**TOP 100 C++ INTERVIEW QUESTIONS AND ANSWERS**

**Basic C++ Questions**

1. **What is C++?**  
   ***Answer:***  C++ is a general-purpose, object-oriented programming language developed by Bjarne Stroustrup.
2. **What are the main features of C++?**  
   ***Answer:***  Object-oriented programming, polymorphism, inheritance, encapsulation, abstraction, and templates.
3. **What is the difference between C and C++?**  
   ***Answer:***  C is procedural, while C++ supports both procedural and object-oriented programming.
4. **What is a namespace in C++?**  
   ***Answer:***  A namespace organizes code into logical groups and prevents name conflicts.
5. **Explain the basic structure of a C++ program.**  
   ***Answer:***  Includes headers (#include), a main() function, and optional user-defined functions and classes.

**OOP Concepts**

1. **What is object-oriented programming (OOP)?**  
   ***Answer:***  A programming paradigm based on objects that contain data and methods.
2. **What are the four pillars of OOP?**  
   ***Answer:***  Encapsulation, inheritance, polymorphism, and abstraction.
3. **What is a class?**  
   ***Answer:***  A blueprint for creating objects, defining data members and methods.
4. **What is an object?**  
   ***Answer:***  An instance of a class that holds data and can perform functions.
5. **What is encapsulation?**  
   ***Answer:***  Bundling data and methods together, restricting access using access specifiers (private, protected, public).

**Constructors and Destructors**

1. **What is a constructor?**  
   ***Answer:***  A special function that initializes objects when they are created.
2. **What are the types of constructors?**  
   ***Answer:***  Default, parameterized, and copy constructors.
3. **What is a destructor?**  
   ***Answer:***  A function that is called automatically when an object is destroyed.
4. **Can a constructor be virtual?**  
   ***Answer:***  No, constructors cannot be virtual.
5. **What is the use of the this pointer?**  
   ***Answer:***  It refers to the current object instance.

**Inheritance**

1. **What is inheritance?**  
   ***Answer:***  A mechanism by which one class inherits properties and methods from another.
2. **What are the types of inheritance?**  
   ***Answer:***  Single, multiple, multilevel, hierarchical, and hybrid inheritance.
3. **What is the syntax of inheritance?**  
   ***Answer:***  class Derived : public Base { };
4. **What is the difference between public, private, and protected inheritance?**  
   ***Answer:***  Access specifiers control the accessibility of inherited members.
5. **What is a virtual base class?**  
   ***Answer:***  A class that prevents multiple "instances" of a base class in a hierarchy using virtual inheritance.

**Polymorphism**

1. **What is polymorphism?**  
   ***Answer:***  The ability of a function or object to take many forms (overloading and overriding).
2. **What is function overloading?**  
   ***Answer:***  Defining multiple functions with the same name but different parameters.
3. **What is operator overloading?**  
   ***Answer:***  Redefining operators to work with user-defined types.
4. **What is function overriding?**  
   ***Answer:***  Redefining a base class function in a derived class.
5. **What is a virtual function?**  
   ***Answer:***  A function declared with virtual keyword in a base class, allowing dynamic dispatch.

**Abstraction**

1. **What is abstraction in C++?**  
   ***Answer:***  Hiding complex implementation details and exposing only essential features.
2. **How is abstraction achieved in C++?**  
   ***Answer:***  Using abstract classes and interfaces.
3. **What is an abstract class?**  
   ***Answer:***  A class with at least one pure virtual function.
4. **What is a pure virtual function?**  
   ***Answer:***  A function declared with = 0 in its signature, making the class abstract.
5. **Can we instantiate an abstract class?**  
   ***Answer:***  No, abstract classes cannot be instantiated.

**Memory Management**

1. **What is dynamic memory allocation in C++?**  
   ***Answer:***  Allocating memory at runtime using new and delete.
2. **What is the difference between malloc() and new?**  
   ***Answer:***  new initializes objects and calls the constructor, while malloc() only allocates memory.
3. **What is the use of the delete operator?**  
   ***Answer:***  It deallocates memory allocated by new.
4. **What is a smart pointer?**  
   ***Answer:***  A pointer that manages the lifetime of an object and deallocates it automatically.
5. **What is a memory leak?**  
   ***Answer:***  It occurs when dynamically allocated memory is not freed.

