

## C++ Review Questions

1. What is a type?
2. What is a typedef?
3. What is static typing?
4. What are the builtin types in C++?
5. How do you define an array in C++?
6. What is a pointer and how is one used?
7. How do you define a pointer?
8. How do you dereference a pointer?
9. Is a pointer the same as an array?
10. What is a function prototype declaration?
11. What is a class?
12. What is an object?
13. How do you determine the size of an object?
14. What does the term "scope" mean?
15. How do you define a scope in C++?
16. What are the qualities of an object?
17. What is encapsulation?
18. What is information hiding?
19. How do you achieve encapsulation and information hiding in C++?
20. How is a class better than a struct?
21. What is the difference between a class and a struct?
22. How is a class better than a function?
23. What is a class member function?
24. What is the difference between the public and private sections of a class?
25. From what areas of storage are objects allocated?
26. What is the life-time of a globally allocated object?
27. What is the life-time of a stack allocated object?
28. What is the life-time of a heap allocated object?
29. How do you allocate an object of some class type from the heap?
30. How do you delete an object allocated from the heap?
31. What is the syntax for deleting a heap allocated array of objects?
32. What happens if you forget to delete an object allocated from the heap?
33. What is a memory leak?
34. What is the purpose of a constructor?
35. How and when is a constructor called?
36. What is the purpose of a destructor?
37. How and when is a destructor called?
38. What is the purpose of a copy constructor?
39. How do you define a default constructor?
40. How do you define a copy constructor?
41. What does ad hoc polymorphism mean?
42. What is operator overloading?
43. What is a pre-increment operator?
44. What is a post-decrement operator?
45. How do you overload the assignment operator?
46. How do you invoke a member function on a global or stack allocated object?
47. How do you invoke a member function on a heap allocated object?
48. What is the 'this' pointer and what does it point to?
49. What is the type of a this pointer?
50. Does every object have a this pointer?
51. What does 'const' mean?
52. What does pass-by-value mean?
53. What does pass-by-reference mean?
54. Does C++ support pass-by-value, pass-by-reference, or both?
55. How do you specify a reference argument to a function?
56. What is a friend function and how is one defined?
57. How do you define static data in a class?
58. How do you initialize static class variables?
59. What is an inline member function?
60. What is operator overloading?
61. Which C++ operators can be overloaded?
62. What does parametric polymorphism mean?
63. What is a template function?
64. How do you define a template function?
65. Can a template function be defined inline?
66. What is a template for a class?
67. How do you define a template for a class?
68. How do you instantiate an object from a template class?
69. What is the benefit of inheritance?
70. What does class inheritance mean?
71. What does subtype polymorphism mean?
72. What is the difference between an is-a and has-a relationship?
73. How do you express inheritance in C++?
74. Can you achieve reuse in C++ without using inheritance?
75. What is an abstract base class?
76. What is a concrete derived class?
77. What does public inheritance mean?
78. What does the protected keyword mean?
79. Why can't a derived class access the private members of its base class?
80. Can you convert a derived class pointer into a pointer to its public base class?
81. What is a virtual member function?
82. What is a pure virtual function?
83. When should you define a member function as virtual?
84. How do you override a virtual member function?
85. How does C++ perform static typing while supporting dynamic binding?
86. Can destructors be virtual?
87. What is the purpose of a virtual destructor?
88. Can a constructor be virtual?
89. Can you inherit a friend function or an assignment operator?