Список вопросов

1. Что такое парадигма программирования?
2. Что такое идиома программирования?
3. Какую парадигму реализует язык C?
4. Какую парадигму реализует язык C++?
5. Язык C++ считается низкоуровневым или высокоуровневым?
6. Что такое ООП?
7. Что подразумевает абстракция с точки зрения ООП?
8. Что такое инкапсуляция?
9. Что такое наследование?
10. Что такое полиморфизм?
11. Какие существуют виды полиморфизма?
12. В чем отличие компилируемых и интерпретируемых языков программирования?
13. Что такое статическая и динамическая типизация?
14. Что такое слабая и сильная типизация?
15. Для чего и на какие файлы производится разбиение программы на C++?
16. Что такое union в C++, когда оно может быть применимо?
17. Опишите процесс преобразования исходного кода в исполняемый файл.
18. В чем отличие ссылки от указателя?
19. Что такое указатель на функцию и как он может быть использован?
20. Какие способы группировки данных в C++ вам известны?
21. Для чего предназначены структуры?
22. Где может быть определена структура или класс?
23. Допустимо ли использование указателей/ссылок/массивов структур?
24. Какие существуют способы передачи параметров в функцию?
25. Для чего предназначены классы, в чем их отличие от структур?
26. Что такое инвариант класса?
27. В чем отличие функций от методов?
28. В каких случаях используются значения по умолчанию в функциях?
29. Что такое публичный интерфейс?
30. Какие существуют модификаторы доступа, для чего они используются?
31. Что такое геттеры и сеттеры?
32. Что такое inline-функции?
33. Где применяется неявный указатель this?
34. Для чего используется ключевое слово const?
35. Что такое константные ссылки/указатели, указатели/ссылки на константу?
36. В чем отличие синтаксической и логической константности методов?
37. Для чего используется ключевое слово mutable?
38. Что такое конструктор?
39. В каких случаях используется перегрузка конструкторов?
40. Какую цель может преследовать создание приватного конструктора?
41. Каким образом и в какой последовательности происходит инициализация полей объекта?
42. Для чего используется ключевое слово explicit?
43. В чем заключается предназначение конструктора по умолчанию?
44. Что такое деструктор, для чего он используется?
45. Каков порядок вызова деструкторов при разрушении объекта?
46. В какой момент вызывается деструктор объекта?
47. Каково время жизни объекта?

48. Зачем нужен виртуальный деструктор?
49. Как осуществляется работа с динамической памятью в C/C++?
50. В чем различие delete и delete[]?
51. Что подразумевается под идиомой RAII?
52. Перечислите основные подходы к обработке ошибок.
53. Для чего предназначен механизм обработки исключительных ситуаций?
54. Что такое исключение?
55. Какие типы данных допустимы для использования в качестве объектов exception?
56. Как происходит возбуждение исключения?
57. Кто отвечает за обработку возникших исключительных ситуаций?
58. Что такое раскрутка стека?
59. Где и для чего используется спецификатор throw?
60. Где и для чего используется спецификатор noexcept?
61. К чему приводит вызов throw без аргументов?
62. Что такое exception-safe операция?
63. Что такое делегирующие конструкторы?
64. Что вы можете сказать о генерации исключений в конструкторе/деструкторе?
65. Что такое ассоциация?
66. Что такое композиция и агрегация, чем они отличаются?
67. Время жизни агрегируемого объекта меньше времени жизни агрегата?
68. Какие классы называются дружественными, для каких целей используется это отношение?
69. В каком случае можно говорить об отношение «реализация»?
70. Как представлены объекты в памяти при использовании механизма наследования?
71. Какие существуют типы наследования, чем они различаются?
72. Наследуются ли конструкторы и деструкторы?
73. Наследуются ли приватные поля базового класса?
74. Что такое виртуальная функция?
75. Как осуществить вызов базовой реализации функции при её переопределении в дочернем классе?
76. Как связаны виртуальные функции и полиморфизм?
77. Что такое переопределение функций?
78. Работает ли переопределение для приватных функций?
79. Что такое таблица виртуальных функций?
80. Как себя ведут виртуальные функции в конструкторе и деструкторе?
81. В каких случаях допустимо приведение указателей/ссылок на дочерний класс к базовому?
82. Что такое чистая виртуальная функция?
83. Какой класс называется абстрактным?
84. Как в C++ реализуются интерфейсы?
85. Что такое перегрузка функций?
86. Как ведет себя перегрузка при наследовании?
87. Опишите процесс выбора функции среди перегруженных.
88. Чем отличаются механизмы раннего и позднего связывания?
89. Что такое множественное наследование?
90. Что такое ромбовидное наследование?
91. Какой существует механизм разрешения проблемы ромбовидного наследования в C++?
92. Как реализовано приведение типов в Си?
93. Что такое статическое приведение типов?
94. Что такое динамическое приведение типов?

95. Что такое константное приведение типов?
96. Что такое интерпретирующее преобразование типов?
97. Как работает преобразование в Си-стиле на языке C++?
98. Что такое умные указатели?
99. Опишите принцип работы boost::scoped_ptr.
100. Опишите принцип работы std::auto_ptr.
101. Опишите принцип работы std::shared_ptr.
102. Опишите принцип работы std::weak_ptr.
103. В чем особенности работы умных указателей с массивами?
104. Какие группы операторов в C++ вам известны?
105. Что такое перегрузка операторов, для чего она используется?
106. Для каких типов допустима перегрузка операторов?
107. Где может быть объявлена перегрузка оператора?
108. Какие особенности у перегрузки операторов инкремента и декремента?
109. Как ведут себя операторы с особым порядком вычисления при перегрузке?
110. Наследует ли производный класс перегруженные операторы?
111. Как защитить объект от копирования?
112. Для чего предназначен механизм RTTI, как его использовать?
113. Что такое шаблоны классов?
114. Что такое шаблоны функций?
115. Как осуществляется вывод аргументов шаблона?
116. Что такое специализация шаблонов?
117. Что такое шаблон проектирования?

+ Вопросы по каждому из 23 шаблонов проектирования: название, область применения, решаемая задача, `uml`-диаграмма (можно не точную), достоинства и недостатки.