**Templates and STL**

1. **What is a template in C++?**  
   ***Answer:***  A blueprint for creating generic classes or functions.
2. **What are the types of templates?**  
   ***Answer:***  Function templates and class templates.
3. **What is the Standard Template Library (STL)?**  
   ***Answer:***  A collection of classes and functions for data structures and algorithms.
4. **What are containers in STL?**  
   ***Answer:***  Objects that store data, such as vector, list, and map.
5. **What are iterators in STL?**  
   ***Answer:***  Objects that allow traversing elements in a container.

**Exception Handling**

1. **What is exception handling in C++?**  
   ***Answer:***  A mechanism to handle runtime errors using try, catch, and throw.
2. **What is the syntax of exception handling?**  
   ***Answer:***

try {

// Code that may throw an exception

} catch (ExceptionType e) {

// Handle exception

}

1. **What is the purpose of the throw keyword?**  
   ***Answer:***  It signals the occurrence of an exception.
2. **What is a catch block?**  
   ***Answer:***  A block that handles exceptions thrown by the try block.
3. **What is a generic catch block?**  
   ***Answer:***  catch(...) catches any exception, regardless of its type.
4. **Can a constructor throw an exception?**  
   ***Answer:***  Yes, but the object will not be created.
5. **What is std::exception?**  
   ***Answer:***  A base class in the standard library for all exceptions.
6. **How to create a custom exception class?**  
   ***Answer:***  By inheriting from std::exception and overriding the what() method.

**File Handling**

1. **What is file handling in C++?**  
   ***Answer:***  It allows reading from and writing to files using classes from <fstream>.
2. **What are the file handling classes in C++?**  
   ***Answer:***  ifstream, ofstream, and fstream.
3. **How to open a file in C++?**  
   ***Answer:***  std::ifstream file("example.txt");
4. **How to write to a file?**  
   ***Answer:***

std::ofstream file("output.txt");

file << "Hello, World!";

1. **How to read from a file?**  
   ***Answer:***

std::string line;

std::ifstream file("input.txt");

while (std::getline(file, line)) {

std::cout << line << std::endl;

}

1. **What is the use of file.close()?**  
   ***Answer:***  It closes the file and ensures that all data is written.
2. **What is file mode in C++?**  
   ***Answer:***  It specifies how a file is opened (ios::in, ios::out, ios::app).

**Multithreading**

1. **What is multithreading?**  
   ***Answer:***  The ability to run multiple threads concurrently within a program.
2. **How to create a thread in C++?**  
   ***Answer:***  Using the std::thread class from the <thread> header.
3. **What is a thread function?**  
   ***Answer:***  A function executed by a thread.
4. **How to join a thread?**  
   ***Answer:***  Using the join() method.

thread1.join();

1. **What is thread synchronization?**  
   ***Answer:***  Controlling access to shared resources using mechanisms like mutexes.
2. **What is a mutex?**  
   ***Answer:***  A synchronization tool that prevents concurrent access to shared resources.
3. **What is the difference between join() and detach()?**  
   ***Answer:***  join() waits for a thread to finish, while detach() allows the thread to run independently.
4. **What is a deadlock?**  
   ***Answer:***  A situation where two or more threads wait indefinitely for each other to release resources.
5. **How to avoid deadlocks?**  
   ***Answer:***  By ensuring a consistent locking order or using tools like std::lock.

**STL (Standard Template Library)**

1. **What are the categories of STL components?**  
   ***Answer:***  Containers, iterators, algorithms, and function objects.
2. **What are sequential containers?**  
   ***Answer:***  Containers that store elements in a linear sequence (vector, list, deque).
3. **What are associative containers?**  
   ***Answer:***  Containers that store elements in key-value pairs (map, set).
4. **What is a vector in C++?**  
   ***Answer:***  A dynamic array that can grow or shrink in size.
5. **How to insert elements in a vector?**  
   ***Answer:***  Using push\_back() method.
6. vec.push\_back(10);
7. **What is a map in C++?**  
   ***Answer:***  An associative container that stores key-value pairs in sorted order.
8. **What is the difference between set and multiset?**  
   ***Answer:***  set allows unique elements, while multiset allows duplicates.
9. **What is a priority queue in C++?**  
   ***Answer:***  A container that stores elements in a heap structure.
10. **What are algorithms in STL?**  
    ***Answer:***  Predefined functions that operate on containers, like sort(), find(), and reverse().
11. **What are iterators?**  
    ***Answer:***  Objects that allow traversing elements in a container.

