C++ Interview Questions

Q: Is it possible to have Virtual Constructor? If yes, how? If not, Why not possible?
Q: What is constructor or ctor?
Q: What about Virtual Destructor?
Q: What is the difference between a copy constructor and an overloaded assignment operator?
Q: Can a constructor throws an exception? How to handle the error when the constructor fails?
Q: What is default constructor?
Q: What is copy constructor?
Q: When are copy constructors called?
Q: Can a copy constructor accept an object of the same class as parameter, instead of reference of the object? If yes why ? If no why?
Q: What is conversion constructor?
Q: What is conversion operator??
Q: What is Virtual Destructor?
Q: Can I overload the destructor for my class?
Q: Should my constructors use "initialization lists" or "assignment"?
Q: Should you use the this pointer in the constructor?
Q: What is virtual function?
Q: What is a "pure virtual" member function?
Q: How virtual functions are implemented C++?
Q: What is pure virtual function? or what is abstract class?
Q: What is Pure Virtual Function? Why and when it is used?
Q: How Virtual functions call up is maintained?
Q: What is inheritance?
Q: When should you use multiple inheritance?
Q: Explain the ISA and HASA class relationships. How would you implement each in a class design?
Q: What is multiple inheritance(virtual inheritance)? What are its advantages and disadvantages?
Q: What is Polymorphism??
Q: What is a class?
Q: What are the differences between a C++ struct and C++ class?
Q: How do you know that your class needs a virtual destructor?
Q: What is encapsulation?
Q: What is "this" pointer?
Q: What happens when you make call "delete this;"?
Q: What is assignment operator?
Q: What are all the implicit member functions of the class? Or what are all the functions which compiler implements for us if we don't define one?
A:
(a) default ctor
(b) copy ctor
(c) assignment operator
(d) default destructor
(e) address operator

Q: What is Overriding?
Q: How do you access the static member of a class?
Q: What is the difference between new/delete and malloc/free?
Q: What is difference between new and malloc?
Q: What is the difference between delete and delete[]?
Q: What is difference between malloc()/free() and new/delete?
Q: What are the two steps that happen when I say delete p?
Q: What should I throw?
A: C++, unlike just about every other language with exceptions, is very accomodating when it comes to what you can throw. In fact, you can throw anything you like. That begs the question then, what should you throw?
Generally, it's best to throw objects, not built-ins. If possible, you should throw instances of classes that derive (ultimately) from the std::exception class. By making your exception class inherit (ultimately) from the standard exception base-class, you are making life easier for your users (they have the option of catching most things via std::exception), plus you are probably providing them with more information (such as the fact that your particular exception might be a refinement of std::runtime_error or whatever).
The most common practice is to throw a temporary:

```
#include
class MyException : public std::runtime_error {
public:
MyException() : std::runtime_error("MyException") { }
};
void f()
{
// ...
throw MyException();
}
```

Here, a temporary of type MyException is created and thrown. Class MyException inherits from class std::runtime_error which (ultimately) inherits from class std::exception.
Q: What should I catch?
A: In keeping with the C++ tradition of "there's more than one way to do that" (translation: "give programmers options and tradeoffs so they can decide what's best for them in their situation"), C++ allows you a variety of options for catching.
You can catch by value.
You can catch by reference.
You can catch by pointer.
In fact, you have all the flexibility that you have in declaring function parameters, and the rules for whether a particular exception matches (i.e., will be caught by) a particular catch clause are almost exactly the same as the rules for parameter compatibility when calling a function.
Given all this flexibility, how do you decide what to catch? Simple: unless there's a good reason not to, catch by reference. Avoid catching by value, since that causes a copy to be made and the copy can have different behavior from what was thrown. Only under very special circumstances should you catch by pointer.

Q: What is a dangling pointer?

Q: What is Memory Leak?
Q: What is a smart pointer?
Q: Is there any problem with the following : char*a=NULL; char& p = *a;?
Q: What is the difference between a pointer and a reference?
Q: What is the difference between const char *myPointer and char *const myPointer?
Q: When should I use references, and when should I use pointers?
Q: Advantages of References over pointers.
Q: Why references are considered safe?
Q: What are the access privileges in C++? What is the default access level?
Q: Name the operators that cannot be overloaded?
A:sizeof, ., .*, .->, ::, ?:
Q: What is overloading??
Q: How are prefix and postfix versions of operator++() differentiated?
Q: Can you overload a function based only on whether a parameter is a value or a reference?
Q: What's the deal with operator overloading?
Q: What are the benefits of operator overloading?
Q: What are some examples of operator overloading?
Q: What operators can/cannot be overloaded?
Q: What is a friend?
Q: Do friends violate encapsulation?
Q: What are some advantages/disadvantages of using friend functions?
Q: Can inline functions have a recursion?
A: No. Syntax wise it is allowed. But then the function is no longer Inline. As the compiler will never know how deep the recursion is at compilation time.
Q: Explain the scope resolution operator?
A: It 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.
Q: When do use "const" reference arguments in function?
Q: When can const be applied to right side of function?
Q: What problem does the namespace feature solve?
Q: What is name mangling in C++??
Q: What is Realisation and Specialisation. Give Example?
Q: When is containment? Explain Aggregation and composition with examples.
Q: In copy constructors what is concept of deep and shallow copy?
Q: Explain need of writing your own copy constructor?
Q: What problem does the namespace feature solve?
Q: What is name mangling in C++??