71 C++ interview questions to ask your applicants

Questions

1. Can you explain the difference between C and C++?
2. What are the main features of object-oriented programming in C++?
3. How does memory management work in C++?
4. What is the purpose of virtual functions in C++?
5. Can you explain the concept of function overloading in C++?
6. What are templates in C++ and why are they useful?
7. How does exception handling work in C++?
8. What is the difference between references and pointers in C++?
9. What are the access specifiers in C++ and how do they affect class members?
10. Can you explain the concept of operator overloading in C++?
11. What is a constructor in C++ and how is it different from a destructor?
12. Can you explain what a copy constructor is and when it is used?
13. What is the difference between deep copy and shallow copy?
14. How would you explain the concept of polymorphism in C++?
15. What is the role of the 'this' pointer in C++?
16. Can you describe what namespaces are and why they are important?
17. What is a friend function and what is its purpose in C++?
18. How do you manage dynamic memory allocation using new and delete in C++?
19. What is the purpose of the 'const' keyword in C++?
20. How does C++ handle multiple inheritance and what are the potential issues?
21. What is the difference between a class and a struct in C++?
22. Can you explain what a pure virtual function is and its role in C++?
23. What are smart pointers and why are they used in C++?
24. How does the Standard Template Library (STL) benefit C++ development?
25. What is RAII (Resource Acquisition Is Initialization) and how does it work?
26. Can you discuss the different types of inheritance in C++?
27. What is the difference between 'overloading' and 'overriding' methods in C++?
28. How do you ensure thread safety in a C++ application?
29. What are some common pitfalls when using pointers in C++?
30. Can you explain the concept of move semantics in C++ and its benefits?
31. Can you explain the concept of function inlining in C++ and when it's beneficial?
32. How does the 'auto' keyword work in C++11 and later versions?
33. What are lambda expressions in C++ and how are they useful?
34. Can you explain the concept of RAII (Resource Acquisition Is Initialization) in C++?
35. What is the difference between std::vector and std::array in C++?
36. How does the 'explicit' keyword affect class constructors in C++?
37. Can you explain what perfect forwarding is in C++ and why it's useful?
38. What is the rule of five in C++ and why is it important?
39. How does std::unique_ptr differ from std::shared_ptr, and when would you use each?
40. Can you explain what a variadic template is in C++ and provide an example of its use?
41. How would you implement a stack using a linked list in C++?
42. Can you explain the difference between std::list and std::vector in C++?
43. What is the time complexity of inserting an element at the beginning of a std::vector versus a std::list?
44. How does a hash table work in C++, and what is its average time complexity for insertion and lookup?
45. Can you describe a scenario where you would choose a binary search tree over a hash table?
46. What is a priority queue in C++, and how is it typically implemented?
47. How would you implement a circular buffer in C++?
48. Can you explain the difference between a min-heap and a max-heap?
49. What is the purpose of the std::set container in C++, and how does it maintain its elements?
50. How would you implement a trie (prefix tree) data structure in C++?
51. Can you describe the internal workings of std::unordered_map in C++?
52. What is the difference between std::map and std::unordered_map in terms of performance and use cases?
53. How would you implement a graph data structure in C++?
54. Can you explain how a B-tree differs from a binary search tree and when you might use it?
55. What is the difference between stack and heap memory allocation in C++?
56. How do you prevent memory leaks in C++?
57. What is a memory leak, and how can it be detected?
58. Can you explain the concept of memory fragmentation and its impact on application performance?
59. What are smart pointers and how do they help in memory management?
60. What is the purpose of the delete operator in C++?
61. Can you explain what a dangling pointer is and how to avoid it?
62. What is the role of the new operator in C++?
63. How can you minimize memory usage in a C++ program?
64. How would you handle a situation where a critical section of your C++ application is running slower than expected?
65. Imagine you have to work on a legacy C++ codebase with little documentation. How would you approach understanding and improving the code?
66. How do you handle a situation where you need to integrate third-party libraries into your C++ project?
67. What steps would you take if you discovered a memory leak in your C++ application?
68. How would you manage version control and collaboration in a C++ project involving multiple developers?
69. Can you describe a scenario where you had to debug a particularly tricky issue in a C++ application? How did you resolve it?
70. How would you ensure the security of a C++ application you're developing?