**Best Practices and Miscellaneous**

1. **Why use smart pointers?**  
   ***Answer:***  To automatically manage memory and prevent memory leaks.
2. **What is RAII (Resource Acquisition Is Initialization)?**  
   ***Answer:***  A C++ idiom where resources are tied to object lifetime.
3. **What is the nullptr keyword?**  
   ***Answer:***  It represents a null pointer.
4. **What is a lambda function in C++?**  
   ***Answer:***  An anonymous function defined using [].

auto add = [](int a, int b) { return a + b; };

1. **What is the auto keyword?**  
   ***Answer:***  It allows the compiler to deduce the type of a variable.
2. **What is type casting in C++?**  
   ***Answer:***  Converting one data type to another (static\_cast, dynamic\_cast).
3. **What is a function pointer?**  
   ***Answer:***  A pointer that points to a function.
4. **What is a virtual destructor?**  
   ***Answer:***  A destructor that ensures proper cleanup of derived class objects when deleted through a base class pointer.
5. **What is slicing in C++?**  
   ***Answer:***  When a derived class object is assigned to a base class object, losing the derived part.
6. **What are friend functions?**  
   ***Answer:***  Functions that have access to private members of a class.
7. **What is a shallow copy?**  
   ***Answer:***  A copy that only duplicates an object's memory address, not the actual data.
8. **What is a deep copy?**  
   ***Answer:***  A copy that duplicates both the object and the data it points to.
9. **What is the difference between new and delete?**  
   ***Answer:***  new allocates memory, and delete deallocates it.
10. **What are move semantics in C++?**  
    ***Answer:***  Transferring resources from one object to another using move constructors.
11. **What is the rule of five in C++?**  
    ***Answer:***  A rule suggesting that if a class needs any one of: destructor, copy constructor, copy assignment operator, move constructor, or move assignment operator, it likely needs all five.
12. **What is a constexpr?**  
    ***Answer:***  A constant expression evaluated at compile time.
13. **What is a virtual table (vtable)?**  
    ***Answer:***  A mechanism used to support dynamic (runtime) polymorphism. It’s a table of function pointers maintained per class with virtual functions.
14. **What is the difference between deep copy and move semantics?**  
    ***Answer:***  Deep copy duplicates all data, while move semantics transfer ownership of resources to a new object, leaving the old one in a valid but empty state.
15. **What is placement new?**  
    ***Answer:***  A way to allocate memory at a specific location using the new operator.

int \*p = new(memory) int(42); // Allocates at a given memory address

1. **What is the difference between static and dynamic polymorphism?**  
   ***Answer:***  Static polymorphism (compile-time) is achieved through function overloading and templates, while dynamic polymorphism (runtime) is achieved through virtual functions.
2. **What is CRTP (Curiously Recurring Template Pattern)?**  
   ***Answer:***  A design pattern where a class inherits from a template instantiation of itself.

template <typename T>

class Base {

void method() { static\_cast<T\*>(this)->method(); }

};

1. **What is a final keyword in C++11?**  
   ***Answer:***  It prevents further inheritance of a class or overriding of a virtual function.

class FinalClass final { };

virtual void func() final;

1. **What is two-phase name lookup?**  
   ***Answer:***  A C++ process where names are first checked in the template definition context and later in the instantiation context.
2. **What are rvalue references (&&) in C++?**  
   ***Answer:***  References to temporary objects, enabling move semantics and perfect forwarding.
3. **What is perfect forwarding in C++?**  
   ***Answer:***  Passing arguments to a function as-is (preserving their value category) using std::forward.
4. **What are C++ design patterns?**  
   ***Answer:***  Reusable solutions to common software design problems, such as Singleton, Factory, and Observer patterns.