do a deep copy of the object.

```
      class SampleClass{
         public:
            int* ptr;
            SampleClass();
            // Copy constructor declaration
            SampleClass(SampleClass &obj);
      };

      SampleClass::SampleClass(){
         ptr = new int();
         *ptr = 5;
      }

      // Copy constructor definition
      SampleClass::SampleClass(SampleClass &obj){
         //create a new object for the pointer
         ptr = new int();
         // Now manually assign the value
         *ptr = *(obj.ptr);
         cout<<"Copy constructor...n";
      }
```

View All Answers

Question - 39:

Tell us why pure virtual functions are used if they don't have implementation / When does a pure virtual function become useful?

Ans:

Pure virtual functions are used when it doesn't make sense to provide definition of a virtual function in the base class or a proper definition does not exists in the context of base class. Consider the above example, class SymmetricShape is used as base class for shapes with symmetric structure(Circle, square, equilateral triangle etc). In this case, there exists no proper definition for function draw() in the base class SymmetricShape instead the child classes of SymmetricShape (Cirlce, Square etc) can implement this method and draw proper shape.

View All Answers

Question - 40:

Explain how to create a reference variable in C++?

Ans:

Appending an ampersand (&) to the end of datatype makes a variable eligible to use as reference variable.

```
int a = 20;
int& b = a;
```

The first statement initializes a an integer variable a. Second statement creates an integer reference initialized to variable a
Take a look at the below example to see how reference variables work.

```
int main ()
{
 int   a;
 int&  b = a;

 a = 10;
 cout << "Value of a : " << a << endl;
 cout << "Value of a reference (b) : " << b  << endl;

 b = 20;
 cout << "Value of a : " << a << endl;
 cout << "Value of a reference (b) : " << b  << endl;

 return 0;
}
```

Above code creates following output.

```
Value of a : 10
Value of a reference (b) : 10
Value of a : 20
Value of a reference (b) : 20
```

View All Answers

Question - 41:

Explain me what will i and j equal after the code below is executed? Explain your answer.
int i = 5;
int j = i++;?

Ans:

After the above code executes, i will equal 6, but j will equal 5.
Understanding the reason for this is fundamental to understanding how the unary increment (++) and decrement (--) operators work in C++.
When these operators precede a variable, the value of the variable is modified first and then the modified value is used. For example, if we modified the above code snippet to instead say int j = ++i;, i would be incremented to 6 and then j would be set to that modified value, so both would end up being equal to 6.
However, when these operators follow a variable, the unmodified value of the variable is used and then it is incremented or decremented. That's why, in the statement int j = i++; in the above code snippet, j is first set to the unmodified value of i (i.e., 5) and then i is incremented to 6.

View All Answers

Question - 42:

Tell us how to make sure a C++ function can be called as e.g. void foo(int, int) but not as any other type like void foo(long, long)?

Ans:

Implement foo(int, int)...

```
void foo(int a, int b) {
// whatever
}
```

...and delete all others through a template:

```
template <typename T1, typename T2> void foo(T1 a, T2 b) = delete;
```

Or without the delete keyword:

```
template <class T, class U>
void f(T arg1, U arg2);
template <>
void f(int arg1, int arg2)
{
   //...
}
```

View All Answers

Question - 43:

Tell me how can a C function be called in a C++ program?

Ans:

Using an extern "C" declaration:

```
//C code
void func(int i)
{
//code
}
void print(int i)
{
//code
}
//C++ code
extern "C"{
void func(int i);
void print(int i);
}
void myfunc(int i)
{
   func(i);
   print(i);
}
```

View All Answers

Question - 44:

Do you know what is the role of mutable storage class specifier?

Ans:

A constant class object's member variable can be altered by declaring it using mutable storage class specifier. Applicable only for non-static and non-constant member variable of the class.

View All Answers

Question - 45:

Tell us can we use malloc() function of C language to allocate dynamic memory in C++?

Ans:

Yes, as C is the subset of C++, we can all the functions of C in C++ too.

View All Answers

Question - 46:

Do you know who designed C++ programming language?

Ans:

Bjarne Stroustrup.

View All Answers

Question - 47:

Explain is it possible to have a recursive inline function?

Ans:

Although you can call an inline function from within itself, the compiler may not generate inline code since the compiler cannot determine the depth of recursion at compile time. A compiler with a good optimizer can inline recursive calls till some depth fixed at compile-time (say three or five recursive calls), and insert non-recursive calls at compile time for cases when the actual depth gets exceeded at run time.

View All Answers

Question - 48:

Tell me is it possible to get the source code back from binary file?

Ans:

Technically it is possible to generate the source code from binary. It is called reverse engineering. There are lot of reverse engineering tools available. But, in actual case most of them will not re generate the exact source code back because many information will be lost due to compiler optimization and other interpretations.

View All Answers

Question - 49:

Explain what do you mean by storage classes?

Ans:

Storage class are used to specify the visibility/scope and life time of symbols(functions and variables). That means, storage classes specify where all a variable or function can be accessed and till what time those variables will be available during the execution of program.

View All Answers

Question - 50:

Tell us what is the use of volatile keyword in c++? Give an example?

Ans:

Most of the times compilers will do optimization to the code to speed up the program. For example in the below code,

```
int a = 10;
while( a == 10){
    // Do something
}
```

compiler may think that value of 'a' is not getting changed from the program and replace it with 'while(true)', which will result in an infinite loop. In actual scenario the value of 'a' may be getting updated from outside of the program.
Volatile keyword is used to tell compiler that the variable declared using volatile may be used from outside the current scope so that compiler wont apply any optimization. This matters only in case of multi-threaded applications.
In the above example if variable 'a' was declared using volatile, compiler will not optimize it. In shot, value of the volatile variables will be read from the memory location directly.

View All Answers

Question - 51:

Tell me what is difference between C and C++?

Ans:

* C++ is Multi-Paradigm (not pure OOP, supports both procedural and object oriented) while C follows procedural style programming.
* In C data security is less, but in C++ you can use modifiers for your class members to make it inaccessible from outside.
* C follows top-down approach (solution is created in step by step manner, like each step is processed into details as we proceed) but C++ follows a bottom-up approach (where base elements are established first and are linked to make complex solutions).
* C++ supports function overloading while C does not support it.
* C++ allows use of functions in structures, but C does not permit that.
* C++ supports reference variables (two variables can point to same memory location). C does not support this.
* C does not have a built in exception handling framework, though we can emulate it with other mechanism. C++ directly supports exception handling, which makes life of developer easy.

View All Answers

Question - 52:

Tell me does an abstract class in C++ need to hold all pure virtual functions?

Ans:

Not necessarily, a class having at least one pure virtual function is abstract class too.

View All Answers

Question - 53:

Explain what is inheritance?

Ans:

Inheritance is the process of acquiring the properties of the exiting class into the new class. The existing class is called as base/parent class and the inherited class is called as derived/child class.

View All Answers

Question - 54:

Explain what are VTABLE and VPTR?

Ans:

vtable is a table of function pointers. It is maintained per class.
vptr is a pointer to vtable. It is maintained per object (See this for an example).
Compiler adds additional code at two places to maintain and use vtable and vptr.
1) Code in every constructor. This code sets vptr of the object being created. This code sets vptr to point to vtable of the class.
2) Code with polymorphic function call (e.g. bp->show() in above code). Wherever a polymorphic call is made, compiler inserts code to first look for vptr using base class pointer or reference (In the above example, since pointed or referred object is of derived type, vptr of derived class is accessed). Once vptr is fetched, vtable of derived class can be accessed. Using vtable, address of derived derived class function show() is accessed and called.

View All Answers

Question - 55:

Please explain is there a difference between class and struct?

Ans:

The only difference between a class and struct are the access modifiers. Struct members are public by default; class members are private. It is good practice to use classes when you need an object that has methods and structs when you have a simple data object.

View All Answers

Question - 56:

Tell me what will the line of code below print out and why?

Ans:

```
#include <iostream>
int main(int argc, char **argv)
{
   std::cout << 25u - 50;
   return 0;
}
```

View All Answers

Question - 57:
Tell me how to create a pure virtual function?

Ans:
A function is made as pure virtual function by the using a specific signature, " = 0" appended to the function declaration as given below,

```
class SymmetricShape {
   public:
      // draw() is a pure virtual function.
      virtual void draw() = 0;
};
```

View All Answers

Question - 58:
Explain void free (void* ptr)?

Ans:
This function is used to deallocate a block of memory that was allocated using malloc(), calloc() or realloc(). If ptr is null, this function does not doe anything.

View All Answers

Question - 59:
Explain me what is an Object/Instance?

Ans:
Object is the instance of a class, which is concrete. From the above example, we can create instance of class Vehicle as given below
Vehicle vehicleObject;
We can have different objects of the class Vehicle, for example we can have Vehicle objects with 2 tyres, 4tyres etc. Similarly different engine capacities as well.

View All Answers

C++ Most Popular & Related Interview Guides